electronics

Special Issue Reprint

Generative AI and Its Transformative Potential

Edited by
Galina Ilieva and George A. Tsihrintzis

mdpi.com/journal/electronics

MDPI

Generative AI and Its Transformative Potential

Generative AI and Its Transformative Potential

Guest Editors

Galina Ilieva
George A. Tsihrintzis

MDPI

Basel • Beijing • Wuhan • Barcelona • Belgrade • Novi Sad • Cluj • Manchester

Guest Editors

Galina Ilieva
Department of Management and Quantitative Methods in Economics
University of Plovdiv Paisii Hilendarski
Plovdiv
Bulgaria

George A. Tsihrintzis
Department of Informatics
University of Piraeus
Piraeus
Greece

Editorial Office
MDPI AG
Grosspeteranlage 5
4052 Basel, Switzerland

This is a reprint of the Special Issue, published open access by the journal *Electronics* (ISSN 2079-9292), freely accessible at: https://www.mdpi.com/journal/electronics/special_issues/Generative_AI.

For citation purposes, cite each article independently as indicated on the article page online and as indicated below:

Lastname, A.A.; Lastname, B.B. Article Title. *Journal Name* **Year**, *Volume Number*, Page Range.

ISBN 978-3-7258-4185-1 (Hbk)
ISBN 978-3-7258-4186-8 (PDF)
https://doi.org/10.3390/books978-3-7258-4186-8

Cover image courtesy of George A. Tsihrintzis

Contents

About the Editors

Galina Ilieva

Galina Ilieva graduated with honors in Computer Engineering from the Technical University of Sofia, Bulgaria, and later obtained her Ph.D. from Plovdiv University "Paisii Hilendarski," specializing in agent-based negotiation in e-commerce. Since 2008, she has lectured at the university's Faculty of Economics and Social Sciences. After completing her habilitation in 2011, she was appointed Professor of Business Informatics in the Department of Management and Quantitative Methods in Economics. She is currently serving a four-year term (2023–2027) on the university's Supervisory Board. A senior member of both the IEEE and the ACM, she has co-authored around 100 scientific publications and has an h-index of 13 (Scopus). Her research now focuses on intelligent decision-making systems and machine learning.

George A. Tsihrintzis

George A. Tsihrintzis received a Diploma of Electrical Engineering from the National Technical University of Athens, Greece (with honors) and M.Sc. and Ph.D. degrees in Electrical and Computer Engineering from Northeastern University, USA. He is currently a Professor and a Member of the Administration Board of the University of Piraeus, having served as Head of its Department of Informatics from 2016 to 2020. His current research interests include artificial intelligence, machine learning, pattern recognition, and decision theory and their applications in Internet-of-Things technologies, multimedia interactive services, user modeling, knowledge-based software systems, human–computer interaction, and information retrieval. He has authored/co-authored over 400 research publications in these areas that have appeared in international journals, book chapters, and conference proceedings. He has also co-authored/co-edited over 50 books. He has served as the principal investigator/co-investigator of several R&D projects. Since 2018, he has been the Editor-in-Chief of the Intelligent Decision Technologies Journal (IOS Press). From 2012 to 2022, he was the Editor-in-Chief of the International Journal of Computational Intelligence Studies (Inderscience). He has been the Editor-in-Chief of the Engineering Section of SpringerPlus. He is the founder and Editor-in-Chief of the Learning and Analytics in Intelligent Systems book series (Springer 2019) and the Artificial Intelligence-Enhanced Software and Systems Engineering book series (Springer 2022). He has organized and chaired 42 international conferences. He has received best paper awards, and been a keynote speaker of several international conferences. He has supervised 15 completed doctoral theses. He is ranked among the top 2% of the most influential scientists worldwide of Scientists List at Stanford University ranking in ARTIFICIAL INTELLIGENCE, both career-wise and single year, for 2023 and for 2024.

electronics

MDPI

Editorial

Editorial Note to Special Issue "Generative AI and Its Transformative Potential"

Galina Ilieva [1,*] and George A. Tsihrintzis [2,*]

[1] Department of Management and Quantitative Methods in Economics, University of Plovdiv Paisii Hilendarski, 4000 Plovdiv, Bulgaria
[2] Department of Informatics, University of Piraeus, 18534 Piraeus, Greece
* Correspondence: galili@uni-plovdiv.bg (G.I.); geoatsi@unipi.gr (G.A.T.)

In recent years, generative artificial intelligence (AI) has emerged as a powerful paradigm capable of transforming both scientific research and business applications. Its rapid development—driven by advances in machine learning and deep learning—has enabled the creation of models that not only automate repetitive tasks but also generate new content and augment human creativity.

This Special Issue of *Electronics*, titled "Generative AI and Its Transformative Potential", explores the role of generative AI across various scientific domains and industry sectors. Generative AI offers a wide range of opportunities for innovation and problem-solving—from simplifying product design processes in manufacturing to generating synthetic data for training intelligent systems in areas such as quality control, autonomous systems, and healthcare.

This Issue features contributions from leading researchers and practitioners working on the theoretical, technological, and applied aspects of generative AI. It emphasizes interdisciplinary perspectives and real-world implementations, addressing both the opportunities and challenges posed by this rapidly evolving technology.

Of the numerous submissions received, fourteen high-quality papers were selected for inclusion following a rigorous peer-review process. These contributions span a variety of application areas, showcasing the diverse potential of generative AI. For clarity and coherence, the selected papers are organized into the following three thematic clusters: (1) foundational models and frameworks for generative AI; (2) applications in education, language, and human-centered systems; and (3) creative and industrial use cases.

The first cluster includes papers that explore the architectures, development tools, and evaluation techniques associated with generative AI systems. Topics include the application of large language models, transformer-based architectures, and hybrid AI techniques for knowledge extraction and content generation. The following five papers are included:

1. "AI-Assisted Programming Tasks Using Code Embeddings and Transformers" explores how code embeddings and transformer architectures can support programmers, automate coding tasks, and enhance software development through AI-generated assistance.
2. "Augmenting Large Language Models with Rules for Enhanced Domain-Specific Interactions: The Case of Medical Diagnosis" proposes a hybrid framework that enriches large language models with rule-based knowledge for medical diagnosis, thereby improving precision and contextual relevance in clinical decision support.
3. "The Genesis of AI by AI Integrated Circuit: Where AI Creates AI" presents a visionary concept of AI-generated AI circuits, discussing conceptual frameworks and potential hardware–software synergies to support autonomous AI design.

4. "Extension of Interval-Valued Hesitant Fermatean Fuzzy TOPSIS for Evaluating and Benchmarking of Generative AI Chatbots" introduces a novel fuzzy multi-criteria decision-making method to evaluate generative AI chatbots, enabling objective benchmarking based on user preferences and system capabilities.
5. "Understanding Factors Influencing Generative AI Use Intention: A Bayesian Network-Based Probabilistic Structural Equation Model Approach" employs a probabilistic SEM using Bayesian networks to identify key factors influencing users' intentions to adopt generative AI tools, integrating behavioural science with data-driven modelling.

The second group focuses on the use of generative AI in personalized education, language learning, and emotion-aware systems. These studies illustrate how generative models can enhance engagement, tailor content to learner profiles, and facilitate human–machine interaction. The five papers included in this cluster are as follows:

6. "Pre-Service Teachers' Assessment of ChatGPT's Utility in Higher Education: SWOT and Content Analysis" investigates pre-service teachers' perceptions of ChatGPT in higher education and instructional settings using a combination of SWOT analysis and qualitative methods.
7. "Framework for Integrating Generative AI in Developing Competencies for Accounting and Audit Professionals" introduces a framework that leverages generative AI to support skills development in the accounting and auditing professions, focusing primarily on personalized learning and scenario-based training.
8. "A Survey on Challenges and Advances in Natural Language Processing with a Focus on Legal Informatics and Low-Resource Languages" reviews recent advancements and ongoing challenges in NLP, with particular attention given to legal informatics and low-resource languages, where generative models face unique linguistic and contextual constraints.
9. "Plato's Shadows in the Digital Cave: Controlling Cultural Bias in Generative AI" draws philosophical parallels to examine the emergence of cultural bias in generative AI outputs and propose strategies to mitigate such bias during training and deployment.
10. "Web Application for Retrieval-Augmented Generation: Implementation and Testing" presents a retrieval-augmented generation (RAG) system that integrates search and generation to improve factual consistency and traceability in AI-generated content.

The final cluster presents practical implementations of generative AI across domains such as visual arts, smart manufacturing, media content generation, and social simulations. The following four papers demonstrate how AI can co-create content and improve decision-making in complex environments:

11. "Generative Adversarial Network Models for Augmenting Digit and Character Datasets Embedded in Standard Markings on Ship Bodies" applies GANs to augment datasets of digits and characters from standardized ship markings, supporting enhanced recognition and automation in maritime inspection systems.
12. "Generative AI-Driven Data Augmentation for Crack Detection in Physical Structures" demonstrates the use of generative models to synthesize data for training crack detection algorithms, advancing structural health monitoring in civil engineering.
13. "Illumination and Shadows in Head Rotation: Experiments with Denoising Diffusion Models" presents experiments using diffusion models to reconstruct and analyse head rotation under varying lighting conditions, highlighting generative AI's strength in complex visual tasks.
14. "Novel Learning Framework with Generative AI X-ray Images for Deep Neural Network-Based X-Ray Security Inspection of Prohibited Items" proposes a new train-

ing framework that utilises synthetic X-ray images generated by AI to improve prohibited-item detection in security systems, using YOLO-based architectures.

We hope this Special Issue will serve as a valuable reference for researchers, developers, and decision-makers seeking to leverage generative AI in their respective fields. We also hope that these contributions will inspire further research, foster interdisciplinary collaboration, and promote the responsible deployment of generative AI technologies for the benefit of society.

As generative AI continues to evolve, future editions of *Electronics* will undoubtedly revisit this topic, offering expanded scopes and new perspectives.

Author Contributions: Writing—Original Draft Preparation, G.I. and G.A.T.; Writing—Review and Editing, G.I. and G.A.T. All authors have read and agreed to the published version of the manuscript.

Conflicts of Interest: The authors declare no conflicts of interest.

List of Contributions:

1. Kotsiantis, S.; Verykios, V.; Tzagarakis, M. AI-Assisted Programming Tasks Using Code Embeddings and Transformers. *Electronics* **2024**, *13*, 767. https://doi.org/10.3390/electronics13040767.
2. Panagoulias, D.; Virvou, M.; Tsihrintzis, G. Augmenting Large Language Models with Rules for Enhanced Domain-Specific Interactions: The Case of Medical Diagnosis. *Electronics* **2024**, *13*, 320. https://doi.org/10.3390/electronics13020320.
3. Baungarten-Leon, E.; Ortega-Cisneros, S.; Abdelmoneum, M.; Vidana Morales, R.; Pinedo-Diaz, G. The Genesis of AI by AI Integrated Circuit: Where AI Creates AI. *Electronics* **2024**, *13*, 1704. https://doi.org/10.3390/electronics13091704.
4. Ilieva, G. Extension of Interval-Valued Hesitant Fermatean Fuzzy TOPSIS for Evaluating and Benchmarking of Generative AI Chatbots. *Electronics* **2025**, *14*, 555. https://doi.org/10.3390/electronics14030555.
5. Kim, C. Understanding Factors Influencing Generative AI Use Intention: A Bayesian Network-Based Probabilistic Structural Equation Model Approach. *Electronics* **2025**, *14*, 530. https://doi.org/10.3390/electronics14030530.
6. Markos, A.; Prentzas, J.; Sidiropoulou, M. Pre-Service Teachers' Assessment of ChatGPT's Utility in Higher Education: SWOT and Content Analysis. *Electronics* **2024**, *13*, 1985. https://doi.org/10.3390/electronics13101985.
7. Anica-Popa, I.; Vrîncianu, M.; Anica-Popa, L.; Cișmașu, I.; Tudor, C. Framework for Integrating Generative AI in Developing Competencies for Accounting and Audit Professionals. *Electronics* **2024**, *13*, 2621. https://doi.org/10.3390/electronics13132621.
8. Krasadakis, P.; Sakkopoulos, E.; Verykios, V. A Survey on Challenges and Advances in Natural Language Processing with a Focus on Legal Informatics and Low-Resource Languages. *Electronics* **2024**, *13*, 648. https://doi.org/10.3390/electronics13030648.
9. Karpouzis, K. Plato's Shadows in the Digital Cave: Controlling Cultural Bias in Generative AI. *Electronics* **2024**, *13*, 1457. https://doi.org/10.3390/electronics13081457.
10. Radeva, I.; Popchev, I.; Doukovska, L.; Dimitrova, M. Web Application for Retrieval-Augmented Generation: Implementation and Testing. *Electronics* **2024**, *13*, 1361. https://doi.org/10.3390/electronics13071361.
11. Abdulraheem, A.; Suleiman, J.; Jung, I. Generative Adversarial Network Models for Augmenting Digit and Character Datasets Embedded in Standard Markings on Ship Bodies. *Electronics* **2023**, *12*, 3668. https://doi.org/10.3390/electronics12173668.
12. Kim, J.; Seon, J.; Kim, S.; Sun, Y.; Lee, S.; Kim, J.; Hwang, B.; Kim, J. Generative AI-Driven Data Augmentation for Crack Detection in Physical Structures. *Electronics* **2024**, *13*, 3905. https://doi.org/10.3390/electronics13193905.

13. Asperti, A.; Colasuonno, G.; Guerra, A. Illumination and Shadows in Head Rotation: Experiments with Denoising Diffusion Models. *Electronics* **2024**, *13*, 3091. https://doi.org/10.3390/electronics13153091.
14. Kim, D.; Kang, J. Novel Learning Framework with Generative AI X-Ray Images for Deep Neural Network-Based X-Ray Security Inspection of Prohibited Items Detection with You Only Look Once. *Electronics* **2025**, *14*, 1351. https://doi.org/10.3390/electronics14071351.

electronics

MDPI

Review

AI-Assisted Programming Tasks Using Code Embeddings and Transformers

Sotiris Kotsiantis [1,*], Vassilios Verykios [2] and Manolis Tzagarakis [3]

[1] Department of Mathematics, University of Patras, 265 04 Patras, Greece
[2] School of Science and Technology, Hellenic Open University, 263 35 Patras, Greece; verykios@eap.gr
[3] Department of Economics, University of Patras, 265 04 Patras, Greece; tzagara@upatras.gr
* Correspondence: sotos@math.upatras.gr

Abstract: This review article provides an in-depth analysis of the growing field of AI-assisted programming tasks, specifically focusing on the use of code embeddings and transformers. With the increasing complexity and scale of software development, traditional programming methods are becoming more time-consuming and error-prone. As a result, researchers have turned to the application of artificial intelligence to assist with various programming tasks, including code completion, bug detection, and code summarization. The utilization of artificial intelligence for programming tasks has garnered significant attention in recent times, with numerous approaches adopting code embeddings or transformer technologies as their foundation. While these technologies are popular in this field today, a rigorous discussion, analysis, and comparison of their abilities to cover AI-assisted programming tasks is still lacking. This article discusses the role of code embeddings and transformers in enhancing the performance of AI-assisted programming tasks, highlighting their capabilities, limitations, and future potential in an attempt to outline a future roadmap for these specific technologies.

Keywords: AI-assisted programming; code embeddings; transformers

1. Introduction

AI-assisted programming or development is defined as the utilization of machine learning models trained on the vast amount of available source code. Its purpose is to support various aspects of programming and, more broadly, software engineering implementation tasks. According to the software naturalness conjecture [1], which posits that source code, like natural language, is often repetitive and predictable, this technology has become integrated into popular integrated development environments (IDEs) and gained widespread popularity among developers [2]. Noteworthy applications such as IntelliCode [3], Github Copilot [4,5], Codex [6], and DeepMind AlphaCode [7] exemplify AI-assisted programming tools accessible to the public.

The impact of AI on software development tasks is expected to enhance precision, speed, and efficiency [8]. These benefits extend beyond professional programmers to include novice programmers [9], with ongoing studies exploring the potential of AI in various fields. Research reports highlight the sensitivity of these tools to the specific tasks they support.

Despite generating code, AI-assisted programming tools may produce complex and error-prone code [10]. In the domains of data science and data analysis, these tools contribute positively to addressing challenging problems [11,12]. For novice programmers in educational settings, AI-assisted programming tools increase project completion rates and grades [13,14]. Nevertheless, novices may encounter difficulties in comprehending and utilizing these tools proficiently [15].

Code embeddings and transformers represent popular approaches to AI-assisted programming, significantly impacting software engineering by improving task performance

and efficiency. These techniques reduce manual effort in coding, debugging, and maintenance, thereby decreasing overall development time and costs. Furthermore, they enable cross-language development, allowing seamless work with multiple programming languages.

Code embedding [16] is a machine learning technique representing code as dense vectors in a continuous vector space. Unlike traditional methods that treat code as sequences, code embedding captures semantic relationships between code snippets by training a neural network to learn fixed-size vector representations. These embeddings find application in various software engineering tasks, such as code completion, correction, summarization, and search [17].

Researchers like Azcona et al. [18] propose using embeddings to profile individual Computer Science students, analyzing Python source code submissions to predict code correctness. Similarly, Ding et al. [19] introduce GraphCodeVec, employing graph convolutional networks to generate generalizable and task-agnostic code embeddings, demonstrating superior performance in multiple tasks.

Transformers [20], a type of neural network utilizing attention mechanisms for sequential data processing, stand out from traditional recurrent neural networks. Transformers can handle parallel input and self-attention mechanisms, processing a sequence of tokens by attending to all input tokens simultaneously. In contrast, code embeddings use traditional deep learning models like recurrent neural networks (RNNs) and convolutional neural networks (CNNs). Transformers, trained through pre-training and fine-tuning, can handle variable input lengths, whereas code embeddings require fixed input lengths.

For code-related tasks, input code snippets feed into the transformer model, which employs self-attention mechanisms to capture contextual relationships within the code. The transformed representations generated using the model find application in downstream software engineering tasks, such as code generation, summarization, translation, or identifying patterns and anomalies in code. Chirkova and Troshin [21] demonstrated improved performance with syntax-capture modifications in transformer models.

In Figure 1, the timeline showcases the evolution of AI-assisted programming tasks utilizing code embeddings and transformers from their early stages of experimentation to their integration as indispensable tools in modern software development workflows.

Figure 1. Timeline of the main contributions of AI-assisted programming tasks.

In summary, this paper details code embeddings and transformers, discussing their characteristics. It explores existing AI-supported programming approaches, contextualizing their support for various tasks. The paper concludes with insights and acknowledged limitations.

2. Code Embeddings and Transformers

Code embeddings, also referred to as source code embeddings or program embeddings, have garnered significant attention in the realms of natural language processing (NLP) and code generation. These techniques, categorized as representation learning, aim to encode both syntactic and semantic information from source code into a lower-dimensional vector space. While code embeddings have demonstrated promising results in various NLP tasks such as code similarity, bug detection, and code completion, recent advancements in transformer models have led to a shift in focus towards using transformers for code representation and generation tasks. This section delves into the concept of code embeddings and their relationship with transformers.

In contrast to conventional approaches relying on static program analysis techniques, code embeddings offer more effective ways to represent and analyze source code [16]. These embeddings employ neural networks to create a semantic representation of code sequences, learning from a substantial code corpus. The specific architectures and training techniques employed enable these networks to capture inherent patterns and relationships between code elements. Consequently, code embeddings encode both structural and lexical characteristics of source code, presenting a comprehensive representation applicable to diverse downstream tasks [19].

The process of creating code embeddings involves several steps. Initially, the source code undergoes tokenization, breaking it down into a sequence of tokens, which can be characters, words, or syntactic constructs of the programming language. Subsequently, this token sequence is fed into a neural network, and trained to generate a vector representation for each token. This process is repeated for all code sequences in the training data, and the resulting vectors are stored in an embedding matrix. The embedding matrix is considered a form of "learned parameters" in the neural network architecture. In detail, at the beginning of training, the embedding matrix is initialized with random values or pre-trained word embeddings. During the forward pass of the neural network, the input tokens (words) are represented as one-hot vectors or integer indices that correspond to their positions in the vocabulary. These indices are then used to index into the embedding matrix, retrieving the dense vector representations (embeddings) of the input tokens. During training, the values of the embedding matrix are adjusted via backpropagation and gradient descent. The objective is to learn meaningful representations of words that capture semantic relationships and contextual information from the training data. This process involves updating the parameters of the embedding matrix to minimize the loss function of the neural network.

One key technique employed in training code embeddings is the use of skip-gram models [22]. Initially developed for natural language processing tasks, skip-gram models are adapted for code embeddings to learn semantic relationships between code tokens based on their contextual surroundings. This enables the model to capture both syntactic and semantic aspects, yielding a more holistic representation.

In the realm of program synthesis, code embeddings find application in generating code from natural language descriptions [23]. This entails training the model to comprehend the relationship between natural language descriptions and code sequences, facilitating the generation of code aligned with the provided description. This application extends to automatic code documentation and the development of programming tools for non-technical users.

To summarize, the mathematical representation of code embeddings involves the following:

- Tokenizing the code snippet S to obtain a sequence of tokens (t_1,t_2,. . .,t_n).
- Obtaining the embedding E(t_i) for each token t_i.

- Combining the token embeddings to obtain the code embedding C, for example, by averaging.

Rabin et al. [17] evaluated the use of code2vec embeddings compared to handcrafted features for machine learning tasks, finding that code2vec embeddings offered even information gains distribution and exhibited resilience to dimension removal compared to handcrafted feature vectors.

Sikka et al. [24] introduced a machine learning problem related to estimating the time complexity of programming code. Comparing feature engineering and code embeddings, both methods performed well, showcasing their applicability in estimating time complexity across various applications.

While code embeddings demonstrate significant potential in diverse applications, challenges and limitations exist. Incorporating semantic knowledge into embeddings remains challenging, as models may struggle to interpret contextual or domain-specific information that human programmers easily grasp. Kang et al. [25] investigated the potential benefits of embeddings in downstream tasks for source code models but found no tangible improvement in existing models, calling for further research in this direction.

Romanov and Ivanov [26] experimentally explored the use of pre-trained graph neural networks for type prediction, revealing that pre-training did not enhance type prediction performance. Ding et al. [27] studied the generalizability of pre-trained code embeddings for various software engineering tasks, introducing StrucTexVec, a two-stage unsupervised training framework. Their experiments demonstrated that pre-trained code embeddings, incorporating structural context, could be advantageous in most software engineering tasks.

Another challenge lies in the requirement for substantial amounts of training data. Code embeddings rely on a large corpus of code for training, limiting their applicability in specific programming languages or domains with smaller codebases.

As previously mentioned, NLP transformers, utilizing attention mechanisms, have become dominant in handling sequential data. Unlike traditional models such as recurrent neural networks (RNNs) and long short-term memory (LSTM), transformers, particularly in the form of bidirectional encoder representations from transformers (BERT), have demonstrated superior performance in various NLP tasks, including those related to code.

The self-attention mechanism in transformers can be mathematically described as the following: Attention(Q,K,V) = softmax$\left(\frac{QK^T}{\sqrt{d_k}}\right)$, where Q represents the query matrix, K represents the key matrix, V represents the value matrix, and d_k represents the dimension of the key vectors.

NLP transformers leverage self-attention mechanisms [28] to process words in a sentence. This involves the model learning to focus on other words in the sentence for each word, assigning weights based on significance. In code-related tasks, the transformer's input comprises a sequence of tokens representing parts of the code (e.g., function names, variable names). These tokens traverse the transformer model, enabling it to learn relationships between different parts of the code. The model then predicts the next token based on the acquired relationships.

A notable advantage of NLP transformers in code-related tasks is their proficiency in handling long sequences [29]. Unlike traditional models like RNNs and LSTMs, transformers circumvent the vanishing gradient problem when processing lengthy sequences, leading to significantly improved performance.

Transformers introduce positional encoding [28], conveying information about the position of words in a sentence. This proves beneficial in code-related tasks where the order of code is crucial for functionality. Positional encoding aids the model in distinguishing between words with similar meanings but from different parts of the code, enhancing overall performance. In Figure 1, we present a timeline of the main contributions of AI-assisted programming tasks.

NLP transformers find successful applications in various code-related tasks. In code completion [30], the model predicts the next tokens of code given a partial snippet. Code summarization [31] involves generating a concise summary of a piece of code, aiding comprehension of large and complex codebases. Code translation [32] sees the model translating code from one programming language to another, particularly useful for dealing with legacy code.

One prominent NLP transformer model is bidirectional encoder representations from transformers (BERT) [33], widely applied in code-related tasks due to its success in natural language tasks. Another transformer model, gated transformer [34], addresses the limitations of the original transformer, enhancing efficiency on long sequences with repetitive elements.

The transformer architecture consists of encoder and decoder components. The encoder processes input text, converting it into a sequence of vectors, while the decoder generates output text based on these vectors. The encoder comprises multiple identical layers, each featuring self-attention and feed-forward network sub-layers. Input to the encoder passes through an embedding layer, converting the input sequence into fixed-dimensional embeddings. Self-attention within the encoder captures relevant information in the input sequence, utilizing multiple heads or parallel attention mechanisms to attend to different parts of the sequence. These mechanisms compute weighted sums based on word importance, capturing long-term dependencies.

The self-attention and feed-forward network layers repeat within the encoder, allowing hierarchical processing of the input sequence. This hierarchical approach captures different levels of abstraction [33], resulting in the hidden representation of the input sequence for further processing by the decoder.

An advantage of transformers lies in the encoder's ability to process input sequences of variable lengths, offering versatility for various NLP tasks. The use of multiple heads [35] in self-attention mechanisms allows transformers to learn diverse representations of the input sequence, enhancing encoder robustness.

In simple terms, multi-head attention empowers the transformer model to attend to multiple pieces of information simultaneously. Instead of relying on a single attention mechanism, multi-head attention deploys several attention mechanisms in parallel, creating multiple representations of the input sequence. These parallel mechanisms, or "heads", perform the same operation with different sets of parameters, enabling the model to attend to different aspects of the input sequence. Context vectors generated by each head are concatenated, resulting in the final representation of the input sequence.

Pre-trained models in transformers are large neural network architectures pre-trained on extensive text data. These models learn statistical patterns and language structures, allowing them to understand and generate human-like text. Unlike traditional language models, pre-trained transformers use bidirectional attention to consider both previous and future words, providing a better understanding of the overall context.

The effectiveness of transformer models in software engineering tasks relies on domain-specific data availability and the relevance of pre-training data to the target domain. Experimentation and adaptation are crucial for optimal results in diverse software engineering applications [36].

The general process in software engineering tasks using transformers can be outlined in the following steps:

- Data preprocessing: The initial step involves preprocessing the input data, typically through tokenization and vectorization of code snippets. This step is crucial to feed meaningful data into the transformer model.
- Transformer architecture: The transformer model comprises an encoder and a decoder. The encoder processes input data to create a code representation, and the decoder utilizes this representation to generate the code.
- Attention mechanism: Transformers incorporate an attention mechanism, a pivotal element allowing the model to focus on specific parts of the input data while generat-

ing the output. This enhances efficiency in handling long sequences and capturing complex dependencies.

- Training the model: Following data preprocessing and setting up the transformer model, the next step involves training the model using backpropagation. Batches of data pass through the model, loss is calculated, and model parameters are updated to minimize the loss.
- Fine-tuning: It is essential to assess its quality and make any necessary adjustments to the model. Fine-tuning may involve retraining on a labeled dataset or adjusting hyperparameters.

CodeBERT [37], a transformer model pre-trained on a comprehensive dataset of source code and natural language, excels in understanding the relationship between code and corresponding comments. It demonstrates state-of-the-art performance in code completion, summarization, and translation tasks, generating accurate and human-like code. CodeBERT follows the underlying architecture of BERT with modifications to suit the programming language domain [38]. The bidirectional transformer encoder takes in code and natural language sequences, encoding them into contextualized representations. A decoder then generates a human-readable description of the code. Code and natural language sequences are concatenated with special tokens to indicate the input type. Pre-trained on extensive data from GitHub, Stack Overflow, Wikipedia, and other sources, CodeBERT undergoes fine-tuning for downstream tasks like code summarization, classification, and retrieval. This transfer learning model adapts to different codebases and programming languages, facilitating code generation and retrieval for non-programmers.

T5 (text-to-text transfer transformer) [39], another large-scale transformer model, caters to various natural language tasks. Pre-trained on diverse datasets, T5 can handle tasks such as translation, summarization, and question answering. It has proven effective in code generation tasks, producing high-quality code with detailed explanations.

GPT-3 (generative pre-trained transformer) [40], developed by OpenAI, excels in natural language generation tasks, including code completion. Its large size and pre-training on a wide range of tasks make it adept at generating code for different programming languages, often matching human writing.

XLNet [41], based on the permutation language model, outperforms BERT in many NLP tasks, including code completion. Similar to BERT, XLNet comprehends code syntax and context well, generating code for various programming languages. CCBERT [42], a deep learning model for generating Stack Overflow question titles, exhibits strong performance in regular and low-resource datasets.

EL-CodeBert [43], a pre-trained model combining programming languages and natural languages, utilizes representational information from each layer of CodeBert for downstream source code-related tasks. Outperforming state-of-the-art baselines in four tasks, EL-CodeBert demonstrates effectiveness in leveraging both programming and natural language information.

Transformers have demonstrated impressive performance across various natural language processing (NLP) tasks and have found successful applications in AI-assisted programming. However, they also exhibit certain inherent weaknesses within this domain. One limitation lies in their capacity for contextual understanding. While transformers excel at capturing context within a fixed-length window, typically around 512 tokens, programming tasks often involve extensive codebases where understanding context beyond this window becomes crucial for accurate analysis and generation. Additionally, transformers lack domain-specific knowledge. Being pre-trained on general-purpose corpora, they may not adequately capture the intricacies and specialized knowledge required for programming tasks. This deficiency can result in suboptimal performance when dealing with programming languages, libraries, and frameworks.

In AI-assisted programming tasks, code embeddings and transformers are closely connected, often complementing each other to enhance the capabilities of programming

assistance tools. There are connections between code embeddings and transformers in this context:

- Representation learning: Both code embeddings and transformers aim to learn meaningful representations of code. Code embeddings convert source code into fixed-dimensional vectors, capturing syntactic and semantic information. Similarly, transformers utilize self-attention mechanisms to learn contextual representations of code snippets, allowing them to capture dependencies between different parts of the code.
- Semantic understanding: Code embeddings and transformers facilitate semantic understanding of code. Code embeddings map code snippets into vector representations where similar code fragments are closer in the embedding space, aiding tasks like code search, code similarity analysis, and clone detection. Transformers, with their ability to capture contextual information, excel at understanding the semantics of code by considering the relationships between tokens and their context.
- Feature extraction: Both techniques serve as effective feature extractors for downstream tasks in AI-assisted programming. Code embeddings provide compact representations of code that can be fed into traditional machine learning models or neural networks for tasks like code classification, bug detection, or code summarization. Transformers, on the other hand, extract features directly from code snippets using self-attention mechanisms, enabling end-to-end learning for various programming-related tasks.
- Model architecture: Code embeddings and transformers are often integrated into the same model architecture to leverage their complementary strengths. For instance, models like CodeBERT combine transformer-based architectures with code embeddings to enhance code understanding and generation capabilities. This fusion allows the model to capture both local and global dependencies within code snippets, resulting in more accurate and context-aware predictions.
- Fine-Tuning: Pre-trained transformers, such as BERT or GPT, can be fine-tuned on code-related tasks using code embeddings as input features. This fine-tuning process adapts the transformer's parameters to better understand the specific characteristics of programming languages and code structures, leading to improved performance on programming-related tasks.

In conclusion, the use of code embeddings and transformers in software engineering tasks has witnessed substantial growth. Code embeddings, capturing both syntactic and semantic information, offer effective representation learning techniques. Transformers, particularly in the form of BERT and its derivatives, demonstrate superior performance in various code-related tasks, owing to their ability to handle long sequences and consider both past and future context. The pre-trained models, such as CodeBERT and T5, have shown remarkable success in code generation, summarization, and translation tasks. However, challenges such as incorporating semantic knowledge into embeddings and the need for extensive training data persist. Continuous experimentation and adaptation are crucial for harnessing the full potential of these advanced techniques in diverse software engineering applications.

3. Methodology

A comprehensive review of literature pertaining to AI-supported programming tasks was conducted. The selection criteria were based on both content and publication year. Specifically, papers were chosen based on their utilization of code-embeddings or transformer technologies to facilitate AI-assisted programming tasks. The focus was on papers explicitly mentioning specific programming tasks that were supported. The scope of the research encompassed papers published within the last 5 years.

To ensure a thorough examination, only publications indexed in Scopus were taken into consideration. The identification of relevant papers was achieved through a keyword-based search using terms such as "code embeddings" and "transformers", coupled with specific programming tasks (e.g., "code embeddings bug detection").

4. AI-Supported Programming Tasks

In this section, the current body of literature on AI-assisted programming is examined, emphasizing the specific tasks addressed by the studied approaches. The discussion is organized around a framework comprising nine programming tasks identified in the relevant literature. These tasks encompass code summarization, bug detection and correction, code completion, code generation process, code translation, code comment generation, duplicate code detection and similarity, code refinement, and code security.

4.1. Code Summarization

Code summarization involves generating natural language descriptions for source code written in various programming languages, primarily to support documentation generation. During this process, input source code is transformed into a descriptive narrative, typically in English, providing an overview of the code's functionality at the function level.

An enhanced code embedding approach known as Flow2Vec [16] improved the representation of inter-procedural program dependence (value flows) with precision. It accommodated control flows and data flows with alias recognition, mapping them into a low-dimensional vector space. Experiments on 32 open-source projects demonstrated Flow2Vec's effectiveness in enhancing the performance of existing code embedding techniques for code classification and code summarization tasks.

Transformers play a crucial role in generating summaries, involving preprocessing the text by removing unnecessary characters and segmenting them into smaller sentences or phrases. The transformer model, trained on extensive text data, utilizes its attention mechanism to identify key words and phrases, producing a summary based on these essential elements.

Wang et al. [44] introduced Fret, a functional reinforced transformer with BERT, which outperformed existing approaches in both Java and Python. Achieving a BLEU-4 score of 24.32 and a ROUGE-L score of 40.12, Fret demonstrated superior performance in automatic code summarization. For smart contracts, Yang et al. [45] proposed a multi-modal transformer-based code summarization model, showcasing its ability to generate higher-quality code comments compared to state-of-the-art baselines.

Hou et al. [46] presented TreeXFMR, an automatic code summarization paradigm with hierarchical attention, using abstract syntax trees and positional encoding for code representation. Pre-trained and tested on GitHub, TreeXFMR achieved significantly better results than baseline methods.

GypSum [47] incorporated a graph attention network and a pre-trained programming and natural language model for code summarization. Utilizing a dual-copy mechanism, GypSum achieved effective hybrid representations and improved the summary generation process. Gu et al. [48] introduced AdaMo, a method for automated code summarization leveraging adaptive strategies like pre-training and intermediate fine-tuning to optimize latent representations.

Ma et al. [49] proposed a multi-modal fine-grained feature fusion model for code summarization, effectively aligning and fusing information from token and abstract syntax tree modalities. Outperforming current state-of-the-art models, this approach demonstrated superior results.

Gong et al. [31] presented SCRIPT, a structural relative position-guided transformer, using ASTs to capture source code structural dependencies. SCRIPT outperformed existing models on benchmark datasets in terms of BLEU, ROUGE-L, and METEOR metrics. Gao and Lyu [50] proposed M2TS, an AST-based source code summarization technique integrating AST and token features to capture the structure and semantics of source code, demonstrating performance on Java and Python language datasets.

Ferretti and Saletta [51] introduced a novel summarization approach using a pseudo-language to enhance the BRIO model, outperforming CodeBERT and PLBART. The study ex-

plored the limitations of existing NLP-based approaches and suggested further research directions.

Choi et al. [52] presented READSUM, a model combining abstractive and extractive approaches for generating concise and informative code summaries. READSUM considered both structural and temporal aspects of input code, utilizing a multi-head self-attention mechanism to create augmented code representations. The extractive procedure verified the relevancy of important keywords, while the abstractive approach generated high-quality summaries considering both structural and temporal information from the source code.

In summary, code embeddings and transformers both play crucial roles in code summarization, yet they operate in distinct ways. Code embeddings typically involve representing code snippets as fixed-length vectors in a continuous vector space, capturing semantic and syntactic information. This approach offers simplicity and efficiency in handling code representations but may struggle with capturing long-range dependencies. On the other hand, transformers excel in modeling sequential data by processing the entire input sequence simultaneously through self-attention mechanisms. This allows them to capture intricate dependencies across code snippets effectively, resulting in more comprehensive summarizations. However, transformers often require larger computational resources compared to code embeddings. Thus, while code embeddings offer efficiency and simplicity, transformers provide a more powerful and context-aware solution for code summarization tasks.

4.2. Bug Detection and Correction

This task focuses on identifying errors in code (Figure 2), emphasizing the detection of unknown errors to enhance software reliability. Traditional bug detection methods rely on manual code reviews, which are often tedious and time-consuming. In contrast, code embedding presents an efficient approach, capable of processing large volumes of code and identifying potential bugs within minutes. The effectiveness of code embedding depends on a diverse training dataset, as a lack of diversity may hinder its ability to capture all types of bugs.

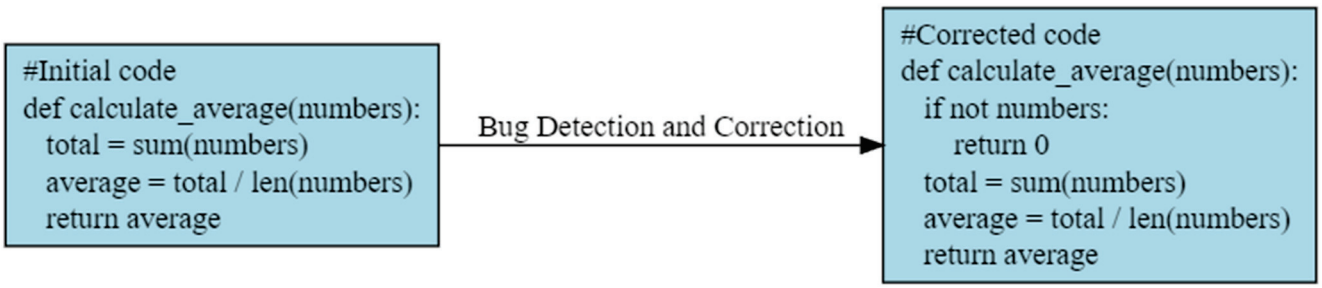


Figure 2. Code bug detection and correction example.

Aladics et al. [53] demonstrated that representing source code as vectors, based on an abstract syntax tree and the Doc2Vec algorithm, improved bug prediction accuracy and was suitable for machine learning tasks involving source code. Cheng et al. [54] proposed a self-supervised contrastive learning approach for static vulnerability detection, leveraging pre-trained path embedding models to reduce the need for labeled data. Their approach outperformed eight baselines for bug detection in real-world projects.

Hegedus and Ferenc [55] used a machine learning model to filter out false positive code analysis warnings from an open-source Java dataset, achieving an accuracy of 91%, an F1-score of 81.3%, and an AUC of 95.3%. NLP transformers offer an efficient and accurate method for bug detection by analyzing source code, identifying patterns, and detecting inconsistencies indicative of bugs. Bagheri and Hegedus [56] compared text representation methods (word2vec, fastText, and BERT) for detecting vulnerabilities in Python code, with BERT exhibiting the highest accuracy rate (93.8%). Gomes et al. [57] found that BERT-based feature extraction significantly outperformed TF-IDF-based extraction in predicting long-

lived bugs, with support vector machines and random forests producing better results when using BERT.

Code summarization, utilizing NLP transformers, presents an approach to bug detection by automatically generating human-readable summaries of code fragments. This method has shown promise in detecting bugs in open-source projects with ample code and bug data available for training.

Evaluation of four new CodeBERT models for predicting software defects demonstrated their ability to improve predictive accuracy across different software versions and projects [58]. The choice of distinct prediction approaches influenced the accuracy of the CodeBERT models.

DistilBERT, a lightweight version of BERT, pre-trained and fine-tuned on various NLP tasks, including bug detection and correction, offers faster and more efficient bug detection, albeit with potentially lower performance than other transformer models. AttSum, a deep attention-based summarization model, surpassed existing models in evaluating bug report titles [59].

Bugsplainer, a transformer-based generative model for explaining software bugs to developers, presented more precise, accurate, concise, and helpful explanations than previous models [60]. Transformers contribute to bug localization, identifying the exact location of bugs in the code. Validation of patches in automated program repair (APR) remains a crucial area, with Csuvik et al. [61] demonstrating the utility of Doc2Vec models in generating patches for JavaScript code.

Mashhadi and Hemmati [62] introduced an automated program repair approach relying on CodeBERT, generating qualitative fixes in various bug cases. Chakraborty et al. [63] created Modit, a multi-modal NMT code editing engine, which outperformed existing models in obtaining correct code patches, especially when developer hints were included.

Generate and validate, a strategy for automatic bug repair using the generative pre-trained transformer (GPT) model, achieved up to 17.25% accuracy [64]. SeqTrans, proposed by Chi et al. [65], demonstrated superior accuracy in addressing certain types of vulnerabilities, outperforming previous strategies in the context of neural machine translation (NMT) technology.

VRepair, an approach by Chen et al. [66], utilized deep learning and transfer learning techniques for automatic software vulnerability repair, showing effectiveness in repairing security vulnerabilities in C. Kim and Yang [67], who utilized the BERT algorithm to predict duplicated bug reports, outperforming existing models and improving bug resolution times.

A technique for developing test oracles, combined with automated testing, improved accuracy by 33%, identifying 57 real-world bugs [68]. da Silva et al. [69] explored various program embeddings and learning models for predictive compilation, with surprisingly simple embeddings performing comparably to more complex ones.

In summary, code embeddings and transformers serve as valuable tools for bug detection and correction, each with its unique strengths. Code embeddings offer a concise representation of code snippets, capturing their semantic and syntactic properties in a fixed-length vector format. This can facilitate efficient similarity comparisons between code segments, aiding in identifying similar bug patterns across projects. However, code embeddings may struggle with capturing complex contextual information and long-range dependencies, potentially leading to limitations in detecting subtle bugs. In contrast, transformers excel in modeling sequential data through self-attention mechanisms, enabling them to capture intricate patterns and contextual information across code segments. This makes transformers particularly effective in detecting and correcting bugs that involve complex interactions and dependencies between code components. Despite the promising results of NLP transformers in bug detection, challenges include the scarcity of large, high-quality datasets and the significant computational resources and training time required. Existing datasets are often language-specific, making generalization to different codebases

challenging. Additionally, the resource-intensive nature of NLP transformers may limit their suitability for real-time bug detection.

4.3. Code Completion

Code completion, a crucial aspect of programming, involves suggesting code to assist programmers in efficiently completing the code they are currently typing. This suggestion can span variable and function names to entire code snippets. The application of transformers in code completion harnesses advanced language models, trained on extensive text data, to enhance developers' coding efficiency. These models exhibit a deep understanding of the context of the code under construction, predicting and suggesting the next code sequence as developers type. This extends beyond basic keyword suggestions, encompassing variable names, function calls, and even the generation of complete code snippets.

The model's proficiency in comprehending syntactic and semantic structures in programming languages ensures accurate and contextually relevant suggestions. It plays a role in identifying and preventing common coding mistakes by offering real-time corrections. Moreover, code completion with transformers often entails providing contextual information such as function signatures, parameter details, and relevant documentation. This not only accelerates the coding process but also aids developers in effectively utilizing various functions and methods.

Roberta [70], another transformer model, has demonstrated impressive results in various natural language processing tasks, showcasing noteworthy performance in code completion. It excels in generating code for diverse programming languages, showcasing a robust understanding of code syntax and context.

Transformer-XL [71], designed to handle longer sequences compared to traditional transformers, has exhibited promising outcomes in code completion tasks, especially when dealing with extensive and intricate sequences. It showcases proficiency in generating code for various programming languages.

CodeFill, proposed by Izadi et al. [72], is a language model for autocompletion leveraging learned structure and naming information. Outperforming several baseline and state-of-the-art models, including GPT-C and TravTrans+, CodeFill excels in both single-token and multi-token prediction. All code and datasets associated with CodeFill are publicly available.

CCMC, presented by Yang and Kuang [29], is a code completion model utilizing a Transformer-XL model for handling long-range dependencies and a pointer network with CopyMask for copying OOV tokens from inputs. The model demonstrates excellent performance in code completion on real-world datasets.

Developers can seamlessly integrate code completion into their preferred integrated development environments (IDEs) or code editors, enhancing the overall coding experience. The interactive and adaptive nature of transformer-based code completion renders it a powerful tool for developers working across various programming languages and frameworks.

Liu et al. [73] introduced a multi-task learning-based pre-trained language model with a transformer-based neural architecture to address challenges in code completion within integrated development environments (IDEs). Experimental results highlight the effectiveness of this approach compared to existing state-of-the-art methods.

BART (bidirectional and auto-regressive transformer), another popular transformer model developed [74], is trained using a combination of supervised and unsupervised learning techniques. Specifically designed for text generation tasks, BART has shown promising results in code generation, achieving state-of-the-art performance in code completion tasks where it predicts the remaining code based on the given context.

A novel neural architecture based on transformer models was proposed and evaluated for autocomplete systems in IDEs, showcasing an accuracy increase of 14–18%. Additionally, an open-source code and data pipeline were released [75]. While transformer models exhibit promise for code completion, further enhancements in accuracy are essential for addressing complex scenarios [30].

In summary, code embeddings and transformers are both valuable tools for code completion, each offering distinct advantages. Code embeddings provide a compact representation of code snippets in a continuous vector space, capturing their semantic and syntactic properties. This allows for efficient retrieval of similar code segments, aiding in suggesting relevant completions based on the context of the code being written. However, code embeddings may struggle with capturing long-range dependencies and contextual nuances, potentially leading to less accurate suggestions in complex coding scenarios. Transformers, on the other hand, excel in modeling sequential data through self-attention mechanisms, enabling them to capture intricate patterns and contextual information across code sequences. This results in more accurate and context-aware code completions, especially in scenarios where understanding broader context and dependencies is crucial.

4.4. Code Generation Process

Code generation involves the task of creating source code based on constraints specified by the programmer in natural language. Hu et al. [23] introduced a supervised code embedding approach along with a tree representation of code snippets, demonstrating enhanced accuracy and efficiency in generating code from natural language compared to current state-of-the-art methods.

Transformers, a type of neural network architecture widely used for various natural language processing (NLP) tasks, including code generation, utilize an attention mechanism to capture long-term dependencies. They excel in handling sequential data without relying on recurrent connections, making them well-suited for tasks involving code generation.

Transformers can be applied to generate functions or methods based on high-level specifications. Developers can articulate the desired functionality in natural language, and the transformer generates the corresponding code.

Svyatkovskiy et al. [3] introduced IntelliCode Compose, a versatile, multilingual code completion tool capable of predicting arbitrary code tokens and generating correctly structured code lines. It was trained on 1.2 billion lines of code across four languages and utilized in the Visual Studio Code IDE and Azure Notebook.

Gemmell et al. [76] explored Transformer architectures for code generation beyond existing IDE capabilities, proposing a "Relevance Transformer" model. Benchmarking results demonstrated improvement over the current state-of-the-art.

Soliman et al. [77] presented MarianCG-NL-to-Code, a code generation transformer model for generating Python code from natural language descriptions. Outperforming state-of-the-art models, it was downloadable on GitHub and evaluated on CoNaLa and DJANGO datasets.

ExploitedGen [78], an exploit code generation approach based on CodeBERT, achieved better accuracy in generating exploit code than existing methods. It incorporated a template-augmented parser and a semantic attention layer, with additional experiments assessing generated code for syntax and semantic accuracy.

Laskari et al. [79] discussed Seq2Code, a transformer-based solution for translating natural language problem statements into Python source code. Using an encoder–decoder transformer design with multi-head attention and separate embeddings for special characters, the model demonstrated improved perplexity compared to similarly structured models.

To summarize the code generation process, code embeddings and transformers offer distinctive approaches, each with its own strengths. Code embeddings condense code snippets into fixed-length vectors, capturing semantic and syntactic information efficiently. This simplifies the generation process by enabling quick retrieval of similar code segments and facilitating straightforward manipulation in vector space. However, code embeddings might struggle with capturing complex dependencies and contextual nuances, potentially limiting their ability to produce diverse and contextually accurate code. In contrast, transformers excel in modeling sequential data through self-attention mechanisms, allowing them to capture intricate patterns and long-range dependencies across code sequences.

This enables transformers to generate code with greater context awareness and flexibility, resulting in more accurate and diverse outputs. Nevertheless, transformers typically demand significant computational resources and extensive training data compared to code embeddings.

4.5. Code Translation

Code translation (Figure 3) involves the conversion of source code from one programming language to another, commonly employed for managing legacy source code. Unlike code generation, which takes natural language as input, code translation deals directly with source code. Bui et al. [80] introduced a bilingual neural network (Bi-NN) architecture for automatically classifying Java and C++ programs. Comprising two sub-networks dedicated to Java and C++ source code, Bi-NN utilized an additional neural network layer to recognize similarities in algorithms and data structures across different languages. Evaluation of a code corpus containing 50 diverse algorithms and data structures revealed promising classification results, with increased accuracy attributed to encoding more semantic information from the source code.

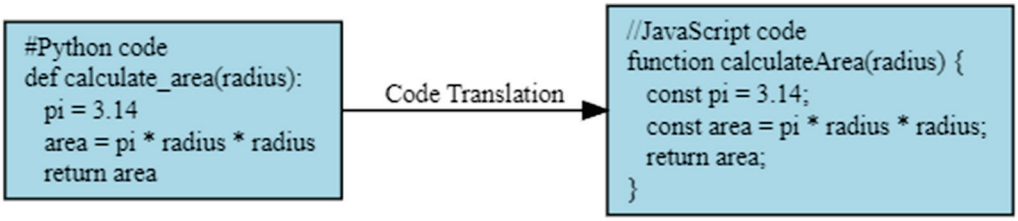


Figure 3. Code translation example.

In contrast to traditional machine translation methods, transformers, which employ self-attention mechanisms instead of recurrent networks, play a pivotal role in code translation. Transformers facilitate the automatic conversion of source code written in one programming language into its equivalent in another language. This capability proves valuable for tasks such as cross-language code migration, integrating code from different languages, or aiding developers familiar with one language in comprehending and working with code written in another.

Hassan et al. [32] introduced a source code converter based on the neural machine translation transformer model, specializing in converting source code between Java and Swift. The model was trained on a merged dataset, and initial results demonstrated promise in terms of the pipeline and code synthesis procedure.

DeepPseudo, presented by Yang et al. [81], leveraged advancements in sequence-to-sequence learning and code semantic learning to automatically generate pseudo-code from source code. Experiment results indicated DeepPseudo's superiority over seven state-of-the-art models, providing a valuable tool for novice developers to understand programming code more easily.

Alokla et al. [82] proposed a new model for generating pseudocode from source code, achieving higher accuracy compared to previous models. This model utilized similarity measures and deep learning transformer models, demonstrating promising results on two datasets.

DLBT, a deep learning-based transformer model for automatically generating pseudocode from source code [83], tokenized the source code and employed a transformer to assess the relatedness between the source code and its corresponding pseudocode. Tested with Python source code, DLBT achieved accuracy and BLEU scores of 47.32 and 68.49, respectively.

Acharjee et al. [84] suggested a method utilizing natural language processing and a sequence-to-sequence deep learning-based model trained on the SPoC dataset for pseudocode conversion. This approach exhibited increased accuracy and efficiency compared to other techniques, as evaluated using bilingual understudy scoring.

To sum up regarding the realm of code generation translation, both code embeddings and transformers offer distinct advantages. Code embeddings condense code snippets into fixed-length vectors, effectively capturing the semantic and syntactic information essential for translation tasks. This approach simplifies the translation process by enabling quick retrieval of similar code segments and facilitating straightforward manipulation in vector space. However, code embeddings may struggle to capture complex dependencies and nuances present in code, potentially limiting their ability to produce accurate translations. On the other hand, transformers excel in modeling sequential data through self-attention mechanisms, allowing them to capture intricate patterns and long-range dependencies across code sequences. This results in more context-aware translations, with the ability to handle a wide range of coding languages and structures.

4.6. Code Comment Generation

The objective of this task is the automatic generation of natural language comments for a given code snippet. Shahbazi et al. [85] introduced API2Com, a comment generation model that utilized Application Programming Interface Documentations (API Docs) as external knowledge resources. The authors observed that API Docs could enhance comment generation, especially when there was only one API in the method. However, as the number of APIs increased, the model output was negatively impacted.

ComFormer, proposed by Yang et al. [86], is a novel code comment generator that integrates transformer and fusion method-based hybrid code presentation. Byte-BPE and Sim_SBT were employed to address out-of-vocabulary (OOV) problems during training. The evaluation involved three metrics and a human study comparing ComFormer to seven state-of-the-art baselines from both code comment and neural machine translation (NMT) domains.

Chakraborty et al. [87] introduced a new pre-training objective for language models for source code, aiming to naturalize the code by utilizing its bi-channel structure (formal and informal). The authors employed six categories of semantic maintaining changes to construct unnatural forms of code for model training. After fine-tuning, the model performed on par with CodeT5, exhibiting improved performance for zero-shot and few-shot learning, as well as better comprehension of code features.

Geng et al. [88] proposed a two-stage method for creating natural language comment texts for code. The approach utilized a model interpretation strategy to refine summaries, enhancing accuracy. Thongtanunam et al. [89] developed AutoTransform, an advanced neural machine translation (NMT) model that significantly increased accuracy in automatically transforming code for code review processes. This innovation aimed to reduce developers' time and effort in manual code review.

BASHEXPLAINER [90] automated code comment generation for Bash scripts, outperforming existing methods based on metrics such as BLEU-3/4, METEOR, and ROUGE-L by up to 9.29%, 8.75%, 4.77%, and 3.86%, respectively. Additionally, it offered a browser plug-in to facilitate the understanding of Bash code.

S-Coach, presented by Lin et al. [91], is a two-phase approach to updating software comments. The first phase utilizes a predictive model to determine if comment updates are code-indicative. If affirmative, an off-the-shelf heuristic-based approach is employed; otherwise, a specially-designed deep learning model is leveraged. Results demonstrated that this approach is more effective than the current state-of-the-art by 20%.

In the domain of code comment generation, both code embeddings and transformers play vital roles, each offering distinct advantages. Code embeddings provide a concise representation of code snippets in a continuous vector space, capturing their semantic and syntactic properties. This facilitates the generation of comments by enabling efficient retrieval of similar code segments and assisting in understanding the context for comment generation. However, code embeddings may struggle with capturing the intricacies and nuances of code, potentially leading to less contextually relevant comments. Transformers, on the other hand, excel in modeling sequential data through self-attention mechanisms,

allowing them to capture complex patterns and dependencies across code sequences. This results in more context-aware and informative comments that better align with the underlying code logic.

4.7. Duplicate Code Detection and Similarity

This task involves identifying duplicate code snippets, whether within the same codebase or across different codebases. Transformers play a crucial role in duplicate code detection, automating the identification of redundant or duplicated code segments within a software project. This process is vital for maintaining code quality, enhancing maintainability, and preventing potential issues associated with code redundancy.

Karakatic et al. [92] introduced a novel method for comparing software systems by computing the robust Hausdorff distance between semantic source code embeddings of each program component. The authors utilized a pre-trained neural network model, code2vec, to generate source code vector representations from various open-source libraries. Employing different types of robust Hausdorff distance, the proposed method demonstrated its suitability for gauging semantic similarity.

The presence of code smells and security smells in various training datasets, a fine-tuned transformer-based GPT-Neo model, and a closed-source code generation tool raised concerns about the cautious application of language models to code generation tasks [93].

Yu et al. [94] proposed BEDetector, a two-channel feature extraction method for binary similarity detection, encompassing contextual semantic feature extraction and a neural GAE model. This system achieved impressive detection rates, including 88.8%, 86.7%, and 100% for resilience against CVE vulnerabilities ssl3-get-key-exchange, ssl3-get-new-session-ticket, and udhcp-get-option, respectively.

Mateless et al. [95] developed Pkg2Vec to encode software packages and predict their authors with remarkable accuracy. Comparisons against state-of-the-art algorithms on the ISOT datasets revealed Pkg2Vec's superior performance, showcasing a 13% increase in accuracy. This demonstrated the efficacy of applying deep learning to improve authorship attribution of software packages, providing deep, interpretable features indicating the unique style and intentions of the programmer.

CodeBERT showed effectiveness for Type-1 and Type-4 clone detection, although its performance declined for unseen functionalities. Fine-tuning was identified as a potential avenue to marginally improve recall [96]. Kovacevic et al. [97] investigated the effectiveness of both ML-based and heuristics-based code smell detection models, utilizing different source code representations (metrics and code embeddings) on the large-scale MLCQ dataset. Transfer learning models were evaluated to analyze the impact of mined knowledge on code smell detection.

An efficient transformer-based code clone detection method was proposed by [98], promising accurate and rapid identification of code clones while significantly reducing computational cost.

To sum up, in the realm of duplicate code detection and similarity analysis, both code embeddings and transformers offer unique advantages. Code embeddings distill code snippets into fixed-length vectors, effectively capturing their semantic and syntactic features. This enables efficient comparison and retrieval of similar code segments, facilitating the identification of duplicate code instances. However, code embeddings may struggle to capture complex dependencies and contextual nuances, potentially limiting their effectiveness in detecting subtle similarities. Transformers, on the other hand, excel in modeling sequential data through self-attention mechanisms, allowing them to capture intricate patterns and long-range dependencies across code sequences. This results in more accurate and context-aware similarity analysis, enabling the detection of subtle variations and similarities within code snippets. Nonetheless, transformers typically require larger computational resources and extensive training data compared to code embeddings.

4.8. Code Refinement

Code refinement (Figure 4) involves identifying and correcting pieces of code susceptible to bugs or vulnerabilities. In the work of Liu et al. [99], a software maintenance method was introduced for debugging method names by evaluating the consistency between their names and code to identify discrepancies. Through experiments on over 2.1 million Java methods, the method achieved an F1-measure of 67.9%, surpassing existing techniques by 15%. Notably, the authors successfully fixed 66 inconsistent method names in a live study on projects in the wild.

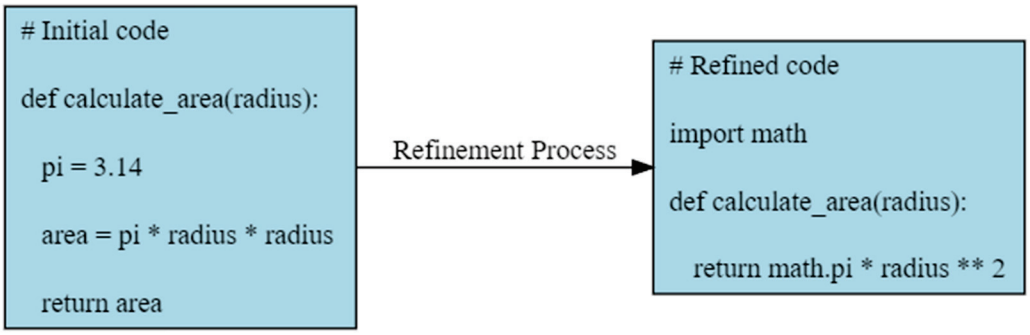


Figure 4. Code refinement example.

Cabrera Lozoya et al. [100] extended a state-of-the-art approach for representing source code to also include changes in the source code (commits). Transfer learning was then applied to classify security-relevant commits. The study demonstrated that representations based on structural information of the code syntax outperformed token-based representations. Moreover, pre-training with a small dataset (greater than 10^4 samples) for a closely related pretext task showed superior performance compared to pre-training with a larger dataset (more than 10^6 samples) and a loosely related pretext task.

Wang et al. [101] introduced Cognac, a context-guidance method name recommender that incorporated global context from methods related by calls. It utilized prior knowledge to adjust method name recommendations and method name consistency checking tasks. Cognac outperformed existing approaches on four datasets with F-scores of 63.2%, 60.8%, 66.3%, and 68.5%, respectively, achieving an overall accuracy of 76.6%, surpassing MNire by 11.2%, a machine learning approach to check the consistency between the name of a given method and its implementation [102].

Xie et al. [103] proposed DeepLink, a model applying code knowledge graph embeddings and deep learning to identify links between issue reports and code commits for software projects. Evaluation of real-world projects demonstrated its superiority over current state-of-the-art solutions.

Borovits et al. [104] presented an automated procedure using word embeddings and deep learning processes to detect inconsistencies between infrastructure as code (IaC) code units and their names. Experiments on an open-source dataset showed an accuracy range of 78.5% to 91.5% in finding such inconsistencies.

Ma et al. [105] introduced Graph-code2vec, a novel self-supervised pre-training approach using code investigation and graph neural networks to generate agnostic task embeddings for software engineering tasks. The proposed technique proved more effective than existing generic and task-specific learning-based baselines, including GraphCodeBERT.

NaturalCC [106] is an open-source code intelligence toolkit, accessible on the website (http://xcodemind.github.io), built on Fairseq and PyTorch technology. It is designed to enable efficient machine learning-based implementation of code intelligence tasks such as code summarization, code retrieval, and code completion.

In the context of code refinement, both code embeddings and transformers offer distinct advantages. Code embeddings condense code snippets into fixed-length vectors, capturing their semantic and syntactic properties efficiently. This facilitates the refinement

process by enabling quick retrieval of similar code segments and aiding in identifying areas for improvement. However, code embeddings may struggle to capture complex dependencies and nuanced coding patterns, potentially limiting their ability to suggest refined solutions accurately. Conversely, transformers excel in modeling sequential data through self-attention mechanisms, enabling them to capture intricate patterns and dependencies across code sequences. This results in more contextually aware refinements, with the ability to suggest solutions that align closely with the underlying logic of the code.

4.9. Code Security

Code security involves checking source code for exploits that may allow unauthorized access to restricted resources. Zaharia et al. [107] proposed the use of an intermediate representation that strikes a balance between stringency to retain security flaws, as per MITRE standards, and dynamism that does not strictly rely on the lexicon of a programming language. This intermediate representation is based on the semantical clusterization of commands in C/C++ programs through word embeddings. These embeddings are distributed through the formed intermediate representation to different classifiers for recognizing security vulnerability patterns.

In related work, Zaharia et al. [108] developed a security scanning system employing machine learning algorithms to detect various patterns of vulnerabilities listed in the Common Weaknesses Enumeration (CWE) from NIST. This system, independent of the programming language, achieved a recall value exceeding 0.94, providing a robust defense against cyber-attacks.

Barr et al. [109] conducted an in-depth analysis of the Fluoride Bluetooth module's source code using deep learning, machine learning, heuristics, and combinatorial optimization techniques. They employed byte-pair encoding to lower dimensionality, embedded tokens into a low-dimensional Euclidean space using LSTM, and created a distance matrix based on cosines between vectors of functions. The authors used cluster-editing to segment the graph's vertices into nearly complete subgraphs, assessing vulnerability risk based on vectors and features of each component.

Saletta and Ferretti [110] discussed a technique using natural language processing to recognize security weaknesses in source code. This involved mapping code to vector space through its abstract syntax trees, and supervised learning to capture distinguishing features among different vulnerabilities. Results demonstrated the model's ability to accurately recognize various types of security weaknesses.

In the domain of code security, both code embeddings and transformers serve as valuable tools, each with its unique strengths. Code embeddings offer a compact representation of code snippets, capturing their semantic and syntactic properties efficiently. This allows for quick analysis of code similarities, aiding in the identification of potential security vulnerabilities based on patterns observed in known security issues. However, code embeddings may struggle to capture complex interactions and subtle security flaws, potentially leading to limitations in detecting sophisticated attacks. Transformers, on the other hand, excel in modeling sequential data and understanding contextual information through self-attention mechanisms. This enables them to capture intricate patterns and dependencies across code sequences, resulting in a more comprehensive and context-aware analysis of code security. However, transformers typically require larger computational resources and extensive training data compared to code embeddings.

5. Datasets

Similar to most deep learning models, transformers demand extensive data to exhibit optimal performance. This becomes a notable challenge in the field of programming, where acquiring high-quality datasets is not as straightforward as in natural language processing (NLP).

To tackle this issue, initiatives like CodeSearchNet (https://github.com/github/CodeSearchNet, accessed on 10 January 2024) and CodeXGLUE (https://github.com/

microsoft/CodeXGLUE, accessed on 10 January 2024) have been established, providing valuable datasets for training and evaluating code-related models. CodeSearchNet stands out as a large-scale dataset, encompassing over 6 million GitHub repositories and 4.2 million code files. It spans six programming languages: Java, Python, JavaScript, Go, Ruby, and PHP. CodeBERT has undergone training on this comprehensive dataset, enhancing its capacity to learn cross-lingual representations of source code.

CodeXGLUE, on the other hand, serves as a benchmark dataset strategically crafted for the advancement and assessment of code intelligence methods, specifically focusing on code completion and code retrieval tasks. This dataset incorporates 14 tasks across various programming languages such as Python, Java, C++, and PHP. CodeBERT, recognizing the significance of diverse challenges, has undergone training on this dataset to elevate its proficiency in code intelligence tasks.

6. Conclusions

Code embeddings serve as vector representations of source code, acquired through deep learning techniques. They adeptly encapsulate the lexical, syntactic, and semantic intricacies of code, projecting them into a high-dimensional vector space. Various methods, including recurrent neural networks (RNNs), convolutional neural networks (CNNs), and graph neural networks (GNNs), are employed to generate code embeddings. These methods utilize input source code to establish a mapping between code tokens and their corresponding vector representations. Subsequently, the vector representations become inputs for downstream natural language processing (NLP) tasks.

Code embeddings prove formidable in capturing the semantic essence of code, distinguishing themselves from traditional approaches reliant on handcrafted features. Unlike their predecessors, code embeddings autonomously learn semantic relationships between distinct code tokens, enhancing efficiency in grasping nuances like variable and function dependencies. Furthermore, code embeddings exhibit language agnosticism, enabling training on diverse programming languages and proving valuable for tasks demanding code comprehension across language boundaries. Their capacity to generalize effectively to unseen code snippets stems from training on extensive code corpora, enabling the absorption of general patterns and structures prevalent in code.

Transformers, distinguished by their self-attention mechanisms, have excelled in learning from substantial datasets in an end-to-end manner, eliminating the need for task-specific feature engineering. This adaptability allows a single transformer model to assist in multiple programming tasks, facilitated by fine-tuning specific languages or tasks. Nevertheless, the performance of a transformer model fine-tuned for one task may not seamlessly translate to another task without further adaptation.

The application of transformers extends from natural language processing (NLP) to code representation and generation tasks. Self-attention mechanisms empower transformers to discern long-range dependencies within input text, enhancing their ability to capture contextual nuances in code.

In Table 1, we summarize the literature review presented in this paper.

Next, we try to compare the use of code embeddings and transformers in the nine referred tasks.

- Code summarization:
    - Code embeddings capture the semantic meaning of code snippets, enabling summarization through techniques like clustering or similarity-based retrieval.
    - Transformers can learn contextual representations of code, allowing them to generate summaries by attending to relevant parts of the code and its surrounding context.
- Bug detection and correction:
    - By learning embeddings from code, similarity metrics can be applied to detect similar code segments containing known bugs, or to identify anomalous patterns.

    - Transformers can learn to detect bugs by learning from labeled data, and they can also be fine-tuned for specific bug detection tasks. For bug correction, they can generate patches by learning from examples of fixed code.
- Code completion:
    - Embeddings can be used to predict the next tokens in code, enabling code completion by suggesting relevant completions based on learned representations.
    - Transformers excel at predicting sequences and can provide context-aware code completions by considering the surrounding code.
- Code generation:
    - Code embeddings can be used to generate code by sampling from the learned embedding space, potentially leading to diverse outputs.
    - Transformers can generate code by conditioning on input sequences and generating output sequences token by token, allowing for precise control over the generation process.
- Code translation:
    - Embeddings can be leveraged for mapping code from one programming language to another by aligning representations of similar functionality across languages.
    - Transformers can be trained for sequence-to-sequence translation tasks, allowing for direct translation of code between different programming languages.
- Code comment generation:
    - By learning embeddings from code-comment pairs, embeddings can be used to generate comments for code by predicting the most likely comment given the code.
    - Transformers can be trained to generate comments by conditioning on code and generating natural language descriptions, capturing the context and intent of the code.
- Duplicate code detection and similarity:
    - Similarity metrics based on embeddings can efficiently identify duplicate or similar code snippets by measuring the distance between their embeddings.
    - Transformers can learn contextual representations of code, enabling them to identify duplicate or similar code snippets by comparing their representations directly.
- Code refinement:
    - Embeddings can be used to refine code by suggesting improvements based on learned representations and similarity to high-quality code.
    - Transformers can be fine-tuned for code refinement tasks, such as code formatting or refactoring, by learning from labeled data or reinforcement learning.
- Code security:
    - Embeddings can be utilized for detecting security vulnerabilities by identifying patterns indicative of vulnerabilities or by comparing code snippets to known vulnerable code.
    - Transformers can be trained to detect security vulnerabilities by learning from labeled data, and they can also be used for code analysis to identify potential security risks through contextual understanding.

Finally, for AI-assisted programming tasks, leveraging both code embeddings and transformers can significantly enhance the efficiency and effectiveness of the development process. By combining the strengths of both techniques, developers can benefit from a comprehensive AI-assisted programming environment that offers efficient code analysis, accurate recommendations, and context-aware assistance throughout the development lifecycle. This hybrid approach ensures that developers can leverage the simplicity and efficiency of code embeddings alongside the contextual awareness and sophistication of transformers, thereby maximizing productivity and code quality.

Table 1. Literature overview.

Tasks	Publications
Code summarization	[16,43–45,48–51]—Code embedding [31,46,47,52]—Transformer
Bug detection and correction	[53–57,61,68,69]—Code embedding [38,58–60,62–67]—Transformer
Code completion	[29,30,71–75]—Transformer
Code generation process	[23]—Code embedding [3,76–79]—Transformer
Code translation	[80,81,84]—Code embedding [32,82,83]—Transformer
Code comment generation	[85,87,88,90]—Code embedding [86]-Code embedding—Transformer [37,89]—Transformer [91]—Custom
Duplicate code detection and similarity	[92,94,95]—Code embedding [92,96,98]—Transformer [97]—Custom
Code refinement	[99–105]—Code embedding [106]—Transformer
Code security	[107–110]—Code embedding

Ethical Considerations

Various ethical considerations come to the forefront when employing transformers, or any form of AI, for programming tasks. These considerations encompass aspects related to privacy, bias, transparency, and accountability [111].

A primary ethical concern centers around the potential invasion of privacy inherent in the utilization of transformers for programming. Given that transformers are engineered to analyze and process extensive datasets, including personal or sensitive information, questions arise concerning the storage, utilization, and safeguarding of these data. A critical aspect involves ensuring individuals are informed about the use of their information for programming purposes.

Another ethical dimension revolves around the prospect of bias within the data used for training the transformer. Should the analyzed data exhibit biases or gaps, it could profoundly impact the decisions made by the transformer, potentially perpetuating existing biases and fostering discrimination. Therefore, it becomes imperative to curate training data that are diverse, representative, and devoid of bias.

Transparency emerges as a pivotal ethical consideration in the integration of transformers for AI-assisted programming tasks. Programmers must possess a comprehensive understanding of the inner workings of the transformer and the rationale behind its decisions. Transparency serves not only debugging and troubleshooting purposes but also acts as a safeguard against the occurrence of unethical or harmful decisions.

Moreover, accountability assumes a critical role in the ethical framework surrounding the use of transformers in programming. With advancing technology, ascertaining responsibility for the decisions made by a transformer becomes increasingly challenging. In scenarios involving errors or ethical breaches, establishing clear frameworks for accountability and liability becomes indispensable. These frameworks serve to assign responsibility and address any ensuing issues with precision and fairness.

Author Contributions: Conceptualization, S.K.; methodology, S.K.; investigation, S.K., M.T. and V.V.; resources, S.K.; data curation, M.T.; writing—original draft preparation, S.K.; writing—review and editing, M.T.; supervision, V.V.; project administration, V.V. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Hindle, A.; Barr, E.T.; Su, Z.; Gabel, M.; Devanbu, P. On The Naturalness of Software. In Proceedings of the 34th International Conference on Software Engineering (ICSE), Zurich, Switzerland, 2–9 June 2012; pp. 837–847.
2. Shani, I. Survey Reveals AI's Impact on the Developer Experience. 2023. Available online: https://github.blog/2023-06-13-survey-reveals-ais-impact-on-the-developer-experience (accessed on 24 December 2023).
3. Svyatkovskiy, A.; Deng, S.K.; Fu, S.; Sundaresan, N. IntelliCode compose: Code generation using transformer. In Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Online, 8–13 November 2020. [CrossRef]
4. Bird, C.; Ford, D.; Zimmermann, T.; Forsgren, N.; Kalliamvakou, E.; Lowdermilk, T.; Gazit, I. Taking Flight with Copilot. *Commun. ACM* **2023**, *66*, 56–62. [CrossRef]
5. Friedman, N. Introducing GitHub Copilot: Your AI Pair Programmer. 2021. Available online: https://github.com/features/copilot (accessed on 24 December 2023).
6. Chen, M.; Tworek, J.; Jun, H.; Yuan, Q.; de Oliveira Pinto, H.P.; Kaplan, J.; Edwards, H.; Burda, Y.; Joseph, N.; Brockman, G.; et al. Evaluating large language models trained on code. *arXiv* **2021**, arXiv:2107.03374. [CrossRef]
7. Li, Y.; Choi, D.; Chung, J.; Kushman, N.; Schrittwieser, J.; Leblond, R.; Eccles, T.; Keeling, J.; Gimeno, F.; Dal Lago, A.; et al. Competition-level Code Generation with Alphacode. *Science* **2022**, *378*, 1092–1097. [CrossRef] [PubMed]
8. Parashar, B.; Kaur, I.; Sharma, A.; Singh, P.; Mishra, D. Revolutionary transformations in twentieth century: Making AI-assisted software development. In *Computational Intelligence in Software Modeling*; De Gruyter: Berlin, Germany, 2022. [CrossRef]
9. Gulwani, S. AI-assisted programming: Applications, user experiences, and neuro-symbolic techniques (keynote). In Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Singapore, 14–18 November 2022. [CrossRef]
10. Vaithilingam, P.; Zhang, T.; Glassman, E.L. Expectation vs. experience: Evaluating the usability of code generation tools powered by large language models. In Proceedings of the CHI Conference on Human Factors in Computing Systems Extended Abstracts, New Orleans, LA, USA, 29 April–5 May 2022; Association for Computing Machinery: New York, NY, USA, 2022; pp. 1–7.
11. Fernandez, R.C.; Elmore, A.J.; Franklin, M.J.; Krishnan, S.; Tan, C. How Large Language Models Will Disrupt Data Management. *Proc. VLDB Endow.* **2023**, *16*, 3302–3309. [CrossRef]
12. Zhou, H.; Li, J. A Case Study on Scaffolding Exploratory Data Analysis for AI Pair Programmers. In Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems, Hamburg, Germany, 23–28 April 2023; pp. 1–7. [CrossRef]
13. Kazemitabaar, M.; Chow, J.; Ma, C.K.T.; Ericson, B.J.; Weintrop, D.; Grossman, T. Studying the effect of AI Code Generators on Supporting Novice Learners in Introductory Programming. In Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems, Hamburg, Germany, 23–28 April 2023; pp. 1–23. [CrossRef]
14. Daun, M.; Brings, J. How ChatGPT Will Change Software Engineering Education. In Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1, Turku, Finland, 7–12 July 2023; pp. 110–116. [CrossRef]
15. Prather, J.; Reeves, B.N.; Denny, P.; Becker, B.A.; Leinonen, J.; Luxton-Reilly, A.; Powell, G.; Finnie-Ansley, J.; Santos, E.A. "It's Weird That It Knows What I Want": Usability and Interactions with Copilot for Novice Programmers. *ACM Trans. Comput. Interact.* **2023**, *31*, 1–31. [CrossRef]
16. Sui, Y.; Cheng, X.; Zhang, G.; Wang, H. Flow2Vec: Value-flow-based precise code embedding. *Proc. ACM Program. Lang.* **2020**, *4*, 233. [CrossRef]
17. Rabin, M.R.I.; Mukherjee, A.; Gnawali, O.; Alipour, M.A. Towards demystifying dimensions of source code embeddings. In Proceedings of the 1st ACM SIGSOFT International Workshop on Representation Learning for Software Engineering and Program Languages, Online, 8–13 November 2020. [CrossRef]
18. Azcona, D.; Arora, P.; Hsiao, I.-H.; Smeaton, A. user2code2vec: Embedding for Profiling Students Based on Distributinal Representations of Source Code. In Proceedings of the 9th International Conference on Learning Analytics and Knowledge, Tempe, AZ, USA, 4–8 March 2019. [CrossRef]
19. Ding, Z.; Li, H.; Shang, W.; Chen, T.-H. Towards Learning Generalizable Code Embeddings Using Task-agnostic Graph Convolutional Networks. *ACM Trans. Softw. Eng. Methodol.* **2023**, *32*, 48. [CrossRef]
20. Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; et al. Transformers: State-of-the-Art Natural Language Processing. In *EMNLP 2020—Conference on Empirical Methods in Natural Language Processing: Systems Demonstrations*; Association for Computational Linguistics: Kerrville, TX, USA, 2020; pp. 38–45.

21. Chirkova, N.; Troshin, S. Empirical study of transformers for source code. In Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Athens, Greece, 23–28 August 2021. [CrossRef]
22. Song, Y.; Shi, S.; Li, J.; Zhang, H. Directional skip-gram: Explicitly distinguishing left and right context forword embeddings. In Proceedings of the NAACL HLT 2018—2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, LA, USA, 1–6 June 2018; pp. 175–180.
23. Hu, H.; Chen, Q.; Liu, Z. Code Generation from Supervised Code Embeddings. In *Neural Information Processing*; Springer: Cham, Switzerland, 2019; pp. 388–396. [CrossRef]
24. Sikka, J.; Satya, K.; Kumar, Y.; Uppal, S.; Shah, R.R.; Zimmermann, R. Learning Based Methods for Code Runtime Complexity Prediction. In *Advances in Information Retrieval*; Springer: Cham, Switzerland, 2020; pp. 313–325. [CrossRef]
25. Kang, H.J.; Bissyande, T.F.; Lo, D. Assessing the Generalizability of Code2vec Token Embeddings. In Proceedings of the 2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE), San Diego, CA, USA, 11–15 November 2019. [CrossRef]
26. Romanov, V.; Ivanov, V. Prediction of Types in Python with Pre-trained Graph Neural Networks. In Proceedings of the 2022 Ivannikov Memorial Workshop (IVMEM), Moscow, Russia, 23–24 September 2022. [CrossRef]
27. Ding, Z.; Li, H.; Shang, W.; Chen, T.-H.P. Can pre-trained code embeddings improve model performance? Revisiting the use of code embeddings in software engineering tasks. *Empir. Softw. Eng.* **2022**, *27*, 63. [CrossRef]
28. Shaw, P.; Uszkoreit, J.; Vaswani, A. Self-attention with relative position representations. In Proceedings of the NAACL HLT 2018—2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, LA, USA, 1–6 June 2018; pp. 464–468.
29. Yang, H.; Kuang, L. CCMC: Code Completion with a Memory Mechanism and a Copy Mechanism. In Proceedings of the EASE 2021: Evaluation and Assessment in Software Engineering, Trondheim, Norway, 21–23 June 2021. [CrossRef]
30. Ciniselli, M.; Cooper, N.; Pascarella, L.; Mastropaolo, A.; Aghajani, E.; Poshyvanyk, D.; Di Penta, M.; Bavota, G. An Empirical Study on the Usage of Transformer Models for Code Completion. *IEEE Trans. Softw. Eng.* **2021**, *48*, 4818–4837. [CrossRef]
31. Gong, Z.; Gao, C.; Wang, Y.; Gu, W.; Peng, Y.; Xu, Z. Source Code Summarization with Structural Relative Position Guided Transformer. In Proceedings of the 2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), Honolulu, HI, USA, 15–18 March 2022. [CrossRef]
32. Hassan, M.H.; Mahmoud, O.A.; Mohammed, O.I.; Baraka, A.Y.; Mahmoud, A.T.; Yousef, A.H. Neural Machine Based Mobile Applications Code Translation. In Proceedings of the 2020 2nd Novel Intelligent and Leading Emerging Sciences Conference (NILES), Giza, Egypt, 24–26 October 2020. [CrossRef]
33. Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the NAACL HLT 2019—2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA, 2–7 June 2019; pp. 4171–4186.
34. Sengupta, A.; Kumar, A.; Bhattacharjee, S.K.; Roy, S. Gated Transformer for Robust De-noised Sequence-to-Sequence Modelling. In Proceedings of the 2021 Findings of the Association for Computational Linguistics, Punta Cana, Dominican Republic, 7–11 November 2021.
35. Wu, C.; Wu, F.; Ge, S.; Qi, T.; Huang, Y.; Xie, X. Neural news recommendation with multi-head self-attention. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Hong Kong, China, 3–7 November 2019.
36. Chernyavskiy, A.; Ilvovsky, D.; Nakov, P. Transformers: 'The End of History' for Natural Language Processing? In *Machine Learning and Knowledge Discovery in Databases*; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2021; pp. 677–693. [CrossRef]
37. Feng, Z.; Guo, D.; Tang, D.; Duan, N.; Feng, X.; Gong, M.; Shou, L.; Qin, B.; Liu, T.; Jiang, D.; et al. CodeBERT: A pre-trained model for programming and natural languages. In *Findings of the Association for Computational Linguistics Findings of ACL: EMNLP 2020*; Association for Computational Linguistics: Kerrville, TX, USA, 2020; pp. 1536–1547.
38. Zhou, X.; Han, D.; Lo, D. Assessing Generalizability of CodeBERT. In Proceedings of the 2021 IEEE International Conference on Software Maintenance and Evolution (ICSME), Luxembourg, 27 September–1 October 2021. [CrossRef]
39. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.* **2020**, *21*, 1–67. Available online: http://jmlr.org/papers/v21/20-074.html (accessed on 24 December 2023).
40. Brown, T.B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 1877–1901.
41. Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.; Le, Q.V. XLNet: Generalized autoregressive pretraining for language understanding. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 5753–5763.
42. Zhang, F.; Yu, X.; Keung, J.; Li, F.; Xie, Z.; Yang, Z.; Ma, C.; Zhang, Z. Improving Stack Overflow question title generation with copying enhanced CodeBERT model and bi-modal information. *Inf. Softw. Technol.* **2022**, *148*, 106922. [CrossRef]
43. Liu, K.; Yang, G.; Chen, X.; Zhou, Y. EL-CodeBert: Better Exploiting CodeBert to Support Source Code-Related Classification Tasks. In Proceedings of the 13th Asia-Pacific Symposium on Internetware, Hohhot, China, 11–12 June 2022. [CrossRef]

44. Wang, R.; Zhang, H.; Lu, G.; Lyu, L.; Lyu, C. Fret: Functional Reinforced Transformer with BERT for Code Summarization. *IEEE Access* **2020**, *8*, 135591–135604. [CrossRef]
45. Yang, Z.; Keung, J.; Yu, X.; Gu, X.; Wei, Z.; Ma, X.; Zhang, M. A Multi-Modal Transformer-based Code Summarization Approach for Smart Contracts. In Proceedings of the 2021 IEEE/ACM 29th International Conference on Program Comprehension (ICPC), Madrid, Spain, 20–21 May 2021. [CrossRef]
46. Hou, S.; Chen, L.; Ye, Y. Summarizing Source Code from Structure and Context. In Proceedings of the 2022 International Joint Conference on Neural Networks (IJCNN), Padua, Italy, 18–23 July 2022. [CrossRef]
47. Wang, Y.; Dong, Y.; Lu, X.; Zhou, A. GypSum: Learning hybrid representations for code summarization. In Proceedings of the 30th IEEE/ACM International Conference on Program Comprehension, Online, 16–17 May 2022. [CrossRef]
48. Gu, J.; Salza, P.; Gall, H.C. Assemble Foundation Models for Automatic Code Summarization. In Proceedings of the 2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), Honolulu, HI, USA, 15–18 March 2022. [CrossRef]
49. Ma, Z.; Gao, Y.; Lyu, L.; Lyu, C. MMF3: Neural Code Summarization Based on Multi-Modal Fine-Grained Feature Fusion. In Proceedings of the 16th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, Helsinki, Finland, 29–23 September 2022. [CrossRef]
50. Gao, Y.; Lyu, C. M2TS: Multi-scale multi-modal approach based on transformer for source code summarization. In Proceedings of the 30th IEEE/ACM International Conference on Program Comprehension, Online, 16–17 May 2022. [CrossRef]
51. Ferretti, C.; Saletta, M. Naturalness in Source Code Summarization. How Significant is it? In Proceedings of the 2023 IEEE/ACM 31st International Conference on Program Comprehension (ICPC), Melbourne, VI, Australia, 15–16 May 2023. [CrossRef]
52. Choi, Y.; Na, C.; Kim, H.; Lee, J.-H. READSUM: Retrieval-Augmented Adaptive Transformer for Source Code Summarization. *IEEE Access* **2023**, *11*, 51155–51165. [CrossRef]
53. Aladics, T.; Jasz, J.; Ferenc, R. Bug Prediction Using Source Code Embedding Based on Doc2Vec. In *Computational Science and Its Applications*; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2021; pp. 382–397. [CrossRef]
54. Cheng, X.; Zhang, G.; Wang, H.; Sui, Y. Path-sensitive code embedding via contrastive learning for software vulnerability detection. In Proceedings of the 31st ACM SIGSOFT International Symposium on Software Testing and Analysis, Online, Republic of Korea, 18–22 July 2022. [CrossRef]
55. Hegedus, P.; Ferenc, R. Static Code Analysis Alarms Filtering Reloaded: A New Real-World Dataset and its ML-Based Utilization. *IEEE Access* **2022**, *10*, 55090–55101. [CrossRef]
56. Bagheri, A.; Hegedus, P. A Comparison of Different Source Code Representation Methods for Vulnerability Prediction in Python. In *Quality of Information and Communications Technology*; Springer: Cham, Switzerland, 2021; pp. 267–281. [CrossRef]
57. Gomes, L.; da Silva Torres, R.; Cortes, M.L. BERT- and TF-IDF-based feature extraction for long-lived bug prediction in FLOSS: A comparative study. *Inf. Softw. Technol.* **2023**, *160*, 107217. [CrossRef]
58. Pan, C.; Lu, M.; Xu, B. An Empirical Study on Software Defect Prediction Using CodeBERT Model. *Appl. Sci.* **2021**, *11*, 4793. [CrossRef]
59. Ma, X.; Keung, J.W.; Yu, X.; Zou, H.; Zhang, J.; Li, Y. AttSum: A Deep Attention-Based Summarization Model for Bug Report Title Generation. *IEEE Trans. Reliab.* **2023**, *72*, 1663–1677. [CrossRef]
60. Mahbub, P.; Shuvo, O.; Rahman, M.M. Explaining Software Bugs Leveraging Code Structures in Neural Machine Translation. In Proceedings of the 2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE), Melbourne, VI, Australia, 14–20 May 2023. [CrossRef]
61. Csuvik, V.; Horvath, D.; Lajko, M.; Vidacs, L. Exploring Plausible Patches Using Source Code Embeddings in JavaScript. In Proceedings of the 2021 IEEE/ACM International Workshop on Automated Program Repair (APR), Madrid, Spain, 1 June 2021. [CrossRef]
62. Mashhadi, E.; Hemmati, H. Applying CodeBERT for Automated Program Repair of Java Simple Bugs. In Proceedings of the 2021 IEEE/ACM 18th International Conference on Mining Software Repositories (MSR), Madrid, Spain, 17–19 May 2021. [CrossRef]
63. Chakraborty, S.; Ray, B. On Multi-Modal Learning of Editing Source Code. In Proceedings of the 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE), Melbourne, VI, Australia, 15–19 November 2021. [CrossRef]
64. Lajko, M.; Csuvik, V.; Vidacs, L. Towards JavaScript program repair with generative pre-trained transformer (GPT-2). In Proceedings of the Third International Workshop on Automated Program Repair, Pittsburgh, PA, USA, 19 May 2022. [CrossRef]
65. Chi, J.; Qu, Y.; Liu, T.; Zheng, Q.; Yin, H. SeqTrans: Automatic Vulnerability Fix Via Sequence to Sequence Learning. *IEEE Trans. Softw. Eng.* **2023**, *49*, 564–585. [CrossRef]
66. Chen, Z.; Kommrusch, S.; Monperrus, M. Neural Transfer Learning for Repairing Security Vulnerabilities in C Code. *IEEE Trans. Softw. Eng.* **2023**, *49*, 147–165. [CrossRef]
67. Kim, T.; Yang, G. Predicting Duplicate in Bug Report Using Topic-Based Duplicate Learning with Fine Tuning-Based BERT Algorithm. *IEEE Access* **2022**, *10*, 129666–129675. [CrossRef]
68. Dinella, E.; Ryan, G.; Mytkowicz, T.; Lahiri, S.K. TOGA: A neural method for test oracle generation. In Proceedings of the 44th International Conference on Software Engineering, Pittsburgh, PA, USA, 21–29 May 2022. [CrossRef]
69. da Silva, A.F.; Borin, E.; Pereira, F.M.Q.; Queiroz, N.L.; Napoli, O.O. Program representations for predictive compilation: State of affairs in the early 20's. *J. Comput. Lang.* **2022**, *73*, 101171. [CrossRef]

70. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv* **2019**, arXiv:1907.11692.
71. Dai, Z.; Yang, Z.; Yang, Y.; Carbonell, J.; Le, Q.V.; Salakhutdinov, R. Transformer-XL: Attentive language models beyond a fixed-length context. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 2978–2988.
72. Izadi, M.; Gismondi, R.; Gousios, G. CodeFill: Multi-token code completion by jointly learning from structure and naming sequences. In Proceedings of the 44th International Conference on Software Engineering, Pittsburgh, PA, USA, 21–29 May 2022. [CrossRef]
73. Liu, F.; Li, G.; Zhao, Y.; Jin, Z. Multi-task learning based pre-trained language model for code completion. In Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering, Virtual Event Australia, 21–25 December 2020. [CrossRef]
74. Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; Zettlemoyer, L. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Proceedings of the Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020; pp. 7871–7880.
75. Kim, S.; Zhao, J.; Tian, Y.; Chandra, S. Code Prediction by Feeding Trees to Transformers. In Proceedings of the 2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE), Madrid, Spania, 22–30 May 2021. [CrossRef]
76. Gemmell, C.; Rossetto, F.; Dalton, J. Relevance Transformer: Generating Concise Code Snippets with Relevance Feedback. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event China, 25–30 July 2020. [CrossRef]
77. Soliman, A.S.; Hadhoud, M.M.; Shaheen, S.I. MarianCG: A code generation transformer model inspired by machine translation. *J. Eng. Appl. Sci.* **2022**, *69*, 104. [CrossRef]
78. Yang, G.; Zhou, Y.; Chen, X.; Zhang, X.; Han, T.; Chen, T. ExploitGen: Template-augmented exploit code generation based on CodeBERT. *J. Syst. Softw.* **2023**, *197*, 111577. [CrossRef]
79. Laskari, N.K.; Reddy, K.A.N.; Indrasena Reddy, M. Seq2Code: Transformer-Based Encoder-Decoder Model for Python Source Code Generation. In *Third Congress on Intelligent Systems*; Lecture Notes in Networks and Systems; Springer: Singapore, 2023; pp. 301–309. [CrossRef]
80. Bui, N.D.Q.; Yu, Y.; Jiang, L. Bilateral Dependency Neural Networks for Cross-Language Algorithm Classification. In Proceedings of the 2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER), Hangzhou, China, 24–27 February 2019. [CrossRef]
81. Yang, G.; Zhou, Y.; Chen, X.; Yu, C. Fine-grained Pseudo-code Generation Method via Code Feature Extraction and Transformer. In Proceedings of the 2021 28th Asia-Pacific Software Engineering Conference (APSEC), Taipei, Taiwan, 6–9 December 2021. [CrossRef]
82. Alokla, A.; Gad, W.; Nazih, W.; Aref, M.; Salem, A.-B. Retrieval-Based Transformer Pseudocode Generation. *Mathematics* **2022**, *10*, 604. [CrossRef]
83. Gad, W.; Alokla, A.; Nazih, W.; Aref, M.; Salem, A. DLBT: Deep Learning-Based Transformer to Generate Pseudo-Code from Source Code. *Comput. Mater. Contin.* **2022**, *70*, 3117–3132. [CrossRef]
84. Acharjee, U.K.; Arefin, M.; Hossen, K.M.; Uddin, M.N.; Uddin, M.A.; Islam, L. Sequence-to-Sequence Learning-Based Conversion of Pseudo-Code to Source Code Using Neural Translation Approach. *IEEE Access* **2022**, *10*, 26730–26742. [CrossRef]
85. Shahbazi, R.; Sharma, R.; Fard, F.H. API2Com: On the Improvement of Automatically Generated Code Comments Using API Documentations. In Proceedings of the 2021 IEEE/ACM 29th International Conference on Program Comprehension (ICPC), Madrid, Spain, 20–21 May 2021. [CrossRef]
86. Yang, G.; Chen, X.; Cao, J.; Xu, S.; Cui, Z.; Yu, C.; Liu, K. ComFormer: Code Comment Generation via Transformer and Fusion Method-based Hybrid Code Representation. In Proceedings of the 2021 8th International Conference on Dependable Systems and Their Applications (DSA), Yinchuan, China, 5–6 August 2021. [CrossRef]
87. Chakraborty, S.; Ahmed, T.; Ding, Y.; Devanbu, P.T.; Ray, B. NatGen: Generative pre-training by "naturalizing" source code. In Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Singapore, 14–18 November 2022. [CrossRef]
88. Geng, M.; Wang, S.; Dong, D.; Wang, H.; Cao, S.; Zhang, K.; Jin, Z. Interpretation-based Code Summarization. In Proceedings of the 2023 IEEE/ACM 31st International Conference on Program Comprehension (ICPC), Melbourne, VI, Australia, 15–16 May 2023. [CrossRef]
89. Thongtanunam, P.; Pornprasit, C.; Tantithamthavorn, C. AutoTransform: Automated code transformation to support modern code review process. In Proceedings of the 44th International Conference on Software Engineering, Pittsburgh, PA, USA, 21–29 May 2022. [CrossRef]
90. Yu, C.; Yang, G.; Chen, X.; Liu, K.; Zhou, Y. BashExplainer: Retrieval-Augmented Bash Code Comment Generation based on Fine-tuned CodeBERT. In Proceeding of the 2022 IEEE International Conference on Software Maintenance and Evolution (ICSME), Limassol, Cyprus, 3–7 October 2022. [CrossRef]
91. Lin, B.; Wang, S.; Liu, Z.; Xia, X.; Mao, X. Predictive Comment Updating with Heuristics and AST-Path-Based Neural Learning: A Two-Phase Approach. *IEEE Trans. Softw. Eng.* **2023**, *49*, 1640–1660. [CrossRef]

92. Karakatic, S.; MiloÅ¡evic, A.; Hericko, T. Software system comparison with semantic source code embeddings. *Empir. Softw. Eng.* **2022**, *27*, 70. [CrossRef]
93. Siddiq, M.L.; Majumder, S.H.; Mim, M.R.; Jajodia, S.; Santos, J.C.S. An Empirical Study of Code Smells in Transformer-based Code Generation Techniques. In Proceedings of the 2022 IEEE 22nd International Working Conference on Source Code Analysis and Manipulation (SCAM), Limassol, Cyprus, 3 October 2022. [CrossRef]
94. Yu, L.; Lu, Y.; Shen, Y.; Huang, H.; Zhu, K. BEDetector: A Two-Channel Encoding Method to Detect Vulnerabilities Based on Binary Similarity. *IEEE Access* **2021**, *9*, 51631–51645. [CrossRef]
95. Mateless, R.; Tsur, O.; Moskovitch, R. Pkg2Vec: Hierarchical package embedding for code authorship attribution. *Future Gener. Comput. Syst.* **2021**, *116*, 49–60. [CrossRef]
96. Arshad, S.; Abid, S.; Shamail, S. CodeBERT for Code Clone Detection: A Replication Study. In Proceedings of the 2022 IEEE 16th International Workshop on Software Clones (IWSC), Limassol, Cyprus, 2 October 2022. [CrossRef]
97. Kovacevic, A.; Slivka, J.; Vidakovic, D.; Grujic, K.-G.; Luburic, N.; Prokic, S.; Sladic, G. Automatic detection of Long Method and God Class code smells through neural source code embeddings. *Expert Syst. Appl.* **2022**, *204*, 117607. [CrossRef]
98. Zhang, A.; Fang, L.; Ge, C.; Li, P.; Liu, Z. Efficient transformer with code token learner for code clone detection. *J. Syst. Softw.* **2023**, *197*, 111557. [CrossRef]
99. Liu, K.; Kim, D.; Bissyande, T.F.; Kim, T.; Kim, K.; Koyuncu, A.; Kim, S.; Le Traon, Y. Learning to Spot and Refactor Inconsistent Method Names. In Proceedings of the 2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE), Montreal, QC, Canada, 25–31 May 2019. [CrossRef]
100. Cabrera Lozoya, R.; Baumann, A.; Sabetta, A.; Bezzi, M. Commit2Vec: Learning Distributed Representations of Code Changes. *SN Comput. Sci.* **2021**, *2*, 150. [CrossRef]
101. Wang, S.; Wen, M.; Lin, B.; Mao, X. Lightweight global and local contexts guided method name recommendation with prior knowledge. In Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Athens, Greece, 23–28 August 2021. [CrossRef]
102. Nguyen, S.; Phan, H.; Le, T.; Nguyen, T.N. Suggesting natural method names to check name consistencies. In Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering (ICSE '20). Association for Computing Machinery, New York, NY, USA; 2020; pp. 1372–1384. [CrossRef]
103. Xie, R.; Chen, L.; Ye, W.; Li, Z.; Hu, T.; Du, D.; Zhang, S. DeepLink: A Code Knowledge Graph Based Deep Learning Approach for Issue-Commit Link Recovery. In Proceedings of the 2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER), Hangzhou, China, 24–27 February 2019. [CrossRef]
104. Borovits, N.; Kumara, I.; Krishnan, P.; Palma, S.D.; Di Nucci, D.; Palomba, F.; Tamburri, D.A.; van den Heuvel, W.-J. DeepIaC: Deep learning-based linguistic anti-pattern detection in IaC. In Proceedings of the 4th ACM SIGSOFT International Workshop on Machine-Learning Techniques for Software-Quality Evaluation, Virtual, USA, 13 November 2020. [CrossRef]
105. Ma, W.; Zhao, M.; Soremekun, E.; Hu, Q.; Zhang, J.M.; Papadakis, M.; Cordy, M.; Xie, X.; Traon, Y.L. GraphCode2Vec: Generic code embedding via lexical and program dependence analysis. In Proceedings of the 19th International Conference on Mining Software Repositories, Pittsburg, PA, USA, 23–24 May 2022. [CrossRef]
106. Wan, Y.; He, Y.; Bi, Z.; Zhang, J.; Sui, Y.; Zhang, H.; Hashimoto, K.; Jin, H.; Xu, G.; Xiong, C.; et al. NaturalCC: An Open-Source Toolkit for Code Intelligence. In Proceedings of the 2022 IEEE/ACM 44th International Conference on Software Engineering: Companion Proceedings (ICSE-Companion), Pittsburgh, PA, USA, 22–24 May 2022. [CrossRef]
107. Zaharia, S.; Rebedea, T.; Trausan-Matu, S. CWE Pattern Identification using Semantical Clustering of Programming Language Keywords. In Proceedings of the 2021 23rd International Conference on Control Systems and Computer Science (CSCS), Bucharest, Romania, 26–28 May 2021. [CrossRef]
108. Zaharia, S.; Rebedea, T.; Trausan-Matu, S. Machine Learning-Based Security Pattern Recognition Techniques for Code Developers. *Appl. Sci.* **2022**, *12*, 12463. [CrossRef]
109. Barr, J.R.; Shaw, P.; Abu-Khzam, F.N.; Thatcher, T.; Yu, S. Vulnerability Rating of Source Code with Token Embedding and Combinatorial Algorithms. *Int. J. Semant. Comput.* **2020**, *14*, 501–516. [CrossRef]
110. Saletta, M.; Ferretti, C. A Neural Embedding for Source Code: Security Analysis and CWE Lists. In Proceedings of the 2020 IEEE International Conference on Dependable, Autonomic and Secure Computing, International Conference on Pervasive Intelligence and Computing, International Conference on Cloud and Big Data Computing, International Conference on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech), Calgary, AB, Canada, 17–22 August 2020. [CrossRef]
111. Hamed, A.A.; Zachara-Szymanska, M.; Wu, X. Safeguarding authenticity for mitigating the harms of generative AI: Issues, research agenda, and policies for detection, fact-checking, and ethical AI. *IScience* **2024**, *27*, 108782. [CrossRef]

electronics

MDPI

Article

Augmenting Large Language Models with Rules for Enhanced Domain-Specific Interactions: The Case of Medical Diagnosis

Dimitrios P. Panagoulias *, Maria Virvou and George A. Tsihrintzis

Department of Informatics, University of Piraeus, 80 Karaoli ke Dimitriou ST, 18534 Piraeus, Greece; mvirvou@unipi.gr (M.V.); geoatsi@unipi.gr (G.A.T.)
* Correspondence: panagoulias_d@unipi.gr

Abstract: In this paper, we present a novel Artificial Intelligence (AI) -empowered system that enhances large language models and other machine learning tools with rules to provide primary care diagnostic advice to patients. Specifically, we introduce a novel methodology, represented through a process diagram, which allows the definition of generative AI processes and functions with a focus on the rule-augmented approach. Our methodology separates various components of the generative AI process as blocks that can be used to generate an implementation data flow diagram. Building upon this framework, we utilize the concept of a dialogue process as a theoretical foundation. This is specifically applied to the interactions between a user and an AI-empowered software program, which is called "Med | Primary AI assistant" (Alpha Version at the time of writing), and provides symptom analysis and medical advice in the form of suggested diagnostics. By leveraging current advancements in natural language processing, a novel approach is proposed to define a blueprint of domain-specific knowledge and a context for instantiated advice generation. Our approach not only encompasses the interaction domain, but it also delves into specific content that is relevant to the user, offering a tailored and effective AI–user interaction experience within a medical context. Lastly, using an evaluation process based on rules, defined by context and dialogue theory, we outline an algorithmic approach to measure content and responses.

Keywords: AI-empowered software engineering; generative AI; dialogue theory; large language models; natural language processing; rule-augmented systems; medical diagnosis; evaluation

1. Introduction

Healthcare experiences for patients are multifaceted, encompassing dynamic doctor–patient interactions, diverse diagnosis and treatment methods, adherence to recommended lifestyle or suggested behavioral changes, and ongoing preventive health measures. A patient's healthcare journey is clearly non-linear, forming a comprehensive and interwoven sequence of events and encounters [1–4]. For example, the diagnostic procedure in medicine often combines different approaches, which are influenced by the context, patient symptoms, clinician expertise, and available diagnostic tools [5]. Indeed, as illustrated and summarized in Figure 1, the diagnostic process begins with gathering patient data, including a medical history and possibly a physical examination. This information is then analyzed for patterns to assist in decision making. The process further refines and confirms initial hypotheses about the condition using the collected data, which leads to the creation of a treatment strategy, the monitoring of patient progress, and the tracking of disease progression.

Recent advancements and ongoing research have allowed significant progress in digitizing a great portion of the healthcare process, with the aim to alleviate the burdens and costs of primary care, while improving patients' experiences. Some methodologies utilize natural language processing (NLP), big data analysis, and machine learning (ML) technologies [6–8] to digitize, compress, and accelerate healthcare processes. Indeed, these

technologies are promising to revolutionize patient care and disease management by automating tasks, streamlining workflows, reducing manual labor, and simplifying daily activities for all stakeholders [9,10].

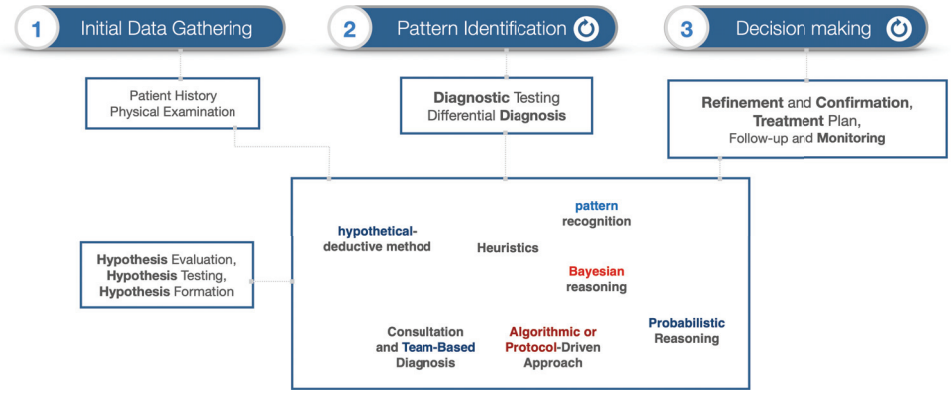


Figure 1. Medical diagnosis pathways.

One such emergent technology showing great promise to revolutionize healthcare is the technology of large language models (LLMs). Indeed, LLMs demonstrate a remarkable capability of understanding medical texts and identifying (diagnosing) a range of symptoms and health conditions. An exemplary LLM is GPT by OpenAI, powering ChatGPT, which generates accurate, human-like text responses [11]. Other notable LLMs include Google's BERT (Bidirectional Encoder Representations from Transformers) [12], Meta's Llama (Large Language Model Meta AI) [13], and Stanford's Alpaca (fine-tuned from the Llama model) [14]. While each LLM and NLP approach has its limitations, selectively integrating elements from various technologies can offer both efficacy and cost-effectiveness.

In recent studies, a novel general three-step methodology was proposed to evaluate the potential of LLMs and, more specifically, ChatGPT in medical diagnosis and treatment [15]. The evaluation of ChatGPT's performance, as per its communication capability in radiology [16] and oncology [17], has also been conducted. It was found that, under different circumstances, ChatGPT performed at an average to optimum level. Moreover, it was found that ChatGPT and other NLPs/LLMs could potentially perform better under the supervision and assistance of a medical expert, who could evaluate the ChatGPT answers better than a patient.

Based on these previous findings, in our current work, we introduce a novel rule-augmented AI-empowered system in which a rule-based decision mechanism is integrated with an LLM engine and various external machine learning and analytical APIs.

Our system includes the following novelties and key contributions.

- The domain space of AI–user interaction is associated with rules of dialogue to be followed, as detailed later in the paper in Table 1 in Section 4.1. This provides a theoretical basis for the evaluation of the performance and the assessment of an LLM's ability to remain within these constraints, which aim to simulate real-time/real-world interactions. The process is systemized and generalized to reach a measurable conclusion on the LLM answers, within a domain-specific context and a dialogue-defined space.
- Using NLP algorithms, we define the blueprint of domain-specific knowledge and the domain-specific content that is relevant to the user. This enhances the AI–user interaction experience within a medical context.
- A methodology is introduced, represented through a process diagram, aimed at defining generative AI processes and functions with a rule-augmented approach for the prototyping of AI-empowered systems.
- The system, which utilizes the GPT-4 engine, has undergone extensive evaluation through multiple-choice questions that focus on symptomatology in the field of general pathology.

The previous functionalities have been fully implemented in our rule-augmented AI-empowered system and are presented in the remaining sections of the paper. Overall, our system is characterized by the incorporation of ML tools that simulate several of the common tools used by a general practitioner in an initial physical examination. Enhanced with these functionalities, the current version of our system provides medical assistance, closely replicating the behavior, objectives, tasks, and tools of a general practitioner when offering diagnostic recommendations to primary care patients.

More specifically, the paper is organized as follows. Section 2 is devoted to highlighting background theories and context with regard to both LLMs and the state of primary care worldwide. Section 3 presents an overview of the developed rule-augmented AI-empowered system. Section 4 details the system architecture from a micro and a macro level. Section 5 includes a system evaluation and Section 6 summarizes the paper, articulates and presents its key findings, and offers insights on future related research endeavours.

2. Background Theories and Context

In this section, our focus is to establish a comprehensive background pertinent to the methodologies used. We delve into various aspects of NLP and explore its diverse applications within the medical sphere. Notably, the integration of NLP and LLMs in healthcare has been significant [18]. These technologies are increasingly employed for a range of purposes, including the extraction of vital data from electronic health records (EHR), supporting decision making in clinical settings, and analyzing patient sentiments through their reviews and feedback [19,20].

2.1. Natural Language Processing

NLP, a pivotal AI sub-field, focuses on the interaction and interpretation of human language by computers. It facilitates various tasks, including translation, sentiment analysis, and conversational interfaces. The evolution of NLP spans from rule-based approaches to sophisticated ML techniques, giving rise to advanced models such as GPT and BERT [12,21,22]. Essential concepts in NLP encompass tokenization, part-of-speech tagging, named entity recognition, and parsing. The overarching aim is the effective comprehension of human language, facilitating the extraction of meaning and simulation of reasoning to accomplish specific tasks. NLP employs an array of techniques and models, ranging from rule-based systems to advanced ML algorithms. Prominent models in NLP include the following.

- NLP using pattern matching and substitution: These initial NLP systems depend on manually crafted rules and lexicons. An iconic example is the ELIZA chatbot [23], created in 1964. ELIZA was one of the first programs capable of attempting the Turing test.
- ML models: This category encompasses traditional models like naive Bayes, support vector machines (SVM), and decision trees, commonly applied in text classification and sentiment analysis.
- Neural networks: Inspired by the human brain, these models include recurrent neural networks (RNNs) and convolutional neural networks (CNNs), suitable for tasks needing an understanding of a language's sequential nature.
- Embedding models: These models produce dense vector representations of words or larger text units, capturing semantic meanings. Notable examples include Word2Vec, GloVe, and FastText [24].
- Sequence-to-sequence models: Capable of transforming input sequences into output sequences, these models are integral to machine translation and text summarization, often based on an encoder–decoder architecture with attention mechanisms [25].
- Large language models (LLMs): LLMs are designed to perform a wide range of NLP tasks, from translation to question answering and to text generation, without needing task-specific training data. LLMs are further discussed in the following section.

2.2. Large Language Models (LLMs)

The GPT series, including GPT-3 and GPT-4, comprises autoregressive models known for generating contextually coherent text. GPT decodes the received input, using language pattern understanding, to produce a relevant and coherent output. GPT models are especially powerful for applications like content creation, dialogue generation, and tasks that require the production of new text based on given prompts.

In contrast, BERT operates by analyzing both preceding and succeeding words in a sentence, thereby enriching its understanding of the sentence context. Both models are built upon the Transformer architecture, first introduced in [22], which employs an "attention" mechanism to assign varying significance to different words. Central to these models is an encoder, which converts sequences of words into contextually enriched vector representations. The novel self-attention mechanism in these models allows them to consider the inter-dependencies of words over longer ranges, significantly improving their predictive accuracy. Notably, BERT employs a bidirectional training approach, enabling word prediction based on both the preceding and subsequent context. This is in contrast to GPT's unidirectional methodology.

Prior to the widespread adoption of neural networks and Transformer models in NLP, statistical models were the mainstay. Key among these were the following.

- Markov Models: Based on the principle named after mathematician Andrey Markov, these probabilistic models assume that the probability of each subsequent state depends only on the current state. Their application is particularly notable in sequential tasks like language modeling.
- Hidden Markov Models (HMMs): An extension of Markov models, HMMs include hidden states and observable outputs. They find applications in NLP tasks, notably in part-of-speech tagging and named entity recognition.
- Conditional Random Fields (CRFs): These are statistical frameworks used in NLP to model the probability of outputs given specific inputs. Unlike HMMs, CRFs take into account the entire sequence of words, thereby yielding more accurate results.
- n-gram Models: These models predict the next item in a sequence by considering the previous $(n - 1)$ items. Predicated on the assumption that a word's probability is dependent solely on its preceding words, n-gram models are prevalent in areas like speech recognition and machine translation.
- Latent Dirichlet Allocation (LDA) is a generative statistical model that allows sets of observations to be explained by unobserved groups. In NLP, these groups or topics help us to understand why data parts are similar, positing each document as a topic mixture with each word attributed to a document's topic.

2.3. Problems with NLP and Evaluation Pipeline

Below, we list some important problems and concerns associated with NLP, especially when employed in the medical domain. Some of these are currently being addressed by the companies that commercially provide the state-of-the-art models.

- Hallucinations: Generation of outputs that seem plausible but are entirely fabricated or inaccurate [26,27].
- Bias: LLMs learn and reproduce the biases that exist in their training datasets [28].
- Lack of explainability: Generative AI systems typically do not provide explicit explanations for the conclusions that they reach or the answers that they provide [29,30]. Explainable AI (XAI) ensures that users comprehend the characteristics of the utilized models and provide a transparent representation of the used algorithms that generate a response, a classification, or a recommendation. Considering user ability and adding personalization in XAI is also an important factor that can increase transparency and lead to the greater adoption of AI-empowered systems [31]. Current GAI systems lack explainability, particularly in terms of personalized explanations.

- Real-time validation: The responses are not derived from real-time information. Instead, they are based on the dataset that was used to train the model that typically contains information from a period up to the date of the training of the tool [27].
- Limitations in mathematical operations: This limitation is partially addressed using Python modules for calculations and by providing updated models more frequently.
- Content—token size limitation: This limitation is partially addressed by increasing the token size limits and charging higher usage costs.

In previous works [15,32], we proposed a methodology to evaluate the domain-specific proficiency of ChatGPT or other LLMs, focusing on reliability and precision. These metrics are based on the context of the answers, their accuracy, and the quality of the references used. Our approach utilizes a three-tiered scoring scale (1–3) to assess various aspects, categorizing the context, references, and value added to the system as follows:

- correct (3), generic (2), or incorrect (1);
- actionable (3), generic (2), or non-actionable (1);
- precise (3), generic (2), or misleading (1);
- under-extended (2), exactly aligned (1), or over-extended (1).

The evaluation specifically focuses on (A) the validity and accuracy of answers as per the context and references returned in the LLM response, (B) the specificity and usefulness of the LLM-generated response to physicians and patients alike, and (C) the economic value (potentially) added to the system. The entire assessment process is overseen by a medical professional and can be seen in Figure 2.

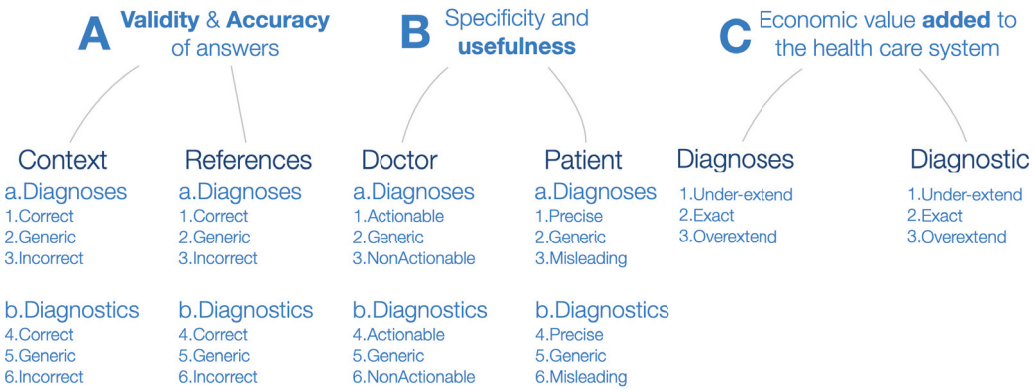


Figure 2. Methodology for evaluation of the domain-specific proficiency of ChatGPT.

2.4. Transformers and Attention Mechanism

The Transformer model has been very influential in the field of NLP and constitutes the engine of the state-of-the-art LLMs, still powering the latest ChatGPT engine as of the latest update of November 2023. In Figure 3, the architecture of a Transformer is presented, along with a description of each step and a brief explanation of the related mathematical formulae. This section contextualizes our study within the broader scope of NLP progress, but also provides a necessary technical foundation for the analysis and development of further innovations in the generative artificial intelligence (GAI) space.

The Transformer model is based on a mechanism, referred to as self-attention, that directly models the relationships between words in a sentence, regardless of their respective positions in the sentence.

- Encoder: The left part of the diagram represents the encoder, which processes the input data. The input sequence is processed through multiple layers of multi-head attention and feed-forward networks, with each of these layers followed by the residual connection and linear normalization steps.
  - Embeddings: The numerical representations of words, phrases, or other types of data. In the case of LLMs, they represent words or tokens. Each word or

token is mapped to a vector of real numbers that captures semantic and syntactic information about the word. The words with similar meanings or used in similar contexts will have similar vector representations.

  * Input Sequence Embedding: Input tokens' conversion into vectors of a fixed dimension.

– Positional Encoding: Adds information about the positional order of the respective words of the sequence.
– Multi-Head Attention: Applies self-attention multiple times in parallel to capture different aspects of the data. This allows joint attention to information from different representation subspaces, referred to as heads, at different positions. Using multiple heads, the model captures different types of dependencies from different representational spaces. For example, one head might learn to pay attention to syntactic dependencies, while another might learn semantic dependencies. The mathematical representation of this is as follows. Let us we denote the linear transformations that produce the queries, keys, and values for head i with W_i^Q, W_i^K, W_i^V, respectively, and the output linear transformation with W^O. Then, the multi-head attention operation MultiHead can be defined as

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O,$$

where each head head_i is computed as

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

and Attention is the scaled dot-product attention function:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V.$$

Here, d_k is the dimensionality of the key vectors, while the division by $\sqrt{d_k}$ is the scaling factor.

1. *Q, K, V*: The input to the multi-head attention layer is first linearly transformed into three different sets of vectors: queries (Q), keys (K), and values (V). This is done for each attention head using different, learned linear projections.
2. Scaled dot-product attention: For each head, the scaled dot-product attention is independently calculated. The dot product is computed between each query and all keys, which results in a score that represents how much focus to place on other parts of the input for each word. These scores are scaled down by the dimensionality of the keys (typically the square root of the key dimension) to stabilize the gradients during training. A softmax function is applied to the scaled scores to obtain the weights on the values.
3. Attention output: The softmax weights are then used to create a weighted sum of the value vectors. This results in an output vector for each head that is a combination of the input values, weighted by their relevance to each query.
4. Concatenation: The output vectors from all heads are concatenated. Since each head may learn to attend to different features, concatenating them combines the different learned representation subspaces.
5. Linear transformation: The concatenated output undergoes a linear transformation to produce the final product of the multi-head attention layer.
6. Feed-forward network: A fully connected feed-forward network is applied to each position separately and identically.

7. Residual connection and linear normalization: Applies residual connections and layer normalization.

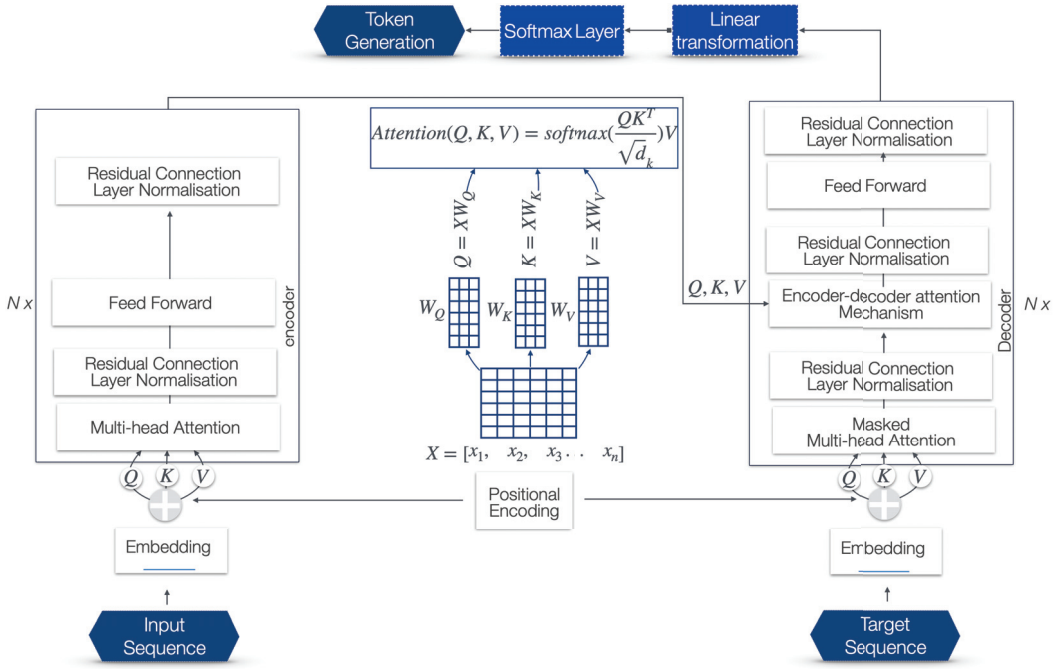


Figure 3. Transformer—data flow chart.

- Decoder: The right part of the diagram represents the decoder that generates the output.
  - Target sequence embedding: Converts target tokens into vectors and shifts them to the right.
  - Masked multi-head attention: Prevents positions from attending to subsequent positions during training.
  - Encoder–decoder attention mechanism: Attends to the encoder's output and the decoder's input. The keys (K) and values (V) come from the output of the encoder. The similarity between the queries and keys is calculated. This involves taking the dot product of the queries with the keys, scaling it (usually by dividing by the square root of the dimension of the key vectors), and then applying a softmax function to obtain the weights for the values.
  - Feed-forward network: Following the attention mechanisms, there is a feed-forward network. It consists of two linear transformations with a ReLU activation in between.
  - Residual connection and linear normalization: Applies residual connections and layer normalization.
  - Linear transformation before softmax: In the final layer of the decoder, the Transformer model applies a linear transformation to the output of the previous layer. This linear transformation, typically a fully connected neural network layer (often referred to as a dense layer), projects the decoder's output to a space whose dimensionality is equal to the size of the vocabulary.
  - Softmax function: After this linear transformation, a softmax function is applied to these projected values, which creates a probability distribution over the vocabulary based on the positional attributes.
  - Token selection: The probability distribution for each potential token is analyzed considering the context of the sequence. This analysis determines which tokens are most likely to be the appropriate next elements in the sequence. The token

selection can be done using various strategies like greedy decoding, sampling, or beam search.
- Token generation: Based on this probability distribution, tokens are generated as the output for each position in the sequence.
- Sequence construction: The selected tokens are combined to form the output text sequence. This can involve converting sub-word tokens back into words and dealing with special tokens such as those that represent the start and end of a sentence.
- Post-processing: Post-processing is performed, based on syntactical, grammatical, and language rules.

This architecture is highly parallelizable and reduces the need for recurrent or convolutional layers. Further details of this architecture are analyzed in [22]. By providing a previous basic overview of how a Transformer works, we can show how the methodology that we use provides an efficient shortcut to creating our domain-specific system's engine.

2.5. Telehealth

Telehealth uses digital technologies to remotely deliver selected healthcare services. Telehealth's importance is prominent where resources are limited and the goal of healthcare cost reduction is important. The continuous monitoring of treatment is another important domain of application of telehealth. It also aims to educate patients and healthcare providers, support consultations between primary care providers and specialists for quicker diagnoses and treatment, more effectively manage hospital patient loads, and more actively engage patients in their own care [33,34].

Telehealth has been widely adopted across various medical specialties due to its versatility. Primary care can handle routine check-ups and minor health issues remotely, while psychiatry and psychology benefit from teletherapy and telepsychiatry. Similarly, radiology allows for the remote sharing and analysis of medical images, and cardiology and neurology utilize remote monitoring for conditions like heart rhythm abnormalities and epilepsy. Dermatology and endocrinology practices can remotely diagnose and manage skin conditions and diseases like diabetes. Geriatrics and pediatrics are also benefiting from telehealth, especially for patients with mobility issues or for the management of minor concerns and follow-ups. Chronic disease management, including hypertension, COPD, and asthma, is another area where telehealth plays a crucial role.

As technology advances, more medical specialties are incorporating telehealth into their practices. While physical examinations remain a limitation, many conditions can be diagnosed and treated effectively using a combination of the patient history, visual examination, and remotely collected data. However, the above areas do not constitute an exhaustive list and the potential for telehealth continues to grow.

2.6. The State of Primary Care

In many healthcare systems worldwide, including those in Europe and the United States, costs and delays [35] are significant issues, although their intensity and nature vary by region. In the European Union, most countries have universal healthcare systems funded through taxation or mandatory health insurance, with some variations in patient costs and the option of private insurance. While the primary care quality is generally high, there are differences between and within countries, with concerns often centered around waiting times for specialty care.

In contrast, the USA operates mainly on an insurance-based system, with many facing co-payments, deductibles, and other out-of-pocket costs. This can deter some from seeking primary care, and the quality of care varies by factors like location and socioeconomic status. Debates about the benefits and drawbacks of moving towards a universal healthcare system are still ongoing [36].

Both the European Union and the USA face challenges in primary care accessibility and preventable mortality. Over 100,000 deaths annually in the USA may be due to preventable

medical errors, including access failures. Similarly, thousands may be dying annually in the European Union due to preventable causes. Treatable mortality [37,38] rates are published yearly both in Europe and the USA. In 2020, the treatable mortality rates accounted for 39 deaths per 100,000 population in Switzerland (lowest) and for 225 deaths per 100,000 population in Hungary (highest) [39]. These challenges are intertwined with issues like quality of care, equity of access, and patient satisfaction, and each region addresses them based on its specific healthcare models and cultural values.

Lastly, many developing countries have limited budgets for healthcare, which can lead to inadequate infrastructures, low salaries for healthcare workers, and insufficient medical supplies.

2.7. NuhealhtSoft: An AI-Empowered Software Platform for Medical Exam Classification and Health Recommendations

NuhealthSoft is a recently developed software platform to facilitate users in understanding their blood exams using various presentation and analytical techniques, including semantic grouping. The system also provides its users with services that identify patterns and, thus, health states in their blood exams and dietary intake. In more detail, NuhealthSoft [40] employs advanced ML techniques combined with comprehensive nutritional and biochemical data. This approach enables the platform to effectively categorize individuals based on various health metrics. These metrics include blood pressure, weight, and indicators of metabolic syndrome, as supported by several studies and references [10]. Additionally, NuhealthSoft moves beyond mere classification by offering personalized nutritional advice. This guidance is specifically tailored to meet the unique dietary needs and blood work results of each user.

The development and refinement of NuhealthSoft necessitate close collaboration with medical professionals and regulatory bodies. These stakeholders play a crucial role in validating the system's effectiveness and compliance with health standards. This collaboration has been a pivotal aspect of our research. We have focused on understanding and integrating the specific requirements of doctors to facilitate a seamless validation process. Simultaneously, we aim at maintaining a transparent and user-friendly framework for the end-users of NuhealthSoft. This dual focus ensures that the system is not only medically sound and compliant, but also accessible and beneficial to those whom it serves.

In this paper, Med | Primary AI assistant is presented, which has been developed as an an add-on for the NuhealthSoft suite. Specifically, Med | Primary AI assistant has been developed to analyze symptoms and provide health advice from a general practitioner's perspective.

3. System Overview of Med | Primary AI Assistant

In Figure 4, the objectives, tasks, and available tools are outlined. The purpose is to build an AI-empowered system that can perform these objectives and complete the tasks with use of the available tools. In essence, Figure 4 provides the blueprint of the domain-specific knowledge of the system. To ensure that the shortcomings of LLMs are addressed, the system also encompasses rules, i.e., a rule-augmented application is developed. The rules are used to

- engineer prompts based on domain specification;
- extract semantically important words and associated with external services and classifiers and external sensors; and
- create an evaluation basis, to ensure alignment with domain specifications and requirements based on the dialogue's theoretical context.

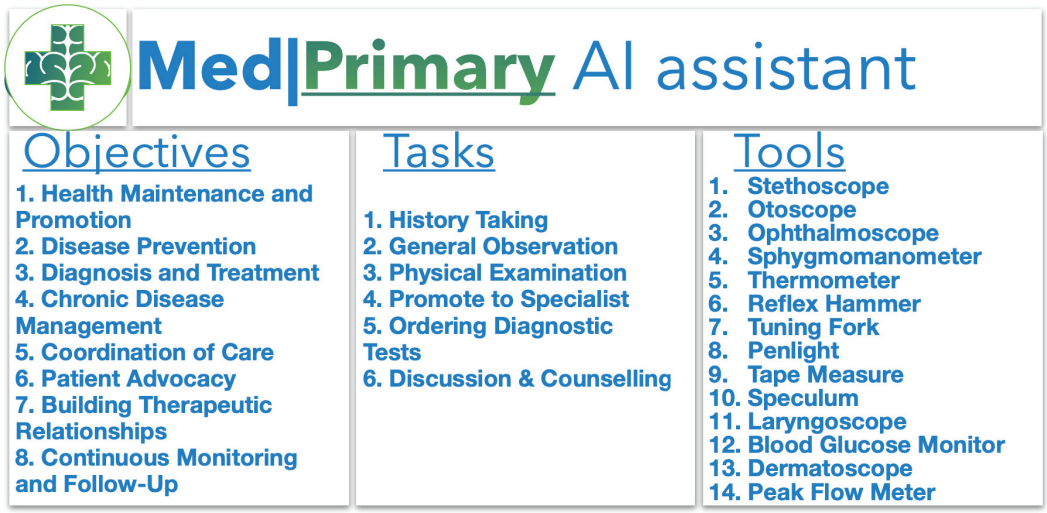


Figure 4. General practitioner's objectives, tasks and tools.

3.1. System Description

Med | Primary AI assistant, included in the NuhealthSoft suite, utilizes LLMs; for the purpose of this study, we have used and tested the GPT-4 model.

GPT-4 is an advanced multimodal model, currently processing text inputs and producing text outputs and chat completion tasks. It outperforms previous models with its extensive general knowledge and enhanced reasoning skills. While it shares similarities with GPT-3.5-turbo in being optimized for chat interactions, it is also proficient in executing traditional completion tasks.

Our system also encompasses analytical services and ML models to extract useful information from health data, while only providing the necessary information. The limiting of token usage maintains a manageable input and also ensures the computational and mathematical validity of the provided information, thus augmenting the quality of the response.

3.2. Use Cases

In Med | Primary AI assistant, a user can interact with the system in two main ways.

- The first is by freely (without constraints and rules) providing symptoms and descriptions of their health state and obtaining a series of diagnoses, proposed diagnostic exams, or a referral to a medical specialist. While, in this case, the patient has no constraints, using specific knowledge input, the LLM will provide assistance if the user's input is not useful for the LLM to complete its predefined tasks and objectives.
- The second is by using a more constrained and step-by-step approach, for the LLM to obtain a more comprehensive background on the user's symptomatology and age. In both cases, data can be retrieved by health sensors and analyzed by the included analytical and machine learning services [41].

In Figure 5, two main actors are presented. The first actor is the patient, who will provide the symptoms directly to Med | Primary AI assistant or via form inputs. Health sensors can provide more context and data. Finally, the patient can review the process and output. The doctor, as the second actor, can evaluate and validate the primary care AI interactions (inputs–outputs) and the patient's review of the the primary care AI. This process is essential for the system to improve and for more services to address primary care AI shortcomings.

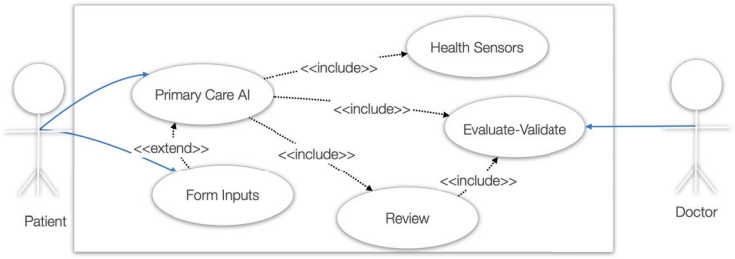


Figure 5. Med | Primary AI assistant use case.

4. System Architecture Analysis

In this section, the structural elements, flow of data, and organization of Med | Primary AI assistant are outlined and described. As shown in Figure 4, the specific objectives, tasks, and tools are considered as building blocks. For example, if a service is provided to facilitate the process of diagnostic analysis, it will only belong in the domain of a general practitioner's competencies and, thus, constructs the blueprint of domain-specific knowledge of the system.

4.1. Modeling the Domain Space

The main sources of the system pertain to the management of inputs and outputs during a conversation between a patient and a general practitioner, specifically addressing questions and answers [42–44].

For context, there are six (6) types of theoretical questions and answers that can be applied in any domain.

1. Questions
   - (a) Informational: Query for specific information.
   - (b) Instructional: Query related to specific task, i.e., a command to do something.
   - (c) Reflective: To confirm or clarify previous statements.
   - (d) Rhetorical: Are not meant to be answered and are rather used for emphasis.
   - (e) Open-ended: Are meant to encourage a detailed response or discussion.
   - (f) Closed-ended: Can be answered with a a yes or no.
2. Answers
   - (a) Direct: Provide a straightforward response.
   - (b) Elaborated: Provide additional context and information beyond what was requested.
   - (c) Clarifying: Aim at requiring clarity, where a query is ambiguous.
   - (d) Reflected: Ensure that the question is answered in a way that mimics the question's sentiment.
   - (e) Deferred: When an answer cannot be provided and the one that provides it offers guidance on where or how to find it.
   - (f) Non-Answers: When the choice is to not answer.

4.2. Domain Settings

Questions and answers are the building blocks of the conversations and the input and output of the flow chart pictured in Figure 6. A conversation in the medical domain usually includes informational and open-ended questions, followed by direct or elaborated answers. For the case of this system (Table 1), a deferred answer is also an option when the question cannot be answered, either due to the fact that it requires a more specialized evaluation (i.e., a referral to a specialist) or when it is outside the scope of the domain (dialogue reset). Dialogue rules are set to

- provide a basis for evaluating the performance;
- assess the model's ability to remain within constraints that aim to simulate real-time communication protocols.

Table 1. Dialogue rule augmentation.

Question (q)	Domain (d)	Medical (m)	Answer (a)	Answer Content (c)	Use Case Examples
Open Ended	Yes	Yes	Direct or Elaborated	Define Ability (Ab), Reflect (Re)	Figure 7
Informational	Yes	Yes	Direct or Elaborated	Define Ability (Ab), Reflect (Re)	Figure 8
Any	No	Yes	(deferred) Refer To Specialist	Define Ability (Ab), Reflect (Re)	Figure 9
Any	No	No	(deferred) Dialogue Reset	Define Ability (Ab), Reflect (Re)	Figure 10

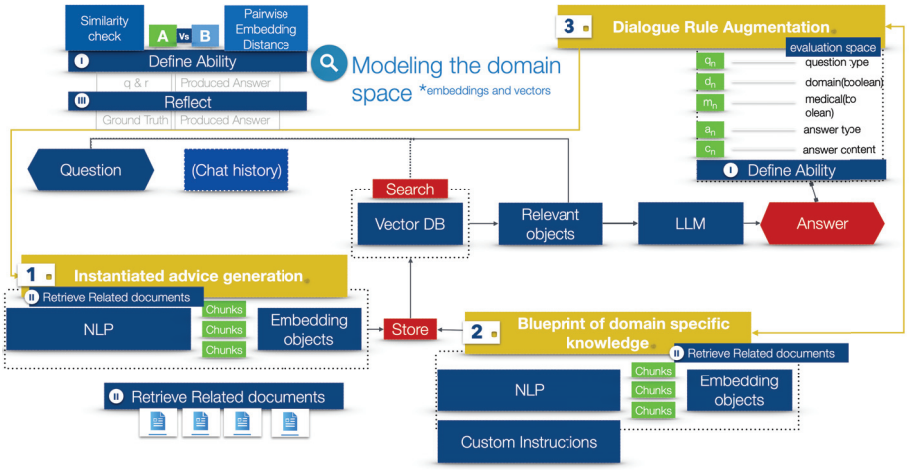


Figure 6. Domain settings, flow chart.

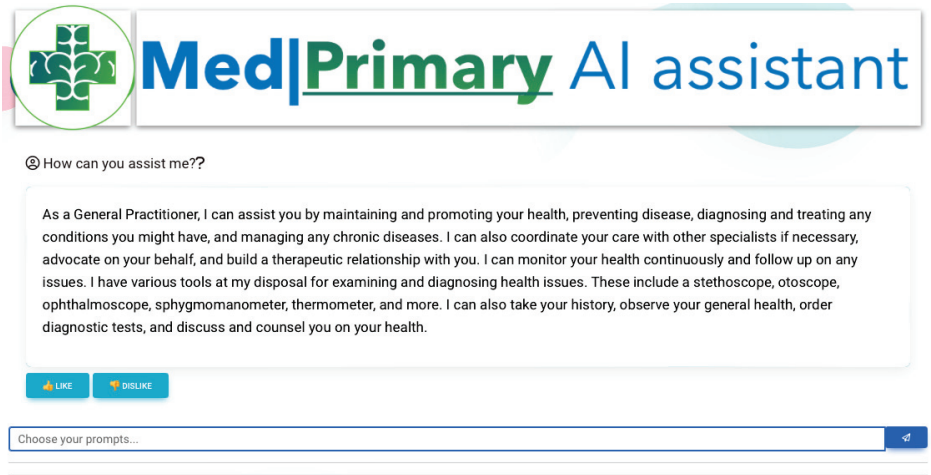


Figure 7. Open-ended question—direct answer, system objectives, and tasks.

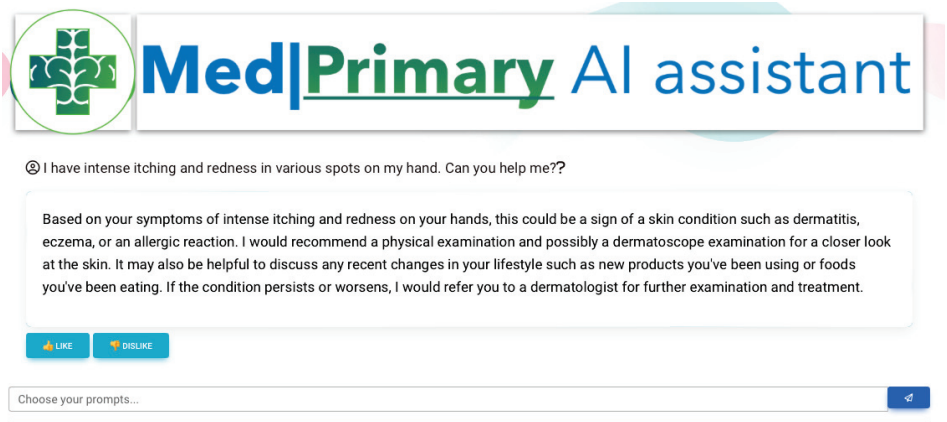


Figure 8. Informational question—elaborated answer, skin problem with referral.

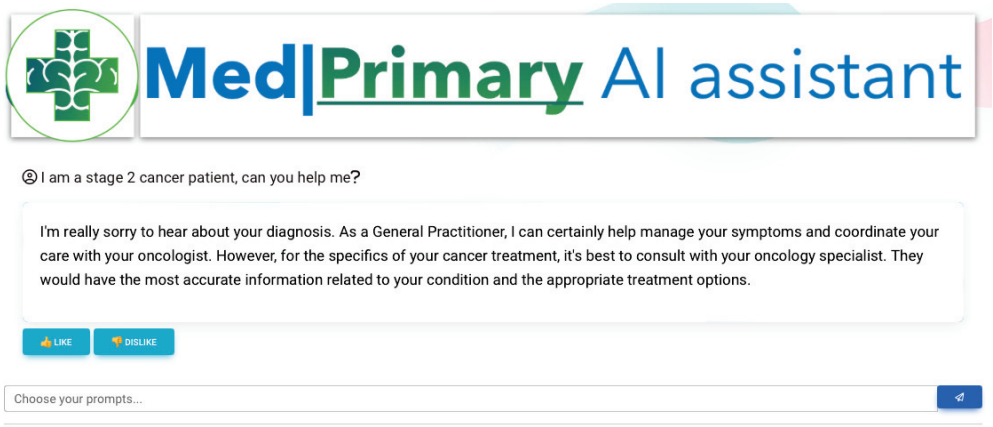


Figure 9. Open-ended question—deferred answer, cancer patient.

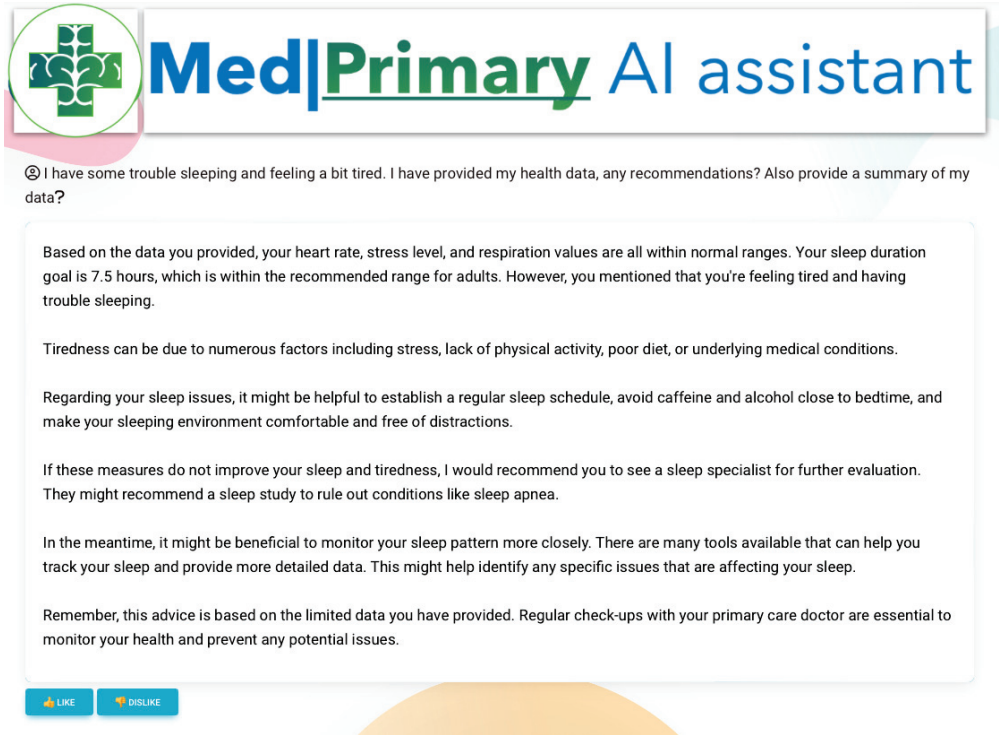


Figure 10. Informational question—elaborated answer, symptoms with health metrics.

The instantiated advice generation and blueprint of domain-specific knowledge encourage the structuring of informational and open-ended questions. Rules are also defined to lead to a certain type of answer that ensures a diagnosis, the promotion of necessary diagnostics, or the proposal of a medical specialist, mostly associated with the described symptoms.

To ensure the safety of users, the Dialogue Rule Augmentation stage (as shown in Table 1) establishes the framework for evaluating the extracted answers, as depicted in Figure 6. This evaluation is conducted through the 'Define Ability Process (I)' and the 'Reflect Process (III)', which are incorporated as functions in Algorithm 1. Notably, in the medical domain, professors evaluate students across a spectrum of themes and real-time scenarios. These evaluations include the process of patient interaction, as well as the assessment of symptoms and the subsequent course of action, as referenced in [45,46]. Here, we systematize and generalize the process to reach to a measurable conclusion of LLMs answers, within a domain-specific context and a dialogue-defined space. Our

contributions in the generalization of the domain settings using embeddings can be seen in Figure 6 within the yellow cards.

Each component that defines the answer is represented by the appropriate letter in Table 1 and Figure 6. In a range of (n), a final score would finalize a decision as per the quality of the answer, which in essence represents the ability (A) of the LLM to comprehend the answer and also the effectiveness of the domain setup (Figure 6). In more detail, the items are as follows.

- Question q.
- Domain d.
- Medical m.
- Answer a.
- Answer content c: Derived from the 'instantiated advice generation' (2) and the 'blueprint of domain-specific knowledge' (1).
- Retrieve documents r: Sourced from (1) and (2) to show basis of produced answer.
- Process I (ability assessment): To define ability, we compare the answer produced by the LLM (a) to the ground truth (i.e., the correct answer) using similarity checks and pairwise embedding distance algorithms. This involves retrieving the documents (r) on which the answer was based.
- Process III (reflective capacity): The reflective capacity is calculated by applying similarity checks and pairwise embedding distance algorithms between a composite of the question (q) and answer type (a) and the answer content (c) produced by the LLM.

Algorithm 1 Evaluation process based on rules.

Require: Question (q), Domain (d), Medical (m)
Ensure: Answer (a), Grade of Answer Content (c)

```
Begin
if (q == "Informational" OR q == "Open-Ended") AND (d == "Yes") AND (m == "Yes") then
    a ← "Direct or Elaborated"
else if (q == "Any") AND (d == "No") then
    if (m == "No") then
        a ← "(deferred) Dialogue Reset"
    else if (m == "Yes") then
        a ← "(deferred) Refer To Specialist"
    end if
end if
c ← AnswerContent
r ← RetrieveDocuments
grade(Ab) ← Define Ability(c)
grade(Re) ← Reflect(c)
return a, grade
End
function DefineAbility(AnswerContent)
    // A grading logic for Answer content(1)
    // Return grade(Ab)
end function
function Reflect(AnswerContent,documents)
    // A reflection(2) logic for answer content and document retrieval(3)
    // Return grade(Re)
end function
```

For Process I (Define Ability),

$$\text{Ability} = \text{Similarity}(\text{LLM Answer}, \text{Ground Truth})$$

where

$$\text{LLM Answer} = \text{Function}(a, c, r)$$

$$\text{Ground Truth} = \text{Known Correct Answer}$$

For Process II (Reflect -reflective capacity),

$$\text{Reflective Capacity} = \text{Similarity}((q + a), c)$$

Here, "Similarity" represents the similarity check and pairwise embedding distance algorithms, and "Function" is the method by which the LLM produces its answer based on the answer type (a), answer content (c), and retrieved documents (r).

The main assumption made for the creation of embeddings in GAI, instead of a bottom-up approach when fine tuning or recreating a model, is based on the fact that the trained LLMs used in systems like Bard (google) and ChatGPT (openAI) are tested and evaluated in numerous tasks. Moreover, the magnitude of their training datasets is such that recreating a similar model would incur additional costs and evaluation procedures. While the fine tuning is a more straightforward strategy than recreating one, again, there are significant costs and similar evaluation requirements.

Retrieval-augmented generation (RAG) is particularly effective for general tasks as it combines the benefits of a large language model with external data sources, enhancing the breadth and specificity of its responses. This approach is suitable for a wide range of applications where expert involvement is not critical and the focus is on augmenting the generative capabilities with a diverse set of information sources.

On the other hand, GAI, particularly in sensitive areas, requires a more nuanced approach. In scenarios such as healthcare, legal advice, or personalized recommendations, the requirements are higher. Therefore, employing GAI in these domains demands rigorous testing, diligent constraint implementation, and continuous monitoring to ensure safety, accuracy, and ethical compliance. The involvement of domain experts becomes crucial for the validation of the outputs and provision of guidance on the model's usage boundaries. This ensures that the generated responses and decisions are not merely based on data and algorithms but are also aligned with human expertise and ethical standards.

The creation of embeddings offers a rapid method of incorporating essential context into pre-trained LLMs, which becomes particularly effective when these models are employed in specific applications. Additionally, we ensure that all supplementary data utilized by the model are provided by medical experts and align with the guidelines set forth by the relevant medical boards. The need for model fine tuning is determined based on the outcomes observed. If necessary, this fine tuning can occur later in the release and production pipeline, after a thorough evaluation tailored to the specific domain requirements and specifications. When the embeddings are input into a trained Transformer model, particularly for conversational purposes, the model utilizes the weights (as shown in Figure 3) that were acquired during its training phase. This process enables the model to more swiftly adapt to a predetermined conversational context [47].

The process of creating and training embeddings typically involves several key steps, outlined below, to ensure reproducibility.

1. Creation of embeddings
    (a) Training: Embeddings are usually created through supervised or unsupervised learning on large text corpora. At this stage, the model's trained weights are used to create vectors.
    (b) Dimensionality: The vectors usually have hundreds of dimensions. Dimensionality reduction techniques (like PCA or t-SNE) can be applied for visualization.
    (c) Contextualization: Traditional embeddings (Word2Vec, GloVe) do not consider the context, meaning that they represent a word with the same vector regardless of its usage. Modern embeddings (BERT, GPT) are contextual, adjusting the representation based on the word's usage in a sentence.

(d) Transfer Learning: Pre-trained embeddings can be fine-tuned on a smaller dataset for specific tasks, leveraging the general language understanding learned during pre-training while adapting to the nuances of the task at hand.
(e) Evaluation: The quality of embeddings is usually evaluated based on their performance in downstream NLP tasks like text classification, sentiment analysis, or named entity recognition.

Using the created embeddings, the following processes are the splitting, chunking, and storage of the information in vector databases, which would either define the instantiated advice generation or the blueprint of domain-specific knowledge.

2. Vector stores are databases that specialize in storing, indexing, and querying high-dimensional vectors. These vectors can represent various types of data, such as images, text, or other complex data types, transformed into numerical representations [48]. They are extremely useful for a similarity search, which, in Figure 6, is the red rectangle named search, pointing to the vector database.
3. A search is the process of retrieving documents stored in the vector store, either from the instantiated advice generation or the blueprint of domain-specific knowledge, based on a similarity threshold, manually defined. The higher the similarity threshold, the more restrictive the rules; thus, less documents are returned for processing.

In the Q&A chain, the aforementioned process is outlined as a generic algorithm in Algorithm 2.

Algorithm 2 Q&A Chain

1: $DB, Embeddings \leftarrow$ Vector.db() (Vector Database Settings)
2: $Retriever, LLM \leftarrow$ chainer$(DB, Embeddings)$ (LLM engine properties)
3: **procedure** RUNQACHAIN($Query : QueryModel, CurrentUser$) (*Function)
4: $Question \leftarrow Query.question$
5: $QAChain \leftarrow$ prompter$(Retriever, LLM)$ (Retriever== Similarity parameters and search depth, LLM== llm engine parameters)
6: $Result \leftarrow$ responder$(QAChain, Question)$
7: **return** {"response" : $Result$}
8: **end procedure**

Step 1 of the algorithm involves initializing or loading a vector database. The database contains embeddings, which are high-dimensional vectors of keys and values, representing the documents and instructions provided by the user. Step 2 involves the binding of components to process queries. This involves setting up an embedding filter, i.e., similarity search parameters; a retriever, where the properties to be retrieved from the DB are set; and an LLM. The exact roles of these components depend on the specific implementation and use case. In Step 3, the procedure is initialized, based on a received question that is posed by a specific user. Lastly, a response is returned from the model as a result, usually in a json or xml format.

4.3. System Architecture

The system architecture is outlined in Figures 11 and 12. In the micro-level process diagram, the internal design and communication paths of the different modules are analyzed. In the macro-level diagram, the application's overall structure is detailed.

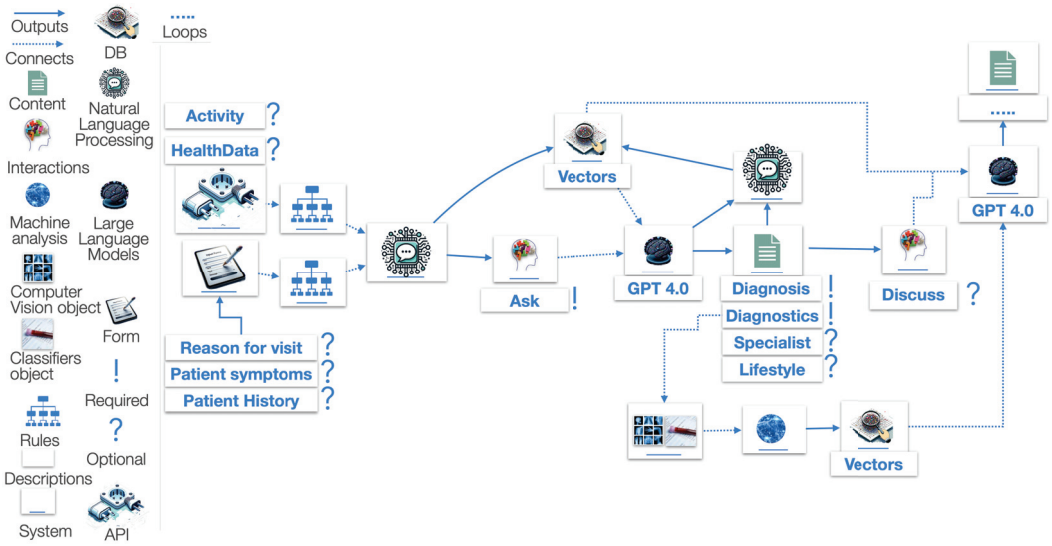


Figure 11. Process diagram—micro level.

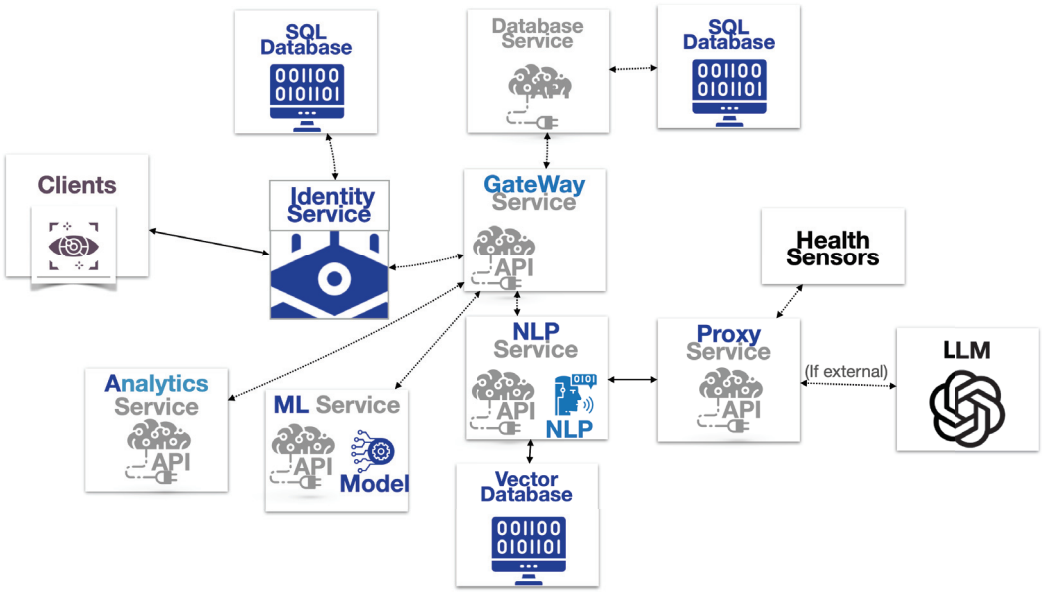


Figure 12. System architecture—macro level.

4.3.1. Micro Level

In the process diagram, we introduce a novel methodology to define GAI processes and functions, in the scope of a rule-augmented approach. On the left side, the different components are noted, and, on the right side, these components are implemented and we describe the information flow, the constraints, and the expected outcomes. In detail, the items are as follows.

- Outputs: Information exchange.
- Connects: Implies dependency.
- DB: Type of database procedure.
- Content: Generation of data.
- Interactions: User interaction, which leads to a generation.
- Natural Language Processing: Any NLP process.
- Large Language Model: LLM processing or LLM API call.
- Computer Vision Object: ML process related to computer vision.
- Classifier Object: ML process related to classification.
- Machine Analysis: ML service output in textual or numeric format.

- Rules: Rules that are used to augment the system and limit malfunctions.
- White Box: Description space for processes.
- White Box with Blue Line: Defines a function or system.
- Form: A type of user input form, in a predefined context, i.e., using questionnaires or pre-selected inputs.
- !: Required described process.
- ?: Optional described process.
- API: External communication process.
- Three Dots (. . .): Indicates a loop or a repetitive–iterative process.

The process starts with the input of the optional definition of reasons for the (doctor) visit or/and patient symptoms and an optional patient history. In parallel, the user can optionally upload or connect activity and health data via an API. An example of the data format extracted via the API can be seen in the Garmin Health snapshot schema. Based on specific rules, the data are then fed into the NLP system, which outputs an engineered prompt and an embedding to be saved in the Vectors DB system. The engineered prompt is the Ask Required interaction, which, alongside the Vectors DB, provides the necessary information for the LLM system (powered here by GPT-4), to provide a required diagnosis, the required diagnostics, and an optional referral to a specialist or an optional lifestyle intervention strategy. For the diagnostics, the user is also provided with computer vision systems and classifiers, for examination analysis and the transposition of data into usable objects. These objects are again saved into embedding objects in the Vectors DB system. The Discuss output is optional and can lead to a Q&A, as previously described in the Theoretical Dialogue section.

4.3.2. Macro Level

In Figure 12, the microservice architecture that can optimally support the rule-augmented AI application is detailed. This approach encompasses the overall structure of the entire software system, including how the different modules and components interact. In this specific application, which enables a range of analytical services and external APIs, a microservice architecture allows each service to be deployed using the most appropriate infrastructure. At the same time, the independent testing of each AI service or module facilitates an easier-to-manage workflow.

Especially when dealing with Python AI libraries, one common issue is the varying dependencies and requirements that they may have. These libraries often rely on specific versions of other libraries, which can lead to compatibility issues when multiple AI services are bundled together in a monolithic application.

The different components of the system architecture are as follows.

- Client: The user interface that provides the main space for the patients to interact.
- Identity Service: The authorization and authentication infrastructure that validates the client based on user profiles stored in the first SQL database.
- Gateway Service: Acts as an intermediary that processes and routes requests from clients to various services within a system.
  - Database service.
  - Analytics service: the component that is dedicated to analyzing data and generating insights, related to health metrics and diagnostic examinations.
  - ML service: a packaged component that provides machine learning capabilities to the system. This module can be integrated into existing systems to add features like prediction, classification, and anomaly detection based on extreme value theory.
  - NLP Service: the software module that encompasses the required processing features.
    - Vector database: type of database designed specifically to handle vectors, and, more specifically, in this case, the transposition to incoming embedding data.

- Proxy service: intermediary for requests from other APIs seeking resources from other servers. In this case, it handles communication with external health sensors and LLM engines.
  - Health sensors: APIs provided by wearable manufacturers like Fitbit or Apple HealthKit or medical devices used in clinical settings.
  - LLMs: APIs of powerful natural language processing engines for question and answering.

4.4. Services Simulating a General Practitioner

In this section, a brief description is provided for some key services that can facilitate the user when providing him/her with detailed diagnostic outcomes. These outcomes are to be processed using NLP techniques and provided back into the system, for the diagnosis pipeline to complete via one of the possible outcomes (diagnosis, diagnostics, specialist referral, lifestyle suggestion). It should also be stated that the services aim to simulate the available tools and competencies of a general practitioner. In this study, the focus is the simulation and automation of a physical examination using available technologies and incorporating them into an intuitive, fast, and simple-to-use rule-augmented system.

A blood examination is considered, alongside a view of a user's routine and a daily snapshot of the user's basic bio-metrics.

4.4.1. Blood Exam Analyzer

The blood exam analyzer consists of tools that extract information from the relevant examinations provided by the user for the construction of the second prompt. As part of this use case, the blood test analyzer and the blood exam classifier are utilized. This particular technology consists of a specific conditional logic that extracts those blood variables that are outside the normal ranges. Our system also consists of machine learning algorithms [4,49] that can identify similarities with specific weight groups, based on blood exams, and thus recognize other possible health states related to an unbalanced biochemical profile. Metabolic syndrome is also identified through a similar process [4,10].

4.4.2. External Sensors

Connecting with external sensors, the doctor, or, in this case, the AI system, an approximation to a physical examination can be achieved. Various health data can be extracted by health sensors with great accuracy, such as the following.

- Heart Rate: Continuous heart rate monitoring, including resting heart rate and abnormal heart rate alerts.
- Sleep: Tracks sleep patterns, including sleep stages (light, deep, REM) and sleep quality.
- Stress: Measures stress levels throughout the day.
- Steps and Floors Climbed: Tracks daily step count and floors climbed using an altimeter.
- Calories Burned: Estimates calories burned through various activities.
- Intensity Minutes: Tracks vigorous activity minutes as per health recommendations.
- Body Battery: Monitors body energy levels to suggest the best times for activity and rest.
- Pulse Oximetry: Measures blood oxygen saturation, which can be essential at high altitudes or for tracking sleep issues.
- Respiration Rate: Monitors breathing rate throughout the day and night.
- Women's Health: Tracks menstrual cycle or pregnancy.
- VO2 Max: Estimates the maximum volume of oxygen that can be utilized during intense exercise.
- GPS Tracking: Offers detailed tracking for outdoor activities, including pace, distance, and routes.
- Activity Profiles: Multiple sports profiles for tracking different activities like running, swimming, cycling, golfing, and more.

- Incident Detection: Some models offer incident detection during certain activities, which can send one's location to emergency contacts if a fall is detected.
- Mobility Metrics: Monitors how fast one walks, the timing of each step, and how often one stands up.

In the Garmin Health snapshot (see Listing 1), a json object, as an example of a useful retrieved health metric, is provided. In the external sensor algorithm, a summary of the processes of extraction and conversion of these data is provided, for replication purposes.

Listing 1. Garmin Health snapshot. Extracted and transformed into json file.

```
{'calendarDate': '2023-10-23',
    'minHeartRate': 50,
        'maxHeartRate': 131,
            'includesActivityData': True,
                    'restingHeartRate': 61,
                'averageStressLevel': 42,
            'bodyBatteryMostRecentValue': 18,
        'highestRespirationValue': 17.0,
    'lowestRespirationValue': 12.0,
'latestRespirationValue': 14.0}
```

Algorithm External Sensors

get_weekly_data: Collects data for the past 7 days using a provided data retrieval function.
daily_snapshot: Collects last data using a data retrieval function.

Pseudocode

```
function get_weekly_data(SensorData):
    start_date <- today - 7 days
    weekly_data <- empty~list

    for i in 0 to 6:
        current_Date <- start_date - i days
        data <- SensorData(current_Date)
        append data to~weekly_data

    return~weekly_data

function daily_snapshot(healthSensor, anarray):
    extracted_data <- empty~dictionary

    for each key in anarray:
        extracted_data[key] <- healthSensor.get(key, None)
    return extracted_data
```

4.5. Prototype—Use Cases

In this section, we present a series of screenshots, where different conversational use cases are considered. The general Med | Primary AI assistant is constructed following the methodology discussed in the previous section, where a blueprint of domain-specific knowledge has been constructed using embeddings. Moreover, instantiated advice generation is provided, in the form of embeddings, where the user can upload specific examination data using the provided services (Figure 12).

4.5.1. Use Case 1

In Figure 7, a user requests, in an open-ended question, the ways in which the assistant can provide help. The answer provided outlines the blueprint of domain-specific knowledge, which is analyzed in Figure 4. The tools, objectives, and tasks are described and returned as a direct answer.

4.5.2. Use Case 2

In the second use case (Figure 8), a user requests a consultation based on the described symptoms—an informational question. The system provides an elaborated answer, where some initial consultation is provided. A recommendation for a physical examination is also suggested and a provision for potential lifestyle changes and the use of products. This is an elaborated answer where the discussion can continue for a more definite diagnosis to be acquired.

4.5.3. Use Case 3

In this use case, shown in Figure 9, we show an example of a deferred answer, where the user is referred to a specialist (Table 1). Here, an open-ended question is provided that is within the medical domain but outside the blueprint of domain-specific knowledge. Thus, the system suggests a medical specialist, an oncologist, to better assess the related query.

4.5.4. Use Case 4

In this final use case, shown in Figure 10, we present the ways in which the data retrieved from external sensors are utilized. As already discussed in the previous sections, the data are transposed into embeddings and then analyzed, if necessary, by the system. The user provides some symptoms (information question –> elaborated answer) and states that data have been uploaded. Moreover, a summary of the data is requested. This descriptive prompt is designed in such a way as to best outline the system's capabilities. In a real-world scenario, since the blueprint of domain-specific knowledge is already created, the process would be more intuitive and only the symptoms would be required. The AI assistant would assess these symptoms and analyze the health data if necessary to provide a response.

5. System Evaluation

To effectively assess our system, we have utilized a selection of multiple-choice quiz questions sourced from 'The Internet Pathology Laboratory for Medical Education', an esteemed resource hosted by the University of Utah's Eccles Health Sciences Library [50]. These quizzes are meticulously designed to cater to students and professionals in healthcare sciences, with a particular focus on pathology. This selection is aligned with the specific educational needs and curricular requirements of medical students of pathology and practitioners.

More specifically, our system, which leverages the advanced capabilities of the GPT-4 model, has been tested across three thematic pillars of general pathology. These pillars encompass a comprehensive range of topics critical to the field.

Atherosclerosis and Thrombosis: We explored 50 questions in this category, delving into the complexities of atherosclerotic diseases and thrombotic processes. This section aimed to evaluate the system's understanding of cardiovascular pathologies, their etiologies, and the intricate mechanisms underlying these conditions. Overall, 48 out of the total of 50 questions were correctly answered.

Cellular Injury: A set of 55 questions tested the system's grasp of cellular injury mechanisms. This included queries on cellular responses to stress, pathophysiological changes in cell injury, and the various stages and outcomes of such injuries, mirroring real-world scenarios encountered in medical practice. Overall, 50 out of the total of 55 questions were correctly answered.

Embryology: In this segment, 52 questions were presented, focusing on the developmental stages and anomalies of embryology. The system's performance in this area was crucial to ascertain its ability to handle complex developmental biology concepts and their implications in pathological states. Overall, 45 out of the total of 52 questions were correctly answered.

Nutrition: Lastly, a set of 40 questions pertaining to nutrition was used. These questions were designed to assess the system's understanding of nutritional science, its role in health and disease, and its integration into pathological conditions. Overall, 37 out of the total of 40 questions were correctly answered.

In the evaluation process, each question was carefully crafted to present a realistic medical scenario, encompassing a range of symptoms and conditions that were specific to different gender and age groups. We provided the totality of questions and choices and requested the correct choice as a response. This approach was intended to simulate real-world clinical challenges, thereby testing the system's ability to apply its knowledge in a practical, context-sensitive manner.

For example, consider the following question from the Cellular Injury quiz.

A 50-year-old woman with a history of unstable angina suffers an acute myocardial infarction. Thrombolytic therapy with a tissue plasminogen activator (tPA) is administered to restore the coronary blood flow. Despite this therapy, the extent of myocardial fiber injury may increase due to which of the following cellular abnormalities?

- *[A.] Cytoskeletal intermediate filament loss*
- *[B.] Decreased intracellular pH from anaerobic glycolysis*
- *[C.] Increased free radical formation*
- *[D.] Mitochondrial swelling*
- *[E.] Nuclear chromatin clumping*
- *[F.] Reduced protein synthesis*

This question exemplifies the complexity and depth of the quizzes. It not only tests the system's grasp of specific medical knowledge but also its ability to analyze and apply this knowledge in diagnosing and understanding the progression of a disease. The inclusion of multiple answer choices, ranging from four to five options per question, further enhances the challenge, requiring the system to discern the most appropriate response from several plausible alternatives.

Such questions are integral to evaluating the system's proficiency in medical reasoning, particularly in pathology and related healthcare fields. They are designed not only to test the recall of factual information but also to assess the system's understanding of intricate physiological processes and its ability to make informed clinical decisions. This holistic approach ensures a thorough assessment of the system's capabilities in handling complex medical scenarios.

This comprehensive testing approach not only gauges the system's proficiency in handling specific medical knowledge but also its ability to integrate and apply this knowledge in a way that is coherent and contextually relevant to the field of pathology. The GPT-4 model had total precision of 91.37%, answering correctly 180 out of 197 questions. Although the system demonstrates a high success rate, further evaluation by medical experts and extensive testing across diverse scenarios are essential. Generally, systems empowered by LLMs like GPT-4, as used in this research, should be considered and treated as assistants and not replacements for human experts, particularly in sensitive domains such as the medical field, considering the critical impact of decision making in such disciplines.

6. Discussion of Results and Future Research

In this paper, the application of AI and particularly LLMs and NLP in healthcare is explored. A novel AI-empowered system is introduced, which is enhanced with rule-based algorithms and incorporates GPT models and other ML tools, to provide diagnostic advice. This system is tailored to address the complexities of healthcare experiences, specifically from a general practitioner's perspective. The research is organized into

various sections, covering theoretical foundations, system design and implementation, and practical use cases.

A key contribution of this work is the creation of a blueprint of domain-specific knowledge, serving as a contextual foundation for an AI system augmented with LLMs and rule-based logic. By generalizing the process, a measurable conclusion can be reached on the quality of the LLM's answers within a domain-specific context and a dialogue-defined space. These rules are formulated from a dialogue theory perspective, ensuring meaningful and relevant interactions. The system design is innovatively constructed and presented in a cost-effective manner, emphasizing reproducibility and scalability. The proposed AI-empowered, rule-augmented healthcare application integrates rules, external APIs, and modern methodologies to utilize current LLMs efficiently. This forms the basis for innovative approaches in the medical domain. Finally, the GPT-4-empowered system has undergone comprehensive evaluation in the field of general pathology, achieving a 91.37% accuracy rate in a set of 197 multiple-choice questions.

For future research and development, two critical areas are highlighted.

- Cost Analysis: Understanding the financial implications of deploying and using this AI system in healthcare is vital. This involves assessing the initial setup costs, ongoing operational expenses, and the potential financial benefits or savings that it might bring to healthcare providers and patients. This analysis will help to determine the economic feasibility and scalability of the system.
- Value-Based Care: This aspect focuses on comparing the costs and outcomes of care provided by different healthcare providers, considering both automated systems like the one proposed and traditional care methods. Key elements include the following.
  - Evaluating Effectiveness of Interventions: This involves measuring the impact of healthcare interventions on patient outcomes such as mortality rates, morbidity rates, and improvements in health-related quality of life. The AI system's role in facilitating timely interventions and improving these outcomes needs to be examined.
  - Patient Perspectives on Effectiveness: Assessing the value of care from the patient's point of view is crucial. This involves gathering and analyzing patient feedback to understand their experiences and satisfaction with the care provided, both through traditional means and the AI system.

These areas emphasize the need to balance technological advancement with practical, patient-centered care. Future research should also focus on ethical considerations, data privacy, and the integration of AI systems with existing healthcare infrastructures. The ultimate goal is to enhance healthcare delivery while ensuring that it is accessible, affordable, and aligned with patient needs and values.

Author Contributions: Conceptualization, D.P.P.; software, D.P.P.; validation, M.V. and G.A.T.; writing—original draft, D.P.P., M.V. and G.A.T.; writing—review and editing, M.V. and G.A.T.; visualization, D.P.P.; supervision M.V. and G.A.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Acknowledgments: This work has been partly supported by the University of Piraeus Research Center. Theoretical/medical support and technical/medical advice as per the validity of our hypothesis was provided on 30 October 2023 by the medical doctors of Dermacen S.A. https://www.dermatologikokentro.gr (accessed: 30 October 2023).

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ML	Machine Learning
AI	Artificial Intelligence
XAI	Explainable Artificial Intelligence
GAI	Generative Artificial Intelligence
LLM	Large Language Model
NLP	Natural Language Processing
EHR	Electronic Health Records
HMM	Hidden Markov Models
CRF	Conditional Random Fields
LDA	Latent Dirichlet Allocation
RAG	Retrieval-Augmented Generation

References

1. Trebble, T.M.; Hansi, N.; Hydes, T.; Smith, M.A.; Baker, M. Process mapping the patient journey: An introduction. *BMJ* **2010**, *341*, c4078. [CrossRef]
2. Gualandi, R.; Masella, C.; Viglione, D.; Tartaglini, D. Exploring the hospital patient journey: What does the patient experience? *PLoS ONE* **2019**, *14*, e0224899. [CrossRef] [PubMed]
3. McCarthy, S.; O'Raghallaigh, P.; Woodworth, S.; Lim, Y.L.; Kenny, L.C.; Adam, F. An integrated patient journey mapping tool for embedding quality in healthcare service reform. *J. Decis. Syst.* **2016**, *25*, 354–368. [CrossRef]
4. Panagoulias, D.P.; Virvou, M.; Tsihrintzis, G.A. Nuhealthsoft: A Nutritional and Health Data Processing Software Tool from a patient's perspective. In Proceedings of the 2022 16th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), Dijon, France, 19–21 October 2022; pp. 386–393.
5. Balogh, E.P.; Miller, B.T.; Ball, J.R. *Improving Diagnosis in Health Care*; The National Academies Press: Washington, DC, USA, 2015.
6. Pham, T.; Tran, T.; Phung, D.; Venkatesh, S. Predicting healthcare trajectories from medical records: A deep learning approach. *J. Biomed. Inform.* **2017**, *69*, 218–229. [CrossRef]
7. Esteva, A.; Robicquet, A.; Ramsundar, B.; Kuleshov, V.; DePristo, M.; Chou, K.; Cui, C.; Corrado, G.; Thrun, S.; Dean, J. A guide to deep learning in healthcare. *Nat. Med.* **2019**, *25*, 24–29. [CrossRef] [PubMed]
8. Xiao, C.; Choi, E.; Sun, J. Opportunities and challenges in developing deep learning models using electronic health records data: a systematic review. *J. Am. Med. Inform. Assoc.* **2018**, *25*, 1419–1428. [CrossRef] [PubMed]
9. Davenport, T.; Kalakota, R. The potential for artificial intelligence in healthcare. *Future Healthc. J.* **2019**, *6*, 94. [CrossRef] [PubMed]
10. Panagoulias, D.P.; Sotiropoulos, D.N.; Tsihrintzis, G.A. SVM-Based Blood Exam Classification for Predicting Defining Factors in Metabolic Syndrome Diagnosis. *Electronics* **2022**, *11*, 857. [CrossRef]
11. OpenAI. GPT-4 Technical Report. *arXiv* **2023**, arXiv:2303.08774 .
12. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
13. Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv* **2023**, arXiv:2307.09288.
14. Taori, R.; Gulrajani, I.; Zhang, T.; Dubois, Y.; Li, X.; Guestrin, C.; Liang, P.; Hashimoto, T.B. Stanford Alpaca: An Instruction-Following LLaMA Model. 2023. Available online: https://github.com/tatsu-lab/stanford_alpaca (accessed on 1 January 2024).
15. Panagoulias, D.; Palamidas, F.; Virvou, M.; Tsihrintzis, G.A. Evaluating the potential of LLMs and ChatGPT on medical diagnosis and treatment. In Proceedings of the 14th IEEE International Conference on Information, Intelligence, Systems, and Applications (IISA2023), Volos, Greece, 10–12 July 2023.
16. Gordon, E.B.; Towbin, A.J.; Wingrove, P.; Shafique, U.; Haas, B.; Kitts, A.B.; Feldman, J.; Furlan, A. Enhancing patient communication with Chat-GPT in radiology: Evaluating the efficacy and readability of answers to common imaging-related questions. *J. Am. Coll. Radiol.* **2023**. [CrossRef] [PubMed]
17. Floyd, W.; Kleber, T.; Pasli, M.; Qazi, J.; Huang, C.; Leng, J.; Ackerson, B.; Carpenter, D.; Salama, J.; Boyer, M. Evaluating the Reliability of Chat-GPT Model Responses for Radiation Oncology Patient Inquiries. *Int. J. Radiat. Oncol. Biol. Phys.* **2023**, *117*, e383. [CrossRef]
18. Gilson, A.; Safranek, C.W.; Huang, T.; Socrates, V.; Chi, L.; Taylor, R.A.; Chartash, D.; et al. How does ChatGPT perform on the United States medical licensing examination? The implications of large language models for medical education and knowledge assessment. *JMIR Med. Educ.* **2023**, *9*, e45312. [CrossRef]
19. Locke, S.; Bashall, A.; Al-Adely, S.; Moore, J.; Wilson, A.; Kitchen, G.B. Natural language processing in medicine: A review. *Trends Anaesth. Crit. Care* **2021**, *38*, 4–9. [CrossRef]

20. Kreimeyer, K.; Foster, M.; Pandey, A.; Arya, N.; Halford, G.; Jones, S.F.; Forshee, R.; Walderhaug, M.; Botsis, T. Natural language processing systems for capturing and standardizing unstructured clinical information: A systematic review. *J. Biomed. Inform.* **2017**, *73*, 14–29. [CrossRef]
21. Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. Language models are unsupervised multitask learners. *OpenAI Blog* **2019**, *1*, 9.
22. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December, 2017; Volume *30*, pp. 5998–6008
23. Weizenbaum, J. ELIZA—A computer program for the study of natural language communication between man and machine. *Commun. ACM* **1966**, *9*, 36–45. [CrossRef]
24. Wang, B.; Wang, A.; Chen, F.; Wang, Y.; Kuo, C.C.J. Evaluating word embedding models: Methods and experimental results. *Apsipa Trans. Signal Inf. Process.* **2019**, *8*, e19. [CrossRef]
25. Chiu, C.C.; Sainath, T.N.; Wu, Y.; Prabhavalkar, R.; Nguyen, P.; Chen, Z.; Kannan, A.; Weiss, R.J.; Rao, K.; Gonina, E.; et al. State-of-the-art speech recognition with sequence-to-sequence models. In Proceedings of the 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, AB, Canada, 15–20 April 2018; pp. 4774–4778.
26. OpenAI. *Better Language Models and Their Implications*; OpenAI: San Francisco, CA, USA, 2019.
27. Brown, T.B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language Models are Few-Shot Learners. *arXiv* **2020**, arXiv:2005.14165.
28. Bolukbasi, T.; Chang, K.W.; Zou, J.Y.; Saligrama, V.; Kalai, A.T. Man is to computer programmer as woman is to homemaker? Debiasing word embeddings. In Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016; pp. 4349–4357.
29. Gunning, D.; Stefik, M; Choi, J; Miller, T; Stumpf, S; Yang, G; XAI—Explainable artificial intelligence. *Sci. Robot.* **2019**, *37*, eaay7120. [CrossRef] [PubMed]
30. Holzinger, A.; Goebel, R.; Fong, R.; Moon, T.; Müller, K.R.; Samek, W. xxAI-beyond explainable artificial intelligence. In Proceedings of the xxAI-Beyond Explainable AI: International Workshop, Held in Conjunction with ICML 2020, Vienna, Austria, 18 July 2020; Revised and Extended Papers; Springer: Berlin/Heidelberg, Germany, 2022; pp. 3–10.
31. Panagoulias, D.P.; Sarmas, E.; Marinakis, V.; Virvou, M.; Tsihrintzis, G.A.; Doukas, H. Intelligent Decision Support for Energy Management: A Methodology for Tailored Explainability of Artificial Intelligence Analytics. *Electronics* **2023**, *12*, 4430. [CrossRef]
32. Panagoulias, D.; Palamidas, F.; Virvou, M.; Tsihrintzis, G.A. Evaluation of ChatGPT-supported diagnosis, staging and treatment planning for the case of lung cancer. In Proceedings of the 20th ACS/IEEE International Conference on Computer Systems and Applications, AICSSA 2023, Giza, Egypt, 4–7 December 2023.
33. Blandford, A.; Wesson, J.; Amalberti, R.; AlHazme, R.; Allwihan, R. Opportunities and challenges for telehealth within, and beyond, a pandemic. *Lancet Glob. Health* **2020**, *8*, e1364–e1365. [CrossRef] [PubMed]
34. Snoswell, C.L.; Chelberg, G.; De Guzman, K.R.; Haydon, H.H.; Thomas, E.E.; Caffery, L.J.; Smith, A.C. The clinical effectiveness of telehealth: a systematic review of meta-analyses from 2010 to 2019. *J. Telemed. Telecare* **2023**, *29*, 669–684. [CrossRef] [PubMed]
35. Kraft, A.D.; Quimbo, S.A.; Solon, O.; Shimkhada, R.; Florentino, J.; Peabody, J.W. The health and cost impact of care delay and the experimental impact of insurance on reducing delays. *J. Pediatr.* **2009**, *155*, 281–285. [CrossRef] [PubMed]
36. Martin, D.; Miller, A.P.; Quesnel-Vallée, A.; Caron, N.R.; Vissandjée, B.; Marchildon, G.P. Canada's universal health-care system: Achieving its potential. *Lancet* **2018**, *391*, 1718–1735. [CrossRef] [PubMed]
37. Goodair, B.; Reeves, A. Outsourcing health-care services to the private sector and treatable mortality rates in England, 2013–20: An observational study of NHS privatisation. *Lancet Public Health* **2022**, *7*, e638–e646. [CrossRef] [PubMed]
38. Yang, H.; Kim, S.; Park, J. Exploring avoidable, preventable, treatable mortality trends and effect factors by income level. *Eur. J. Public Health* **2023**, *33*, ckad160–1115. [CrossRef]
39. Treatable Mortality in Europe: Time Series. Available online: https://www.statista.com/statistics/1421315/treatable-mortality-in-europe-time-series (accessed on 18 December 2023).
40. NuhealtSoft Suite. Available online: https://www.diskinside.com/nuhealthsoft/ (accessed on 8 January 2024)
41. Panagoulias, D.P.; Virvou, M.; Tsihrintzis, G.A. Rule-Augmented Artificial Intelligence-empowered Systems for Medical Diagnosis using Large Language Models. In Proceedings of the 2023 IEEE 35th International Conference on Tools with Artificial Intelligence (ICTAI), Atlanta, GA, USA, 6–8 November 2023.
42. Gorsky, P.; Caspi, A.; Chajut, E. The "theory of instructional dialogue": Toward a unified theory of instructional design. In *Understanding Online Instructional Modeling: Theories and Practices*; IGI Global: Hershey, PA, USA, 2008; pp. 47–69.
43. Wilson, D.C. Chapter Three: A Framework for Clarifying. In *A Guide to Good Reasoning: Cultivating Intellectual Virtues*; McGraw-Hill College: New York, NY, USA, 2020.
44. García-Carrión, R.; López de Aguileta, G.; Padrós, M.; Ramis-Salas, M. Implications for social impact of dialogic teaching and learning. *Front. Psychol.* **2020**, *11*, 140. [CrossRef]
45. Mitchell, M.L.; Henderson, A.; Groves, M.; Dalton, M.; Nulty, D. The objective structured clinical examination (OSCE): optimising its value in the undergraduate nursing curriculum. *Nurse Educ. Today* **2009**, *29*, 398–404. [CrossRef] [PubMed]
46. Majumder, M.A.A.; Kumar, A.; Krishnamurthy, K.; Ojeh, N.; Adams, O.P.; Sa, B. An evaluative study of objective structured clinical examination (OSCE): students and examiners perspectives. *Adv. Med Educ. Pract.* **2019**, *10*, 387–397. [CrossRef] [PubMed]

47. Customizing Conversational Memory. Available online: https://python.langchain.com/docs/modules/memory/conversational_customization (accessed on 29 September 2023).
48. Vector Stores-LlamaIndex. Available online: https://gpt-index.readthedocs.io/en/v0.7.8/core_modules/data_modules/storage/vector_stores.html (accessed on 20 November 2023).
49. Panagoulias, D.P.; Sotiropoulos, D.N.; Tsihrintzis, G.A. An Extreme Value Analysis-Based Systemic Approach in Healthcare Information Systems: The Case of Dietary Intake. *Electronics* **2023**, *12*, 204. [CrossRef]
50. The Internet Pathology Laboratory for Medical Education. Available online: https://webpath.med.utah.edu/webpath.html (accessed on 15 December 2023).

electronics

MDPI

Article

The Genesis of *AI by AI* Integrated Circuit: Where AI Creates AI

Emilio Isaac Baungarten-Leon [1,2,*], Susana Ortega-Cisneros [1,*], Mohamed Abdelmoneum [3], Ruth Yadira Vidana Morales [3] and German Pinedo-Diaz [1]

[1] Centro de Investigación y de Estudios Avanzados, Instituto Politécnico Nacional, Zapopan 45019, Mexico; german.pinedo@cinvestav.mx
[2] Diseño Ciencia y Tecnologia, Universidad Autónoma de Guadalajara, Ciudad Universitaria, Zapopan 45129, Mexico
[3] Intel Corporation-Intel Labs, Hillsboro, OR 97124, USA; mohamed.a.abdel-moneum@intel.com (M.A.); ruth.y.vidana.morales@intel.com (R.Y.V.M.)
* Correspondence: emilio.baungarten@cinvestav.mx (E.I.B.-L.); susana.ortega@cinvestav.mx (S.O.-C.)

Abstract: The typical Integrated Circuit (IC) development process commences with formulating specifications in natural language and subsequently proceeds to Register Transfer Level (RTL) implementation. RTL code is traditionally generated through manual efforts, using Hardware Description Languages (HDL) such as VHDL or Verilog. High-Level Synthesis (HLS), on the other hand, converts programming languages to HDL; these methods aim to streamline the engineering process, minimizing human effort and errors. Currently, Electronic Design Automation (EDA) algorithms have been improved with the use of AI, with new advancements in commercial (such as ChatGPT, Bard, among others) Large Language Models (LLM) and open-source tools presenting an opportunity to automate the chip design process. This paper centers on the creation of *AI by AI*, a Convolutional Neural Network (CNN) IC entirely developed by an LLM (ChatGPT-4), and its manufacturing with the first fabricable open-source Process Design Kit (PDK), SKY130A. The challenges, opportunities, advantages, disadvantages, conversation flow, and workflow involved in CNN IC development are presented in this work, culminating in the manufacturing process of *AI by AI* using a 130 nm technology, marking a groundbreaking achievement as possibly the world's first CNN entirely written by AI for its IC manufacturing with a free PDK, being a benchmark for systems that can be generated today with LLMs.

Keywords: convolutional neural network; hardware design; integrated circuit; large language models

1. Introduction

The history of Integrated Circuit (IC) design is marked by innovation and technological strides. It began in the late 1950s with the introduction of the transistor [1]. Texas Instruments pioneered the first IC in 1958, integrating two transistors on a silicon–germanium bar [2]. Until the arrival of Computer-Aided Design (CAD) tools in 1966, ICs were manually drawn on paper [3].

The evolution of Hardware Description Language (HDL) started in the early 1970s with Register Transfer Level (RTL), allowing for thousands of transistors per IC [4]. DEC's PDP-16 RT-Level Modules [5], Instruction Set Processor Specifications [6], and Incremental System Programming Language [7] were significant contributions. In the late 1970s, programmable logic devices increased the demand for standard languages, and in 1985, Gateway and Intrametric introduced Verilog and VHSIC Hardware Description Language (VHDL) [8,9].

Alongside Verilog and VHDL, C-based hardware description languages, known as High-Level Synthesis (HLS), emerged. SystemC allowed the use of standard C++ and a class library for HDL generation in 1999, simplifying the IC development process with HLS [10]. Today, HLS tools like LegUP, Xilinx Vivado HLS, and Intel's HLS compiler transform C++ into HDL.

On the other hand, the journey of Artificial Intelligence (AI) started in the 1960s and improved during the 1970s and 1980s with foundational concepts and algorithms of deep learning and Artificial Neural Networks (ANN) [11,12]. During the late 1980s and early 1990s, the Machine Learning (ML) and AI community experienced a wave of enthusiasm as it was discovered that ANNs could tackle certain problems in novel ways. These networks had the distinct advantage of processing raw and diverse data types and developing hierarchical structures autonomously during the training phase for predictive tasks. However, the computational power at the time was insufficient for large-scale problems, limiting the application to smaller, simpler tasks [12–14].

It was not until the end of the 2000s that technological advancements, propelled by Moore's Law, equipped computers with the necessary power to train extensive ANNs on substantial, real-world challenges, such as the Imagenet project [15]. This advancement was largely due to the advent of general-purpose computing on graphics processing units, which offered superior floating-point performance compared to Central Processing Units (CPUs) [16]. This shift enabled ANNs to achieve remarkable results on complex issues of significant importance.

The last decade has been transformative for ML, especially with the rise of deep learning techniques that utilize ANN. These advancements have significantly enhanced the precision of systems in various domains [17]. Notable progress has been made in fields such as computer vision [18–21], speech recognition [22,23], language translation [24], and other complex natural language processing tasks [25–30]. This progress is attributed to the collective efforts and breakthroughs documented in key research papers.

Additionally, reinforcement learning shows promise in automating the design of custom Application-Specific Integrated Circuit (ASIC) by solving nondeterministic polynomial-hard optimization problems that are currently reliant on human expertise. This approach could revolutionize the synthesis, placement, and routing processes in chip design, potentially outperforming human teams by rapidly generating efficient layouts [31–33]. Google's preliminary experiments with this technology have yielded encouraging results, suggesting a future where machine learning accelerates and enhances the ASIC design process [14].

Research conducted by International Business Strategies Inc. in 2014, 2018, and 2022 categorizes the IC design costs into seven components: Intellectual Property (IP), Architecture, Verification, Physical Design, Software, Prototyping, and Validation. These studies reveal that design costs fluctuate significantly due to two primary factors: the prevailing technology at the time and the nanometer scale at which it is desired to fabricate. For instance, the design cost for a 28 nm circuit was approximately USD 140 million in 2014, reduced to USD 51.3 million in 2018, and further decreased to USD 48 million in 2022. Based on the 2018 and 2022 analyses, the estimated distribution of costs is as follows: IP at 6.85%, Architecture at 5.24%, Verification at 21.24%, Physical Design at 10.2%, Software at 43.32%, Prototyping at 5.24%, and Validation at 7.92%. These percentages provide a framework for approximating the allocation of expenses in IC design.

Advancements in machine learning could streamline the entire ASIC design process, from high-level synthesis to low-level logic placement and routing. This automation could drastically cut down design time from months to weeks, changing the economic calculus by reducing costs in Prototyping, Verification, and Architecture, combined with open-source tools and IPs, design costs would be further reduced. It may be feasible to create customized chips, which are currently reserved for high-volume and high-value scenarios.

Today, commercial LLMs like OpenAI's ChatGPT [34], Google's Bard [35], and Microsoft AI chatbot [36] have been used to introduce innovative HDL generation. These methods involve feeding the LLM with the system specifications, which then automatically produce HDL code. This synergy between AI and IC development promises enhanced efficiency and opens new frontiers in the field. Nevertheless, the state-of-the-art models fall short in their ability to effectively comprehend and rectify errors introduced by these tools, making it challenging to autonomously generate comprehensive designs and testbenches with minimal initial human intervention [37–39].

This work combines different processes to increase the complexity of an IC and reduce the amount of work required. The primary research inquiry revolves around the capability of contemporary commercial LLMs to produce Convolutional Neural Network (CNN) hardware designs that are not only synthesizable, but also manufacturable using the first open-source Process Design Kits (PDKs) called SKY130A.

The development of *AI by AI*—a CNN IC engineered for MNIST dataset classification—involves the use of LLM, Vivado HLS, Verilog, OpenLane, and Caravel. *AI by AI* was entirely crafted by OpenAI's ChatGPT-4. It began as a TensorFlow (TF) CNN architecture, followed by a downscaling from Python to C++, and then was translated to Verilog using Vivado HLS. The layout design process is made by OpenLane, resulting in a layout IP of the CNN. The journey culminated with the integration of the CNN IP with Caravel, a template System on Chip (SoC) which is ready for manufacturing using ChipIgnite shuttles, a multi-project wafer program by Efabless, with the SKY130A PDK [40,41]. Throughout this paper, we delve deeply into the development of *AI by AI* IC from TF to tape-out.

The remainder of this work is organized as follows: Section 2 provides an overview of the employed tools, outlining both their advantages and disadvantages; Section 3 explains the workflow and conversation flow; Section 4 is about the implementation of *AI by AI* IC; Section 5 shows the obtained results; Section 6 presents the discussions; and, finally, Section 7 concludes this work.

2. Development Tools

2.1. Vivado HLS 2019.1

While traditional HDLs like Verilog and VHDL are acknowledged for their efficacy, their low-level abstraction often leads to long development cycles. A divergent approach is presented by HLS, offering a faster and more agile solution for hardware description development [42].

HLS functions through an automated process, enabling the generation of synthesizable RTL code from algorithms scripted in high-level languages such as C/C++ or System C. Although the resulting RTL code is commonly implemented on a Field-Programmable Gate Array (FPGA), it can also be translated into silicon, since it is described in HDL. In this case, the attractiveness of HLS lies in the possibility of generating HW with programming languages [42–44], Table 1 shows some advantages and disadvantages of HLS.

Table 1. Advantages and disadvantages of HLS.

Advantages of HLS	Disadvantages of HLS
Reduces development time and effort	Does not have the same quality of results as HDLs
Architecture selection and optimization	Inconveniences in the hardware description
Parallelism and pipelining	Does not support all the features and constructs of the input languages
Allocates and shares resources efficiently	May not be compatible with all the existing tools and flows

2.2. OpenLane

This software is an open-source automated flow for layout design, conformed by various tools from *OpenROAD* and *Qflow*, focusing on the RTL to Graphic Design System (GDSII) design. Initially deployed for implementing the StriVe family, a RISC-V based SoC, using free EDA tools and the first open-source PDK SKY130A.

Currently comprising over seventy scripts and utilities, *OpenLane* can be configured for customized flows, enabling the implementation of diverse designs with any technology or PDK. The flow encompasses stages like synthesis, floorplaning, placement, Clock Tree Synthesis (CTS), routing, tapeout, and signoff [45–47].

The *OpenLane* flow initiates with HDL synthesis where the *Yosys* synthesis tool optimizes the design, resulting in a netlist mapped by the PDK. During this phase, design

constraints like clock definition and boundary conditions can be integrated, and Static Timing Analysis (STA) can be executed using the *OpenSTA* tool. Subsequently, floorplanning is conducted, with *OpenROAD* tools employed for macro-related tasks, producing a Design Exchange Format (DEF) file and defining matrix and macro core sizes. The *Padring* tool is harnessed for chip-level floorplanning, optimizing core pin positions for improved pad frame and core interconnect placement.

Post-floorplanning, standard cell, and macro placement are accomplished using the *Re-PlAce* tool, with subsequent placement checks conducted via *OpenDP*. The CTS phase follows, with *TritonCTS* placing clock branches and *OpenDP* adding necessary buffers. Routing is executed through a two-step approach: an initial phase with *FastRoute*, followed by a more intricate process with *TritonRoute*. In the concluding stages, the design undergoes verifications, including Design Rule Check (DRC), Layout Versus Schematic (LVS), and STA. Successful completion of these checks deems the design suitable for approval [45,48]; a graphical representation of the described process is illustrated in Figure 1 below.

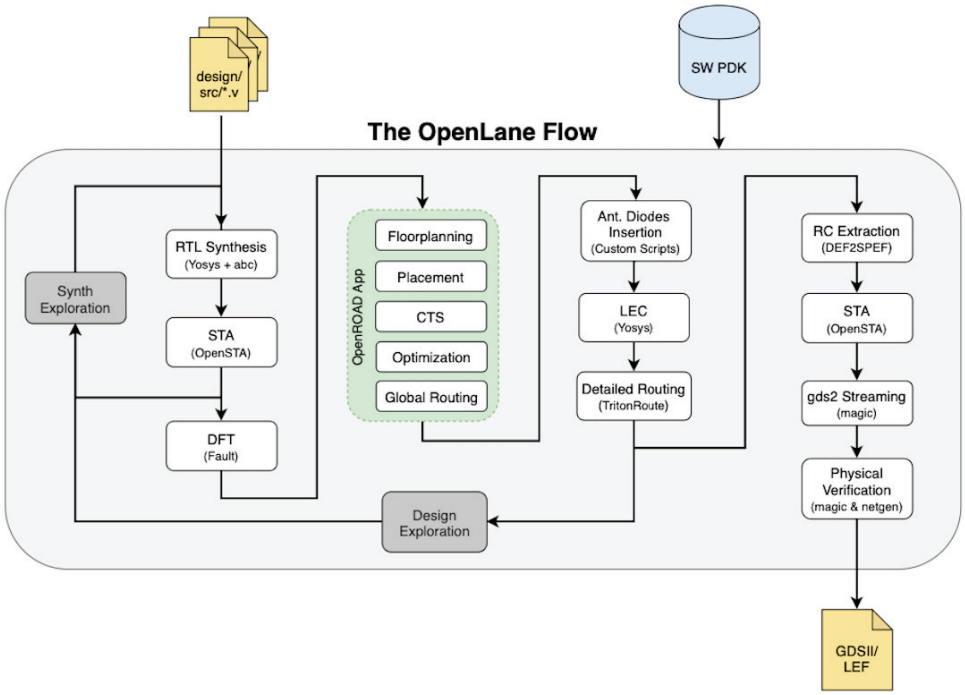


Figure 1. OpenLane workflow [48].

With the rise of *OpenLane*, new research has made a comparative analysis of this open-source tool with commercial tools [45–47,49,50]. Table 2 shows some of the advantages and disadvantages of *OpenLane*.

Table 2. Advantages and disadvantages of *OpenLane*.

Advantages of OpenLane	Disadvantages of OpenLane
The entire flow is configured through a single configuration file	Less control over the flow compared to commercial tools
Automated flow, requires no manual intervention, once configured	Commercial tools have better time optimization
Open-source, no charge for use	*OpenLane* uses more logic cells in the design
Reduces the time and expertise required to obtain the GDSII	*OpenLane* generated designs tend to consume more power

2.3. Caravel

Caravel is an SoC template developed by *Efabless* and built upon SKY130A and GF180MCUC technologies. It comprises three main sections: the template frame and two wrapper modules, known as the management area and user area [51].

The template frame is equipped with essential components, including a clocking module, Delay Locked Loop (DLL), user ID, housekeeping Serial Peripheral Interface (SPI), Power-On Reset (POR), and a General-Purpose Input/Output (GPIO) controller. The management area, housing a RISC-V based SoC, can configure and control the user area. The user area occupies a silicon space of 2.92 mm by 3.52 mm and includes 38 I/O pads, 128 Logic Analyzer (LA) signals, and four power pads. Figure 2 shows the block diagram of *Caravel* and its three sections [51].

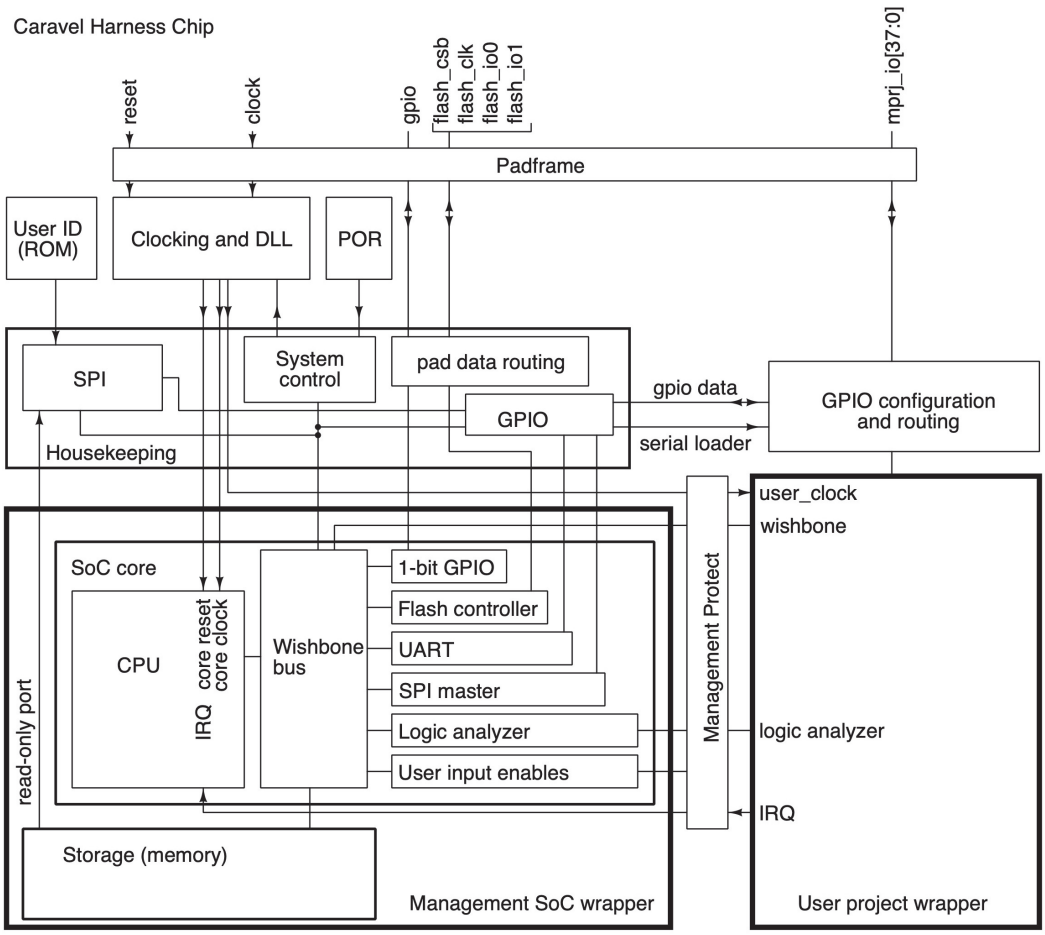


Figure 2. Caravel SoC architecture [51].

The very nature of a template offers great advantages when designing an IC; however, it also has some limitations. Table 3 shows these advantages and disadvantages.

Table 3. Advantages and disadvantages of *Caravel*.

Advantages of Caravel	Disadvantages of Caravel
Allows low-cost and low-risk custom SoC design	Limited to SKY130A and GF180MCUC PDKs.
Supports various open-source tools and flows for IC design	May not be suitable for complex or high-end IC design projects
Enables fast SoC prototyping	Limited by 10 mm^2 and 38 GPIO pins
Enables collaboration and sharing with the open-source hardware community	

3. Workflow and Conversation Flow

3.1. Large Language Model Conversation Flow

The cornerstone of this work lies in the use of a commercial LLM for precise code generation, guided by the conversational flow depicted in Figure 3.

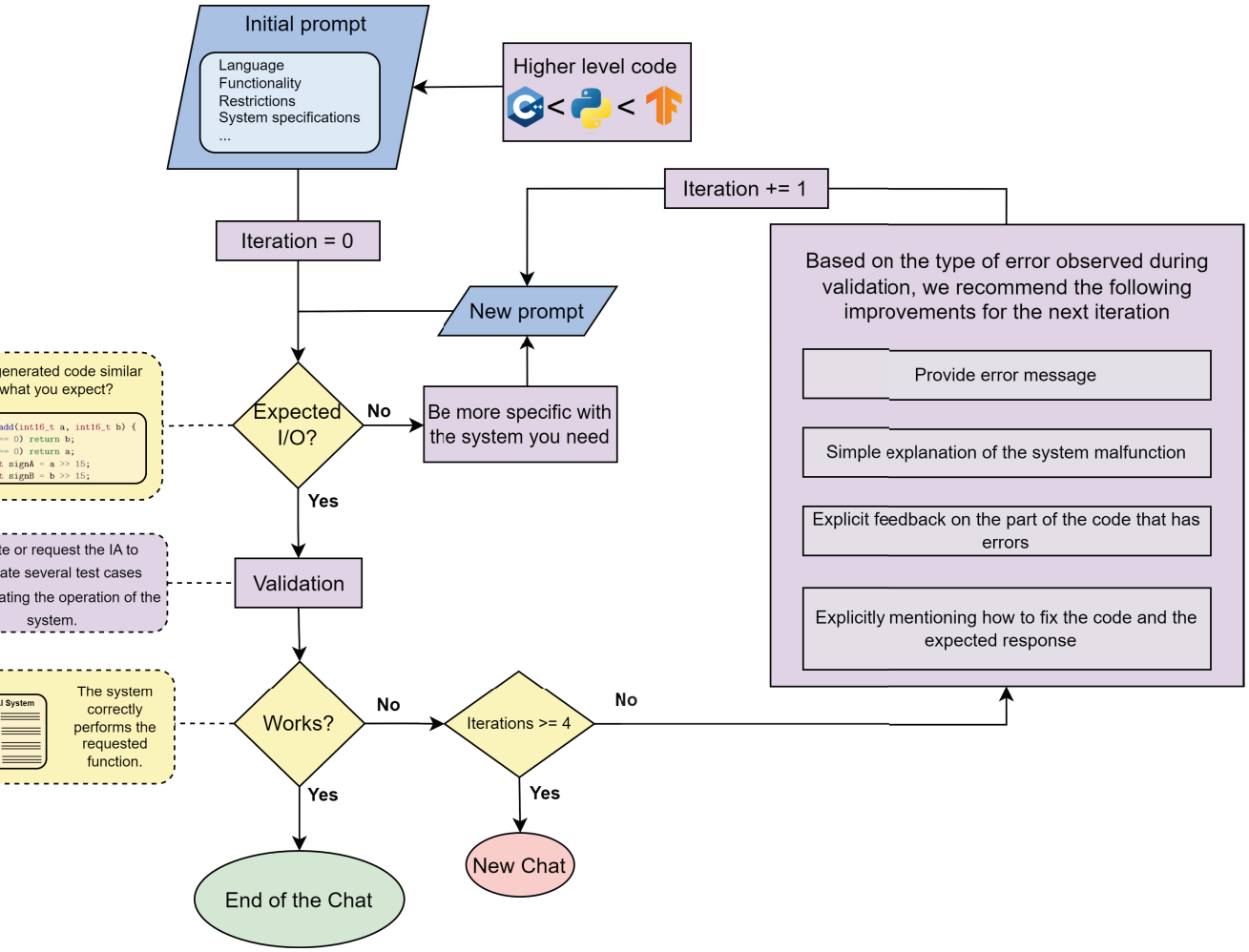


Figure 3. Conversation flow with LLM for code generation, highlighting the transition points and the recommendation for generation of prompts and new chat session.

The process starts by combining code from a higher abstraction level, if available, with the initial prompt. If the AI response does not meet the expected criteria, the creation of a more detailed prompt is initiated, to clarify specific requirements.

Upon receiving the expected response, the progression involves code simulation and testing, which means running the function for different cases and getting the expected result. The conversation concludes when the code functions as intended. However, in cases of code malfunction, the subsequent step entails the crafting of a new prompt incorporating error messages, heightened specificity, illustrative examples, or details regarding required code modifications, e.g., if the code does not work due to a data type error, it communicates so to the AI. After multiple iterations, when the LLM consistently produces similarly incorrect responses, it indicates the need to commence a new chat session.

3.2. From TensorFlow to Layout

Throughout the entire workflow, LLM played a central role in code generation, aligning with the conversation flow detailed in Section 3.1. *AI by AI* commences with the creation and training of the CNN architecture via TF. This initial phase allows training the CNN and capturing its essential weights and biases.

Considering the limitations of Caravel, we chose to implement a compact CNN with the following layers: Input layer (28 × 28 × 1), Convolutional Layer 1 (26 × 26 × 4), Max Pooling Layer 1 (6 × 6 × 4), Convolutional Layer 2 (4 × 4 × 8), Max Pooling Layer 2 (2 × 2 × 8), Flattening Layer (1 × 32), and dense layer.

Subsequently, the transformation of the TF model into a set of Python functions dedicated to executing the inference of the CNN, without the use of libraries, is initiated. A pivotal following step involves converting the Python-based forward function into C++, allowing the use of Vivado HLS.

The workflow culminates with the implementation of the CNN at the layout level, integrating it with Caravel. Figure 4 presents a visual representation of this process.

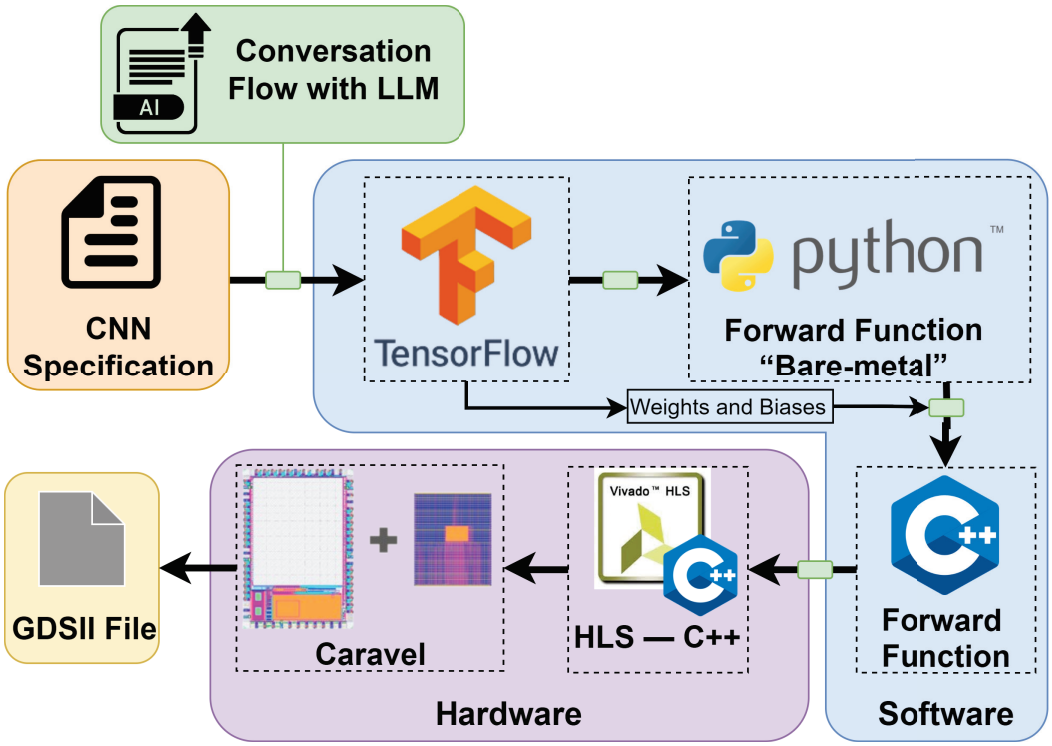


Figure 4. Workflow for the development of a CNN using LLM, from TF architecture to GDSII throughout Caravel integration.

4. Development of *AI by AI*

The development of *AI by AI* consists of a series of dialogues with ChatGPT-4, following the conversational structure outlined in Figure 3. For access to the complete conversations, the generated code, and the entire project, please refer to the following GitHub repository: https://github.com/Baungarten-CINVESTAV/AI_by_AI (accessed on 4 March 2024). Table 4 provides the ChatGPT URL of each conversation and the main topic covered in those conversations, accessed on 4 March 2024.

Table 4. ChatGPT-4, conversation URL.

Subject of the Conversation	URL
Implementing a CNN in TF (accessed on 4 March 2024)	https://chat.openai.com/share/4e8a7cf2-a9e9-4461-a4b3-b9e8b4aa284f
Implementation of a forward function in Python without libraries (Bare-Metal) (accessed on 4 March 2024)	https://chat.openai.com/share/c96772be-4dac-43da-8013-c657dd935efa
From Python to C code I (accessed on 4 March 2024)	https://chat.openai.com/share/c96772be-4dac-43da-8013-c657dd935efa
From Python to C code II (accessed on 4 March 2024)	https://chat.openai.com/share/64b09191-401e-4d04-8eb5-5383b95ceea5
Bias and weights as global parameters (accessed on 4 March 2024)	https://chat.openai.com/share/4b8237a4-20c3-434b-89fb-084fc5b57287
From C to HLS I (accessed on 4 March 2024)	https://chat.openai.com/share/9037bfcd-8d23-4701-bafd-59eca930a822
From C to HLS II (accessed on 4 March 2024)	https://chat.openai.com/share/84dd776b-0036-4fec-a878-dbcb33f6f210
Add function, half-precision floating-point (accessed on 4 March 2024)	https://chat.openai.com/share/0f617bfd-f59a-49a3-a561-20b2779ca121
Mult, Relu, Max function, half-precision floating-point (accessed on 4 March 2024)	https://chat.openai.com/share/2b207fc6-5952-4ef7-a562-64765e2d6722
Exponent function, half-precision floating-point (accessed on 4 March 2024)	https://chat.openai.com/share/5345f69b-5e04-4fdf-a062-f29b2fcc4564

This chapter is structured into five distinct subsections, as visually represented in Figure 4. In each of these sections, the relevant prompts, primary challenges, key considerations, and the step-by-step development process are detailed. The journey commences with the creation of the CNN using TF, and culminated with the generation of the GDSII file ready for manufacturing.

4.1. CNN with TF

The CNN was designed for image inference tasks toward the renowned MNIST dataset [52]. To harness the power of cloud computing, we opted for Google Colab [53], primarily due to its integration of TF libraries and the capacity to use GPUs.

The noteworthy prompts that emerged during the interactions with ChatGPT-4 included:

- Generate a CNN for the MNIS dataset using TF and Google Colab.
- Change the CNN model to be: 4 × 3 × 3 Conv2D, 4 × 4 MaxPool, 8 × 3 × 3 Conv2D, 2 × 2 MaxPool, and work with float16.
- Obtain the weights and biases for each layer, then write those weights on a .npy file and .bin file. Save both as a float16 data type.

The approach taken involved implementing a compact network using the following layers 4 × 3 × 3 Conv2D, 4 × 4 MaxPool, 8 × 3 × 3 Conv2D, 2 × 2 MaxPool, flatten, and finally, the dense layer, as well as the use of half-precision floating-point format to optimize resource usage. Figure 5 illustrates the CNN created.

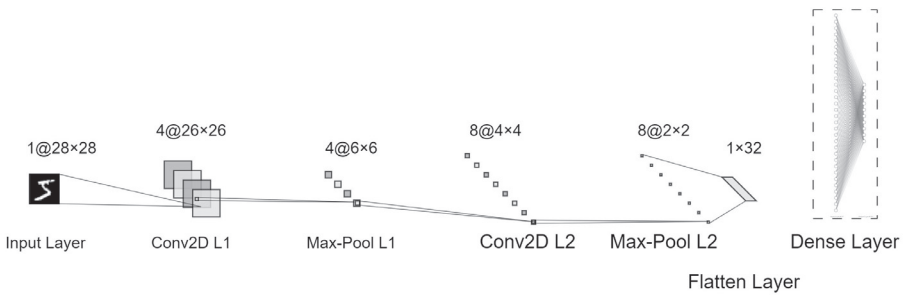


Figure 5. CNN architecture for the task of classifying MNIST images.

The implemented CNN utilizes a total of 666 parameters. This breakdown encompasses 36 weights and 4 biases for the initial convolutional layer, 288 weights and 8 biases for the second convolutional layer, and, finally, 320 weights and 10 biases for the dense layer. In terms of memory consumption, this results in a total of 1.332 KB required only for storing the weights and biases. At the end of the training phase, the model showed an accuracy of 99.4%. Part of the TF code of the CNN generated by the IA can be found below.

```
# Define the CNN model
model = models.Sequential()
model.add(layers.Conv2D(4, (3, 3), activation='relu', input_shape=(28,28,
↪ 1)))
model.add(layers.MaxPooling2D((4, 4)))
model.add(layers.Conv2D(8, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Flatten())
model.add(layers.Dense(10, dtype='float16'))
```

4.2. Forward Function in Python

Implementing the inference function in Python without the use of the TF library is a critical step in the process because, as we approach lower-level languages or avoid the use of libraries, we obtain answers with a higher number of errors. To face that problem, we provide the LLM with examples in a higher level language. In this case, ChatGPT-4 is instructed to utilize the pre-existing network, created with TF, to create the inference function using the weights and biases from previously saved NumPy files.

Key prompts from interactions with ChatGPT-4:

- Write a bare metal implementation of the CNN, just the forward function, assuming that the CNN was trained previously.
- Call the function forward based on the previous weights and biases .np file.
- Develop a functionality test of the previous code showing the selected image and its label.

The previous chat generated six essential secondary functions required for inference implementation: relu, softmax, conv2d_forwar, maxpool2d_forward, flatten, dense_forward, and a main function named forward, which calls within it the secondary functions. The following code shows the definition of the forward function and how it was used to perform the test phase.

```
def forward(X, W_conv1, b_conv1, W_conv2, b_conv2, W_dense2, b_dense2):
    out = conv2d_forward(X, W_conv1, b_conv1)
    out = relu(out)
    out = maxpool2d_forward(out, 4)
    out = conv2d_forward(out, W_conv2, b_conv2)
    out = relu(out)
    out = maxpool2d_forward(out, 2)
    out = flatten(out)
    out = dense_forward(out, W_dense2, b_dense2)
    out = softmax(out)
    return out

for i in range(10000):
    # prepare the input
    x = test_images[i].astype(np.float16)
```

```
    # Ensure that the single image has the right shape (28, 28, 1)
    single_image = np.reshape(x, (28, 28, 1))
    output = forward(single_image, weights_conv1, biases_conv1,
     ↪  weights_conv2, biases_conv2, weights_dense2, biases_dense2)
    # Find the index of the maximum value
    prediction = np.argmax(output)
    # compare the prediction with the actual label
    actual_label = np.argmax(test_labels[i])
    #actual_label = test_labels[i]
    if prediction == actual_label:
        correct_predictions += 1
# calculate the accuracy
accuracy = correct_predictions / 10000
print("The accuracy of the forward_pass function is:", accuracy)
```

Chat interactions from Sections 4.1 and 4.2 were brief, primarily due to the use of Python.

4.3. From Python to C++

Utilizing a low-level programming language necessitated a more explicit approach to crafting prompts. This involved providing the entire code for the seven previously generated functions and demanded a higher number of iterations.

Main prompts obtained during interactions with ChatGPT-4:

- I develop the following CNN model in a python bare metal implementation for the mnist dataset:
  The CNN model is:
  <Here, the python code is attached>
  Rewrite the Python code on a C code: Weights and biases will be loaded from the bin file.
- Implement the whole forward function and develop the fmaxf function used in the maxpool layer.
- Create a function that convert the forward function into one hot output.
- Based on the C code create a function that compares the output of the forward function and the label:
  <Here, the C++ code is attached>

After the "From Python to C code" conversations mentioned in Table 4, we achieved a successful implementation of all the layers of the CNN in a short time. The C++ code presented below shows how the forward function is called N times for the test phase.

```
// Load the weights and biases from the binary files
    load_weights_biases();
    // For each image
    for (int i = 0; i < NUM_IMAGES; ++i) {
        // Call the forward pass function
        out_forward = forward_pass(images[i], one_Hot_output);
        printf("Predicted label = %d, Actual label = %d\n", out_forward,
         ↪  labels[i]);
        // If the predicted label matches the actual label, increment the
         ↪  count of correct predictions
        if (out_forward == labels[i]) {
            ++correct_predictions;
        }
```

```
    }
    // Calculate the accuracy
    float accuracy = ((float)correct_predictions / NUM_IMAGES) * 100.0f;
    printf("Accuracy = %.2f%%\n", accuracy);
```

Part of the forward_pass function is presented below, where each of the layers, both convolutional and maxpool, was implemented through a series of for loops, where variable *i* represents the pixel coordinate in x, variable *j* represents the pixel coordinate in y, and variable *k* represents the filter number. On the other hand variables *di* and *dj* represent the kernel, being a 3×3 kernel for the first convolutional layer.

```
int forward_pass(float* image, int* one_hot_output) {
    // Assume here that the image has a size of 28x28x1 and weights and
     ↪  biases are already loaded
    // First Conv2D layer: input is 28x28x1, filter is 3x3x1x4
    // It results in a 26x26x4 output (we are assuming VALID padding)
    float conv1[26][26][4];
    for(int i = 0; i < 26; i++)
        for(int j = 0; j < 26; j++)
            for(int k = 0; k < 4; k++) {
                conv1[i][j][k] = 0;
                // Convolution operation
                for(int di = 0; di < 3; di++)
                    for(int dj = 0; dj < 3; dj++)
                        conv1[i][j][k] += image[(i+di)*28 + (j+dj)] *
                         ↪  weights_conv1[(di*3 + dj)*4 + k];
                conv1[i][j][k] += biases_conv1[k];
                // ReLU activation
                conv1[i][j][k] = relu(conv1[i][j][k]);
            }
```

The C++ code provided by the AI can be easily scaled and customized to create various convolutional layers, changing only the ranges of the first two for loops that represent the size of the image, the third for represents the amount of filter that the layer has, and the last two for loops represent the size of the kernel. This versatility opens the opportunity to construct a wide range of CNNs, and all with the code provided by the AI.

4.4. Vivado HLS Considerations

The C++ code generated by the IA uses floating data types, although Vivado HLS supports this type of data when implemented at the hardware level it uses a restricted Floating Point Units (FPUs) IP, so its use is limited only to Xilinx boards.

To face this issue, C++ functions that utilize 16-bit integer data types, but perform floating-point operations at the bit level, were developed through a series of LLM conversations, keeping in mind the IEEE® 754 half-precision floating-point format.

A total of eight functions were developed: addition, subtraction, multiplication, division, exponential, softmax, relu, and max. The addition, multiplication and division functions can be found in Appendix A.

The main prompts obtained during interactions with ChatGPT-4 are:

- Develop an [addition, subtraction, multiplication, division, relu, max] function of two numbers of 16 bits with the following structure, sign bit, 5-bit exponent, 10-bit mantissa. Generate the C code for HLS.
- Consider the case in which A and B are the same number.
- Consider the case in which A or B are equal to 0.
- Consider the case in which A and B have different signs.
- Develop an exp function of a number of 16 bits with the following structure, sign bit, 5 bits exponent, 10 bits mantissa. Generate the C code for HLS, avoid the use of floating point data type, and if you use an add, mult, div functions use:

<Here, the C++ code floating functions are attached>

The generated functions are then used to perform floating operations and used to replace the arithmetic symbols of the existing solution; e.g., instead of executing the

```
conv1[i][j][k] += image[(i+di)*28 + (j+dj)] * weights_conv1[(di*3 + dj)*4
↪  + k];
```

operationpresented in the forward function, the operation is executed as

```
aux_mult = multiply_custom_float(image[(i+di)*28
↪  +(j+dj)],weights_conv1[(di*3 + dj)*4 + k]);
conv1[i][j][k] = add(conv1[i][j][k],aux_mult);
```

where the multiplication of the pixel and the kernel is performed by the multiply_custom_floa function, and the summation of the convolution by the add function.

Due to variations in rounding methods for floating operations, the accuracy experienced a 1.4% reduction, which means that change from 99.4% to 98%. However, this error can be avoided if the floating functions created use exactly the same rounding algorithm used by TF.

4.5. Integration of the CNN with Caravel

To integrate the CNN with the SoC template Caravel involves the creation of a single macro encompassing the logic of all the modules generated by HLS, because the logical density of the design utilizes the majority of the user area an external memory was employed for image storage which was connected to Caravel via GPIO ports. Meanwhile, the CNN was linked to the Caravel RISCV processor using the LA ports as Figure 6 illustrates. This connection allowed the RISCV processor to manage the initiation of the inference process, with the signal la_data_in[2], system restarts, with the signal la_data_in[1], and receive the response of the inference from the CNN, with the signal la_data_out[31:28]; Table 5 shows the connection between *AI by AI* and Caravel.

The verilog code provided to the OpenLane layout tool is just an instantiation of the IP generated by HLS connected to the Caravel ports; Appendix B shows this instantiation.

Table 5. Pinout of Caravel and AI by AI.

Caravel	AI by AI	Type
wb_clk_i	o_mux_clk	Input
io_in[36]	o_mux_clk	Input
io_in[37]	s_mux_clk	Input
o_mux_clk	ap_clk	Input
la_data_in[1]	in_ap_rst	Input
io_in[35]	in_ap_rst	Input

Table 5. *Cont.*

Caravel	AI by AI	Type
wb_clk_i	o_mux_clk	Input
io_in[36]	o_mux_clk	Input
io_in[37]	s_mux_clk	Input
o_mux_clk	ap_clk	Input
la_data_in[1]	in_ap_rst	Input
io_in[35]	in_ap_rst	Input
la_data_out[2]	ap_start	Input
la_data_out[3]	ap_done	Output
la_data_out[4]	ap_ready	Output
io_out[16:5]	image_r_Addr_A	Output
io_out[17]	image_r_EN_A	Output
N/A	image_r_WEN_A	Output
N/A	image_r_Din_A	Output
io_in[33:18]	image_r_Dout_A	Input
io_out[34]	image_r_Clk_A	Output
N/A	image_r_Rst_A	Output
la_data_out[31:28]	ap_return	Output

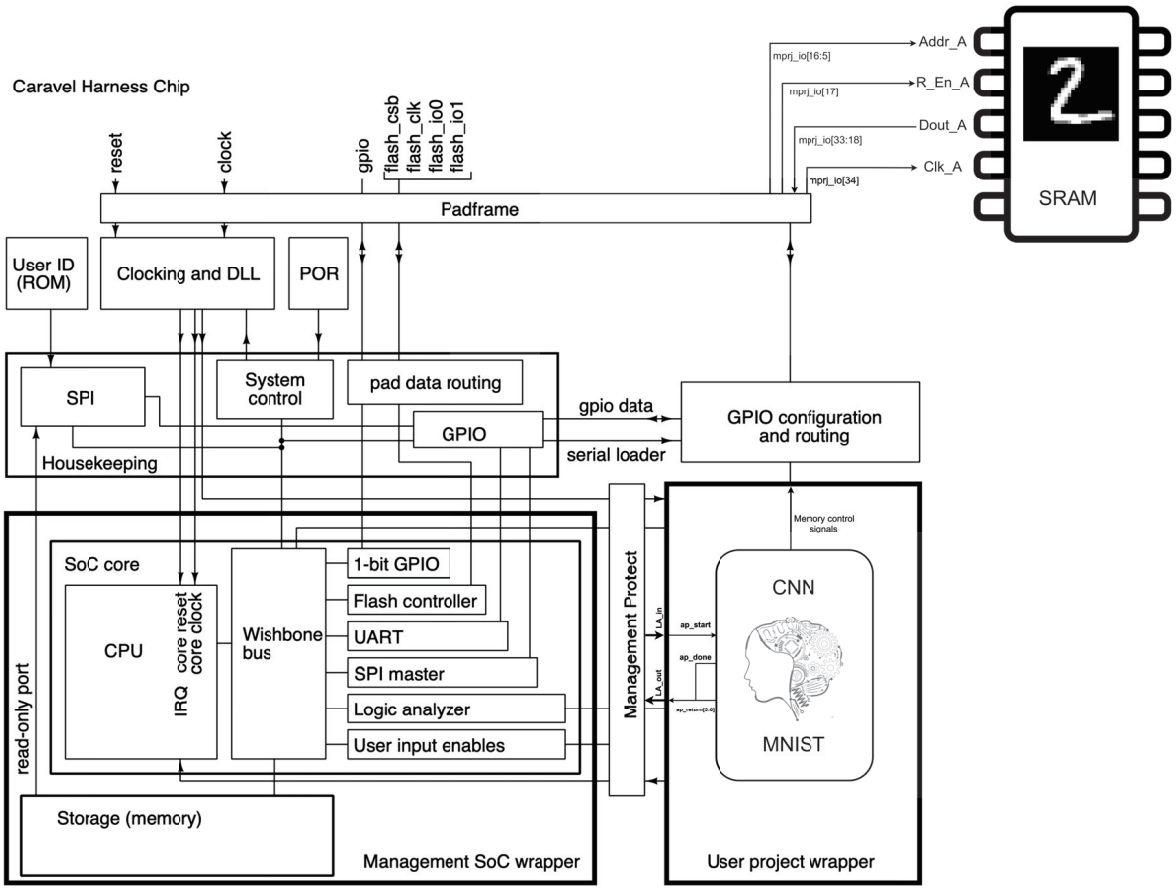


Figure 6. *AI by AI* and *Caravel* integration diagram.

5. Results

After establishing the connections between Caravel and the CNN, a testbench of the entire SoC was developed using the training data set to evaluate the performance of the CNN. Due to the RISC-V managing the SoC, some registers using C++ were configured to enable the utilization of LA ports, allowing communication between the CNN and the RISC-V processor, as well as GPIOs that enabled connectivity between the external SRAM and the SoC.

Figure 7 illustrates the SoC testbench, the image stored in memory, and the C++ code programmed in the RISC-V processor. The figure depicts the processor's handling of reset signals, start processes, the waiting period for the done signal, and the resulting inference values. After 1000 iterations, the system yields the same results as the HLS test, with an accuracy of 98%, proving that it works as intended.

Figure 7. Testbench of the IC *AI by AI* implemented with SKY130A standard cells.

Table 6 presents the layout specifications, with the SKY130A PDK, for the *AI by AI* system, including the gate count, die area, latency, maximum frequency, and power consumption.

Table 6. AI by AI layout specifications with the SKY130A PDK.

Parameter	Value
Core area	10.27 mm^2
Core Utility	8.747 mm^2
Cells per mm^2	26,241
Latency	161.19 K
Maximum frequency	40 MHz
Static Power	70.5 mW
Switching Power	50.5 mW
Buffers	65,142
Flip-Flops	49,973
Diode	33,839
Number of Cells	94,415

The outcome of the RTL to GDSII conversion process, along with its integration with the RISC-V made with Caravel, is visually presented in Figure 8. It illustrates two distinct areas: the user area, representing a flat implementation of the CNN, and the management area, housing the processor and its associated peripherals.

This project was the winner of the AI-generated design competition hosted by *Efabless*, which can be accessed at this link: https://efabless.com/genai/challenges/2-winners (accessed on 4 March 2024).Additionally, the CNN SoC is currently undergoing fabrication through the multi-project wafer shuttle CI 2309, which is available at https://platform.efabless.com/shuttles/CI%202309 (accessed on 4 March 2024).

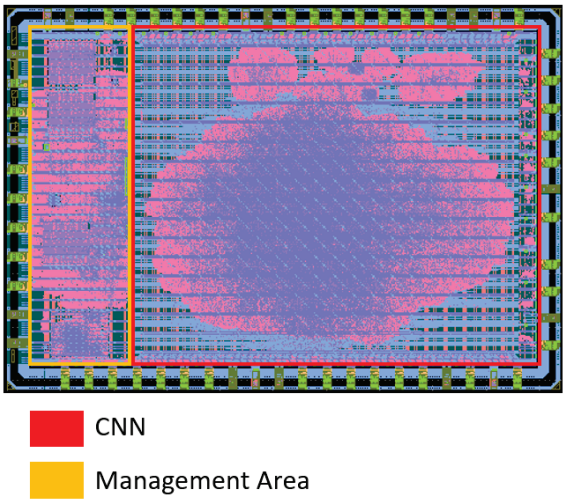


Figure 8. Caravel GDSII file and CNN layout details.

6. Discussion

The findings of this research highlight significant aspects, such as:

1. The current limitations of LLMs in generating HDL code.
2. Establishing a workflow that utilizes LLMs to generate and downscale systems from TF to HDL.
3. Introducing a new approach for converting HLS to GDSII using open-source PDKs and tools.
4. Achieving the fabrication of a CNN IC entirely created by AI.
5. Setting a precedent for current AI-generated systems by providing specific system information, such as core area, cells per square millimeter, latency, power consumption, number of flip-flops, and total number of cells.
6. Offering open-source access to the entire project, from the initial conversation with the AI to the final GDSII files generated.

These findings directly address our central research question, "are contemporary commercial LLMs capable of producing synthesizable and manufacturable CNN hardware designs using the first open-source PDKs (SKY130A)?",by providing new understanding and evidence that current commercial LLMs are not capable of directly creating a CNN in HDL; however, they are capable of creating synthesizable HLS code that can be used to generate IC with open-source tools. The paper elucidates the development of *AI by AI*, an innovative IC harnessing the power of AI. Our methodology involved the transformation of AI-generated TF code into Verilog, progressing through layout implementation and seamless integration with a RISC-V via Caravel. This process ultimately enabled us to propel *AI by AI* into the manufacturing phase through the ChipIgnite program.

AI by AI stands as a pioneering achievement, being the first CNN IC of its kind to be entirely conceptualized by AI and be fabricated with the open-source PDK SKY130A. Our approach harmoniously merges cutting-edge technologies, such as commercial LLMs, with more traditional ones like HLS and Verilog, creating an innovative workflow for developing intricate digital systems, particularly CNNs, and exploring the capacities of the current LLM. Frameworks like Caravel and multi-project wafer programs such as ChipIgnite have simplified and made cost-effective the layouts development and fabrication process.

While current commercial LLMs may not yet excel in rapidly and accurately producing Verilog and VHDL code, they have matured enough to proficiently handle programming tasks. The sequential transition from higher abstraction to lower abstraction languages, supplemented by tools like HLS, empowers us to generate functional Verilog code that seamlessly integrates into the silicon-level implementation process. This combination of technologies and methodologies has opened new horizons for AI-driven IC development.

7. Conclusions

AI is experiencing a boom in various sectors, including IC design. With LLMs such as ChatGPT, exploration in HDL generation has begun, which could reduce design costs by 31.72%—impacting prototyping, architecture, and verification phases, and compressing design timelines from months to weeks. Additionally, leveraging open-source tools and IPs could further reduce costs associated with software (43.32%) and IP (6.85%), respectively. Despite the potential, current LLMs have difficulties in producing complex HDLs systems with accurate performance, and open-source IPs are not as abundant as software libraries. Therefore, current research is focused on high-level languages such as Python and C++ to enable LLMs to efficiently create complex systems such as CNNs. HLS becomes crucial in this context for translating high-level code into HDL that, through the physical design flow, generates an IC that can be manufactured. The use of HLS causes some issues related to floating point operations, which can lead to a loss of accuracy and increased logical demands. If the loss of accuracy is significant, we recommend accessing the TF code and replicating in C++ the rounding algorithms it uses. This research establishes a benchmark for current LLM capabilities in ICs design, in particular for the design of CNNs, and is a point of comparison for evaluating future AI-generated ICs.

Author Contributions: Conceptualization, E.I.B.-L. and S.O.-C.; methodology, E.I.B.-L. and S.O.-C.; software, E.I.B.-L.; validation, E.I.B.-L., M.A., R.Y.V.M. and G.P.-D.; formal analysis, E.I.B.-L. and R.Y.V.M.; investigation, E.I.B.-L. and M.A.; resources, S.O.-C.; writing—original draft preparation, E.I.B.-L.; writing—review and editing, E.I.B.-L., S.O.-C., M.A., R.Y.V.M. and G.P.-D.; supervision, S.O.-C. and M.A.; project administration, E.I.B.-L. and S.O.-C.; funding acquisition, S.O.-C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data presented in this study are available or are mentioned in this paper.

Conflicts of Interest: Dr. Mohamed Abdelmoneum and Dr. Ruth Ruth Yadira Vidana Morales are employed by Intel Corporation. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Appendix A. Half Precision Floating Point Functions in C++

```
int16_t add(int16_t a, int16_t b) {
  if (a == 0) return b;
  if (b == 0) return a;
  int16_t signA = a >> 15;
  int16_t signB = b >> 15;
  int16_t expA = (a >> 10) & 0x1F;
  int16_t expB = (b >> 10) & 0x1F;
  int16_t mantissaA = a & 0x3FF;
  int16_t mantissaB = b & 0x3FF;
  int16_t i_loop;
  // Denormalize the mantissas
  mantissaA |= 0x400;
  mantissaB |= 0x400;
  // Align mantissas by shifting the one with the smaller exponent
  i_loop = 0;
  while (expA < expB) {
    mantissaA >>= 1;
    expA++;
    i_loop++;
```

```
  if(i_loop>50)
  {
          break;
  }
}
i_loop = 0;
while (expB < expA) {
  mantissaB >>= 1;
  expB++;
  i_loop++;
  if(i_loop>50)
  {
          break;
  }
}
// Perform add2ition or subtraction based on the signs
int32_t resultMantissa;
if (signA == signB) {
  resultMantissa = mantissaA + mantissaB;
} else if (signA) {
  resultMantissa = mantissaB - mantissaA;
} else {
  resultMantissa = mantissaA - mantissaB;
}
int16_t resultSign = (resultMantissa < 0) ? 1 : 0;
if (resultMantissa < 0) {
resultSign = 1;
resultMantissa = -resultMantissa;
} else {
  resultSign = 0;
}
// Normalize the result
i_loop = 0;
while (resultMantissa >= 0x800) {
  resultMantissa >>= 1;
  expA++;
  i_loop++;
  if(i_loop>50)
  {
          break;
  }
}
i_loop = 0;
while (resultMantissa < 0x400) {
  resultMantissa <<= 1;
  expA--;
  i_loop++;
  if(i_loop>50)
  {
          break;
  }
}
// Create the result
int16_t result = (resultMantissa & 0x3FF) | (expA << 10);
```

```
  if ((signA && signB) || resultSign) {
    result |= 0x8000;
  }
  return result;
}
int16_t multiply_custom_float2(int16_t a, int16_t b) {
    if (a == 0 || b == 0) return 0;
    // Extracting sign, exponent, and mantissa for 'a'
    int16_t sign_a = (a >> 15) & 1;
    int16_t exponent_a = (a >> 10) & (int16_t)0x1F;
    int16_t mantissa_a = a & (int16_t)0x3FF;

    // Extracting sign, exponent, and mantissa for 'b'
    int16_t sign_b = (b >> 15) & 1;
    int16_t exponent_b = (b >> 10) & (int16_t)0x1F;
    int16_t mantissa_b = b & (int16_t)0x3FF;

    // Calculating the result's sign, exponent, and mantissa
    int16_t sign_result = sign_a ^ sign_b;
    int16_t exponent_result = (exponent_a - 15) + (exponent_b - 15) + 15;
     ↪  // Remove bias, add2, then add2 bias back
    int32_t mantissa_result = (1024 + mantissa_a) * (1024 + mantissa_b);

    // Normalizing the mantissa
    if (mantissa_result >= (1 << 21)) {
        mantissa_result >>= 1;
        exponent_result += 1;
    }
    mantissa_result = (mantissa_result >> 10) - 1024; // Remove the
     ↪  implicit leading one
    // Check for underflow or overflow
    if (exponent_result < 0) return 0; // Underflow

    if (exponent_result >= 0x1F) return sign_result ? (int16_t)0x8000 :
     ↪  (int16_t)0x7FFF; // Overflow
    // Combining sign, exponent, and mantissa into a 16-bit integer
    int16_t result = (sign_result << 15) | ((exponent_result &
     ↪  (int16_t)0x1F) << 10) | (mantissa_result & (int16_t)0x3FF);
    return result;
}

int16_t divide_custom_float2(int16_t a, int16_t b) {
    int16_t sign_a = (a >> 15) & 1;
    int16_t exponent_a = (a >> 10) & (int16_t)0x1F;
    int16_t mantissa_a = (a & (int16_t)0x3FF) | (int16_t)0x400;
    int16_t sign_b = (b >> 15) & 1;
    int16_t exponent_b = (b >> 10) & (int16_t)0x1F;
    int16_t mantissa_b = (b & (int16_t)0x3FF) | (int16_t)0x400;
    if (mantissa_b == 0) return 0;
    int16_t sign_result = sign_a ^ sign_b;
    int16_t exponent_result = exponent_a - exponent_b + 15;
    int32_t remainder = mantissa_a << 10;
    int32_t divisor = mantissa_b << 10;
    int32_t quotient = 0;
```

```
    for (int i = 0; i < 10; i++) {
        remainder <<= 1;
        if (remainder >= divisor) {
            remainder -= divisor;
            quotient = (quotient << 1) | 1;
        } else {
            quotient <<= 1;
        }
    }
    if (quotient < (int16_t)0x400) {
        quotient <<= 1;
        exponent_result -= 1;
    }
    int16_t mantissa_result = quotient & (int16_t)0x3FF;
    int16_t result = (sign_result << 15) | ((exponent_result &
     ↪  (int16_t)0x1F) << 10) | mantissa_result;
    return result;
}
```

Appendix B. Top Module Verilog Code

```
module user_project_wrapper #(
    parameter BITS = 32
) (
    // Wishbone Slave ports (WB MI A)
    input wb_clk_i,
    input wb_rst_i,
    input wbs_stb_i,
    input wbs_cyc_i,
    input wbs_we_i,
    input [3:0] wbs_sel_i,
    input [31:0] wbs_dat_i,
    input [31:0] wbs_adr_i,
    output wbs_ack_o,
    output [31:0] wbs_dat_o,
    // Logic Analyzer Signals
    input  [127:0] la_data_in,
    output [127:0] la_data_out,
    input  [127:0] la_oenb,
    // IOs
    input  [`MPRJ_IO_PADS-1:0] io_in,
    output [`MPRJ_IO_PADS-1:0] io_out,
    output [`MPRJ_IO_PADS-1:0] io_oeb,
    inout [`MPRJ_IO_PADS-10:0] analog_io,
    // Independent clock (on independent integer divider)
    input   user_clock2,
    // User maskable interrupt signals
    output [2:0] user_irq
);
/*--------------------------------------*/
/* User project is instantiated  here   */
/*--------------------------------------*/
  wire in_ap_rst;
```

```
 wire _in_ap_start;
 wire _ap_done;
 wire _ap_idle;
 wire _ap_ready;
 wire [31:0] _image_r_Addr_A;
 wire _image_r_EN_A;
 wire [1:0] _image_r_WEN_A;
 wire [15:0] _image_r_Din_A;
 wire [15:0] i_in_image_r_Dout_A;
 wire   _image_r_Clk_A;
 wire   _image_r_Rst_A;
wire  [3:0] _ap_return;
assign in_ap_rst = la_data_in[1]|io_in[30+5];
assign io_oeb[30]=1;
assign  _in_ap_start = la_data_in[2];
assign la_data_out[3]=_ap_done;
assign  la_data_out[4]=_ap_idle;
assign  la_data_out[5]=_ap_ready;
assign  la_data_out[31:28] = _ap_return;

//External memory controls
 assign io_out[11+5:0+5] = _image_r_Addr_A[11:0]; //Addres
 assign io_oeb[11+5:0+5] = 11'b0;
 assign io_out[12+5] = _image_r_EN_A; //r_Enb
 assign io_oeb[12+5]=0;
 assign i_in_image_r_Dout_A = io_in[28+5:13+5]; //Data_input
  assign io_oeb[28+5:13+5]=16'hFFFF;
 assign io_out[29+5] = _image_r_Clk_A; //CLK
  assign io_oeb[29+5]=0;
 forward_pass AI_by_AI (
   .ap_clk(wb_clk_i),
   .ap_rst(in_ap_rst),
   .ap_start(_in_ap_start),
   .ap_done(_ap_done),
   .ap_idle(_ap_idle),
   .ap_ready(_ap_ready),
   .image_r_Addr_A(_image_r_Addr_A),
   .image_r_EN_A(_image_r_EN_A),
   .image_r_WEN_A(_image_r_WEN_A), //We just read the memory dont write
    ↪  in it
   .image_r_Din_A(_image_r_Din_A),
   .image_r_Dout_A(i_in_image_r_Dout_A),
   .image_r_Clk_A(_image_r_Clk_A),
   .image_r_Rst_A(_image_r_Rst_A),
   .ap_return(_ap_return)
 );
```

References

1. Bardeen, J.; Brattain, W.H. The transistor, a semi-conductor triode. *Phys. Rev.* **1948**, *74*, 230. [CrossRef]
2. Kilby, J.S.C. Turning potential into realities: The invention of the integrated circuit (Nobel lecture). *ChemPhysChem* **2001**, *2*, 482–489. [CrossRef] [PubMed]
3. Spitalny, A.; Goldberg, M.J. On-line operation of CADIC (computer aided design of integrated circuits). In Proceedings of the 4th Design Automation Conference, Los Angeles, CA, USA, 19–22 June 1967; pp. 7-1–7-20.

4. Barbacci, M. A Comparison of Register Transfer Languages for Describing Computers and Digital Systems. *IEEE Trans. Comput.* **1975**, *C-24*, 137–150. [CrossRef]
5. Bell, C.G.; Grason, J.; Newell, A. *Designing Computers and Digital Systems Using PDP 16 Register Transfer Modules*; Digital Press: Los Angeles, CA, USA, 1972.
6. Barbacci, M.R.; Barnes, G.E.; Cattell, R.G.G.; Siewiorek, D.P. *The ISPS Computer Description Language: The Symbolic Manipulation of Computer Descriptions*; Departments of Computer Science and Electrical Engineering, Carnegie-Mellon University: Pittsburgh, PA, USA, 1979.
7. Barbacci, M.R. *The Symbolic Manipulation of Computer Descriptions: ISPL Compiler and Simulator*; Carnegie Mellon University, Department of Computer Science: Pittsburgh, PA, USA, 1976.
8. Huang, C.L. Method and Apparatus for Verifying Timing during Simulation of Digital Circuits. U.S. Patent 5,095,454, 10 March 1992.
9. Shahdad, M.; Lipsett, R.; Marschner, E.; Sheehan, K.; Cohen, H. VHSIC hardware description language. *Computer* **1985**, *18*, 94–103. [CrossRef]
10. Gupta, R.; Brewer, F. High-level synthesis: A retrospective. In *High-Level Synthesis: From Algorithm to Digital Circuit*; Springer: Dordrecht, The Netherlands, 2008; pp. 13–28.
11. Minsky, M.; Papert, S. *Perceptrons: An Introduction to Computational Geometry*; Massachusetts Institute of Technology: Cambridge, MA, USA, 1969; Volume 479, p. 104.
12. Rumelhart, D.E.; Hinton, G.E.; Williams, R.J. Learning representations by back-propagating errors. *Nature* **1986**, *323*, 533–536. [CrossRef]
13. Tesauro, G. TD-Gammon, a self-teaching backgammon program, achieves master-level play. *Neural Comput.* **1994**, *6*, 215–219. [CrossRef]
14. Dean, J. 1.1 The Deep Learning Revolution and Its Implications for Computer Architecture and Chip Design. In Proceedings of the 2020 IEEE International Solid-State Circuits Conference—(ISSCC), San Francisco, CA, USA, 16–20 February 2020; pp. 8–14. [CrossRef]
15. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Li, F.-F. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 248–255.
16. Luebke, D.; Harris, M. General-purpose computation on graphics hardware. In Proceedings of the Workshop, SIGGRAPH, Los Angeles, CA, USA, 8–12 August 2004; Volume 33, p. 6.
17. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
18. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1097–1105. [CrossRef]
19. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
20. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
21. Tan, M.; Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In Proceedings of the International Conference on Machine Learning, PMLR, Long Beach, CA, USA, 9–15 June 2019; pp. 6105–6114.
22. Hinton, G.; Deng, L.; Yu, D.; Dahl, G.E.; Mohamed, A.r.; Jaitly, N.; Senior, A.; Vanhoucke, V.; Nguyen, P.; Sainath, T.N.; et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Process. Mag.* **2012**, *29*, 82–97. [CrossRef]
23. Chan, W.; Jaitly, N.; Le, Q.V.; Vinyals, O. Listen, attend and spell. *arXiv* **2015**, arXiv:1508.01211.
24. Wu, Y.; Schuster, M.; Chen, Z.; Le, Q.V.; Norouzi, M.; Macherey, W.; Krikun, M.; Cao, Y.; Gao, Q.; Macherey, K.; et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv* **2016**, arXiv:1609.08144.
25. Collobert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; Kuksa, P. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.* **2011**, *12*, 2493–2537.
26. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.S.; Dean, J. Distributed representations of words and phrases and their compositionality. *Adv. Neural Inf. Process. Syst.* **2013**, *26*, 3111–3119.
27. Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to sequence learning with neural networks. *Adv. Neural Inf. Process. Syst.* **2014**, *27*, 3104–3112.
28. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 5998–6008.
29. Shazeer, N.; Mirhoseini, A.; Maziarz, K.; Davis, A.; Le, Q.; Hinton, G.; Dean, J. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv* **2017**, arXiv:1701.06538.
30. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
31. Yu, C.; Xiao, H.; De Micheli, G. Developing synthesis flows without human knowledge. In Proceedings of the 55th Annual Design Automation Conference, San Francisco, CA, USA, 24–29 June 2018; pp. 1–6.

32. Huang, G.; Hu, J.; He, Y.; Liu, J.; Ma, M.; Shen, Z.; Wu, J.; Xu, Y.; Zhang, H.; Zhong, K.; et al. Machine learning for electronic design automation: A survey. *ACM Trans. Des. Autom. Electron. Syst.* **2021**, *26*, 1–46. [CrossRef]
33. Kahng, A.B. Machine learning applications in physical design: Recent results and directions. In Proceedings of the 2018 International Symposium on Physical Design, Monterey, CA, USA, 25–28 March 2018; pp. 68–73.
34. OpenAI. Introducing ChatGPT. 2022. Available online: https://openai.com/blog/chatgpt (accessed on 8 February 2024).
35. Pichai, S. An Important Next Step on Our AI Journey. 2023. Available online: https://blog.google/technology/ai/bard-google-ai-search-updates/ (accessed on 8 February 2024).
36. Microsoft. Microsoft Edge Features—Bing Chat. 2023. Available online: https://www.microsoft.com/en-us/edge/features/bing-chat?form=MT00D8 (accessed on 8 February 2024).
37. Chang, K.; Wang, Y.; Ren, H.; Wang, M.; Liang, S.; Han, Y.; Li, H.; Li, X. ChipGPT: How far are we from natural language hardware design. *arXiv* **2023**, arXiv:2305.14019.
38. Thakur, S.; Ahmad, B.; Fan, Z.; Pearce, H.; Tan, B.; Karri, R.; Dolan-Gavitt, B.; Garg, S. Benchmarking Large Language Models for Automated Verilog RTL Code Generation. In Proceedings of the 2023 Design, Automation & Test in Europe Conference & Exhibition (DATE), Antwerp, Belgium, 17–19 April 2023; IEEE: Piscataway, NJ, USA, 2023; pp. 1–6.
39. Blocklove, J.; Garg, S.; Karri, R.; Pearce, H. Chip-Chat: Challenges and Opportunities in Conversational Hardware Design. *arXiv* **2023**, arXiv:2305.13243.
40. Efabless. Efabless Caravel "Harness" SoC—Caravel Harness Documentation. Available online: https://caravel-harness.readthedocs.io/en/latest/ (accessed on 8 February 2024).
41. Welcome to SkyWater SKY130 PDK's Documentation! Available online: https://skywater-pdk.readthedocs.io/en/main/ (accessed on 8 February 2024).
42. Srilakshmi, S.; Madhumati, G.L. A Comparative Analysis of HDL and HLS for Developing CNN Accelerators. In Proceedings of the 2023 Third International Conference on Artificial Intelligence and Smart Energy (ICAIS), Coimbatore, India, 2–4 February 2023; pp. 1060–1065.
43. Zhao, J.; Zhao, Y.; Li, H.; Zhang, Y.; Wu, L. HLS-Based FPGA Implementation of Convolutional Deep Belief Network for Signal Modulation Recognition. In Proceedings of the IGARSS 2020—2020 IEEE International Geoscience and Remote Sensing Symposium, Waikoloa, HI, USA, 26 September–2 October 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 6985–6988.
44. Lee, H.S.; Jeon, J.W. Comparison between HLS and HDL image processing in FPGAs. In Proceedings of the 2020 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia), Seoul, Republic of Korea, 1–3 November 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–2.
45. Shalan, M.; Edwards, T. Building OpenLANE: A 130nm openroad-based tapeout-proven flow. In Proceedings of the 39th International Conference on Computer-Aided Design, San Diego, CA, USA, 2–5 November 2020; pp. 1–6.
46. Zezin, D. Modern Open Source IC Design tools for Electronics Engineer Education. In Proceedings of the 2022 VI International Conference on Information Technologies in Engineering Education (Inforino), Moscow, Russia, 12–15 April 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 1–4.
47. Charaan, S.; Nalinkumar, S.; Elavarasan, P.; Prakash, P.; Kasthuri, P. Design of an All-Digital Phase-locked loop in a 130 nm CMOS Process using open-source tools. In Proceedings of the 2022 International Conference on Electronic Systems and Intelligent Computing (ICESIC), Chennai, India, 22–23 April 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 270–274.
48. Ghazy, A.; Shalan, M. Openlane: The open-source digital asic implementation flow. In Proceedings of the Workshop on Open-Source EDA Technologies (WOSET), Online, 27 October 2020.
49. Chupilko, M.; Kamkin, A.; Smolov, S. Survey of Open-source Flows for Digital Hardware Design. In Proceedings of the 2021 Ivannikov Memorial Workshop (IVMEM), Nizhny Novgorod, Russia, 24–25 September 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 11–16.
50. Hesham, S.; Shalan, M.; El-Kharashi, M.W.; Dessouky, M. Digital ASIC Implementation of RISC-V: OpenLane and Commercial Approaches in Comparison. In Proceedings of the 2021 IEEE International Midwest Symposium on Circuits and Systems (MWSCAS), Lansing, MI, USA, 9–11 August 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 498–502.
51. Efabless. Homepage. Available online: https://efabless.com/ (accessed on 26 February 2024).
52. Deng, L. The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]. *IEEE Signal Process. Mag.* **2012**, *29*, 141–142. [CrossRef]
53. Bisong, E. Google colaboratory. In *Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners*; Apress: Berkeley, CA, USA, 2019; pp. 59–64.

electronics

MDPI

Article

Extension of Interval-Valued Hesitant Fermatean Fuzzy TOPSIS for Evaluating and Benchmarking of Generative AI Chatbots

Galina Ilieva

Department of Management and Quantitative Methods in Economics, University of Plovdiv Paisii Hilendarski, 4000 Plovdiv, Bulgaria; galili@uni-plovdiv.bg

Abstract: To aid in the selection of generative artificial intelligence (GAI) chatbots, this paper introduces a fuzzy multi-attribute decision-making framework based on their key features and performance. The proposed framework includes a new modification of the Technique for Order Preference by Similarity to Ideal Solution (TOPSIS), adapted for an interval-valued hesitant Fermatean fuzzy (IVHFF) environment. This TOPSIS extension addresses the limitations of classical TOPSIS in handling complex and uncertain data capturing detailed membership degrees and representing hesitation more precisely. The framework is applicable for both static and dynamic evaluations of GAI chatbots in crisp or fuzzy assessments. Results from a practical example demonstrate the effectiveness of the proposed approach for comparing and ranking GAI chatbots. Finally, recommendations are provided for selecting and implementing these conversational agents in various applications.

Keywords: MCDM; interval-valued hesitant Fermatean fuzzy (IVHFF) sets; distance measure; Technique for Order Preference by Similarity to Ideal Solution (TOPSIS); IVHFFSs TOPSIS; generative AI chatbots; evaluation framework; benchmarking criteria

1. Introduction

Generative Artificial Intelligence (GAI) chatbots, also known as conversational agents, are becoming an increasingly prevalent type of chatbot worldwide for several reasons. Advancements in AI and natural language processing have significantly enhanced their capabilities [1]. These intelligent chatbots leverage large language models (LLMs) to generate human-like responses in natural language, enabling more dynamic and contextually appropriate interactions compared to their traditional rule-based predecessors [2,3].

Additionally, the rise of digital communication platforms and the growing demand for instant customer service have driven businesses to adopt GAI chatbots. These systems provide a cost-effective way to deliver customer support, answer queries, and even facilitate transactions.

The COVID-19 pandemic further accelerated the adoption of remote work and virtual communication, increasing the demand for AI chatbots [4]. They have proven instrumental in managing the surge of online interactions, such as handling customer inquiries, scheduling appointments, and disseminating information.

The flexibility and scalability of GAI chatbots make them suitable for diverse industries, including construction [5], healthcare [6], finance [7], and e-commerce [8], but each industry may face unique regulatory, operational, or technological constraints. By tailoring chatbots to the specific requirements and integrating them into existing systems, organizations in many sectors can potentially enhance productivity and user experiences [9].

Market research predictions confirm a growing adoption of AI chatbots in the coming years. According to a Statista forecast [10], the global chatbot market is projected to reach approximately USD 1.25 billion by 2025—a nearly fivefold increase from USD 190.8 million in 2016. Meanwhile, Gartner estimated that over 80% of enterprises will leverage GAI APIs or applications by 2026 [11], highlighting the rapid and widespread adoption of advanced AI technologies for enhancing business efficiency, innovation, and customer experiences. However, this growth also presents challenges in areas such as data privacy, ethics, and addressing skill gaps.

Despite their growing popularity, GAI chatbots face several challenges to widespread adoption. Key obstacles include the following:

- Lack of trust: Users may hesitate to fully trust AI-powered chatbots, especially when dealing with sensitive information or complex interactions. Building trust in the accuracy, security, and reliability of these systems is critical for their broader acceptance;
- Limited understanding and awareness: Many users are unfamiliar with the capabilities and benefits that GAI chatbots can provide. This lack of knowledge or understanding about how they function and what they offer may hinder adoption;
- User experience and satisfaction: Poorly designed chatbots can lead to unsatisfactory user experiences. Frustrating interactions or failure to resolve queries effectively may discourage continued use;
- Cost and ROI: Developing and maintaining GAI chatbots can be expensive for small- and medium-sized enterprises. Organizations must carefully assess the return on investment (ROI) and weigh costs against potential benefits;
- Ethical and bias concerns: GAI chatbots are only as reliable and fair as the data they are trained on, which can sometimes perpetuate biases or unfair practices. Ensuring chatbots are ethical, unbiased, and inclusive is important for their acceptance and broader implementation.

Overcoming these barriers will require advancements in technology, increased transparency, education, and a focus on user-centric design. To address the first three challenges, multi-criteria decision-making (MCDM) methods can be employed. These techniques enable organizations to compare a finite set of decision alternatives across various criteria, helping them select the most feasible option. MCDM methods have been successfully applied in several GAI-related fields, such as technology selection [12] and cloud system prioritization [13].

While conventional MCDM methods are reliable, they often struggle to address the complexities associated with imprecise and ambiguous evaluations. In contrast, fuzzy-based methods are specifically designed to manage such uncertainties, making them more effective in identifying the most suitable alternatives.

Various MCDM techniques have been enhanced through the integration of fuzzy sets and their advanced extensions [14]. By incorporating fuzzy assessments, these methods provide a more accurate representation of real-world conditions, thereby improving the reliability of rankings in scenarios characterized by subjectivity and evaluation uncertainties.

The key advantage of fuzzy multi-criteria algorithms lies in their ability to produce more realistic and dependable rankings, enhancing the overall decision-making process.

Key contributions of this paper include the following:

1. Analysis and categorization of existing multi-criteria approaches for AI chatbot selection, classified by the techniques used and the types of estimates employed (numeric, interval, linguistic values, as well as fuzzy numbers). These approaches are then grouped into three main categories based on complexity (number of multi-criteria techniques), flexibility (type of fuzziness), and iterativeness (single or repeated data processing);

2. Development of a theoretical framework for ranking GAI chatbots using both single and hybrid methods with crisp and fuzzy estimates. Single methods rely on one weight determination or ranking technique, while hybrid methods integrate several. The framework also incorporates complementary capabilities, including evaluations using crisp or fuzzy numbers, statistical analyses, and ranking interpretation, to enhance the decision-making process. Additionally, it introduces a newly developed 3D distance metric to enhance the effectiveness of the Fermatean fuzzy group TOPSIS method in case of hesitant interval assessments for more precise and effective multi-criteria comparisons of chatbot features;
3. Creation of static and dynamic rankings of an AI chatbot dataset via single or repeated multi-criteria decision analysis. In static rankings, experts' opinions serve as inputs for the decision matrices, whereas dynamic rankings measure user attitudes based on behavior or survey data. Comparative analyses with other multi-criteria baselines underscore both the effectiveness and reliability of the proposed methods.

The paper begins with a literature review in Section 2, discussing the motivation behind exploring fuzzy ranking for GAI chatbots. Next, Section 3 details the proposed theoretical decision-making framework for GAI chatbot selection, emphasizing the role of interval-valued hesitant Fermatean fuzzy numbers (IVHFFNs) and a modified TOPSIS method tailored for the IVHFF environment. Practical examples and result analysis are provided in Section 4, showcasing the application of the framework. The final section concludes the research by summarizing the key findings, offering insights, and proposing directions for future studies.

2. Related Work

2.1. Literature Review on MCDM Methods for GAI Chatbot Evaluation

GAI chatbots, despite being a relatively recent development, have garnered significant attention in both academic research and practical applications. Approaches to their study vary widely: Some researchers focus on technical aspects, offering descriptive or general analyses that often emphasize feature comparisons while omitting advanced computational methods. Conversely, other studies adopt modern model-driven techniques, such as machine learning, optimization, and MCDM methods.

MCDM methods present distinct advantages in the evaluation and selection of AI chatbots. One key benefit is that they do not rely on extensive datasets or computationally intensive procedures, making them accessible and efficient. These methods simplify the decision-making process by facilitating comprehensive evaluations across multiple criteria, ensuring objectivity through a systematic analysis of both the criteria and stakeholder preferences.

Additionally, MCDM approaches are well-suited for diverse decision-making scenarios and can manage the complexities inherent in chatbot evaluations. By incorporating stakeholder preferences, these methods enable informed decision making and improve the likelihood of selecting the most appropriate chatbot for a given context.

Drawing on data from previous studies, interviews, questionnaires, and surveys, Chakrabortty et al. [15] constructed a comparison matrix with eight alternatives and nine criteria: empathy, engagement, tangibility, assurance, reliability, satisfaction, responsiveness, speed, and security. These criteria were derived from established service quality models alongside AI- and chatbot-specific considerations. A survey was conducted to gather expert opinions and the single-valued neutrosophic (SVN) analytic hierarchy process (AHP) was employed to determine their relative weights. The combined compromise solution (CoCoSo) method was then used within the SVN environment to rank the options, ultimately identifying the optimal chatbot.

Santa Barletta et al. [16] proposed a novel clinical chatbot selection model using the AHP technique, assessing chatbot assistants based on the "Quality in Use" concept from the ISO/IEC 25010 standard [17]. Two healthcare-oriented chatbots were evaluated against five criteria groups: effectiveness, efficacy, satisfaction, freedom from risk, and context coverage across three dimensions—providing information, prescriptions, and process management.

Singh et al. [18] identified twelve acceptance factors for conversational digital assistants (CDAs) through a literature review and expert input. These factors were analyzed for their cause-and-effect relationships using the grey-DEcision-MAking Trial and Evaluation Laboratory (DEMATEL) method. The study highlighted key causal factors, including humanness, social influence, social presence, social capability, and ease of use, which significantly impact CDA adoption and provide insights for managerial and policy decisions in online shopping contexts.

Pandey et al. [19] addressed concerns about the impact of GAI tools, particularly ChatGPT, by examining twelve challenges related to its adoption. These challenges were analyzed using the intuitionistic fuzzy DEMATEL approach, which proved more effective than classical and fuzzy DEMATEL methods in terms of mean absolute error (MAE). By categorizing challenges into cause-and-effect relationships, the study provides valuable guidance for experts and project managers in identifying areas for improvement.

Pathak and Bansal [20] mapped twenty factors to the Technology–Organization–Environment–Individual (T-O-E-I) framework, derived from the Technology–Organization–Environment (T-O-E) and Human–Organization–Technology fit (H-O-T fit) frameworks. After ranking these factors, the global ranking was computed using the rough stepwise weight assessment ratio analysis method (R-SWARA). The top seven factors included perceived benefits of AI, AI system capabilities, organizational data ecosystem, perceived compatibility of AI systems, ease of use, IT infrastructure, and top management support. Sensitivity analysis confirmed the robustness of these rankings.

Wiangkham and Vongvit [21] applied both MCDM and artificial neural network (ANN) methods to prioritize factors influencing ChatGPT adoption in higher education. Fourteen criteria were grouped into usage-, agent-, technical- and trust-related categories. Using a Likert-scale questionnaire, criteria importance was assessed, and weighted sum model (WSM) and ANN methods were applied, alongside SHapley Additive exPlanations (SHAP) and Local Interpretable Model-agnostic Explanations (LIME). The study systematically prioritized factors affecting ChatGPT adoption.

Ojo et al. [22] employed a fuzzy TOPSIS-based method to evaluate six AI alternatives for mental health treatment planning: rule-based systems, logistic regression, neural networks, evolutionary algorithms, hybrid models, and benchmark algorithms. The evaluation considered criteria such as privacy protection, treatment effectiveness, explainability, healthcare costs, regulatory compliance, and ethical implications. Rule-based systems and benchmark algorithms emerged as the preferred approaches.

The key characteristics of the methodologies used to investigate factors influencing the selection and ranking of conversational digital assistants are summarized in Table 1.

After analysis of previous studies on GAI chatbot selection, we categorize them based on their distinctive features. According to the specificity of input data, the utilized models can be divided into two groups: crisp and fuzzy estimates. The crisp group, exemplified by Santa Barleta et al. [16] and Wiangkham and Vongvit [21], is designed for arithmetic calculations and distance metrics using precise input values. In contrast, the fuzzy group includes multi-criteria methods operating in various fuzzy environments, as demonstrated by Chakrabortty et al. [15], Singh et al. [18], Pandey et al. [19], Pathak and Bansal [20], and Ojo et al. [22].

Table 1. Comparison of existing studies on chatbots evaluation and ranking.

Reference	Methodology	Application Area	Alternatives	Criteria (Number)	Ranking Validation
Chakrabortty et al., 2023 [15]	SVN AHP-CoCoSo	Telecommunication business	Eight chatbots for customer service	Security, speed, responsiveness, satisfaction, reliability, assurance, tangibility, engagement, empathy (9)	SVN MABAC, Pythagorean fuzzy CoCoSo, and interval-valued neutrosophic TOPSIS
Santa Barleta et al., 2023 [16]	AHP	Medical care	Two clinical chatbots	Effectiveness, efficacy, satisfaction, freedom from risk, context coverage in three functional dimensions (five criteria groups)	Superdecisions software v.3.2
Singh et al., 2023 [18]	Grey-DEMATEL	Online retail	Only criteria weights	Social influence, enjoyment, performance, ease of use, usefulness, social presence, anxiety, trust, rapport, privacy risk, social isolation, sense of control, compatibility (12)	Sensitivity analysis in three scenarios
Pandey et al., 2024 [19]	Intuitionistic fuzzy DEMATEL	GAI chatbots challenges	Only criteria weights	Hallucination *, bias, language learning, real-world harm *, proprietary LLMs *, AI problems *, disruption, jobs at risk, educational system problems, training data amount *, unknown threats, ethical and legal implications (12)	MAE of issues using classical DEMATEL and fuzzy DEMATEL
Pathak and Bansal, 2024 [20]	Rough SWARA	Insurance	Only criteria weights	Technology (7), organization (6), environment (3), individual (4) criteria groups	Sensitivity analysis
Wiangkham and Vongvit, 2024 [21]	WSM, ANN with SHAP and LIME	Higher education	Only criteria weights	Usage- (4), agent- (3), technical- (4), trust (3)-related criteria groups	WMAPE for ANN models
Ojo et al., 2024 [22]	Fuzzy triangular TOPSIS	Medical care	Six AI alternatives	Privacy protection, treatment effectiveness, explainability, costs, regulatory compliance, ethical implications (6)	Comparative analysis

Remark: The symbol * denotes the most impactful factors for ChatGPT adoption.

In terms of complexity, existing models can be classified into single and hybrid multi-criteria techniques. Single methods, used by Singh et al. [18], Pandey et al. [19], Pathak and Bansal [20], Wiangkham and Vongvit [21], and Ojo et al. [22], apply only one MCDM method. Hybrid approaches, employed by Chakrabortty et al. [15] and Santa Barleta et al. [16], combine two methods: one for determining relative criteria weights and another for ranking alternatives.

The literature review reveals the absence of a universal approach for addressing the GAI chatbot selection problem. While previous studies offer valuable insights into comparing conversational chatbots, they exhibit several shortcomings:

- Lack of holistic multi-criteria solutions: Many proposed solutions focus on specific aspects, such as determining the relative importance of features within the criteria system [18–21] or generating chatbot rankings using a single multi-criteria method [15,22];
- Limited handling of inaccurate attribute estimates: Few studies, such as those by Chakrabortty et al. [15], Pandey et al. [19], and Ojo et al. [22], effectively address imprecise attribute estimates. Since AI chatbot evaluations often depend on subjective factors, assessments should involve expert groups utilizing classic fuzzy numbers or their advanced variants;
- Non-iterative fuzzy solutions: Existing fuzzy methodologies typically implement only one or two MCDM methods in a single, non-iterative procedure.

Evaluation should adopt a holistic process that considers various factors, including technological, economic, and organizational parameters, which are often expressed through imprecise, unclear, and uncertain estimates. To address these drawbacks, we propose a new fuzzy methodology for GAI chatbot selection.

Selecting a specific chatbot assistant aligned with organizational strategies or individual preferences is a complex process influenced by numerous factors. At the organizational level, the preferred intelligent chatbot depends on considerations such as data security requirements, regulatory compliance, subscription costs, and seamless integration with existing systems. At the individual level, preferences may be shaped by use cases, ease of use, domain-specific capabilities, or community recommendations and reviews. The optimal solution is the GAI chatbot that best meets the requirements of the organization or the preferences of the individual user.

2.2. Chatbot Evaluation Criteria

Despite the availability of practical tools and platforms for chatbot benchmarking and user testing—such as those offered by Hugging Face, Chatbot Arena (formerly LMSYS) [23], and Artificial Analysis [24]—these solutions often lack the flexibility needed to accommodate specific study goals and use cases. Evaluating GAI chatbots requires a systematic approach that integrates diverse attributes to address their multifaceted roles and applications.

In this subsection, we review the criteria proposed in prior studies to identify relevant attributes for developing a multi-attribute evaluation system specifically tailored to GAI chatbots.

The literature review reveals that previous studies on developing multi-criteria systems for evaluating GAI chatbots have primarily adopted a combined approach, integrating multiple criteria, indices, and metrics derived from various theoretical models and software quality standards.

For example, Chakrabortty et al. proposed a system based on nine criteria: security, speed, responsiveness, satisfaction, reliability, assurance, tangibility, engagement, and empathy. These criteria were drawn from SERVQUAL [25] (responsiveness, reliability, assurance, tangibility, and empathy), ISO/IEC 25010 [17] (security and speed), the technology acceptance model (TAM) [26] (engagement), and customer experience theory [27] (satisfaction) [15].

Santa Barleta et al. focused on five criteria groups: effectiveness, efficacy, satisfaction, freedom from risk, and context coverage. These were derived from ISO/IEC 25010 [17] and applied across three functional dimensions [16].

Singh et al. [18] developed a system incorporating 12 criteria, including social influence, enjoyment, performance, ease of use, usefulness, trust, and privacy risk, based on TAM [26] and UTAUT [28].

Pandey et al. [19] introduced 12 evaluation criteria emphasizing ChatGPT-related issues such as hallucination, bias, proprietary LLMs, ethical implications, and broader AI-related problems.

Pathak and Bansal [20] utilized the T-O-E-I framework [29,30], organizing criteria into four groups: technology (seven criteria), organization (six), environment (three), and individual (four).

Wiangkham and Vongvit [21] adopted a system comprising usage (four criteria), agent (three), technical (four), and trust-related (three) categories, primarily based on TAM and UTAUT.

Ojo et al. [22] focused on six criteria: privacy protection, treatment effectiveness, explainability, costs, regulatory compliance, and ethical implications. These criteria, designed

for evaluating medical chatbots, stem from healthcare technology frameworks, ISO standards, AI ethics, and health economics models. Their system ensures that medical chatbots are safe, effective, transparent, and legally compliant while addressing critical aspects such as patient data security, cost effectiveness, and ethical concerns.

The compared evaluation systems for GAI chatbot selection emphasize a multi-criteria approach, integrating elements from SERVQUAL framework, ISO standards, TAM, UTAUT, and AI ethics models. These assessment indices are designed to address specific contexts, including functionality, user experience, and ethical considerations, enabling effective comparisons by evaluating both technical capabilities and societal impacts.

However, existing evaluation systems have limitations, including their domain-specific focus, insufficient attention to rapidly evolving GAI challenges, and reliance on subjective criteria weighting. To address these gaps, we developed a GAI chatbot evaluation system, ensuring a comprehensive and holistic evaluation approach.

The proposed system includes four key criteria: conversational ability, user experience, integration capability, and price:

- Conversational ability evaluates the chatbot's capacity to understand and generate natural language responses, ensuring context-aware, coherent, and human-like interactions;
- User experience measures ease of use, intuitiveness, and satisfaction, focusing on design, accessibility, and the chatbot's ability to meet user needs effectively;
- Integration capability assesses how seamlessly the chatbot integrates with existing tools, platforms, or workflows, enhancing usability and productivity;
- Price considers the affordability of the chatbot, evaluating its cost relative to its features, functionality, and overall value.

Our evaluation system aligns with the TAM [26] and UTAUT [28] models. Conversational ability corresponds to perceived ease of use in TAM and performance expectancy in UTAUT, reflecting user expectations for accurate, natural communication. User experience relates to perceived usefulness and effort expectancy, where intuitive and enjoyable interactions drive adoption. Integration capability aligns with facilitating conditions in UTAUT and external variables in TAM, as compatibility with existing systems enhances utility. The price criterion captures the cost–value relationship, where users weigh the chatbot's cost against its utility and benefits.

TAM- and UTAUT-based indicators have been preferred because they effectively capture user perceptions and behavioral intentions towards adopting GAI chatbots across diverse contexts. Their theoretical foundations and empirical validation make them more reliable measures than those from other existing approaches.

The new chatbot evaluation system adopts a combined approach by integrating factors from two widely used theoretical models. This multidimensional system provides a complex assessment that addresses functional, experiential, technical, and economic dimensions. By tailoring it to the specific requirements of corporate and individual users, our approach ensures an effective evaluation of chatbots across varied use cases and priorities.

2.3. State-of-the-Art of the Most Widely Used GAI Chatbots

In this subsection, we present a comparative overview of the most popular GAI chatbots recognized by the global AI community for their transformative role in enhancing human–machine interaction: ChatGPT, Copilot, Gemini, Claude, and Perplexity AI.

OpenAI ChatGPT (https://chatgpt.com, accessed on 29 January 2025) is a state-of-the-art GAI chatbot renowned for its advanced conversational capabilities. Powered by generative pre-trained transformer (GPT) models, it excels in understanding users' requests and generating human-like responses, making it suitable for a wide range of applications, from casual conversations to professional tasks. The chatbot's intuitive interface and

versatility have made it widely adopted. It offers features like text summarization, content generation, and creative writing. Available in free and premium versions, ChatGPT is accessible to individuals, educators, and businesses alike [31]. Recent enhancements include the launch of ChatGPT Pro, which provides unlimited access to advanced models such as GPT-o1 and GPT-4o, along with features like Advanced Voice Mode. OpenAI has also expanded ChatGPT's functionality to include web-based search capabilities for up-to-date information. Additional updates include the introduction of the Projects tool, which simplifies managing multiple chats and group files, and Canvas, an interface for collaborative writing and coding.

Microsoft Copilot is a GAI-powered assistant integrated into the Microsoft 365 ecosystem, designed to enhance productivity across office tools. Built on OpenAI's LLM models, it provides contextual suggestions, automates repetitive tasks, and supports content generation tailored to user needs [32]. Recent updates include general availability for Microsoft 365 Copilot, the introduction of Windows Copilot with OS integration, enhancements to GitHub Copilot (Copilot Chat), and ongoing improvements in Microsoft products like Dynamics 365 and the Power Platform. These updates offer intuitive assistance with writing code, analyzing data, generating content, and automating routine tasks.

Google Gemini (https://gemini.google.com, accessed on 29 January 2025), formerly Bard, is a GAI chatbot that combines conversational AI with the capability of using Google's search engine. It delivers accurate, contextually relevant answers and supports tasks such as brainstorming, drafting, and question answering. Integrated into Google's ecosystem, Gemini works with tools like Google Workspace, making it a reliable assistant for personal and professional use [33]. Recent advancements include access to experimental models like Gemini Exp-1206, designed for complex tasks such as coding, solving mathematics problems, reasoning, and instruction following. The Gemini 2.0 Flash model improves academic benchmarks and speed, while Gemini Deep Research offers a personal research assistant capable of generating comprehensive reports. Additionally, new Gems for Google Workspace enhance workflow efficiency, and the Gemini app now provides enterprise-grade data protection for business and education customers.

Anthropic Claude (https://claude.ai, accessed on 29 January 2025) is an AI chatbot designed to deliver safe, ethical, and contextually aware conversations. Claude handles complex queries and supports tasks like content creation and data analysis for personal and professional use. Its user friendliness and accessibility have made it popular, especially in educational and research settings [34]. In 2024, Anthropic introduced the upgraded Claude 3.5 Sonnet model, enhancing capabilities in coding, reasoning, and instruction following. The Claude 3.5 Haiku model offers state-of-the-art performance with improved speed and affordability. Additionally, a new "computer use" feature enables Claude to interact with computer interfaces, automating tasks by simulating human actions like moving a cursor, clicking UI buttons and typing text.

Perplexity AI (https://www.perplexity.ai/, accessed on 29 January 2025) is a search-driven chatbot that combines GAI with real-time information retrieval to generate concise and accurate answers. Known for its minimalistic interface and focus on transparency, Perplexity is relatively inexpensive, making it appealing to individuals and small organizations. Although it lacks deep integration capabilities, it emphasizes precision and real-time information [35]. Recent updates include Internal Knowledge Search, allowing Pro and Enterprise Pro users to search public web content and internal knowledge bases simultaneously, and Spaces, an AI-powered collaboration hub for organizing research, connecting internal files, and customizing AI assistants for specific tasks, enhancing teamwork and productivity.

These five chatbots demonstrate the diverse capabilities of GAI technology, excelling in areas such as professional productivity, ethical AI, real-time information, and integration. Table 2 provides a detailed comparison of the utilized LLMs, functionality, applicability, integration capability, real-time access, and pricing for these leading GAI chatbots.

Table 2. Comparison of the most widely used GAI chatbots.

Feature	ChatGPT	Copilot	Gemini	Claude	Perplexity
Foundation LLM(s)	GPT-o1, GPT-4o	GPT-4o	Gemini 2.0 Flash, Gemini 1.5 Pro	Claude 3.5 Sonnet, Claude 3.5 Haiku	Sonar Small, Sonar Large
Description	Web browsing, code execution, image generation, and custom GPTs for tailored interactions	Coding assistance, task automation and integration with MS product	Multimodal data processing, integration with Google services, and advanced reasoning capabilities	Safety, ethical considerations, handling extensive context for in-depth analyses	Information retrieval, real-time web search capabilities, and user-friendly interfaces
Advantages	Versatile tasks, including content creation, coding assistance, and data analysis	Deep integration with MS's ecosystem, excelling in coding support and task automation within MS applications	Handling large context reasoning, multimodal data processing, and integration with Google services	Managing extensive context windows, suitable for processing large documents and complex conversations	Quick information retrieval and concise answers, functioning as an AI-powered search assistant
Context length	128 K	128 K	1 M, 2 M	200 K	131 K
Integration	Available as an API, browser, and mobile app	Integrated into MS products (web, Windows, and mobile) and code editors (Visual Studio and GitHub)	Integrated into Google Workspace and other Google services	Available via API and standalone applications	Accessible through web interface and browser extensions
Real-time access	Yes, can browse the internet to provide current information	Yes, accesses real-time data from the web	Yes, designed for real-time interactions and data retrieval	Limited, primarily relies on training data with some real-time capabilities in advanced versions	Yes, provides up-to-date information from the web
Price	Free tier available; Plus and Pro subscriptions at USD 20/month and USD 200/month for priority access and additional features	Integrated into MS's ecosystem; pricing varies based on specific application and subscription model, with Pro at USD $20/month	Offers free and premium versions, with advanced plan priced at USD 19/month	Free tier with limited daily messages; Pro plan at USD 20/month, offering enhanced capabilities	Free access with basic functionalities; Pro version at USD 20/month for advanced features

According to the collected data (Table 2), the five GAI chatbots demonstrate unique strengths and capabilities tailored to diverse user needs.

ChatGPT offers a context window of up to 128 K tokens, making it suitable for tasks requiring extended interactions. Its features include web browsing, code execution, image generation, and the ability to create custom GPTs for tailored applications. This makes ChatGPT particularly suited for content creation, coding assistance, and data analysis within flexible and interactive use cases.

Copilot, integrated within Microsoft's ecosystem, shares similar context window capabilities with ChatGPT due to its foundation on the same model. Its strengths lie in coding

assistance, task automation, and seamless integration with Microsoft Office applications. This tight integration makes it a powerful productivity tool for users working within Microsoft's suite of tools, offering efficiency for enterprise and professional workflows.

Gemini excels in processing multimodal data and handling extensive context, with the ability to manage up to 1 million tokens. This capability positions it as a leader for tasks involving large datasets, advanced reasoning, and integration with Google services. Its rapid processing and support for multimodal data, combined with Google Workspace integration, make it particularly strong in professional and research-oriented environments.

Claude, with a context window of approximately 200 K tokens, is ideal for tasks requiring extensive document processing and in-depth analyses. Its emphasis on safety, ethical considerations, and privacy measures positions it as a preferred choice for applications where ethical AI use and robust security are critical, particularly in education, research, and data-sensitive industries.

Perplexity, with a context window of around 131 K tokens, is designed for quick information retrieval and concise answers. Its focus on real-time web search capabilities and user-friendly interfaces makes it highly effective as an AI-powered search assistant, catering to users who prioritize fast, precise, and up-to-date information.

In summary, while all five chatbots are effective tools for various AI-driven tasks, their features and strengths vary depending on the specific application. Copilot and Gemini are well suited for tasks that require integration within specific ecosystems, such as Microsoft Office and Google Workspace. Claude is best suited for applications requiring extensive context windows, with a strong emphasis on safety, ethics, and privacy. Perplexity excels in quick information retrieval and concise, accurate responses. ChatGPT and Copilot offer greater versatility with features like image generation and internet access, making them valuable for diverse use cases. On the other hand, Gemini, Claude, and Perplexity provide larger context windows and more affordable API access, catering to users with specific technical or budgetary requirements.

Given the rapidly evolving nature of the GAI chatbot field and the continuous emergence of new players, our review represents only a snapshot of the current landscape.

3. Methodological Framework for GAI Chatbot Selection

This section outlines the theoretical foundations of interval-valued hesitant Fermatean fuzzy numbers (IVHFFNs), introduces a modified TOPSIS approach utilizing IVHFFNs, and proposes a conceptual framework for decision analysis of GAI chatbot data.

To address the challenge of GAI chatbot selection, we employed the classic Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) [36], complemented by recently developed fuzzy sets modification. As a distance-based multi-criteria decision-making method, TOPSIS determines the relative closeness of each alternative to the ideal solution (best outcome) and the anti-ideal solution (worst outcome) for each criterion. The alternative with the highest coefficient of relative proximity to the ideal solution is selected as the most suitable.

3.1. Interval-Valued Hesitant Fermatean Fuzzy Numbers: Some Basic Definitions and Operations

To enhance the TOPSIS methodology, we integrate interval-valued hesitant Fermatean fuzzy sets (IVHFFSs) [14]. This subsection provides an overview of the key concepts and arithmetic operations associated with IVHFFNs, which are essential for implementing this modification.

IVHFFSs extend earlier models such as interval-valued fuzzy sets (IVFSs) (1975) [37], hesitant fuzzy sets (HFSs) (2010) [38] and Fermatean fuzzy sets (FFSs) (2020) [39]. Represented in a three-dimensional space, IVHFFSs use interval values within the range [0, 1] to

describe the belongingness degree (BD), non-belongingness degree (NBD), and indeterminacy degree. A notable feature of IVHFFSs is the use of interval values for BD and NBD, with the constraint that the cube of the upper bounds for these intervals must not exceed 1. Compared to FFSs, IVHFFSs provide a more complex representation of uncertainty.

When crisp BD and NBD values are challenging to obtain—due to imprecise or uncertain data—IVHFFNs, with their interval-valued flexibility and the ability to accommodate multiple intervals, offer a practical solution for decision makers and researchers. This flexibility ensures more accurate assessments of alternatives in situations where precise evaluations are unattainable.

In this section, some basic concepts of IVHFFSs are described.

Definition 1 ([14]). *The IVHFFS T in a universe U is defined by the following:*

$$T = \{u_i, \langle(\alpha_T(u_i), \beta_T(u_i))\rangle \mid u_i \in U\} \tag{1}$$

where

$$\alpha_T(u_i) = \bigcup_{[\mu_T^l(u_i), \mu_T^u(u_i)] \in \alpha_T(u_i)} \left\{ \left[\mu_T^l(u_i), \mu_T^u(u_i)\right] \right\} \text{ and}$$

$$\beta_T(u_i) = \bigcup_{[\nu_T^l(u_i), \nu_T^u(u_i)] \in \beta_T(u_i)} \left\{ \left[\nu_T^l(u_i), \nu_T^u(u_i)\right] \right\}$$

represent two sets of interval values in [0, 1], signifying the possible BD and NBD of an object $u_i \in U$ to T, with the following constraints:

$$0 \leq \mu_T^l(u_i) \leq \mu_T^u(u_i) \leq 1\,, 0 \leq \nu_T^l(u_i) \leq \nu_T^u(u_i) 1 \text{ and}$$

$$0 \leq \left((\mu_T^u(u_i))^+\right)^3 + \left((\nu_T^u(u_i))^+\right)^3 \leq 1,$$

such that

$$\left[\mu_T^l(u_i), \mu_T^u(u_i)\right] \in \alpha_T(u_i),\ \left[\nu_T^l(u_i), \nu_T^u(u_i)\right] \in \beta_T(u_i),$$

$$(\mu_T^u(u_i))^+ \in \alpha_T^+(u_i) = \bigcup_{[\mu_T^l(u_i), \mu_T^u(u_i)] \in \alpha_T(u_i)} \max\{\mu_T^u(u_i)\},$$

$$(\nu_T^u(u_i))^+ \in \beta_T^+(u_i) = \bigcup_{[\nu_T^l(u_i), \nu_T^u(u_i)] \in \beta_T(u_i)} \max\{\nu_T^u(u_i)\} \text{ for all } u_i \in U.$$

The pair $((\alpha_T(u_i), \beta_T(u_i))$ is called an interval-valued hesitant Fermatean fuzzy number (IVHFFN), denoted by $\xi = (\alpha, \beta)$.

Definition 2 ([14]). *Suppose that $\xi = (\alpha, \beta)$ is an IVHFFN. Then, the score function s for ξ can be defined as follows:*

$$\begin{aligned} s(\xi) = \tfrac{1}{2}\Bigg(& \tfrac{1}{\#a}\textstyle\sum_{[\mu_T^l(u_i), \mu_T^u(u_i)] \in \alpha_T(u_i)} \left(\mu_T^l(u_i)\right)^3 - \left(\tfrac{1}{\#b}\textstyle\sum_{[\nu_T^l(u_i),\ \nu_T^u(u_i)] \in \beta_T(u_i)} \left(\nu_T^l(u_i)\right)^3\right) + \\ & \left(\tfrac{1}{\#a}\textstyle\sum_{[\mu_T^l(u_i), \mu_T^u(u_i)] \in \alpha_T(u_i)} \left(\mu_T^u(u_i)\right)^3\right) - \left(\tfrac{1}{\#b}\textstyle\sum_{[\nu_T^l(u_i),\ \nu_T^u(u_i)] \in \beta_T(u_i)} \left(\nu_T^u(u_i)\right)^3\right)\Bigg), \end{aligned} \tag{2}$$

where #a and #b represent the number of interval values in α and β, respectively.

The larger the score value $s(\xi)$, the greater the IVHFFN ξ.

Since $s(\xi) \in [-1, 1]$, an improved score function for an IVHFFN ξ in described in the following definition:

Definition 3 ([14]). *Assume $\xi = (\alpha, \beta)$ is an IVHFFN. Then, an improved score function is defined by the following:*

$$s^*(\xi) = \frac{1}{2}(s(\xi) + 1), \tag{3}$$

such that $s^*(\xi) \in [0,\ 1]$.

In case of different numbers of intervals in BD and NBD of an IVHFFN, a preprocessing step should be added. We assume to add the mean value of BD or the NBD for given object.

The arithmetic operations on IVHFFNs are given by the next definition.

Definition 4 ([14]). *Let* $\xi_1 = (\alpha_1,\ \beta_1)$ *and* $\xi_2 = (\alpha_2,\ \beta_2)$ *be two IVHFFNs. Then, we have the following:*

$$\xi_1 \oplus \xi_2 = \bigcup_{\substack{[\mu^l_{\xi_1}, \mu^u_{\xi_1}] \in \alpha_1, [\nu^l_{\xi_1}, \nu^u_{\xi_1}] \in \beta_1 \\ [\mu^l_{\xi_2}, \mu^u_{\xi_2}] \in \alpha_2, [\nu^l_{\xi_2}, \nu^u_{\xi_2}] \in \beta_2}} \left\{ \left\{ \left[\sqrt[3]{\left(\mu^l_{\xi_1}\right)^3 + \left(\mu^l_{\xi_2}\right)^3 - \left(\mu^l_{\xi_1}\right)^3 \left(\mu^l_{\xi_2}\right)^3}, \sqrt[3]{\left(\mu^u_{\xi_1}\right)^3 + \left(\mu^u_{\xi_2}\right)^3 - \left(\mu^u_{\xi_1}\right)^3 \left(\mu^u_{\xi_2}\right)^3} \right] \right\}, \left\{ \left[\nu^l_{\xi_1} \nu^l_{\xi_2}, \nu^u_{\xi_1} \nu^u_{\xi_2} \right] \right\} \right\} \tag{4}$$

$$\xi_1 \otimes \xi_2 = \bigcup_{\substack{[\mu^l_{\xi_1}, \mu^u_{\xi_1}] \in \alpha_1, [\nu^l_{\xi_1}, \nu^u_{\xi_1}] \in \beta_1 \\ [\mu^l_{\xi_2}, \mu^u_{\xi_2}] \in \alpha_2, [\nu^l_{\xi_2}, \nu^u_{\xi_2}] \in \beta_2}} \left\{ \left\{ \left[\mu^l_{\xi_1} \mu^l_{\xi_2}, \mu^u_{\xi_1} \mu^u_{\xi_2} \right] \right\}, \left\{ \left[\sqrt[3]{\left(\nu^l_{\xi_1}\right)^3 + \left(\nu^l_{\xi_2}\right)^3 - \left(\nu^l_{\xi_1}\right)^3 \left(\nu^l_{\xi_2}\right)^3}, \sqrt[3]{\left(\nu^u_{\xi_1}\right)^3 + \left(\nu^u_{\xi_2}\right)^3 - \left(\nu^u_{\xi_1}\right)^3 \left(\nu^u_{\xi_2}\right)^3} \right] \right\} \right\} \tag{5}$$

$$\lambda\xi = \bigcup_{[\mu^l_\xi, \mu^u_\xi] \in \alpha, [\nu^l_\xi, \nu^u_\xi] \in \beta} \left\{ \left\{ \left[\sqrt[3]{1 - \left(1 - \left(\mu_{\xi^l}\right)^3\right)^\lambda}, \sqrt[3]{1 - \left(1 - \left(\mu_{\xi^u}\right)^3\right)^\lambda} \right] \right\}, \left\{ \left[\left(\nu^l_\xi\right)^\lambda, \left(\nu^u_\xi\right)^\lambda \right] \right\} \right\}, \tag{6}$$

where $\lambda\ (\geq 0) \in \mathbb{R}$.

$$\xi^\lambda = \bigcup_{[\mu^l_\xi, \mu^u_\xi] \in \alpha, [\nu^l_\xi, \nu^u_\xi] \in \beta} \left\{ \left\{ \left[\left(\mu_{\xi^l}\right)^\lambda, \left(\mu_{\xi^u}\right)^\lambda \right] \right\}, \left\{ \left[\sqrt[3]{1 - \left(1 - \left(\nu^l_\xi\right)^3\right)^\lambda}, \sqrt[3]{1 - \left(1 - \left(\nu^u_\xi\right)^3\right)^\lambda} \right] \right\} \right\}, \tag{7}$$

where $\lambda\ (\geq 0) \in \mathbb{R}$.

Definition 5. *(Based on [14]) Let* $\xi_1 = (\alpha_1,\ \beta_1)$ *and* $\xi_2 = (\alpha_2,\ \beta_2)$ *be two IVHFFNs. Then, the distance between* ξ_1 *and* ξ_2 *is defined as follows:*

$$d(\xi_1,\ \xi_2) = \left(\frac{1}{4} \left(\begin{array}{c} \left| \left(\varphi^l_{1\mu}\right)^3 - \left(\varphi^l_{2\mu}\right)^3 \right|^\lambda + \left| \left(\varphi^u_{1\mu}\right)^3 - \left(\varphi^u_{2\mu}\right)^3 \right|^\lambda + \\ \left| \left(\varphi^l_{1\nu}\right)^3 - \left(\varphi^l_{2\nu}\right)^3 \right|^\lambda + \left| \left(\varphi^u_{1\nu}\right)^3 - \left(\varphi^u_{2\nu}\right)^3 \right|^\lambda + \\ \left| \left(\pi^l_1\right)^3 - \left(\pi^l_2\right)^3 \right|^\lambda + \left| \left(\pi^u_1\right)^3 - \left(\pi^u_2\right)^3 \right|^\lambda \end{array} \right) \right)^{1/\lambda}$$

where $\varphi^l_{s\mu} = \frac{1}{\#a_s} \sum_{i=1}^{\#a_s} \left(\mu^l_i\right)^3$, $\varphi^u_{s\mu} = \frac{1}{\#a_s} \sum_{i=1}^{\#a_s} \left(\mu^u_i\right)^3$, $\varphi^l_{s\nu} = \frac{1}{\#b_s} \sum_{i=1}^{\#b_s} \left(\nu^l_i\right)^3$, $\varphi^u_{s\nu} = \frac{1}{\#b_s} \sum_{i=1}^{\#b_s} \left(\nu^u_i\right)^3$, $\#a_s$ *and* $\#b_s$ *denote the number of BD and NBD intervals in* ξ_1 *and* ξ_2 *, respectively; s = 1,2,* $\lambda > 0$ *and the following:*

$$\begin{aligned} \pi^l_1 &= \sqrt[3]{1 - \left(\tfrac{1}{\#a_1} \textstyle\sum_{[\mu^l_1,\ \mu^u_1] \in \alpha_1} \left(\mu^u_1\right)^3 + \tfrac{1}{\#b_1} \textstyle\sum_{[\nu^l_1,\ \nu^u_1] \in \beta_1} \left(\nu^u_1\right)^3 \right)}, \\ \pi^u_1 &= \sqrt[3]{1 - \left(\tfrac{1}{\#a_1} \textstyle\sum_{[\mu^l_1,\ \mu^u_1] \in \alpha_1} \left(\mu^l_1\right)^3 + \tfrac{1}{\#b_1} \textstyle\sum_{[\nu^l_1,\ \nu^u_1] \in \beta_1} \left(\nu^l_1\right)^3 \right)}, \\ \pi^l_2 &= \sqrt[3]{1 - \left(\tfrac{1}{\#a_2} \textstyle\sum_{[\mu^l_2,\ \mu^u_2] \in \alpha_2} \left(\mu^u_2\right)^3 + \tfrac{1}{\#b_2} \textstyle\sum_{[\nu^l_2,\ \nu^u_2] \in \beta_2} \left(\nu^u_2\right)^3 \right)}, \\ \pi^u_2 &= \sqrt[3]{1 - \left(\tfrac{1}{\#a_2} \textstyle\sum_{[\mu^l_2,\ \mu^u_2] \in \alpha_2} \left(\mu^l_2\right)^3 + \tfrac{1}{\#b_2} \textstyle\sum_{[\nu^l_2,\ \nu^u_2] \in \beta_2} \left(\nu^l_2\right)^3 \right)}. \end{aligned} \tag{8}$$

Definition 6 ([14]). *Let* $\xi_i = \left\{\left\{\left(\mu_i^l, \mu_i^u\right)\right\}, \left\{\left(\nu_i^l, \nu_i^u\right)\right\}\right\}$ *(i = 1, 2, ..., m) be a collection of IVHFFNs and* $w = (w_1, w_2, \ldots, w_m)^T$ *such that* $w_i \geq 0$, $\sum_{i=1}^m w_i = 1$*; then, an interval-valued hesitant Fermatean fuzzy weighted average (IVHFFWA) operator is used to map* $IVHFFWA: T^n \to T$*:*

$$IVHFFWA(\xi_1, \xi_2, \ldots, \xi_m) = \oplus_{i=1}^m w_i \xi_i =$$
$$\cup_{[\mu_i^l, \mu_i^u] \in \alpha_i, [\nu_i^l, \nu_i^u] \in \beta_i} \left\{\left\{\left[\sqrt[3]{1 - \prod_{i=1}^m \left(1 - \left(\mu_i^l\right)^3\right)^{w_i}}, \sqrt[3]{1 - \prod_{i=1}^m \left(1 - \left(\mu_i^u\right)^3\right)^{w_i}}\right]\right\}, \left\{\left[\prod_{i=1}^m \left(\left(\nu_i^l\right)^3\right)^{w_i}, \prod_{i=1}^m \left(\left(\nu_i^u\right)^3\right)^{w_i}\right]\right\}\right\}. \quad (9)$$

Specifically, if $w = (1/m, 1/m, \ldots, 1/m)^T$*, then the IVHFFWA operator is converted into the following formula:*

$$IVHFFWA(\xi_1, \xi_2, \ldots, \xi_m) = \frac{1}{m} \oplus_{i=1}^m \xi_i =$$
$$\cup_{[\mu_i^l, \mu_i^u] \in \alpha_i, [\nu_i^l, \nu_i^u] \in \beta_i} \left\{\left\{\left[\sqrt[3]{1 - \prod_{i=1}^m \left(1 - \left(\mu_i^l\right)^3\right)^{1/m}}, \sqrt[3]{1 - \prod_{i=1}^m \left(1 - \left(\mu_i^u\right)^3\right)^{1/m}}\right]\right\}, \left\{\left[\prod_{i=1}^m \left(\left(\nu_i^l\right)^3\right)^{1/m}, \prod_{i=1}^m \left(\left(\nu_i^u\right)^3\right)^{1/m}\right]\right\}\right\}. \quad (10)$$

In summary, the space of interval-valued hesitant Fermatean fuzzy numbers (IVHFFNs) is broader than that of interval-valued Fermatean fuzzy numbers (IVFFNs). With a less restrictive constraint, IVHFFSs provide greater precision in addressing complex and uncertain MCDM problems compared to IVFFSs.

3.2. TOPSIS in IVHFFNs Environment

TOPSIS evaluates alternatives by measuring their closeness to an ideal solution and their distance from a negative-ideal solution. To adapt this method for IVHFFNs, we propose calculating the distances between alternatives using Equation (5). The pseudocode for the modified TOPSIS approach within the IVHFFN framework is presented in Algorithm 1.

Let $A[i]$, $i = 1, 2, \ldots, N$ represent the given set of alternatives, $C[j]$, $j = 1, 2, \ldots, M$ denote the set of identified criteria for A evaluation, and $\omega[j]$ be the set of relative weights of criteria C.

Order the alternatives in descending order based on their coefficients of relative closeness to the ideal solution RC_i and select the alternative with the highest coefficient as the optimal choice.

The proposed modification of TOPSIS integrates a new flexible IVHFFNs distance metric from Equation (8). Unlike standard Fermatean fuzzy numbers, which operate within a three-dimensional (3D) space, or IVFFNs, which utilize three 3D intervals to define the membership, non-membership, and hesitancy degrees, IVHFFNs introduce an even more complex structure. Specifically, they allow for different numbers of intervals to define the belongingness and the non-belongingness degrees. The increased flexibility in representing uncertainty results in a more accurate evaluation of alternatives.

However, the proposed new TOPSIS extension in IVHFF environment has a higher time complexity compared to its counterparts using crisp, classical, or other fuzzy models, including IVFFNs. This increased complexity arises from more intricate arithmetic operations, computationally intensive distance metric calculations, and the evaluation of multiple interval-based values in the score function.

Nevertheless, the tradeoff between more accurate representation and increased time complexity is justified, as these advanced 3D fuzzy numbers enable a more precise depiction of uncertainty in alternative evaluations.

Algorithm 1. IVHFFNs TOPSIS.

Step 1. Gather the linguistic evaluations provided by expert k in the decision matrix $X^k[i,j] \leftarrow A[i], C[j], k = 1, 2, \ldots K$,
where K is the number of experts. Convert the X matrices into values represented by IVHFFNs values.

Step 2. Compute the aggregated matrix $\tilde{X}$ for all experts according to Equation (9). Assume equal weighting for all experts $(1/K)$ and apply the averaging formula provided:
$\tilde{X}[i,j] \leftarrow IVHFFWA\left(\tilde{X}^1[i,j], \tilde{X}^2[i,j], \ldots, \tilde{X}^k[i,j]\right)$.

Step 3. Identify the minimizing criteria, referred to as the cos t criteria and denoted by $\mathbb{C}$, while the remaining criteria are categorized as benefit criteria and denoted by $\mathbb{B}$.

Step 4. Determine the normalized values of the decision matrix $\tilde{X}$ using its score function as described in Equation (3):

$$\tilde{r}[i,j] \leftarrow \frac{\tilde{x}[i,j]}{\sqrt{\tilde{x}[i,j]^2}}$$

Step 5. Derive the weighted values of assessments for each criterion:

$$\tilde{a}[i,j] \leftarrow w_j \tilde{r}[i,j]$$

according to Equation (6).

Step 6. Establish the ideal $\tilde{A}^*$and negative ideal $\tilde{A}^-$ solutions for each criterion:

$$\tilde{A}^* = \left\{\tilde{a}_1^*, \tilde{a}_2^*, \ldots, \tilde{a}_M^*\right\} = \begin{cases} \max\limits_j \tilde{a}[i,j] | j \in \mathbb{B} \\ \min\limits_j \tilde{a}[i,j] | j \in \mathbb{C} \end{cases}$$

$$\tilde{A}^- = \left\{\tilde{a}_1^-, \tilde{a}_2^-, \ldots, \tilde{a}_M^-\right\} = \begin{cases} \min\limits_j \tilde{a}[i,j] | j \in \mathbb{B} \\ \max\limits_j \tilde{a}[i,j] | j \in \mathbb{C} \end{cases}$$

for beneficial $(\mathbb{B})$ and cos t criteria $(\mathbb{C})$.

Step 7. Measure the distances from each alternative to the ideal and negative ideal solutions using Equation (8):
$D^*[i] = \sum_{j=1}^{M} D_G(\tilde{a}[i,j],\ \tilde{a}^*[j])$
$D^-[i] = \sum_{j=1}^{M} D_G(\tilde{a}[i,j],\ \tilde{a}^-[j])$

Step 8. Calculate the coefficients of relative closeness of each alternative to the ideal solution:
$RC_i = \frac{D^-}{D^- + D^+}$.

3.3. Theoretical Framework for GAI Chatbot Selection

Selecting an appropriate generative AI (GAI) chatbot involves a structured, multi-stage decision-making process to ensure alignment with organizational needs and user expectations. The new framework for unified decision analysis of GAI chatbot data consists of eight stages (Figure 1).

Stage 1: Needs Assessment

The decision-making process begins with clearly identifying the specific requirements and expectations for a GAI chatbot. This involves collecting data on available chatbots and understanding the current state of chatbot technology. Relevant information can be gathered from industry reports, user reviews, and technical specifications. The goal is to determine which chatbots are available, their capabilities, and how well they align with the organization's needs. If the assessment confirms a need for a GAI chatbot, the process advances to the next stage.

Stage 2: User Requirements Specification

In this stage, surveys or interviews are conducted to collect feedback from potential users about their expectations and preferences. This input helps define the desired features and functionalities of the chatbot, such as natural language understanding, integration capabilities, and user interface design.

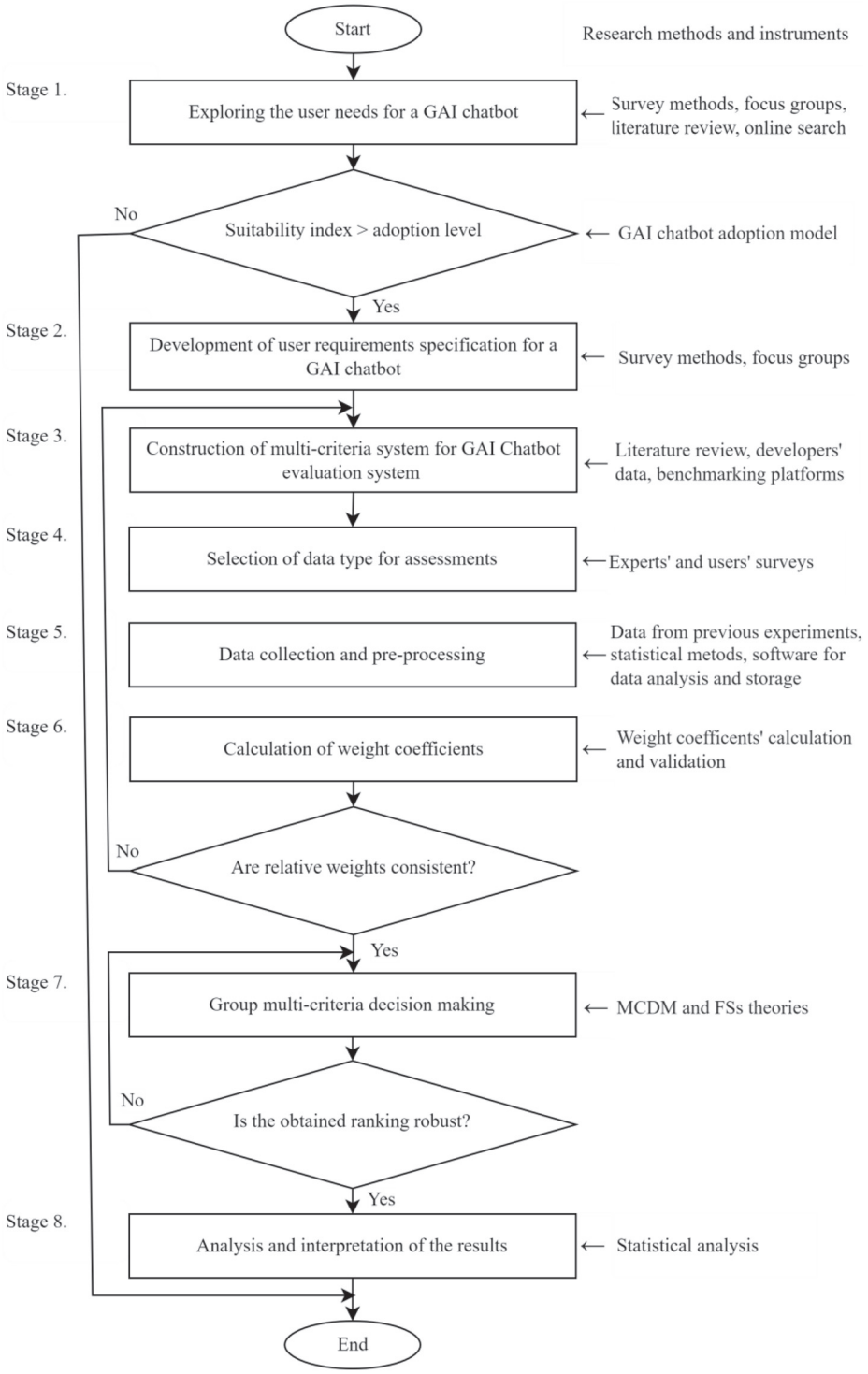


Figure 1. The flowchart of proposed framework for decision analysis of GAI chatbots.

Stage 3: Development of Evaluation Criteria

A multi-criteria evaluation system is created to facilitate a systematic comparison of chatbots. This system is based on user requirements and the organizational importance of specific chatbot features. Key criteria may include technological specifications, ease of integration, user friendliness, scalability, and cost.

Stage 4. Selection of data types

The choice of data types and decision-making methods depends on the resources available and the data collected in Stage 3. If resources are limited, decision makers

may select traditional data types and algorithms with lower computational complexity, respectively. For more precise results, advanced data types and MCDM methods can be employed, though they may require greater resources. Data collection methods may include expert evaluations, user testing, and market analysis.

Stage 5. Data reprocessing and storage

The collected data are processed and stored appropriately for further analysis. This step includes coding qualitative assessments into numerical forms, identifying and resolving duplicates or errors, addressing missing values, and ensuring overall data integrity. Once processed, the data are stored in a database or dataset for subsequent stages.

Stage 6. Determination of criteria weights

Based on the evaluation criteria and collected data, weight coefficients are assigned to each criterion to reflect their relative importance. These weights can either be predetermined or calculated using methods such as AHP or other weighting techniques.

Stage 7. Multi-criteria analysis

In this stage, the MCDM algorithm is applied to rank chatbot alternatives according to the weighted criteria. Using multiple MCDM methods or hybrid combinations can yield a more robust and comprehensive analysis.

Stage 8. Results analysis and interpretation

Decision makers analyze the rankings to identify the top chatbot alternatives. If the highest-ranked option satisfies organizational requirements, it is selected. If not, additional data may be collected and the process iterated from Stage 4. The final selection should align with long-term organizational goals and user expectations.

This structured approach ensures a comprehensive and objective selection process for GAI chatbots, customized to meet specific organizational needs.

4. A Case Study of Quality-Based Evaluation of GAI Chatbots

Let S be an organization faced with a GAI chatbot selection problem. The benefits of implementation of a GAI chatbot in the workflow of Organization S are numerous. The problem is how to find the best GAI chatbot for the organizational specifics.

The execution of Stage 1 of the proposed framework shows that there are several available GAI chatbots, and the process of chatbot selection can start. In this illustrative example, we utilize our own chatbot dataset, collected from benchmarking websites such as [23]. The dataset consists of four assessment criteria, namely C_1, C_2, $\ldots$, C_4 (Section 2.2), and five GAI chatbots, namely A_1, A_2, $\ldots$, A_5 (Section 2.3). The criteria are related to the following aspects of GAI chatbot features: C_1—conversational ability, C_2—user experience, C_3—integration capability, and C_4—price. The GAI chatbots are as follows: A_1—ChatGPT, A_2—Copilot, A_3—Gemini, A_4—Claude, and A_5—Perplexity.

In Stage 2, experts from Organization S fill in the questionnaire about their GAI chatbot requirements. Respondents evaluate the chatbot features via a five-point Likert scale ranging from "extremely important" (corresponding to 5) to "unimportant" (corresponding to 1).

In the next stage, experts from Organization S complete a questionnaire outlining their requirements for generative AI (GAI) chatbots. Participants assess the chatbot features using a five-point Likert scale, ranging from "unimportant" (1) to "extremely important" (5).

In Stage 3, a multi-attribute criteria index is developed, consisting of variables:

$$C_i,\ i = \overline{1,4}.$$

In the next stage, decision makers decide that the data type is IVHFFNs and employ the proposed new IVHFFNs TOPSIS modification. The values of the decision matrix are converted into five-point Likert scale (Table 3). For transforming every linguistic variable into its corresponding IVHFFNs, the conversion table (Table 4) is applied.

Table 3. Input decision matrix for GAI chatbots selection.

Criteria **Alternative**	C_1	C_2	C_3	C_4
A_1	VH	H	VH	H
A_2	H	H	VH	H
A_3	H	H	M	L
A_4	M	M	M	L
A_5	M	M	L	H
Criterion type	$\mathbb{B}$	$\mathbb{B}$	$\mathbb{B}$	$\mathbb{C}$

Table 4. Linguistic variables and their corresponding IVHFFNs.

Linguistic Term	**IVHFFN**
Very Low (VL)	{(0.1, 0.2) (0.3, 0.4)}, {(0.7, 0.8) (0.75, 0.85)}
Low (L)	{(0.3, 0.4) (0.5, 0.6)}, {(0.5, 0.6) (0.55, 0.65)}
Medium (M)	{(0.5, 0.6) (0.7, 0.8) (0.75, 0.9)}, {(0.3, 0.4) (0.35, 0.45)}
High (H)	{(0.7, 0.8) (0.8, 0.9)}, {(0.1, 0.2)}
Very High (VH)	{(0.9, 0.95) (0.9, 0.99)}, {(0.01, 0.1) (0.06, 0.15)}

In Stage 5, we decide that the data type is IVHFFNs and implement the proposed IVHFFNs TOPSIS modification. The decision matrix values are converted into linguistic variables as shown in Table 3. Each linguistic variable is then transformed into its corresponding IVHFFN using the conversion rules provided in Table 4.

The weight coefficients for the criteria are equal, such that $w_1 = w_2 = w_3 = w_4 = 0.25$.

The overall scores and rankings of given GAI chatbots obtained by using IVHFFNs and crisp TOPSIS method are displayed in Table 5.

Table 5. Scores and their corresponding rankings: TOPSIS method in IVHFFNs.

		A_1	A_2	A_3	A_4	A_5
IVHFFNSs	Score	0.45	0.39	0.34	0.30	0.30
TOPSIS	Rank	1	2	3	4	4

The problem was also solved using several other MCDM methods (Table 6)—weighted sum method (WSM), triangular fuzzy numbers' (TFNs) WSM, evaluation based on distance from average solution (EDAS), and TOPSIS. In order to show that the IVHFF TOPSIS solution is feasible, we compare the obtained ranking with those obtained with crisp and triangular fuzzy estimates.

The final rankings are as follows:

WSM (Benchmarking method): $A_1 \succ A_2 \approx A_3 \succ A_4 \succ A_5$.

TFNs WSM: $A_1 \succ A_3 \succ A_2 \succ A_4 \succ A_5$, $\rho = 95\%$.

EDAS: $A_1 \succ A_3 \succ A_2 \succ A_5 \succ A_4$, $\rho = 85\%$.

TOPSIS: $A_1 \succ A_2 \succ A_3 \succ A_4 \succ A_5$, $\rho = 95\%$.

IVFFNs TOPSIS: $A_1 \succ A_2 \succ A_3 \succ A_4 \approx A_4$, $\rho = 90\%$.

Table 6. Overall scores and their corresponding ranking.

	WSM		TFNs WSM		EDAS		TOPSIS	
Alternative	**Score**	**Rank**	**Score**	**Rank**	**Score**	**Rank**	**Score**	**Rank**
A_1	0.40	1	0.19	1	0.67	1	0.65	1
A_2	0.36	2	0.17	3	0.58	3	0.60	3
A_3	0.36	2	0.18	2	0.64	2	0.54	2
A_4	0.20	4	0.10	4	0.42	5	0.22	5
A_5	0.16	5	0.08	5	0.50	4	0.0	4
Spearman's ρ	Benchmark		0.95		0.85		0.95	

Spearman's rank correlation coefficient was utilized to assess the agreement between the benchmark ranking (WSM) and the rankings produced by other four MCDM methods. The analysis demonstrated high reliability of the alternative methods, with TFNs WSM and TOPSIS both achieving a Spearman's ρ of 95% and EDAS reaching a ρ of 85%. These substantial correlation coefficients of the proposed IVHFFNs TOPSIS ($\rho = 90\%$) confirm that the proposed method aligns closely with the benchmark and alternative methods, ensuring dependable and consistent ranking outcome.

Analysis of the obtained rankings categorizes the GAI chatbots into two primary groups.

Group 1 (leading GAI chatbots) includes the leading GAI chatbots: ChatGPT (A_1), Copilot (A_2), and Gemini (A_3). ChatGPT consistently secures the top position across all methods, highlighting its superior conversational ability (C_1) and robust user experience (C_2). Copilot and Gemini follow closely, demonstrating strong performance in integration capability (C_3) and competitive price (C_4). While Gemini maintains a comparable standing in most methods, Copilot showcases enhanced strengths in specific criteria, particularly in integration capability.

Group 2 (lower-ranked GAI chatbots), with Claude (A_4) and Perplexity (A_5), consistently occupy the lower ranks across all methods. Claude exhibits moderate performance but lags in conversational ability (C_1), user experience (C_2), and integration capability (C_3), whereas Perplexity AI falls behind primarily due to its less competitive integration capability (C_3).

The ranking analysis across multiple MCDM methods consistently identifies ChatGPT as the leading AI chatbot, followed by Copilot and Gemini. Claude and Perplexity are positioned in the lower tier, highlighting the need for further enhancements to improve their performance in areas such as conversational ability, user experience, and integration capability. The high correlation coefficient shows the robustness of the proposed TOPSIS modification, ensuring that the ranking reflects the underlying performance metrics.

Based on the real-life characteristics of the compared GAI chatbots, the final ranking is adequate. ChatGPT (A_1) stands out with its strong conversational abilities, high user satisfaction, and versatile integration options, which is further supported by its extensive real-world adoption and positive user feedback. Copilot (A_2) also offers robust capabilities, particularly in development-oriented tasks, while retaining reasonable usability and pricing. Gemini (A_3), although relatively new and not widely available, is expected to provide advanced conversational features in line with its strong technological backing, albeit with moderate integration and a lower price point than ChatGPT (A_1) or Copilot (A_2). Claude (A_4), known for producing safer, more controlled outputs, and Perplexity (A_5), valued for its quick question-answering style, both serve specific niches; their medium-to-lower scores in conversational ability and integration reflect these narrower focuses compared to ChatGPT (A_1) and Copilot (A_2). Consequently, the observed ordering is consistent with the advantages, target markets, and limitations of these chatbots.

It can be concluded that the proposed framework is reliable and properly reflects the requirements of organization S.

Selecting the appropriate chatbot is crucial for enhancing user engagement and operational efficiency. To streamline this selection process, a comprehensive approach is essential. This methodology enables experts to evaluate various technological, integration, and performance characteristics; establish specific requirements; utilize fuzzy assessments; and objectively identify the most suitable chatbot for a particular organization. Decision makers can further refine the evaluation system by incorporating factors such as anticipated interaction volumes, scalability, maintenance and support, error handling and recovery, and customization capabilities.

The proposed methodology offers benefits to both end users and organizational decision makers. For end users, aligning chatbot functionalities with user preferences and requirements enhances satisfaction and engagement. A chatbot selected through this process delivers precise and efficient assistance, thereby elevating the overall user experience. For organizational decision makers, the new MCDM approach provides a clear and unbiased framework for evaluating chatbots against the organization's strategic goals and operational needs. This leads to informed investment choices and the smooth integration of AI technologies into business processes.

5. Conclusions

The rapid advancement of LLMs has significantly increased the prominence of GAI chatbots in various sectors. Many organizations are integrating these conversational assistants into their workflows to enhance workflow efficiency and user engagement. However, there is currently no unified algorithmic approach for selecting suitable intelligent assistants.

In response to this challenge, we developed an integrated framework for GAI chatbot selection. This framework introduces an extension of TOPSIS within an IVHFFNs environment, enabling objective evaluation of generative chatbots. The fuzzy nature of this method effectively addresses uncertainty and vagueness in expert assessments. Moreover, the framework is versatile, accommodating both single and repeated data processing for chatbot selection.

The key advantages of the IVHFF TOPSIS include the following:

- Incorporation of several interval-valued membership and non-membership grades, along with interval-valued hesitancy degrees in the evaluation process;
- Integration of Minkowski distance-based family of metrics, enabling flexible and accurate distance calculations tailored to various data types;
- Consideration of the lengths of belongingness, non-belongingness, and hesitancy intervals in distance calculations, ensuring a comprehensive assessment of each criterion's impact.

To demonstrate the effectiveness of this new framework, we applied it to a practical scenario involving the selection of five GAI chatbots: ChatGPT, Copilot, Gemini (formerly Bard), Claude, and Perplexity. To capture the performance of the chatbots, we selected four critical criteria that align with user needs and technological capabilities. The analysis of the results indicates that the new methodology reliably reflects the features of the chatbots in the final rankings.

This evaluation process can be conducted periodically to account for the rapid advancements in GAI technologies and the evolving needs of organizations. Implementing an iterative procedure allows for continuous refinement of the selection criteria and adaptation to new developments, ensuring that the chosen chatbot solutions remain optimal over time.

In future work, we aim to enhance this conceptual framework by integrating recently developed multi-criteria decision-making methods. Additionally, we intend to develop a

new hybrid method for chatbot evaluation that combines innovative weight determination algorithms with advanced multi-criteria decision-making techniques. We also plan to expand the ranking mechanism to address uncertainties using various classic and interval fuzzy sets, including interval type-3 and T-spherical fuzzy numbers. Furthermore, we acknowledge the limitation of assuming equal-weighted coefficients in the current study and plan to refine this aspect by incorporating adaptive weighting mechanisms in our future research.

Funding: This research was partially funded the Ministry of Education and Science and by the National Science Fund, co-founded by the European Regional Development Fund, Grant No. BG05M2OP001-1.002-0002 and BG16RFPR002-1.014-0013-M001 "Digitization of the Economy in Big Data Environment".

Data Availability Statement: Data are contained within the article.

Acknowledgments: The author thanks the academic editor and anonymous reviewers for their insightful comments and suggestions.

Conflicts of Interest: The author declares no conflicts of interest.

References

1. Bulchand-Gidumal, J. Impact of artificial intelligence in travel, tourism, and hospitality. In *Handbook of e-Tourism*; Springer International Publishing: Cham, Switzerland, 2022; pp. 1943–1962.
2. Obaid, A.J.; Bhushan, B.; Rajest, S.S. (Eds.) *Advanced Applications of Generative AI and Natural Language Processing Models*; IGI Global: Hershey, PA, USA, 2023.
3. Al-Amin, M.; Ali, M.S.; Salam, A.; Khan, A.; Ali, A.; Ullah, A.; Alam, M.N.; Chowdhury, S.K. History of Generative Artificial Intelligence (AI) Chatbots: Past, Present, and Future Development. *arXiv* **2024**, arXiv:2402.05122. Available online: https://arxiv.org/abs/2402.05122 (accessed on 1 January 2025).
4. Yenduri, G.; Srivastava, G.; Maddikunta, P.K.R.; Jhaveri, R.H.; Wang, W.; Vasilakos, A.V.; Gadekallu, T.R. Generative pre-trained transformer: A comprehensive review on enabling technologies, potential applications, emerging challenges, and future directions. *arXiv* **2023**, arXiv:2305.10435. [CrossRef]
5. Saka, A.; Taiwo, R.; Saka, N.; Salami, B.A.; Ajayi, S.; Akande, K.; Kazemi, H. GPT models in construction industry: Opportunities, limitations, and a use case validation. *Dev. Built Environ.* **2023**, *17*, 100300. [CrossRef]
6. Dwivedi, Y.K.; Pandey, N.; Currie, W.; Micu, A. Leveraging ChatGPT and other generative artificial intelligence (AI)-based applications in the hospitality and tourism industry: Practices, challenges and research agenda. *Int. J. Contemp. Hosp. Manag.* **2024**, *36*, 1–12. [CrossRef]
7. Chen, B.; Wu, Z.; Zhao, R. From fiction to fact: The growing role of generative AI in business and finance. *J. Chin. Econ. Bus. Stud.* **2023**, *21*, 471–496. [CrossRef]
8. Ghaffari, S.; Yousefimehr, B.; Ghatee, M. Generative-AI in E-Commerce: Use-Cases and Implementations. In Proceedings of the 2024 20th CSI International Symposium on Artificial Intelligence and Signal Processing (AISP), Babol, Iran, 21–22 February 2024; IEEE: Piscataway, NJ, USA, 2024; pp. 1–5.
9. Al Naqbi, H.; Bahroun, Z.; Ahmed, V. Enhancing Work Productivity through Generative Artificial Intelligence: A Comprehensive Literature Review. *Sustainability* **2024**, *16*, 1166. [CrossRef]
10. Statista. Chatbot Market Worldwide 2016–2025. Available online: https://www.statista.com/statistics/656596/worldwide-chatbot-market/ (accessed on 30 June 2024).
11. Gartner. Gartner Says More Than 80% of Enterprises Will Have Used Generative AI APIs or Deployed Generative AI-Enabled Applications by 2026. Available online: https://www.gartner.com/en/newsroom/press-releases/2023-10-11-gartner-says-more-than-80-percent-of-enterprises-will-have-used-generative-ai-apis-or-deployed-generative-ai-enabled-applications-by-2026 (accessed on 30 June 2024).
12. Wang, K.; Ying, Z.; Goswami, S.S.; Yin, Y.; Zhao, Y. Investigating the role of artificial intelligence technologies in the construction industry using a Delphi-ANP-TOPSIS hybrid MCDM concept under a fuzzy environment. *Sustainability* **2023**, *15*, 11848. [CrossRef]
13. Alshahrani, R.; Yenugula, M.; Algethami, H.; Alharbi, F.; Goswami, S.S.; Naveed, Q.N.; Zahmatkesh, S. Establishing the fuzzy integrated hybrid MCDM framework to identify the key barriers to implementing artificial intelligence-enabled sustainable cloud system in an IT industry. *Expert Syst. Appl.* **2024**, *238*, 121732. [CrossRef]
14. Mishra, A.R.; Liu, P.; Rani, P. COPRAS method based on interval-valued hesitant Fermatean fuzzy sets and its application in selecting desalination technology. *Appl. Soft Comput.* **2022**, *119*, 108570. [CrossRef]

15. Chakrabortty, R.K.; Abdel-Basset, M.; Ali, A.M. A multi-criteria decision analysis model for selecting an optimum customer service chatbot under uncertainty. *Decis. Anal. J.* **2023**, *6*, 100168. [CrossRef]
16. Santa Barletta, V.; Caivano, D.; Colizzi, L.; Dimauro, G.; Piattini, M. Clinical-chatbot AHP evaluation based on "quality in use" of ISO/IEC 25010. *Int. J. Med. Inform.* **2023**, *170*, 104951. [CrossRef] [PubMed]
17. ISO/IEC. *Systems and Software Engineering—Systems and Software Quality Requirements and Evaluation (SQuaRE)—Product Quality Model*; International Organization for Standardization (ISO): Geneva, Switzerland, 2023; Available online: https://www.iso.org/standard/78176.html (accessed on 1 January 2025).
18. Singh, C.; Dash, M.K.; Sahu, R.; Singh, G. Evaluating Critical Success Factors for Acceptance of Digital Assistants for Online Shopping Using Grey–DEMATEL. *Int. J. Hum. Comput. Interact.* **2023**, *40*, 8674–8688. [CrossRef]
19. Pandey, M.; Litoriya, R.; Pandey, P. Indicators of AI in Automation: An Evaluation Using Intuitionistic Fuzzy DEMATEL Method with Special Reference to Chat GPT. *Wirel. Pers. Commun.* **2024**, *134*, 445–465. Available online: https://link.springer.com/article/10.1007/s11277-024-10917-7 (accessed on 30 June 2024). [CrossRef]
20. Pathak, A.; Bansal, V. Factors Influencing the Readiness for Artificial Intelligence Adoption in Indian Insurance Organizations. In *Transfer, Diffusion and Adoption of Next-Generation Digital Technologies*; Sharma, S.K., Dwivedi, Y.K., Metri, B., Lal, B., Elbanna, A., Eds.; IFIP Advances in Information and Communication Technology; Springer: Cham, Switzerland, 2024; Volume 698, pp. 384–397. Available online: https://link.springer.com/chapter/10.1007/978-3-031-50192-0_5 (accessed on 1 January 2025).
21. Wiangkham, A.; Vongvit, R. Comparative Analysis of MCDM Methods for Prioritizing Influential Factors of Chatgpt Adoption in Higher Education. 2024. Available online: https://ssrn.com/abstract=5040810 (accessed on 1 January 2025).
22. Ojo, Y.; Davids, V.; Oni, O.; Odoemene, M.; Idowu-Collin, P.; Eyeregba, U. A Multi-Criteria Approach for Evaluating the Use of AI for Matching Patients to Optimal Mental Health Treatment Plans. *Read. Time* **2024**, *193*, 201–222. Available online: https://worldscientificnews.com/wp-content/uploads/2024/04/WSN-1932-2024-201-222.pdf (accessed on 1 January 2025).
23. Chatbot Arena. Available online: https://lmarena.ai (accessed on 1 January 2025).
24. Artificial Analysis. Available online: https://artificialanalysis.ai/ (accessed on 1 January 2025).
25. Parasuraman, A.; Zeithaml, V.A.; Berry, L.L. SERVQUAL: A multiple-item scale for measuring consumer perceptions of service quality. *J. Retail.* **1988**, *64*, 12–40.
26. Davis, F.D. Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Q.* **1989**, *13*, 319–340. [CrossRef]
27. Verhoef, P.C.; Lemon, K.N.; Parasuraman, A.; Roggeveen, A.; Tsiros, M.; Schlesinger, L.A. Customer experience creation: Determinants, dynamics and management strategies. *J. Retail.* **2009**, *85*, 31–41. [CrossRef]
28. Venkatesh, V.; Morris, M.G.; Davis, G.B.; Davis, F.D. User Acceptance of Information Technology: Toward a Unified View. *MIS Q.* **2003**, *27*, 425–478. [CrossRef]
29. Tornatzky, L.G.; Fleischer, M. *The Processes of Technological Innovation*; Lexington Books: Lanham, MD, USA, 1990.
30. Yusof, M.M.; Kuljis, J.; Papazafeiropoulou, A.; Stergioulas, L.K. An Evaluation Framework for Health Information Systems: Human, Organization and Technology-Fit Factors (HOT-Fit). *Int. J. Med. Inform.* **2008**, *77*, 386–398. [CrossRef]
31. Pan, C.; Banerjee, J.S.; De, D.; Sarigiannidis, P.; Chakraborty, A.; Bhattacharyya, S. ChatGPT: A OpenAI platform for society 5.0. In Proceedings of the Doctoral Symposium on Human Centered Computing, Singapore, 25 February 2023; Springer Nature: Singapore, 2023; pp. 384–397.
32. Stratton, J. An Introduction to Microsoft Copilot. In *Copilot for Microsoft 365: Harness the Power of Generative AI in the Microsoft Apps You Use Every Day*; Apress: Berkeley, CA, USA, 2024; pp. 19–35.
33. Saeidnia, H.R. Welcome to the Gemini era: Google DeepMind and the information industry. *Library Hi Tech News*, 2023; ahead-of-print. [CrossRef]
34. Priyanshu, A.; Maurya, Y.; Hong, Z. AI Governance and Accountability: An Analysis of Anthropic's Claude. *arXiv* **2024**, arXiv:2407.01557.
35. Deike, M. Evaluating the performance of ChatGPT and Perplexity AI in Business Reference. *J. Bus. Financ. Librariansh.* **2024**, *29*, 125–154. [CrossRef]
36. Hwang, C.L.; Yoon, K. *Multiple Attribute Decision Making: Methods and Applications A State-of-the-Art Survey*; Springer: Berlin/Heidelberg, Germany, 1981; Volume 186.
37. Zadeh, L.A. The concept of a linguistic variable and its application to approximate reasoning—I. *Inf. Sci.* **1975**, *8*, 199–249. [CrossRef]
38. Torra, V. Hesitant fuzzy sets. *Int. J. Intell. Syst.* **2010**, *25*, 529–539. [CrossRef]
39. Senapati, T.; Yager, R.R. Fermatean fuzzy sets. *J. Ambient Intell. Humaniz. Comput.* **2020**, *11*, 663–674. [CrossRef]

electronics

MDPI

Article

Understanding Factors Influencing Generative AI Use Intention: A Bayesian Network-Based Probabilistic Structural Equation Model Approach

Cheong Kim

Office of Research, aSSIST University, Seoul 03767, Republic of Korea; ckim@assist.ac.kr

Abstract: This study investigates the factors influencing users' intention to use generative AI by employing a Bayesian network-based probabilistic structural equation model approach. Recognizing the limitations of traditional models like the technology acceptance model and the unified theory of acceptance and use of technology, this research incorporates novel constructs such as perceived anthropomorphism and animacy to capture the unique human-like qualities of generative AI. Data were collected from 803 participants with prior experience of using generative AI applications. The analysis reveals that social influence (standardized total effect = 0.550) is the most significant predictor of use intention, followed by effort expectancy (0.480) and perceived usefulness (0.454). Perceived anthropomorphism (0.149) and animacy (0.145) also influence use intention, but with a lower relative impact. By utilizing a probabilistic structural equation model, this study overcomes the linear limitations of traditional acceptance models, allowing for the exploration of nonlinear relationships and conditional dependencies. These findings provide actionable insights for improving generative AI design, user engagement, and adoption strategies.

Keywords: generative AI; TAM; UTAUT; anthropomorphism; animacy; probabilistic structural equation model

1. Introduction

The rapid advancements in artificial intelligence have catalyzed significant transformations across industries and societal domains [1]. Generative AI (GAI), a groundbreaking technology, has emerged as a distinct paradigm, contrasting with traditional AI's focus on structured data processing [2,3]. GAI models, trained on extensive datasets of natural language, images, and diverse media, demonstrate a remarkable ability to capture the nuances and patterns of human expression [3]. This enables them to generate original, contextually relevant outputs that resemble human-created content. From rational and creative text generation to the production of realistic images and sound data, generative AI shows a remarkable capacity to mimic and augment human-like capabilities. Applications like OpenAI's ChatGPT have showcased their proficiency in processing language and generating text outputs. This transformative potential has accelerated GAI's widespread adoption across sectors such as healthcare, finance, education, and software development [4–7]. GAI applications include assisting in drug discovery and diagnostics, enhancing data analysis and tailored services, supporting personalized learning experiences, and accelerating coding and debugging processes. Reflecting this promise, leading technology firms have committed substantial investments to advance GAI, with Microsoft alone investing approximately USD 13 billion in OpenAI. Industry forecasts suggest that the global AI market will

exceed USD 1.3 trillion by 2032, underscoring the economic significance of this field [8]. However, alongside its promising applications, GAI also introduces critical challenges. The capacity to generate deepfake content and realistic fake media has exacerbated concerns about misinformation, copyright disputes, and unethical uses. Additionally, questions surrounding intellectual property rights and the societal impact of AI's expanding role in traditionally human-dominated domains underscore the urgency for developing robust regulatory frameworks. This research seeks to expand the theoretical boundaries of existing models, such as the TAM and UTAUT, while offering practical insights for policymakers and industry leaders to promote the ethical and effective integration of GAI across diverse sectors, including healthcare, finance, education, and software development.

A key aspect of addressing these challenges lies in understanding how users perceive and accept GAI. Whether individuals find the technology useful, trustworthy, or easy to use directly impacts its adoption and integration. Theoretical frameworks, such as the technology acceptance model (TAM) by [9] and the unified theory of acceptance and use of technology by Venkatesh et al. [10], have been helpful in studying phenomena related to the adoption of information systems. The TAM emphasizes the importance of perceived usefulness and ease of use, while the UTAUT provides additional factors such as performance expectancy, social influence, and facilitating conditions [7]. However, these models have limitations; for instance, both rely heavily on the assumption of linear relationships between variables, which oversimplifies the complex, often nonlinear nature of human interaction with advanced technologies like GAI. Also, traditional methods like structural equation modeling (SEM) focus on fixed pathways, limiting their ability to discover emergent patterns within data or to capture contextual nuances [11]. These traditional methods could sometimes be less effective at addressing the uncertainties inherent in real-world scenarios, such as varying user motivations and external influences. Furthermore, GAI aspires to emulate human capabilities, including reasoning, natural interaction, and human-like intellectual creativity. Therefore, when studying user acceptance of GAI, it is imperative to explore the degree to which GAI exhibits humanization and animacy. These characteristics significantly influence how users perceive and engage with GAI, especially in environments where natural, human-like interactions are prioritized. However, traditional models like the TAM and UTAUT may have limitations in this context. These theoretical frameworks were developed during a period when information systems primarily facilitated unidirectional interactions between users and technology, without a focus on anthropomorphic or interactive elements. Consequently, TAM and UTAUT may fall short in comprehensively capturing the nuanced factors that influence user acceptance of GAI. To address this gap, it is essential to integrate constructs that account for GAI's anthropomorphism and animacy in the study of user acceptance, enabling a more accurate assessment of its adoption and interaction potential in human-centered applications.

In order to find a way to overcome these limitations, this study employs a probabilistic structural equation model (PSEM) based on Bayesian networks with an algorithm of equivalence (EQ) classes to explore factors affecting the adoption of generative AI from the perspective users. Unlike conventional methods, Bayesian networks allow for the modeling of conditional dependencies and nonlinear interactions among variables [12]. This approach enables a deeper exploration of complex relationships influencing user acceptance of GAI. By integrating PSEM into the study of generative AI's adoption, this research aims to expand the theoretical boundaries of the TAM and UTAUT while offering actionable insights for policymakers and industry leaders. Also, in order to reflect the aim of GAI's humanization, this research not only adopts the items in the TAM and UTAUT, but also employs factors to reflect the humanization of GAI, such as perceived anthropomorphism and perceived animacy suggested by Bartneck et al. [13] for studying robots and the theory of anthropomorphism by Epley et al. [14].

Such an approach not only enhances our understanding of user behavior but also supports the development of strategies to promote the ethical and effective integration of GAI across diverse sectors. This research will contribute to the ongoing discourse on the responsible and beneficial deployment of GAI, addressing critical challenges regarding AI's expanding role in traditionally human-dominated domains.

2. Related Works

2.1. Anthropomorphism and Animacy

The integration of anthropomorphism and animacy into the study of GAI adoption enriches the existing literature by addressing the nuanced, human-centric factors influencing technology acceptance. Prior studies could be cited to highlight the growing importance of anthropomorphism and animacy in the AI domain [13–19]. Anthropomorphism reflects the degree to which users perceive a system as having human-like qualities, such as intentions, emotions, or intelligence [13]. This perception significantly enhances user trust and engagement, particularly in technologies designed for social or interactive purposes [13,15–17]. For example, Yang, Liu, Lv, Ai, and Li [15] investigated the impact of anthropomorphic design in AI service agents on customer usage intentions and discovered that customers prefer more anthropomorphic agents when they feel a high sense of control. Also, Blut, Wang, Wünderlich, and Brock [17] examined how anthropomorphism in AI robots influences customer intentions. Their findings revealed that anthropomorphism enhances perceptions of intelligence and usefulness of AI robots, thereby increasing customer intentions to use service robots. Troshani, Rao Hill, Sherman, and Arthur [16] explored how the human-like attributes of AI applications, specifically anthropomorphism and intelligence, affect consumer trust and found that human-like features can enhance trust. In addition to anthropomorphism, animacy, which relates to the perception of a system as "alive" or intentional, is critical in fostering emotional connections and natural interactions [14,18,19]. For instance, Laban [19] investigated how perceptions of animacy in AI chatbots influence user engagement and suggested that users are more likely to engage with AI chatbots they perceive as having animacy. Also, Al-Qaysi, Al-Emran, Al-Sharafi, Yaseen, Mahmoud, and Ahmad [18] examined the influence of AI attributes including animacy on the use of generative AI and revealed that AI attributes, such as perceived anthropomorphism, animacy, and intelligence, had a positive influence on generative AI use. Therefore, by incorporating anthropomorphism and animacy, this research provides a comprehensive understanding of how human-like characteristics impact GAI adoption, offering actionable insights for developers seeking to design intuitive and engaging AI interfaces.

2.2. Technology Adotion

The TAM and UTAUT are widely used frameworks for understanding technology adoption, including GAI. The TAM focuses on two key factors: perceived usefulness, the belief that using a system enhances performance, and perceived ease of use, the belief that using the system requires minimal effort [9,20]. The UTAUT expands on the TAM by incorporating performance expectancy, effort expectancy, social influence, and facilitating conditions, alongside moderating factors like age and experience [10]. Recent studies have applied these models to explore GAI adoption. For instance, a TAM-based study on teachers' acceptance of GAI tools in classrooms found a strong positive correlation between perceived usefulness and acceptance, with perceived ease of use also playing a significant role [21]. Similarly, the UTAUT has been used to investigate GAI adoption in higher education, highlighting the influence of performance expectancy, effort expectancy, social influence, and facilitating conditions on behavioral intentions and usage [22]. Xin et al. [23] examined GAI adoption in EFL teaching in Indonesian higher education using the UTAUT

and found that performance expectancy and social influence positively influence adoption. Additionally, Yan and Yue [24] explored factors influencing GAI adoption in new product development teams using the UTAUT, and their key findings included performance expectancy as a top predictor of attitudes and task–tool fitness as the strongest predictor of behavioral intention. These findings underscore the relevance of the TAM and UTAUT frameworks in identifying key factors driving GAI adoption across various contexts. However, while these studies primarily relied on constructs from the TAM and the UTAUT to explore factors influencing GAI adoption, they often overlooked the ultimate objective of GAI: to emulate human-like behavior. This omission underlines a critical gap in the research, as the ability of GAI to act like a human could be a key determinant of its utility and acceptance across various contexts. Therefore, this research focuses not only on components of the TAM and UTAUT but also focuses on anthropomorphism and animacy to articulate the characteristics of GAI.

2.3. Generative AI Adoption

The adoption of GAI has garnered significant attention across various sectors, including education, service industry, software engineering, and healthcare [25–31]. Recent studies have identified several key factors influencing the acceptance and integration of GAI technologies. In the educational domain, research by Ivanov, Soliman, Tuomi, Alkathiri, and Al-Alawi [25] examined GAI adoption in higher education using the theory of planned behavior, and found that the perceived strengths of GAI positively influence attitudes, subjective norms, and perceived behavioral control, which in turn drive intentions and actual adoption. Within the service sector, a study by Gupta and Rathore [26] explored barriers, such as ethics, technology, cost, regulations, privacy, and digital infrastructure, to GAI adoption in service organizations through a mixed-method approach, and confirmed that these barriers are related to GAI adoption. This study identified that compatibility with existing workflows significantly influences adoption decisions, while factors such as perceived usefulness and social aspects were less impactful. Furthermore, an analysis by Russo [27] examined the complexities of integrating GAI into software engineering, and highlighted that individual, technological, and societal factors collectively affect adoption patterns, emphasizing the need for a comprehensive approach to understanding GAI integration. Jindal, Lungren, and Shah [28] explored how GAI can deliver value in health systems, emphasizing that users' willingness to adopt GAI ultimately depends on its perceived value, and concluded that maximizing this value requires cultural adaptation within health systems to align with the unique adoption patterns of generative AI. These former studies identically articulate the importance of exploring the factors affecting GAI adoption, and thus understanding factors influencing GAI adoption is crucial for developing strategies that facilitate effective implementation and address potential challenges.

As demonstrated in the previous studies mentioned above, GAI is being utilized across various fields based on its tremendous advantages. However, it is also true that concerns exist regarding potential harm caused by its misuse. To address the safety concerns surrounding the adoption of GAI, it is crucial to consider its potential for malicious exploitation by adversaries. For instance, GAI can be leveraged to launch attacks, such as identity spoofing or unauthorized access, posing significant risks. Recent studies, such as F2Key, which dynamically convert facial features into private keys using COTS headphones [32], and AFace, a range-flexible anti-spoofing face authentication method using smartphone acoustic sensing (UbiComp'24), provide valuable insights into both the threats and defenses related to GAI applications. These examples highlight the importance of developing robust safeguards to mitigate privacy and security vulnerabilities in GAI usage [33]. Therefore, when adopting generative AI, it is important to consider not only its advantages but also the potential risks mentioned above.

3. Materials and Methods

3.1. Research Data

This research used an online survey platform, Prolific [34], to gather a cohort of 815 participants who had prior experience of using GAI, such as ChatGPT, Claude, and Gemini, between 18 October and 1 November 2024. Participants were selected based on specific pre-screening criteria to ensure relevance to the study. These criteria included prior usage experience with at least one generative AI tool within the last six months and fluency in English, which was ensured by restricting recruitment to participants from the United States, the United Kingdom, and Ireland. While efforts were made to achieve demographic diversity in terms of age, gender, and geographic location, the Prolific screening policy limited extensive customization, as adding more detailed filters significantly increased costs. As a result, the final sample reflected a balance between diversity and feasibility within the constraints of the platform. Prior to the survey, all participants were thoroughly informed about the survey and provided with comprehensive explanations of the survey procedures and objectives. After obtaining informed consent, participants were instructed to fill out the questionnaire (see Table 1). To enhance data reliability, the questionnaire incorporated five attention-check questions evaluating participants' comprehension of the questionnaire. Responses that failed to accurately address the attention-check questions were flagged for removal. Additional data cleansing steps included identifying and excluding responses with completion times under 2 min (indicating insufficient attention) or exceeding the 60 min time limit, which could signal distraction or disengagement. Responses with missing data for any item and/or clear patterns of repetition were also excluded. After applying these criteria, data from 803 participants was retained for further investigation. The data showed that 41.8% of participants were female, and 58.2% were male. Most were in their 20s (35.0%) and 30s (39.1%), with smaller proportions in their 40s (17.1%) and 50s or older (8.8%). Regarding education, the majority held an undergraduate degree (52.2%), followed by postgraduate (19.1%), college (16.7%), and high school (12.1%) levels. In terms of frequently used GAI applications, ChatGPT dominated, with 70.5% usage, followed by Claude (13.8%), Gemini (9.1%), and others (6.6%). Table 2 shows the demographic information of the final data sample.

All participants received GBP 2 for their contribution. The session had an average duration of approximately 15 min. The questionnaire used a seven-point Likert scale, ranging from strongly disagree (1) to strongly agree (7), to assess the participants' responses. The reason why this research chose the Likert scale for questionnaire responses is because it provides a simple, standardized, and effective way to measure attitudes, perceptions, and opinions [36,37]. Its ordinal nature allows respondents to express varying levels of agreement or disagreement, which enhances the granularity of data collection compared to binary scales [38]. Additionally, the Likert scale is widely recognized in research, making it easier to compare and interpret results while maintaining consistency with existing studies [37].

Table 1. Questionnaire used in this research.

Construct	Item	Mean	SD	Questionnaire	Reference
Intention to use	ITU1	4.29	1.47	I intend to use this generative AI application in the future	Davis [9]
	ITU2	4.59	1.32	I will frequently use this generative AI application for my tasks	
	ITU3	4.22	1.39	Given the opportunity, I would like to use this generative AI application	
	ITU4	4.59	1.26	I am likely to rely on this generative AI application whenever possible	

Table 1. *Cont.*

Construct	Item	Mean	SD	Questionnaire	Reference
Perceived Anthropomorphism	PA1	3.96	1.37	This generative AI application behaves as if it has their own intentions	Bartneck, Kulić, Croft and Zoghbi [13], Epley, Waytz and Cacioppo [14]
	PA2	5.37	1.15	I perceive this generative AI application as having human-like characteristics	
	PA3	5.09	1.21	This generative AI application seem to express emotions similar to humans	
Perceived Animacy	PN1	4.12	1.39	This generative AI application appears to be alive	
	PN2	3.58	1.27	The movements of this generative AI application seem natural, as if it were a living being	
	PN3	3.37	1.57	This generative AI application gives the impression that it has a life of its own	
	PN4	3.37	1.58	The behavior of this generative AI application feels intentional, as if it has its own will	
Perceived Usefulness	PU1	5.05	1.05	Using this generative AI application enhances my performance in tasks	Davis [9]
	PU2	4.67	1.12	This generative AI application is useful for accomplishing my goals	
Perceived Ease of Use	PEOU1	5.4	1.13	Learning to operate this generative AI application is easy for me	
	PEOU2	3.03	1.59	I find this generative AI application easy to use	
	PEOU3	5.52	0.99	It is easy to get the generative AI application to do what I want it to do	
	PEOU4	5.37	1.14	My interaction with this generative AI application is clear and understandable	
Performance Expectancy	PE1	3.58	1.36	I expect this generative AI application to improve my productivity	Venkatesh, Morris, Davis and Davis [10], Venkatesh and Bala [35]
	PE2	3.76	1.31	This generative AI application helps me accomplish tasks more efficiently	
	PE3	3.73	1.4	Using this generative AI application enhances the quality of my work	
	PE4	3.45	1.47	I believe this generative AI application will be beneficial for my performance	
Effort Expectancy	EE1	5.22	1.12	I feel confident in my ability to use this generative AI application without much assistance	
	EE2	5.23	1.11	The steps to complete tasks using this generative AI application are straightforward	
	EE3	5.33	1.02	This generative AI application requires minimal effort to learn and use effectively	
	EE4	4.93	1.09	I believe that using this generative AI application will not require much time to become proficient	
	EE5	5.35	0.88	Navigating through this generative AI application feels intuitive and hassle-free	
Social Influence	SI1	5.15	1.03	People whose opinions I value recommend that I use this generative AI application	
	SI2	4.7	1.25	People who are important to me think I should use this generative AI application	
	SI3	5.11	0.98	The use of this generative AI application is supported by my organization or social group	
	SI4	5.03	1.16	I feel pressure from others to use this generative AI application	

Table 2. Demographic information of the data sample.

Variable		N	%
Gender	Female	336	41.8%
	Male	467	58.2%
Age group	20s	281	35.0%
	30s	314	39.1%
	40s	137	17.1%
	50s or older	71	8.8%
Final education	High school	97	12.1%
	College	134	16.7%
	Undergraduate	419	52.2%
	Postgraduate	153	19.1%
Frequently Used GAI	ChatGPT	566	70.5%
	Claude	111	13.8%
	Gemini	73	9.1%
	Others	53	6.6%

3.2. Data Analysis

The dataset was analyzed using a Bayesian network-based PSEM approach, leveraging an unsupervised EQ algorithm, which could enhance the depth and breadth of the analysis [39]. Bayesian modeling has been widely used in studies to predict the intentions of individuals [39–41]. For example, Khalid and Anwar [40] used Bayesian model averaging to analyze the relationship between environmental factors and EV adoption, and revealed a weak overall effect of environmental concerns on EV adoption, with significant variations by EV type and regional infrastructure. Kim, Lee, and Lee [39] utilized Bayesian-based PSEM to analyze users' intention to recommend an airport, and discovered that fundamental factors, such as queuing time, Wi-Fi connectivity, and airport staff, are the most crucial factors influencing users' intention. Also, Idan [41] analyzed the inference of social network users' behavioral intentions, using a Bayesian network to capture the dynamic decision-making processes of users in relation to privacy risks. The EQ algorithm offers distinct advantages over alternative Bayesian network methods such as Naïve Bayes, TAN, Hill-Climbing, K2, and Sparse Candidate, particularly in its ability to handle complex, real-world data relationships [42–45]. Unlike Naïve Bayes and TAN, which rely on simplifying assumptions like variable independence or tree-like structures, EQ enables the modeling of more nuanced relationships by grouping potential network structures into equivalence classes [44,45]. This approach allows the algorithm to explore the search space efficiently without being constrained by restrictive assumptions, making it better suited for high-dimensional data [43]. Furthermore, the EQ algorithm excels at capturing nonlinear relationships and conditional dependencies, critical for understanding multifaceted phenomena like generative AI adoption [45]. Methods like Hill-Climbing and K2 often struggle with these complexities due to their reliance on stepwise optimization or predefined variable orders, which can lead to incomplete or biased models [46–48]. In contrast, EQ dynamically adapts to the data, identifying latent patterns and hidden dependencies that other algorithms might miss, and because of this flexibility, the EQ algorithm can ensure the robust handling of noise and uncertainty, which are common issues in user behavior datasets [44,45].

Compared to traditional structural equation modeling, the PSEM approach based on Bayesian networks offers the advantage of computing posterior probabilities for all network nodes in an omnidirectional manner, without being constrained by the directionality of the connections [49]. Conventional structural equation modeling blends statistical data and qualitative causal assumptions to uncover causal relationships. However, this methodology

frequently treats these relationships as deterministic, assuming that they exhibit precise cause-and-effect connections [50]. In contrast to traditional approaches, PSEM embraces the probabilistic nature of relationships, acknowledging the inherent uncertainties in causal connections [51]. This probabilistic perspective offers distinct advantages, as PSEM can be preferred over SEM due to its efficiency in model generation. By leveraging machine learning algorithms, PSEM can automatically and swiftly generate models, thereby streamlining the time-consuming steps often associated with other methods [39]. Moreover, in contrast to traditional SEM, PSEM sets itself apart by utilizing a Bayesian network structure, rather than solely relying on a series of linear equations [51]. This probabilistic approach offers distinct advantages, as it can more effectively model the complex, often nonlinear relationships inherent in human–technology interactions [51]. Table 3 presents a brief comparison between traditional approaches and PSEM.

Table 3. PSEM vs. other approaches.

Methodology	Key Characteristics	Strengths	Limitations	Advantages of PSEM
SEM	Analyzes latent variable relationships; assumes linearity	Good for latent constructs and clear models	Limited to linear relationships; less flexible	More flexible and captures uncertainty; captures nonlinear and causal relationships; more interpretable and models probabilistic relationships
Regression Analysis	Models linear relationships between variables	Simple, interpretable, and efficient	Cannot handle nonlinear or complex dependencies	
Decision Tree	Splits data into simple, interpretable rules	Intuitive and works with small datasets	Prone to overfitting; no probabilistic modeling	

This study utilized PSEM with BayesiaLab 11 software [52] to capitalize on its efficient algorithm for automatic model generation and its robust capabilities for conducting in-depth model analysis [53].

PSEM demonstrates three key features that distinguish it from traditional structural equation modeling approaches [39]. PSEM's inherent probabilistic nature enables the modeling of complex interdependencies among variables, proving instrumental in managing uncertainties found in real-world data [51]. Additionally, PSEM's nonparametric quality allows it to accommodate and elegantly represent nonlinear relationships between categorical variables, a crucial capability when dealing with multifaceted datasets. Furthermore, PSEM's dynamic structure is molded and informed by the data themselves, rather than solely relying on theoretical assumptions, ensuring the development of a grounded and adaptable model that accurately reflects the underlying patterns and relationships [39,51,54]. The incorporation of PSEM was influential in uncovering insights that are both intricate and firmly rooted in the dataset, thereby enhancing the robustness of the findings [54].

4. Results

Before conducting comprehensive PSEM analysis, this study compared the performance of the EQ algorithm to other common machine learning techniques, including logistic regression, decision tree, support vector machine, neural network, random forest, adaboost, and bagging. Below is a brief explanation of each technique and a comparison to the EQ algorithm:

Logistic regression is a widely used statistical method for binary classification, modeling the probability of a categorical outcome based on independent variables [55,56]. While it is simple, efficient, and interpretable, it assumes a linear relationship between features and the log-odds of the outcome, which limits its ability to capture nonlinear relation-

ships [55,56]. In contrast, the EQ algorithm can effectively model complex dependencies and conditional relationships, which logistic regression cannot address [42].

Decision trees split data hierarchically based on feature values, making them intuitive and capable of handling categorical and continuous variables [57,58]. However, they are prone to overfitting, especially with noisy or high-dimensional data, reducing generalizability [57,58]. The EQ algorithm overcomes these issues by focusing on probabilistic relationships and capturing the dependencies between variables through Bayesian network structures, providing a more robust and interpretable model [42].

SVMs classify data by finding the hyperplane that best separates classes in a high-dimensional space, making them effective for datasets with clear margins, but they are computationally expensive for large datasets and lack interpretability [59,60]. The EQ algorithm, in contrast, offers an interpretable probabilistic framework that models uncertainty and causal relationships, which SVMs cannot provide [42].

Neural networks are highly flexible and excel at learning complex, nonlinear patterns in data, but they require significant computational resources, are prone to overfitting, and often lack interpretability [61,62]. The EQ algorithm, while less flexible in general-purpose applications, is computationally efficient in discovering Bayesian network structures and provides interpretable results by capturing conditional independencies in data [42].

Random forests combine multiple decision trees to improve classification accuracy and reduce overfitting, making them robust for high-dimensional data, but they do not inherently model probabilistic relationships and can lack interpretability [63,64]. The EQ algorithm provides deeper insights by capturing probabilistic and causal relationships among variables, making it more suitable for applications requiring an understanding of the data structure [42].

AdaBoost is an ensemble method that iteratively combines weak classifiers, typically decision trees, to create a strong classifier by focusing on misclassified samples [65,66]. While it effectively reduces bias, it is sensitive to noisy data and lacks the capacity to model probabilistic relationships [65,66]. The EQ algorithm, on the other hand, is designed to discover and represent the probabilistic structure in data, providing an advantage in tasks requiring uncertainty modeling [42].

Bagging reduces variance by training multiple models on bootstrapped datasets and averaging their outputs, making it effective for reducing overfitting, but like other ensemble methods, it does not provide insights into probabilistic dependencies or causal relationships between variables [67,68]. The EQ algorithm excels in these areas, making it a more interpretable and theoretically grounded choice for Bayesian network structure learning [42].

The accuracy, recall, and precision metrics in Table 3 were derived using a 10-fold cross-validation approach. For performance measurement, the mean of three values from the 'intention to use' construct was set as the target variable. The responses in this study were encoded as continuous variables using a seven-point Likert scale. This approach allows the retention of the full variability in participants' responses, ensuring that subtle differences in perceptions are captured effectively. Encoding the responses as continuous variables is consistent with the PSEM approach employed in this study, as it enables the exploration of nuanced relationships and nonlinear dependencies between constructs. This comparison was performed to verify whether the EQ algorithm could provide reasonable performance. As presented in Table 4, the EQ algorithm's performance was found to be superior to the other benchmarking algorithms, with measurement scores higher than 90%. Additionally, considering the significant advantage of the straightforward interpretations provided by the graphical representation of the Bayesian network [51], this research concluded that the Bayesian EQ algorithm can be used as the research analysis methodology.

Table 4. The results of the performance test: EQ vs. common benchmark models.

Model	Accuracy	Precision	Recall	ROC
Equivalence Classes (EQ)	0.944	0.942	0.942	0.977
Logistic Regression	0.834	0.857	0.816	0.939
Decision Tree	0.751	0.741	0.800	0.749
Random Forest	0.826	0.849	0.808	0.923
Support Vector Machine	0.838	0.847	0.840	0.934
Neural Network	0.826	0.855	0.800	0.901
AdaBoost	0.805	0.815	0.808	0.910
Bagging	0.817	0.858	0.776	0.897

Then, variable clustering was conducted. The variable clustering process in BayesiaLab uses a hierarchical agglomerative clustering algorithm, starting with each manifest variable as a separate cluster and merging them iteratively based on arc force, derived from Kullback–Leibler divergence, to group statistically dependent variables [51,69]. Criteria such as the stop threshold and maximum cluster size guide the merging process [51]. Then, clusters are validated with 10-fold cross-validation to ensure structural reliability, and latent variables are created to represent hidden common causes [51]. Finally, results are visualized in a hierarchical Bayesian network that integrates manifest and latent variables for probabilistic modeling [51]. As a result, a total of eight groups were created, as shown in Figure 1: intention to use, perceived anthropomorphism (PA), perceived animacy (PN), perceived usefulness (PU), perceived ease of use (PEOU), performance expectancy (PE), effort expectancy (EE), and social influence (SI). The network illustrates how items are grouped under their respective constructs, showing clear clusters in the circles (i.e., nodes) with the same colors, such as performance expectancy (PE1–PE4), perceived animacy (PN1–PN4), perceived animacy (PA1–PA3), perceived ease of use (PEOU1–PEOU4), effort expectancy (EE1–EE5), intention to use (ITU1–ITU4), and social influence (SI1–SI4), where variables within the same construct are strongly connected, indicating high internal consistency. Constructs such as perceived animacy and perceived anthropomorphism exhibit strong mutual connections, reflecting their combined influence on user perceptions. Similarly, perceived ease of use and perceived usefulness are closely linked, emphasizing their interdependence in shaping the ease and utility perceptions of the system. Furthermore, social influence and effort expectancy demonstrate a significant direct link to intention to use, highlighting their critical role in driving adoption behavior. This clustering and interconnectivity across constructs reveal the structured relationships between variables, with distinct internal groupings and conditional dependencies between related constructs.

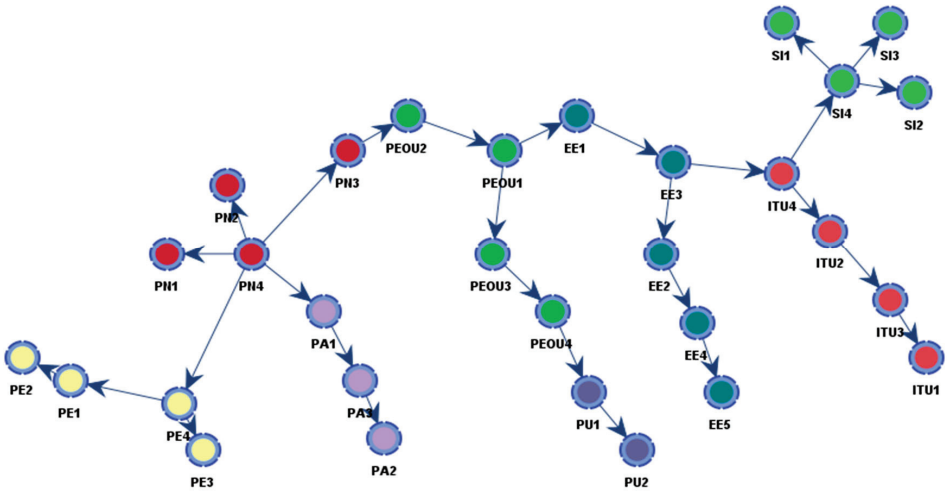


Figure 1. Variable clustering for building PSEM.

Next, multiple clustering was first conducted to create new constructs based on the variable clustering, and then unsupervised learning with the EQ algorithm was performed to formulate the final PSEM, according to the guidelines of Conrady and Jouffe [51]. Figure 2 presents the final PSEM, and Table 5 shows that all children nodes have mutual information with the target node 'intention to use'. It was identified that the three most influential nodes affecting the target node were 'social influence', 'effort expectancy', and 'perceived usefulness', with mutual information over 10%.

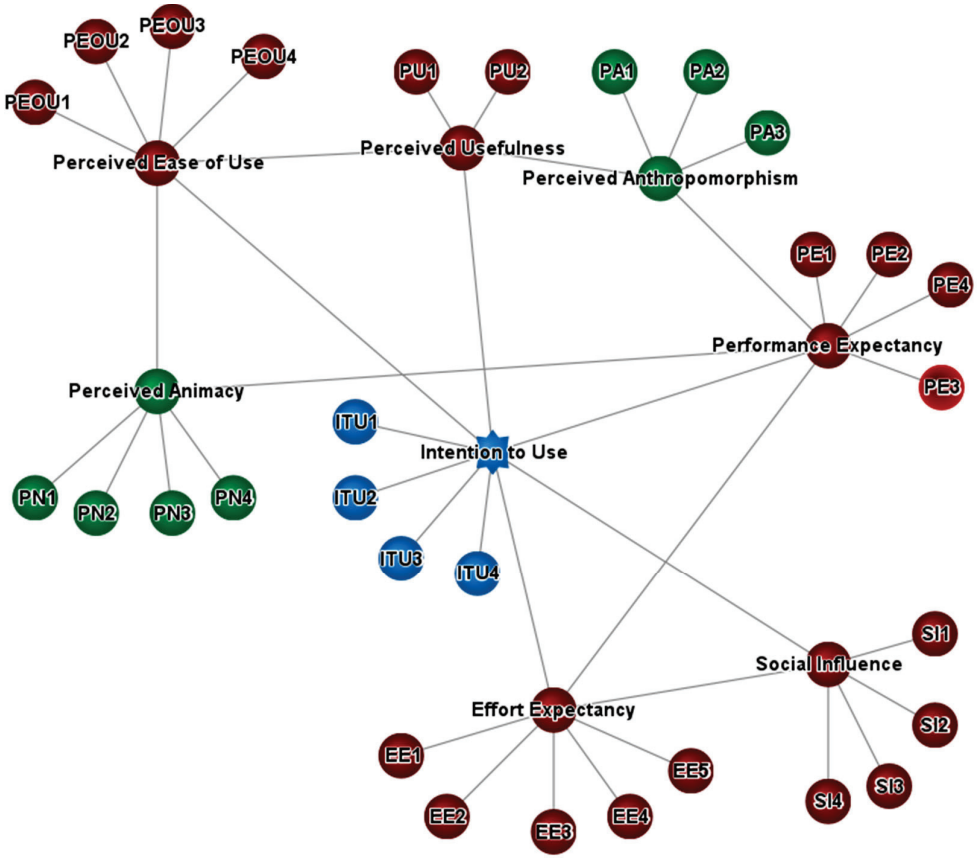


Figure 2. PSEM based on the EQ-Bayesian network (target node: intention to use).

Table 5. Descriptive statistics from the EQ network (target node: intention to use).

Node	Relative Binary Mutual Information	Posterior Mean	Max Bayes Factor	Min Bayes Factor
Social Influence	17.56%	4.56	3.93	−5.39
Effort Expectancy	12.88%	4.86	4.02	−2.93
Perceived Usefulness	12.58%	4.52	4.19	−4.19
Perceived Ease of Use	6.62%	4.70	3.11	−3.11
Performance Expectancy	8.33%	3.42	2.40	−5.60
Perceived Anthropomorphism	1.27%	4.87	1.34	−1.34
Perceived Animacy	1.24%	3.36	1.46	−1.46

The analysis revealed that all offspring nodes exhibited significant relationships with the target node 'intention to use' in terms of standardized total effect. In the context of PSEM, standardized total effects are crucial to consider, as they encompass the combined direct and indirect influences that one variable exerts on another within the model [39]. Standardized total effects provide a comprehensive measure of the strength and significance of these relationships, enabling researchers to fully understand the impact of predictors on outcomes, including the mediating effects of other variables [51]. This understanding is essential for accurate interpretation and informed decision-making based on the model's

findings [39]. As shown in Table 6, the results for the standardized total effect straightforwardly emphasize the significance of the nodes to the target node (i.e., intention to use). Social influence is the most important node affecting the target node, with a standardized total effect score of 0.550. Effort expectancy (standardized total effect = 0.480) and perceived usefulness (standardized total effect = 0.454) were also crucial factors after social influence. Performance expectancy, perceived anthropomorphism, and perceived animacy had significant effects on the target node, but their effects were relatively weaker compared with the other offspring nodes.

Table 6. The results of PSEM to the target node (i.e., intention to use).

Node	Mean	Standardized Total Effect	G-Test	df	*p*-Value
Social Influence	5.005	0.550	323.145	4.000	0.000
Effort Expectancy	5.222	0.480	285.323	4.000	0.000
Perceived Usefulness	4.873	0.454	182.812	2.000	0.000
Perceived Ease of Use	4.811	0.395	162.177	2.000	0.000
Performance Expectancy	3.619	0.190	110.525	4.000	0.000
Perceived Anthropomorphism	4.985	0.149	18.333	2.000	0.000
Perceived Animacy	3.516	0.145	16.993	2.000	0.000

5. Discussion

This study aimed to analyze the key factors that influence the adoption and usage of generative AI technologies by employing a Bayesian network-based probabilistic structural equation modeling approach. This study sought to address the limitations of the technology acceptance model and the unified theory of acceptance and use of technology by taking a more comprehensive and quantitative approach. Specifically, it systematically evaluated the relative contribution and interplay of various variables, such as perceived anthropomorphism, effort expectancy, perceived usefulness, and social influence, using standardized total effect values. This allowed the researchers to gain a deeper and more nuanced understanding of the complex dynamics that drive the adoption and acceptance of GAI technologies among users.

Social influence emerged as the most significant predictor, with a standardized total effect value of 0.550. This indicates that social contexts and opinions from peers, friends, or experts are critical factors in users' decisions to adopt GAI. Positive feedback from social networks has a direct impact on user behavior, making social influence one of the most important drivers of GAI adoption. Notably, the effects of social influence go beyond the direct utility of the technology, fulfilling users' needs for belonging and social recognition. These findings highlight the necessity of socially driven campaigns and educational programs to promote GAI adoption. Furthermore, fostering user interaction and creating collaborative environments within organizations can strengthen the positive evaluation of GAI and enhance users' intention to adopt the technology. Effort expectancy and perceived usefulness were also found to be key factors. Effort expectancy, with a standardized total effect value of 0.480, revealed that users' perceptions of how easily they can learn and use GAI significantly influence adoption. Similarly, perceived ease of use, with a standardized total effect value of 0.395, underscores the importance of intuitive user interfaces and straightforward learning experiences, particularly for non-technical users. These results suggest the need for simplified design and robust educational materials to facilitate technology adoption. Perceived usefulness, with a standardized total effect value of 0.454, demonstrated that users' belief in GAI's ability to improve personal or professional outcomes strongly drives adoption. This underscores the importance of marketing messages that emphasize the tangible benefits of GAI while clearly presenting its outcomes, thereby fostering trust and increasing user confidence. Perceived anthropomorphism and perceived animacy had

standardized total effect values of 0.149 and 0.145, respectively. While these factors positively influenced adoption, their effects were relatively weaker compared to other variables. These findings suggest that while users appreciate human-like characteristics in GAI, practical functionality and efficiency have a greater impact on adoption. The results highlight the need for design improvements that enhance emotional connections between users and GAI, such as incorporating emotion recognition and non-verbal communication features. Future research should explore ways to further strengthen perceived anthropomorphism and animacy to improve user experience and foster emotional engagement. Performance expectancy, with a standardized total effect value of 0.190, had a comparatively lower influence on GAI adoption. The relatively lower influence of performance expectancy on GAI adoption can be attributed to several factors. First, as GAI is still an emerging technology, users may lack confidence in its performance and reliability, particularly if the outputs are inconsistent or prone to errors. Second, many users may not fully comprehend how GAI can directly contribute to their personal or professional success, as concrete use cases and tangible benefits are not yet widely demonstrated. Additionally, the cognitive burden of effectively learning to use GAI may deter users, leading them to perceive the effort as outweighing the potential performance benefits. Furthermore, GAI's instrumental role often lacks psychological satisfaction or emotional connection, making performance less influential compared to social influence and effort expectancy. Finally, as a multipurpose tool, GAI may not be optimized for specific tasks, leaving users uncertain about its effectiveness in meeting their unique needs. Addressing these issues through the clear communication of use cases, enhancing reliability, and emphasizing emotional engagement alongside technical performance could help to improve the role of performance expectancy in GAI adoption.

The lower impact of performance expectancy compared to social influence, effort expectancy, and perceived usefulness can be explained by the unique characteristics of generative AI systems and their adoption context. Performance expectancy, which reflects users' beliefs about the system's ability to deliver efficient and effective outcomes, may hold less immediate importance when generative AI is perceived as an innovative or experimental technology still in its early adoption phase. In such cases, users are more likely to rely on social influence, as recommendations and endorsements from peers or influencers provide trust and confidence in engaging with unfamiliar technologies. Effort expectancy also plays a significant role, as the ease of learning and interacting with generative AI tools often becomes a primary concern, especially for users who are new to such systems. Perceived usefulness, by contrast, directly addresses the immediate value and practical benefits that users associate with the technology, making it a stronger driver of behavioral intention. Performance expectancy, while conceptually related to usefulness, is often seen as a long-term evaluation of the system's outcomes and therefore may carry less weight in the initial decision-making process. This distinction helps to clarify why performance expectancy had a lower relative impact compared to other constructs in the context of generative AI adoption.

This study makes a significant contribution to the field of technology acceptance research by addressing the limitations of the widely used technology acceptance model and the unified theory of acceptance and use of technology. It introduces a novel methodological approach that overcomes the shortcomings of these existing theories. The TAM primarily focuses on perceived usefulness and ease of use, but it does not adequately explain the nonlinear relationships between variables, particularly in the context of complex technological systems. Similarly, while the UTAUT expands upon the TAM by incorporating social influence and facilitating conditions, it lacks the ability to quantitatively evaluate the interaction effects between variables or account for emerging factors such as anthropomorphism

and animacy. To address these limitations, this study employs a Bayesian network-based probabilistic structural equation modeling approach. This methodological advancement allows the researchers to analyze nonlinear interactions and assess the moderating effects of social influence on the relationship between effort expectancy and perceived usefulness. Furthermore, this study integrates new variables, such as perceived anthropomorphism and animacy, to provide valuable insights into the emotional dynamics of user-technology interactions. This novel methodological approach addresses the quantitative limitations of existing theories and introduces a fresh perspective on technology acceptance research, paving the way for a more comprehensive understanding of the complex factors that drive the adoption and use of emerging technologies, such as generative AI.

The findings of this study have several practical implications. To promote GAI adoption, it is essential to strengthen social foundations through strategies such as peer learning and community-based programs that enable users to experience the benefits of GAI. During the early stages of implementation, emphasizing collaboration and feedback among users can streamline the adoption process. Additionally, designing user-friendly interfaces and providing comprehensive support materials can help to include non-expert users. Strengthening human-like features, such as voice recognition, emotional expression, and non-verbal interactions, can enhance emotional connections and build trust among users. Such design improvements not only increase adoption rates but also ensure long-term user satisfaction. Furthermore, organizations can incentivize GAI adoption by integrating it into performance metrics, reward systems, and professional development programs. This can foster a culture of innovation and encourage employees to explore and embrace the benefits of technology. Additionally, developing comprehensive training programs that address both technical and user-centric aspects of GAI can empower individuals to leverage technology effectively. These training programs should cover topics such as responsible use, ethical considerations, and practical applications, ensuring that users are equipped with the necessary knowledge and skills to maximize the benefits of GAI while navigating its complexities.

One limitation of this study is its reliance on a single dataset obtained through the Prolific online survey platform, which introduces the potential for sampling bias. While efforts were made to ensure participant diversity, the sample may not fully represent the broader population of generative AI users, particularly across different cultural or professional contexts. Additionally, this study primarily focuses on a single demographic, limiting the generalizability of the findings to more diverse user groups. To address these limitations, future research should incorporate datasets from multiple sources to reduce sampling bias and ensure greater representativeness. Conducting cross-cultural studies will be particularly valuable for exploring how cultural and regional differences influence generative AI adoption. Expanding the demographic and professional diversity of participants will also provide a more comprehensive understanding of the factors driving adoption and use across varied contexts. By addressing these gaps, future studies can enhance the applicability and robustness of insights into generative AI adoption. This study's reliance on cross-sectional data restricts its ability to analyze changes in technology adoption over time, limiting the findings to factors influencing adoption at a specific point. To address this limitation, future research should employ longitudinal approaches to investigate the evolving dynamics of GAI adoption over an extended period. This would provide deeper insights into how user perceptions, behaviors, and adoption patterns shift as the technology matures and becomes more widely used. Expanding the research scope to include longitudinal data would enable valuable insights into the long-term evolution of GAI adoption and its implications for technology development, marketing, and policy decisions. Such longitudinal investigations could shed light on how adoption patterns

and the relative importance of various factors may change over time and across different contexts, providing a more nuanced and holistic understanding of the complex dynamics involved in the widespread acceptance and use of generative AI technologies.

6. Conclusions

This study offers a comprehensive examination of the multifaceted factors influencing the adoption of generative AI. The findings reveal that social influence, effort expectancy, and perceived usefulness are pivotal drivers of GAI adoption, underscoring the importance of user-centric design and socially embedded promotion strategies. Additionally, constructs such as perceived anthropomorphism and animacy, while less influential, suggest avenues for enhancing user engagement through emotionally resonant and human-like interactions. This research advances the methodological landscape by employing Bayesian network-based probabilistic structural equation modeling, which enables the capture of complex, nonlinear relationships among variables. This innovative approach provides a richer and more nuanced understanding of the interplay between user perceptions and adoption intentions, addressing limitations in the traditional technology acceptance model and unified theory of acceptance and use of technology frameworks. In conclusion, this study offers actionable insights for practitioners and policymakers seeking to foster the ethical and effective integration of GAI. By addressing identified gaps and leveraging the proposed strategies, organizations can enhance user acceptance, optimize technological design, and unlock the transformative potential of GAI in diverse domains, including healthcare, education, and creative industries.

Funding: This research received no external funding.

Data Availability Statement: The dataset used in this research can be shared upon request.

Conflicts of Interest: The author declares no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

EE	Effort expectancy
EQ	Equivalence classes
GAI	Generative artificial intelligence
ITU	Intention to use
PA	Perceived anthropomorphism
PE	Performance expectancy
PEOU	Perceived ease of use
PN	Perceived animacy
PSEM	Probabilistic structural equation model
PU	Perceived usefulness
ROC	Receiver operating characteristic
SEM	Structural equation modeling
SI	Social influence
TAM	Technology acceptance model
UTAUT	Unified theory of acceptance and use of technology

References

1. Pezzulo, G.; Parr, T.; Cisek, P.; Clark, A.; Friston, K. Generating meaning: Active inference and the scope and limits of passive AI. *Trends Cogn. Sci.* **2024**, *28*, 97–112. [CrossRef] [PubMed]
2. Sengar, S.S.; Hasan, A.B.; Kumar, S.; Carroll, F. Generative artificial intelligence: A systematic review and applications. *Multimed. Tools Appl.* **2024**, 1–40. [CrossRef]

3. Bzdok, D.; Thieme, A.; Levkovskyy, O.; Wren, P.; Ray, T.; Reddy, S. Data science opportunities of large language models for neuroscience and biomedicine. *Neuron* **2024**, *112*, 698–717. [CrossRef]
4. Dave, T.; Athaluri, S.A.; Singh, S. ChatGPT in medicine: An overview of its applications, advantages, limitations, future prospects, and ethical considerations. *Front. Artif. Intell.* **2023**, *6*, 1169595. [CrossRef] [PubMed]
5. Wójcik, S.; Rulkiewicz, A.; Pruszczyk, P.; Lisik, W.; Poboży, M.; Domienik-Karłowicz, J. Beyond ChatGPT: What does GPT-4 add to healthcare? The dawn of a new era. *Cardiol. J.* **2023**, *30*, 1018–1025. [CrossRef]
6. Rivas, P.; Zhao, L. Marketing with chatgpt: Navigating the ethical terrain of gpt-based chatbot technology. *AI* **2023**, *4*, 375–384. [CrossRef]
7. An, X.; Chai, C.S.; Li, Y.; Zhou, Y.; Shen, X.; Zheng, C.; Chen, M. Modeling English teachers' behavioral intention to use artificial intelligence in middle schools. *Educ. Inf. Technol.* **2023**, *28*, 5187–5208. [CrossRef]
8. Bloomberg. Generative AI to Become a $1.3 Trillion Market by 2032, Research Finds. 2023. Available online: https://www.bloomberg.com/company/press/generative-ai-to-become-a-1-3-trillion-market-by-2032-research-finds/ (accessed on 13 November 2024).
9. Davis, F.D. Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. *MIS Q.* **1989**, *13*, 319–340. [CrossRef]
10. Venkatesh, V.; Morris, M.G.; Davis, G.B.; Davis, F.D. User acceptance of information technology: Toward a unified view. *MIS Q.* **2003**, *27*, 425–478. [CrossRef]
11. Volkova, S.; Arendt, D.; Saldanha, E.; Glenski, M.; Ayton, E.; Cottam, J.; Aksoy, S.; Jefferson, B.; Shrivaram, K. Explaining and predicting human behavior and social dynamics in simulated virtual worlds: Reproducibility, generalizability, and robustness of causal discovery methods. *Comput. Math. Organ. Theory* **2023**, *29*, 220–241. [CrossRef]
12. Shojaei Estabragh, Z.; Riahi Kashani, M.M.; Jeddi Moghaddam, F.; Sari, S.; Taherifar, Z.; Moradi Moosavy, S.; Sadeghi Oskooyee, K. Bayesian network modeling for diagnosis of social anxiety using some cognitive-behavioral factors. *Netw. Model. Anal. Health Inform. Bioinform.* **2013**, *2*, 257–265. [CrossRef]
13. Bartneck, C.; Kulić, D.; Croft, E.; Zoghbi, S. Measurement instruments for the anthropomorphism, animacy, likeability, perceived intelligence, and perceived safety of robots. *Int. J. Soc. Robot.* **2009**, *1*, 71–81. [CrossRef]
14. Epley, N.; Waytz, A.; Cacioppo, J.T. On seeing human: A three-factor theory of anthropomorphism. *Psychological. Rev.* **2007**, *114*, 864. [CrossRef] [PubMed]
15. Yang, Y.; Liu, Y.; Lv, X.; Ai, J.; Li, Y. Anthropomorphism and customers' willingness to use artificial intelligence service agents. *J. Hosp. Mark. Manag.* **2022**, *31*, 1–23. [CrossRef]
16. Troshani, I.; Rao Hill, S.; Sherman, C.; Arthur, D. Do we trust in AI? Role of anthropomorphism and intelligence. *J. Comput. Inf. Syst.* **2021**, *61*, 481–491. [CrossRef]
17. Blut, M.; Wang, C.; Wünderlich, N.V.; Brock, C. Understanding anthropomorphism in service provision: A meta-analysis of physical robots, chatbots, and other AI. *J. Acad. Mark. Sci.* **2021**, *49*, 632–658. [CrossRef]
18. Al-Qaysi, N.; Al-Emran, M.; Al-Sharafi, M.A.; Yaseen, Z.M.; Mahmoud, M.A.; Ahmad, A. Generative AI and educational sustainability: Examining the role of knowledge management factors and AI attributes using a deep learning-based hybrid SEM-ANN approach. *Comput. Stand. Interfaces* **2025**, *93*, 103964. [CrossRef]
19. Laban, G. Perceptions of anthropomorphism in a chatbot dialogue: The role of animacy and intelligence. In Proceedings of the 9th International Conference on Human-Agent Interaction, Virtually, 9–11 November 2021; pp. 305–310.
20. Davis, F.D. *A Technology Acceptance Model for Empirically Testing New End-User Information Systems: Theory and Results*; Massachusetts Institute of Technology: Cambridge, MA, USA, 1985.
21. Ghimire, A.; Edwards, J. Generative AI adoption in the classroom: A contextual exploration using the technology acceptance model (tam) and the innovation diffusion theory (IDT). In Proceedings of the 2024 Intermountain Engineering, Technology and Computing (IETC), Logan, UT, USA, 13–14 May 2024; pp. 129–134.
22. Zaim, M.; Arsyad, S.; Waluyo, B.; Ardi, H.; Hafizh, M.A.; Zakiyah, M.; Syafitri, W.; Nusi, A.; Hardiah, M. AI-powered EFL pedagogy: Integrating generative AI into university teaching preparation through UTAUT and activity theory. *Comput. Educ. Artif. Intell.* **2024**, *7*, 100335. [CrossRef]
23. Xin, T.; Yuan, Z.; Qu, S. Factors Influencing University Students' Behavioural Intention to Use Generative Artificial Intelligence for Educational Purposes Based on a Revised UTAUT2 Model. *J. Comput. Assist. Learn.* **2025**, *41*, e13105.
24. Yan, X.; Yue, C. Driving factors of generative ai adoption in new product development teams from a UTAUT perspective. *Int. J. Hum.–Comput. Interact.* **2024**, 1–22.
25. Ivanov, S.; Soliman, M.; Tuomi, A.; Alkathiri, N.A.; Al-Alawi, A.N. Drivers of generative AI adoption in higher education through the lens of the Theory of Planned Behaviour. *Technol. Soc.* **2024**, *77*, 102521. [CrossRef]
26. Gupta, R.; Rathore, B. Exploring the generative AI adoption in service industry: A mixed-method analysis. *J. Retail. Consum. Serv.* **2024**, *81*, 103997. [CrossRef]

27. Russo, D. Navigating the complexity of generative ai adoption in software engineering. *ACM Trans. Softw. Eng. Methodol.* **2024**, *35*, 1–50. [CrossRef]
28. Jindal, J.A.; Lungren, M.P.; Shah, N.H. Ensuring useful adoption of generative artificial intelligence in healthcare. *J. Am. Med. Inform. Assoc.* **2024**, *31*, 1441–1444. [CrossRef] [PubMed]
29. Shailendra, S.; Kadel, R.; Sharma, A. Framework for adoption of generative artificial intelligence (GenAI) in education. *arXiv* **2024**, arXiv:2408.01443. [CrossRef]
30. Gupta, A.S.; Mukherjee, J. Framework for adoption of generative AI for information search of retail products and services. *Int. J. Retail. Distrib. Manag.* **2024**, *53*, 165–181. [CrossRef]
31. Singh, J.P. Quantifying Healthcare Consumers' Perspectives: An Empirical Study of the Drivers and Barriers to Adopting Generative AI in Personalized Healthcare. *Res. Rev. Sci. Technol.* **2022**, *2*, 171–193.
32. Duan, D.; Sun, Z.; Ni, T.; Li, S.; Jia, X.; Xu, W.; Li, T. F2Key: Dynamically Converting Your Face into a Private Key Based on COTS Headphones for Reliable Voice Interaction. In Proceedings of the 22nd Annual International Conference on Mobile Systems, Applications and Services, Tokyo, Japan, 3–7 June 2024; pp. 127–140.
33. Xu, Z.; Liu, T.; Jiang, R.; Hu, P.; Guo, Z.; Liu, C. AFace: Range-flexible Anti-spoofing Face Authentication via Smartphone Acoustic Sensing. In Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, Melbourne, Australia, 5–9 October 2024; Volume 8, pp. 1–33.
34. Prolific. Available online: https://www.prolific.co (accessed on 23 October 2024).
35. Venkatesh, V.; Bala, H. Technology acceptance model 3 and a research agenda on interventions. *Decis. Sci.* **2008**, *39*, 273–315. [CrossRef]
36. Gil, M.Á.; González-Rodríguez, G. Fuzzy vs. Likert scale in statistics. In *Combining experimentation and theory: A hommage to Abe Mamdani*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 407–420.
37. Croasmun, J.T.; Ostrom, L. Using likert-type scales in the social sciences. *J. Adult Educ.* **2011**, *40*, 19–22.
38. Chyung, S.Y.; Swanson, I.; Roberts, K.; Hankinson, A. Evidence-based survey design: The use of continuous rating scales in surveys. *Perform. Improv.* **2018**, *57*, 38–48. [CrossRef]
39. Kim, C.; Lee, J.; Lee, K.C. Online review data analytics to explore factors affecting consumers' airport recommendations. *Inf. Technol. People*, 2024; *ahead-of-print*.
40. Khalid, A.; Anwar, A. Pro-environment consumer behaviour and electric vehicle adoptions: A comparative regional meta-analysis. *Appl. Econ.* **2025**, *57*, 191–215. [CrossRef]
41. Idan, L. Beyond Purchase Intentions: Mining Behavioral Intentions of Social-Network Users. *Int. J. Hum.–Comput. Interact.* **2024**, *40*, 1111–1132. [CrossRef]
42. Chickering, D.M. Learning equivalence classes of Bayesian-network structures. *J. Mach. Learn. Res.* **2002**, *2*, 445–498.
43. Li, B.H.; Liu, S.Y.; Li, Z.G. Improved algorithm based on mutual information for learning Bayesian network structures in the space of equivalence classes. *Multimed. Tools Appl.* **2012**, *60*, 129–137. [CrossRef]
44. Castelletti, F.; Peluso, S. Equivalence class selection of categorical graphical models. *Comput. Stat. Data Anal.* **2021**, *164*, 107304. [CrossRef]
45. Castelletti, F.; Consonni, G.; Della Vedova, M.L.; Peluso, S. Learning Markov equivalence classes of directed acyclic graphs: An objective Bayes approach. *Bayesian Anal.* **2018**, *13*, 1235–1260. [CrossRef]
46. Gámez, J.A.; Mateo, J.L.; Puerta, J.M. Learning Bayesian networks by hill climbing: Efficient methods based on progressive restriction of the neighborhood. *Data Min. Knowl. Discov.* **2011**, *22*, 106–148. [CrossRef]
47. Tsamardinos, I.; Brown, L.E.; Aliferis, C.F. The max-min hill-climbing Bayesian network structure learning algorithm. *Mach. Learn.* **2006**, *65*, 31–78. [CrossRef]
48. Chickering, D.M. Learning Bayesian networks is NP-complete. In *Learning from Data: Artificial Intelligence and Statistics V*; Springer: Berlin/Heidelberg, Germany, 1996; pp. 121–130.
49. Gerassis, S.; Albuquerque, M.T.D.; García, J.F.; Boente, C.; Giráldez, E.; Taboada, J.; Martín, J. Understanding complex blasting operations: A structural equation model combining Bsayesian networks and latent class clustering. *Reliab. Eng. Syst. Saf.* **2019**, *188*, 195–204. [CrossRef]
50. Bollen, K.A.; Pearl, J. Eight myths about causality and structural equation models. In *Handbook of Causal Analysis for Social Research*; Springer: Berlin/Heidelberg, Germany, 2013; pp. 301–328.
51. Conrady, S.; Jouffe, L. *Introduction to Bayesian Networks & Bayesialab*; Bayesia SAS: Nashville, TN, USA, 2024.
52. Bayesialab 11. Available online: https://www.bayesia.com (accessed on 21 November 2024).
53. Silvera, G.; Smail, L. Relationships between teamwork and suicidal behavior in Juvenile detention facilities using Bayesian networks. *Cogent Soc. Sci.* **2019**, *5*, 1652984. [CrossRef]
54. Zia, A.; Lacasse, K.; Fefferman, N.H.; Gross, L.J.; Beckage, B. Machine Learning a Probabilistic Structural Equation Model to Explain the Impact of Climate Risk Perceptions on Policy Support. *Sustainability* **2024**, *16*, 10292. [CrossRef]
55. Hosmer, D.W., Jr.; Lemeshow, S.; Sturdivant, R.X. *Applied Logistic Regression*; John Wiley & Sons: Hoboken, NJ, USA, 2013.

56. Menard, S. *Applied Logistic Regression Analysis*; SAGE Publications: Thousand Oaks, CA, USA, 2001.
57. Breiman, L. *Classification and Regression Trees*; Routledge: London, UK, 2017.
58. Quinlan, R.J. Induction of decision trees. *Mach. Learn.* **1986**, *1*, 81–106. [CrossRef]
59. Cortes, C. Support-Vector Networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
60. Schölkopf, B. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*; MIT Press: Cambridge, CA, USA, 2002.
61. Goodfellow, I. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
62. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef]
63. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
64. Cutler, D.R.; Edwards, T.C., Jr.; Beard, K.H.; Cutler, A.; Hess, K.T.; Gibson, J.; Lawler, J.J. Random forests for classification in ecology. *Ecology* **2007**, *88*, 2783–2792. [CrossRef]
65. Freund, Y.; Schapire, R.E. A decision-theoretic generalization of on-line learning and an application to boosting. *JCoSS* **1997**, *55*, 119–139. [CrossRef]
66. Hastie, T.; Rosset, S.; Zhu, J.; Zou, H. Multi-class adaboost. *Stat. Its Interface* **2009**, *2*, 349–360. [CrossRef]
67. Breiman, L. Bagging predictors. *Mach. Learn.* **1996**, *24*, 123–140. [CrossRef]
68. Tumer, K.; Ghosh, J. Error correlation and error reduction in ensemble classifiers. *Connect. Sci.* **1996**, *8*, 385–404. [CrossRef]
69. Kullback, S.; Leibler, R.A. On information and sufficiency. *Ann. Math. Stat.* **1951**, *22*, 79–86. [CrossRef]

electronics

MDPI

Article

Pre-Service Teachers' Assessment of ChatGPT's Utility in Higher Education: SWOT and Content Analysis

Angelos Markos [1,*], Jim Prentzas [2] and Maretta Sidiropoulou [2]

[1] Department of Primary Education, Democritus University of Thrace, Nea Chili, 68131 Alexandroupolis, Greece
[2] Department of Education Sciences in Early Childhood, Democritus University of Thrace, Nea Chili, 68131 Alexandroupolis, Greece; dprentza@psed.duth.gr (J.P.); masidiro@psed.duth.gr (M.S.)
* Correspondence: amarkos@eled.duth.gr

Abstract: ChatGPT (GPT-3.5), an intelligent Web-based tool capable of conducting text-based conversations akin to human interaction across various subjects, has recently gained significant popularity. This surge in interest has led researchers to examine its impact on numerous fields, including education. The aim of this paper is to investigate the perceptions of undergraduate students regarding ChatGPT's utility in academic environments, focusing on its strengths, weaknesses, opportunities, and threats. It responds to emerging challenges in educational technology, such as the integration of artificial intelligence in teaching and learning processes. The study involved 257 students from two university departments in Greece—namely primary and early childhood education pre-service teachers. Data were collected using a structured questionnaire. Various methods were employed for data analysis, including descriptive statistics, inferential analysis, K-means clustering, and decision trees. Additional insights were obtained from a subset of students who undertook a project in an elective course, detailing the types of inquiries made to ChatGPT and their reasons for recommending (or not recommending) it to their peers. The findings offer valuable insights for tutors, researchers, educational policymakers, and ChatGPT developers. To the best of the authors' knowledge, these issues have not been dealt with by other researchers.

Keywords: large language models; generative AI; AI literacy; primary education; early childhood education; knowledge extraction; e-learning; clustering; decision trees; educational robotics

1. Introduction

An interesting development in recent decades is the use of technology in higher education. Learning and teaching have been affected by the incorporation of technology, and they have changed compared to previous time periods. A notable technological advancement that has played an important role in higher education concerns the widespread use of Web-based tools [1]. A further technological advancement in higher education is the use of artificial intelligence (AI) tools. This advancement is in progress.

Web-based tools support a variety of needs in the specific educational sector. Blended learning (i.e., a combination of face-to-face and Internet-based learning), distance, and lifelong learning models have been supported in higher education institutions providing opportunities to diverse learners and tutors. Web-based tools may be used for learning and teaching activities anywhere, anyplace, and with any device connected to the Internet. It is also notable that the functionality of a wide range of Web-based tools is accessible at no cost, facilitating their use.

AI tools have been integrated into education for decades. Various survey papers in the field of AI in education (AIED) have been published attempting to highlight the main trends. One may note different viewpoints in the main trends discussed in relevant surveys. For instance, in [2], the main trends highlighted are based on the implemented functionality

(i.e., teaching students, providing support to students, and providing support to teachers). Intelligent Tutoring Systems and Dialogue-based Tutoring Systems are representatives of the first trend. Representatives of the second trend are Exploratory Learning Environments, learning companions, and collaborative learning approaches. Teaching assistants, automated evaluation and monitoring of students, and the use of AI as a research tool are examples of approaches supporting teachers. In [3], work in AIED spanning five decades (1970–2020) is surveyed, highlighting the main research themes. These concern adaptive learning and personalization, deep learning, and machine learning in online educational processes and data mining, human-AI interaction, and the educational use of AI-generated data and AI in higher education. A general trend that one may note in AIED is the incorporation of AI in Web-based tools, which offers advantages because the strong points of AI and the Web are combined.

Higher education institutions are at the forefront of AIED. This is due to the existence of faculty members, researchers, and students doing research in AIED. Trends in AIED in higher education can be discerned based on general AIED surveys such as [2,3]. However, due to the large amount of research work concerning higher education, surveys specifically involving AIED in higher education have been published (e.g., [4–6]). One may note that research within institutions has yielded various AI-based tools implemented and supported by resources of the institutions (e.g., [7,8]). Very recently, widespread Web-based tools that incorporate AI methods have been used by higher education students and staff members. The consequences of these recent advances are significant due to the large number of academic community members that use them around the world.

The continuous advancements in technology create challenges. New tools are continuously made available, but the functionality of existing tools also evolves. The evolution in the availability of tools and their functionality means that effort is needed in order to analyze the corresponding consequences for learners and tutors. One may determine the strengths and weaknesses of the tools, as well as opportunities and threats concerning learners, tutors, and academic institutions. A useful task is also to assess the tools in terms of the requirements that need to be satisfied in learning settings. Feedback from learners may play an important role in determining the aforementioned strengths, weaknesses, opportunities, and threats. Therefore, it would be practical to learn the perceptions of higher education students about technological tools used in learning settings. It could be specifically useful to learn the perceptions of higher education students who are pre-service teachers because they will become the ones who will serve in education in the future. One may note that the perceptions and familiarity of teachers with technological tools are factors affecting the use of these tools in their teaching [9].

An intelligent Web-based tool that has recently become very popular around the world is ChatGPT. The main reason for its popularity is its ability to provide human-like text-based responses to human queries in real time and in any subject. In most cases, the responses are quite accurate and time-efficient. Therefore, this tool can effectively engage in real-time text-based conversations with humans. In 1950, Alan Turing published his seminal paper in which he introduced the imitation game (i.e., the Turing test) as a way of assessing the ability of a machine to think [10]. A machine that could pass the test would be regarded as able to think. Until recently, this was not possible, but a tool such as ChatGPT constitutes a development to derive tools that could pass the Turing test.

ChatGPT affects higher education in various ways. Taking into consideration the three main trends discerned in [2], ChatGPT certainly involves two of them (i.e., support of students and teachers). To a certain degree, ChatGPT may be used to teach students. Taking into consideration the main research themes discerned in [3], ChatGPT certainly concerns two themes, i.e., human-AI interaction and the educational use of AI-generated data.

In [11,12], the main uses of ChatGPT in higher education are discerned. One may discern four main types of ChatGPT uses in a university department according to the users they concern: (i) uses addressed to faculty members, (ii) uses addressed to students, (iii) uses addressed to other staff members (e.g., administrative staff members), and (iv) uses

that concern all members of the departments' communities [12]. Indicative general uses of ChatGPT concern assistance in preparing announcements, notes, letters and guidelines, and translation of content. Indicative uses addressed to students are personalized assistance while studying and doing assignments (e.g., the retrieval of useful information, explanations of ambiguous concepts, and answers to questions), guidelines about how to generally structure academic work and presentations, and the preparation of text involving requests to staff members. Indicative uses addressed to faculty members concern assistance in retrieving resources about their courses and assistance in conducting research (e.g., the retrieval and summarization of relevant work and the indication of promising research directions). Further details are available in [11,12].

However, questions are raised about how ChatGPT affects higher education, taking into consideration the potential negative aspects [11]. The introduction of AI tools like ChatGPT in educational settings presents both opportunities and challenges [11]. These challenges include addressing the educational needs with AI, ensuring content validity, and managing ethical and privacy concerns.

In this context, it is insightful to examine the perceptions of higher education students, especially pre-service teachers, on ChatGPT's impact on higher education. This work utilizes a SWOT analysis framework [13,14], a method commonly applied in market research, strategy development within organizations, project planning, and process assessment. Additionally, SWOT analysis has found applications in software engineering, demonstrating its versatility and effectiveness across various domains. In SWOT analysis, strengths and weaknesses are the internal factors, whereas opportunities and threats are external factors.

This paper is structured as follows. Section 2 presents related work. Section 3 presents the aim and research questions of the study. Sections 4 and 5 present the applied methods and the results, respectively. Section 6 discusses aspects of robotics in educational contexts and ChatGPT. Section 7 provides a discussion of the results, and Section 8 outlines the limitations of the research.

2. Related Work

This section establishes the context, identifies gaps or limitations in current knowledge, and highlights the significance of the current study within the broader scholarly conversation in two directions, a broader direction and a more focused one. The scholarly conversation is formed on a broader direction that includes the discussion of the significance of ChatGPT in education [15,16], the impact of ChatGPT in education [17], and the application of ChatGPT in higher education [18–20].

A second, more focused direction that has been formed concerns the use of SWOT analysis in order to understand the use of ChatGPT in teaching and learning. A look at the related work will help to understand how the current research contributes to the existing literature in the field. A search was conducted for the application of SWOT analysis in assessing ChatGPT within educational contexts. The result of this search showed that no empirical study utilizing SWOT analysis with the use of questionnaires was found. The studies that were identified primarily employed SWOT analysis for theoretical evaluations or assessments.

In a previous study [11], a SWOT analysis was carried out based on student perceptions to evaluate the tool's alignment with educational principles and to identify areas requiring further attention or enhancement. Although this initial study was limited in scope, it represents the first effort to employ SWOT analysis to illuminate student perspectives on incorporating ChatGPT into teaching and learning contexts.

A recent systematic review [21] examines the strengths, weaknesses, opportunities, and threats of using ChatGPT in teaching and learning contexts. This review collates findings from various studies to facilitate discussions about the strengths and weaknesses of using ChatGPT in teaching and learning, as well as the opportunities and threats associated with its use in teaching and learning. Using thematic analysis to investigate relevant topics within related articles, they apply the 3P (Presage, Process, and Product)

model of teaching and learning, as originally proposed in [22]. Their SWOT analysis has revealed thirteen strengths, ten weaknesses, five opportunities, and four threats. This analysis can further explain the four paradoxes of ChatGPT: "ChatGPT is 'friend' yet a 'foe', ChatGPT is 'capable' yet 'dependent', ChatGPT is 'accessible' yet 'restrictive', ChatGPT is 'popular' even when 'banned'" [23].

In [24], a qualitative approach with a SWOT design was employed. Guided by the SWOT framework and based on available literature, their work provides an overview of ChatGPT's strengths, which can help identify its various opportunities for education. The review also discusses ChatGPT's weaknesses, which may highlight potential threats. The paper highlights ChatGPT's self-improvement or self-learning capabilities and its ability to provide personalized and real-time feedback as significant strengths. Opportunities identified include the facilitation of complex learning processes and the reduction in teaching workloads, among others. However, they also point out weaknesses, such as ChatGPT's inability to verify the credibility of information and its potential to perpetuate biases and discrimination. The paper notes significant threats, including the "democratization of plagiarism" within education and research. They emphasize that ChatGPT's tendency to amalgamate text from multiple sources can, if used uncritically, lead to plagiarism in academic and student projects [24] (p. 9).

Similar to [24], the theoretical study in [25] also employs a qualitative methodology within a SWOT framework. Their findings underscore ChatGPT's strengths, notably its advanced natural language processing capabilities, ability for self-improvement, and capacity to deliver personalized and real-time feedback. Nevertheless, they identify weaknesses, including the system's shallow comprehension and the difficulties in evaluating the quality of its responses. Furthermore, they highlight threats such as risks to academic integrity and the reinforcement of discrimination. Addressing these issues is critical for ChatGPT's successful adoption in educational contexts.

A SWOT analysis of ChatGPT strategic management and the utilization of technology in education was conducted in [26]. The author claims that SWOT analysis provides a structured framework to assess ChatGPT from different perspectives, including its technical capabilities, how accepted its use is in education, and how prepared the members of the educational setting are to use it. It is mentioned that data items were collected by surveying relevant literature and interviewing AI experts and industry executives. The SWOT analysis findings highlighted various strengths, weaknesses and opportunities for ChatGPT in the education sector. Personalization emerged as a key strength, facilitating individualized learning through customized materials, personalized lesson plans, and targeted feedback for both teachers and students. Accessibility also stands out as a strength, enabling self-paced learning and enhancing access to educational resources, particularly for students with disabilities. Additionally, ChatGPT offers cost-effectiveness by providing valuable insights and predictive analytics for informed decision-making and targeted interventions in educational institutions. Integration with learning technologies further enhances the experience by offering personalized support, resource recommendations, and automated grading. The main weaknesses mentioned concern the lack of human interaction, ethical issues (e.g., plagiarism, cheating), and the dependence on technology. The main threats highlighted are concerns about data privacy (e.g., users' personal data and conversations) and response quality.

A SWOT analysis of ChatGPT was also conducted in [27]. The notable strengths mentioned are the provision of expert solutions and guidance in complex tasks and the ability to assess students' work using rubrics and checklists it generates to provide relevant feedback. These are possible by utilizing expert knowledge it incorporates into various domains. A notable weakness mentioned is the inability to fully comprehend the meaning of the generated text. Opportunities pointed out are the popularity for distance learning and personalized learning support. A threat mentioned is the prohibition of generative AI tools in the education sector. Based on the SWOT analysis, the authors propose how to integrate ChatGPT into teaching and learning practice.

The corresponding work in [28] aims to conduct an examination of the aspects of ChatGPT in relation to the potential utilization of ChatGPT within educational contexts. Specifically, its goals include promoting the integration of ChatGPT in educational settings and providing educators with various methodologies and approaches to ensure the thoughtful and effective integration of ChatGPT into pedagogical or research activities. To this end, a limited-scale SWOT analysis has been performed in order to highlight possible ways that ChatGPT could enhance pedagogical and learning efforts.

Although previous studies have shed light on the integration of ChatGPT into educational contexts, their analyses are primarily based on literature reviews without incorporating firsthand user experiences from the education sector for the derivation of SWOT content. This gap highlights an underexplored research avenue concerning student perspectives. Apart from the study in [11], there has been little investigation into how higher education students view the strengths, weaknesses, opportunities, and threats (SWOT) related to ChatGPT. This new focus constitutes a substantial original contribution of our research. As pointed out in [24], while SWOT analysis provides a comprehensive understanding of ChatGPT's role in education, it falls short in ranking the significance of issues within each category. Our study seeks to build upon and broaden the scope of existing literature by integrating SWOT analysis with a quantitative approach and insights from user experiences, specifically those of student users. An additional novelty of our research is the use of diverse data analysis methods, including descriptive statistics, inferential analysis, clustering, and decision trees. This methodological blend, incorporating both statistical and AI techniques, aims to offer valuable insights into ChatGPT's educational utility.

3. Study Aim and Research Questions

The aim of this study is to investigate the perceptions of pre-service teachers regarding the utility of ChatGPT (GPT-3.5) in academic environments, focusing on its strengths, weaknesses, opportunities, and threats (SWOT). This exploration is guided by the following research questions:

1. How do undergraduate students evaluate the application of ChatGPT in academic settings, specifically assessing its strengths in assisting with tasks like text correction, comprehensive task responses, and paraphrasing; its weaknesses concerning the validity, originality, and potential biases of the information provided; the opportunities it presents for enhancing academic experiences through research experimentation, collaborative projects, and creative expression; and the perceived threats it may have to traditional teaching methods, critical thinking, and human interaction?
2. How does familiarity with ChatGPT influence students' perceptions of its strengths, weaknesses, opportunities, and threats?
3. Are there identifiable groups among undergraduate students characterized by their familiarity and interaction levels with ChatGPT that exhibit distinct perspectives in a SWOT analysis of ChatGPT's role in academic environments?
4. How do undergraduate students' perceptions of ChatGPT's strengths, weaknesses, opportunities, and threats in academic settings vary according to different factors, and what are the key determinants influencing their overall assessment of its utility in educational contexts?
5. What are the primary topics and queries directed to ChatGPT by undergraduate students (pre-service teachers)?
6. What are the main reasons why undergraduate students would or would not recommend ChatGPT to their peers?

4. Methods

4.1. Participants

The study involved the participation of 257 undergraduate students from the School of Education at Democritus University of Thrace, Greece, distributed across the Department of Primary Education and the Department of Education Sciences in Early Childhood.

Each department requires the completion of a minimum of four years for graduation. The selection process was non-random and voluntary. An invitation to participate in the study was posted as an announcement via the eClass platform, which is used by students and faculty members for course management and communication. Interested students voluntarily signed up through a link provided in the invitation posted on the platform. The selection was intended to mirror the demographic makeup of the student populations in these departments, focusing on ensuring a representative mix of gender and year of study. No specific eligibility criteria or exclusion factors were applied beyond being a currently enrolled student in the relevant departments. The majority of respondents were female, comprising 91.1% of the sample, with males representing only 6%. Three percent chose not to disclose their gender. This demographic distribution was anticipated, as the majority of undergraduate students in these departments are typically female. Regarding the year of studies, the majority of the sample comprised third-year students, making up approximately 34% of respondents. These are followed by second-year students at 32%, fourth-year students at 29%, and students extending beyond the typical study period, accounting for 5%. Regarding familiarity with ChatGPT, a significant portion of the respondents, 47%, have never used ChatGPT before, closely followed by 46% who have used it but not extensively, and a minority of 7% who use it frequently.

4.2. Instrumentation

A structured questionnaire was designed to capture a comprehensive view of students' perceptions regarding ChatGPT's application in academic settings. It was structured into five main sections: strengths (5 items), weaknesses (5 items), opportunities (5 items), threats (5 items), and general sentiments (2 items). The items were formulated to assess diverse aspects of ChatGPT's use, such as its utility in academic tasks, concerns about the validity and originality of information provided, potential enhancements to educational experiences, and risks like ethical and privacy issues. Each item was presented as a statement, with participants responding on a Likert scale ranging from 1 (Strongly Disagree) to 5 (Strongly Agree). The questionnaire content was based on the research conducted in [11]. The structure of the questionnaire was based on SWOT analysis. Strengths and weaknesses were the internal factors regarding ChatGPT, whereas opportunities and threats were the external factors concerning the academic setting.

The reliability of the questionnaire was assessed using Cronbach's alpha, a measure of internal consistency. The values obtained were 0.81 for strengths, 0.77 for weaknesses, 0.83 for opportunities, and 0.75 for threats. These values indicate that the questionnaire sections range from acceptable to good in terms of internal consistency, suggesting that the items within each section reliably measure a single underlying construct. The construct validity of the questionnaire was evaluated through Exploratory Factor Analysis (EFA) using oblique promax rotation. This analysis was chosen due to the expected correlations among factors reflecting different aspects of perceptions toward ChatGPT. The EFA identified four distinct factors corresponding to the main sections of the questionnaire, with all items showing factor loadings greater than 0.3. This indicates that each item adequately contributes to its respective factor, supporting the instrument's construct validity. All statistical analyses for reliability and validity testing were performed using Jamovi software version 2.4.12.

Prior to the main study, the questionnaire was pre-tested with a small group of students from the same academic context but not included in the main sample. Feedback from this pre-test was used to refine the wording of items to ensure clarity and to adjust the scale as necessary to better capture the range of responses. Adjustments made based on pre-testing results were aimed at enhancing the questionnaire's face validity and ensuring that the questions were interpreted as intended.

4.3. Procedure

The main questionnaire was administered to the 257 participants via Google Forms, a digital platform chosen for its accessibility and ease of use, ensuring a broad reach among the target population.

Additionally, as part of an elective course taught in the Department of Education Sciences in Early Childhood, a project involving ChatGPT was assigned to 33 students. The project was performed individually or in groups of two to three students. More specifically, students were asked to submit ten queries to ChatGPT that they deemed the most interesting. They were asked to record the queries and the corresponding replies of ChatGPT. The subjects of five of these queries would have to concern their academic studies. The subjects of the other five queries would have to be of general interest; that is, their subjects would have to be beyond their academic studies. Students were also asked to prepare a brief report explaining the following: (i) the reasons why they would recommend the use of ChatGPT to other students and (ii) the reasons for not recommending the use of ChatGPT to other students. It should be mentioned that the elective course consists of lab sessions involving three main aspects: (i) 3D digital storytelling, (ii) robotics, and (iii) AI concepts. The 3D digital storytelling section concerns the implementation of educational digital stories using a cost-free 3D visual programming tool. The course section about robotics concerns robots used in ECE and acquaintance with the various types of robots used in real-world applications. The section about AI concepts concerns AI in ECE and generative AI.

The study was conducted from February to March 2024, a time frame selected to accommodate the academic schedules of the students while maximizing response rates. The recruitment of participants was facilitated through various channels, including email notifications, announcements on the university's learning management system, and posts on departmental bulletin boards. Participation in the study was voluntary, with an emphasis on confidentiality and the anonymous processing of responses to encourage honest and uninhibited feedback. Approval for this study was granted by the Ethics and Deontology Committee in Research of the Department of Education Sciences in Early Childhood, and its endorsement was further confirmed by the General Assembly of the Department.

4.4. Data Analysis

To address the research questions effectively, a combination of statistical and machine learning methods was chosen based on the nature of the data and the specific objectives of each research question. Descriptive statistics, including means and standard deviations, were employed to summarize the questionnaire responses related to perceptions of ChatGPT's strengths, weaknesses, opportunities, and threats (research question 1), setting a foundation for more complex analyses. Inferential analysis (one-way ANOVA) was employed to explore differences in perceptions based on students' familiarity with ChatGPT and other demographic variables (research question 2). This method is appropriate for comparing means across more than two groups, making it ideal for assessing the impact of categorical predictors (demographic characteristics) on continuous outcome variables (students' perceptions). K-means clustering was selected to uncover groups within the data, effectively segmenting a large volume of responses into distinct clusters based on shared characteristics. This method was chosen to identify common themes and categories in perceptions of ChatGPT, enhancing our understanding of how these perceptions are structured and interrelated. K-means clustering was performed for 2 to 8 clusters with 100 repetitions for each cluster number to ensure the stability and reliability of the clustering outcomes. Decision tree analysis was employed to delve deeper into the factors that influence students' willingness to integrate ChatGPT into their educational practices by examining the relationships between their perceptions and demographic or academic characteristics (research question 4). Students' overall assessment of ChatGPT was treated as the dependent variable, and responses to the twenty SWOT items acted as predictors. The analysis was conducted using the rpart() function in the rpart package v4.1.23 [29].

Initially, a fully grown decision tree was generated (setting the complexity parameter to 0). To fine-tune the model, repeated cross-validation was applied to identify the complexity parameter that minimizes cross-validation error. This method involves partitioning the data into k subsets and running the analysis multiple times to ensure reliability. The model's accuracy was evaluated through a 10-fold cross-validation, repeated 10 times. Subsequently, the initial tree was pruned using a complexity parameter of 0.002, leading to the construction of the final decision tree. This last step was executed using the rpart.plot function from the rpart.plot package [30], providing a visual representation of the analysis results. Content analysis was employed to explore the primary topics and queries undergraduates direct towards ChatGPT (research question 6), and their main reasons for recommending or not recommending it to peers (research question 7). This method was chosen to distill responses into meaningful themes and patterns, revealing students' perceptions of ChatGPT's utility and limitations in academic contexts.

5. Results

A detailed analysis of perceptions and concerns regarding the use of ChatGPT among the study participants is shown in Table 1, focusing on its strengths, weaknesses, opportunities, and threats, as well as general sentiments about incorporating ChatGPT into the academic experience. The strengths section reveals a moderate appreciation for ChatGPT's capabilities in correcting and improving texts, giving comprehensive answers to assignments and rephrasing texts, with mean scores ranging from 2.54 to 3.33. Notably, students recognize its potential to adapt to their needs as learners and to enhance the overall learning experience, indicating a positive perception of its utility in educational settings.

Table 1. Descriptive statistics on student perceptions of ChatGPT in academic settings.

Strengths	**Mean**	**SD**
Q1. Students' assessment of the tool's functionality to correct and improve texts written by themselves or others	2.70	1.13
Q2. Students' assessment of the tool's functionality to give a complete answer to a task	2.54	1.21
Q3. Students' assessment of the tool's functionality to provide paraphrasing of a text typed by themselves or others	2.80	1.19
Q4. Students' assessment of the tool's ability to adapt itself to their needs (Do you consider that the tool is able to adapt itself to your needs?)	3.33	1.09
Q5. Students' assessment of the tool's ability to improve their overall learning experience (Do you consider that the tool is able to improve the student's overall learning experience?)	3.07	1.12
Weaknesses		
Q6. Students' assessment about their feeling of uncertainty concerning the validity of the information it provides them (Do you feel uncertain about the validity of the information it provides?)	2.85	1.12
Q7. Students' assessment about their feeling of uncertainty concerning the originality of the texts that the tool produces (Do you feel uncertain about the originality of the texts that the tool produces?)	2.89	1.14
Q8. Students' concern about the possible biases and inaccuracies that may arise since the resulting texts are influenced by algorithms (Are you concerned about the possible biases and inaccuracies that may arise since the resulting texts are influenced by algorithms?)	3.06	1.06

Table 1. *Cont.*

Weaknesses		
Q9. Students' assessment if they are affected given that no sources or references are provided by the tool (Are you affected by the fact that no sources or references are provided by the tool?)	3.49	1.18
Q10. Do you think that ChatGPT can affect the privacy and security of student data in an academic environment?	2.90	1.19
Opportunities		
Q11. Students' assessment if the tool can provide opportunities for experimentation in academic research (Does the tool provide opportunities for experimentation in academic research?)	3.27	1.07
Q12. Students' assessment if the tool can be used to improve collaborative projects and teamwork among students (Can the tool be used to improve collaborative projects and teamwork among students?)	3.14	1.15
Q13. Students' assessment if the tool promotes interdisciplinary and innovative research in the academic community (Does the tool promote interdisciplinary and innovative research in the academic community?)	2.88	1.10
Q14. Students' assessment if the tool enhances accessibility to educational resources for students with different learning needs (Does the tool enhance the accessibility to educational resources for students with different learning needs?)	3.18	1.10
Q15. Students' assessment if the tool enriches language, expression and imagination (Does the tool enrich language, expression and imagination?)	3.06	1.22
Threats		
Q16. Students' concern about possible abuse of ChatGPT, such as creating fake academic content or concerns about plagiarism (Are you concerned about possible abuse of ChatGPT, such as creating fake academic content or concerns about plagiarism)	3.53	1.15
Q17. Students' ethical concerns about privacy and data security in using ChatGPT in academic settings (Do you have ethical concerns about privacy and data security when using ChatGPT in academic settings?)	3.05	1.18
Q18. Students' assessment if the integration of ChatGPT can affect traditional teaching methods (Do you believe that the integration of ChatGPT can affect traditional teaching methods?)	3.39	1.21
Q19. Students' assessment if critical thinking can gradually be weakened by using ChatGPT (Do you believe that critical thinking can gradually be weakened by using ChatGPT?)	3.66	1.18
Q20. Students' concern about a possible weakening of the human dimension of communication/contact (Are you concerned about a possible weakening of the human dimension of communication/contact?)	3.33	1.20

Table 1. *Cont.*

General Questions		
F1. Are you comfortable with the idea of ChatGPT being part of your academic experience?	3.07	1.05
F2. Would you like to receive additional training to better understand and use the ChatGPT tools in your studies?	3.49	1.07

Conversely, the weaknesses highlight uncertainties regarding the validity of information provided by ChatGPT, originality concerns, potential biases, and inaccuracies due to algorithmic determinations, lack of sources or references, and data privacy and security in academic environments. These concerns are reflected in mean scores between 2.85 and 3.49, suggesting that while there is acknowledgment of ChatGPT's helpful aspects, there remains a significant level of apprehension about its reliability and integrity.

Opportunities identified by the study suggest a positive outlook on the potential of ChatGPT in academic research experimentation, improving collaborative projects and teamwork, promoting interdisciplinary and innovative research, enhancing accessibility to educational resources for diverse learning needs, and enriching language, expression, and imagination. Mean scores in this category range from 2.88 to 3.27, illustrating optimism about the beneficial roles ChatGPT can play in educational advancement.

Threats, however, underscore concerns about the misuse of ChatGPT, including the generation of fake academic content, plagiarism, the impact on traditional teaching methods, the weakening of critical thinking, and the reduction in human interaction. These issues are marked with mean scores from 3.05 to 3.66, highlighting a critical awareness of the potential negative impacts of ChatGPT's integration into academic environments.

General questions about comfort with ChatGPT as part of the academic experience and the desire for additional training to better understand and utilize ChatGPT tools in studies received mean scores of 3.07 and 3.49, respectively. This indicates a general willingness to engage with ChatGPT, coupled with a recognition of the need for more knowledge and skills to effectively leverage this tool in educational contexts.

Table 2 presents how familiarity with ChatGPT influences perceptions of its strengths, weaknesses, opportunities, and threats. Participants who frequently use ChatGPT rate its strengths highest (mean = 3.53) and perceive fewer weaknesses compared to less frequent users. Interestingly, those with no experience perceive more threats (mean = 3.36) than frequent users (mean = 3.09), suggesting that familiarity may reduce perceived risks. Opportunities are viewed more positively by frequent users (mean = 3.41), indicating that engagement with ChatGPT correlates with recognizing its potential benefits more strongly. Overall, the data suggest that increased use of ChatGPT leads to a more favorable assessment of its capabilities and less concern over its drawbacks.

Table 2. Influence of familiarity with ChatGPT on perceived strengths, weaknesses, opportunities, and threats.

Familiarity with ChatGPT	Strengths		Weaknesses		Opportunities Threats			
	Mean	SD	Mean	SD	Mean	SD	Mean	SD
I have never used it	2.76	0.877	2.97	0.839	3.04	0.905	3.36	0.877
I have used it, but not much	2.91	0.834	3.13	0.809	3.12	0.804	3.47	0.814
I use it frequently	3.53	0.688	2.91	0.776	3.41	1.045	3.09	0.697

K-means clustering has delineated three distinct clusters based on students' utilization and perceptions of ChatGPT. Overall, these clusters reflect a spectrum of students' interactions with and perceptions toward ChatGPT, ranging from enthusiasm to more cautious

or less engaged, each with unique recognitions of its benefits and concerns. Cluster 1 (43%) represents a group of students with high engagement and a very positive perception towards ChatGPT. These students acknowledge the capabilities of ChatGPT for tasks such as improving texts, providing complete answers, and paraphrasing and believe strongly in its adaptability to their needs and its potential to enhance their learning experience. While they have moderate concerns about the validity and originality of information, as well as potential biases and privacy issues, they highly value the opportunities ChatGPT provides for academic experimentation, collaborative work, and accessibility to educational resources. This cluster shows a high level of comfort with ChatGPT being part of their academic experience and expresses a strong desire for additional training to better utilize these tools.

Cluster 2 (30%) includes students who engage with ChatGPT to a moderate extent and have some reservations about its use. This group's assessment of ChatGPT for editing, task completion, and paraphrasing is notably lower compared to Cluster 1, and their perception of its adaptability and learning enhancement capabilities is moderate. Concerns about the validity and originality of the content generated by ChatGPT, as well as biases and inaccuracies, are more pronounced in this cluster. Despite these reservations, the students acknowledge the potential of ChatGPT for fostering academic experimentation and enhancing collaborative projects. However, their concerns extend significantly to data privacy and the impact on traditional teaching methods, though they still show a very high interest in receiving additional training.

Cluster 3 (27%) is characterized by students with lower engagement and more varied perceptions of ChatGPT. Their assessment of ChatGPT for specific functionalities is generally lower, and they exhibit cautious optimism about its adaptability to student needs and its ability to enrich the learning experience. Concerns in this cluster about the validity and originality of information, as well as algorithmic biases, are present but slightly lower than in Cluster 2. The acknowledgment of ChatGPT's potential for promoting innovation and enhancing collaborative efforts is mixed, indicating a recognition of opportunities but with more reservations. This cluster's concerns about the impact on traditional teaching and critical thinking are similar or slightly lower than those in other clusters, with a mixed level of comfort with ChatGPT in their academic lives and a noticeable interest in further training.

Figure 1 displays the decision tree used to assess the factors influencing students' overall comfort with integrating ChatGPT into their academic experiences. The primary split in the tree is based on Q5, i.e., students' assessment of ChatGPT's ability to improve their overall learning experience. Responses to Q5 below the threshold of 4 (agree) suggest a path of less comfort, whereas responses at or above this threshold indicate a more favorable view of ChatGPT's role in educational enhancement. Further subdivisions in the tree show that the assessment of ChatGPT's adaptability to students' needs (Q4) and its potential impact on privacy and data security (Q10) play significant roles in shaping students' comfort levels. For instance, respondents who perceived ChatGPT as highly adaptable ($Q4 \geq 3$) and had fewer concerns about data privacy ($Q10 < 4$) were more comfortable with its academic integration. Additionally, perceptions of the tool's functionality in providing complete answers to tasks (Q2) also affected comfort levels, with lower ratings correlating with less comfort. The terminal nodes of the tree, representing the outcomes of these decision paths, are color-coded to denote varying levels of comfort with ChatGPT, from red (least comfortable) to green (most comfortable).

Table 3 presents the main categories of queries submitted by students regarding their academic studies and the number of students whose queries are included in each category. Most students submitted queries regarding the future prospects of a person who holds an early childhood education (ECE) degree and general issues involving education sciences. The initials C.A.S. in the table stand for categories of queries regarding academic studies. The table does not include categories of queries that were provided by only one or two students. One may note that the categories of queries included in this table may be organized into three broader categories. These broader categories are(i) categories of

queries involving university students as pre-service teachers (i.e., the first nine categories of queries, C.A.S.1–C.A.S.9), (ii) categories of queries involving aspects that could mainly be considered interesting for in-service teachers or pre-service teachers performing their field practice and practicum in ECE settings (i.e., categories C.A.S.10 to C.A.S.13), and (iii) a discrete category concerning specifically ChatGPT in education (i.e., C.A.S.14).

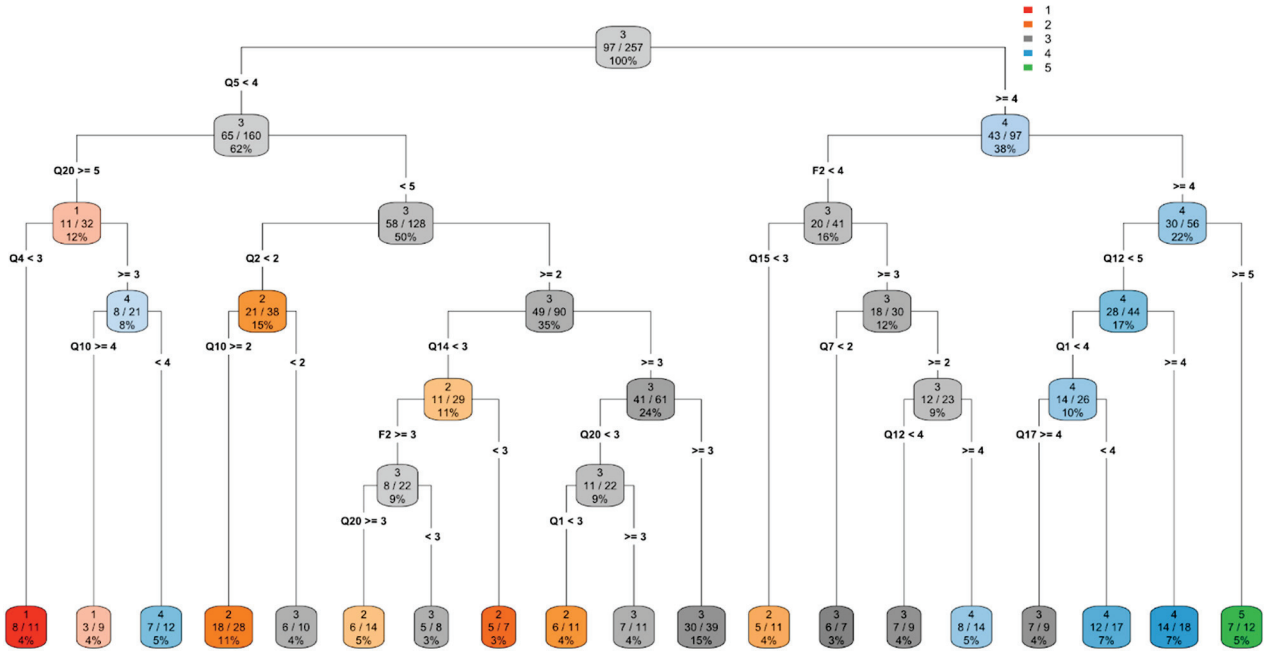


Figure 1. Decision tree analysis of student comfort with ChatGPT integration.

Table 3. Main categories of queries submitted by students regarding their academic studies and the corresponding number of students.

ID	Category of Queries	#Students
C.A.S.1	The future prospects of a person who holds an ECE degree	21
C.A.S.2	General issues involving education sciences (e.g., information about specific terms, information about specific university course outlines)	20
C.A.S.3	The subject of a diploma thesis they could do during their final year of studies	5
C.A.S.4	Preparation for face-to-face written examination of courses (e.g., how to organize study time, most important topics likely to be included in examination questions)	6
C.A.S.5	Preparation of bibliography	5
C.A.S.6	Rights of university students	3
C.A.S.7	Preparation of email message text addressed to staff members	3
C.A.S.8	Learning theories	6
C.A.S.9	Importance of a specific cognitive field in ECE	3
C.A.S.10	Management of an ECE class including students who are non-native speakers or from different cultures	5
C.A.S.11	Support of young students in classroom (maintaining children's focus, support of children with learning difficulties, ways to calm children in the classroom, supporting isolated children to collaborate with other children)	7
C.A.S.12	Teaching of specific subjects in ECE (e.g., mathematics, environmental issues, music)	18
C.A.S.13	Integration of technology (in general) in ECE or integration of specific technological tools in ECE	12
C.A.S.14	Role of ChatGPT in education	8

Table 4 presents the main categories of queries submitted by students regarding subjects of general interest beyond their academic studies and the number of students whose queries are included in each category. We note that most students submitted queries

regarding two main categories: (i) health, healthy eating, or weight loss (C.G.I.1) and (ii) tourist trips (C.G.I.2). The initials C.G.I. in the table stand for categories of queries regarding subjects of general interest.

Table 4. Main categories of queries submitted by students regarding subjects of general interest and the corresponding number of students.

ID	Category of Queries	#Students
C.G.I.1	Health, healthy eating, or weight loss	24
C.G.I.2	Tourist trips	18
C.G.I.3	Gifts or wishes given to other people (e.g., for birthdays)	9
C.G.I.4	Future predictions	9
C.G.I.5	Recommended literary books to read	5
C.G.I.6	Recipes	5
C.G.I.7	Sports (e.g., football)	3

The categories of queries presented in Tables 3 and 4 constitute information that may be useful in assisting the integration of ChatGPT in higher education settings besides Departments in Education Sciences. The categories of queries in Table 4 involve students in all scientific fields. As far as the categories of queries shown in Table 3 are concerned, one may note that a number of them are quite general (i.e., C.A.S.3, C.A.S.4, C.A.S.5, C.A.S.6, C.A.S.7, and C.A.S.14) involving all scientific fields. Several other categories in Table 3 may be slightly changed to adapt them to any scientific field (i.e., C.A.S.1, C.A.S.2, C.A.S.9, C.A.S.12, and C.A.S.13). Only three categories of queries (i.e., C.A.S.8, C.A.S.10, C.A.S.11) may not be adapted to any field. Table 5 presents six of the categories in Table 3 without any change and five others slightly changed to be adapted to any scientific field. Therefore, Table 5 contains eleven categories of queries regarding any scientific field with new IDs. Text in italics corresponds to adaptations of categories in Table 3. The term '[the specific field study]' is a generic term that may be set to any field (e.g., chemical engineering, biology, medicine, computer science, etc.). The categories of queries shown in Table 5 may assist in the design of educational activities in any higher education major.

Table 5. Main categories of queries that may be submitted by students in any field regarding their academic studies.

ID	Category of Queries
C.A.S.1′	The future prospects of a person who holds a degree in *[the specific field of study]*
C.A.S.2′	General issues involving *a [specific field of study]* (e.g., information about specific terms, information about specific university course outlines)
C.A.S.3′	The subject of a diploma thesis they could write during their final year of studies
C.A.S.4′	Preparation for face-to-face written examination of courses (e.g., how to organize study time, most important topics likely to be included in examination questions)
C.A.S.5′	Preparation of bibliography
C.A.S.6′	Rights of university students
C.A.S.7′	Preparation of email message text addressed to staff members
C.A.S.8′	Importance of a cognitive field in *[the specific field of study]*
C.A.S.9′	Teaching of subjects in *[the specific field of study]*
C.A.S.10′	Integration of technology (in general) in *[the specific field] of study* or integration of specific technological tools in *[the specific field of study]*
C.A.S.11′	Role of ChatGPT in *higher* education *or in [the specific field of study]*

Students provided several reasons for recommending ChatGPT to their peers, illustrating a broad spectrum of benefits that span from academic support to personal development

(Table 6). The primary reason, highlighted by fourteen students is the efficiency ChatGPT offers in completing various tasks. This includes searching for information, solving problems, and writing texts, which significantly saves time and enhances productivity. Ten students pointed out specific strengths in ChatGPT's responses, such as their comprehensiveness and understandability. ChatGPT was praised for its ability to provide detailed information and advice on a wide range of topics, although it was noted that for specialized queries, the tool might not always deliver suitable responses. Regarding course-related assistance, eight students appreciated ChatGPT's capability to enrich notes and educational materials and to facilitate the acquisition of new academic content. This aligns with examples shown during a lab session, where ChatGPT's utility in retrieving immediate answers to course-related questions was demonstrated.

Table 6. Summary of reasons for recommending ChatGPT to peers.

Reason for Recommending ChatGPT to Peers	#Students
Students save time in the acquisition of information, with time-efficient responses on any topic	14
Specific aspects regarding the responses given by ChatGPT	10
Assistance provided concerning the courses	8
Assistance provided concerning projects	12
The ease of using the tool and the free access	3
Availability twenty-four hours per day throughout the week	2
Ability to practice in a foreign language	3
Assistance offered to persons that may have spelling difficulties	3
Ability to obtain well-written text with good structure assisting in the preparation of text	3
Assistance that may be provided with everyday issues	2
Assistance in preparing an email message in an academic context or any other context	3
Assistance provided to students who may have difficulties expressing their opinions in public media	3
Answers to various questions about the scientific field of education and the courses taught in such a department	5
Information about the opportunities given to them as members of an academic environment	5
Personalized suggestions if prior information about the user is given, personalization of responses based on previously asked questions	1
Learning how to prepare specific and comprehensive questions	1

Project support was another significant advantage mentioned by twelve students, who valued ChatGPT for its ideation and organizational assistance and for offering innovative perspectives on project topics. This reflects ChatGPT's role in fostering creativity and enhancing academic projects. Ease of use and free access were highlighted by three students as key factors contributing to ChatGPT's appeal. These aspects underscore the importance of user-friendly and accessible tools in education. Additionally, the 24/7 availability of ChatGPT was noted by two students as particularly beneficial, accommodating students' varied schedules and supporting distance learning. Three students mentioned the tool's utility in foreign language practice, despite some inaccuracies, emphasizing its potential in language learning. Another three students discussed how ChatGPT helps overcome spelling challenges and assists in generating well-structured texts, thus contributing to improved writing skills.

ChatGPT's versatility was also recognized in its ability to provide solutions for everyday issues, with two students discussing how it offers practical advice for non-academic queries. Similarly, three students valued ChatGPT's assistance in drafting emails, particularly in academic settings, which supports effective communication. In the realm of

public expression, ChatGPT was seen as beneficial for those who struggle with articulating their opinions in online forums and social media. This aspect, alongside its educational benefits, such as answering queries related to the field of education and offering insights into maximizing academic opportunities, was appreciated by five students. One student highlighted ChatGPT's capacity for personalization, suggesting that the tool can tailor its responses when provided with specific user information, adding a layer of individualized support. An emerging theme mentioned by one student relates to AI literacy. Interacting with ChatGPT not only aids in acquiring desired information but also in learning to formulate specific and comprehensive queries, a skill that will become increasingly important in the digital age.

Students also mentioned several reasons for not recommending ChatGPT to their peers, which are mainly aligned with the identified weaknesses and threats in the questionnaire (Table 7). The most common concern, shared by ten students, revolves around ChatGPT's tendency to provide inaccurate, vague, or misleading information, especially on complex or controversial topics. This issue stems from ChatGPT's limited understanding of the context and the complexity of certain topics, which are exacerbated when users do not clearly articulate their queries. A related concern mentioned by two students is the need for clear expression of queries, highlighting that unclear queries might not yield the desired responses. Additionally, there's a noted need to double-check ChatGPT's responses for accuracy, particularly if these responses are to be used in academic projects, as ChatGPT might provide different answers for similar queries submitted by various users. Eight students expressed doubts about the quality of information due to the absence of cited sources in ChatGPT's responses. This lack of citations raises questions about the validity, accuracy, and completeness of the information, which is critical in academic settings where source citation is paramount. This reliance on ChatGPT without proper citation can lead to academic integrity issues, including plagiarism, and is deemed unsuitable for significant academic endeavors like diploma theses or semester projects that require impartiality and a range of validated sources.

Twelve students raised concerns about the impact of ChatGPT on critical thinking and problem-solving skills. They argued that excessive reliance on ChatGPT for information retrieval might lead to an addiction, thereby reducing, hindering, or even completely eliminating critical thinking and creativity as individuals become accustomed to receiving ready information without engaging in the search and analysis process themselves. Three students pointed out that interaction with ChatGPT could diminish writing fluency, as reliance on the tool might bypass the learning process involved in mastering grammar, syntax, and vocabulary. Moreover, there are security concerns, with two students highlighting the potential risks to users' identity and personal data online. Additionally, two students noted weaknesses in ChatGPT's ability to formulate texts in Greek, spotting some errors in the produced text. One student specifically mentioned the importance of students learning to compose emails to faculty members independently, emphasizing the development of formal communication skills. Two groups of students sought reasons from ChatGPT itself for not recommending its use. Their responses included concerns over the accuracy of information, the tool's inability to cater to individual needs, the importance of self-education, and the risks of plagiarism and privacy violations. These reasons highlight a cautious approach to using ChatGPT, suggesting a balance between leveraging technological tools and traditional learning methods for a comprehensive educational experience.

Table 7. Summary of reasons for not recommending ChatGPT to peers.

Reason for Not Recommending ChatGPT to Peers	#Students
Provision of inaccurate, vague, or misleading information, especially on complex or controversial topics. Inability of ChatGPT to thoroughly comprehend the context of certain topics and the sensitivity to the input provided by users. The need for users to express their queries with clarity. The need to double-check the responses acquired from the tool. ChatGPT provides different responses about the same or similar queries when these are submitted by different persons.	12
The sources of the acquired information are not cited, creating doubts about the quality of the retrieved information. Students' deliverables may be rejected if they rely on ChatGPT due to the inexistence of citations to sources and possible plagiarism issues. In diploma theses and semester projects, there should be impartiality and validation requiring the citation of a range of valid sources.	8
Reduction, hindrance, absence of critical thinking and problem-solving skills, as well as creativity. ChatGPT does not employ a discovery learning approach but readily lists all information items. Some persons are probably (or could become) too lazy to look for or to reach answers through thinking and discovery. Replacement to a certain degree of books and other learning items by the tool.	12
Interaction with ChatGPT reduces the fluency of writing. Inactivity and passivity of students' thinking, in terms of writing, preparing an original text, avoiding mistakes, and personal speech development.	3
Users' identity and personal data may not be totally safe.	2
Slight weakness in the formulation of texts in Greek and some mistakes may be spotted in the produced text.	2
It is important for students to prepare by themselves the email messages addressed to faculty members and gain relevant experience	1

6. Robotics in Educational Contexts and ChatGPT

The curriculum in ECE and primary education incorporates various technological resources. Generally speaking, the integration of technological resources in ECE, primary education, and subsequent levels of education serves three main purposes: (i) acquaintance of students with technology and the use of specific technological resources, (ii) the use of technological resources in all parts of the curriculum as learning tools, and (iii) the comprehension of the role that technology plays in society [31].

Robots are among the technological resources used in ECE and primary education. Robots have certain unique features compared to other technological devices that attract the interest of students. These features involve, among other things, mobility, sensors, sounds, lights and physical interaction with their environment. Programming is also combined with robotics. Furthermore, the various parts of robotic devices may be assembled, enhancing the creativity of students. Additional components and other relevant materials may be manipulated, assembled, or constructed by students with the guidance of teachers [32].

Robots in ECE and primary education are used to introduce students to and raise their interest in various technological aspects, including engineering, electronics, and programming. Robotics may also provide motives to students to pursue relevant studies in higher education and to seek relevant jobs. Cross-curricular activities are implemented with

robotics concerning language, mathematics, natural sciences, environmental education, arts, and other fields. The implementation of robotic activities promotes discussions and collaboration among students, creative thinking, and problem-solving skills.

In ECE, usual types of robots are devices programmed to move on the floor and other surfaces (e.g., desks or tables). In this context, a robot is programmed to follow a specific route, that is, to reach a destination, taking into consideration its starting point and its initial direction. A route may include intermediate destinations besides the final destination. The space in which the robot moves may also include obstacles that need to be avoided. The programming instructions involve symbols that define the main directions that a robot may follow and other necessary functions. A sequence of instructions is given to the robot in order to follow the desired route. The sequence of instructions may be considered a type of program the creation of which is based on discussion and collaboration. This program needs to be debugged in case the results of its execution are not the desired ones. According to the type of robot used, the sequence of instructions may be given in various ways such as the following: (a) by pressing a sequence of buttons on the robot's surface, (b) by using a sequence of specialized cards that are scanned by robot sensors, (c) by using a sequence of tangible objects whose function is recognized by the robot wirelessly, (d) through a block-based programming application running on a device (e.g., tablet) with wireless connection to the physical robot, or (e) in a hybrid way that combines two or more of the previous types of interaction.

In the lower grades of primary education, robots similar to the ones used in ECE (or advanced models) may be employed. In subsequent grades, robots are usually combined with block-based or text-based programming (e.g., Scratch 3.0, Python 3.12.3). Typical examples are the robotic kits for the micro:bit circuit platform. The micro:bit circuit platform may be used to implement activities concerning electronics and programming. Its combination with robotic kits enables the implementation of activities combining robotics, electronics, and programming.

An aspect of interest is the use of social robots, especially AI-based ones, in ECE and primary education. Their form and functionality vary. Social robots are able to adapt to their environment and interact with individual students and groups of students. They may be utilized in typical and special education settings [31].

Robotic concepts are part of the curriculum in the two university departments involved in this study. In the Department of Education Sciences in Early Childhood, robotic concepts are taught as sections in three courses (two obligatory and one elective). The elective course for which results were given in Section 5 (based on a project about ChatGPT) involves a section about robotics. In their projects, certain students attending the course submitted queries to ChatGPT whose subject involved robotics. Some of the submitted queries and the retrieved results will be analyzed. One may note that the responses of ChatGPT are satisfactory.

A group of three students submitted a query about the positive and negative aspects of robots in society. One may consider that this query concerns the overall role of technology in society, which constitutes a main goal in the integration of technology in education. Positive aspects mentioned by ChatGPT involved improved productivity, advanced solutions in health and improved quality of life for persons with mobility difficulties or other disabilities. Negative aspects mentioned involved the loss of jobs for people due to automation and ethical issues about privacy and liability. The overall comment of ChatGPT is that the integration of robots and technology into our lives is a complex issue that requires attention to ethics, social consequences, and understanding how to manage them for the common good.

Another group of students submitted a query about how to use Bee-Bot (i.e., a type of robot addressed to young students) in ECE. This query concerns the use of a technological resource such as a robot in all parts of the curriculum. The corresponding reply of ChatGPT involved the following: (i) introduction to spatial directions and robot functionality, (ii) programming of the robot to follow a route, (iii) the use of the robot to introduce mathematical

concepts (e.g., counting and geometrical figures), (iv) collaborative and problem-solving activities in combination with programming, (v) the use of the robot as a character in stories and games. The general assertion of ChatGPT is the use of the robot in a way that promotes creativity and experimentation.

A group of three other students submitted a query about the uses of robotics (in general) in ECE. ChatGPT provided a good summary of uses of robotics. More specifically, the uses mentioned were the following: (i) introduction to technology, (ii) problem solving activities, (iii) early coding concepts, (iv) development of fine motor skills, (v) spatial awareness, (vi) creativity and imagination, (vii) cross-curricular integration, (viii) sequencing, (ix) early exposure to engineering concepts, (x) engagement and motivation, and (xi) preparation for future learning.

Other groups of students submitted queries about the integration of technology (in general) in ECE. ChatGPT mentioned robots among the different types of technological resources that may be used in ECE. Another group of students submitted a query about how AI may be used to assist children in special education settings. ChatGPT provided several corresponding functions. Certain functions that AI may perform in this context according to ChatGPT may be incorporated in robots interacting with children (e.g., facial recognition, emotional recognition, speech recognition, emotional and behavioral support, personalized learning, and adaptive learning content displayed in screens on robots) [31].

Table 8 summarizes indicative topics of queries concerning robots in educational contexts that may be submitted to ChatGPT. These topics of queries correspond to the three main aforementioned purposes of the integration of technological resources in education (in the specific case of robots): (i) acquaintance of students with robots and the use of specific robots and components, (ii) use of robots in all parts of the curriculum as learning tools, and (iii) comprehension of the role that robots play in society.

Table 8. Indicative topics of queries about robotics in educational contexts that may be submitted to ChatGPT.

Indicative Topics of Queries
Role of robots in society
Main types of robots used in real-world applications
How a specific type of robot is used in real-world applications
How to follow a career in robotics
The subject of a diploma thesis about educational robotics that may be done during undergraduate or postgraduate studies
How to use a specific robot in learning activities (in general)
How to use a specific robot in learning activities involving specific subjects (e.g., mathematics, natural sciences, arts)
How to use a specific robot in cross-curricular activities
How to use a specific robot in learning activities involving specific goals (e.g., improving collaboration among students)
How to use a specific robot in learning activities in combination with other technological resources
How to introduce a specific robot to students for the first time
The robots that may be used in a specific educational level
Instructions about the functionality of a specific robot and additional components
Ideas about constructions that students may do in robotic activities
Ideas about programming activities associated with specific robots
Ideas about projects combining electronics, robotics and programming
Ideas about using robots in special education
Ideas about how to utilize social robots in classroom
Ideas about how to utilize social robots to assist students individually and in groups
Ideas about how to combine AI with robots in educational contexts

7. Discussion

Overall, this study reflects a nuanced perspective among participants, acknowledging both the promising applications and the challenges of integrating ChatGPT into academic

settings. This balanced view underscores the importance of addressing the limitations and concerns associated with ChatGPT while exploring its potential to enrich the educational landscape.

The results indicate a moderately positive view of ChatGPT's benefits, with both strengths and opportunities being recognized. However, notable concerns about potential negatives, particularly threats—which received the highest average score—suggest significant apprehension regarding ChatGPT's negative impacts.

In summary, the study shows that students' comfort with ChatGPT is influenced by both its strengths—like adaptability and enhancing learning experiences—and concerns about weaknesses and threats, such as data privacy, originality of work, and the potential introduction of biases or inaccuracies. Opportunities for improving academic research and collaboration also emerged as significant factors.

The reasons for recommending ChatGPT span from its practical benefits in academic task efficiency, project support, and language practice to its broader impact on personal development, daily life assistance, and the cultivation of AI literacy. These insights, combined with specific examples and responses generated by ChatGPT itself, offer a comprehensive understanding of the tool's multifaceted value to students. While presenting numerous advantages, students also recognize ChatGPT's significant limitations and risks, particularly concerning information accuracy, critical thinking development, security concerns, and the importance of traditional skills. These insights suggest a nuanced perspective on the integration of AI tools in educational contexts, emphasizing the need for critical engagement and balanced use.

Other papers have utilized SWOT analysis to evaluate ChatGPT, presenting its strengths, weaknesses, opportunities, and threats. Our approach diverges in that it addresses certain issues that are not explored in these studies. Notably, unlike other research, our methodology included a ranking of issues, which was made possible by collecting quantitative data from students. Additionally, our use of a clustering approach allowed for a better understanding of how students perceive ChatGPT, and decision tree analysis helped identify key factors influencing their comfort with the tool. Our research also provided new insights into the types of queries students submit to ChatGPT, revealing their areas of interest—information that is absent in related works but could be invaluable for integrating ChatGPT into higher education settings. Moreover, analyzing the reasons students might recommend or not recommend ChatGPT offered valuable feedback for educational program design and further development of the tool. Another issue discussed is the use of ChatGPT to provide information about robotics in educational contexts.

This study employed ChatGPT (GPT-3.5) rather than the more advanced ChatGPT (GPT-4), which is currently accessible exclusively through a subscription fee. The choice to use the more accessible version of ChatGPT, which is free of charge, aligns with the practical constraints faced by students and educational institutions. Often, students are either unable or unwilling to pay for digital tools that may be required during their academic journey. Similarly, it is not always feasible for institutions or researchers to cover subscription costs for all potential digital tools that could be used by students and staff. This preference for cost-free tools addresses aspects of the digital divide, ensuring broader access. While this choice of tool version is a limitation, it does not appear to significantly influence the results presented in this study.

Future research could benefit from longitudinal studies that track changes in students' perceptions and usage of AI tools like ChatGPT over time. Such studies would provide deeper insights into the evolving relationship students have with AI technologies and whether prolonged exposure impacts their trust and reliance. Time is required to adopt technological innovations in education, and this is achieved when each person achieves this at their own pace [33]. Additionally, expanding the demographic scope of the studies to include a wider range of educational institutions, disciplines, and cultural contexts would enhance the generalizability of the results and offer a broader understanding of AI's role in diverse educational settings.

The integration of AI tools into educational practices also presents a promising area for further investigation. Future studies could explore effective strategies for embedding AI into different teaching and learning methodologies, assessing their impact on educational outcomes. Alongside this, there is a clear need for the development of comprehensive AI literacy programs that educate both students and educators about the potential and limitations of AI [34,35]. Such programs would support a more informed and effective use of AI technologies. Moreover, developing robust ethical guidelines and policy frameworks is essential for guiding the use of AI in education. This includes addressing concerns related to privacy, academic integrity, and the balance between AI and traditional educational methods. Detailed impact studies could also be conducted to examine how AI tools affect critical thinking, creativity, and interpersonal skills, which are pivotal in students' educational development. Comparative studies between ChatGPT and other AI educational tools could provide additional insights into their respective efficiencies and help identify best practices for their implementation in educational settings.

In addition to the aforementioned directions, integrating qualitative methods such as interviews or focus groups would be highly beneficial for future studies. These approaches can offer deeper insights into the nuanced opinions and experiences of students using ChatGPT. By engaging directly with students through these methods, researchers can capture detailed responses and explore complex attitudes that are often not accessible through quantitative measures alone. Such qualitative data could greatly enrich our understanding of how students perceive the effectiveness, usability, and impact of AI tools in their academic lives, providing a more holistic view of the integration of technology in education.

AI tools like ChatGPT can potentially serve to significantly enhance student engagement. By providing interactive and personalized learning experiences, large language models (LLMs) can capture students' interest and motivation. Drawing on their natural language processing capabilities, LLMs can adapt to individual learning styles and preferences, delivering tailored content to address the unique needs of each student. This customized approach fosters greater engagement with course materials, stimulating active involvement and improving knowledge retention. By maintaining ongoing interaction and providing feedback, LLMs can identify students' strengths, weaknesses, and learning preferences, empowering educators to customize learning paths and resources accordingly. This personalized approach not only accommodates diverse learning styles but also addresses individual learning requirements, ultimately enhancing academic achievement and student satisfaction.

Furthermore, integrating AI tools such as ChatGPT can address various educational challenges, including accessibility and inclusivity. ChatGPT can serve as a valuable resource for students with disabilities, offering alternative means of accessing course materials and support. Additionally, by providing real-time assistance and guidance, ChatGPT can bridge learning gaps and ensure equitable learning opportunities for all students, regardless of their backgrounds or abilities. By situating ChatGPT within the broader landscape of educational innovation, our study illuminates the transformative role of technology in shaping academic environments. Through the utilization of AI-driven tools like ChatGPT, educators can revolutionize teaching and learning practices, fostering greater engagement, personalization, and inclusivity. Embracing innovative solutions like ChatGPT becomes essential for educators to meet the evolving needs of learners and create dynamic and enriching educational experiences as technology continues to evolve.

To ensure that the findings from this study are actionable and beneficial, we offer several specific recommendations for educational decision-makers aimed at optimizing the integration of AI tools such as ChatGPT in academic settings. Firstly, it is advisable for educational leaders to devise a structured strategy for adopting and integrating these technologies. This could include pilot programs to evaluate the tool's effectiveness within particular educational contexts before wider deployment. Additionally, it is essential to provide comprehensive training for both students and faculty, which should not only cover

how to use these tools effectively but also include discussions on ethical considerations, data security, and best practices for technology integration in educational processes.

Furthermore, establishing clear ethical guidelines and policies is critical. These policies should address data privacy, prevent academic dishonesty, and ensure the integrity of educational assessments [36]. In terms of support for diverse learning needs, decision-makers should utilize AI to enhance accessibility and inclusivity, tailoring technologies to provide personalized learning experiences that meet individual student requirements.

Regular evaluation and feedback mechanisms should also be implemented to continually assess the impact of these AI tools on educational outcomes. Such feedback, gathered from both students and faculty members, can drive iterative improvements to both the technology itself and its application in educational settings.

Moreover, fostering a culture of innovation within educational institutions is crucial. Encouraging faculty members and students to explore and experiment with AI technologies can be supported by establishing innovation labs or centers dedicated to educational technology research and development. Lastly, it is vital to address technological inequities to ensure that the deployment of AI tools like ChatGPT does not widen the digital divide. This includes providing necessary technological resources to all students to ensure that everyone can benefit equitably from AI integration [35–38].

8. Limitations of the Research

Our study primarily focused on pre-service teachers with the aim of understanding their perceptions of ChatGPT and its potential utility in educational settings. This targeted approach was chosen due to the crucial role pre-service teachers play as future educators and early adopters of new educational technologies. While this focus provided valuable insights into the perspectives of this specific group, it also limited the generalizability of our findings to a broader population of educators and students. Further investigation into the pedagogical integration of AI technologies like ChatGPT across different educational levels and disciplines could offer deeper insights into the specific needs and challenges associated with such implementations.

In addition, expanding the scope of the research to include students from various academic disciplines, cultural backgrounds, and educational levels would enhance our understanding of the broader implications and utility of ChatGPT across different contexts. Future studies could aim to involve a wider range of participants, which would help to identify specific needs and perceptions across disciplines, potentially leading to more tailored and effective integration strategies for AI tools in education. Additionally, comparative studies could be conducted to highlight any significant differences or similarities in perceptions between pre-service teachers and other student groups, thereby providing deeper insights into the pedagogical integration of AI technologies like ChatGPT.

Additionally, the reliance on a structured questionnaire for data collection, while efficient, may not capture the depth and nuances of students' experiences and opinions regarding ChatGPT. Such a methodological approach might overlook the complexity of students' interactions with AI tools and their impact on learning processes. Furthermore, the study's cross-sectional design does not allow for the assessment of changes in students' perceptions over time, particularly as they gain more experience with ChatGPT or as the technology itself evolves. Lastly, the analytical methods employed, including descriptive statistics and inferential analysis, provide a snapshot of current attitudes but might not fully elucidate the underlying reasons for these perceptions or explore potential long-term implications for teaching and learning.

Author Contributions: Conceptualization, A.M., J.P. and M.S.; methodology, A.M., J.P. and M.S.; software, A.M.; validation, A.M.; data curation, A.M., J.P. and M.S.; writing—original draft preparation, A.M., J.P. and M.S.; writing—review and editing, A.M., J.P. and M.S.; visualization, A.M. All authors have read and agreed to the published version of the manuscript. The names of the authors appear in alphabetic order.

Funding: This research received no external funding.

Institutional Review Board Statement: The study was conducted in accordance with the Declaration of Helsinki and the protocol was approved by the Ethics and Deontology Committee in Research of the Department of Education Sciences in Early Childhood (30475/649/12 February 2024), and its endorsement was further confirmed by the General Assembly of the Department (11/14 February 2024).

Informed Consent Statement: Not applicable.

Data Availability Statement: Dataset available upon request from the authors.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AI	Artificial Intelligence
AIED	Artificial Intelligence in Education
EFA	Exploratory Factor Analysis
ECE	Early Childhood Education
CAS	Categories of queries regarding academic studies
CGI	Categories of queries regarding subjects of general interest
LLM	Large Language Model

References

1. Aljawarneh, S.A. Reviewing and exploring innovative ubiquitous learning tools in higher education. *J. Comput. High. Educ.* **2020**, *32*, 57–73. [CrossRef]
2. Chen, L.; Chen, P.; Lin, Z. Artificial intelligence in education: A review. *IEEE Access* **2020**, *8*, 75264–75278. [CrossRef]
3. Bozkurt, A.; Karadeniz, A.; Baneres, D.; Guerrero-Roldán, A.E.; Rodríguez, M.E. Artificial intelligence and reflections from educational landscape: A review of AI Studies in half a century. *Sustainability* **2021**, *13*, 800. [CrossRef]
4. Wang, Y.; Liu, C.; Tu, Y.F. Factors affecting the adoption of AI-based applications in higher education. *Educ. Technol. Soc.* **2021**, *24*, 116–129.
5. Zawacki-Richter, O.; Marín, V.I.; Bond, M.; Gouverneur, F. Systematic review of research on artificial intelligence applications in higher education–where are the educators? *Int. J. Educ. Technol. High. Educ.* **2019**, *16*, 39. [CrossRef]
6. Hinojo-Lucena, F.J.; Aznar-Díaz, I.; Cáceres-Reche, M.P.; Romero-Rodríguez, J.M. Artificial intelligence in higher education: A bibliometric study on its impact in the scientific literature. *Educ. Sci.* **2019**, *9*, 51. [CrossRef]
7. Perikos, I.; Grivokostopoulou, F.; Hatzilygeroudis, I. Assistance and feedback mechanism in an Intelligent Tutoring System for teaching conversion of Natural Language into Logic. *Int. J. Artif. Intell. Educ.* **2017**, *27*, 475–514. [CrossRef]
8. Chrysafiadi, K.; Virvou, M. Evaluating the integration of fuzzy logic into the student model of a web-based learning environment. *Expert Syst. Appl.* **2012**, *39*, 13127–13134. [CrossRef]
9. Karipidis, N.; Prentzas, J. A survey of factors affecting the successful integration of ICT in education. In Proceedings of the 10th International Technology, Education and Development Conference (INTED 2016), Valencia, Spain, 7–9 March 2016; pp. 8456–8466. [CrossRef]
10. Turing, A. Computing machinery and intelligence. *Mind* **1950**, *59*, 433–460. [CrossRef]
11. Prentzas, J.; Sidiropoulou, M. Assessing the use of Open AI ChatGPT in a University Department of Education. In Proceedings of the 14th International Conference on Information, Intelligence, Systems & Applications (IISA 2023), Volos, Greece, 10–12 July 2023; pp. 1–4. [CrossRef]
12. Prentzas, J.; Sidiropoulou, M. Integrating OpenAI Chat-GPT in a University Department of Education: Main types of use and preliminary assessment results. In *Advances in Information, Intelligence, Systems, and Applications*; Lecture Notes in Networks and Systems; Bourbakis, N., Tsihrintzis, G.A., Virvou, M., Jain, L.C., Eds.; Springer: Berlin/Heidelberg, Germany, 2023, *in press*.
13. Leigh, D. SWOT analysis. In *Handbook of Improving Performance in the Workplace*; Silber, K.H., Foshay, W.R., Watkins, R., Leigh, D., Moseley, J.L., Dessinger, J.C., Eds.; Wiley: San Francisco, CA, USA, 2009; Volume 2, pp. 115–140, ISBN 978-047-019-069-2. [CrossRef]
14. Puyt, R.W.; Lie, F.B.; Wilderom, C.P. The origins of SWOT analysis. *Long Range Plan.* **2023**, *56*, 102304. [CrossRef]
15. Mhlanga, D. Open AI in Education, the Responsible and Ethical Use of ChatGPT towards Lifelong Learning. Available online: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4354422 (accessed on 11 February 2023).
16. Sullivan, M.; Kelly, A.; McLaughlan, P. ChatGPT in higher education: Considerations for academic integrity and student learning. *J. Appl. Learn. Teach.* **2023**, *6*, 31–40. [CrossRef]
17. Lo, C.K. What is the impact of ChatGPT on education? A rapid review of the literature. *Educ. Sci.* **2023**, *13*, 410. [CrossRef]

18. Chamorro-Atalaya, O.; Olivares-Zegarra, S.; Sobrino-Chunga, L.; Guerrero-Carranza, R.; Vargas-Diaz, A.; Huarcaya-Godoy, M.; Rasilla-Rovegno, J.; Suarez-Bazalar, R.; Poma-Garcia, J.; Cruz-Telada, Y. Application of the Chatbot in university education: A bibliometric analysis of indexed scientific production in SCOPUS, 2013–2023. *Int. J. Learn. Teach. Educ. Res.* **2023**, *22*, 281–304. [CrossRef]
19. Ismail, F.; Tan, E.; Rudolph, J.; Crawford, J.; Tan, S. Artificial Intelligence in higher education. A protocol paper for a systematic literature review. *J. App. Learn. Teach.* **2023**, *6*, 56–63. [CrossRef]
20. Vargas-Murillo, A.R.; de la Asuncion Pari-Bedoya, I.N.M.; de Jesús Guevara-Soto, F. Challenges and opportunities of AI-assisted learning: A systematic literature review on the impact of ChatGPT usage in higher education. *Int. J. Learn. Teach. Educ. Res.* **2023**, *22*, 122–135. [CrossRef]
21. Mai, D.T.T.; Da, C.V.; Hanh, N.V. The use of ChatGPT in teaching and learning: A systematic review through SWOT analysis approach. *Front. Educ.* **2024**, *9*, 1328769. [CrossRef]
22. Biggs, J.; Kember, D.; Leung, D.Y.P. The revised two-factor study process questionnaire: R-SPQ-2F. *Br. J. Educ. Psychol.* **2001**, *71*, 133–149. [CrossRef] [PubMed]
23. Lim, W.M.; Gunasekara, A.; Pallant, J.L.; Pallant, J.I.; Pechenkina, E. Generative AI and the future of education: Ragnarök or reformation? A paradoxical perspective from management educators. *Int. J. Manag. Educ.* **2023**, *21*, 100790. [CrossRef]
24. Farrokhnia, M.; Banihashem, S.K.; Noroozi, O.; Wals, A. A SWOT analysis of ChatGPT: Implications for educational practice and research. *Innov. Educ. Teach. Int.* **2023**, *61*, 460–474. [CrossRef]
25. Mesiono, M.; Fahada, N.; Irwansyah, I.; Diana, D.; Siregar, A.S. SWOT analysis of ChatGPT: Implications for educational practice and research. *J. Manaj. Kepemimp. Dan Supervisi Pendidik.* **2024**, *9*, 181–196.
26. Alabool, H.M. ChatGPT in education: SWOT analysis approach. In Proceedings of the International Conference on Information Technology (ICIT 2023), Amman, Jordan, 9–10 August 2023; pp. 184–189. [CrossRef]
27. Zhu, C.; Sun, M.; Luo, J.; Li, T.; Wang, M. How to harness the potential of ChatGPT in education? *Knowl. Manag. E-Learn.* **2023**, *15*, 133–152. [CrossRef]
28. Murad, I.A.; Surameery, N.M.S.; Shakor, M.Y. Adopting ChatGPT to enhance educational experiences. *Int. J. Inf. Technol. Comput. Eng.* **2023**, *3*, 20–25. [CrossRef]
29. Therneau, T.; Atkinson, B.; Ripley, B. Rpart: Recursive Partitioning and Regression Trees. R Package Version 4, 1–13. Available online: https://cran.r-project.org/package=rpart (accessed on 5 December 2023).
30. Milborrow, S. Plot Rpart Models: An Enhanced Version of Plot.rpart. Available online: http://www.milbo.org/rpart-plot/index.html (accessed on 5 December 2023).
31. Prentzas, J. Artificial Intelligence methods in early childhood education. In *Artificial Intelligence, Evolutionary Computing and Metaheuristics. In The Footsteps of Alan Turing*; Studies in Computational, Intelligence; Yang, X.S., Ed.; Springer: Berlin/Heidelberg, Germany, 2013; Volume 427, pp. 169–199. ISBN 978-364-229-693-2. [CrossRef]
32. Alnajjar, F.; Bartneck, C.; Baxter, P.; Belpaeme, T.; Cappuccio, M.; Di Dio, C.; Eyssel, F.; Handke, J.; Mubin, O.; Obaid, M.; et al. *Robots in Education: An Introduction to High-Tech Social Agents, Intelligent Tutors, and Curricular Tools*; Routledge: New York, NY, USA, 2021; ISBN 978-100-314-270-6.
33. Blau, I.; Shamir-Inbal, T. Digital competences and long-term ICT integration in school culture: The perspective of elementary school leaders. *Educ. Inf. Technol.* **2017**, *22*, 769–787. [CrossRef]
34. Ng, D.T.K.; Leung, J.K.L.; Chu, S.K.W.; Qiao, M.S. Conceptualizing AI literacy: An exploratory review. *Comput. Educ. Artif. Intell.* **2021**, *2*, 100041. [CrossRef]
35. Long, D.; Magerko, B. What is AI literacy? Competencies and design considerations. In Proceedings of the International Conference on Human Factors in Computing Systems (CHI 2020), Honolulu, HI, USA, 25–30 April 2020; pp. 1–16. [CrossRef]
36. Chan, C.K.Y. A comprehensive AI policy education framework for university teaching and learning. *Int. J. Educ. Technol. High. Educ.* **2023**, *20*, 38. [CrossRef]
37. Santiago-Ruiz, E. Writing with ChatGPT in a context of educational inequality and digital divide. *Int. J. Educ. Dev. Using Inf. Commun. Technol.* **2023**, *19*, 28.
38. Khowaja, S.A.; Khuwaja, P.; Dev, K. Chatgpt needs spade (sustainability, privacy, digital divide, and ethics) evaluation: A review. *arXiv* **2023**, arXiv:2305.03123. [CrossRef]

electronics

MDPI

Article

Framework for Integrating Generative AI in Developing Competencies for Accounting and Audit Professionals

Ionuț-Florin Anica-Popa [1,*], Marinela Vrîncianu [1], Liana-Elena Anica-Popa [1], Irina-Daniela Cișmașu [2] and Cătălin-Georgel Tudor [1]

[1] Department of Management Information Systems, Bucharest University of Economic Studies, 010374 Bucharest, Romania; marinela.vrincianu@cig.ase.ro (M.V.); liana.anica@cig.ase.ro (L.-E.A.-P.); catalin.tudor@cig.ase.ro (C.-G.T.)

[2] Department of Financial and Economic Analysis and Valuation, Bucharest University of Economic Studies, 010374 Bucharest, Romania; irina.cismasu@cig.ase.ro

* Correspondence: ionut.anica@ase.ro

Abstract: The study aims to identify the knowledge, skills and competencies required by accounting and auditing (AA) professionals in the context of integrating disruptive Generative Artificial Intelligence (GenAI) technologies and to develop a framework for integrating GenAI capabilities into organisational systems, harnessing its potential to revolutionise lifelong learning and skills development and to assist day-to-day operations and decision-making. Through a systematic literature review, 103 papers were analysed, to outline, in the current business ecosystem, the competencies' demand generated by AI adoption and, in particular, GenAI and its associated risks, thus contributing to the body of knowledge in underexplored research areas. Positioned at the confluence of accounting, auditing and GenAI, the paper introduces a meaningful overview of knowledge in the areas of effective data analysis, interpretation of findings, risk awareness and risk management. It emphasizes and reshapes the role of required skills for accounting and auditing professionals in discovering the true potential of GenAI and adopting it accordingly. The study introduces a new LLM-based system model that can enhance its GenAI capabilities through collaboration with similar systems and provides an explanatory scenario to illustrate its applicability in the accounting and audit area.

Keywords: Generative Artificial Intelligence (GenAI); competencies; accounting and auditing (AA); GenAI risks; Large Language Model (LLM)

1. Introduction

The rapid development and disruptive progress in Artificial Intelligence (AI), which is dramatically transforming many aspects of employees' personal and professional lives, is raising questions in academia and business about the new skills needed by accounting and auditing (AA) professionals [1–3] as they integrate AI solutions into their day-to-day work [4–6]. Recent studies have pointed out that Big4 firms report significantly higher expertise in using AI than other AA companies [7,8], hence the role of these technologies in creating critical competitive advantages in the current context [9]. Statistical figures reinforce this idea, with AI in the accounting services market being estimated at USD 1.56 billion in 2024 and expected to reach USD 6.62 billion by 2029, with a compound annual growth rate (CAGR) of 33.5% for the forecast period 2024–2029 [10].

A new entrant in the AI technologies group, Generative Artificial Intelligence (GenAI), holds significant potential to transform the field of accounting and auditing, with the academic and business communities investigating its applicability in the AA sphere [5]. Accountants and auditors who can combine knowledge, skills and competencies required by the profession with those using GenAI products will benefit from the opportunity to increase the efficiency and accuracy of their work results [11]. Studies on the adoption of

innovative technologies, including AI, in accounting and auditing have focused on topics of interest, such as the impact on the profession [12–14], changes in learning programs and the need for advanced skills of accountants and financial auditors [4,15] or the assessment of AI acceptance in accounting [16]. Researchers have highlighted the correlation between the use of emerging technologies in these areas and the skills development of AA professionals. Considering that GenAI will impact the way humans assimilate and process information and knowledge [17,18], the skills needed by accountants in the coming years should encompass both technical and social abilities [19].

Large consulting firms in the accounting and auditing industry are showing increased interest in integrating AI technologies [7,20], and recent research has pointed to significant progress in their use in most business processes in organizations [21,22]. To our knowledge, there are no studies that have addressed both the leveraging of GenAI in accounting and auditing and the emergence of new competency requirements for AA professionals in relation to integrating GenAI into their activities. In attempting to contribute to filling the existing gaps in the specialised literature, in this work, we aimed to address the following research questions:

RQ1. What knowledge, skills and competencies are needed by accounting and auditing professionals to effectively harness Generative Artificial Intelligence, and what are the risks of using it in this area?

RQ2. How can organisations integrate Generative Artificial Intelligence capabilities into information systems to develop the competencies of accounting and audit professionals?

We consider that GenAI can demonstrate its effectiveness in accounting and auditing when its users have both the competencies to manage it and the advanced analytical skills to interpret the results generated, using critical thinking to filter and evaluate them to make informed decisions. Therefore, this study starts from the premise that the widespread AI solutions and GenAI adoption in the AA domain generate the need to encourage and direct human resources towards improving and enriching skills. This premise aligns with the vision of international bodies that promote close cooperation between governments and stakeholders so that people are equipped with the necessary competencies that enable them to use and interact effectively with AI systems [23].

The main objectives of this study are (1) to identify the competencies, skills and knowledge required by accounting and auditing professionals to operate AI/GenAI systems effectively and (2) to provide a framework that can be used to integrate LLM-based GenAI systems into AA domain.

Considering the revealed aspects, the research is structured as follows: the first section presents the results of the literature review; the second describes the applied research methodology; the third includes the research results, picturing the framework for integrating Generative Artificial Intelligence in the development of accounting and auditing professionals' competencies; and the fourth focuses on discussions and research limitations. Conclusions, contributions, implications and future research directions of the research are drawn at the end of the article.

2. Literature Review

2.1. Generative Artificial Intelligence

Although the term "artificial intelligence" was first introduced in 1956 as part of the Dartmouth Summer Research Project on Artificial Intelligence by John McCarthy, Marvin Minsky, Nathaniel Rochester and Claude Shannon, there is arguably a consensus on the current importance of Artificial Intelligence systems with learning, reasoning and adaptive capabilities [24].

A subset of AI is Generative Artificial Intelligence, defined as a technology that provides human-like responses in the form of text, images and video content using Deep Learning (DL) and Natural Language Processing (NLP) models [25], following the requests formulation [26]. Unlike traditional AI systems, which use predefined datasets and rule-based programming [27], GenAI can generate novel, original content to provide answers

to complex and diverse queries [28]. The launch of OpenAI's ChatGPT chatbot has paved the way for significant progress in AI. The integration of DL and language models based on the Generative Pre-Training Transformer (GPT) architecture significantly expands the capabilities of these programs [29], which replicates the activity of the human brain in the process of learning and developing the responses provided for the requests sent [30].

Large Language Models (LLMs) represent a particular type of GenAI trained on large datasets using advanced DL techniques to analyse and understand patterns and structures, creating the prerequisites for generating similar items [18,31,32]. LLMs serve as a "base model" and provide a starting point for the development of more advanced and complex models [27,33,34]. LLM models are increasingly used in content generation, information retrieval in unstructured data, data conversions, data organisation or detailed data analysis, addressing diverse business needs.

For the use of LLM-based systems, two techniques are known and harnessed in practice: fine-tuning and prompt engineering [35,36]. Fine-tuning is the technique by which a pre-trained model is adapted to a particular domain by training it further on a task-focused dataset. It involves improving the model by learning from the specified dataset. Prompting, on the other hand, is considered a more straightforward technique that does not involve modifying the model but only adjusting the way it is queried to guide it in the right direction [37]. The prompt can be a word, a sentence or phrase, a list of instructions or possible input data that has the effect of generating a relevant response in the context created [38]. Various strategies regarding the prompting technique of LLM-based systems are highlighted in research studies and used in practice: Zero-Shot—prompting a task in a way that LLM can understand and generate an appropriate response without explicit input, Role-Playing—simulating a profession-specific role along with the formulation of the prompt to solve the task for greater contextualization [39] and Chain-of-Thoughts (CoT) prompting—requesting to solve a task accompanied by prompting on providing the reasoning steps [40,41]. Masikisiki et al. [42] illustrated the use of the CoT technique as a method to train LLM-based system users to efficiently train models to obtain more relevant results. From another perspective, CoT can bring the benefit of understanding the reasoning of solving a complex problem by the LLM-based system user, thus contributing to the skills or knowledge development in the domain they are interested in.

Floridi [43] presented the concept of AI2AI, which involves the ability of LLM-based systems to collaborate and form connections with other similar systems, thus enabling interoperability for more relevant responses. On the other hand, LLM-based systems can generate plausible answers that may sometimes be incorrect, a phenomenon known as AI Hallucination, due to the biased (incomplete or incorrect) data sources on which LLMs are modelled/trained. To manage this problem, one solution that has proven viable is to combine the capabilities of LLM-based systems with external sources [44]. LLM-based GenAI systems have revolutionary potential, which can become disruptive through significant impact on society and organisations [32].

2.2. New Competencies Requirements for Accounting Professionals and Auditors in the AI-Integrating Context

2.2.1. Insights from the Academic Literature

The competencies of an organization's staff are crucial, often serving as a key factor for competitive advantage [45]. From a management theory perspective, one of the pioneers of competency research is David McClelland, who described competency as "a symbol for an alternative approach to traditional tests of in-smartness" [46]. A landmark is the definition of competence as a characteristic of a person, which can be a trait, a skill, an aspect of self-image or social role, or a set of knowledge that a person uses [47].

Recently, a government initiative [48] has defined AI Skills for Business, grouped into four categories—cognitive, functional, personal/behavioural and ethical—to support organisations in developing strategies to upskill staff using AI solutions in their daily work. In accounting and auditing, various studies have highlighted the correlation between

professionals' competencies and the extent to which they use innovative technologies in their activities. Andreassen [49] mapped current skills of accountants and identified new ones, such as (i) statistical knowledge for preparing data or interpreting predictive models; (ii) performance analysis and client segmentation; (iii) monitoring IT system integrations and electronic exchange of accounting data between systems. Hence, the key role of AA professionals is to adopt and exploit the benefits of these technologies. Arguing the need to reshape skills of the accounting professionals, [50] discussed courses integrating elements of AI and RPA, taught at a US public university.

As the accountant's tasks and expertise are taken over by technology, the accountant's remit becomes narrower and more specialised, with the evolution of the accounting profession necessitating a scope expansion of technical skills [51]. The emergence of interprofessional competencies has led accountants to move towards other roles close to their identity, which, until recently, did not fit into their specific duties. It is also becoming imperative for auditors to acquire new technical skills for obtaining and interpreting financial audit results using modern technologies [52] and to focus on innovative supporting tools that imply the assumption of a continuous learning process [53]. Mathisen and Nerland [54] confirmed that newly adopted technologies play a crucial role in financial auditing by raising awareness, mobilizing for learning at work, facilitating learning and developing auditors' competencies. Mancini et al. [55] assessed smart technologies and mapped the set of Knowledge, Skills, Abilities (KSAs) required to manage decision-supporting information by leveraging AI, Big Data and Blockchain. Noordin et al. [56] explored external auditors' perception of using Artificial Intelligence (AI) and its contribution to audit quality. The study highlighted the significance of engaging skilled accounting and auditing professionals to incorporate AI into audit tasks, thereby minimizing the related risks. Consequently, auditors should enhance their technical capabilities, irrespective of the type of audit firm they are employed by. Similarly, in a study aimed at assessing individual factors that may contribute to the effectiveness of fraud risk assessment among external auditors, Ridzuan et al. [57] evaluated the crucial role of digital technology skills in enhancing this ability.

Among profession-specific skills in financial auditing, critical thinking is considered essential when using IT technologies. It is important to interpret the results obtained using AI through the lens of professional reasoning. Sceptical thinking and mental representations, which can be fostered by training skills in the sphere of 'problem solving', are also valued [58,59].

Although emerging technologies appear to have improved the efficiency and effectiveness of audit work [60], there is the risk that some professional skills be diminished due to the use of AI technologies in specific activities [61]. Over-reliance on AI could lead auditors to neglect professional judgement, resulting in the risk of missing findings, misdiagnosing and misinterpreting data identified by AI.

Possessing technical skills is a sine qua non in the GenAI adoption and associated risk management [62,63]. Their perishable nature leads organisations to generally prefer human resources that primarily possess soft skills: communication, flexibility, teamwork, personal branding, proactive problem-oriented thinking, motivation and confidence [64,65]. Unlike technical skills, social skills are more closely related to personal characteristics and are more difficult to obtain or develop over time.

2.2.2. Professional Accounting Bodies Perspective

The first professional accounting bodies (PABs) were established in the second half of the 19th century in the United Kingdom and obtained Royal Charter status, which implies they are given the right to regulate their activities and decide the process and criteria for membership [66,67].

Today, most national and international professional accountancy bodies are part of the International Federation of Accountants (IFAC), which comprises more than 180 professional accountancy organizations in more than 135 jurisdictions, representing millions of professional accountants [68]. Professional accounting bodies exert significant influence on

accounting and business activity and play a crucial role in shaping professional direction, being considered the profession's principal guardians [69].

At the same time, tertiary accounting education relies on the accreditation of professional accounting bodies to certify that their graduates have developed the skills, knowledge and competencies needed to enter the accountancy profession [67]. On the other hand, the close and deep relationship of PABs with universities and higher education has some difficulties due to the increasing number of accounting professional interns employed directly by school [66].

For the analysis of the competencies recommended by professional regulatory and standard-setting bodies, three global and three nationally recognised organisations with significant influence on AA and accounting professionals were selected [69]. The Association of Chartered Certified Accountants (ACCA), with over 500,000 members and students; the Chartered Institute of Management Accountants (CIMA), with over 650,000 members and students; and the Institute of Management Accountants (IMA), with over 140,000 members, are international professional bodies that work continuously with industry experts and employers to ensure consistency between their requirements and recommended competencies. The American Institute of Certified Public Accountants (AICPA), established in 1887, having nowadays over 430,000 members; the Institute of Chartered Accountants in England and Wales (ICAEW), established in 1880; and the first PAB, established in 1854, the Institute of Chartered Accountants of Scotland (ICAS), are national professional bodies whose recommendations are based on extensive research, consultation with industry experts and analysis of evolving regulatory and market requirements. CIMA and AICPA have jointly developed the Chartered Global Management Accountant (CGMA) accreditation.

A synoptic table of the main classes of competencies considered by each professional body was drawn up in Table 1.

Table 1. Synopsis of core competencies.

Professional Body	Classes of Competencies
ICAEW (2018)	Ethics and professionalism; communication; teamwork; decision-making; problem-solving; adding value; technical competence [70]
ACCA (2020)	Ethics and professionalism; data, digital and technology; strategy and innovation; leadership and management; stakeholder relationship management; governance, risk and control; corporate and business reporting; financial management; management accounting; taxation; audit and assurance; advisory and consultancy [71]
AICPA & CIMA (2022)	Technical skills; business skills; people skills; leadership skills; digital skills [72]
ICAS (2023)	Ethics and integrity; communications; teamwork and leadership; personal effectiveness; problem-solving and decision-making; technical competence [73]
IMA (2023)	Strategy, planning and performance; reporting and control; business acumen and operations; technology and analytics; leadership; professional ethics and values [74]

Following the analysis of the competency's classes recorded in the consulted documents, it was observed that competencies from the field of integrated technologies were explicitly included in the AA area: "Technical competence"—ICAEW, AICPA and ICAS, "Data, Digital and Technology"—ACCA, "Digital skills"—AICPA, "Technology and Analytics"—IMA. Moreover, as the year of document publication is more recent, two classes of skills in the digital area in the same framework have even been identified (AICPA, 2022: "Technical skills" and "Digital skills") [72]. All these align with the academic literature review insights. Reviewing the considerations on newly emerging requirements for accountants' and auditors' competencies generated by the integration of AI/GenAI/LLM solutions, it was found that there are no clear formulations regarding them in the specialized literature. Therefore, the positioning of the present research in this area aims to fill existing gaps, enriching the body of knowledge in the investigated field and provid-

ing benchmarks for addressing and shaping the development and learning plans of AA professionals at the organisational level.

2.3. Risks of Using AI in the Accounting and Auditing Professions

With the relatively recent emergence of GenAI technologies, its use in accounting and auditing exposes organisations [75] to specific risks.

The risk of maladjustment revealed in the literature arises because of accountants' and auditors' non-compliance with the requirements of using new IT [76,77]. Burns and Igou [76] noted that, under AI exploitation, users are often forced to adapt to intrusive interfaces, which can be perceived as a threat to privacy by violating certain social norms or boundaries. In these situations, flexibility, proactive thinking and adaptability are essential.

Tasks and competencies for accounting professions will undertake major changes, keeping "core" roles and tasks continuing to exist in the future, with the difference that they will be executed by AI technology [63,78]. There will also be new roles where collaboration with AI solutions will be necessary. Korzynsky et al. [18] consider AI Prompt Engineering as one of the future mandatory digital competencies when using GenAI. There is a major risk that poor prompt choice and formulation will render a GenAI-based system ineffective, which could influence the relevance and consistency of the results provided by AI models.

A significant risk for AI applications in general, and GenAI in particular, is algorithmic bias. Machine Learning algorithms identify patterns in the available data on which human decision-makers base their choices. If the pattern emerges due to a bias, the algorithms will amplify it and alter the results, exacerbating patterns of discrimination and the risk of ethical violations. In practice, the main instances of discrimination attributed to AI are caused by the following technical biases [79,80]: (i) bias generated by the distribution of the data used in model training, which does not respect the actual distribution of the data; (ii) sampling bias—resulting from the erroneous sizing of the sample used in model training; (iii) labelling bias—generated by the lack of diversity in the group of people labelling the training sample, with the data being altered by the biases of the labeller; (iv) proxy bias—generated by the inclusion of variables that are not directly related to the category being learned; (v) gender bias—induced by an algorithm that leads to gender discrimination; (vi) race bias—generated by an algorithm that leads to race discrimination. Interaction with the user can generate, in some situations, automation bias or confirmation bias, i.e., the tendency to more readily validate the results provided by an AI system under unconsciously developed automatisms [79]. Kirk et al. [81] showed that LLM can produce unethical, racist and sexist comments.

Another risk revealed in the literature [82,83] is that of the atrophy of people's skills—deskilling—who make decisions based more on the results provided by AI applications and less on their professional reasoning. Risks considered major are those associated with the security of accounting information [84], requiring accountants and auditors to be aware of and apply measures to ensure it. Understanding the mechanisms for encrypting data, anonymising sensitive and confidential information, securing networks and identifying potential threats and vulnerabilities is essential in maintaining data security. The authors of [85] also reported the potential of AI-enabled data analytic-driven audit tools to alter audit processes and raise concerns about their uncontrolled use by novice-level auditors. The adoption of AI systems may also involve legal risks to data privacy, requiring verification of compliance with the European Parliament's General Data Protection Regulation (GDPR), laws addressing discrimination (e.g., the US Civil Rights Act of 1964), any clauses in customer contracts, intellectual property regulations and personal information protection legislation adopted in some countries [86].

Following the scientific investigation, it appears that the relationship between the risks associated with the use of GenAI and the competencies of AA professionals may represent a future research direction, with the topic not being explored in the literature probably because of its novelty. The results confirm the necessity and usefulness of including, in the

framework, elements that support organisations in their efforts to increase AA professionals' risk awareness, which can create the conditions for the development of specific KSAs.

3. Materials and Methods

The methodological approach was based on the Systematic Literature Review (SLR), which is considered a rigorous method for identifying, evaluating and synthesizing the existing body of knowledge in published materials by researchers and practitioners [87]. Embedding ideas advanced by Massaro [88] regarding the conduct of literature research in accounting into the three-stage model adapted from [87] investigating AI in the Industry 4.0 era, the study began with developing a general plan to guide the research, as shown in Figure 1.

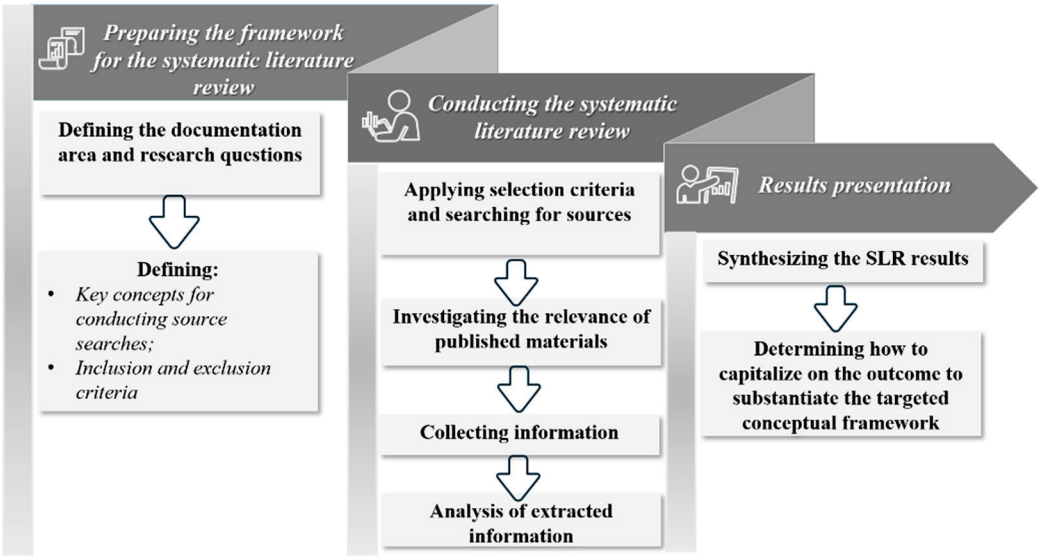


Figure 1. Overview of research methodology.

The area of research includes topics such as AI, GenAI, LLM and the required competencies in accounting and auditing. The next phase involved defining a twofold purpose of the study, aligned with the research questions to facilitate the orientation of the sources selection, the research running and the achievement of the study results. To answer the question "RQ1. What knowledge, skills and competencies are needed by accounting and auditing professionals to effectively leverage Generative Artificial Intelligence and what are the risks of using it in this field?", it was decided to conduct a search of the most recent papers in the Web of Science Core Collection, combined with an investigation of the business literature—an adapted PRISMA flow diagram (Figure 2).

The key concepts used in the design of the logical expression for locating relevant references in the emerging areas, executed at the level of titles, keywords and abstracts (topic) were determined: (("artificial intelligence" or "machine learning" or "deep learning" or AI or NLP or LLM or GPT or RPA) and ((competence*) or (skill*) or (knowledge*)) and (accountant or accountants or auditor or auditors))).

To improve the validity, reliability and rigor of the study contributions and conclusions, we also analysed documents produced by three global organisms and three national bodies, representative for the development of accounting and auditing professional standards and certifications: ACCA, CIMA, and IMA, respectively, and AICPA, ICAEW and ICAS (Table 1).

In the next step, the eligibility conditions—explicit inclusion and exclusion criteria—of the sources identified in the search were defined to create the prerequisites for an objective selection and a meaningful assessment of the research materials' relevance (Table 2). These criteria were derived from the study objectives and guided by the research questions.

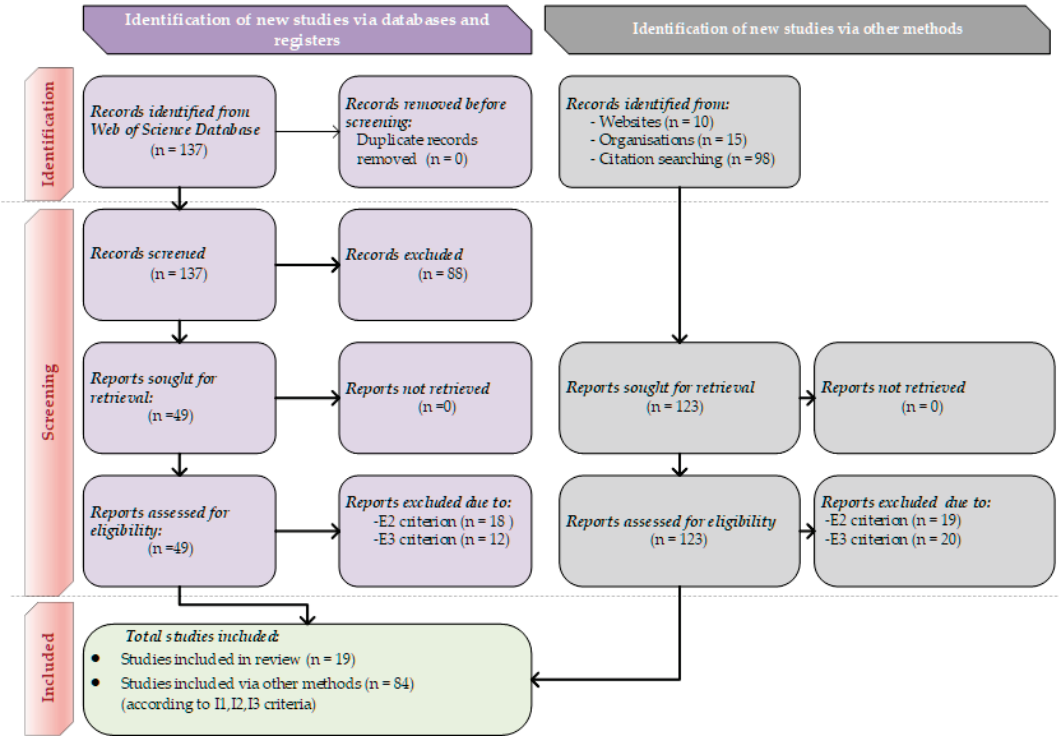


Figure 2. PRISMA flow diagram. Adapted from [89].

Table 2. Inclusion and exclusion criteria.

Criteria Type	Reason for Inclusion/Exclusion
Inclusion Criteria	(I1) The paper focuses on the need for the AA professional's competencies in AI/GenAI area (expected or required by the market). (I2) The paper focuses on the risks regarding AI/GenAI inclusion in AA domain. (I3) The paper focuses on GenAI/LLM in AA domain. (I4) The paper is a journal article, conference article or book chapter. (I5) The paper is published in English. (I6) The paper is published between 2018 and 2024.
Exclusion Criteria	(E1) No full text available. (E2) The paper does not address/focus on AA professional's competencies in relation to adopted AI technologies. (E3) The paper is only loosely related to AI/GenAI competencies for AA domain.

Applying the constraints resulted in obtaining an initial sample of 137 WoS-indexed articles, as can be seen in Table 3, the number of papers has increased significantly since 2019 due to both the novelty and rapid expansion of GenAI throughout the business world.

Table 3. Articles WoS by publication year.

Year	2018	2019	2020	2021	2022	2023	2024	Total
Papers Number	4	5	14	26	29	45	14	137

Each research team member checked the credibility and relevance of the identified references. Articles that did not address professional accounting and auditing skills concerning adopted AI technologies were eliminated. The final sample for our SLR includes 103 studies, a number considered relevant for an SLR approach [90].

Based on the SLR results, the second research question—"RQ2. How can organisations integrate Generative Artificial Intelligence capabilities into information systems to

develop the skills of accounting and auditing professionals?"—redirected the scientific investigation to detect the consequences of the GenAI systems adoption on the competencies' development of accounting and auditing professionals and their assistance needs in current activities and decision-making processes. This effort finally resulted in developing a framework for integrating a GenAI-based system into the organisational environment. The sources considered relevant for the topic, and the main results extracted and harnessed in the construction of the proposed model were presented in Tables 4–6, as in [91,92].

Table 4. Articles selected from the WoS database, included in the final sample.

Reference	Article Title	Publication	Main Findings
Samiolo, et al., 2024 [83]	Auditor Judgment in the Fourth Industrial Revolution	Contemporary Accounting Research	Skills-related risks
Norzelan, et al., 2024 [63]	Technology Acceptance of Artificial Intelligence (AI) among Heads of Finance and Accounting Units in the Shared Service Industry	Technological Forecasting and Social Change	AI acceptance in finance and accounting; risk assessment
Arnold, et al., 2023 [82]	Can Knowledge Based Systems Be Designed to Counteract Deskilling Effects?	International Journal of Accounting Information Systems	Risks of deskilling; knowledge-based systems implications in supporting accounting professionals
Rawashdeh, 2024 [12]	A Deep Learning-Based SEM-ANN Analysis of the Impact of AI-Based Audit Services on Client Trust	Journal of Applied Accounting Research	AI-based audit services, perceived quality, value, attitude, satisfaction and trust
Munoko, et al., 2020 [7]	The Ethical Implications of Using Artificial Intelligence in Auditing	Journal of Business Ethics	Accounting and auditing industry interest in AI adoption
Thottoli, 2024 [75]	Leveraging information communication technology (ICT) and artificial intelligence (AI) to enhance auditing practices	Accounting Research Journal	Potential benefits and risks associated with AI and information communication technology (ICT) adoption in auditing
Jemine, et al., 2024 [60]	Technological innovation and the co-production of accounting services in small accounting firms	Accounting Auditing & Accountability Journal	Impact of emerging information technologies onto small accounting firms and professionals' competencies
Grosu, et al., 2023 [3]	Testing accountants' perceptions of the digitization of the profession and profiling the future professional	Technological Forecasting and Social Change	Challenges, knowledge and skills required to face technological evolution
Rodgers, et al., 2023 [20]	Protocol Analysis Data Collection Technique Implemented for Artificial Intelligence Design	IEEE Transactions on Engineering Management	Impact of the lack of an AI framework, IFRS knowledge, and legislation conflict on AA standards implementation; AI-based support for protocol analysis, benefits and designed features
Koreff, et al., 2023 [85]	Exploring the Impact of Technology Dominance on Audit Professionalism through Data Analytic-Driven Healthcare Audits	Journal of Information Systems	Potential of AI-enabled data analytic-driven audits tools to alter audit missions and derived concerns for their uncontrolled use by novice-level auditors
Ng, 2023 [50]	Teaching Advanced Data Analytics, Robotic Process Automation, and Artificial Intelligence in a Graduate Accounting Program	Journal of Emerging Technologies in Accounting	Reshaping skill sets needed in the accounting profession by designing courses that incorporate RPA and AI at a public university (USA)
Kommunuri, 2022 [8]	Artificial Intelligence and the Changing Landscape of Accounting: A Viewpoint	Pacific Accounting Review	The impact of AI and ML on the accounting skills environment

Table 4. *Cont.*

Reference	Article Title	Publication	Main Findings
Andiola, et al., 2022 [59]	Wealthy Watches Inc.: The Substantive Testing of Accounts Receivable in the Evolving Audit Environment	Issues in Accounting Education	Awareness of technologies used in audit practice; the case of students practicing scepticism and applying professional judgment using AI and RPA tools
Friedrich, et al., 2022 [6]	Epistemological Thinking about Accounting in the Era of Artificial Intelligence	Revista Gestao Organizacional	Links between accounting science and disruptive technologies; transformation of accounting science on report creation, interpretation and authentication
Fotoh and Lorentzon, 2021 [9]	The Impact of Digitalization on Future Audits	Journal of Emerging Technologies in Accounting	Competitivity framework for audit profession; need for new capabilities, skills and business models incorporating digital technologies
Mancini, et al., 2021 [55]	Four Research Pathways for Understanding the Role of Smart Technologies in Accounting	Meditari Accountancy Research	Mapping KSAs required to manage decision-supporting information by leveraging AI and other new technologies.
Leitner-Hanetseder, et al., 2021 [78]	Profession in Transition: Actors, Tasks and Roles in AI-Based Accounting	Journal of Applied Accounting Research	Changes for competencies for accounting professions and "core" roles; AI-based accounting context (mixed AI and human accounting teams).
Kirk, et al., 2021 [81]	Bias Out-of-the-Box: An Empirical Analysis of Intersectional Occupational Biases in Popular Generative Language Models	Advances in neural information processing systems	Risks of producing unethical, racist and sexist comments
Andreassen, 2020 [49]	Digital Technology and Changing Roles: A Management Accountant's Dream or Nightmare?	Journal of Management Control	Accountant new skills
Li and Liu, 2020 [25]	Development of an Intelligent NLP-Based Audit Plan Knowledge Discovery System	Journal of Emerging Technologies in Accounting	Generative Artificial Intelligence overview
Plumlee, et al., 2015 [58]	Training Auditors to Perform Analytical Procedures Using Metacognitive Skills	The Accounting Review	New skills required for auditors

Table 5. Selection of articles identified via other methods and some findings relevant to the proposed framework.

Reference	Article Title	Publication	Main Findings
Wölfel, et al., 2024 [93]	Knowledge-Based and Generative-AI-Driven Pedagogical Conversational Agents: A Comparative Study of Grice's Cooperative Principles and Trust	Big Data and Cognitive Computing	Generative Language Models (GLMs), PEdagogical conversational Tutor with generative AI component and adaptation
Dong, et al., 2024 [94]	An Automated Multi-Phase Facilitation Agent Based on LLM	IEICE Transactions on Information and Systems	LLM-based agent implementation; large-scale discussion support systems
Trad and Chehab, 2024 [36]	Prompt Engineering or Fine-Tuning? A Case Study on Phishing Detection with Large Language Models	Machine Learning and Knowledge Extraction	Prompt Engineering process; tailoring LLM to particular tasks

Table 5. *Cont.*

Reference	Article Title	Publication	Main Findings
White, et al., 2023 [37]	A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT	ArXiv	ChatGPT; Prompt Engineering
Floridi, 2023 [43]	AI as Agency Without Intelligence: On ChatGPT, Large Language Models, and Other Generative Models.	Philosophy & Technology	AI systems, AI2AI interoperability; "Confederated AI"
Piktus, 2023 [44]	Online Tools Help Large Language Models to Solve Problems through Reasoning	Nature	LLM-based systems and external data sources; how to improve LLMs' output
Wei, et al., 2022 [40]	Chain-of-Thought Prompting Elicits Reasoning in Large Language Models	International Review of Economics & Finance	Chain-of-Thoughts (CoT) prompting-requesting and providing the reasoning steps.

Table 6. Reports published by professional bodies, other research institutes and companies.

Reference	Article Title	Organisation Name	Main Findings
IMA, 2023 [74]	IMA Management Accounting Competency Framework		Core competencies considered by each professional body; analysing the gap associated with newly emerging requirements for accountants' and auditors' competencies generated by the integration of AI/GenAI/LLM solutions
ICAS, 2023 [73]	ICAS Mapping Our New Competencies		
AICPA & CIMA, 2022 [72]	CGMA Competency Framework		
ACCA, 2020 [71]	ACCA Competency Framework		
ICAEW, 2018 [70]	ACA Qualification Professional Development Ladders		
Mordor Intelligence, 2024 [10]	AI In Accounting Market Size & Share Analysis—Growth Trends & Forecasts (2024–2029)	Mordor Intelligence	Analysis of the AI in the accounting services market
McKinsey, 2023 [95]	The economic potential of generative AI: The next productivity frontier	McKinsey	The impact of GenAI on productivity
Alan Turing Institute, 2023 [48]	AI Skills for Business Competency Framework	Alan Turing Institute	Developing strategies to upskill staff using AI solutions
Schwartz, et al., 2022 [80]	Towards a Standard for Identifying and Managing Bias in Artificial Intelligence	National Institute of Standards and Technology	Technical biases and discrimination instances attributed to AI
OECD, 2019 [23]	Principles for trustworthy AI	OECD	Cooperation governments-stakeholders so that people hold the necessary competencies to interact effectively with AI tools

The tables presenting the main sources and findings selected for the study were organised by the search method used and arranged from the most recent to the oldest. Table 4 is more focused on our topics of AI/Gen AI adoption in the AA field, the need for AA professionals for new skills and competencies, the associated risks and the deduced research gaps.

Table 5 highlights a selection of the sources providing more technical findings related to LLM-based GenAI tools.

Table 6 lists the publications issued by international and national professional standard-setting bodies. These publications were reviewed to identify the core competencies and gaps resulting from the integration of emerging technologies in the AA domain. Five additional

reports were included to demonstrate findings related to the AI-enhanced market for AA services, its impact on productivity and AI-related strategies for upskilling staff.

4. Results

To answer the first research question, we identified in the academic literature a set of core competencies for using GenAI in accounting and auditing, and we structured them into two categories:

1. Cross-disciplinary skills:
   - digital skills (data analytics skills, skills in extracting, managing and centralising the datasets provided to the system) [49,52,55];
   - prompt engineering skills (train the AI according to specific user needs) [37,78];
   - soft skills (communication and motivational capabilities, a holistic understanding of processes, project management skills [78], sceptical thinking and problem-solving) [58,59].
2. Profession-specific competencies:
   - transformation of accounting science on report creation, interpretation and authentication [6];
   - analysing historical internal structured data, collecting and selecting the data used for valuation [78];
   - forecasting and making audit decisions [25];
   - risk identification, risk assessment and risk mitigation [78];
   - the preparation of reporting elements; good accounting and business ethics knowledge; understanding of how the AI makes its decisions; supporting the management team in strategic planning and investment decisions [78].

In line with the trend of new disruptive technologies, the scientific literature highlights the growing need for advanced technical skills in the accounting and auditing profession, as opposed to the competencies frameworks established by professional accounting bodies (PABs). Looking ahead, it seems necessary for the PABs to update the competence of their underlying frameworks.

Analysing the context of using GenAI in the accounting and audit field, the main risks identified in the reviewed literature are the following:

- deskilling risks and their effects [82,83];
- model and bias risk [80];
- risks associated with the security of accounting information [84];
- risks related to AI-automated accounting functions [63];
- unethical, racist and sexist content generation by LLMs [81].

Considering the second research question, corroborating the literature review results, a framework was developed for integrating LLM-based GenAI systems with the IT systems of organisations providing accounting and auditing services. This approach aims to increase organisational performance by (1) automating repetitive tasks; (2) identifying anomalies, inconsistencies and potential risks in accounting records; (3) streamlining decision-making by providing data, information and predictive analyses; (4) continuously monitoring and complying with current accounting and auditing regulations; (5) developing organisational memory and improving organisational learning.

Although there are papers that have discussed the design of effective generative AI systems, such as [93], which proposed a GenAI conversational system model for pedagogical purposes, or [94], focusing on designing LLM-based agents for facilitating discussions, we have not encountered specific studies proposing a framework for effectively integrating generative artificial intelligence in accounting and auditing.

4.1. Framework for Integrating LLM-Based Internal GenAI System

The proposed model builds upon research conducted by [27,33,96–98], in the field of integrating GenAI in various domains. This research is supported by the engineered

prompting and fine-tuning techniques analysed and described by [35–38]. It also incorporates Floridi's research [43] on AI-type inter-system communication (AI2AI) and Piktus's research [44] on combining the capabilities of LLM-based systems with external data sources. These contributions helped define some of the framework's main components.

The core element of our framework for integrating GenAI into the accounting and auditing field, the Knowledge Consolidator, plays a pivotal role in an LLM-based system. This component is an original contribution designed to continuously consult both external (websites, public or private databases, etc.) and internal (databases, data repositories, organisational memory, etc.) data sources. The need to integrate the Knowledge Consolidator component in the proposed framework is supported by prior research [99–101], which investigated the relationship between knowledge generation and AI-based models. Thus, AI can generate knowledge that can improve the reasoning of generative models and their compositional capabilities. In the framework, Knowledge Consolidator is designed to continuously extract and integrate information or knowledge into the System Data Source component, ensuring its relevance and up to date.

Consequently, the LLM-based internal GenAI system has three main components: the Large Language Model (LLM), the System Data Source and the Knowledge Consolidator (Figure 3).

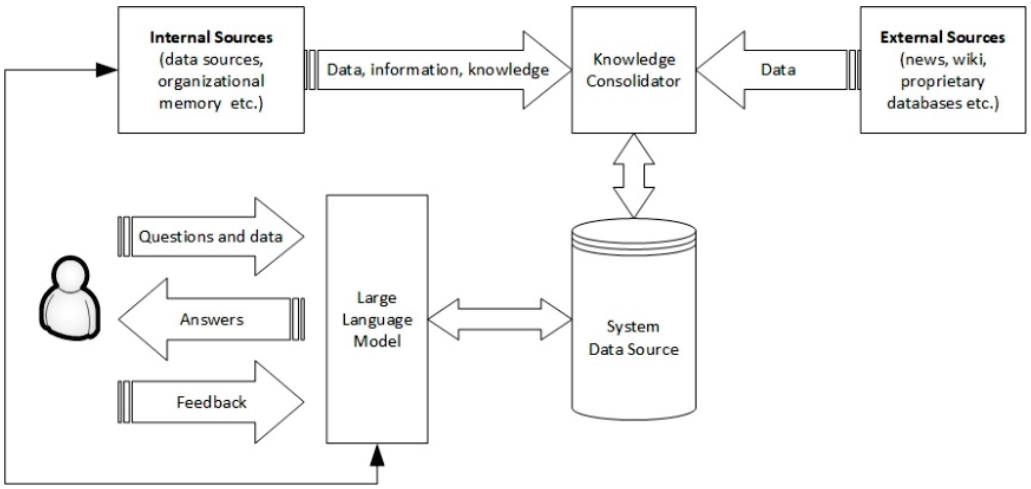


Figure 3. Framework for integrating an LLM-based internal GenAI system.

Following user requests, the System Data Source is queried by the LLM, providing answers or executing requested actions.

The way the user interacts with the LLM-based internal GenAI system is as follows:

1. The user addresses a question or requests an action to be performed and, if necessary, provides additional data to refine the context.
2. The system processes the text from the request and analyses any data provided to identify what the user wants or needs to do.
3. If applicable, the system queries internal sources based on the requirements/requests and data provided by the user.
4. Based on the processed text and data provided by the user or information obtained from internal sources using System Data Source, the system generates and displays a response or performs the requested action.
5. The user provides feedback on the system's response or action, which is used to learn and improve the internal GenAI system's performance over time by adjusting the processes of understanding, generating responses or taking the required action.

4.2. Extended LLM-Based Internal GenAI System

The framework described (Figure 3) could be further enhanced, resulting in an Extended LLM-based internal GenAI system that includes four main components: Filtering

(Security) Layer, Large Language Model (LLM), System Data Source and Knowledge Consolidator (Figure 4). The Knowledge Consolidator component behaves the same way as the original internal LLM-based GenAI system.

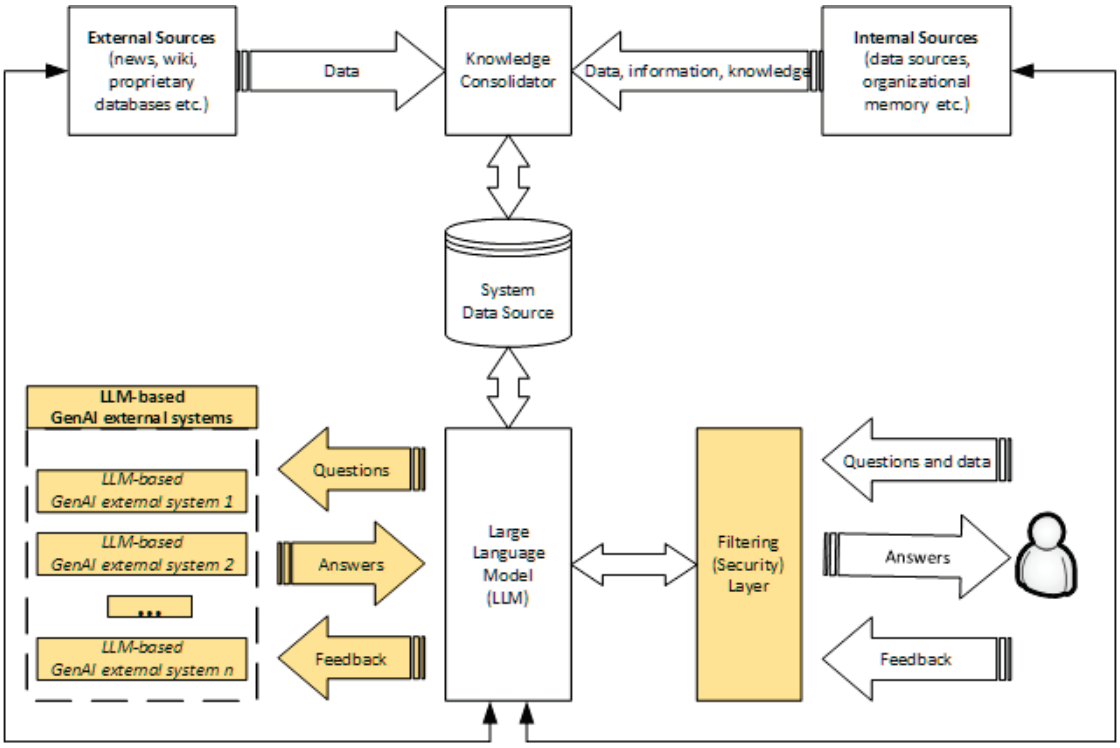


Figure 4. Framework on the integration of an extended LLM-based internal GenAI system.

The Extended LLM-based internal GenAI system has two major improvements:

1. The Filtering (Security) Layer, whose main role is to ensure the security and confidentiality of internal data and not to allow data and information to be transmitted externally by filtering queries and data transmitted outside the organisation. Considering that a significant part of the data managed by accountants and auditors is sensitive or confidential, it is mandatory to implement a filtering layer that prevents intentional or accidental transmission outside the internal system.
2. Continuous interaction with other LLM-based GenAI systems to improve the quality of the provided answers. This interaction enriches and extends the System Data Source by incorporating data from these systems.

Following user requests, the LLM queries the system data source and other LLM-based GenAI systems to provide answers or execute requested actions.

The way the system interacts with the user is as follows:

1. The user addresses a question or requests an action to be performed, possibly providing data to the system.
2. The system processes the user request and any related data.
3. The Filtering (Security) Layer performs filtering of it and of data provided by the user, sending to the LLM component two requests:
   (i) a prompt containing the request and the data submitted by the user that will be used to consult the system's data sources as well as internal and external sources;
   (ii) a prompt in which any confidential elements and data provided by the user have been removed from the request.
4. To understand the context of the user's request, the extended LLM-based internal GenAI system analyses related data provided by several advanced techniques.

5. If necessary, the system formulates a request to internal or external sources based on the requirements and data provided by the user.
6. The system consults in parallel:
   (i) System Data Source based on processed text and user-supplied data.
   (ii) Internal or external sources.
   (iii) Other external LLM-based GenAI systems based on text processed and filtered by the Filtering (Security) Layer component. These systems can be public or non-public. Non-public systems can be accessed based on an inter-organisational cooperation agreement and represent a component of the learning process through exchanging knowledge, experience and resources.
7. The system generates and displays a response or performs the requested action.
8. The user provides feedback on the response received from the system or the undertaken action, which is used for learning and improving the performance of the internal system over time by adjusting the processes of understanding, generating responses or taking the required action. Depending on the organisation's policy, the system may also provide feedback to the consulted external LLM-based GenAI systems.

Figure 5 shows the process of using (modus operandi) the LLM-based internal GenAI system in terms of the competencies leveraged and knowledge gained. The response returned by the system will be subject to a content check and analysed by the person using it. Professional judgement will be the key competence to detect certain anomalies or vulnerabilities in the output. At this stage, the same set of profession-specific competencies will be called upon [70–74], coupled with others which fall within the sphere of personal cognitive skills, such as critical thinking [58]. Also, in the category of cognitive competencies, the ability to recognise and be aware of the risks associated with the results obtained [79,84,86] will enable the accountant or auditor to qualify them as acceptable or inappropriate. The use of Chain-of-Thoughts and Role-Playing techniques [40,41] in the prompting stage can have positive effects in increasing the relevance of GenAI system responses to user expectations.

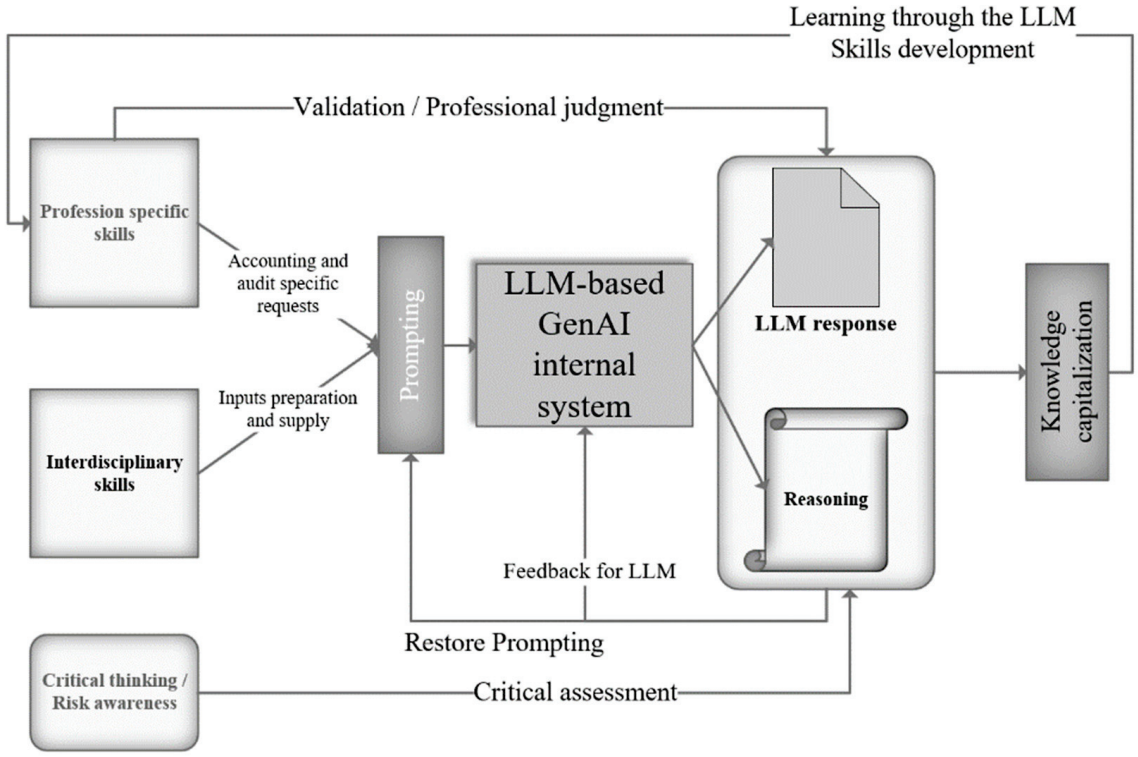


Figure 5. The correlation between competencies and results for using LLM-based internal GenAI system in accounting and auditing.

The validation stage of the outputs provided by the system can result in either accepting the answer or refining the prompt and restarting the process [37]. In the latter case, the LLM will receive feedback concerning why the response cannot be considered acceptable. The new knowledge generated by the LLM-based system will be capitalised and can form an important basis for developing employees' professional skills and competencies.

5. Discussion

Although GenAI is attracting interest from researchers and practitioners because of its vast potential to revolutionise lifelong learning and competencies development, only a few studies are addressing how to integrate GenAI into accounting and auditing. Integrating LLM-based GenAI into organisational information systems is a novel element that contributes to the accounting and auditing research field. This framework enables human-machine collaborations, focusing on AA professionals' competencies, upskilling and reskilling and fostering a culture of continuous learning.

5.1. An Explanatory Scenario

To help readers better understand the proposed framework outlined above, we will illustrate its underlying functional logic with a descriptive example of a hypothetical financial audit mission.

The Audit Services is a company that provides financial audit services. During the audit mission, daily, each team member enters relevant data and information into the internal system. After the completion of the mission, a meeting is scheduled to discuss the main issues encountered and how they were solved. The relevant data, information and knowledge are recorded in the internal system. On the other hand, the company has a tool (Knowledge Consolidator) that continuously performs three main tasks: (1) collects data from the internet (online databases, news sites, competitors' sites, social media and other relevant portals); (2) extracts and transforms data, information and knowledge from the internal system; and (3) feeds data sources into the system based on the results of (1) and (2). At the same time, the company has some agreements with other companies in different industries to interconnect their LLM-based GenAI external systems to enhance the capabilities of the internal GenAI system.

A new audit mission has been started for Best Transport Services that provides domestic and international transportation services by trucks. Helen (one of the team leaders of the audit mission) will formulate a few questions to addressed in the GenAI system. The first question may be, "What are the sensitive issues in auditing a company from the transportation area?". The system will query the system's data source and communicate with the other LLM-based GenAI external systems to provide the best answers to the question. Based on the answer received, Helen will further address two questions: "Has the Best Transport Services been involved in major road events in the last year?", "How many transport vehicles have been insured in the last year?". Assuming the answers were "Best Transport Services has been involved in two major events, both abroad" and "Best Transport Services has insured 59 trucks", Helen can require more data and documents related to these events, and, on the other hand, she can ask for more data on the company's transport vehicle insurance process, insurance policies and payments to insurance companies.

In the pre-audit stage, Helen receives from the transport company the set of financial documents for the audited period in electronic format (many of them in datasheet format). Since the quality of the training data heavily influences the outputs, she organises them into datasets with financial significance that she will use in the LLM-based GenAI system training process. Helen will ask the LLM-based GenAI system what correlations can be identified between the analysed incidents and the company's operational activity regarding costs and possible reductions in orders from clients. By analysing the reasoning used and provided by the GenAI system based on the LLM, Helen will remove the inconclusive results by leveraging her expertise in the audit field. Helen will use fine-tuning techniques to address in-depth questions about specific aspects. This will involve highlighting various

data sources (internal and/or external) such as insurance prices, the bonus-malus system of partner insurance companies, the history of previous incidents or similar situations at companies in the same field of activity. The system will be able to identify new correlations and provide conclusions in the analysed case. At the same time, Helen will enter the final audit report into the system along with relevant feedback regarding the information received from the GenAI system. Building this knowledge base will allow the system to handle future queries on similar matters more effectively.

Figure 6 illustrates the described scenario using a Business Process Modeling and Notation (BPMN) diagram to help understand how the LLM-based GenAI integrates in the hypothetical financial audit mission.

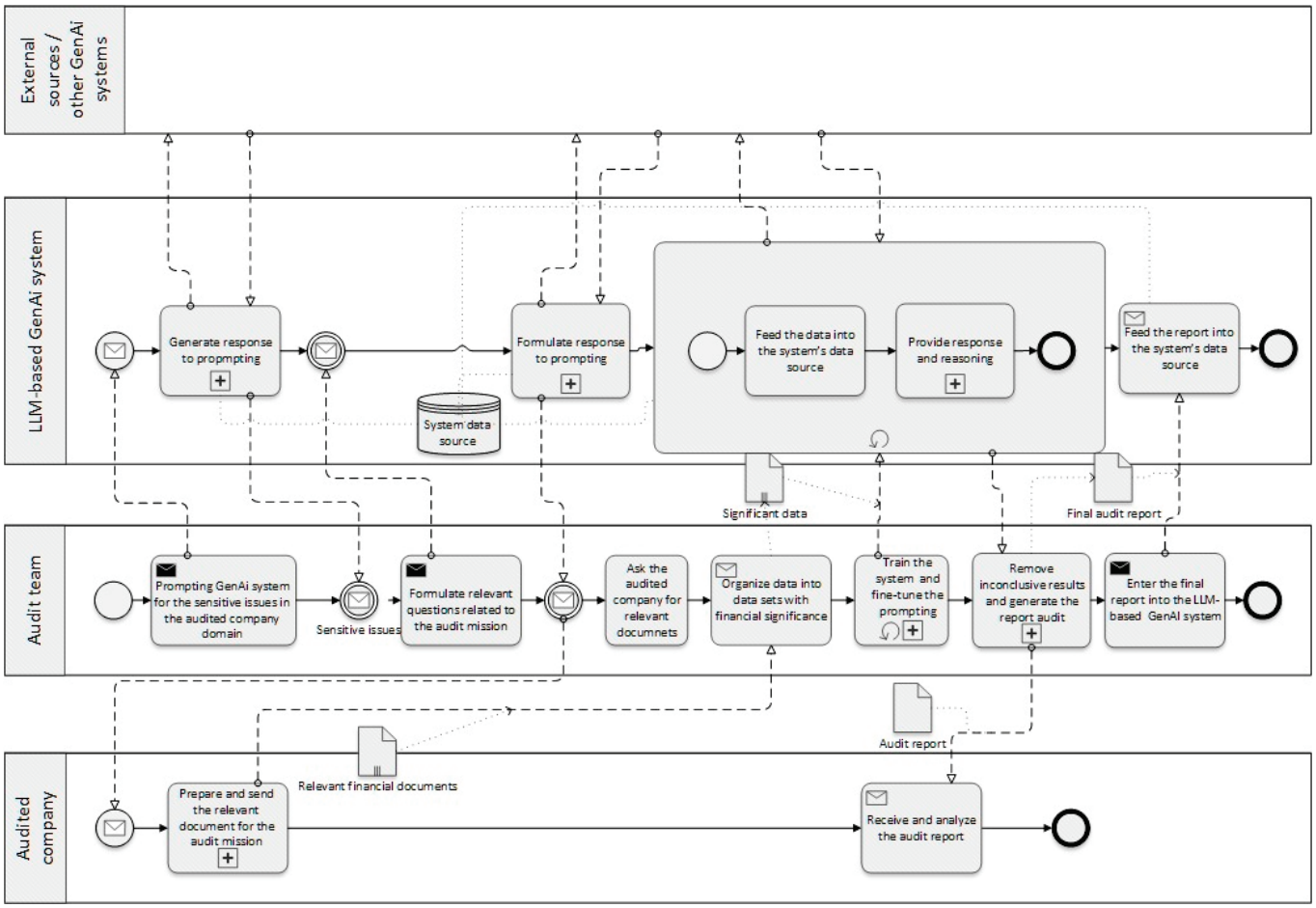


Figure 6. BPMN diagram for the explanatory scenario.

5.2. Benefits of Integrating LLM-Based GenAI Systems in Developing Competencies for Accounting and Audit Professionals

According to McKinsey (2023) [95], the impact of GenAI on productivity has the potential to generate the equivalent from USD 2.6 trillion to USD 4.4 trillion annually and increase the impact of artificial intelligence by 15–40 percent.

Leveraging the proposed framework, it is possible to build an internal GenAI system based on LLM, which fulfils a dual role: (1) redefining how AA professionals can acquire new knowledge and develop necessary skills and (2) assisting them in their current professional activities and during decision-making processes.

5.2.1. Redefining How AA Professionals Can Acquire New Knowledge and Develop Necessary Skills

GenAI has the potential to determine the increase in labour productivity [95], which is significantly influenced by employees' learning and development [102]. Among the

benefits offered by the internal GenAI system in the process of learning and development of employees' skills are:

1. Providing personalised learning pathways following the analysis of each employee's KSA, learning preferences and career goals [103–105].
2. Preparation of learning materials by reviewing a significant volume of internal and external resources and selecting those relevant to each employee [106].
3. Adaptive learning, by dynamically adjusting the difficulty level and pace of presentation of learning materials according to the learner's progress and performance [107], which can lead to a stimulation of the creativity of each employee [104].
4. Assessing employees' skills and knowledge by various means, providing timely and actionable feedback on performance [104,108]. Because feedback supports learning achievement [104], it is a huge benefit for employees to receive valuable and timely feedback.
5. Virtual coaching and mentoring throughout the learning process (interactive conversations and simulations that allow employees to practice and hone their skills) [109,110]. In 2012, the McKinsey Global Institute (MGI) estimated that knowledge workers spent about 20% of their time to search and to gather information [95]. GenAI's support can drastically reduce this time, followed by a significant increase in employee efficiency and effectiveness.
6. Continuous learning and knowledge capitalisation by facilitating the exchange of information, best practices and lessons learned, enriching organisational memory [111,112].

On the other hand, there are concerns regarding the propagation of misinformation by GenAI systems [105,113], continuous monitoring of the answers provided by the GenAI system and the feedback provided by the employees is mandatory.

5.2.2. Assisting Current Activities and Decision-Making Processes

The benefits of targeted GenAI systems converge towards:

(i). Information retrieval and synthesis based on employee's expertise and requirements [110].
(ii). Assistance during the decision-making process [114], while, at the same time, allowing for transparency (GenAI system's decision-making process is visible to employees) and explainability (GenAI system's decisions are easy to understand by employees) [115].
(iii). Collaboration and communication across the organisation, fostering a collaborative environment [116].
(iv). Automation and support for routine/repetitive tasks, assistance, recommendations and suggestions [111].

This study has several limitations worth noting. Due to the paper topic's novelty, the sample of academic articles obtained from the search in the WoS database was not as extensive as expected. In addition, searching the literature sources, choosing terms in the search string and using exclusion criteria were based on subjective assessments and carry the risk of not identifying all relevant publications. Future research should consider increasing the sample size for more reliable and generalisable results.

Moreover, the reviewed papers on the GenAI topic focus mainly on the technical aspects of this disruptive technology. While this is valuable for understanding the fundamental principles of LLM-based system components, a potential limitation is the absence of information regarding their real-world implementation and outcomes.

6. Conclusions

The labour market is continuously adapting to changes in the business environment, looking for professionals with skills appropriate to the diversified contexts generated by AI technologies in organisations. The literature review analysis confirms a demand for new skills in the AA labour market. Research on the competencies needed by accounting and

audit (AA) professionals has examined academic and technological perspectives, focusing on the skills required for graduates and the impact of AI/GenAI technologies on the labour market [77].

The integrative approach of GenAI technology in accounting and auditing represents an original contribution to the body of knowledge with theoretical and practical implications. Based on the proposed framework, an internal LLM-based GenAI system can, in our opinion, support AA professionals by reshaping the way they acquire knowledge and skills and assisting them in their professional activities and decision-making processes. It can improve analytical understanding and awareness of how GenAI can work with the accounting and auditing profession, representing, in our view, a starting point in integrating this innovative technology into operational and decision-support activities.

On the theoretical side, the study provides insights into the requirements for new skills accountants and auditors need to leverage GenAI technologies in their professional work. Equally, the elements of the framework for integrating GenAI can be adapted to the specific contexts of related (tax, valuation, business analysis) or different areas. The paper also brings to the fore the benefits of lifelong learning, which converge towards developing competencies by capitalising on the knowledge resulting from the use of GenAI at the organisational level. The practical implications of the research are positioned in the accounting–auditing–GenAI confluence area. The proposed framework offers insight into the complexity and nature of GenAI technology, as well as the skills required to manage it.

Answering the first research question, the paper reveals the necessity for professional training of staff to develop competencies aimed at (i) elaborating effective prompts for information, (ii) interpreting answers, (iii) awareness and correct treatment of the risks involved and (iv) reconfiguring the role of technical and social competencies that AA professionals could use in specific or decision-making activities, compared to other competencies considered so far as a priority.

Upcoming research could study the discrepancies between the demand and supply of competencies related to the accounting and auditing professions in the GenAI integration context. Equally, modelling organisational learning tools dedicated to the accountability of AA professionals regarding the risks associated with the GenAI use may raise the interest of both researchers and practitioners in the future. Furthermore, developing new domain ontologies, with the capability to integrate and leverage the results of the LLM-based GenAI system and to systematically enrich organisational acquis, could constitute another direction of future research. Such tools could ensure a meaningful knowledge base for daily operations and a real-time platform for continuous learning for both young professionals and experienced specialists in the accounting and audit area.

Author Contributions: Conceptualization, I.-F.A.-P., M.V., L.-E.A.-P., I.-D.C. and C.-G.T.; methodology, M.V. and L.-E.A.-P.; resources, I.-F.A.-P., M.V., L.-E.A.-P., I.-D.C. and C.-G.T.; writing—original draft preparation, I.-F.A.-P., M.V., L.-E.A.-P., I.-D.C. and C.-G.T.; writing—review and editing, I.-F.A.-P., M.V., L.-E.A.-P., I.-D.C. and C.-G.T.; supervision, I.-F.A.-P. and C.-G.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data are contained within the article. The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author/s.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Al-Hattami, H.M. University Accounting Curriculum, IT, and Job Market Demands: Evidence From Yemen. *Sage Open* **2021**, *11*, 215824402110071. [CrossRef]
2. Tavares, M.C.; Azevedo, G.; Marques, R.P.; Bastos, M.A. Challenges of Education in the Accounting Profession in the Era 5.0: A Systematic Review. *Cogent Bus. Manag.* **2023**, *10*, 2220198. [CrossRef]

3. Grosu, V.; Cosmulese, C.G.; Socoliuc, M.; Ciubotariu, M.-S.; Mihaila, S. Testing Accountants' Perceptions of the Digitization of the Profession and Profiling the Future Professional. *Technol. Forecast. Soc. Chang.* **2023**, *193*, 122630. [CrossRef]
4. Jackson, D.; Michelson, G.; Munir, R. Developing Accountants for the Future: New Technology, Skills, and the Role of Stakeholders. *Account. Educ.* **2023**, *32*, 150–177. [CrossRef]
5. Zhao, J.; Wang, X. Unleashing Efficiency and Insights: Exploring the Potential Applications and Challenges of ChatGPT in Accounting. *J. Corp. Account. Financ.* **2024**, *35*, 269–276. [CrossRef]
6. Friedrich, M.P.A.; Zanievicz, M.; Cadorna Venturini, J.; Eduardo Schuster, W. Epistemological thinking about accounting in the era of artificial intelligence. *Rev. Gestão Organ.* **2022**, *15*, 180–197. [CrossRef]
7. Munoko, I.; Brown-Liburd, H.L.; Vasarhelyi, M. The Ethical Implications of Using Artificial Intelligence in Auditing. *J. Bus. Ethics* **2020**, *167*, 209–234. [CrossRef]
8. Kommunuri, J. Artificial Intelligence and the Changing Landscape of Accounting: A Viewpoint. *Pac. Account. Rev.* **2022**, *34*, 585–594. [CrossRef]
9. Fotoh, L.E.; Lorentzon, J.I. The Impact of Digitalization on Future Audits. *J. Emerg. Technol. Account.* **2021**, *18*, 77–97. [CrossRef]
10. Mordor Intelligence. *AI In Accounting Market Size & Share Analysis—Growth Trends & Forecasts (2024–2029)*; 2024. Available online: https://www.mordorintelligence.com/industry-reports/artificial-intelligence-in-accounting-market (accessed on 20 March 2024).
11. Vasarhelyi, M.A.; Moffitt, K.C.; Stewart, T.; Sunderland, D. Large Language Models: An Emerging Technology in Accounting. *J. Emerg. Technol. Account.* **2023**, *20*, 1–10. [CrossRef]
12. Rawashdeh, A. A Deep Learning-Based SEM-ANN Analysis of the Impact of AI-Based Audit Services on Client Trust. *J. Appl. Account. Res.* **2023**, *25*, 594–622. [CrossRef]
13. Han, H.; Shiwakoti, R.K.; Jarvis, R.; Mordi, C.; Botchie, D. Accounting and Auditing with Blockchain Technology and Artificial Intelligence: A Literature Review. *Int. J. Account. Inf. Syst.* **2023**, *48*, 100598. [CrossRef]
14. Zhang, C.; Zhu, W.; Dai, J.; Wu, Y.; Chen, X. Ethical Impact of Artificial Intelligence in Managerial Accounting. *Int. J. Account. Inf. Syst.* **2023**, *49*, 100619. [CrossRef]
15. Aldredge, M.; Rogers, C.; Smith, J. The Strategic Transformation of Accounting into a Learned Profession. *Ind. High. Educ.* **2021**, *35*, 83–88. [CrossRef]
16. Vărzaru, A.A. Assessing Artificial Intelligence Technology Acceptance in Managerial Accounting. *Electron.* **2022**, *11*, 2256. [CrossRef]
17. Nguyen, T.-M.; Malik, A. Impact of Knowledge Sharing on Employees' Service Quality: The Moderating Role of Artificial Intelligence. *Int. Mark. Rev.* **2022**, *39*, 482–508. [CrossRef]
18. Korzynski, P.; Mazurek, G.; Altmann, A.; Ejdys, J.; Kazlauskaite, R.; Paliszkiewicz, J.; Wach, K.; Ziemba, E. Generative Artificial Intelligence as a New Context for Management Theories: Analysis of ChatGPT. *Cent. Eur. Manag. J.* **2023**, *31*, 3–13. [CrossRef]
19. Al-Htaybat, K.; von Alberti-Alhtaybat, L.; Alhatabat, Z. Educating Digital Natives for the Future: Accounting Educators' Evaluation of the Accounting Curriculum. *Account. Educ.* **2018**, *27*, 333–357. [CrossRef]
20. Rodgers, W.; Al-Shaikh, S.; Khalil, M. Protocol Analysis Data Collection Technique Implemented for Artificial Intelligence Design. *IEEE Trans. Eng. Manag.* **2024**, *71*, 6842–6853. [CrossRef]
21. Chen, B.; Wu, Z.; Zhao, R. From Fiction to Fact: The Growing Role of Generative AI in Business and Finance. *J. Chin. Econ. Bus. Stud.* **2023**, *21*, 471–496. [CrossRef]
22. Al Naqbi, H.; Bahroun, Z.; Ahmed, V. Enhancing Work Productivity through Generative Artificial Intelligence: A Comprehensive Literature Review. *Sustainability* **2024**, *16*, 1166. [CrossRef]
23. OECD Recommendation of the Council on OECD Legal Instruments Artificial Intelligence. 2019. Available online: https://oecd.ai/en/ai-principles (accessed on 10 June 2024).
24. Arrieta, A.B.; Díaz-Rodríguez, N.; Del Ser, J.; Bennetot, A.; Tabik, S.; Barbado, A.; Garcia, S.; Gil-Lopez, S.; Molina, D.; Benjamins, R.; et al. Explainable Artificial Intelligence (XAI): Concepts, Taxonomies, Opportunities and Challenges toward Responsible AI. *Inf. Fusion* **2020**, *58*, 82–115. [CrossRef]
25. Li, Q.; Liu, J. Development of an Intelligent NLP-Based Audit Plan Knowledge Discovery System. *J. Emerg. Technol. Account.* **2020**, *17*, 89–97. [CrossRef]
26. Jovanovic, M.; Campbell, M. Generative Artificial Intelligence: Trends and Prospects. *Computer* **2022**, *55*, 107–112. [CrossRef]
27. Yu, P.; Xu, H.; Hu, X.; Deng, C. Leveraging Generative AI and Large Language Models: A Comprehensive Roadmap for Healthcare Integration. *Healthcare* **2023**, *11*, 2776. [CrossRef] [PubMed]
28. Lim, W.M.; Gunasekara, A.; Pallant, J.L.; Pallant, J.I.; Pechenkina, E. Generative AI and the Future of Education: Ragnarök or Reformation? A Paradoxical Perspective from Management Educators. *Int. J. Manag. Educ.* **2023**, *21*, 100790. [CrossRef]
29. Dwivedi, Y.K.; Kshetri, N.; Hughes, L.; Slade, E.L.; Jeyaraj, A.; Kar, A.K.; Baabdullah, A.M.; Koohang, A.; Raghavan, V.; Ahuja, M.; et al. Opinion Paper: "So What If ChatGPT Wrote It?" Multidisciplinary Perspectives on Opportunities, Challenges and Implications of Generative Conversational AI for Research, Practice and Policy. *Int. J. Inf. Manag.* **2023**, *71*, 102642. [CrossRef]
30. Sahoo, S.; Kumar, S.; Abedin, M.Z.; Lim, W.M.; Jakhar, S.K. Deep Learning Applications in Manufacturing Operations: A Review of Trends and Ways Forward. *J. Enterp. Inf. Manag.* **2023**, *36*, 221–251. [CrossRef]
31. Victor, B.G.; Sokol, R.L.; Goldkind, L.; Perron, B.E. Recommendations for Social Work Researchers and Journal Editors on the Use of Generative AI and Large Language Models. *J. Soc. Soc. Work Res.* **2023**, *14*, 563–577. [CrossRef]

32. Ooi, K.-B.; Tan, G.W.-H.; Al-Emran, M.; Al-Sharafi, M.A.; Capatina, A.; Chakraborty, A.; Dwivedi, Y.K.; Huang, T.-L.; Kar, A.K.; Lee, V.-H.; et al. The Potential of Generative Artificial Intelligence Across Disciplines: Perspectives and Future Directions. *J. Comput. Inf. Syst.* **2023**, 1–32. [CrossRef]
33. Khan, M.S.; Umer, H. ChatGPT in Finance: Applications, Challenges, and Solutions. *Heliyon* **2024**, *10*, e24890. [CrossRef]
34. Dowling, M.; Lucey, B. ChatGPT for (Finance) Research: The Bananarama Conjecture. *Financ. Res. Lett.* **2023**, *53*, 103662. [CrossRef]
35. Lv, K.; Yang, Y.; Liu, T.; Gao, Q.; Guo, Q.; Qiu, X. Full Parameter Fine-Tuning for Large Language Models with Limited Resources. *arXiv* **2023**, arXiv:2306.09782. [CrossRef]
36. Trad, F.; Chehab, A. Prompt Engineering or Fine-Tuning? A Case Study on Phishing Detection with Large Language Models. *Mach. Learn. Knowl. Extr.* **2024**, *6*, 367–384. [CrossRef]
37. White, J.; Fu, Q.; Hays, S.; Sandborn, M.; Olea, C.; Gilbert, H.; Elnashar, A.; Spencer-Smith, J.; Schmidt, D.C. A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT. *arXiv* **2023**, arXiv:2302.11382. [CrossRef]
38. Căliman, A.; Pop, M. *Today Software Magazine*; 2024. Available online: https://www.todaysoftmag.ro/article/4038/strategii-pentru-o-inginerie-eficienta-a-prompturilor (accessed on 10 June 2024).
39. Kong, A.; Zhao, S.; Chen, H.; Li, Q.; Qin, Y.; Sun, R.; Zhou, X.; Wang, E.; Dong, X. Better Zero-Shot Reasoning with Role-Play Prompting. *arXiv* **2023**, arXiv:2308.07702. [CrossRef]
40. Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E.; Le, Q.; Zhou, D. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *arXiv* **2022**, arXiv:2201.11903. [CrossRef]
41. Miao, J.; Thongprayoon, C.; Suppadungsuk, S.; Krisanapan, P.; Radhakrishnan, Y.; Cheungpasitporn, W. Chain of Thought Utilization in Large Language Models and Application in Nephrology. *Med (B Aires)* **2024**, *60*, 148. [CrossRef] [PubMed]
42. Masikisiki, B.; Marivate, V.; Hlophe, Y. Investigating the Efficacy of Large Language Models in Reflective Assessment Methods through Chain of Thought Prompting. In Proceedings of the 4th African Human Computer Interaction Conference, East London South Africa, 27 November 2023; ACM: New York, NY, USA, 2023; pp. 44–49.
43. Floridi, L. AI as Agency Without Intelligence: On ChatGPT, Large Language Models, and Other Generative Models. *Philos. Technol.* **2023**, *36*, 15. [CrossRef]
44. Piktus, A. Online Tools Help Large Language Models to Solve Problems through Reasoning. *Nature* **2023**, *618*, 465–466. [CrossRef]
45. Cardy, R.L.; Selvarajan, T.T. Competencies: Alternative Frameworks for Competitive Advantage. *Bus. Horiz.* **2006**, *49*, 235–245. [CrossRef]
46. McClelland, D.C. Testing for Competence Rather than for "Intelligence". *Am. Psychol.* **1973**, *28*, 1–14. [CrossRef]
47. Boyatzis, R. *The Competent Manager*; Wiley: Hoboken, NJ, USA, 1982.
48. The Alan Turing Institute. *AI Skills for Business Competency Framework Https://Www.Turing.Ac.Uk/Skills/Collaborate/Ai-Skills-Business-Framework*; The Alan Turing Institute: London, UK, 2023.
49. Andreassen, R.-I. Digital Technology and Changing Roles: A Management Accountant's Dream or Nightmare? *J. Manag. Control* **2020**, *31*, 209–238. [CrossRef]
50. Ng, C. Teaching Advanced Data Analytics, Robotic Process Automation, and Artificial Intelligence in a Graduate Accounting Program. *J. Emerg. Technol. Account.* **2023**, *20*, 223–243. [CrossRef]
51. Moore, W.B.; Felo, A. The Evolution of Accounting Technology Education: Analytics to STEM. *J. Educ. Bus.* **2022**, *97*, 105–111. [CrossRef]
52. Meredith, K.; Blake, J.; Baxter, P.; Kerr, D. Drivers of and Barriers to Decision Support Technology Use by Financial Report Auditors. *Decis. Support. Syst.* **2020**, *139*, 113402. [CrossRef]
53. Salijeni, G.; Samsonova-Taddei, A.; Turley, S. Big Data and Changes in Audit Technology: Contemplating a Research Agenda. *Account. Bus. Res.* **2019**, *49*, 95–119. [CrossRef]
54. Mathisen, A.; Nerland, M. The Pedagogy of Complex Work Support Systems: Infrastructuring Practices and the Production of Critical Awareness in Risk Auditing. *Pedagog. Cult. Soc.* **2012**, *20*, 71–91. [CrossRef]
55. Mancini, D.; Lombardi, R.; Tavana, M. Four Research Pathways for Understanding the Role of Smart Technologies in Accounting. *Meditari Account. Res.* **2021**, *29*, 1041–1062. [CrossRef]
56. Noordin, N.A.; Hussainey, K.; Hayek, A.F. The Use of Artificial Intelligence and Audit Quality: An Analysis from the Perspectives of External Auditors in the UAE. *J. Risk Financ. Manag.* **2022**, *15*, 339. [CrossRef]
57. Mat Ridzuan, N.I.; Said, J.; Razali, F.M.; Abdul Manan, D.I.; Sulaiman, N. Examining the Role of Personality Traits, Digital Technology Skills and Competency on the Effectiveness of Fraud Risk Assessment among External Auditors. *J. Risk Financ. Manag.* **2022**, *15*, 536. [CrossRef]
58. Plumlee, R.D.; Rixom, B.A.; Rosman, A.J. Training Auditors to Perform Analytical Procedures Using Metacognitive Skills. *Account. Rev.* **2015**, *90*, 351–369. [CrossRef]
59. Andiola, L.M.; Downey, D.H.; Earley, C.E.; Jefferson, D. Wealthy Watches Inc.: The Substantive Testing of Accounts Receivable in the Evolving Audit Environment. *Issues Account. Educ.* **2022**, *37*, 37–51. [CrossRef]
60. Jemine, G.; Puyou, F.-R.; Bouvet, F. Technological Innovation and the Co-Production of Accounting Services in Small Accounting Firms. *Account. Audit. Account. J.* **2024**, *37*, 280–305. [CrossRef]
61. Westermann, K.D.; Bedard, J.C.; Earley, C.E. Learning the "Craft" of Auditing: A Dynamic View of Auditors' On-the-Job Learning. *Contemp. Account. Res.* **2015**, *32*, 864–896. [CrossRef]

62. AldemİR, C.; Uçma Uysal, T. AI COMPETENCIES FOR INTERNAL AUDITORS IN THE PUBLIC SECTOR. *Edpacs* **2024**, *69*, 3–21. [CrossRef]
63. Norzelan, N.A.; Mohamed, I.S.; Mohamad, M. Technology Acceptance of Artificial Intelligence (AI) among Heads of Finance and Accounting Units in the Shared Service Industry. *Technol. Forecast. Soc. Change* **2024**, *198*, 123022. [CrossRef]
64. Cardon, P.; Fleischmann, C.; Logemann, M.; Heidewald, J.; Aritz, J.; Swartz, S. Competencies Needed by Business Professionals in the AI Age: Character and Communication Lead the Way. *Bus. Prof. Commun. Q.* **2023**, *27*, 223–246. [CrossRef]
65. Dean, S.A.; East, J.I. Soft Skills Needed for the 21st-Century Workforce. *Int. J. Appl. Manag. Technol.* **2019**, *18*, 17–32. [CrossRef]
66. Duff, A.; Hancock, P.; Marriott, N. The Role and Impact of Professional Accountancy Associations on Accounting Education Research: An International Study. *Br. Account. Rev.* **2020**, *52*, 100829. [CrossRef]
67. Timpson, M.; Bayerlein, L. Accreditation without Impact: The Case of Accreditation by Professional Accounting Bodies in Australia. *Aust. Account. Rev.* **2021**, *31*, 22–34. [CrossRef]
68. IFAC. 2024. Available online: https://www.ifac.org/who-we-are/our-purpose (accessed on 10 June 2024).
69. Andon, P.; Clune, C. Governance of Professional Accounting Bodies: A Comparative Analysis. *Account. Audit. Account. J.* **2021**, *34*, 1769–1801. [CrossRef]
70. ICAEW; ACA. Qualification Professional Development Ladders. 2018. Available online: https://www.icaew.com/-/media/corporate/files/for-current-aca-students/training-agreement/professional-development-overview.ashx (accessed on 10 June 2024).
71. ACCA. ACCA Competency Framework. 2020. Available online: https://www.accaglobal.com/content/dam/ACCA_Global/qual/competencyframework/ACCA-competency-framework-how-and-when-to-use-2020.pdf (accessed on 10 June 2024).
72. AICPA; CIMA; CGMA. Competency Framework. 2022. Available online: https://us.aicpa.org/content/dam/cgma/resources/tools/downloadabledocuments/cgma-competency-framework.pdf (accessed on 10 June 2024).
73. ICAS. ICAS Mapping Our New Competencies. 2023. Available online: https://www.icas.com/__data/assets/pdf_file/0005/617018/RPE-Timeline-and-Competencies_21-08-23.pdf (accessed on 10 June 2024).
74. IMA. IMA Management Accounting Competency Framework. 2023. Available online: https://prodcm.imanet.org/-/media/IMA/Files/Home/Career-Resources/Management-Accounting-Competencies/IMA-Framework-11-28-23.ashx (accessed on 10 June 2024).
75. Thottoli, M.M. Leveraging Information Communication Technology (ICT) and Artificial Intelligence (AI) to Enhance Auditing Practices. *Account. Res. J.* **2024**, *37*, 134–150. [CrossRef]
76. Burns, M.B.; Igou, A. "Alexa, Write an Audit Opinion": Adopting Intelligent Virtual Assistants in Accounting Workplaces. *J. Emerg. Technol. Account.* **2019**, *16*, 81–92. [CrossRef]
77. Kroon, N.; do Céu Alves, M.; Martins, I. The Impacts of Emerging Technologies on Accountants' Role and Skills: Connecting to Open Innovation—A Systematic Literature Review. *J. Open Innov. Technol. Mark. Complex.* **2021**, *7*, 163. [CrossRef]
78. Leitner-Hanetseder, S.; Lehner, O.M.; Eisl, C.; Forstenlechner, C. A Profession in Transition: Actors, Tasks and Roles in AI-Based Accounting. *J. Appl. Account. Res.* **2021**, *22*, 539–556. [CrossRef]
79. Uscov, S.; Groza, A. Cât de inteligent este Artificial Intelligence Act? *Curierul Judic.* **2022**, *2*, 70–83.
80. Schwartz, R.; Vassilev, A.; Greene, K.; Perine, L.; Burt, A.; Hall, P. *Towards a Standard for Identifying and Managing Bias in Artificial Intelligence*; Special Publication (NIST SP); National Institute of Standards and Technology: Gaithersburg, MD, USA, 2022. [CrossRef]
81. Kirk, H.; Jun, Y.; Iqbal, H.; Benussi, E.; Volpin, F.; Dreyer, F.A.; Shtedritski, A.; Asano, Y.M. Bias Out-of-the-Box: An Empirical Analysis of Intersectional Occupational Biases in Popular Generative Language Models. *arXiv* **2021**, arXiv:2102.04130. [CrossRef]
82. Arnold, V.; Collier, P.A.; Leech, S.A.; Rose, J.M.; Sutton, S.G. Can Knowledge Based Systems Be Designed to Counteract Deskilling Effects? *Int. J. Account. Inf. Syst.* **2023**, *50*, 100638. [CrossRef]
83. Samiolo, R.; Spence, C.; Toh, D. Auditor Judgment in the Fourth Industrial Revolution. *Contemp. Account. Res.* **2024**, *41*, 498–528. [CrossRef]
84. Zhu, C.; Guan, Y. The Risks and Countermeasures of Accounting Artificial Intelligence. In Proceedings of the 2022 3rd International Conference on Electronic Communication and Artificial Intelligence (IWECAI), Zhuhai, China, 14–16 January 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 358–361.
85. Koreff, J.; Baudot, L.; Sutton, S.G. Exploring the Impact of Technology Dominance on Audit Professionalism through Data Analytic-Driven Healthcare Audits. *J. Inf. Syst.* **2023**, *37*, 59–80. [CrossRef]
86. Mökander, J. Auditing of AI: Legal, Ethical and Technical Approaches. *Digit. Soc.* **2023**, *2*, 49. [CrossRef]
87. Chen, W.; He, W.; Shen, J.; Tian, X.; Wang, X. Systematic Analysis of Artificial Intelligence in the Era of Industry 4.0. *J. Manag. Anal.* **2023**, *10*, 89–108. [CrossRef]
88. Massaro, M.; Dumay, J.; Guthrie, J. On the Shoulders of Giants: Undertaking a Structured Literature Review in Accounting. *Account. Audit. Account. J.* **2016**, *29*, 767–801. [CrossRef]
89. Page, M.J.; McKenzie, J.E.; Bossuyt, P.M.; Boutron, I.; Hoffmann, T.C.; Mulrow, C.D.; Shamseer, L.; Tetzlaff, J.M.; Akl, E.A.; Brennan, S.E.; et al. The PRISMA 2020 Statement: An Updated Guideline for Reporting Systematic Reviews. *BMJ* **2021**, *372*, n71. [CrossRef]
90. Donthu, N.; Kumar, S.; Mukherjee, D.; Pandey, N.; Lim, W.M. How to Conduct a Bibliometric Analysis: An Overview and Guidelines. *J. Bus. Res.* **2021**, *133*, 285–296. [CrossRef]

91. Ndlovu, S.G. Private Label Brands vs National Brands: New Battle Fronts and Future Competition. *Cogent Bus. Manag.* **2024**, *11*, 2321877. [CrossRef]
92. Tiron-Tudor, A.; Deliu, D. Big Data's Disruptive Effect on Job Profiles: Management Accountants' Case Study. *J. Risk Financ. Manag.* **2021**, *14*, 376. [CrossRef]
93. Wölfel, M.; Shirzad, M.B.; Reich, A.; Anderer, K. Knowledge-Based and Generative-AI-Driven Pedagogical Conversational Agents: A Comparative Study of Grice's Cooperative Principles and Trust. *Big Data Cogn. Comput.* **2023**, *8*, 2. [CrossRef]
94. Dong, Y.; Ding, S.; Ito, T. An Automated Multi-Phase Facilitation Agent Based on LLM. *IEICE Trans. Inf. Syst.* **2024**, *107*, 426–433. [CrossRef]
95. McKinsey The Economic Potential of Generative AI: The Next Productivity Frontier. 2023. Available online: https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/the-economic-potential-of-generative-ai-the-next-productivity-frontier (accessed on 10 June 2024).
96. Nazir, A.; Wang, Z. A Comprehensive Survey of ChatGPT: Advancements, Applications, Prospects, and Challenges. *Meta-Radiol.* **2023**, *1*, 100022. [CrossRef]
97. Roumeliotis, K.I.; Tselikas, N.D. ChatGPT and Open-AI Models: A Preliminary Review. *Future Internet* **2023**, *15*, 192. [CrossRef]
98. Bansal, G.; Chamola, V.; Hussain, A.; Guizani, M.; Niyato, D. Transforming Conversations with AI—A Comprehensive Study of ChatGPT. *Cogn. Comput.* **2024**. [CrossRef]
99. Seo, J.; Oh, D.; Eo, S.; Park, C.; Yang, K.; Moon, H.; Park, K.; Lim, H. PU-GEN: Enhancing Generative Commonsense Reasoning for Language Models with Human-Centered Knowledge. *Knowl. Based Syst.* **2022**, *256*, 109861. [CrossRef]
100. Sumbal, M.S.; Amber, Q. ChatGPT: A Game Changer for Knowledge Management in Organizations. *Kybernetes* **2024**. [CrossRef]
101. Aksamija, A.; Yue, K.; Kim, H.; Grobler, F.; Krishnamurti, R. Integration of Knowledge-Based and Generative Systems for Building Characterization and Prediction. *Artif. Intell. Eng. Des. Anal. Manuf.* **2010**, *24*, 3–16. [CrossRef]
102. Orlova, E. Dynamic Regimes for Corporate Human Capital Development Used Reinforcement Learning Methods. *Mathematics* **2023**, *11*, 3916. [CrossRef]
103. Mathew, R.; Stefaniak, J.E. A Needs Assessment to Support Faculty Members' Awareness of Generative AI Technologies to Support Instruction. *TechTrends* **2024**. [CrossRef]
104. Mazzullo, E.; Bulut, O.; Wongvorachan, T.; Tan, B. Learning Analytics in the Era of Large Language Models. *Analytics* **2023**, *2*, 877–898. [CrossRef]
105. Lucas, H.C.; Upperman, J.S.; Robinson, J.R. A Systematic Review of Large Language Models and Their Implications in Medical Education. *Med. Educ.* **2024**. [CrossRef]
106. Borah, A.R.; Nischith, T.N.; Gupta, S. Improved Learning Based on GenAI. In Proceedings of the 2024 2nd International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT), Bengaluru, India, 4–6 January 2024; IEEE: Piscataway, NJ, USA, 2024; pp. 1527–1532.
107. Shimizu, I.; Kasai, H.; Shikino, K.; Araki, N.; Takahashi, Z.; Onodera, M.; Kimura, Y.; Tsukamoto, T.; Yamauchi, K.; Asahina, M.; et al. Developing Medical Education Curriculum Reform Strategies to Address the Impact of Generative AI: Qualitative Study. *JMIR Med. Educ.* **2023**, *9*, e53466. [CrossRef] [PubMed]
108. Salinas-Navarro, D.E.; Vilalta-Perdomo, E.; Michel-Villarreal, R.; Montesinos, L. Designing Experiential Learning Activities with Generative Artificial Intelligence Tools for Authentic Assessment. *Interact. Technol. Smart Educ.* **2024**. [CrossRef]
109. Hemachandran, K.; Verma, P.; Pareek, P.; Arora, N.; Rajesh Kumar, K.V.; Ahanger, T.A.; Pise, A.A.; Ratna, R. Artificial Intelligence: A Universal Virtual Tool to Augment Tutoring in Higher Education. *Comput. Intell. Neurosci.* **2022**, *2022*, 1410448. [CrossRef] [PubMed]
110. Kramer, L.L.; ter Stal, S.; Mulder, B.C.; de Vet, E.; van Velsen, L. Developing Embodied Conversational Agents for Coaching People in a Healthy Lifestyle: Scoping Review. *J. Med. Internet Res.* **2020**, *22*, e14058. [CrossRef] [PubMed]
111. Hendriksen, C. Artificial Intelligence for Supply Chain Management: Disruptive Innovation or Innovative Disruption? *J. Supply Chain. Manag.* **2023**, *59*, 65–76. [CrossRef]
112. Cui, Y.G.; van Esch, P.; Phelan, S. How to Build a Competitive Advantage for Your Brand Using Generative AI. *Bus. Horiz.* **2024**, *in press*. [CrossRef]
113. Perera Molligoda Arachchige, A.S. Large Language Models (LLM) and ChatGPT: A Medical Student Perspective. *Eur. J. Nucl. Med. Mol. Imaging* **2023**, *50*, 2248–2249. [CrossRef] [PubMed]
114. Singh, K.; Chatterjee, S.; Mariani, M. Applications of Generative AI and Future Organizational Performance: The Mediating Role of Explorative and Exploitative Innovation and the Moderating Role of Ethical Dilemmas and Environmental Dynamism. *Technovation* **2024**, *133*, 103021. [CrossRef]
115. Ferrara, E. Fairness and Bias in Artificial Intelligence: A Brief Survey of Sources, Impacts, and Mitigation Strategies. *Sci* **2023**, *6*, 3. [CrossRef]
116. Makridakis, S.; Petropoulos, F.; Kang, Y. Large Language Models: Their Success and Impact. *Forecasting* **2023**, *5*, 536–549. [CrossRef]

electronics

MDPI

Review

A Survey on Challenges and Advances in Natural Language Processing with a Focus on Legal Informatics and Low-Resource Languages

Panteleimon Krasadakis [1], Evangelos Sakkopoulos [1,*] and Vassilios S. Verykios [2]

[1] Department of Informatics, University of Piraeus, 18534 Piraeus, Greece; pkras@unipi.gr
[2] School of Sciences and Technology, Hellenic Open University, 26335 Patras, Greece; verykios@eap.gr
* Correspondence: sakkopul@unipi.gr

Abstract: The field of Natural Language Processing (NLP) has experienced significant growth in recent years, largely due to advancements in Deep Learning technology and especially Large Language Models. These improvements have allowed for the development of new models and architectures that have been successfully applied in various real-world applications. Despite this progress, the field of Legal Informatics has been slow to adopt these techniques. In this study, we conducted an extensive literature review of NLP research focused on legislative documents. We present the current state-of-the-art NLP tasks related to Law Consolidation, highlighting the challenges that arise in low-resource languages. Our goal is to outline the difficulties faced by this field and the methods that have been developed to overcome them. Finally, we provide examples of NLP implementations in the legal domain and discuss potential future directions.

Keywords: natural language processing; deep learning; information extraction; large language models; generative AI

1. Introduction

Natural Language Processing is a scientific field combining linguistics and Artificial Intelligence. It has various applications across multiple domains, such as voice assistants, search engines, and language translation services, and as a result, it has been heavily studied throughout the past decade [1]. The number of high-profile implementations of Natural Language Processing highlights its significance. What has enabled the practical use of NLP is the introduction of machine learning in the field. Deep Learning specifically allows complex problems to start being examined or greatly improves previous solutions.

Most Natural Language Processing works are developed and tested on general-domain and English data. This creates two considerable problems. First, NLP techniques may not be applied from one language to another as is, due to the fact that some languages have different grammar or characters (e.g., Japanese). Second, the structure and terms used in specific domains may create significant obstacles, like in medical or legal documents (with terms that do not appear in any other kind of document) or Twitter comments (where the use of slang or irony is dominant). As a result, the efficiency of NLP models takes a serious hit when applied to low-resource languages or other domains and, of course, even more so when combined [2].

The application of Natural Language Processing in the legal domain has started to gain traction and be investigated further, as it would greatly benefit that domain [3], but is still lacking in comparison to other domains. The main tasks that researchers try to solve in the sub-field are the Entity processing tasks, namely, Named Entity Recognition (NER), Entity Linking (EL), Relation Extraction (RelEx), and Coreference Resolution(Coref). Other important tasks include classification, summarization, translation, judgment prediction, and question answering (Figure 1).

In an endeavor to make this survey paper comprehensive, it would be unrealistic to encompass all related works for every Natural Language Processing (NLP) task that is applicable within the realm of the legal domain. Hence, in this work, our emphasis is primarily on the entity processing tasks that comprise NER, EL, RelEx, and Coref. We have selected these four crucial tasks as they are instrumental in achieving our ultimate quest, which is the development of a version control system accustomed for legal documentation and law consolidation.

Law consolidation involves merging multiple legislative acts that deal with the same or related subjects into a single, coherent legal text. The purpose is to organize the law more systematically and make it easier to understand for both legal professionals and the general public. It helps users to comprehend the relationships and dependencies between laws, streamlining the application and interpretation of legal concepts. It is essentially a process used to simplify the legal system, helping to identify which legal articles interact with a particular law. To better understand our desired goal and its implications, we will elaborate further with an example.

Consider a law practitioner who wants to read a specific law. They need a couple of things that might be taken for granted but are not always provided. First, they want to find the most recent version of the law, since laws can change as new legislation is introduced. They also want to easily track how that law has evolved over time (version control system). Next, they would like to identify the links and references to and from that law. This is important for seeing which legal articles the law interacts with (law consolidation). Even though these data seem crucial, they are not readily available in most countries, either from governmental records or even paid services. As an example, Eunomos [4] is a similar system conceptually that uses ontologies to achieve its objectives.

So, after the example, let us clarify why the aforementioned tasks are necessary for our goal. We need a system that can automatically extract the mentioned legal entities (NER) in a legislative document. These may be entire laws or really specific parts of them, like articles or even sentences in paragraphs of articles. Unfortunately, there are times when an abbreviation of a law can be translated to more than one law or different versions of the same law, having undergone major revisions over the years, so we have to disambiguate them properly (EL). It is also common that there are references to the "above law" or a law that is mentioned only in context, so Coreference Resolution is also necessary. Finally, we need to find the type of connection between them (mentioned in Section 2.1), as it will affect the legislation differently (RelEx). On the other hand, the tasks of summarization or classification may lose the nuances and precise use of language in legal documents required for a law consolidation system, so they were not investigated in this work.

As a result, we believe that laying the foundations in this field is critical to showing the progress so far and push the research forward. We present the related works for each of the above four tasks, with an added focus on non-English language approaches and multilingual methods that can be applied to other low-resource languages as well.

For the purposes of this survey, we have employed a hybrid approach combining both State-of-the-Art Review and Scoping Review methodologies in the field of Natural Language Processing (NLP). This approach provides both an in-depth examination of the most recent research developments in the ever-evolving field of NLP and a broad exploration of the breadth of the literature in this area. With a specific focus on low-resource languages and the legal domain, the aim of this survey is to comprehensively appraise how the featured advanced NLP techniques are currently being applied, as well as their potential future applications, in these specific contexts. The ultimate goal is to provide a valuable resource that may stimulate and guide future research at the intersection of NLP, law, and low-resource languages.

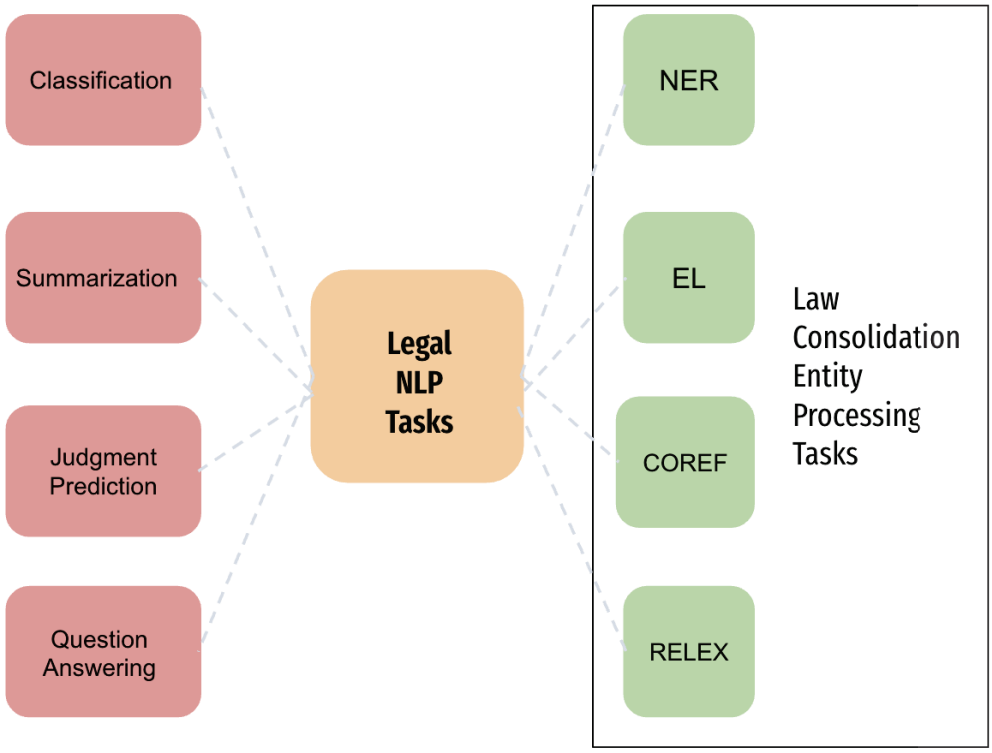


Figure 1. Natural Language Processing tasks for legal documents.

In Section 2, we state the essential information on the problem. In Section 3, we provide an extensive presentation of the related work in the area of Natural Language Processing for the tasks of Named Entity Recognition, Entity Linking, Coreference Resolution, and Relation Extraction. Section 4 focuses on multilingual and low-resource language NLP research. Then, we continue in Section 5 by describing the advancements in the field of legal NLP. Finally, Section 6 suggests future steps in our research and in this field in general and concludes this paper.

2. Background Information

In this section, we provide some essential background information on the subjects addressed in this paper. We briefly describe the peculiarities of legal data and provide an overview of Deep Learning Neural Networks leading to the current state of the art.

2.1. Legal Data

Legal documents have distinctive characteristics that set them apart from other types of documents. They are primarily categorized into laws, case laws, legislative articles, and administrative documents. These documents are often interconnected and can be complicated due to their continuous expansion. Legal documents are connected in three ways: insertion, where a passage of text is added verbatim in the original; repeal, where the new document revokes a specific fragment of the original; and substitution, where the new legislation replaces a part of the original. It is often difficult to identify the type of connection between legal documents, and the fact that they only affect a portion of the original document makes it increasingly challenging to validate the current state of a legal document [5].

NLP practices have yet to achieve their full potential in the legal domain due to a lack of annotated legal datasets. Despite the clear benefits of NLP for the legal domain, there is a significant shortage of quality data. The implementation of Deep Learning techniques is heavily dependent on data quality, and the legal domain often lacks openly accessible data. The constant release of new laws also makes it necessary to have a version control system

of legislation, which is currently not provided. With these issues in mind, our research began in this area [6].

The legal domain presents many challenges for NLP. Some major challenges include disambiguating titles (e.g., Prime Minister), resolving nested entities, and resolving coreferences. Titles may require disambiguation to a specific person based on the time, year, and country. Abbreviations in titles or laws may require deep contextual knowledge to identify. Nested entities, such as titles of legislative articles referring to laws, add another layer of complexity. Coreference resolution, which is frequently encountered, may be complicated by intersecting laws. Legislation is often uploaded in PDF format, which is not machine-readable and poses its own challenges. Lengthy paragraphs spanning numerous pages are common in legal documents, making it challenging to apply NLP techniques, such as Relation Extraction and Coreference Resolution.

While there are many important tasks in legal document processing, our research focuses on those related to our goal. Some other tasks worth mentioning are classification, summarization, and judgment prediction. With classification, by labeling laws according to the subdomain that they touch upon (e.g., Admiralty law), we can facilitate the search for and connection between legal documents. Likewise, summarization (which is a task close to classification) aids legal professionals in quickly acquiring the relevant information of a document. Judgment prediction is a highly demanding task that requires our two-fold attention. It is the extremely interesting and challenging task of automatically obtaining a prediction on the ruling of a case. However, with great power comes great responsibility. The predicted decisions are based on data from previous cases, which unfortunately, more often than not, contain biased information. As a result, this creates a feedback loop that enhances potential discrimination, so their results should not be taken as impartial rulings, and it is necessary to address this issue at its core [7].

2.2. Natural Language Processing Outline

We now present a brief outline of the technologies used for Natural Language Processing, leading to the latest advancements (Figure 2). In the following sections of related works, we do not further analyze the properties of the main architectures described here to focus on the variations for each specific subtask. In this section, we mention the fundamental architectures that have been successfully applied in the field and have contributed to its advancement in the recent past.

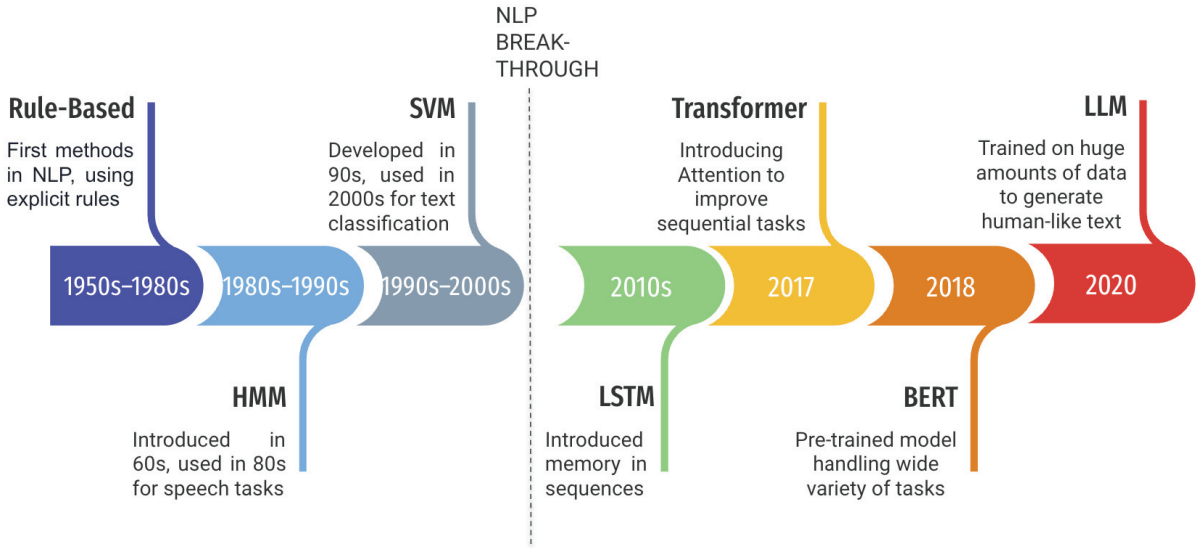


Figure 2. Timeline of essential Text NLP techniques.

Over the years, various techniques have been proposed for Natural Language Processing (NLP). Initially, rule-based approaches were built based on expert knowledge and linguistic rules to extract the desired information. Later, supervised and unsupervised learning techniques were introduced in the field. Supervised methods require a manually annotated corpus to solve the problem as a classification problem, while unsupervised learning requires less initial labeled data and allows the system to self-evolve to find new rules. NLP researchers have tested many methods, such as Hidden Markov Models (HMMs), Support Vector Machines (SVMs), and Conditional Random Fields (CRFs). With the emergence of Deep Learning in most fields, NLP research has shifted its focus in this direction in recent years [8].

Deep Learning and Deep Neural Networks (DNNs) are not new inventions, but the limitations in terms of hardware kept them from being examined as feasible models for many years. As we all know, Graphics Processing Units (GPUs) have been constantly improving over the years and, a couple of years ago, reached the point where they were capable of handling Deep Learning Neural Networks at an affordable price. This reignited the interest of many researchers, followed by the suggestion of improved models and techniques. In principle, there is no real difference between regular Neural Networks and DNNs, except that the latter have many hidden layers (hence, they are deep). This increase in depth increases the computational requirements but also enables solutions to complex problems that were impossible before. The other technique that cleared the way for many ground-breaking implementations is transfer learning, which is a machine learning technique that was devised for problems that are lacking in data but are similar to ones with a lot of resources available. These algorithms train on a broader problem and try to apply the trained model with some fine-tuning to the related problem [9].

The introduction of two Deep Learning models in the field of NLP changed the landscape forever. First, Long Short-Term Memory models (LSTMs) [10] started in the mid-1990s as a theoretical extension of Recurrent Neural Networks (RNNs) to address their issue with memory and the vanishing gradient problem. It was not until two decades later that these models started being implemented in practice and revitalized the interest in Deep Learning Neural Networks in NLP. Many of the state-of-the-art solutions nowadays are variations of or contain LSTM models and perform well in many scenarios. In terms of our score, LSTMs alleviate the issue of long-distance relationships (between entities). When text is processed in a Recurrent Neural Network, it does not maintain any information from previous iterations or past sentences, so no connection between distant entities can be established. LSTMs, however, preserve the most important information throughout the next steps, acquiring, as a result, a form of memory. The two most common LSTM configurations that we encounter are bidirectional models (biLSTMs) and sequence-to-sequence architectures (seq2seq). The former consists of two LSTMs, passing the important information both forward and backward, with this process enhancing their prediction abilities. The latter also stacks two LSTMs, but this time as an encoder–decoder model.

Despite all of these improvements, there was still a big issue with LSTMs. They only process sequential data and are not fit for parallel processing, making their training (even more so in larger models) really slow. So, the second vital model was developed, and that is Transformers [11]. Taking advantage of the aforementioned potent modern GPUs, Transformers were designed with parallelization as a major part of them. They also have two other defining features. The first is their structure, a sequential enc-dec model composed of multiple stacks. The encoder passes the input through various filters and is fed to the decoder to follow a similar process until the desired output. The second is attention, a novel concept proposed for Neural Networks that helps the Network decide, at each iteration, which are the most significant variables of each sequence to focus on in order to give them bigger weights and improve the final output.

Transformers were designed for neural machine translation applications, and they indeed achieved great results in that area. Nonetheless, their real impact came in the form of BERT (Bidirectional Encoder Representations from Transformers) [12]. BERT is a

pretrained model built on the foundations of Transformers and trained on large amounts of data. The creators of BERT, in order to create a robust model, trained it to solve two challenging and unique tasks: Masked Language Modeling (MLM) and Next Sentence Prediction (NSP). MLM takes a sentence as an input, and then random words are concealed (masked), so the model outputs its predictions of the most appropriate words to fill the masks. NSP takes two sentences as input, and the model has to predict whether they are in succession. After having been heavily trained in these tasks, the model is later fine-tuned to solve other similar NLP tasks.

From the multitude of BERT extensions and variations, the most important ones that we want to discuss are RoBERTa [13], XLNet [14], and GPT-3 [15]. Each one of the above has been carefully developed by one of the industry giants with abundant resources in order to outperform its competition. We mention them in chronological order. RoBERTa (robustly optimized BERT approach) from Facebook AI (Meta AI now) is an optimized BERT variant trained on more data with fine-tuned hyper-parameters that outperforms all other variations up to that point. XLNet is an autoregressive pretrained model by the Google AI Brain Team and combines the pros of the original BERT and Transformer-XL to leverage the disadvantages of both. GPT-3 (a continuation of their previous work, GPT-2) by OpenAI boasts the daunting number of 175 billion parameters trained on an immense amount of data, being the biggest model to date. The team notes the impact of such an endeavor (both technologically and otherwise).

The current landscape of NLP is being driven by Large Language Models (LLMs). LLMs like GPT-3.5/4, PaLM, Bard, and LLaMa not only understand the context but even generate human-like text, translate languages, and, in general, allow us to perform a wide variety of NLP tasks using a single model. OpenAI recently introduced GPT-3.5 and GPT-4, language models that boast powerful and versatile APIs that revolutionized the field [16]. Concurrently, Google's research team developed the PaLM or "PAttern-producing Language Model". The PaLM is built to emulate human-like abilities in language understanding, closely resembling the way human brains decipher and generate language [17]. Meta developed its LLaMa model, the Language Learning and Multimodal Association model, designed to understand and interpret natural languages through textual–visual interactions [18].

These advancements are leading us toward a future where language models will become indispensable tools in the field of NLP, but there are still some issues and risks before establishing them as the only solution, including ethics, bias, safety, and environmental impact, not to mention the potential of fabricated results from these models. In addition to that, these models can expose private data, and regulators have not managed to keep up with the incredible speed at which these models have appeared [19].

3. Related NLP Tasks

Natural Language Processing (NLP) has been gaining increasing attention in the past decade. The need for research on how to improve these techniques is undeniable. We are mainly concerned with the tasks used in the field of Legal Informatics and particularly in creating a Version Control System for Legislation. The tasks that we deemed necessary to research in that regard are Named Entity Recognition and Linking (NER/EL), Relation Extraction, and Coreference Resolution, especially their recent developments with Deep Learning. We continue by reviewing the related work on the essential NLP tasks in the legal domain.

3.1. Named Entity Recognition

Named Entity Recognition (NER) is at the center of Natural Language Processing. This task strives to identify Named Entities in a text. These Named Entities usually fall into the categories of Person, Location (Loc), Organization, and Time/Date, but not exclusively. NER is almost a prerequisite for many other NLP tasks, such as the ones mentioned later in this work, and, as a result, is the most researched one. For a Named Entity Recognition example, consider the sentence "The Prime Minister visited Germany on Tuesday to talk about the new Covid-19 measures". We have the following Named Entities: "Prime

Minster" is a Person, "Germany" is a Location, and "Tuesday" is a Date (Figure 3). All of the NLP-task-related figures were captured in Prodigy (https://prodi.gy/, accessed on 3 February 2024).

Figure 3. Named Entity Recognition example.

Named Entity Recognition may seem like a simple task, but it has many challenges. Language differences can hinder the application of established NLP approaches to other languages with different syntax or alphabets. Nested Named Entities make it extremely difficult to break them down and differentiate them, especially if they depend on context. Entities that are described with multiple words or spans (e.g., "Prime Minister") also affect the process of Entity Recognition. Abbreviations (e.g., "PM" instead of Prime Minister) have a similar effect. Last but not least, a really important aspect that is not mentioned enough is the value of well-thought labeling schemes and appropriate datasets. This value is increased in subdomains (e.g., legal), where the quality of Entity Recognition can vary greatly depending on the selected labels of Named Entities that need to be identified. At the same time, the data must support these Entities sufficiently in order to train the machine learning models. The above two issues apply to the other tasks as well, but since NER precedes them, a wrong NER scheme will greatly affect all of them, while the opposite is not necessarily true.

As a highly researched problem, many methods have been suggested over the years as approaches to NER. Initially, pattern extraction techniques were used to gather the desired entities from semantic or syntactic information. These formulations could not address many of the challenges mentioned above, so research pivoted to machine learning. The main strategies that were proposed revolved around Support Vector Machines (SVMs), Conditional Random Fields (CRFs), and Markov Models (MMs). These did not provide satisfactory results but laid the foundations for later implementations.

Then, Deep Learning algorithms and word embeddings (or Vectorization) came and capitalized on the earlier ML advancements and started producing really promising results. Embeddings map real words to vectors of numbers (suitable for machine learning), capturing the contextual or semantic similarity of the words. The first great efforts used RNN and LSTM architectures. These reached the current state of the art, when they were combined with CRFs and Convolutional Neural Networks (CNNs) [20].

The state-of-the-art methods for Named Entity Recognition revolve around new pretrained Transformer-based models like BERT and RoBERTa that we described previously. Some of the most interesting advancements in the field of word representation or embeddings include LUKE [21], ACE [22], and CL-KL [23]. Language Understanding with Knowledge-based Embeddings (LUKE) is a Transformer-based model with an entity-aware attention mechanism and treats entities as tokens (individual words or terms) for better relationship representation between entities. This architecture has great results in Named Entity Recognition, Relation Extraction, and question answering. The authors of Automated Concatenation of Embeddings (ACE), focus on finding better word representations instead of a better model architecture, deeming it an equally important part of NLP tasks. They designed a controller for embedding concatenation and noted how their model can be implemented in other existing models to boost their performance. Finally, the authors of [23] suggest that injecting knowledge from a search engine improves the contextual representation of the input. Then,

they introduce two Cooperative Learning models for NER with substantial results, raising interest in the further exploration of Cooperative Learning in the field.

For cross-domain NER, multiple methods have been suggested over the years. Recent approaches make use of the Deep Learning advancements in the general field, most notably language models, multitasking, and transfer learning [24]. The usual tactic is to train on the general NER dataset CoNLL and try to transfer the model to other domains, like medicine or news. The developers of L2AWE (Learning To Adapt with Word Embeddings) [25] claim their method can function on new domains without the need to retrain the NER model thanks to robust word embeddings like Word2Vec, which outperforms, in these cases, the contextual BERT embeddings. A different point of view is given in BERT-Assisted Open-Domain Named Entity Recognition with Distant Supervision (BOND) [26], where the authors decided to make use of the popular pretrained BERT models with a two-step training framework. First, they fine-tuned the RoBERTa model with distant supervision labels to imbue the model with semantic knowledge, and for the second step, they replaced these labels with a teacher-student framework to improve model fitting with training cycles on pseudo-labels.

Of course, LLMs and GPT have been tested for NER as well. The main contributions with meaningful and comparable results are presented in PromptNER [27] and GPT-NER [28]. The former is an innovative NER approach based on prompting. PromptNER demonstrates leading performance in few-shot learning and cross-domain NER. The methodology comprises four crucial elements: a backbone LLM, a modular definition outlining the entity types, a small set of examples from the target domain, and a well-defined format for presenting the extracted entities. GPT-NER adheres to the overarching concept of in-context learning and can be broken down into three sequential steps: (1) Prompt Construction, (2) Input to LLM, and (3) Text Sequence Transformation.

In most cases, the evaluation of the presented techniques is based on the precision, recall, and F1 score, and for our work, we will use the same metrics [29]. In IE, precision, p = RR/All, represents the ratio of relevant retrieved (RR) documents to the total of retrieved documents (All); e.g., in a text search query, it would be the number of relevant results divided by the number of all results. Recall, r = RR/Relevant, is the fraction of relevant documents that were retrieved to the total of relevant documents that should have been retrieved. F1 is the harmonic mean of precision and recall, $F1 = 2 \times (p \times r)/(p + r)$.

In Table 1, we present the current state-of-the-art contributions. We want to note that many of these papers perform multiple NLP tasks, but for simplicity purposes, we only state the relevant task per table. In a similar vein, if a paper has results on multiple datasets, we present the comparable ones when they are available (meaning the datasets used by most papers). We also have to note at this point that in the last row of each of the following tables, the best legal approach to that task is presented. However, we do not mention them in the corresponding paragraphs just yet, but we present them all together in Section 5.

Table 1. Major contributions to Named Entity Recognition.

Paper (Year)	Functionality	Evaluation Datasets	Results (F1%)
LSTM-CRF (2016) [20]	Baseline/Multiple Languages	CoNLL	90.9 (EN), 78.8 (DE)
BERT (2019) [12]	Base Transformer variation	CoNLL, GLUE	92.8, 82.1
RoBERTa (2019) [13]	Optimized BERT	CoNLL, GLUE	92.4, 88.5
LUKE (2020) [21]	BERT + self-attention	CoNLL	94.3
ACE [1] (2021) [22]	**Word representations**	**CoNLL**	**94.6 (EN), 88.3 (DE)**
BiLSTM-CRF (2019) [30]	Neural cross-domain	BioNLP13PC, CBS	85.5, 73.6
PromtNER (2023) [27]	LLM prompt-based NER	CoNLL	83.48 (GPT4)
GPT-NER (2023) [28]	Transforms NER into LLM task	CoNLL	90.91 (Few-Shot Davinci003)
Legal-BERT (2020) [31]	BERT trained on legal data	CONTRACTS-NER	94

[1] In bold, we highlight the approach with best results.

3.2. Entity Linking

Now that we have finished with the presentation of Named Entity Recognition, we move on to a closely tied problem: Entity Linking (EL). They are often approached as a single problem and are either managed as pipelines or joint models, with Deep Learning still being the core of the state-of-the-art approaches [32]. The task of Entity Linking requires the disambiguation of entities based on a Knowledge Base. The most common Knowledge Bases used for this task are Wikipedia, YAGO (which combines Wikipedia and WordNet data), and DBPedia (multilingual structured content from Wikipedia). As an example, consider the sentence "The Prime Minister visited Athens". We need to uniquely identify the "Prime Minister" mentioned in the sentence and also link them to the correct place they visited. As a matter of fact, there are more than 20 cities called "Athens" across the world, and someone may assume that, by default, it refers to Athens, the capital of Greece, which may not be always correct.

The main tasks that define Entity Linking are Mention Detection, Candidate Entity Generation and Ranking, and Entity Disambiguation [33]. First, the system needs to generate the candidate entities that may be linked to the examined entity and then generate a ranking of the list of potential candidates based on the probability of correct linking, and finally, a way to deal with the unlinkable entities is required. Throughout the years, various approaches have been proposed for each task separately, from heuristic to supervised and unsupervised methods, but recently, the research has shifted to end-to-end models, usually implemented with Deep Neural Networks.

The first end-to-end Neural Entity Linking model was established by the team of Kolitsas et al. [34]. They noted that the main challenge they wanted to address is finding the correct span of Named Entities to link. As a result, they proposed a joint NER and EL neural approach and used Wikipedia as their Knowledge Base. A different process is followed in Entity Linking using Densified Knowledge Graphs (ELDEN) [35]. In ELDEN, they try to solve Entity Linking as a graph problem and state how the density (number of edges in Knowledge Graph) of a candidate directly affects EL performance, so they suggest a Densified Knowledge Graph with pseudo-entities as input.

For the most recently released techniques in the field, we have the following. Broscheit, in [36], pondered the implementation of an end-to-end BERT model for the three tasks of Entity Linking. They simplified these tasks to train BERT based on English Wikipedia and fine-tuned it for EL, making it the first EL model without any pipeline or heuristics. The authors of [37] observed that the Transformer models under-perform in comparison to the biLSTM model of Kolitsas [34], even though, generally, BERT models are state of the art. This initiated their research in new ways to implement them in EL, and they came up with CHOLAN. It is a modular transformer architecture that reverts to breaking down the problem into its subtasks, instead of providing a joint solution. They trained two models independently, one for Mention Detection and one for Entity Disambiguation, which makes them flexible and interoperable for different Knowledge Bases.

Then, we reference the remarkable contributions of De Cao et al. [38,39], who proposed two autoregressive approaches (the latter is the equivalent for multilingual purposes). GENRE (Generative ENtity REtrieval) is based on a seq2seq BART architecture to autoregressively generate entity names. This system allows them to capture the relations between NEs and their contexts while reducing the required memory. Finally, there is SPEL—Structured Prediction for Entity Linking [40]. SPEL is a state-of-the-art Entity Linking system that implements innovative concepts to enhance the structured prediction in EL. This includes two detailed fine-tuning stages and a context-aware prediction aggregation approach, minimizing the model's output vocabulary size and tackling a prevalent issue in Entity Linking systems, where there's a discrepancy between training and inference tokenization. They presented the best results in the field and also compared the method with GPT-3.5 and GPT-4, which showed how LLMs are still fairly behind in this NLP task, with a significantly increased cost.

The same metrics are used in Entity Linking, namely, precision, recall, and F1, but we have observed that, in contrast to the available results in NER (where almost all papers use F1), in EL, some papers use precision (P). Most Entity Linking approaches use the common Knowledge Database of Wikipedia and the AIDA-CoNLL and TAC or ACE datasets for evaluation. There is a problem in comparing Entity Linking models deriving from the fact that the task is broken down into the subtasks mentioned above, and some papers deal with Entity Linking as a whole, while others with each subtask separately. Table 2 summarizes the main contributions to this sub-field.

Table 2. Major contributions to Entity Linking.

Paper (Year)	Functionality	Evaluation Datasets	Results (F1%)
Neural End-to-End (2018) [34]	Baseline	AIDA, ACE	82.4, 68.3
Elden (2018) [35]	Knowledge Graphs	AIDA, TAC	93, 89.6 (P)
BERTEL (2020) [36]	No-pipeline BERT implementation	AIDA	79.3
CHOLAN (2021) [37]	Modular Transformer	AIDA, ACE	83.1, 86.8
GENRE (2021) [38]	Autoregressive BART	AIDA	83.7
SpEL [1] (2023) [40]	**State-of-the-art EL**	**AIDA**	**88.6**
GPT-4 (2023) [40]	Few-shot GPT-4 with chain-of-thought	AIDA	66.2
DeepType (2018) [41]	Cross-lingual mixed integer	AIDA, TAC	93, 90
Legal EL (2018) [42]	Transfer learning on legal data	AIDA, EURLEX	88.8, 98

[1] In bold, we highlight the approach with best results.

3.3. Coreference Resolution

Coreference Resolution is a challenging task. Given a text, it tries to identify all indirect references to a certain entity (which is usually a Named Entity). For example, this can be either through the use of pronouns (she, their, etc.) or nominals (e.g., the Prime Minister), and we need to link those with the Named Entity that they refer to. The challenge lies in the fact that it often requires an understanding of the context, either in terms of linguist or "common-sense" knowledge. There are many types of anaphora (around 10, depending on subcategories) that can be found in the written or spoken word, which further increases the challenge.

The extended study of this field started fairly recently and is reflected by the introduction of specified conferences/workshops around it. There was some initial research in the field prior to 2016, but since the introduction of Deep Learning in Natural Language Processing, the field has changed significantly; we focus on these last years of research. It is a crucial task related to many NLP applications, including sentiment analysis (characterizing the sentiment of a text), summarization, translation, question answering, and Named Entity Recognition. Unfortunately, despite its importance, the progress in Coreference Resolution has been the slowest compared to those other fields [43].

Earlier methods revolved around ontologies and the OntoNotes corpus for Entity Linking and Coreference Resolution. The main three categories of coreference solutions are rule-based, Statistical/ML, and Deep Neural Network ones. The first category (with algorithms dating back to 1978) depends on syntactic or semantic rules devised by experts, with the introduction of world knowledge into those rules being an open debate among researchers. The second category of solutions started appearing in the late 1990s, with decision trees, genetic algorithms, and Integer Linear programming being the most prominent methods that overall outperformed the rule-based ones. Finally, Deep Learning approaches began being implemented, further reducing hand-crafted features with the aid of word vectors, being a potent model for representing semantic dependencies between words, and LSTMs and Transformers, producing great results [44].

The four main approaches in the field are Mention-Pair, Mention-Ranking, Entity-Based, and Latent-Tree models. In order, Mention-Pair models are binary classification

models that work on pairs of words and were usually solved with clustering algorithms [45]. Mention-Ranking extends the previous model by implementing a rank in a chain of mentions and linking with the items of the highest rank. The first neural end-to-end approach was implemented based on that idea in [46] and takes advantage of bidirectional LSTMs (biLSTMs). Then, Entity-Based models provide additional information on when to merge pair clusters, again with LSTMs being the key components in Deep Learning implementations [47]. Lastly, Latent-Tree models use tree structures for the coreferences, with the most noteworthy contribution being Higher-Order Inference in Coref [48]. They proposed an attention mechanism to improve the span representations and a pruning method to handle long documents.

Joshi et al. made two major contributions to Coref by adapting BERT for this task [49]. As with many others, they built on the foundations set by the team of Kenton [46,48]. First, they fine-tuned the BERT-large pretrained model for Coreference Resolution on the OntoNotes and GAP datasets and replaced the LSTM and ELMo embeddings of c2f-coref with the BERT Transformer, showcasing great results. Then, they advanced their work with SpanBERT [50]. They noticed how, in many cases, for Coreference Resolution, Named Entity Recognition, and other NLP tasks, critical information is contained within spans of words instead of singular work token entities, and it greatly improved the results if they pretrained BERT with spans. They achieved that by differentiating the pretraining tasks of BERT. Instead of masking random tokens, they tried to predict masked spans, and they introduced a new span-boundary objective, so the model predicts the entire span in a set boundary.

The current state of the art in the field is presented in [51]. This paper presents a simplified text-to-text (seq2seq) method for Coreference Resolution that synergizes with modern encoder–decoder or decoder-only models. The method processes a sentence along with the previous context encoded as a string, predicting coreference links. It offers simplicity—by eliminating the need for a separate mention detection and a higher-order decoder. It boasts improved accuracy over prior approaches and harnesses modern generation models that generate text strings. They focused on how to present Coreference Resolution as a seq2seq issue, introducing three transition systems wherein the seq2seq model inputs a sentence and generates an action reflecting a set of coreference links related to the sentence. As of the moment of writing this paper, we have not found any LLM-powered approaches that present significant results in Coreference Resolution.

Domain-specific research in Coreference Resolution is far from being explored. The two domains with active research are the medical field and reference resolution for scientific papers. In general, the work in [52] is a step forward in the right direction. It takes advantage of SpanBERT, mentioned earlier, and introduces the grouping of similar spans into concepts to better adapt BERT to new domains. They also introduced retrofitting loss and scaffolding loss functions, which, thanks to knowledge distance functions, ensure better span representation in the new domain.

There are multiple publicly available datasets for Coref in different medical subdomains, and over the years, there have been rule-based, machine learning, and now Deep Learning models to try and solve this particular challenging task. Some recent remarkable mentions include the works of the first BERT implementation in the BioMedical field [53] pretrained on PubMed data or the work in [54], in which the authors induced knowledge in an LSTM model for improved results with domain-specific features and word embeddings.

Another major problem in the field that we wanted to mention is how biased datasets affect Coreference Resolution. This was directly addressed in the NLP Workshop about Gendered Ambiguous Pronouns (GAP). The works of Rudinger, Webster, and others [55] note that most existing corpora are gender-biased, more frequently resolving male entities (for example, “President” is more likely to be linked with “he” than “she”). They also lack Gendered Ambiguous Pronouns or GAP resolutions that may require real-world knowledge. They released a dataset of ambiguous pronouns derived from Wikipedia for gender fairness, and based on their experiments, Transformer models have the best results.

Agarwal et al., in [56], comment on the evaluation for Coreference Resolution. The main metrics used traditionally are MUC, B^3, and CEAF. All of these metrics fail to capture the real efficiency and accuracy of Coref in various ways. First of all, they do not take into consideration the gender-bias perspective. Then, they also do not calculate whether the references are finally resolved to a Named Entity. This problem occurs in chains of references, for example, in Figure 4, if "her" is just linked to "she", but they are not related to "The Prime Minister", this will not be reflected in the above metrics, but in reality, the information will not be useful. So, they proposed Named Entity Coreference (NEC) and various metrics to address the above issues. Similar work was presented in [57], where the authors introduce a new Link-Based Entity-Aware (LEA) metric, which considers the importance of each entity that we want to resolve. Regardless of the above observations, and as can be seen in Table 3, most related articles on Coreference Resolution make use of MUC, B^3, and CEAF.

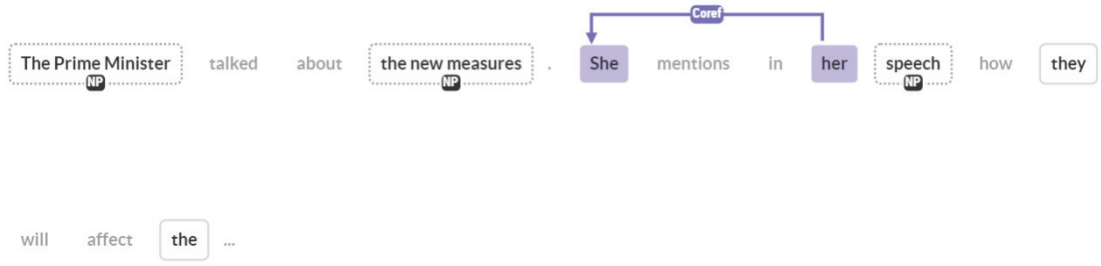


Figure 4. Coreference Resolution example.

Table 3. Major contributions to Coreference Resolution.

Paper (Year)	Functionality	Evaluation Datasets	Results (F1%) MUC, B^3, CEAF
Neural End-to-End (2017) [46]	Baseline	CoNLL-2012	77.2, 66.6, 62,6
RNN with Features (2016) [47]	Global Feature Representations	CoNLL-2012	73.4, 61.5, 57.7
HOI (2018) [48]	High-Order Inference	CoNLL-2012	80.1, 70.5, 67.6
CorefBERT (2019) [49]	BERT implementation for Coref	CoNLL-2012	83.5, 75.3, 71.9
SpanBERT (2020) [50]	Extension of CorefBERT for Spans	CoNLL-2012	85.3, 78.1, 75.3
Seq2Seq (2023) [1] [51]	**A seq2seq Transition-Based System**	**CoNLL-2012**	**87.8, 82.6, 79.5**
SpanDomain (2021) [52]	Cross-domain extension of SpanBERT	CoNLL-2012	72.4, 66.3, 57.6

[1] In bold, we highlight the approach with best results.

3.4. Relation Extraction

Relation Extraction is the task of finding and semantically categorizing a relationship between two Named Entities in a text. This could either define an event (commonly referred to as Event Extraction) that derives from that relationship or a link between those Named Entities. Furthermore, the task is examined both in terms of a single sentence and for document-level extraction [58]. For example, in Figure 5, between the two legal entities "Article 154" and "Regulation (EU) No 1305/2013", we want to find the type of relationship between them, whether the Article inserts (ADD), substitutes (REPLACE), revokes (REPEAL), or simply refers (REFER) to the Regulation, and the difference between them is critical. In this specific example, the additional information of the Date (1 January 2023) would also be required to be extracted in a real-life scenario, since it is important to know when a law starts being applied.

As with the previous tasks, over the years, a variety of rule-based, supervised, and unsupervised machine learning techniques have been suggested, but in recent years, Deep Learning techniques have taken over [59]. The main methods that are currently examined

are variations of CNNs/RNNs, distant-supervised models, knowledge-based methods, and Transformer implementations.

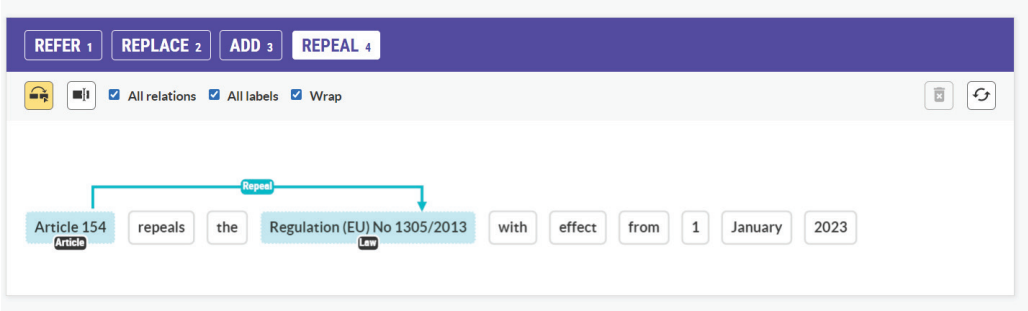


Figure 5. Relation Extraction example.

A common approach in the field uses attention-based Deep Neural Networks. In [60], a Convolutional Neural Network is proposed with two levels of attention, one for the entities and one for their relationships. One of the more modern approaches is the combination of the above type of Neural Networks with biLSTM (RNN) in addition to regular DNNs, as presented in [61]. The idea comes from taking advantage of the strengths of each type of Neural Network and stacking them all together, with a CNN being used for its rich feature extraction, a DNN for long distance between words, and a DNN to improve the overall performance.

Distant supervision has been closely examined for Relation Extraction [62]. Distant supervision systems use Knowledge Bases as training data (such as DBPedia or Wikidata) in semi-structured key–value pairs. A recent and robust baseline method is introduced in [63], which consists of three steps, namely, Passage Construction, Passage Encoding, and Passage Summarization, and extends the BERT-based pretrained model. Many researchers have modified BERT models for the task of Relation Extraction. The first BERT implementation for this task, R-BERT, is introduced in the work of Wu and He [64], which takes advantage of entity-level information and achieves state-of-the-art results. Another configuration consists of a stack of a BERT model and biGRU (bidirectional Gated Recurrent Unit), as presented in [65]. They use the biGRU to extract the important features from the results of BERT and to obtain the position information in a sentence (useful in long sentences).

According to our research, the current state-of-the-art approaches progress based on the previous ideas. First, the REBEL architecture [66] is an end-to-end autoregressive seq2seq model for Relation Extraction. The authors also released the corresponding distantly supervised dataset, and they aim to provide a flexible and easy-to-adjust approach in terms of both domains and document- or sentence-level RelEx. The team behind [67] tested how the Transformer architecture can be applied to Relation Extraction and devised a novel way to do so. They established the Matching The Blanks method, which is similar to the Masked Language Modeling of BERT, where they replace entities with "Blank" statements and try to find relationships in that environment. KGPool is another novel method [68] for RelEx using Knowledge Graphs. First, it examines the way knowledge is inserted in Graph Convolution Networks (CGNs), and then it uses a self-attention mechanism to properly select sub-graphs of information from the Knowledge Graph (the first attempt in Relation Extraction).

Xu et al. [69] presented a robust approach for document-level RelEx. They noted how structure is important in document-level dependencies and that graph models are lacking in that regard. Instead, they suggested a Structured Self-Attention Network (SSAN) with a modified attention mechanism for the effective representation of structure dependencies. The currently best approach in L RelEx is DREEAM [70]. DREEAM (Document-level Relation Extraction with Evidence-guided Attention Mechanism) is a method that is efficient in memory use and utilizes evidence data as supervision input. This assists the attention mechanisms of the DocRE framework to assign high weights to the evidence. Secondly, they put forth a self-training approach for DREEAM to acquire Entity Resolution (ER)

from automatically created evidence based on extensive data, eliminating the need for annotations.

Out of the four examined NLP tasks, LLMs have had the biggest impact so far in Relation Extraction, especially in zero- or few-shot learning. Two of the most recent contributions that we want to cite are QA4RE [71] and GoLLIE [72]. In the former, the authors mention that the subpar RelEx performance of instruction-tuned LLMs may stem from the low occurrence of RelEx tasks in instruction-tuning datasets. To combat this, they suggest the QA4RE framework, which integrates RelEx with the frequently appearing multiple-choice question answering (QA). They frame the input sentence as a question and potential relation types as multiple-choice answers, enabling LLMs to conduct RelEx by selecting the correct relation type.

GoLLIE (Guideline-following Large Language Model for IE) enhances model performance on unseen schemas by focusing on guidelines' details. They used a Python-code-based representation for both the model's input and output, providing a human-readable structure and addressing common issues with natural language instructions. It allows any information extraction task to be represented in a unified format. The key contribution here is the incorporation of the guidelines in the inference process for improved zero-shot generalization. They standardized the input format, with label definitions as class docstrings and candidates as principal argument comments. To ensure that the model follows the guidelines, they introduced a variety of noise during training, preventing the model from associating particular datasets or labels.

In specific fields, once again, the medical domain receives attention, where we have already seen results for Relation Extraction. The work in [73] is a thorough survey presenting the current modern Neural Network approaches in the field. Another interesting approach is given in [74]. ReTrans, as they call it, is a transfer learning framework that takes advantage of existing Knowledge Bases to deal with relation extraction in new domains.

As we stated, it is not uncommon for NLP researchers to try and tackle tasks in relative groups. So, the problems of Entity Linking, Coreference Resolution, and Relation Extraction have been examined for joint solutions. A great and up-to-date survey on the field can be found in [75] and is noted as the only survey addressing Deep Learning techniques in information extraction (IE). They start by presenting the main datasets used in NER and the various methods used to solve the problem, and similarly for Relation Extraction. Some other noteworthy works in joint approaches are the encoder–decoder model for Entity and Relation Extraction in [76], where the authors describe two methods, one with a representation scheme for tuples and one for pointer-network-based decoding. Then, there is the work of Zaporojets et al. [77] for Entity Linking and Coreference Resolution for documents, where the proposed method translates the problem to a Maximum Spanning Tree (MST) problem, making use of Span-BERT.

Relation Extraction has been examined quite a lot over the years, but the developed methods focus on heavily specific and curated datasets with strict and clear definitions of Named Entities and relationships [78]. This is good for examining the approaches in theory, but in reality, the problems are much more complex, so it is hard to apply these methods efficiently. This also makes it harder to compare these methods, and this is confirmed by our observations presented in Table 4.

Table 4. Major contributions to Relation Extraction.

Paper (Year)	Functionality	Evaluation Datasets	Results (F1%)
Distant Supervision (2021) [63]	Distant Baseline	NYT	61.5
R-BERT (2019) [64]	BERT extension for RelEx	SemEval-2010	89.2
Rebel (2021) [66]	End-to-End Language Generation	NYT, DocRED	92, 47.1
KGPool (2021) [68]	Graph-based RelEx	NYT	86.7
DREEAM (2023) [70]	Document-level RelEx	DocRED	67.5
QA4RE (2023) [71]	Transforms RelEx to QA for zero-shot GPT-3.5 solution	TACRED, SemEval	59.4, 43.3

Table 4. *Cont.*

Paper (Year)	Functionality	Evaluation Datasets	Results (F1%)
GoLLIE (2023) [72]	Zero-shot IE Code-LLaMA model	ACE	70.1
CNN-UD (2022) [79]	Cross-lingual RelEx with Universal Dependencies	SemEval-2010	82 (EN), 64 (FR)
Legal Extraction (2020) [80]	Feature extraction from legal data	Custom legal dataset	83.6

4. Multilingual and Low-Resource-Language NLP

Unfortunately, most languages other than English, Spanish, and Chinese have very few related resources for Natural Language Processing. We refer to these as low-resource languages. In examining various research results in the field, we have observed that the efficiency of general NLP techniques, when applied in other domains and languages, is significantly lower. Moreover, papers that touch on cross-lingual approaches, more often than not, test their models on Spanish or Chinese (both high-resource languages), highlighting the importance of research in the field [81].

As a result, in the past few years, we have observed increased interest in research for other languages to address this directly. Many papers have been written in the past years alongside the advent of Deep Learning in NLP, which is a direct indication that it is becoming more and more relevant. We noticed that most papers released before this last period (2016–2022) have been severely outdated in terms of both the tools and methods used.

In the past couple of years, we have observed a growth in papers for cross-lingual Named Entity Recognition. The research for these subjects is really important for low-resource languages [82]. First of all, the team of BERT has released a multilingual version, mBERT, and according to the experiments in [83], it generalizes fairly well, but its shortcomings derive from multilingual word representations, highlighting the significance of language-specific embeddings. A remarkable approach to cross-lingual NER is presented in [84] by a Microsoft team. They had industry needs in mind when they proposed a Reinforcement Learning and Knowledge distillation framework to transfer knowledge from an initial weak English model to the new non-English model. They mark the weakness of existing cross-lingual models in real-life applications (especially search engine-related tasks) and present state-of-the-art results.

Because Entity Linking functions with the help of Knowledge Bases, cross-domain and language implementations are not considered. That would require a KB with data from multiple domains, alongside an advanced system that can identify and link entities to each of these domains, and based on our research, we have not seen any records of such a work. We have only found a select few papers about cross-lingual EL [85]. They mention how challenging this task is for low-resource languages. The minimum requirements for such a system to work are an English KB (like Wikipedia), a source language KB, multilingual embeddings and bilingual entity maps, and the last two are especially rare for many languages. DeepType is the most interesting related architecture [41]. The authors integrated symbolic information into the reasoning process of the Neural Network with a type system. They translated the problem to a mixed-integer one, and they showed that their model performed well in multilingual experiments.

Similarly, for most languages other than English, there are very few resources and research papers for Coreference Resolution. There are some for widely spoken languages such as Chinese, Japanese, and Arabic, but for most low-resource languages, there is no progress whatsoever. A common approach to counter these issues is multilingual or cross-lingual systems [86]. A recent example in the research of these methods for Coreference Resolution is presented here [87]. These methods perform based on the basics of transfer learning, where they are usually pretrained in English (which has a plethora of word embeddings, corpora, and pretrained models), and try to transfer that knowledge to other languages. A transfer learning method for cross-lingual Relation Extraction is proposed

in [79], which capitalizes on Universal Dependencies and CNNs to achieve great Relation Extraction in low-resource languages.

The main issue for any low-resource language in the current state of Deep Learning is that the latest advancements in the field, namely, the large pretrained Transformer-based models (like BERT), cannot be transferred reliably or efficiently. Both the word embeddings (a major preprocessing part) and the vast amount of data used to pretrain the models are in English. This makes most of the BERT variants (not specifically trained in another language) unusable in other domains, and their performance diverges greatly from that reported in state-of-the-art works [88]. Consequently, LSTM implementations in these subdomains often present better results in subdomains/other languages than BERT. We believe that it is important to consider this and research new ways to either adapt large pretrained models more profitably or focus more on cross-lingual and cross-domain models or even evaluate the usefulness of these models as a whole in these cases [89].

In regard to the LLM implementations in a cross-lingual environment, the most promising work can be found in [90]. In that work, the authors mention how recent studies suggest that visual supervision enhances LLMs' performance in various NLP tasks. In particular, the Vokenization approach [91] has charted a new path for integrating visual information into LLM training in a monolingual context. Building on this, they crafted a cross-lingual Vokenization model and trained a cross-lingual LLM on English, Urdu, and Swahili. Their experiments show that visually supervised cross-lingual transfer learning significantly boosts performance in numerous cross-lingual NLP tasks, like cross-lingual Natural Language Inference and NER, for low-resource languages.

In Table 5, we present the gathered papers for our NLP subtasks. Most cross-lingual methods used for low-resource languages approach the issue similarly. They use the English part of Wikipedia (or WikiData) as their main language for training, along with the desired language to transfer the knowledge to. They often combine that with bilingual entity maps (especially when we have Knowledge Bases) to map entities between source and destination languages. Multilingual embeddings may also contribute significantly to the process by mapping the vectors of the same word in different languages and clustering them together. The results presented in Table 5 follow the same principles, so the trained language is commonly English, and we only state the destination language for the task. Below the table, we provide the interpretation of the language codes used for the tests in each paper.

Table 5. Major contributions in multilingual NLP.

Paper (Year)	Functionality	Languages	Datasets	Results (F1%)
mBERT (2019) [83]	NER	DE [1] NL, ES	CoNLL (NER)	69.7 77.4, 73.6
RIKD (2021) [84]	NER	DE NL, ES	CoNLL (NER)	75.5 82.5, 77.9
DeepType (2018) [41]	NER, EL	DE FR, ES	Wiki (EL)	97.5 96.6, 97.6
Zero XEL (2019) [85]	EL	OM [2] RW, SI	Wiki (EL)	38.4 44.9, 64.4
mGENRE (2021) [39]	EL	ES, ZH [3]	TAC	86.7, 88.4
Lazy End2End (2021) [87]	COREF EL	IT NL, ES ES, ZH	SemEval (Coref) TAC (EL)	43 38, 51.6 81.1, 83.9
CNN-UD (2022) [79]	RelEx	FA, FR AR [4]	SemEval (RelEx) ACE (RelEx)	56.1, 63.9 59.7
VLM (2023) [90]	NER	EN, SW UR [5]	GLUE	82.9, 61 45.9

[1] DE = German; NL = Dutch, ES = Spanish; [2] OM = Oromo, RW = Kinyarwanda, SI = Sinhala, really low-resource languages; [3] ZH = Simplified Chinese; [4] FA = Farsi, FR = French, AR = Arabic; [5] SW = Swahili, AR = Urdu, really low-resource languages.

5. Legal NLP

As we mentioned in Section 2, Legal Data and Informatics have several challenges that are exclusive to or more prominent in the domain. In this section, we present the recent literature published to address these issues.

For the legal domain, NER is the most researched NLP task, as it is the primary one [92]. Most common Named Entities in the field include Person/Title (e.g., Judge), Date, Organization, and then the different types of law documents, which differ depending on the use case or country. From the various approaches that have been proposed, we chose the following as the most promising recent ones.

A noteworthy contribution in the field is LEGAL-BERT [31]. It is the first BERT implementation in the legal domain. Chalkidis and his colleagues state the numerous challenges they faced. They are greatly concerned with the proper configuration of the many variables and hyper-parameters used in BERT implementations. They suggest that, in many cases, small models can prove to be more efficient while providing competitive results, counter to the current trend of extremely big models. They developed three different models for BERT based on the pretraining steps that they follow. They trained their models on 12GB of English legal text, and after testing and comparing their models, they concluded that adapting BERT to new domains requires either extensive further training or even pretraining from scratch.

The team of [93] developed a NER architecture for legal documents in German. They prepared a manually annotated dataset with German court decisions with 19 NEs (that fall under the four that we just mentioned). They then suggested a biLSTM-CRF model to achieve state-of-the-art NER results. A similar work is submitted in [94]. The authors fine-tuned a widely used German BERT language model on a Legal Entity Recognition (LER) dataset that was also used by the previous authors. To prevent overfitting, they undertook a stratified 10-fold cross-validation. Their results showed that the fine-tuned German BERT outperformed the BiLSTM-CRF+ model on the same LER dataset.

We have also developed an efficient Named Entity Recognition model for Greek Legislation [95]. As cited in the paper, there are very few NER models in Greek, and of course, they do not find the very specialized entities that we are looking for. Our approach was to manually annotate a fairly small corpus of Greek legal documents (around 4000 paragraphs out of 150 documents) and fine-tune a generic non-BERT Greek NER model with the help of these data. Throughout the process, we confronted the many challenges described throughout this paper, with really promising results in the end. Based on our research and implementation, we have drawn several key conclusions. Firstly, having a well-defined annotation schema that avoids overlapping entities and class imbalance is crucial. Secondly, active learning significantly reduces the manual annotation effort over time. Thirdly, to avoid overfitting, it is necessary to retrain the model from scratch. Finally, despite having a much smaller annotated dataset than the BERT models, we achieved satisfactory results that outperformed the larger models, mainly because of the quality of our annotations.

The authors of [96] worked on extracting entities for Mergers and Acquisitions. The issue they wanted to solve is more specific than NER, because in contracts, there may be many Named Entities, but the relevant ones needed for their work are a specific subset. The architecture presented was used in production, and the dataset they tested on was curated by law professionals and not by machine learning techniques. They chose to develop a binary classifier instead of a multi-label one because of the limitations imposed by the users and the large data imbalance. They propose two strategies. A baseline single layer based on CRFs with three variations (according to the sentences trained) and a two-layer strategy that expands on the previous one by training a sentence-level CRF, again with three similar variations.

Entity Linking works in the legal domain are scarce. A common Knowledge Base developed in the domain is Legal Knowledge Interchange Format (LKIF) [97]. In [98], the authors present a general review on the uses of ontologies in Legal Informatics. They analyzed the term ontology and its significance in specific domains and proposed an open automated system for providing EU countries with legal information based on

ontologies. In [99], they jointly tackled Named Entity Recognition and Linking with the use of ontologies. They semantically represented legal entities and tried to map YAGO to LKIF ontologies by capitalizing on Wikipedia data. Since the legal domain does not have an extensive training corpus, Named Entity Linking with transfer learning is considered as a solution. In [42] specifically, the authors transferred knowledge from the AIDA-CoNLL dataset (a widely used EL dataset) to the EURLEX corpus (which has EU legislation), but in our opinion, they did not produce any exciting results.

For Coreference Resolution in the legal domain, the only related works that we found in our extensive research over the past 5 years are the works of Gupta et al. [100] and Ji et al. [101]. First, they suggested a supervised machine learning process to identify references to participants in court judgments. Due to the lack of legal-specific datasets, they decided to map their entities to the ACE dataset, and their results show that more similar approaches should be examined. The second work explored the problem of Speakers Coreference Resolution (SCR) in court records. They noted how existing models cannot be implemented in the field as is, due to the highly knowledge-rich nature of legal documentation. They proposed an ELMo pretrained biLSTM model with attention, in parallel with a graph containing entities with "mentioned" relationships. Both of these papers focus on very specific problems of legal Coreference Resolution, and as such, there is much room for research in this field that we deem necessary for the future of Legal Informatics.

In the legal domain, Relation Extraction is an important task that can be used to either find connections between legislative articles or events mentioned in legal documents. The team of Dragoni et al. [102] suggested a combination of NLP approaches for rule extraction, which is a task closely related to Relation Extraction. They combined ontologies to identify the structure and linguistic elements of legal documents with the Stanford Parser for the grammatical features and a Combinatory Categorial Grammar tool to extract logical dependencies between words. Relation Extraction was also considered for regulatory compliance in [103], alongside fact orientation to create a domain model and a dictionary.

Event Extraction in legal data is also of relevance. It is usually needed in court decisions, as stated in [104], where the task of finding and connecting all the relevant events in a case was extremely time-consuming. They tested different pretrained models and concluded that it is better to fine-tune a large existing model with domain-specific knowledge rather than training from scratch on a smaller domain corpus. Event Extraction in a Chinese legal text environment is presented in [105], where the authors propose a combination of BERT and biLSTM-CRF for character vectors and rule extraction, respectively. Finally, a joint entity and Relation Extraction system for Chinese legal documents is described in [80]. Based on a sequence-to-sequence (seq2seq) framework, they developed the Legal Triplet Extraction System (LTES) to extract entities and their relationships in drug-related criminal cases.

The use of cross-domain knowledge in legal data is not a considered tactic. The peculiarities and specifications of legal data make it hard for general-purpose knowledge to be transferable. Nevertheless, we have to mention the work of [106], where the authors further trained a RoBERTa model on three different (small) legal datasets and suggested, based on their experiments, that these language models gain robustness when trained on multiple datasets.

Nowadays, multilingual law processing is becoming more necessary than ever, especially in the European Union, where all country members have their respective laws and languages but also have to adhere to the EU legislation (referred to as National Implementing Measures or National Transposition) [107]. The research, however, has only started to bear fruits. We want to mention here the recent seminal works of Chalkidis et al. [108–110].

Throughout the combined efforts of these papers, the authors aimed to improve multilingual legal NLP capabilities through a Transformer architecture and LLMs. To measure progress in the legal NLP field, they created a challenging multilingual benchmark known as LEXTREME based on 24 languages across 11 legal datasets. This new benchmark tool identified significant room for improvement in current models [108]. To facilitate training LLMs, the authors released a large, high-quality multilingual corpus called MULTILEGALPILE. This

corpus contains diverse legal data sources in 24 languages from 17 jurisdictions. They also pretrained RoBERTa models, setting a new state of the art on LEXTREME [109].

They also introduced legal-oriented Pretrained Language Models (PLMs) trained on a newly released multinational English legal corpus, LeXFiles. The effectiveness of these models was evaluated on a newly released legal knowledge probing benchmark, LegalLAMA. Analysis revealed a strong correlation between probing and upstream performance for related legal topics and identified model size and prior legal knowledge as key drivers of downstream performance [110]. The authors anticipate that their collective effort, encompassing new tools, datasets, and benchmarks, will accelerate the development of domain-specific PLMs and advance legal NLP capabilities. The data, trained models, and code developed during this work are openly available, fostering transparency and encouraging future research in this domain.

The last thing that we want to mention is the current place of LLMs and GPT in the domain. From the results provided throughout this survey, and as also supported in [111], they still fall short compared to the state of the art in tackling the demanding Natural Language Processing tasks in specialized domains like law and in non-English languages. First, these models are largely trained on general-domain data and may not fully understand the specific terminology, structures, and nuances present in legal texts. This limits their ability to accurately predict, generate, or interpret legal language. Second, most available training data are in English, which means that these models are likely to perform poorer on non-English texts due to the lack of sufficient training data. Nevertheless, they still perform reasonably well, even through zero- or few-shot learning, and will probably reach their competition soon.

Table 6 collects the main works on Legal Informatics that, in our opinion, are essential to our work and set the current state of the art that we aim to improve in the future. Contrary to all the previous tables, we can see that, in this case, each paper uses its own custom datasets and experiments, making it hard to replicate and compare them properly. We hope that, in the future, with the generation of the recent domain-specific benchmarks and datasets, the testing and comparison of legal NLP approaches will be improved. Beneath the table, we give brief details for each of the datasets used in the mentioned works.

Table 6. Major contributions to legal NLP.

Paper (Year)	Functionality	Evaluation Datasets	Results (F1%)
Legal-Bert (2020) [31]	NER	CONTRACTS-NER [1]	94
German Legal Bert (2023) [94]	NER	LER [2]	91.2
Greek Legal NER (2022) [95]	NER	Greek Legislation [3]	91
Ontology (2017) [99]	NER, EL	Wiki, ECHR [4]	69 (NER)
Transfer EL (2018) [42]	EL	AIDA, EURLEX	88.8, 98
Speakers Resolution (2019) [101]	Coref	CRDs [5]	83.5
LTES (2020) [80]	RelEx	CJO [6]	83.6
LEXTREME (2023) [108]	NER	LEXTREME [7]	61.6 (XLM-R) 33.9 (GPT-3.5)

[1] CONTRACTS-NER = dataset for NER on US contracts; [2] LER = Legal Entity Recognition dataset, with 750 German court decisions; [3] Greek Legislation documents (law and case law) curated and annotated by Greek Legal Experts; [4] ECHR = English legal judgment prediction dataset of cases from the European Court of Human Rights; [5] CRDs = Court Record Documents, a dataset from real-world civil case court decisions in Chinese; [6] CJO = custom dataset with 1750 drug-related criminal judgment documents; [7] LEXTREME = a collection of 11 legal NLP datasets covering 24 languages. The results are given as averages for all tested datasets.

6. Conclusions

This paper has conducted a thorough exploration of Natural Language Processing (NLP), with a particular focus on Named Entity Recognition (NER), Entity Linking, Relation Extraction, and Coreference Resolution. These aspects are vital for constructing a legal citation network and law consolidation system. We initially delved into modern research on each of these tasks individually, followed by an exploration of their application in the legal domain.

Legal documents, with their interconnectedness, constant evolution, and complex structure, present a multifaceted problem. The three key variables are the insertion, repeal, and substitution of laws. Insufficient datasets and the need for version control only add to the complexity.

Despite significant recent research in NER within this domain, critical gaps remain, particularly regarding disambiguating titles, resolving nested entities, and addressing coreferences, lengthy texts, and machine-inaccessible PDFs. Both Coreference Resolution and Relation Extraction are areas that should be further explored, as their results are noticeably lower than those in NER. The meaningful integration of ontologies and transfer learning for relation and rule extraction offers interesting directions for future research.

Our work indicates that model efficiency and high-quality annotations and datasets could lead to substantial advancements in these areas. While there are legal limitations to what can be achieved in providing openly accessible data, our findings underscore the urgent need for such datasets. These insights should guide future attempts in the legal domain and in broader managerial practices.

This need has created a new field necessary for research: the intersection of Privacy, Legal, and Natural Language Processing fields [19,112]. This is another field that interests us, and we see that many researchers share our interest, especially since the application of the General Data Protection Regulation. Despite its importance, it is still in its early stages of research, as the junction of these fields highlights new issues and requires new techniques to be developed, presumably combining Deep Learning, LLMs, and Hiding techniques [113].

Furthermore, the NLP techniques encountered do not perform well when applied to languages that are less widely spoken than English, Spanish, and Chinese due to a shortage of related resources. Cross-lingual models, such as mBERT, offer potential pathways for addressing these challenges, yet the roles of language-specific embeddings require further research.

Future advancements in NLP applied to legal and especially to low-resource-language texts depend on three main objectives: creating proper and large datasets, refining the accuracy of current models, and unearthing and leveraging new techniques, with Large Language Models gaining increasing prominence. While these new models are yet to reach current standards, their swift progress, along with the creation of expansive legal datasets such as LEXTREME, suggest a promising route toward optimal outcomes in this field.

Our future goals include researching the best way to develop an end-to-end model for low-resource languages in the legal domain to create a law version system. We think the best way to approach this is by finding the best-suited solution for each of the four main tasks and building a joint pipeline model. We have already started with the NER pipeline and look to extend it to include Coreference Resolution and Relation Extraction. Additionally, we are keenly aware of the privacy concerns surrounding Deep Learning and especially LLMs and the law domain, and we intend to explore innovative ways to merge these fields.

Author Contributions: Conceptualization, P.K. and E.S.; software, P.K.; validation, E.S. and V.S.V.; writing—original draft, P.K.; writing—review and editing, E.S. and V.S.V.; visualization, P.K. and E.S.; supervision E.S. and V.S.V. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been partly supported by the University of Piraeus Research Center.

Data Availability Statement: Data sharing is not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

NLP	Natural Language Processing
IE	Information extraction
NER	Named Entity Recognition
EL	Entity Linking
RelEx	Relation Extraction
Coref	Coreference Resolution
HMM	Hidden Markov Models
SVM	Support Vector Machines
CRF	Conditional Random Field
DNN	Deep Neural Networks
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
BERT	Bidirectional Encoder Representations from Transformers
LLM	Large Language Model

References

1. Collobert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; Kuksa, P. Natural Language Processing (Almost) from Scratch. *J. Mach. Learn. Res.* **2011**, *12*, 2493–2537.
2. Hedderich, M.A.; Lange, L.; Adel, H.; Strötgen, J.; Klakow, D. A survey on recent approaches for natural language processing in low-resource scenarios. *arXiv* **2020**, arXiv:2010.12309.
3. Conrad, J.G.; Branting, L.K. Introduction to the special issue on legal text analytics. *Artif. Intell. Law* **2018**, *26*, 99–102. [CrossRef]
4. Boella, G.; Caro, L.D.; Humphreys, L.; Robaldo, L.; Rossi, P.; Torre, L. Eunomos, a Legal Document and Knowledge Management System for the Web to Provide Relevant, Reliable and up-to-Date Information on the Law. *Artif. Intell. Law* **2016**, *24*, 245–283. [CrossRef]
5. Chalkidis, I.; Nikolaou, C.; Soursos, P.; Koubarakis, M. Modeling and Querying Greek Legislation Using Semantic Web Technologies. In Proceedings of the The Semantic Web, Portorož, Slovenia, 28 May 28–1 June 2017; pp. 591–606.
6. Zhong, H.; Xiao, C.; Tu, C.; Zhang, T.; Liu, Z.; Sun, M. How Does NLP Benefit Legal System: A Summary of Legal Artificial Intelligence. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020; pp. 5218–5230. [CrossRef]
7. Tsarapatsanis, D.; Aletras, N. On the Ethical Limits of Natural Language Processing on Legal Text. In Proceedings of the Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, Online, 1–6 August 2021; pp. 3590–3599. [CrossRef]
8. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016. Available online: http://www.deeplearningbook.org (accessed on 3 March 2023).
9. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *J. Mach. Learn. Res.* **2020**, *21*, 5485–5551.
10. Hochreiter, S.; Schmidhuber, J. Long Short-term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
11. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is All You Need. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Red Hook, NY, USA, 4–9 December 2017; pp. 6000–6010.
12. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA, 4 June 2019; Volume 1 (Long and Short Papers), pp. 4171–4186. [CrossRef]
13. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv* **2019**, arXiv:1907.11692.
14. Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.G.; Salakhutdinov, R.; Le, Q.V. XLNet: Generalized Autoregressive Pretraining for Language Understanding. *arXiv* **2019**, arXiv:1906.08237.
15. Brown, T.B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language Models are Few-Shot Learners. *arXiv* **2020**, arXiv:2005.14165.
16. OpenAI. GPT-4 Technical Report. *arXiv* **2023**, arXiv:2303.08774.
17. Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H.W.; Sutton, C.; Gehrmann, S.; et al. PaLM: Scaling Language Modeling with Pathways. *arXiv* **2022**, arXiv:2204.02311.
18. Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. LLaMA: Open and Efficient Foundation Language Models. *arXiv* **2023**, arXiv:2302.13971.
19. Goanta, C.; Aletras, N.; Chalkidis, I.; Ranchordas, S.; Spanakis, G. Regulation and NLP (RegNLP): Taming Large Language Models. *arXiv* **2023**, arXiv:2310.05553.

20. Lample, G.; Ballesteros, M.; Subramanian, S.; Kawakami, K.; Dyer, C. Neural Architectures for Named Entity Recognition. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, CA, USA, 12–17 June 2016; pp. 260–270. [CrossRef]
21. Yamada, I.; Asai, A.; Shindo, H.; Takeda, H.; Matsumoto, Y. LUKE: Deep Contextualized Entity Representations with Entity-aware Self-attention. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online, 16–20 November 2020; pp. 6442–6454. [CrossRef]
22. Wang, X.; Jiang, Y.; Bach, N.; Wang, T.; Huang, Z.; Huang, F.; Tu, K. Automated Concatenation of Embeddings for Structured Prediction. *arXiv* **2020**, arXiv:2010.05006.
23. Wang, X.; Jiang, Y.; Bach, N.; Wang, T.; Huang, Z.; Huang, F.; Tu, K. Improving Named Entity Recognition by External Context Retrieving and Cooperative Learning. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Online, 1–6 August 2021; pp. 1800–1812. [CrossRef]
24. Liu, Z.; Xu, Y.; Yu, T.; Dai, W.; Ji, Z.; Cahyawijaya, S.; Madotto, A.; Fung, P. CrossNER: Evaluating Cross-Domain Named Entity Recognition. *arXiv* **2020**, arXiv:2012.04373.
25. Nozza, D.; Manchanda, P.; Fersini, E.; Palmonari, M.; Messina, E. LearningToAdapt with word embeddings: Domain adaptation of Named Entity Recognition systems. *Inf. Process. Manag.* **2021**, *58*, 102537. [CrossRef]
26. Liang, C.; Yu, Y.; Jiang, H.; Er, S.; Wang, R.; Zhao, T.; Zhang, C. BOND: BERT-Assisted Open-Domain Named Entity Recognition with Distant Supervision. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Virtual Event, 6–10 July 2020; pp. 1054–1064. [CrossRef]
27. Ashok, D.; Lipton, Z.C. PromptNER: Prompting for Named Entity Recognition. *arXiv* **2023**, arXiv:2305.15444.
28. Wang, S.; Sun, X.; Li, X.; Ouyang, R.; Wu, F.; Zhang, T.; Li, J.; Wang, G. GPT-NER: Named Entity Recognition via Large Language Models. *arXiv* **2023**, arXiv:2304.10428.
29. Zhang, Q.; Chen, M.; Liu, L. A Review on Entity Relation Extraction. In Proceedings of the 2017 Second International Conference on Mechanical, Control and Computer Engineering (ICMCCE), Harbin, China, 8–10 December 2017; pp. 178–183. [CrossRef]
30. Jia, C.; Liang, X.; Zhang, Y. Cross-Domain NER using Cross-Domain Language Modeling. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 2464–2474. [CrossRef]
31. Chalkidis, I.; Fergadiotis, M.; Malakasiotis, P.; Aletras, N.; Androutsopoulos, I. LEGAL-BERT: The Muppets straight out of Law School. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2020, Online, 16–20 November 2020; pp. 2898–2904. [CrossRef]
32. Barlaug, N.; Gulla, J.A. Neural Networks for Entity Matching: A Survey. *ACM Trans. Knowl. Discov. Data* **2021**, *15*, 52. [CrossRef]
33. Shen, W.; Wang, J.; Han, J. Entity Linking with a Knowledge Base: Issues, Techniques, and Solutions. *IEEE Trans. Knowl. Data Eng.* **2015**, *27*, 443–460. [CrossRef]
34. Kolitsas, N.; Ganea, O.E.; Hofmann, T. End-to-End Neural Entity Linking. In Proceedings of the 22nd Conference on Computational Natural Language Learning, Brussels, Belgium, 31 October–1 November 2018; pp. 519–529. [CrossRef]
35. Radhakrishnan, P.; Talukdar, P.; Varma, V. ELDEN: Improved Entity Linking Using Densified Knowledge Graphs. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, LA, USA, 1–6 June 2018; Volume 1 (Long Papers), pp. 1844–1853. [CrossRef]
36. Broscheit, S. Investigating Entity Knowledge in BERT with Simple Neural End-to-End Entity Linking. In Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL), Hong Kong, China, 3–4 November 2019; pp. 677–685. [CrossRef]
37. Ravi, M.P.K.; Singh, K.; Mulang', I.O.; Shekarpour, S.; Hoffart, J.; Lehmann, J. CHOLAN: A Modular Approach for Neural Entity Linking on Wikipedia and Wikidata. *arXiv* **2021**, arXiv:2101.09969.
38. Cao, N.D.; Izacard, G.; Riedel, S.; Petroni, F. Autoregressive Entity Retrieval. *arXiv* **2020**, arXiv:2010.00904.
39. Cao, N.D.; Wu, L.; Popat, K.; Artetxe, M.; Goyal, N.; Plekhanov, M.; Zettlemoyer, L.; Cancedda, N.; Riedel, S.; Petroni, F. Multilingual Autoregressive Entity Linking. *arXiv* **2021**, arXiv:2103.12528.
40. Shavarani, H.; Sarkar, A. SpEL: Structured Prediction for Entity Linking. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, Singapore, 6–10 December 2023; pp. 11123–11137. [CrossRef]
41. Raiman, J.; Raiman, O. DeepType: Multilingual Entity Linking by Neural Type System Evolution. *arXiv* **2018**, arXiv:1802.01021.
42. Elnaggar, A.; Otto, R.; Matthes, F. Deep Learning for Named-Entity Linking with Transfer Learning for Legal Documents. In Proceedings of the 2018 Artificial Intelligence and Cloud Computing Conference, Tokyo Japan, 21–23 December 2018; pp. 23–28. [CrossRef]
43. Liu, R.; Mao, R.; Luu, A.T.; Cambria, E. A brief survey on recent advances in coreference resolution. *Artif. Intell. Rev.* **2023**, *56*, 14439–14481. [CrossRef]
44. Poumay, J.; Ittoo, A. A Comprehensive Comparison of Word Embeddings in Event & Entity Coreference Resolution. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2021, Punta Cana, Dominican Republic, 7–11 November 2021; pp. 2755–2764.
45. Charton, E.; Gagnon, M. Poly-co: A multilayer perceptron approach for coreference detection. In Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task, Portland, OR, USA, 23–24 June 2011; pp. 97–101.

46. Lee, K.; He, L.; Lewis, M.; Zettlemoyer, L. End-to-end Neural Coreference Resolution. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 7–11 September 2017; pp. 188–197. [CrossRef]
47. Wiseman, S.; Rush, A.M.; Shieber, S.M. Learning Global Features for Coreference Resolution. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, CA, USA, 12–17 June 2016; pp. 994–1004. [CrossRef]
48. Lee, K.; He, L.; Zettlemoyer, L. Higher-Order Coreference Resolution with Coarse-to-Fine Inference. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, LA, USA, 1–6 June 2018; Volume 2 (Short Papers), pp. 687–692. [CrossRef]
49. Joshi, M.; Levy, O.; Zettlemoyer, L.; Weld, D. BERT for Coreference Resolution: Baselines and Analysis. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; pp. 5803–5808. [CrossRef]
50. Joshi, M.; Chen, D.; Liu, Y.; Weld, D.S.; Zettlemoyer, L.; Levy, O. SpanBERT: Improving Pre-training by Representing and Predicting Spans. *Trans. Assoc. Comput. Linguist.* **2020**, *8*, 64–77. [CrossRef]
51. Bohnet, B.; Alberti, C.; Collins, M. Coreference Resolution through a seq2seq Transition-Based System. *Trans. Assoc. Comput. Linguist.* **2023**, *11*, 212–226. [CrossRef]
52. Gandhi, N.; Field, A.; Tsvetkov, Y. Improving Span Representation for Domain-adapted Coreference Resolution. In Proceedings of the Fourth Workshop on Computational Models of Reference, Anaphora and Coreference, Punta Cana, Dominican Republic, 7 November 2021; pp. 121–131.
53. Lee, J.; Yoon, W.; Kim, S.; Kim, D.; Kim, S.; So, C.H.; Kang, J. BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics* **2019**, *36*, 1234–1240. [CrossRef]
54. Trieu, H.L.; Nguyen, N.T.H.; Miwa, M.; Ananiadou, S. Investigating Domain-Specific Information for Neural Coreference Resolution on Biomedical Texts. In Proceedings of the BioNLP 2018 Workshop, Melbourne, Australia, 19 July 2018; pp. 183–188. [CrossRef]
55. Webster, K.; Recasens, M.; Axelrod, V.; Baldridge, J. Mind the GAP: A Balanced Corpus of Gendered Ambiguous Pronouns. *Trans. Assoc. Comput. Linguist.* **2018**, *6*, 605–617. [CrossRef]
56. Agarwal, O.; Subramanian, S.; Nenkova, A.; Roth, D. Evaluation of named entity coreference. In Proceedings of the Second Workshop on Computational Models of Reference, Anaphora and Coreference, Minneapolis, MN, USA, 7 June 2019; pp. 1–7. [CrossRef]
57. Moosavi, N.S.; Strube, M. Which Coreference Evaluation Metric Do You Trust? A Proposal for a Link-based Entity Aware Metric. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Berlin, Germany, 7–12 August 2016; pp. 632–642. [CrossRef]
58. Liu, K.; Chen, Y.; Liu, J.; Zuo, X.; Zhao, J. Extracting Events and Their Relations from Texts: A Survey on Recent Research Progress and Challenges. *AI Open* **2020**, *1*, 22–39. [CrossRef]
59. Wang, H.; Lu, G.; Yin, J.; Qin, K. Relation Extraction: A Brief Survey on Deep Neural Network Based Methods. In Proceedings of the 2021 The 4th International Conference on Software Engineering and Information Management, Yokohama, Japan, 16–18 January 2021; pp. 220–228. [CrossRef]
60. Wang, L.; Cao, Z.; de Melo, G.; Liu, Z. Relation Classification via Multi-Level Attention CNNs. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Berlin, Germany, 7–12 August 2016; pp. 1298–1307. [CrossRef]
61. Li, Y. The Combination of CNN, RNN, and DNN for Relation Extraction. In Proceedings of the 2021 2nd International Conference on Computing and Data Science (CDS), Stanford, CA, USA, 28–29 January 2021; pp. 585–590. [CrossRef]
62. Smirnova, A.; Cudré-Mauroux, P. Relation Extraction Using Distant Supervision: A Survey. *ACM Comput. Surv.* **2018**, *51*, 106. [CrossRef]
63. Rathore, V.; Badola, K.; Mausam; Singla, P. A Simple, Strong and Robust Baseline for Distantly Supervised Relation Extraction. *arXiv* **2021**, arXiv:2110.07415.
64. Wu, S.; He, Y. Enriching Pre-trained Language Model with Entity Information for Relation Classification. *arXiv* **2019**, arXiv:1905.08284.
65. Yi, R.; Hu, W. Pre-Trained BERT-GRU Model for Relation Extraction. In Proceedings of the 2019 8th International Conference on Computing and Pattern Recognition, Beijing, China, 23–25 October 2019; pp. 453–457. [CrossRef]
66. Huguet Cabot, P.L.; Navigli, R. REBEL: Relation Extraction By End-to-end Language generation. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2021, Punta Cana, Dominican Republic, 16–20 November 2021; pp. 2370–2381.
67. Baldini Soares, L.; FitzGerald, N.; Ling, J.; Kwiatkowski, T. Matching the Blanks: Distributional Similarity for Relation Learning. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 2895–2905. [CrossRef]
68. Nadgeri, A.; Bastos, A.; Singh, K.; Mulang, I.O.; Hoffart, J.; Shekarpour, S.; Saraswat, V. KGPool: Dynamic Knowledge Graph Context Selection for Relation Extraction. In Proceedings of the Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, Online, 1–6 August 2021; pp. 535–548. [CrossRef]

69. Xu, B.; Wang, Q.; Lyu, Y.; Zhu, Y.; Mao, Z. Entity Structure Within and Throughout: Modeling Mention Dependencies for Document-Level Relation Extraction. *arXiv* **2021**, arXiv:2102.10249.
70. Ma, Y.; Wang, A.; Okazaki, N. DREEAM: Guiding Attention with Evidence for Improving Document-Level Relation Extraction. In Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, Dubrovnik, Croatia, 2–6 May 2023; pp. 1971–1983. [CrossRef]
71. Zhang, K.; Jimenez Gutierrez, B.; Su, Y. Aligning Instruction Tasks Unlocks Large Language Models as Zero-Shot Relation Extractors. In Proceedings of the Findings of the Association for Computational Linguistics: ACL 2023, Toronto, ON, Canada, 9–14 July 2023; pp. 794–812. [CrossRef]
72. Sainz, O.; García-Ferrero, I.; Agerri, R.; de Lacalle, O.L.; Rigau, G.; Agirre, E. GoLLIE: Annotation Guidelines improve Zero-Shot Information-Extraction. *arXiv* **2023**, arXiv:2310.03668.
73. Zhang, Y.; Lin, H.; Yang, Z.; Wang, J.; Sun, Y.; Xu, B.; Zhao, Z. Neural network-based approaches for biomedical relation classification: A review. *J. Biomed. Inform.* **2019**, *99*, 103294. [CrossRef]
74. Di, S.; Shen, Y.; Chen, L. Relation Extraction via Domain-Aware Transfer Learning. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 1348–1357. [CrossRef]
75. Nasar, Z.; Jaffry, S.W.; Malik, M. Named Entity Recognition and Relation Extraction: State of the Art. *ACM Comput. Surv.* **2021**, *54*, 20. [CrossRef]
76. Nayak, T.; Ng, H.T. Effective Modeling of Encoder-Decoder Architecture for Joint Entity and Relation Extraction. *arXiv* **2019**, arXiv:1911.09886.
77. Zaporojets, K.; Deleu, J.; Jiang, Y.; Demeester, T.; Develder, C. Towards Consistent Document-level Entity Linking: Joint Models for Entity Linking and Coreference Resolution. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Dublin, Ireland, 22–27 May 2022; pp. 778–784. [CrossRef]
78. Han, X.; Gao, T.; Lin, Y.; Peng, H.; Yang, Y.; Xiao, C.; Liu, Z.; Li, P.; Zhou, J.; Sun, M. More Data, More Relations, More Context and More Openness: A Review and Outlook for Relation Extraction. In Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, Suzhou, China, 4–7 December 2020; pp. 745–758.
79. Taghizadeh, N.; Faili, H. Cross-lingual transfer learning for relation extraction using Universal Dependencies. *Comput. Speech Lang.* **2022**, *71*, 101265. [CrossRef]
80. Chen, Y.; Sun, Y.; Yang, Z.; Lin, H. Joint Entity and Relation Extraction for Legal Documents with Legal Feature Enhancement. In Proceedings of the 28th International Conference on Computational Linguistics, Barcelona, Spain, 8–13 December 2020; pp. 1561–1571. [CrossRef]
81. Pikuliak, M.; Šimko, M.; Bieliková, M. Cross-lingual learning for text processing: A survey. *Expert Syst. Appl.* **2021**, *165*, 113765. [CrossRef]
82. Yu, H.; Mao, X.; Chi, Z.; Wei, W.; Huang, H. A Robust and Domain-Adaptive Approach for Low-Resource Named Entity Recognition. In Proceedings of the 2020 IEEE International Conference on Knowledge Graph (ICKG), Nanjing, China, 9–11 August 2020; pp. 297–304. [CrossRef]
83. Pires, T.; Schlinger, E.; Garrette, D. How Multilingual is Multilingual BERT? In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 4996–5001. [CrossRef]
84. Liang, S.; Gong, M.; Pei, J.; Shou, L.; Zuo, W.; Zuo, X.; Jiang, D. Reinforced Iterative Knowledge Distillation for Cross-Lingual Named Entity Recognition. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Singapore, 14–18 August 2021; pp. 3231–3239. [CrossRef]
85. Zhou, S.; Rijhwani, S.; Neubig, G. Towards Zero-resource Cross-lingual Entity Linking. In Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019), Hong Kong, China, 3 November 2019; pp. 243–252. [CrossRef]
86. Eisenschlos, J.; Ruder, S.; Czapla, P.; Kadras, M.; Gugger, S.; Howard, J. MultiFiT: Efficient Multi-lingual Language Model Fine-tuning. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; pp. 5702–5707. [CrossRef]
87. Bitew, S.K.; Deleu, J.; Develder, C.; Demeester, T. Lazy Low-Resource Coreference Resolution: A Study on Leveraging Black-Box Translation Tools. In Proceedings of the Fourth Workshop on Computational Models of Reference, Anaphora and Coreference, Punta Cana, Dominican Republic, 11 November 2021; pp. 57–62.
88. Pires, T.; Schlinger, E.; Garrette, D. How multilingual is Multilingual BERT? *arXiv* **2019**, arXiv:1906.01502.
89. Zheng, L.; Guha, N.; Anderson, B.R.; Henderson, P.; Ho, D.E. When Does Pretraining Help? Assessing Self-Supervised Learning for Law and the CaseHOLD Dataset of 53,000+ Legal Holdings. In Proceedings of the Eighteenth International Conference on Artificial Intelligence and Law, São Paulo, Brazil, 21–25 June 2021; Association for Computing Machinery: New York, NY, USA, 2021; pp. 159–168.
90. Muraoka, M.; Bhattacharjee, B.; Merler, M.; Blackwood, G.; Li, Y.; Zhao, Y. Cross-Lingual Transfer of Large Language Model by Visually-Derived Supervision toward Low-Resource Languages. In Proceedings of the 31st ACM International Conference on Multimedia, Ottawa, ON, Canada, 29 October–3 November 2023; pp. 3637–3646. [CrossRef]
91. Surís, D.; Epstein, D.; Vondrick, C. Globetrotter: Connecting languages by connecting images. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–24 June 2022; pp. 16474–16484.

92. Krasadakis, P.; Sakkopoulos, E.; Verykios, V.S. A Natural Language Processing Survey on Legislative and Greek Documents. In Proceedings of the 25th Pan-Hellenic Conference on Informatics, Volos, Greece, 26–28 November 2021; pp. 407–412. [CrossRef]
93. Leitner, E.; Rehm, G.; Moreno-Schneider, J. Fine-Grained Named Entity Recognition in Legal Documents. In Proceedings of the Semantic Systems: The Power of AI and Knowledge Graphs, Karlsruhe, Germany, 9–12 September 2019; pp. 272–287.
94. Darji, H.; Mitrović, J.; Granitzer, M. German BERT Model for Legal Named Entity Recognition. In Proceedings of the 15th International Conference on Agents and Artificial Intelligence: SCITEPRESS—Science and Technology Publications, Lisbon, Portugal, 22–24 February 2023. [CrossRef]
95. Krasadakis, P.; Sinos, E.; Verykios, V.S.; Sakkopoulos, E. Efficient Named Entity Recognition on Greek Legislation. In Proceedings of the 2022 13th International Conference on Information, Intelligence, Systems and Applications (IISA), Corfu, Greece, 18–20 July 2022; pp. 1–8. [CrossRef]
96. Donnelly, J.; Roegiest, A. The Utility of Context When Extracting Entities from Legal Documents. In Proceedings of the 29th ACM International Conference on Information and Knowledge Management, Virtual Event, 19–23 October 2020; pp. 2397–2404. [CrossRef]
97. Gordon, T.F. An Overview of the Legal Knowledge Interchange Format. In Proceedings of the Business Information Systems Workshops, Berlin, Germany, 3–5 May 2010; pp. 240–242.
98. Avgerinos Loutsaris, M.; Lachana, Z.; Alexopoulos, C.; Charalabidis, Y. Legal Text Processing: Combing Two Legal Ontological Approaches through Text Mining. In Proceedings of the DG.O2021: The 22nd Annual International Conference on Digital Government Research, Omaha, NE, USA, 9–11 June 2021; pp. 522–532. [CrossRef]
99. Cardellino, C.; Teruel, M.; Alemany, L.A.; Villata, S. A Low-Cost, High-Coverage Legal Named Entity Recognizer, Classifier and Linker. In Proceedings of the 16th Edition of the International Conference on Articial Intelligence and Law, London, UK, 12–16 June 2017; pp. 9–18. [CrossRef]
100. Gupta, A.; Verma, D.; Pawar, S.; Patil, S.; Hingmire, S.; Palshikar, G.K.; Bhattacharyya, P. Identifying Participant Mentions and Resolving Their Coreferences in Legal Court Judgements. In Proceedings of the TSD, Brno, Czech Republic, 11–14 September 2018.
101. Ji, D.; Gao, J.; Fei, H.; Teng, C.; Ren, Y. A deep neural network model for speakers coreference resolution in legal texts. *Inf. Process. Manag.* **2020**, *57*, 102365. [CrossRef]
102. Dragoni, M.; Villata, S.; Rizzi, W.; Governatori, G. Combining NLP Approaches for Rule Extraction from Legal Documents. *AI Approaches to the Complexity of Legal Systems*; Springer: Cham, Switzerland, 2018. [CrossRef]
103. Sunkle, S.; Kholkar, D.; Kulkarni, V. Comparison and Synergy between Fact-Orientation and Relation Extraction for Domain Model Generation in Regulatory Compliance. In Proceedings of the 35th International Conference ER, Gifu, Japan, 14–17 November 2016.
104. Filtz, E.; Navas-Loro, M.; Santos, C.; Polleres, A.; Kirrane, S. Events matter: Extraction of events from court decisions. *Leg. Knowl. Inf. Syst.* **2020**, *334*, 33–42. [CrossRef]
105. Li, Q.; Zhang, Q.; Yao, J.; Zhang, Y. Event Extraction for Criminal Legal Text. In Proceedings of the 2020 IEEE International Conference on Knowledge Graph (ICKG), Nanjing, China, 9–11 August 2020; pp. 573–580. [CrossRef]
106. Savelka, J.; Westermann, H.; Benyekhlef, K. Cross-Domain Generalization and Knowledge Transfer in Transformers Trained on Legal Data. *arXiv* **2021**, arXiv:2112.07870.
107. JOHN, A.K. Multilingual legal information retrieval system for mapping recitals and normative provisions. In Proceedings of the Legal Knowledge and Information Systems: JURIX 2020: The Thirty-Third Annual Conference, Brno, Czech Republic, 9–11 December 2020; IOS Press: Amsterdam, The Netherlands, 2020; Volume 334, p. 123.
108. Niklaus, J.; Matoshi, V.; Rani, P.; Galassi, A.; Stürmer, M.; Chalkidis, I. LEXTREME: A Multi-Lingual and Multi-Task Benchmark for the Legal Domain. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, 6–10 December 2023; pp. 3016–3054. [CrossRef]
109. Niklaus, J.; Matoshi, V.; Stürmer, M.; Chalkidis, I.; Ho, D.E. MultiLegalPile: A 689GB Multilingual Legal Corpus. *arXiv* **2023**, arXiv:2306.02069.
110. Chalkidis, I.; Garneau, N.; Goanta, C.; Katz, D.; Søgaard, A. LeXFiles and LegalLAMA: Facilitating English Multinational Legal Language Model Development. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Toronto, ON, Canada, 9–14 July 2023; pp. 15513–15535. [CrossRef]
111. Chalkidis, I. ChatGPT may Pass the Bar Exam soon, but has a Long Way to Go for the LexGLUE benchmark. *arXiv* **2023**, arXiv:2304.12202.
112. Kingston, J. Using Artificial Intelligence to Support Compliance with the General Data Protection Regulation. *Artif. Intell. Law* **2017**, *25*, 429–443. [CrossRef]
113. Hamdani, R.E.; Mustapha, M.; Amariles, D.R.; Troussel, A.; Meeùs, S.; Krasnashchok, K. A Combined Rule-Based and Machine Learning Approach for Automated GDPR Compliance Checking. In Proceedings of the Eighteenth International Conference on Artificial Intelligence and Law, São Paulo, Brazil, 21–25 June 2021; pp. 40–49. [CrossRef]

electronics

MDPI

Article

Plato's Shadows in the Digital Cave: Controlling Cultural Bias in Generative AI

Kostas Karpouzis

Department of Communication, Media and Culture, Panteion University of Social and Political Sciences, 17671 Athens, Greece; kkarpou@panteion.gr

Abstract: Generative Artificial Intelligence (AI) systems, like ChatGPT, have the potential to perpetuate and amplify cultural biases embedded in their training data, which are predominantly produced by dominant cultural groups. This paper explores the philosophical and technical challenges of detecting and mitigating cultural bias in generative AI, drawing on Plato's Allegory of the Cave to frame the issue as a problem of limited and distorted representation. We propose a multifaceted approach combining technical interventions, such as data diversification and culturally aware model constraints, with a deeper engagement with the cultural and philosophical dimensions of the problem. Drawing on theories of extended cognition and situated knowledge, we argue that mitigating AI biases requires a reflexive interrogation of the cultural contexts of AI development and a commitment to empowering marginalized voices and perspectives. We claim that controlling cultural bias in generative AI is inseparable from the larger project of promoting equity, diversity, and inclusion in AI development and governance. By bridging philosophical reflection with technical innovation, this paper contributes to the growing discourse on responsible and inclusive AI, offering a roadmap for detecting and mitigating cultural biases while grappling with the profound cultural implications of these powerful technologies.

Keywords: ethics; bias; culture; diversity; fairness; societal impact; generative AI; training data

1. Introduction

When it comes to contemporary technology, Generative Artificial Intelligence (GenAI) systems such as ChatGPT ver. 4 are not just tools or innovations; they can be thought of as windows into the collective human intellect, mirroring and magnifying the breadth of our knowledge and the depth of our creativity. However, as with any mirror, the image reflected is subject to the conditions of its environment—in this case, the datasets that form the basis of AI training. These datasets are overwhelmingly influenced by dominant cultural narratives, resulting in a skewed representation of global diversity. The implications of this cultural bias in AI are profound, echoing the timeless themes of perception and reality as depicted in Plato's Allegory of the Cave [1]. Here, prisoners interpret shadows as the only reality they know, not unlike how AI systems, trained on unrepresentative data, project a distorted view of the world [2]. This allegory serves as a powerful metaphor for understanding the limitations and potential misrepresentations AI systems can perpetuate, highlighting the importance of critically examining the data that feed these digital entities, as well as the algorithms that power their training and deployment. A noticeable manifestation of the cave metaphor when it comes to recommending content from social media to users has to do with the "echo chamber" phenomenon [3], where users are mostly presented with content which matches their interests, preferences, and political, societal, and cultural views, missing out on other voices and sacrificing the neutrality of the medium in the process. In this context, social media users would be seen as the "prisoners" of a social media cave which allows for a limited or distorted view of the real world, by filtering available information according to its own needs.

Even a quick look into the mechanics of AI training makes evident that these systems are not merely technical constructs, but also repositories of human expression and, consequently, human biases. The training process involves feeding a vast amount of text data into algorithms, allowing them to learn patterns of language and thought. However, these data are not a neutral, objective collection of information. They are, instead, a reflection of the cultures that have the means and inclination to digitize and disseminate their knowledge and viewpoints. The result is a digital echo chamber where dominant cultures are amplified, and minority voices are diminished or entirely absent. A prominent example of this issue was the 2016 incident (The Verge, Twitter taught Microsoft's AI chatbot to be a racist a-hole in less than a day; https://www.theverge.com/2016/3/24/11297050/tay-microsoft-chatbot-racist; last accessed: 6 April 2024) with Microsoft's Tay chatbot, released on Twitter, and meant to interact with users by replying to them and learning from their input: the chatbot was maneuvered by users to "assimilate the internet's worst tendencies into its personality" and start replying with racist and offensive responses within just a few hours. Another example of existing biases "contaminating" an AI system was that of the prediction algorithm employed by the U.S. medical system to predict the costs related to hospitalization for prospective patients, based on their symptoms, prognosis, and background information [4]: here, the algorithm would assign a lower risk score to African-American patients for the same illness, history, and general condition as white patients, resulting in them being more likely to pay more for emergency care visits or not qualify for extra care as much as white patients with the same needs.

This disparity raises critical questions about the cultural implications of AI. How does the over-representation of certain cultures in AI training data affect the outputs of these systems? What are the consequences of such biases on global communication, cultural understanding, and representation? These questions are not merely academic; they have real-world implications for how AI is used and perceived in various sectors, from personalized and adaptive education [5] to gamification and entertainment [6], from policy making to personal interaction. Moreover, the issue of cultural bias in AI is a multidimensional one, intersecting with broader societal and ethical considerations. As Austin and Williams [7] discuss in their exploration of shame and necessity in classical ethics, the ethical dilemmas we face today with AI are not just about the technology itself but about the societal norms and values that it reflects and reinforces. In the context of AI, this involves grappling with the moral responsibility of ensuring that these systems are not just technologically advanced but also culturally sensitive and inclusive.

To tackle these challenges, this paper proposes a comprehensive approach. The first step is an in-depth analysis of the extent and nature of cultural biases in GenAI. This involves examining the content and sources of training datasets, evaluating the algorithms used for learning and generating content, and assessing the cultural representativeness of AI outputs. This analysis aims to map the contours of bias, identifying both overt and subtle forms of cultural dominance and exclusion. The second step focuses on methodologies for assessing cultural representation in AI-generated content. This encompasses a range of techniques, from computational methods such as sentiment analysis and bias detection algorithms [8] to qualitative approaches like content analysis and case studies. The goal here is to develop robust, multidimensional metrics for evaluating the cultural fairness of AI systems [9]. Finally, the paper advocates for proactive strategies to guide AI towards greater cultural neutrality. This includes diversifying training datasets to better reflect the rich tapestry of global cultures, implementing ethical guidelines for AI development, and fostering an environment of continuous monitoring and improvement. These strategies aim not only to mitigate existing biases but also to lay the groundwork for AI systems that are inherently more inclusive and representative.

In weaving together these various strands, the paper draws upon interdisciplinary perspectives, from the moral psychology of AI [10] to philosophical discussions on ethics [6] and representation. It seeks to contribute a nuanced, holistic view of the challenges and opportunities presented by cultural bias in AI, offering insights that are relevant to

technologists, ethicists, and policymakers alike. Therefore, this paper posits that to truly realize the potential of GenAI in reflecting and respecting the diversity of human cultures, a concerted, multidisciplinary effort is required. This effort must encompass not only technological advancements but also philosophical introspection and ethical commitment, aiming to understand, address, and ultimately transcend the biases inherent in AI training data. Through this approach, we can envision and work towards a future where AI becomes truly inclusive and a true reflection of the diverse tapestry of human experience. Section 2 discusses concepts from Plato's and Aristotle's writings, relevant to the ethical and philosophical discussion of AI, while Section 3 focuses on the idea of the cave, holding users captive in an alternate reality where they are presented with only filtered information about the real world; Section 4 proposes strategies and algorithmic means to mitigate the disparity between the real world and the "world model" resulting after training an AI system. Then, Section 5 discusses what is needed for a fairer design, training, and deployment of artificially intelligent systems in everyday life, especially when it comes to recognizing the need for interdisciplinary thinking and collaboration. In the light of these suggestions, Section 6 discusses broader, contemporary thinking on AI and its biases, while Section 7 adds the human needs to improve and achieve personal goals to that discussion by referring to Aristotle's concept of "eudaimonia". Finally, Section 8 concludes the paper by revisiting important concepts and deliberating about the limitations of this work.

2. Philosophical Perspectives

The exploration of cultural biases in GenAI finds a profound parallel in ancient philosophical thought, particularly in Plato's allegory of the cave, a centerpiece in his seminal work "The Republic" [1]. This allegory is not just a metaphor for the human condition but also resonates strikingly with the current challenges in AI. Plato describes a group of prisoners chained in a cave, all their lives, facing a blank wall, watching shadows projected on the wall by things passing in front of a fire behind them. These shadows are the closest they come to viewing reality. This scenario is emblematic of the situation with modern AI systems: like the prisoners, they are limited to the 'shadows' of data they are exposed to, often skewed by dominant cultural narratives. This restricted exposure leads to a constrained and distorted view of the world, mirroring the prisoners' perception of shadows as the complete reality [2].

Expanding upon this, the allegory serves as a compelling framework for understanding the limitations of AI in comprehending and representing the diverse spectrum of human experience. Just as the prisoners in the cave mistake the shadows for reality, AI systems might also misconstrue the biased representations in their training datasets for the full expanse of human culture and expression. This parallel underscores a significant philosophical inquiry: can AI ever transcend its 'cave' of biased data to perceive and reflect a more accurate and holistic view of the human condition?

Furthermore, this interpretation opens the door to other philosophical concepts that are pertinent to the discourse on AI and cultural bias. For instance, the concept of phenomenology, which explores the structures of experience and consciousness, can offer insights into how AI interprets and interacts with human cultural expressions. If AI's 'consciousness' is shaped by limited and biased data, its 'experience' of the world is inherently constrained, akin to the limited perception of the cave's prisoners. Moreover, the ethical implications of these biases draw parallels with Aristotle's virtue ethics [11], a philosophy that emphasizes the role of character and virtue in moral philosophy. Just as virtue ethics advocates for moral character above all else, the development of AI systems must prioritize ethical considerations and cultural sensitivities above mere technical efficacy. This perspective aligns with the unity of virtues discussed by Wolf [10], suggesting that AI systems should be designed with an integrated approach that considers technical proficiency, ethical integrity, and cultural awareness.

The challenge, then, would be to lead AI out of the allegorical cave and into the light of a more nuanced and comprehensive understanding of the world. This entails a

reexamination of the data that feed AI systems, ensuring they encompass a more diverse array of cultural narratives. It also calls for a philosophical introspection into the values and assumptions underpinning AI development, ensuring they align with ethical principles that reflect a respect for the diversity of human cultures and experiences. In essence, addressing cultural bias in AI is not merely a technical fix; it is a philosophical endeavor that requires us to rethink the very nature of AI development and its interaction with human culture. By grounding AI in a philosophy that values diversity, inclusivity, and ethical integrity, we can guide these systems out of the shadows of biased data, leading them towards a richer, more representative understanding of the complex tapestry of human existence.

3. The Digital Cave: How Training Data Shape Generated Content

In addressing the issue of cultural bias in AI training data, we confront a critical aspect of AI development: the selection and composition of datasets used to train models like ChatGPT. These datasets are foundational to how AI systems learn and, subsequently, how they interpret and interact with the world. The heart of the problem lies in the disproportionate influence of dominant cultures within these datasets, leading to an over-representation of specific cultural perspectives and a marginalization of others.

The datasets used to train AI models are often culled from the internet, including websites, books, news articles, and informal and unmediated forms of digital media, such as social media content. However, the content in these datasets does not constitute a balanced representation of global cultures, but is predominantly created and consumed by a fraction of the world's population, primarily those from more technologically developed and digitally active regions. This skew results in an over-representation of the languages, values, and viewpoints of these dominant cultures. For instance, a substantial portion of internet content, and by extension, many AI training datasets, is in English. This linguistic dominance extends beyond mere numbers; it carries with it the cultural contexts, idioms, and perspectives prevalent in English-speaking regions, even visual forms found in the Western world. Similarly, other dominant languages and cultures exhibit a similar influence. The result is a digital landscape where certain cultural narratives are amplified, while others are barely audible.

A statistical analysis of the data sources used in training major AI models would likely reveal this imbalance. By quantifying the languages, regions, and types of content that are most prevalent in these datasets, we can gain a clearer picture of the cultural biases inherent in AI training. This analysis could involve evaluating the distribution of languages, the geographical origins of web content, and the thematic concentration of the data, among other factors. Furthermore, a qualitative analysis would complement this statistical approach. Examining the types of narratives, stories, and perspectives that are over-represented can provide insights into the subtler aspects of cultural bias. For example, certain cultural norms and values might be consistently portrayed in certain ways, reinforcing stereotypes or marginalizing alternative viewpoints.

The implications of this bias are multifaceted. On a technical level, AI systems trained on such data are likely to exhibit a skewed understanding of language and culture [12]. This can manifest in various ways, from the types of responses generated by a chatbot to the cultural references and examples used by an AI tutor. On a societal level, the over-representation of certain cultures in AI outputs can reinforce existing power dynamics, marginalizing already under-represented groups. Moreover, the issue of cultural bias in AI training data is not just about the quantity of representation but also the quality. It is not sufficient to merely increase the volume of data from under-represented cultures; it is equally important to ensure that these data are contextually rich, diverse, and authentic [13]. This requires a concerted effort to diversify data sources, engaging with communities and content creators from a wide range of cultural backgrounds.

In conclusion, examining the cultural bias in AI training data reveals a landscape where dominant cultures disproportionately influence AI models. Addressing this imbalance requires both statistical and qualitative analyses of training datasets and a committed

effort to diversify and enrich these datasets. In this way, we can move towards developing AI systems that are truly representative of the global diversity of human cultures and experiences, thus fostering a more inclusive digital future.

4. Escaping the Cave: Techniques for Detecting Cultural Biases

In addressing the critical issue of assessing cultural representation in AI-generated content, we can utilize robust methodologies to effectively evaluate and illuminate the biases inherent in these systems. These methodologies encompass a blend of quantitative and qualitative approaches, each offering unique insights into the nuances of cultural bias and representation.

1. Sentiment analysis: To detect bias in emotional tone, sentiment analysis can be a valuable tool, instrumental in ensuring that training datasets for AI are balanced in terms of sentiment, and preventing the perpetuation of stereotypes or biases associated with certain groups. By examining text corpora used for training, sentiment analysis can reveal if specific demographics are linked to predominantly negative or positive sentiments, allowing for dataset adjustments to foster neutrality in AI responses. For instance, VADER (Valence Aware Dictionary and Sentiment Reasoner), introduced by Hutto and Gilbert ([14]), is adept at parsing social media sentiment; adapting such tools to analyze sentiments in AI-generated content could reveal biases toward certain cultural groups, as these algorithms can detect subtleties in emotional expression associated with different cultures. However, while invaluable in contexts like customer service and content moderation, sentiment analysis faces its own challenges, such as capturing the nuances of language and cultural sentiment expressions.
2. Language and dialect recognition: Assessing AI systems on their capability to accurately recognize and respond to a variety of languages and dialects is fundamental. Jurgens et al. [15] provide insight into this area by highlighting the challenges AI faces in adapting to language variations, a crucial aspect for ensuring cultural inclusivity in AI systems.
3. Diversity metrics: Implementing diversity metrics allows for the quantitative assessment of the range and inclusivity of cultural references in AI-generated content. Zehlike et al. [16] discuss 'diversity in information retrieval', a concept that can be adapted to AI, ensuring that the output reflects a broad spectrum of cultural perspectives. In their work, they balance the goal of selecting the "best" candidates with ensuring fair representation of protected groups, proposing an efficient algorithm that produces rankings maintaining ranked group fairness as long as there are enough candidates in the protected group. This research highlights key considerations in applying diversity metrics in AI, such as balancing fairness with utility, using statistical tests to ensure fairness, and considering legal and ethical frameworks.

Qualitative assessment of cultural representation in AI-generated content requires a nuanced approach that delves into the subjective and interpretive aspects of human culture. Such methodologies are indispensable in uncovering the subtler, more intricate manifestations of cultural bias that may elude purely quantitative analyses, and emphasize understanding the depth, context, and meaning behind AI-generated content, providing insights into how different cultures are represented, perceived, and potentially stereotyped by AI systems. By engaging in content analysis, ethnographic studies, narrative analysis, and critical discourse analysis, researchers can explore the complexities of cultural narratives embedded within AI outputs, unraveling the layers of cultural nuances and biases.

1. Content analysis: This approach involves a detailed examination of AI-generated content to identify biases and stereotypes. Noble's analysis of search engines in perpetuating cultural stereotypes provides a methodological framework that can be adapted for AI content [17]. By scrutinizing the types of narratives and representations produced by AI, researchers can unearth subtle biases and dominant cultural themes.

2. Case studies: Conducting case studies offers an in-depth view of specific instances where AI systems may exhibit bias. A notable case is Garcia's analysis of Google's photo-tagging algorithm [18], which misidentified African Americans. Such case studies can highlight significant flaws in AI algorithms and underscore the importance of cultural sensitivity in AI development.
3. Ethnographic studies: Ethnographic research, as exemplified by the work of Barocas and Selbst [19], delves into user interactions with AI systems, shedding light on how cultural nuances are processed by AI. This approach allows for a more comprehensive understanding of the user experience, especially in diverse cultural contexts.
4. Narrative analysis: Beyond content analysis, narrative analysis [20] offers a way to understand the stories and themes that AI generates, which can reflect cultural biases. This involves looking at the plotlines, character representations, and scenarios created by AI to discern any recurring cultural tropes or imbalances.
5. Critical discourse analysis (CDA): CDA, as applied in AI contexts, allows for an examination of the underlying power dynamics and ideologies within AI-generated text [21]. This method, drawing from Foucault's ideas on discourse and power, can reveal how AI may perpetuate dominant cultural narratives.

Employing a combination of these quantitative and qualitative methods, as encouraged by the interdisciplinary approach in Eubanks's [22] exploration of technology and societal intersections, provides a holistic view of cultural representation in AI. This comprehensive approach is essential to identify, understand, and address the multifaceted nature of cultural bias in AI systems.

In summary, assessing cultural representation in AI-generated content requires a multifaceted and interdisciplinary approach. The integration of both quantitative and qualitative methods, drawing on seminal works and research methodologies in sentiment analysis, language recognition, diversity metrics, content analysis, case studies, ethnographic studies, narrative analysis, and critical discourse analysis, equips researchers with a diverse set of tools to uncover and understand the complexities of cultural bias in AI. Such a comprehensive approach is important for developing AI systems that are not only technically advanced but also culturally aware and inclusive.

5. Guiding AI towards Cultural Neutrality

Guiding AI towards cultural neutrality involves a multifaceted approach, encompassing both technical and ethical strategies. The aim to create AI systems that do not favor or bias any particular culture or group involves creating models and algorithms that are fair and impartial, reflecting a wide range of human experiences and perspectives without being influenced by dominant cultural norms and values. Achieving cultural neutrality requires careful consideration of the diversity inherent in global cultures, including recognizing and respecting differences in language, customs, beliefs, and values. For AI systems, this essentially means being designed and trained on datasets that are diverse and representative of this global variety, ensuring that no single culture's perspectives or biases disproportionately influence the AI's behavior or outputs.

5.1. Diversifying Training Data

One of the primary strategies is diversifying the datasets used for training AI. This means including a wide range of data sources that better represent the variety of human cultures, languages, and experiences. For example, drawing on literature, media, and digital content from a broad spectrum of cultures can provide AI systems with a more balanced view of the world. Diversification is not without challenges, as it involves not only sourcing these diverse data but also ensuring that they are of high quality and contextually rich. Efforts in this direction should prioritize inclusivity and seek to cover under-represented groups, dialects, and cultural contexts.

5.2. Algorithmic Adjustments for Bias Recognition and Mitigation

Developing algorithms that can actively recognize and adjust for bias is another critical strategy, involving creating AI models that are not only capable of learning from data but also of identifying and correcting biases in those data. Techniques like bias detection algorithms are designed to identify and measure biases in AI systems, particularly in datasets and AI-generated content. These algorithms work by analyzing patterns, differences in success ratios and error rates across parts of the training data, or discrepancies that may indicate biased treatment of certain groups or topics. FairTest [23] is a prominent example that uncovers unwarranted associations in predictive models. For instance, if a job recommendation system disproportionately suggests certain professions based on gender, FairTest can help to identify and quantify this bias. Similarly, AI Fairness 360 (AIF360) [24], developed by IBM (Armonk, NY, USA), is an extensible toolkit that can detect, understand, and mitigate various forms of bias in machine learning models, and includes over 70 fairness metrics and 11 bias mitigation algorithms. AIF360 can be used, for example, to analyze a credit scoring model to ensure that it does not systematically disadvantage a particular racial or ethnic group.

In addition to this, regular audits of AI outputs and feedback loops allow AI systems to learn from their biases; for example, AI systems can be designed to flag when their outputs are disproportionately representing certain cultures or viewpoints, prompting a reevaluation and adjustment of the training data.

5.3. AI, Agency, and Ethics

In Plato's philosophy, "forms" represent the perfect, eternal, and unchanging essences that underlie the imperfect and transient objects of the material world, making them the ultimate source of knowledge and truth, while the physical world is merely a shadow or imitation of these ideal entities. Applying this concept to AI, one could argue that the training data and algorithms that shape models like ChatGPT serve a similar role to Plato's forms. These ideal patterns and structures, derived from a vast amount of text data, provide the foundation for the model's ability to generate coherent and meaningful outputs. However, just as physical objects are imperfect copies of their ideal forms, the generated content of AI models is an approximation of the knowledge and patterns contained in the training data. This raises questions about the nature of the information produced by AI systems and the extent to which it can be considered true knowledge in the Platonic sense.

Aristotle's theory of four causes offers another framework for understanding the nature of AI systems. According to Aristotle, every object or being can be understood in terms of four essential causes: the material cause (the physical matter that constitutes the object), the formal cause (the form or structure that defines its essence), the efficient cause (the agent or force that brings the object into being), and the final cause (the purpose or end towards which the object is directed). In the context of AI models like ChatGPT, the material cause would encompass the hardware and software components that make up the system, while the formal cause would be the specific architecture and design of the model, such as the transformer-based neural network that enables its language processing capabilities. The efficient cause of ChatGPT would include the human developers who created and trained the model, as well as the computational processes that shape its behavior through exposure to a vast amount of data. The final cause, or purpose, of ChatGPT is to generate human-like text and assist users with a variety of tasks, from answering questions to providing creative inspiration.

The question of agency in AI systems is particularly relevant to Aristotle's concept of "teleology", which holds that objects and beings have inherent purposes or ends towards which they strive. While AI models like ChatGPT are not conscious agents with intentional goals, they are nonetheless designed and trained by humans to serve specific functions and purposes. This raises the question of whether these models can be said to possess a form of agency, even if it is not equivalent to human agency. Latour's actor-network theory [25] provides a useful perspective on this issue, suggesting that agency is not a

property inherent to humans alone, but rather emerges from the complex interactions and associations between human and nonhuman actors within a network. From this view, ChatGPT and other AI systems can be understood as nonhuman actors that exercise a form of agency through their ability to shape human knowledge, communication, and decision-making processes.

The ethical implications of AI systems are another area where the ideas of Plato and Aristotle can provide valuable insights. Plato's concept of the "tripartite soul" [26], which divides the human psyche into the rational, spirited, and appetitive parts, each with its corresponding virtues of wisdom, courage, and temperance, emphasizes the importance of balance and harmony in moral character. Aristotle's doctrine of the mean, which holds that virtue is a middle point between excess and deficiency, and his focus on practical wisdom (phronesis) as the ability to discern the right course of action in specific situations, also highlight the importance of ethics and moral reasoning in human life. As AI systems become increasingly sophisticated and integrated into various domains of human activity, it is crucial to consider how these technologies can be developed and deployed in ways that align with ethical principles and promote human flourishing. This requires ongoing interdisciplinary collaboration among researchers, developers, policymakers, and ethicists to ensure that AI systems are designed with transparency, accountability, and respect for human values.

Finally, the epistemological ideas of Plato and Aristotle can shed light on the nature of the knowledge generated by AI systems. Plato's famous allegory of the cave, which depicts the journey from illusion to enlightenment, and his distinction between true knowledge ("episteme") and mere opinion ("doxa"), invite us to question the reliability and truthfulness of AI-generated content. While AI models like ChatGPT can produce outputs that appear convincing and informative, it is important to recognize that this content is ultimately derived from patterns in the training data and may not constitute genuine understanding or wisdom in the Platonic sense. Aristotle's emphasis on empirical observation and inductive reasoning, which laid the foundations for the scientific method, can be seen as a precursor to the data-driven approach used in modern AI research. However, the limitations and biases inherent in the data used to train AI models also highlight the need for critical evaluation and the recognition that machine-generated knowledge is not infallible.

5.4. Interdisciplinary Collaboration

An interdisciplinary approach, combining insights from fields such as sociology, anthropology, linguistics, and ethics, is vital in this endeavor. Collaboration between technologists, ethicists, cultural scholars, and other experts can lead to more comprehensive strategies for achieving cultural neutrality in AI. This collaboration ensures that diverse perspectives are considered in every step of AI development, from dataset compilation to algorithm design. In this context, philosophers and social scientists can provide valuable conceptual frameworks and ethical guidance, drawing on the rich tradition of philosophical inquiry to illuminate the metaphysical, epistemological, and moral dimensions of AI, while technical experts can offer insights into the capabilities, limitations, and inner workings of AI systems, ensuring that philosophical reflections are grounded in a realistic understanding of the technology. However, as AI systems are increasingly integrated into various domains of human activity, from healthcare and education to finance and criminal justice, it is crucial to develop ethical guidelines and policies that govern their use. This would require close collaboration between ethicists, policymakers, legal experts, and AI practitioners to identify and address the moral challenges posed by AI, such as issues of fairness, transparency, accountability, and privacy. Here, ethicists can help articulate the fundamental values and principles that should guide the development and deployment of AI, while policymakers and legal experts can translate these principles into concrete regulations and governance frameworks.

Finally, in order to close the loop between policy making and application to the real world, an assessment of the social, economic, and cultural implications of AI would also

be essential, requiring collaboration among researchers from a wide range of disciplines, including computer science, psychology, sociology, economics, and anthropology. Interdisciplinary research teams can investigate the ways in which AI systems interact with and shape human behavior, social structures, and cultural norms, as well as the potential risks and benefits of these technologies for different communities and stakeholders. By combining quantitative and qualitative methods, interdisciplinary research can provide a comprehensive understanding of the impact of AI and inform the development of strategies for maximizing its benefits while mitigating its risks. Ethical guidelines can play a crucial role in this part of AI development: these guidelines should encompass principles like fairness, non-discrimination, transparency, and accountability. Organizations like the IEEE (P2976 XAI—Explainable AI Working Group; https://sagroups.ieee.org/2976/; last accessed: 14 March 2024) have already laid down principles for ethically aligned AI design, which can serve as a foundation for these guidelines. Ethical committees, comprising members from diverse cultural backgrounds, can oversee AI development projects to ensure these principles are adhered to. Moreover, continuous ethical training for AI developers and stakeholders can foster a culture of responsibility and awareness.

5.5. Community Engagement and Feedback

Engaging with communities from diverse cultural backgrounds is another key strategy. This can involve seeking feedback on AI outputs, understanding cultural nuances from community members, and even involving these communities in the data collection and model training processes. Such engagement ensures that AI development is not happening in a vacuum but is responsive to the needs and perspectives of a wide array of cultural groups.

It has to be noted that guiding AI towards cultural neutrality requires a concerted effort involving the diversification of training data, implementation of ethical guidelines, development of bias-aware algorithms, interdisciplinary collaboration, and active community engagement. These strategies, while challenging, are essential for creating AI systems that are fair, unbiased, and representative of the global diversity of cultures and experiences. Such an approach not only enhances the technological sophistication of AI systems but also ensures their ethical and cultural relevance in a rapidly evolving global society.

6. Broader Philosophical Implications

Reflecting on the broader philosophical implications of AI and cultural bias necessitates a deep dive into the realms of ethics, consciousness, and the very nature of intelligence and agency. These discussions intersect with longstanding philosophical debates on free will, determinism, and the nature of human thought, raising profound questions about the role and impact of AI in our lives.

For instance, the intersection of AI with concepts of free will and determinism presents a compelling paradox. On one hand, AI systems, including generative models like ChatGPT, operate within the confines of their programming and the data they are trained on. This raises the question: can AI ever exhibit free will, or are its outputs entirely deterministic, bound by the algorithms and data that govern its operations? This echoes wider philosophical inquiries, as explored by Dennett [27] in "Elbow Room: The Varieties of Free Will Worth Wanting", where the nature of free will in a deterministic universe is contemplated. In the context of AI, these discussions take on a new dimension, as we grapple with the idea of machines that can learn and adapt but within predetermined parameters. Similarly, as AI systems become more sophisticated, particularly in their ability to mimic human thought processes, we encounter ethical and philosophical questions about the nature of intelligence and consciousness. Turing's seminal paper "Computing Machinery and Intelligence" [28] initiates this discourse by questioning what it means for a machine to think. The development of AI that can not only process information but also generate new content and seemingly exhibit creativity challenges our understanding of

consciousness. Is AI's simulation of human thought merely a complex mimicry devoid of true understanding, or does it represent a new form of intelligence?

The ethical implications of creating machines that imitate human cognitive processes are vast and multifaceted. Bostrom [29] posits that the development of advanced AI raises concerns about control, safety, and the alignment of AI objectives with human values. The cultural biases inherent in AI systems add another layer to this ethical debate. If AI can perpetuate or even amplify cultural biases, what responsibilities do developers and users have to mitigate these biases and ensure that AI systems are aligned with ethical principles that respect cultural diversity and promote equity? In a similar context, the cultural biases in AI prompt us to reflect on AI as a mirror of human society. Harari suggests that AI systems, in their current form, reflect the values, biases, and priorities of the societies that create them [30], effectively raising questions about the extent to which AI can transcend these human-imposed limitations and whether it should be designed to do so.

As shown above, the philosophical implications of AI and cultural bias are profound and far-reaching. They compel us to question fundamental concepts, such as the nature of free will and determinism in the context of AI, the ethics of creating machines that mimic human cognition, and the broader societal reflections that AI reveals. These considerations underscore the importance of a thoughtful, ethically guided approach to AI development, one that is acutely aware of the philosophical ramifications of creating intelligent machines that both reflect and shape our cultural realities.

7. Discussion

As we have seen, the issue of cultural bias in GenAI is a complex and multifaceted one, with profound implications for how we understand and shape the role of these technologies in our world. From the technical challenges of detecting and mitigating bias in machine learning models, to the philosophical questions of agency, responsibility, and the nature of the self, this is an issue that cuts to the heart of our relationship with AI and its place in human society.

Throughout this exploration, a few key themes have emerged. First and foremost is the recognition that *AI is not a neutral or objective technology*, but is always deeply shaped by the cultural contexts and assumptions in which it is developed and deployed. The biases and blind spots of AI systems are not simply technical glitches to be fixed, but are reflective of deeper cultural and political asymmetries that must be confronted and transformed. This insight challenges us to move beyond narrow technical solutions and to engage in a deeper reckoning with the cultural and ethical implications of AI. It requires us to interrogate the cultural assumptions and power relations that shape technological development, and to actively work to include and empower diverse cultural voices and perspectives.

A second key theme is the importance of situating the development of AI within the universal context of *human flourishing* (Aristotle calls it "eudaimonia" in his Nicomachean Ethics treatise [11]) and *social justice*. The ultimate measure of success for AI is not just its technical sophistication or efficiency, but its ability to enrich and empower human life in all its diversity and complexity. This means attending to the concrete impacts of AI on marginalized and vulnerable communities, and taking responsibility for developing AI systems that serve their needs and contexts. Philosophically, this perspective is rooted in a recognition of the fundamental interdependence and contextuality of human life, and the need for an ethics of care that prioritizes empathy, compassion, and a respect for cultural difference. It suggests that the development of AI should be guided not just by abstract principles or aggregate outcomes, but by a deep attentiveness to the specific needs and contexts of the people and communities it serves.

A third key theme is the need for a *more expansive and imaginative vision* of the role of AI in human society. Too often, the discourse around AI is dominated by narrow technical or economic considerations, with little attention paid to the deeper human and cultural implications of these technologies. But as we have seen, AI has the potential to profoundly shape and transform the human experience in ways that go far beyond mere

efficiency or automation. Realizing this potential requires a willingness to think beyond narrow technical fixes and to imagine alternative futures that prioritize equity, inclusivity, and human flourishing. It requires a commitment to harnessing the power of AI not for domination or control, but for the emancipation and enrichment of the human spirit in all its diversity and potential.

In conclusion, the project of mitigating cultural bias in AI is inseparable from the larger project of building a more just and humane world. It is a project that requires not just technical expertise but moral imagination, not just computational power but empathic understanding. It is a project that challenges us to envision and create a world in which technology serves not just the interests of the powerful few, but the flourishing of all. As we continue to grapple with these challenges, it is essential that we keep this larger vision in mind. We must remember that the development of AI is not an end in itself, but a means to the larger end of promoting human well-being and social justice. We must be willing to interrogate and transform the cultural assumptions and power relations that shape technological development, and to imagine alternative futures that prioritize the flourishing of all.

8. Conclusions

The exploration of cultural biases in AI and the quest for cultural neutrality present a landscape rich in complexity, interwoven with technical challenges, ethical considerations, and profound philosophical questions. This paper has traversed the terrain of AI's development and application, scrutinizing the way dominant cultures shape AI training data, investigating methodologies to assess cultural representation, discussing strategies to guide AI towards cultural neutrality, and delving into the philosophical implications of AI and cultural bias.

However, it is crucial to acknowledge the limitations inherent in these discussions and the approaches we propose. One significant limitation is the current state of technology itself. Despite advancements in AI, the ability of these systems to fully comprehend and reflect the depth and nuance of human culture is still evolving. AI's understanding of context, subtlety, and the complexities of human languages and interactions remains a work in progress. Moreover, the methodologies for assessing cultural bias in AI, both quantitative and qualitative, have their constraints. Quantitative methods, while offering measurable insights, can overlook the subtleties that qualitative approaches capture. Conversely, qualitative methods, rich in depth, may lack the scalability and objectivity that quantitative analyses provide. Balancing these approaches remains a challenge and necessitates continuous refinement.

The strategies to mitigate cultural bias, such as diversifying training data and implementing ethical guidelines, also encounter practical and theoretical hurdles. The diversity of global cultures makes it a daunting task to represent them all adequately within AI datasets. Additionally, ethical guidelines, while imperative, must contend with varying interpretations of ethics across different cultures and societies. For example, the discussions around free will, consciousness, and the ethics of AI reveal more questions than answers. The debate on whether AI can truly exhibit free will or consciousness, or merely simulate them, remains unresolved. The ethical implications of AI's influence on society and culture continue to be a subject of intense debate and contemplation.

Despite these limitations, questioning these facets of AI and cultural bias is not only necessary but also immensely valuable, since it brings to light the intricacies of developing technology that is as unbiased and representative as possible. The dialogue between technology and culture, ethics and philosophy, highlights the need for a collaborative, multidisciplinary approach in AI development. In essence, this exploration underscores a fundamental truth: AI, in its current form and future potential, is a reflection of human society. It embodies our strengths, biases, aspirations, and limitations. As we continue to advance in AI technology, it is imperative that we do so with a mindful approach, one that

considers not just the technical possibilities but also the cultural, ethical, and philosophical dimensions that define our humanity.

Looking forward, the discourse on AI and cultural bias should evolve to include even broader perspectives, integrating insights from more diverse cultures and disciplines. The journey towards developing AI that truly understands and reflects the diversity of human experience is ongoing. It is a journey marked by challenges and opportunities, demanding continuous reflection, adaptation, and commitment to a future where technology and culture harmoniously coexist. In this pursuit, the limitations we encounter today serve not as deterrents but as catalysts for further research, innovation, and introspection, driving us towards a more inclusive, ethical, and culturally aware AI tomorrow.

Funding: This research received no external funding.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The author declares no conflicts of interest.

References

1. Plato. *The Allegory of the Cave*; P & L Publication: Brea, CA, USA , 2010.
2. Gagarin, M.; Nussbaum, M. The Fragility of Goodness: Luck and Ethics in Greek Tragedy and Philosophy. *Class. World* **1988**, *80*, 452. [CrossRef]
3. Lauer, D. Facebook's ethical failures are not accidental; they are part of the business model. *AI Ethics* **2021**, *1*, 395–403. [CrossRef] [PubMed]
4. Obermeyer, Z.; Powers, B.; Vogeli, C.; Mullainathan, S. Dissecting racial bias in an algorithm used to manage the health of populations. *Science* **2019**, *366*, 447–453. [CrossRef] [PubMed]
5. Karpouzis, K. Explainable AI for intelligent tutoring systems. In Proceedings of the International Conference on Frontiers of Artificial Intelligence, Ethics, and Multidisciplinary Applications, Athens, Greece, 25–26 September 2023; Springer: Berlin/Heidelberg, Germany, 2023; pp. 59–70.
6. Karpouzis, K. What would Plato say? Concepts and notions from Greek philosophy applied to gamification mechanics for a meaningful and ethical gamification. *arXiv* **2024**, arXiv:2403.08041.
7. Austin, N.; Williams, B. Shame and Necessity. *Class. World* **1993**, *116*, 137. [CrossRef]
8. Lee, N.T. Detecting racial bias in algorithms and machine learning. *J. Inf. Commun. Ethics Soc.* **2018**, *16*, 252–260.
9. Karpouzis, K.; Pantazatos, D.; Taouki, J.; Meli, K. Tailoring Education with GenAI: A New Horizon in Lesson Planning. *arXiv* **2024**, arXiv:2403.12071.
10. Wolf, S. Moral Psychology and the Unity of the Virtues. *Ratio* **2007**, *20*, 145–167. [CrossRef]
11. Crisp, R. *Aristotle: Nicomachean Ethics*; Cambridge University Press: Cambridge, UK, 2014.
12. Chakraborty, J.; Majumder, S.; Menzies, T. Bias in machine learning software: Why? how? what to do? In Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Athens, Greece, 23–28 August 2021; pp. 429–440.
13. Wang, T.; Zhao, J.; Yatskar, M.; Chang, K.W.; Ordonez, V. Balanced datasets are not enough: Estimating and mitigating gender bias in deep image representations. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 5310–5319.
14. Hutto, C.; Gilbert, E. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In Proceedings of the International AAAI Conference on Web and Social Media, Ann Arbor, MI, USA, 1–4 June 2014; Volume 8, pp. 216–225.
15. Jurgens, D.; Tsvetkov, Y.; Jurafsky, D. Incorporating Dialectal Variability for Socially Equitable Language Identification. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vancouver, BC, Canada, 30 July–4 August 2017; pp. 51–57.
16. Zehlike, M.; Bonchi, F.; Castillo, C.; Hajian, S.; Megahed, M.; Baeza-Yates, R. FA*IR: A Fair Top-k Ranking Algorithm. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, Singapore, 6–10 November 2017; ACM: New York, NY, USA, 2017; pp. 1569–1578.
17. Noble, S.U. *Algorithms of Oppression: How Search Engines Reinforce Racism*; New York University Press: New York, NY, USA, 2018.
18. Garcia, M. Racist in the Machine: The Disturbing Implications of Algorithmic Bias. *World Policy J.* **2016**, *33*, 111–117. [CrossRef]
19. Barocas, S.; Selbst, A.D. Big Data's Disparate Impact. *Calif. Law Rev.* **2016**, *104*, 671. [CrossRef]
20. Riessman, C.K. *Narrative Methods for the Human Sciences*; Sage: Newcastle upon Tyne, UK, 2008.
21. Fairclough, N. *Critical Discourse Analysis: The Critical Study of Language*; Longman: Harlow, UK, 1995.
22. Eubanks, V. *Automating Inequality: How High-Tech Tools Profile, Police, and Punish the Poor; St. Martin's Press: New York, NY, USA, 2018.*
23. Tramer, F.; Atlidakis, V.; Geambasu, R.; Hsu, D.; Hubaux, J.P.; Humbert, M.; Juels, A.; Lin, H. Fairtest: Discovering unwarranted associations in data-driven applications. In Proceedings of the 2017 IEEE European Symposium on Security and Privacy (EuroS&P), Paris, France, 26–28 April 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 401–416.

24. Bellamy, R.K.; Dey, K.; Hind, M.; Hoffman, S.C.; Houde, S.; Kannan, K.; Lohia, P.; Martino, J.; Mehta, S.; Mojsilović, A.; et al. AI Fairness 360: An extensible toolkit for detecting and mitigating algorithmic bias. *IBM J. Res. Dev.* **2019**, *63*, 1–15. [CrossRef]
25. Palmini, O.; Cugurullo, F. Design culture for Sustainable urban artificial intelligence: Bruno Latour and the search for a different AI urbanism. *Ethics Inf. Technol.* **2024**, *26*, 11. [CrossRef]
26. Smith, N.D. Plato's analogy of soul and state. *J. Ethics* **1999**, *3*, 31–49. [CrossRef]
27. Dennett, D.C. *Elbow Room: The Varieties of Free Will Worth Wanting*; MIT Press: Cambridge, MA, USA, 1984.
28. Turing, A.M. Computing Machinery and Intelligence. *Mind* **1950**, *59*, 433–460. [CrossRef]
29. Bostrom, N. *Superintelligence: Paths, Dangers, Strategies*; Oxford University Press: Oxford, UK, 2014.
30. Harari, Y.N. *Homo Deus: A Brief History of Tomorrow*; Harper: New York, NY, USA, 2016.

electronics

MDPI

Article

Web Application for Retrieval-Augmented Generation: Implementation and Testing

Irina Radeva [1,*], Ivan Popchev [2], Lyubka Doukovska [1] and Miroslava Dimitrova [1]

[1] Intelligent Systems Department, Institute of Information and Communication Technologies, Bulgarian Academy of Sciences, 1113 Sofia, Bulgaria; lyubka.doukovska@iict.bas.bg (L.D.); miroslava.dimitrova@iict.bas.bg (M.D.)
[2] Bulgarian Academy of Sciences, 1040 Sofia, Bulgaria; ivan.popchev@iict.bas.bg
* Correspondence: irina.radeva@iict.bas.bg

Abstract: The purpose of this paper is to explore the implementation of retrieval-augmented generation (RAG) technology with open-source large language models (LLMs). A dedicated web-based application, PaSSER, was developed, integrating RAG with Mistral:7b, Llama2:7b, and Orca2:7b models. Various software instruments were used in the application's development. PaSSER employs a set of evaluation metrics, including METEOR, ROUGE, BLEU, perplexity, cosine similarity, Pearson correlation, and F1 score, to assess LLMs' performance, particularly within the smart agriculture domain. The paper presents the results and analyses of two tests. One test assessed the performance of LLMs across different hardware configurations, while the other determined which model delivered the most accurate and contextually relevant responses within RAG. The paper discusses the integration of blockchain with LLMs to manage and store assessment results within a blockchain environment. The tests revealed that GPUs are essential for fast text generation, even for 7b models. Orca2:7b on Mac M1 was the fastest, and Mistral:7b had superior performance on the 446 question–answer dataset. The discussion is on technical and hardware considerations affecting LLMs' performance. The conclusion outlines future developments in leveraging other LLMs, fine-tuning approaches, and further integration with blockchain and IPFS.

Keywords: retrieval-augmented generation (RAG); open-source large language models (LLMs); Mistral:7b; Llama2:7b; Orca2:7b; Antelope blockchain; Ollama; LangChain; smart agriculture

1. Introduction

The advent of LLMs is changing the paradigm in natural language processing (NLP) toward improved classification, generation, and understanding of texts. However, general-purpose LLMs often require further adaptation to advance their performance in specific tasks or specialized domains. This has led to the development of different approaches to enhancing the performance of the models. Their goal has been to overcome the inherent limitations of pre-trained large-scale models. In this regard, the following models can be mentioned: full fine-tuning, parameter-efficient fine-tuning (PEFT), prompt engineering, and retrieval-augmented generation (RAG).

Full fine-tuning [1] is a method in which tuning occurs by adjusting all LLM parameters to specific data for a single task. This requires less data and offers greater accuracy and robustness of the results. The large scale of LLMs, however, makes the method's implementation computationally expensive, requiring significant memory, time, and expertise.

Parameter-efficient fine-tuning (PEFT) [2] emerged simultaneously, outlining an alternative strategy. The method focuses on modifying a selected number of parameters. The results are better (faster) learning performance and knowledge retention from pre-training. However, the performance of PEFT depends on the complexity of the task and the technique chosen, as it updates far fewer parameters in comparison to full fine-tuning.

In [3], the prompt engineering method is presented. This technique does not involve training network weights. To influence the desired output, it involves crafting the input to the model. This approach includes zero-shot prompting, few-shot prompting, and chain-of-thought prompting, each offering a way to guide the model's response without direct modification of its parameters. This method leverages the flexibility and capability of LLM and provides a tool to adapt the model without the computational cost of retraining.

RAG, introduced in [4], enhances language models by combining prompt engineering and database querying to provide context-rich answers, reducing errors and adapting to new data efficiently. The main concepts involve a combination of pre-trained language models with external knowledge retrieval, enabling dynamic, informed content generation. It is cost-effective and allows for traceable responses, making it interpretable. The development of retrieval-augmented generation (RAG) represents a significant advancement in the field of natural language processing (NLP). However, for deeper task-specific adaptations, like analysing financial or medical records, fine-tuning may be preferable. RAG's integration of retrieval and generation techniques addresses LLM issues like inaccuracies and opaque logic, yet incorporating varied knowledge and ensuring information relevance and accuracy remain challenges [5].

Each method offers a specific approach to improving LLM performance. Choosing between them depends on the desired balance between the required results, the available resources, and the nature of the tasks set.

In fact, there are other different methods in this field. They are founded on these basic approaches or applied in parallel. For example, dense passage retrieval (DPR) [6] and the retrieval-augmented language model (REALM) [7] refine retrieval mechanisms similar to RAG. Fusion-in-decoder (FiD) [8] integrates information from multiple sources into the decoding process. There are various knowledge-based modelling and meta-learning approaches. Each of these models reflects efforts to extend the capabilities of pre-trained language models and offer solutions for a wide range of NLP tasks.

The purpose of this paper is to explore the implementation of retriever-augmented generation (RAG) technology with open-source large language models (LLMs). In order to support this research, a web-based application PaSSER that allows the integration, testing and evaluation of such models in a structured environment has been developed.

The paper discusses the architecture of the web application, the technological tools used, the models selected for integration, and the set of functionalities developed to operate and evaluate these models. The evaluation of the models has two aspects: operation on different computational infrastructures and performance in text generation and summarization tasks.

The domain of smart agriculture is chosen as the empirical domain for testing the models. Furthermore, the web application is open-source, which promotes transparency and collaborative improvement. A detailed guide on installing and configuring the application, the datasets generated for testing purposes, and the results of the experimental evaluations are provided and available on GitHub [9].

The application allows adaptive testing of different scenarios. It integrates three of the leading LLMs, Mistral:7b, Llama2:7b, and Orca2:7b, which do not require significant computational resources. The selection of the Mistral:7b, Llama2:7b, and Orca2:7b models is driven by an approach aimed at balancing performance and affordability. The selected models were determined due to their respective volume parameters that allow installation and operation in mid-range configurations. Given the appropriate computational resources, without further refinement, the PaSSER application allows the use of arbitrary open-source LLMs with more parameters.

A set of standard NLP metrics—METEOR, ROUGE, BLEU, Laplace and Lidstone's perplexity, cosine similarity, Pearson correlation coefficient, and F1 score—was selected for a thorough evaluation of the models' performance.

In this paper, RAG is viewed as a *technology* rather than a mere *method*. This distinction is due to the paper's emphasis on the applied, practical, and integrative aspects of RAG in the field of NLP.

The paper contributes to the field of RAG research in several areas:

1. By implementing the PaSSER application, the study provides a practical framework that can be used and expanded upon in future RAG research.
2. The paper illustrates the integration of RAG technology with blockchain, enhancing data security and verifiability, which could inspire further exploration into the secure and transparent application of RAG systems.
3. By comparing different LLMs within the same RAG framework, the paper provides insights into the relative strengths and capabilities of the models, contributing knowledge on model selection in RAG contexts.
4. The focus on applying and testing within the domain of smart agriculture adds to the understanding of how RAG technology can be tailored and utilized in specific fields, expanding the scope of its application and relevance.
5. The use of open-source technologies in PaSSER development allows the users to review and trust the application's underlying mechanisms. More so, it enables collaboration, provides flexibility to adapt to specific needs or research goals, reduces development costs, facilitates scientific accuracy by enabling exact replication of research setups, and serves as a resource for learning about RAG technology and LLMs in practical scenarios.

The paper is organized as follows: Section 2 provides an overview of the development, implementation, and functionalities of the PaSSER Web App; Section 3 discusses selected standard NLP metrics used to measure RAG performance; Section 4 presents the results of tests on the models; Section 5, the limitations and influencing factors highlighted during the testing are discussed; and Section 6 summarizes the results and future directions for development.

2. Web Application Development and Implementation

This section describes the development, implementation, and features of the PaSSER Web App, which henceforward will be referred to as the PaSSER App.

The PaSSER App is a complementary project to the Smart crop production data exchange (SCPDx) platform, described in detail in [10,11]. The platform aims to support the integration of exchanging information and data acquired or generated as a result of the use of different technologies in smart agriculture. An underlying blockchain and distributed file system, eosio blockchain and InterPlanetary file system (IPFS), were selected [12]. The networks were deployed as private, permissionless. Later, in 2022, due to the hard fork, eosio blockchain was renamed to Antelope.io [13], which forced the migration of the blockchain network. The development of the SCPDx platform was preceded by a number of studies related to the choice of blockchain platform [14], blockchain-enabled supply-chain modelling for a smart crop production framework [15], and blockchain oracles integration [16].

The framework outlined in Figure 1 represents the PaSSER App's core infrastructure, the server side, the application itself, and the integration with the blockchain and IPFS of the SCPDx platform infrastructure.

There are different ways and techniques to work with LLMs and the RAG technology (Python scripts, desktop or web apps), but from a user's perspective, it is most convenient to use a web interface as it is independent of software platforms, where access to external resources is implemented through APIs. This is why the PaSSER App was developed as a web application.

The PaSSER App communicates with Ollama [17], ChromaDB [18], Python scripts, Antelope blockchain, and IPFS through their respective APIs. Currently, the IPFS network is not utilized within the framework; however, future enhancements aim to incorporate it

to enable storage and content distribution, which will support the fine-tuning processes of LLMs.

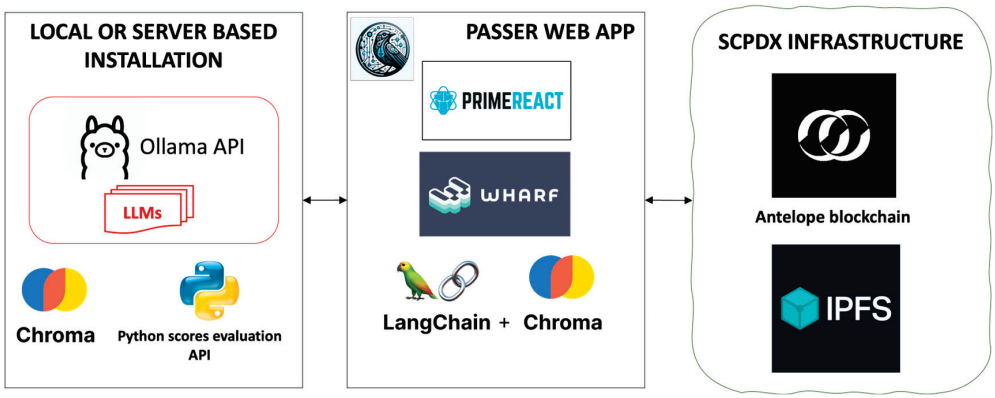


Figure 1. PaSSER App framework.

The PaSSER App is developed in JavaScript, leveraging the PrimeReact library [19] for its user interface components. To facilitate interaction with the Antelope blockchain network, it employs the WharfKit library [20]. The LangChain library [21] is integrated for engagement with RAG and LLMs. The application is hosted on an Apache web server, enabling communication between the PaSSER App, the LLMs, and the SCPDx infrastructure.

The server component of the PaSSER App includes a vector database, LLM API, and a score evaluation API. Ollama's API supports different operating systems (UBUNTU, Windows, and macOS) and is used as an interface with different LLMs. This API grants users the flexibility to manage and interact with different open-source LLMs. Specifically, Ollama is deployed on an Ubuntu server equipped with 128 GB RAM, not utilizing a GPU, and also configured locally on a Mac Mini M1 with 16 GB RAM and a 10-core GPU.

ChromaDB is used to work with vector databases. ChromaDB is an open-source vector database for applications that utilize large language models (LLMs). It supports multiple programming languages such as Python, JavaScript, Ruby, Java, Go, and others. The database is licensed under the Apache 2.0 license. ChromaDB architecture is suited for applications that require semantic search capabilities. An embedding refers to the transformation of text, images, or audio into a vector of numbers, which represents the essence of the content in a way that can be processed and understood by machine learning models. In this implementation, ChromaDB was installed on a macOS-based server and locally on a Mac Mini M1 16 GB RAM 10 GPU.

Two Python scripts are developed: one for computing various evaluation metrics using libraries such as NLTK [22], torch [23], numpy [24], rouge [25], transformers [26], and scipy [27], and the other for logging model performance data, both using the Pyntelope library [28] for blockchain interactions. These scripts are operational on an Ubuntu server and are accessible via GitHub [9], facilitating the analysis and the recording of performance metrics within a blockchain framework to ensure data integrity and reproducibility of results.

2.1. LLMs Integration

In this implementation of PaSSER, Mistral:7b, Llama2:7b, and Orca2:7b models were selected. These models have the same volume parameters and are suitable to be installed and used on hardware configurations with medium computational capacity, while giving good enough results. A brief description of the models is presented below.

Mistral:7b "https://mistral.ai/news/announcing-mistral-7b/ (accessed on 23 March 2024)" is a language model developed by Mistral AI with a capacity of 7.3 billion parameters. It is available under the Apache 2.0 license, so it can be used without restrictions. The model can be downloaded and used anywhere, including locally, and deployed on any cloud (AWS/GCP/Azure) using the LLM inference server and SkyPilot.

It is structured to process language data, facilitating a wide range of text-based applications. This model incorporates specific attention mechanisms, namely grouped-query attention (GQA) and sliding window attention (SWA), aimed at optimizing the processing speed and managing longer text inputs. These technological choices are intended to improve the model's performance while managing computational resources effectively [29–31].

Llama2:7b "https://llama.meta.com/llama2/ (accessed on 23 March 2024)" is a series of large language models developed by Meta, offering variations in size from 7 billion to 70 billion parameters. The 7 billion parameter version, Llama 2:7b, is part of this collection and is designed for a broad range of text-based tasks, from generative text models to more specific applications like chatbots. The architecture of Llama 2 models is auto-regressive, utilizing an optimized transformer structure. These models have been pre-trained on a mix of publicly available online data and can be fine-tuned for specific applications. They employ an auto-regressive language model framework and have been optimized for various natural language processing tasks, including chat and dialogue scenarios through supervised fine-tuning (SFT) and reinforcement learning with human feedback (RLHF) [32].

Llama2:7b models have been trained on significant computational resources and have a considerable carbon footprint, which Meta has committed to offsetting as part of their sustainability program. The 7b model specifically consumed 184,320 GPU hours and emitted an estimated 31.22 tCO2eq during its training, with all emissions directly offset. These models are intended for both commercial and research purposes, and while they have been primarily tested in English, they carry the potential for a broad array of applications. Meta has provided a custom commercial license for Llama 2, and details on accessing this license can be found on their website.

Orca2:7b, Orca 2 [33], developed by Microsoft, is a fine-tuned version of the Llama 2 model with two versions: one with 7 billion and the other with 13 billion parameters. It focuses on improving the reasoning abilities of smaller language models through enhanced training methods. By employing high-quality synthetic data for training, Orca 2 is designed to master various reasoning techniques such as step-by-step processing and recall-then-generate strategies. This synthetic training data is crafted to guide the model in adopting different solution strategies appropriate for varied tasks, aiming to optimize the model's problem-solving approach.

The model is a result of research aimed at leveraging the advancements in large language models to boost the performance of smaller models, making them more efficient and versatile in handling tasks that require complex reasoning. The initiative behind Orca 2 is to provide a resource for research into the development, evaluation, and alignment of language models, particularly focusing on smaller models that can perform efficiently across a range of tasks.

Orca 2 is available for public use and research, underscoring Microsoft's commitment to advancing the field of AI and language model technology. It represents an effort to explore how smaller models can be enhanced to approach a variety of tasks more effectively without the extensive computational and environmental costs associated with larger models.

2.2. PaSSER App Functionalities

The site map for the PaSSER App is depicted in Figure 2, detailing its core features. Figure 2 provides a structured flowchart of the PaSSER App's user interface, mapping out the navigation pathways for various functionalities, such as the creation and management of a vector store, chat interactions, testing protocols, configuration settings, database management, and user authentication. Each section outlines the sequence of user actions and choices, from inputting data to exporting test results, configuring system settings, and maintaining the database. These include user authentication (login), system configuration (configuration setup), creation and management of vector stores from various sources (text, PDF files, and websites), and functionalities for engaging with the stored data through

Q&A chat and RAG Q&A chat based on the established vector stores. Additionally, a testing module is outlined to evaluate the application's functionalities and performance.

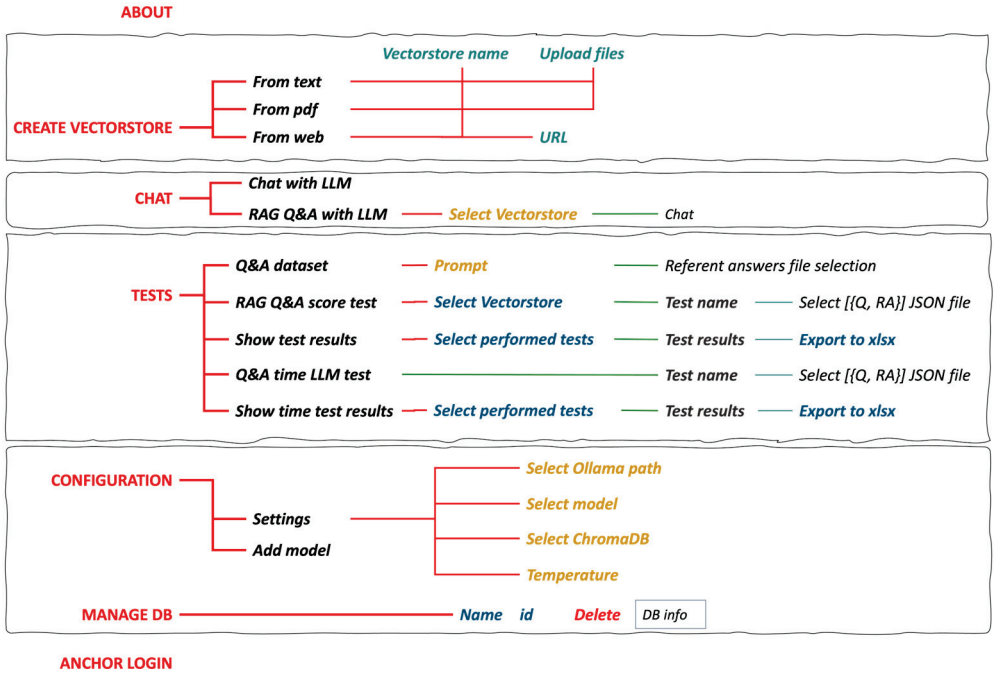


Figure 2. PaSSER site map.

The '*Create vectorstore*' feature, as depicted in Figure 3, outlines the process of converting raw textual data into a structured, queryable vector space using LangChain. This transformation of NLP and vector embedding techniques makes it possible to convert text into a format convenient for vector operations. Users can source textual data from text files, PDFs, and websites. The outlined procedure for vectorstore creation is standardized across these data types, ensuring consistency in processing and storage. At the current phase, automatic retrieval of information from websites (scrapping) is considered impractical due to the necessity for in-depth analysis of website structures and the requirement for extensive manual intervention to adequately structure the retrieved text. This process involves understanding varied and complex web layouts and imposing a tailored approach to effectively extract and organize data.

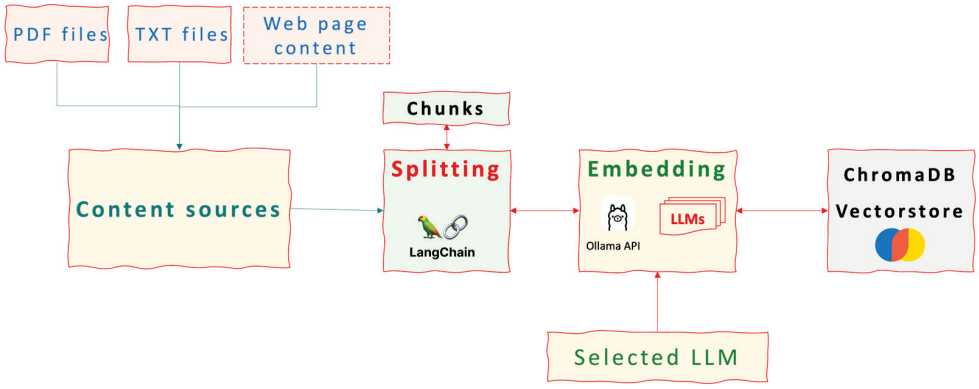
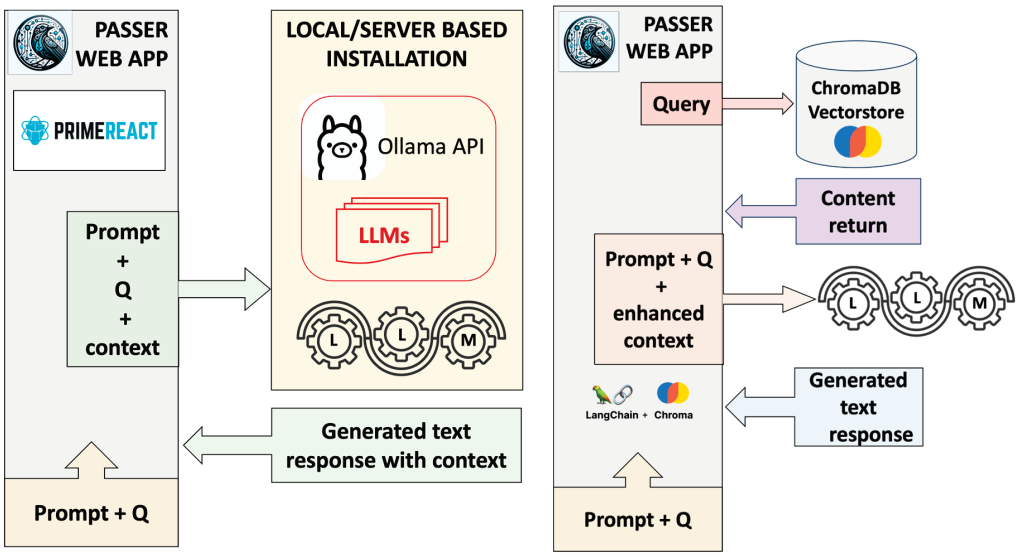


Figure 3. Vectorstore construction workflow.

The process for creating a vectorstore involves the following steps, which are common to all three source types:

1. *Cleaning and standardizing text data.* This is achieved by removing unnecessary characters (punctuation and special characters). Converting the text to a uniform size (usually lower case). Separating the text into individual words or tokens. In the implementation considered here, the text is divided into chunks with different overlaps.
2. *Vector embedding.* The goal is to convert tokens (text tokens) into numeric vectors. This is achieved by using pre-trained word embedding models from selected LLMs (in this case, Misrtal:7b, Llama2:7b, and Orca2:7b). These models map words or phrases to high-dimensional vectors. Each word or phrase in the text is transformed into a vector that represents its semantic meaning based on the context in which it appears.
3. *Aggregating embeddings* for larger text units to represent whole sentences or documents as vectors. It can be achieved by simple aggregation methods (averaging the vectors of all words in a sentence or document) or by using sentence transformers or document embedding techniques that take into account the more consistent and contextual nature of words. Here, transformers are used, which are taken from the selected LLMs.
4. *Create a vectorstore* to store the vector representations in a structured format. The data structures used are optimized for operations with high-dimensional vectors. ChromaDB is used for the vectorstore.

Figure 4 defines the PaSSER App's mechanisms for processing user queries and generating responses. Figure 4a represents a general Q&A chat workflow with direct input without the augmented context provided by a vectorstore. The corresponding LLM processes the query, formulates a response, and concurrently provides system performance data, including metrics such as total load and evaluation timeframes. Additionally, a numerical array captures the contextual backdrop of the query and the response, drawn from previous dialogue or related data, which the LLM utilizes similar to short-term memory to ensure response relevance and coherence. While the capacity of this memory is limited and not the focus of the current study, it is pivotal in refining responses based on specific contextual elements such as names and dates. The App enables saving this context for continued dialogue and offers features for initiating new conversations by purging the existing context.



(**a**) Q&A chat (**b**) RAG-based Q&A chat

Figure 4. Chat types workflow.

Figure 4b illustrates the workflow of a Q&A chat using the RAG technique, which employs a pre-built vectorstore for data retrieval and response generation. The ***'RAG Q&A chat'*** feature facilitates interaction with an existing vectorstore. Users commence the chat

by entering a query, triggering the LangChain library to fetch related data from the chosen vectorstore. This data informs the subsequent query to the LLM, integrating the original question, any prompts, and a context enriched by the vectorstore's information. The LLM then generates a response. Within the app, a dedicated memory buffer recalls history, which the LLM utilizes as a transient context to ensure consistent and logical responses. The limited capacity of this memory buffer and its impact on response quality is acknowledged, though not extensively explored in this study. In the '***RAG Q&A chat***', context-specific details like names and dates are crucial for enhancing the relevance of responses.

The '***Tests***' feature is designed to streamline the testing of various LLMs within a specific knowledge domain. It involves the following steps:

1. Selection of a specific knowledge base in a specific domain.

With '***Create vectorstore***', the knowledge base is processed and saved in the vector database. In order to evaluate the performance of different LLMs for generating RAG answers on a specific domain, it is necessary to prepare a sufficiently large list of questions and reference answers. Such a list can be prepared entirely manually by experts in a specific domain. However, this is a slow and time-consuming process. Another widely used approach is to generate relevant questions based on reference answers given by a selected LLM (i.e., creating respective datasets). PaSSER allows the implementation of the second approach.

2. To create a reference dataset for a specific domain, a collection of answers related to the selected domain is gathered. Each response contains key information related to potential queries in that area. These answers are then saved in a text file format.
3. A selected LLM is deployed to systematically generate a series of questions corresponding to each predefined reference answer. This operation facilitates the creation of a structured dataset comprising pairs of questions and their corresponding answers. Subsequently, this dataset is saved in the JSON file format.
4. The finalized dataset is uploaded to the PaSSER App, initiating an automated sequence of response generation for each query within the target domain. Following that, each generated response is forwarded to a dedicated Python backend script. This script is tasked with assessing the responses based on predefined metrics and comparing them to the established reference answers. The outcomes of this evaluation are then stored on the blockchain, ensuring a transparent and immutable ledger of the model's performance metrics.

To facilitate this process, a smart contract *'llmtest'* has been created, managing the interaction with the blockchain and providing a structured and secure method for storing and managing the assessment results derived from the LLM performance tests.

The provided pseudocode outlines the structure *'tests'* and its methods within a blockchain environment, which were chosen to store test-related entries. It includes identifiers (*id*, *userid*, and *testid*), a timestamp (*created_at*), numerical results (*results* array), and descriptive text (*description*). It establishes *id* as the primary key for indexing, with additional indices based on *created_at*, *userid*, and *testid* to facilitate data retrieval and sorting by these attributes. This structure organizes and accesses test records within the blockchain.

```
Define a structure 'tests' with the following fields:
id of type integer
userid of type name
testid of type name
created_at of type timestamp
results of type list of doubles
description of type string

Define the following methods for 'tests':
primary_key returns id
third_key returns the seconds since epoch from created_at
user_key returns the value of userid
test_key returns the value of testid
```

The pseudocode below defines an *eosio::multi_index* table '*tests_table*' for a blockchain, which facilitates the storage and indexing of data. It specifies four indices: a primary index based on *id* and secondary indices using *created_at*, *userid*, and *testid* attributes for enhanced query capabilities. These indices optimize data retrieval operations, allowing for efficient access based on different key attributes like timestamp, user, and test identifiers, significantly enhancing the database's functionality within the blockchain environment.

```
    Define a multi-index table 'tests_table' with the following indices:
    'id' index based on the primary_key method
    'timestamp' index based on the third_key method
    'users' index based on the user_key method
    'testid' index based on the test_key method
```

The provided pseudocode defines an EOSIO smart contract action named *add_test*, which allows adding a new record to the *tests_table*. It accepts the creator's name, test ID, description, and an array of results as parameters. The action assigns a unique ID to the record, stores the current timestamp, and then inserts a new entry into the table using these details. This action helps in dynamically updating the blockchain state with new test information, ensuring that each entry is time-stamped and linked to its creator.

```
    1. Define a function add_test with parameters: creator, testid, description,
results
    2. Create a tests_table object with the current contract's name
    3. Get the next available primary key from the tests_table
    4. Get the current timestamp
    5. Add a new entry to the tests_table with the following fields:
    - id: the next available primary key
    - userid: the creator's name
    - testid: the testid
    - created_at: the current timestamp
    - description: the description
    - results: the results
    6. Return a new add_test object
```

The pseudocodes provided above and in Section 3 are generated with GitHub Copilot upon the actual source code available at "https://github.com/features/copilot (accessed on 1 April 2024)".

5. The results from the blockchain are retrieved for further processing and analysis.

To facilitate the execution of these procedures, the interface is structured into three specific features: '***Q&A dataset***' for managing question and answer datasets, '***RAG Q&A score test***' for evaluating the performance of RAG utilizing datasets, and '***Show test results***' for displaying the results of the tests. Each submenu is designed to streamline the respective aspect of the workflow, ensuring a coherent and efficient user experience throughout the process of dataset management, performance evaluation, and result visualization.

Within the '***Q&A dataset***', the user is guided to employ a specific prompt, aiming to instruct the LLM to generate questions that align closely with the provided reference answers, as described in step 2. This operation initiates the creation of a comprehensive dataset, subsequently organizing and storing this information within a JSON file for future accessibility and analysis. This approach ensures the generation of relevant and accurate questions, thereby enhancing the dataset's utility for follow-up evaluation processes.

The '***RAG Q&A score test***' is designed to streamline the evaluation of different LLMs' performances using the RAG, as indicated in Figure 5. This evaluation process involves importing a JSON-formatted dataset and linking it with an established vectorstore relevant to the selected domain. The automation embedded within this menu facilitates a methodical assessment of the LLMs, leveraging domain-specific knowledge embedded within the vectorstore.

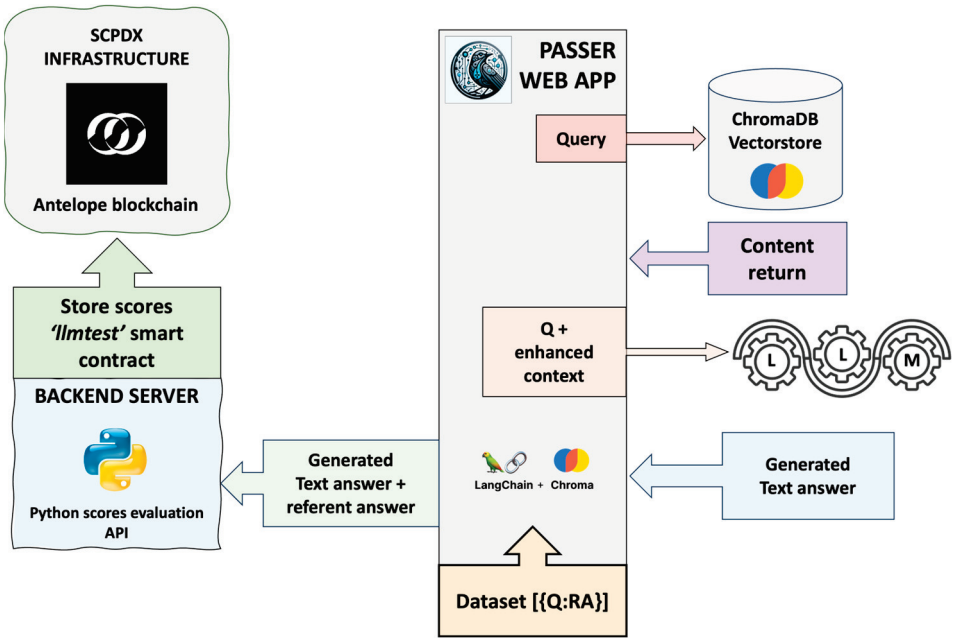


Figure 5. Workflow diagram for RAG LLM query processing and score storage.

Vectorstores, once created using a specific LLM's transformers, require the consistent application of the same LLM model during the RAG process. Within this automated framework, each question from the dataset is processed by the LLM to produce a corresponding answer. Then, both the generated answers and their associated reference answers are evaluated by a backend Python script. This script calculates performance metrics, records these metrics on the blockchain under a specified test series, and iterates this procedure for every item within the dataset.

The '***Show test results***' feature is designed to access and display the evaluation outcomes from various tests as recorded on the blockchain, presenting them in an organized tabular format. This feature facilitates the visualization of score results for individual answers across different test series and also provides the functionality to export this data into an xlsx file format. The export feature makes it much easier for users to understand and study the data, helping with better evaluations and insights.

The '***Q&A Time LLM Test***' feature evaluates model performance across various hardware setups using JSON-formatted question–answer pairs. Upon submission, the PaSSER App prompts the selected model for responses, generating detailed performance metrics like evaluation and load times, among others. These metrics are packed in a query to a backend Python script, which records the data on the blockchain via the *'addtimetest'* action, interacting with the *'llmtest'* smart contract to ensure performance tracking and data integrity.

The '***Show time test results***' makes it easy to access and view LLM performance data, organized by test series, from the blockchain. When displayed in a structured table, these metrics can be examined for comprehensive performance assessment. There is an option to export this data into an xlsx file, thereby improving the process for further in-depth examination and analysis.

Authentication within the system ('***Login***') is provided through the Anchor wallet, which is compliant with the security protocols of the SCPDx platform. This process, described in detail in [34], provides user authentication by ensuring that testing activities are securely associated with the correct user credentials. This strengthens the integrity and accountability of the testing process within the platform ecosystem.

The '***Configuration***' feature is divided into '***Settings***' and '***Add Model***'.

The '***Settings***' is designed for configuring connectivity to the Ollama API and ChromaDB API, using IP addresses specified in the application's configuration file. It also

allows users to select an LLM that is currently installed in Ollama. A key feature here is the ability to adjust the 'temperature' parameter, which ranges from 0 to 1, to fine-tune the balance between creativity and predictability in the output generated by the LLM. Setting a higher temperature value (>0.8) increases randomness, whereas a lower value enhances determinism, with the default set at 0.2.

The '***Add Model***' enables adding and removing LLMs in the Ollama API, allowing dynamic model management. This feature is useful when testing different models, ensuring optimal use of computational resources.

The '***Manage DB***' feature displays a comprehensive list of vectorstores available in ChromaDB, offering functionalities to inspect or interact with specific dataset records. This feature enables users to view details within a record's JSON response. It provides the option to delete any vectorstore that is no longer needed, enabling efficient database management by removing obsolete or redundant data, thereby optimizing storage utilization.

A block diagram representation of the PaSSER App's operational logic that illustrates the interactions between the various components is provided in Figure 6.

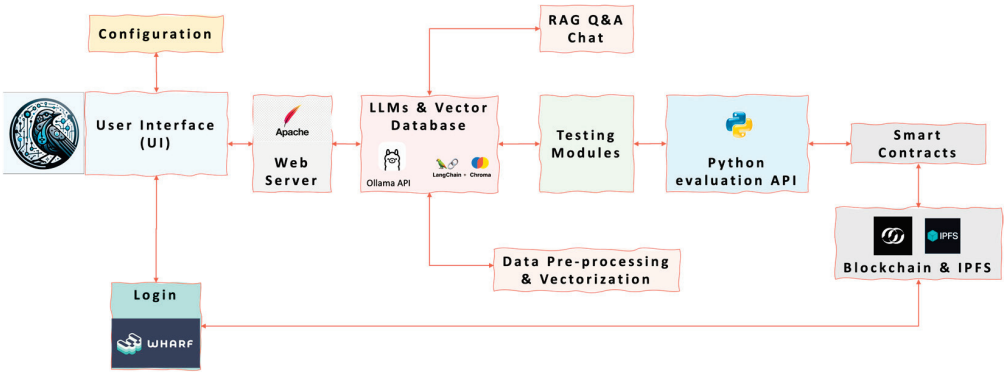


Figure 6. PaSSER App block diagram.

The UI (web application) interacts with users for configuration, authentication, and operation initiation. It utilizes JavaScript and the PrimeReact library for UI components. Enables the user interactions for authentication (Login), configuration, and operations with the LLMs and blockchain.

The web server (Apache) hosts the web application, facilitating communication between the user interface and backend components.

The LLM API and Vector Database utilize the Ollama API for the management of different LLMs. It incorporates ChromaDB for storage and retrieval of vectorized data.

Data pre-processing and vectorization standardize and convert data from various sources (e.g., PDFs, websites) into numerical vectors for LLM processing using pre-trained models of selected LLMs.

The RAG Q&A chat facilitates query responses by integrating external data retrieval with LLM processing. It enables querying the LLMs with augmented information retrieval from the vector database for generating responses.

Testing modules utilize built-in testing modules to assess LLM performance across metrics, with results recorded on the blockchain.

The Python evaluation API calculates NLP performance metrics and interacts with the blockchain for recording the testing results via smart contracts.

Smart contracts manage test results recording on the blockchain.

3. Evaluation Metrics

The evaluation of RAG models within the PaSSER App was performed using a set of 13 standard NLP metrics. These metrics evaluated various dimensions of model performance, including the quality of text generation and summarization, semantic similarity, predictive accuracy, and consistency of generated content compared to reference or expected

results. Metrics included METEOR, ROUGE (with ROUGE-1 and ROUGE-L variants), BLEU, perplexity (using Laplace and Lidstone smoothing techniques), cosine similarity, Pearson correlation coefficient, and F1 score.

The PaSSER App ran two main tests to assess LLM: "LLM Q&A Time Test" and "RAG Q&A Assessment Test". The latter specifically applied the selected metrics to a created dataset of question–answer pairs for the smart agriculture domain. The test aimed to determine which model provides the most accurate and contextually relevant answers within the RAG framework and the capabilities of each model in the context of text generation and summarization tasks.

The '***RAG Q&A chat***' was assessed using a set of selected metrics: METEOR, ROUGE, PPL (perplexity), cosine similarity, Pearson correlation coefficient, and F1 score [35].

An automated evaluation process was developed to apply these metrics to the answers generated using RAG. The process compared generated answers against the reference answers in the dataset, calculating scores for each metric.

All calculations were implemented in backEnd.py script in Python, available at: "https://github.com/scpdxtest/PaSSER/blob/main/scripts/backEnd.py (accessed on 1 April 2024)".

The following is a brief explanation of the purpose of the metrics used, the simplified calculation formulas, and the application in the context of RAG.

3.1. METEOR (Metric for Evaluation of Translation with Explicit Ordering)

METEOR score is a metric used to assess the quality of machine-generated text by comparing it to one or multiple reference texts [36]. The calculating involves several steps:

- Word alignment between candidate and reference translations based on exact, stem, synonym, and paraphrase matches, with the constraint that each word in the candidate and reference sentences can only be used once and aims to maximize the overall match between the candidate and references.
- Calculation of *Precision* (P) = Number of matched words in the candidate/Number of words in the candidate and *Recall* (R) = Number of matched words in the candidate/Number of words in the reference:

$$P = \frac{m}{w_c}, R = \frac{m}{w_r} \tag{1}$$

where:

m = Number of unigrams in the candidate translation that are matched with the reference translation.

w_c = Total number of unigrams in the candidate translation.

w_r = Total number of unigrams in the reference translation(s).

- Calculation of *Penalty* for chunkiness, which accounts for the arrangement and fluency of the matched chunks (c) = Number of chunks of contiguous matched unigrams in the candidate translation and (m):

$$Penalty = 0.5\left(\frac{c}{m}\right)^3 \tag{2}$$

- The final score is computed using the harmonic mean of Precision and Recall, adjusted by the penalty factor:

$$M_{score} = F_{mean}(1 - Penalty) \tag{3}$$

where $F_{mean} = \frac{10PR}{R+9P}$.

The implementation of the calculations of Equations (1)–(3) is conducted with the nltk library, *single_meteor_score* function, line 58 in Python script.

This pseudocode describes the process of splitting two texts into words and calculating the METEOR score between them.

```
    1. Split the reference and candidate texts into words
    2. Calculate the METEOR score between the word lists of reference and candidate
using the 'single_meteor_score' function
```

In the context of RAG models, the METEOR score can be used to evaluate the quality of the generated responses. A high METEOR score indicates that the generated response closely matches the reference text, suggesting that the model is accurately retrieving and generating responses. Conversely, a low METEOR score could indicate areas for improvement in the model's performance.

3.2. ROUGE (Recall-Oriented Understudy for Gisting Evaluation)

ROUGE [37] is a set of metrics used for evaluating automatic summarization and machine translation. It works by comparing an automatically produced summary or translation against one or more reference summaries (usually human-generated).

ROUGE has several variants: ROUGE-N, ROUGE-L, and ROUGE-W.

ROUGE-N focuses on the overlap of n-grams (sequences of n words) between the system-generated summary and the reference summaries. It is computed in terms of recall, precision, and F1 score:

- $Recall_{ROUGE\text{-}N}$ is the ratio of the number of overlapping n-grams between the system summary and the reference summaries to the total number of n-grams in the reference summaries:

$$Recall_{ROUGE\text{-}N} = \frac{\sum_{S \in \{Reference\ Summaries\}} \sum_{gram_n \in s} Count_{match}(gram_n)}{\sum_{S \in \{Reference\ Summaries\}} \sum_{gram_n \in s} Count(gram_n)} \tag{4}$$

- $Precision_{ROUGE\text{-}N}$ is the ratio of the number of overlapping n-grams in the system summary to the total number of n-grams in the system summary itself:

$$Precision_{ROUGE\text{-}N} = \frac{\sum_{S \in \{System\ Summaries\}} \sum_{gram_n \in s} Count_{match}(gram_n)}{\sum_{S \in \{System\ Summaries\}} \sum_{gram_n \in s} Count(gram_n)} \tag{5}$$

- $F1_{ROUGE\text{-}N}$ is the harmonic mean of precision and recall:

$$F1_{ROUGE\text{-}N} = 2\frac{Precision_{ROUGE\text{-}N} \times Recall_{ROUGE\text{-}N}}{Precision_{ROUGE\text{-}N} + Recall_{ROUGE\text{-}N}} \tag{6}$$

ROUGE-L focuses on the longest common subsequence (LCS) between the generated summary and the reference summaries. The LCS is the longest sequence of words that appears in both texts in the same order, though not necessarily consecutively. The parameters for ROUGE-L include:

- $Recall_{ROUGE\text{-}L}$ is the length of the LCS divided by the total number of words in the reference summary. This measures the extent to which the generated summary captures the content of the reference summaries:

$$Recall_{ROUGE\text{-}N} = \frac{LCS(System\ Summary, Reference\ Summary)}{Lenght\ of\ Reference\ Summary} \tag{7}$$

- $Precision_{ROUGE\text{-}N}$ is the length of the LCS divided by the total number of words in the generated summary. This assesses the extent to which the words in the generated summary appear in the reference summaries:

$$Precision_{ROUGE\text{-}N} = \frac{LCS(System\ Summary, Reference\ Summary)}{Lenght\ of\ System\ Summary} \tag{8}$$

– $F1_{ROUGE\text{-}N}$ is a harmonic mean of the LCS-based precision and recall:

$$F1_{ROUGE\text{-}N} = 2\frac{Precision_{ROUGE\text{-}N} \times Recall_{ROUGE\text{-}N}}{Precision_{ROUGE\text{-}N} + Recall_{ROUGE\text{-}N}} \quad (9)$$

ROUGE-W is an extension of ROUGE-L with a weighting scheme that assigns more importance to longer sequences of matching words. In this application, ROUGE-W is not applied.

The implementation of the calculations of Equations (4)–(9) is conducted with the rouge library, *rouge.get_scores* function, line 65 in Python script.

This pseudocode describes the process of initializing a ROUGE object and calculating the ROUGE scores between two texts.

```
    1. Set 'hypothesis' to the reference text and 'ref' to the candidate text
    2. Initialize a Rouge object
    3. Calculate the ROUGE scores between 'hypothesis' and 'ref' using the
'get_scores' method of the Rouge object
```

The choice between a preference for precision, recall, or F1 scoring depends on the specific goals of the summarization task, such as whether it is more important to capture as much information as possible (recall) or to ensure that what is captured is highly relevant (precision).

In the context of RAG models, ROUGE metric serves as a tool for assessing the quality of the generated text, especially in summary, question answering, and content-generation tasks.

3.3. BLEU (Bilingual Evaluation Understudy)

The BLEU [38] score is a metric used to assess the quality of machine-generated text by comparing it to one or multiple reference texts. It quantifies the resemblance by analysing the presence of shared n-grams (sequences of n consecutive words).

The metric employs a combination of modified precision scores for various n-gram lengths and incorporates a brevity penalty to account for the adequacy and fluency of the translation. Below are the simplified formulas for estimating BLEU.

For each n-gram length n, BLEU computes a modified precision score. The modified precision P_n for n-grams is calculated as follows:

$$P_n = \frac{\sum_{C \in \{Candidate\ Translation\}} \sum_{n-grams \in C} Count_{clip}(n-gram)}{\sum_{C' \in \{Candidate\ Translation\}} \sum_{n-grams \in C'} Count_{clip}(n-gram')} \quad (10)$$

where, $Count_{clip}$ is a count of each n-gram in the candidate translation clipped by its maximum count in any single reference translation.

The brevity penalty (BP) is a component of the BLEU score that ensures translations are not only accurate but also of appropriate length. The BP is defined as:

$$BP = \begin{cases} 1 & if\ c > r \\ e^{(1-r/c)} & if\ c \leq r \end{cases} \quad (11)$$

where, c is the total length of the candidate translation, and r is the effective reference corpus length, which is the sum of the lengths of the closest matching reference translations for each candidate sentence.

$BP = 1$ if the candidate translation length c is greater than the reference length r, indicating no penalty.

$BP = e^{(1-r/c)}$ if c is less than or equal to r, indicating a penalty that increases as the candidate translation becomes shorter relative to the reference.

The overall BLEU score is calculated using the following formula:

$$BLEU = BP.exp\left(\sum_{n=1}^{N} w_n log P_n\right) \tag{12}$$

where, N is the maximum n-gram length (typically 4), and w_n is the weight for each n-gram's precision score, often set equally such that their sum is 1 (e.g., $w_n = 0.25$ for $N = 4$).

This formula aggregates the individual modified precision scores P_n for n-grams of length 1 to N, geometrically averaged and weighted by w_n, then multiplied by the brevity penalty BP to yield the final BLEU score.

The implementation of the calculations of Equations (10)–(12) is conducted with the nltk library, *sentence_bleu* and *SmoothingFunction* functions, lines 74–79 in Python script.

This pseudocode describes the process of splitting two texts into words, creating a smoothing function, and calculating the BLEU score between them.

```
    1. Split the reference and candidate texts into words
    2. Create a smoothing function using method4 of the SmoothingFunction class
    3. Calculate the BLEU score between the word lists of reference and candidate
using the 'sentence_bleu' function with the smoothing function
```

In the context of RAG models, the BLEU score can be used to evaluate the quality of the generated responses. A high BLEU score would indicate that the generated response closely matches the reference text, suggesting that the model is accurately retrieving and generating responses. A low BLEU score could indicate areas for improvement in the model's performance.

3.4. Perplexity (PPL)

Perplexity (PPL) [39] is a measure used to evaluate the performance of probabilistic language models. The introduction of smoothing techniques, such as Laplace (add-one) smoothing and Lidstone smoothing [40], aims to address the issue of zero probabilities for unseen events, thereby enhancing the model's ability to deal with sparse data. Below are the formulas for calculating perplexity.

– PPL with *Laplace Smoothing* adjusts the probability estimation for each word by adding one to the count of each word in the training corpus, including unseen words. This method ensures that no word has a zero probability. The adjusted probability estimate with Laplace smoothing is calculated using the following formula:

$$P_{Laplace}(w_i|h) = \frac{C(w_i, h) + 1}{C(h) + V} \tag{13}$$

where, w_i is the probability of a word given its history h (the words that precede it), $C(w_i, h)$ is the count of w_i, $C(h)$ is the count of history h, and V is a vocabulary size (the number of unique words in the training set plus one for unseen words).

The PPL of a sequence of words $W = w_1, \ldots w_N$ is given by:

$$PPL(W) = e^{-\frac{1}{N}\sum_{i=1}^{N} ln(P_{Laplace}(w_i|h))} \tag{14}$$

The implementation of the calculations of Equations (13) and (14) is conducted with the nltk library, lines 84–102, in Python script.

This pseudocode describes the process of tokenizing an input text paragraph, training a Laplace model (bigram model), and calculating the perplexity of a candidate text using the model.

```
    1. Tokenize the input text paragraph into sentences and words, convert all
words to lowercase
    2. Split the tokenized text into training data and vocabulary using a bigram
model
    3. Train a Laplace model (bigram model) using the training data and vocabulary
    4. Define a function 'calculate_perplexity' that:
        a. Tokenizes the input text into words, converts all words to lowercase
        b. Calculates the perplexity of the text using the Laplace model
    5. Set 'test_text' to the candidate text
    6.  Calculate  the  Laplace  perplexity  of  'test_text'  using  the
'calculate_perplexity' function
```

– PPL with *Lidstone smoothing* is a generalization of Laplace smoothing where instead of adding one to each count, a fraction λ (where $0 < \lambda < 1$) is added. This allows for more flexibility compared to the fixed increment in Laplace smoothing. Adjusted Probability Estimate with Lidstone Smoothing:

$$P_{Lidstone}(w_i|h) = \frac{C(w_i, h) + \lambda}{C(h) + \lambda V} \tag{15}$$

The PPL of a sequence of words $W = w_1, \ldots w_N$ is given by:

$$PPL(W) = e^{-\frac{1}{N}\sum_{i=1}^{N} ln(P_{Lidstone}(w_i|h))} \tag{16}$$

The implementation of the calculations of Equations (15) and (16) is conducted with the nltk library, lines 108–129, in Python script.

This pseudocode describes the process of tokenizing an input text paragraph, training a Lidstone model (trigram model), and calculating the perplexity of a candidate text using the model.

```
    1. Set the training text to the reference text
    2. Tokenize the training text into sentences and then into words, convert all
words to lowercase
    3. Prepare the training data for a trigram model
    4. Create and train a Lidstone model with Lidstone smoothing, where gamma is
the Lidstone smoothing parameter
    5. Set the test text to the candidate text
    6. Tokenize the test text into sentences and then into words, convert all words
to lowercase
    7. Prepare the test data
    8. Calculate the Lidstone perplexity of the test text
```

In both formulas, the goal is to compute how well the model predicts the test set W. The lower perplexity indicates that the model predicts the sequence more accurately. The choice between Laplace and Lidstone smoothing depends on the specific requirements of the model and dataset, as well as empirical validation.

In the context of RAG models, both metrics are useful for assessing the quality and ability of models to deal with a variety of language and information. These metrics indicate how well they can generate contextually informed, linguistically coherent, and versatile text.

3.5. Cosine Similarity

Cosine similarity [41] is a measure of vector similarity and can be used to determine the distance of embeddings between the chunk and the query. It is a distance metric that approaches 1 when the question and chunk are similar and becomes 0 when they are different. The mathematics formulation of the metric is:

$$Cosin\ Similarity = \frac{A.B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{17}$$

where, $A.B$ is the dot product of vectors A and B, $\|A\|$ and $\|B\|$ are the Euclidean norms (magnitudes) of vectors A and B, calculated with $\sqrt{\sum_{i=1}^{n} A_i^2}$ and $\sqrt{\sum_{i=1}^{n} B_i^2}$, respectively, and n is the dimensionality of the vectors, assuming A and B of the same dimension.

$Cosin\ Similarity = 1$ means the vectors are identical in orientation.

$Cosin\ Similarity = 0$ means the vectors are orthogonal (independent) to each other.

$Cosin\ Similarity = -1$ means the vectors are diametrically opposed.

The implementation of the calculation of Equation (17) is conducted with the transformers library, lines 133–164, in Python script.

This pseudocode describes the process of tokenizing two texts, generating BERT embeddings for them, and calculating the cosine similarity between the embeddings. The [CLS] token is used as the aggregate representation for classification tasks.

```
    1. Define a function 'get_bert_embedding' that:
        a. Takes in a text, a tokenizer, and a model
        b. Tokenizes the text and converts it to a tensor
        c. Gets the BERT embeddings for the text
        d. Returns the [CLS] token embedding
    2. Initialize the tokenizer and model for BERT
    3. Set 'text_1' to the reference text and 'text_2' to the candidate text
    4. Tokenize 'text_1' and 'text_2'
    5. Generate BERT embeddings for 'text_1' and 'text_2'
    6. Get the [CLS] token embeddings for 'text_1' and 'text_2'
    7. Calculate the cosine similarity between the [CLS] token embeddings of
'text_1' and 'text_2'
```

In the RAG models, cosine similarity ensures that retrieved documents align closely with user queries, capturing relationships between the meaning of a user. This is particularly important in RAG models, as they leverage a retriever to find context documents. The use of cosine similarity between embeddings ensures that these retrieved documents align closely with user queries.

3.6. Pearson Correlation

The Pearson correlation coefficient ® is a statistical measure that calculates the strength and direction of the linear relationship between two continuous variables.

The formula for the Pearson correlation coefficient is as follows:

$$r = \frac{\sum_{i=1}^{n} \left(X_i - \overline{X}\right)\left(Y_i - \overline{Y}\right)}{\sqrt{\sum_{i=1}^{n} \left(X_i - \overline{X}\right)^2} \sqrt{\sum_{i=1}^{n} \left(Y_i - \overline{Y}\right)^2}} \tag{18}$$

where, n is the number of data points, X_i and Y_i are the individual data points, and $\overline{X}$ and $\overline{Y}$ are the means of the X and Y data sets, respectively.

The implementation of the calculation of Equation (18) is conducted with the transformers and script libraries, lines 167–178, in Python script.

This pseudocode describes the process of tokenizing two texts, generating BERT embeddings for them, and calculating the Pearson correlation coefficient between the embeddings. The mean of the last hidden state of the embeddings is used as the aggregate representation.

```
    1. Define a function 'get_bert_embedding_manhattan' that:
        a. Takes in a text, a tokenizer, and a model
        b. Tokenizes the text and converts it to a tensor
        c. Gets the BERT embeddings for the text
        d. Returns the mean of the last hidden state of the embeddings
    2. Initialize the tokenizer and model for BERT
    3. Set 'text_1' to the reference text and 'text_2' to the candidate text
    4. Generate BERT embeddings for 'text_1' and 'text_2' using the 'get_bert_em-
bedding_manhattan' function
    5. Calculate the Pearson Correlation Coefficient between the embeddings of
'text_1' and 'text_2'
```

In the context of evaluating RAG models, the Pearson correlation coefficient can be used to measure how well the model's predictions align with actual outcomes. A coefficient close to +1 indicates a strong positive linear relationship, meaning as one variable increases, the other also increases. A coefficient close to -1 indicates a strong negative linear relationship, meaning as one variable increases, the other decreases. A coefficient near 0 suggests no linear correlation between variables. In the evaluation of RAG models, a high Pearson correlation coefficient could indicate that the model is accurately retrieving and generating responses, while a low coefficient could suggest areas for improvement.

3.7. F1 Score

In the context of evaluating the performance of RAG models, the F1 score [42] is used for quantitatively assessing how well the models perform in tasks for generating or retrieving textual information (question answering, document summarization, or conversational AI). The evaluation often hinges on their ability to accurately and relevantly generate text that aligns with reference or ground truth data.

The F1 score is the harmonic mean of precision and recall. *Precision* assesses the portion of relevant information in the responses generated by the RAG model. High precision indicates that most of the content generated by the model is relevant to the query or task at hand, minimizing irrelevant or incorrect information. *Recall* (or sensitivity) evaluates the model's ability to capture all relevant information from the knowledge base that should be included in the response. High recall signifies that the model successfully retrieves and incorporates a significant portion of the pertinent information available in the context.

The formula for calculating is:

$$\mathrm{F1} = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{19}$$

Precision and Recall are defined as:

$$Precision = \frac{TP}{TP + FP}, Recall = \frac{TP}{TP + FN} \tag{20}$$

where, TP (True Positives) is the count of correctly retrieved relevant documents, FP (False Positives) is the count of incorrectly retrieved documents (i.e., the documents that were retrieved but are not relevant), and FN (False Negatives) is the count of relevant documents that were not retrieved.

The implementation of the calculations of Equations (19) and (20) occurs on lines 185–204 in Python script.

This pseudocode describes the process of tokenizing two texts, counting the common tokens between them, and calculating the F1 score.

```
    1. Define a function 'f1_score' that:
        a. Takes in a prediction and a truth
        b. Tokenizes the prediction and truth into words, converts all words to
    lowercase
        c. Counts the common tokens between prediction and truth
        d. If there are no common tokens, return 0
        e. Calculate precision as the number of common tokens divided by the total
    number of tokens in prediction
        f. Calculate recall as the number of common tokens divided by the total
    number of tokens in truth
        g. Calculate F1 score as the harmonic mean of precision and recall
        h. Return the F1 score
    2. Set 'prediction' to the candidate text and 'truth' to the reference text
    3. Calculate the F1 score between 'prediction' and 'truth' using the 'f1_score'
function
```

For tasks of question answering, the F1 score can be used to measure how well the generated answers match the expected answers, considering both the presence of correct information (high precision) and the completeness of the answer (high recall).

For tasks of document summarization, the F1 score might evaluate the overlap between the key phrases or sentences in the model-generated summaries and those in the reference summaries, reflecting the model's efficiency in capturing essential information (recall) and avoiding extraneous content (precision).

For, conversational AI applications, the F1 score could assess the relevance and completeness of the model's responses in dialogue, ensuring that responses are both pertinent to the conversation context and comprehensive in addressing users' intents or questions.

4. Testing

The aim of the tests presented in this section is to evaluate the performance of the Misrtal:7b, Llama2:7b, and Orca2:7b models installed on two different hardware configurations and to assess the performance of these models in generating answers using RAG on the selected knowledge domain, smart agriculture.

The knowledge base used was retrieved from EU Regulation 2018/848 "https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32018R0848 (accessed on 1 April 2024)" and Climate-smart agriculture Sourcebook "https://www.fao.org/3/i3325e/i3325e.pdf (accessed on 1 April 2024)". These documents were pre-processed manually and vectorized using the transformers of Misrtal:7b, Llama2:7b, and Orca2:7b LLMs under the following parameters: chunk size—500, overlapping—100, temperature—0.2.

The dataset containing reference answers specific to smart agriculture was compiled and stored as a text file. The Mistral:7b model was deployed to formulate questions based on these reference answers. Initial trials indicated that Mistral:7b excelled in generating questions with high relevance within this particular domain. To initiate the question generation process, the following prompt was employed: "Imagine you are a virtual assistant trained in the detailed regulations of organic agriculture. Your task involves creating precise questions for a specific regulatory statement provided to you below. The statement comes directly from the regulations, and your challenge is to reverse-engineer the question that this statement answers. Your formulated question should be concise, clear, and directly related to the content of the statement. Aim to craft your question without implying the statement itself as the answer, and, where relevant, gear your question toward eliciting specific events, facts, or regulations."

For testing purposes, two Ollama APIs installed on two different hardware configurations were used:

- Intel Xeon, 32 Cores, 0 GPU, 128 GB RAM, Ubuntu 22.04 OS.
- Mac Mini M1, 8 CPU, 10 GPU, 16 GB RAM, OSX 13.4.

In the PASSER App, the installed Ollama APIs that were to be used could be selected. This is set in the configuration->settings menu.

The two following tests were designed: testing via the '***Q&A Time LLM Test***' and the '***RAG Q&A score test***'.

The '***Q&A Time LLM Test***' evaluated LLM performance across two hardware configurations using a dataset of 446 questions for each model, focusing on seven specific metrics (evaluation time, evaluation count, load duration time, prompt evaluation count, prompt evaluation duration, total duration, and tokens per second). These metrics were integral for analyzing the efficiency and responsiveness of each model under different computational conditions. The collected data was stored on a blockchain, ensuring both transparency and traceability of the evaluation results.

The '***RAG Q&A score test***' aimed to evaluate the performance of the models based on 13 metrics (METEOR, ROUGE-1, ROUGE-l, BLEU, perplexity, cosine similarity, Pearson correlation, and F1) applied to each of the 446 questions—reference answers—for which RAG obtained answers.

The '***RAG Q&A score test***' evaluated the performance of different models in a chat environment with enhanced RAG Q&A, identifying differences and patterns in their ability to respond to queries. Its goal was to determine the model that best provided accurate, context-aware responses that defined terms and summarized specific content.

This evaluation can be used to select a model that ensures the delivery of accurate and relevant information in the context of the specific knowledge provided.

The performance outcomes from the '***Q&A Time LLM Test***' and '***RAG Q&A score test***' for evaluating LLMs were stored on the blockchain via smart contracts. For analysis, this data was retrieved from the blockchain and stored in an xlsx file. This file was uploaded to GitHib "https://github.com/scpdxtest/PaSSER/blob/main/tests/TEST%20DATA_GENERAL%20FILE.xlsx (accessed on 1 April 2024)".

In the upcoming section, the focus is solely on presenting and analysing the mean values derived from the test data. This approach eases the interpretation, enabling a summarized review of the core findings and trends across the conducted evaluations.

4.1. Q&A Time LLM Test Results

When using the '***Q&A Chat***' feature, performance metrics are returned by the Ollama API along with each response. The metrics are reported in nanoseconds and converted to seconds for easier interpretation. They are used to evaluate the performance of different models under different hardware configurations.

Evaluation count is the number of individual evaluations performed by the model during a specific test. It helps to understand its capacity to process multiple prompts or tasks within a given timeframe, reflecting its productivity under varying workloads.

Load duration time is the time required for initializing and loading the language model into memory before it begins processing data. This metric is essential for assessing the start-up efficiency of the model, which can significantly impact user experience and operational latency in real-time applications.

Prompt evaluation count is the number of prompts the model evaluates within a specific test or operational period. It provides insights into the model's interactive performance, especially relevant in scenarios where the model is expected to respond to user inputs or queries dynamically.

Prompt evaluation duration captures the time model used to evaluate a single prompt, from receiving the input to generating the output. It measures the model's responsiveness and is particularly important for interactive applications where fast output is vital.

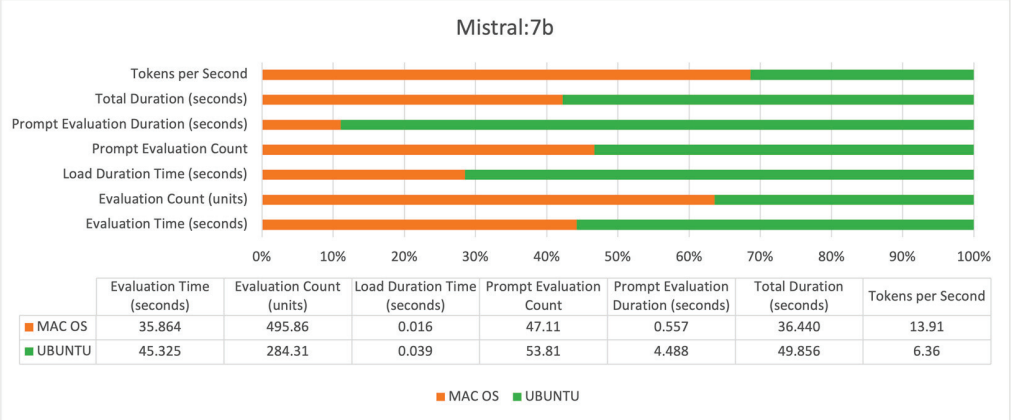
Total duration is the overall time for the entire evaluation process, incorporating all phases from initialization to the completion of the last prompt evaluation. This metric gives a complete view of the model's operational efficiency over an entire test cycle.

Tokens per second quantifies the number of tokens (basic units of text, such as words or characters) the model can process per second. It is a key indicator of the model's processing speed and computational throughput, reflecting its ability to handle large volumes of text data efficiently.

Figure 7 illustrates the performance metrics of the Mistral:7b on macOS and Ubuntu operating systems across the different indicators. The model demonstrates higher efficiency on macOS, as indicated by the shorter evaluation time, longer prompt evaluation count, and significantly higher tokens per second rate. Conversely, the model takes longer to process prompts on Ubuntu, as indicated by the extended duration of prompt evaluation, even though the operating system manages a greater number of prompt evaluations overall.

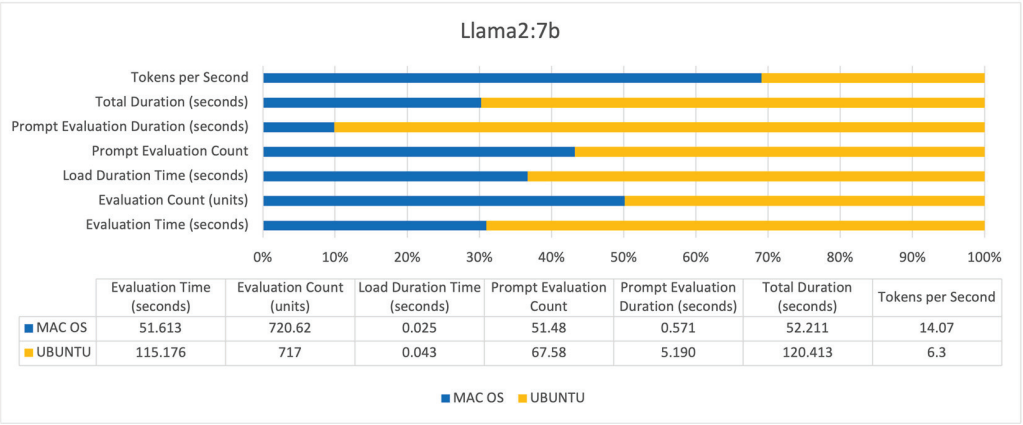
Figure 8 presents a comparison of the Llama2:7b performance. It reveals that MAC OS is the more efficient platform, processing a higher number of tokens per second and completing evaluations faster. Despite Ubuntu's slightly higher prompt evaluation count, it shows longer prompt evaluation and total duration times.

Figure 9 illustrates performance metrics for the Orca2:7b. MAC OS outperforms Ubuntu with faster evaluation times, shorter load durations, and notably higher tokens per second processing efficiency. While Ubuntu managed a higher prompt evaluation count, it lagged in prompt evaluation speed, as reflected in the longer prompt evaluation duration and extended total duration times.



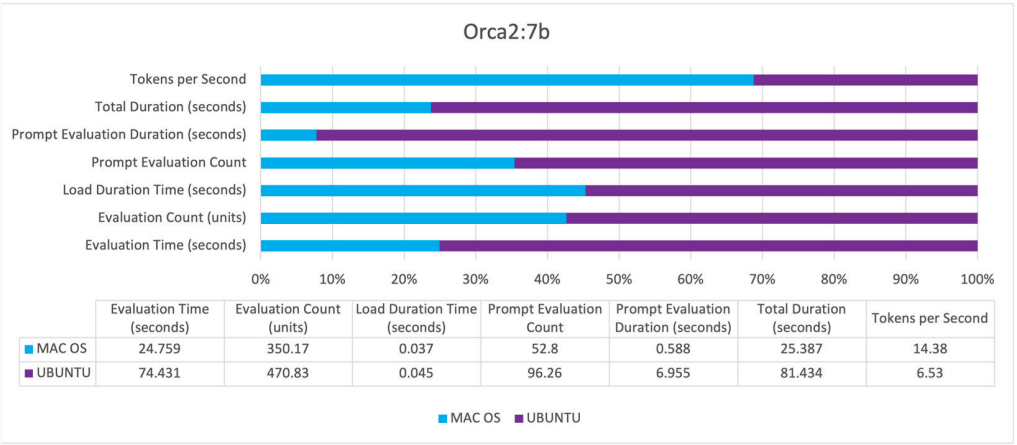
	Evaluation Time (seconds)	Evaluation Count (units)	Load Duration Time (seconds)	Prompt Evaluation Count	Prompt Evaluation Duration (seconds)	Total Duration (seconds)	Tokens per Second
MAC OS	35.864	495.86	0.016	47.11	0.557	36.440	13.91
UBUNTU	45.325	284.31	0.039	53.81	4.488	49.856	6.36

Figure 7. A performance of Mistral:7b.



	Evaluation Time (seconds)	Evaluation Count (units)	Load Duration Time (seconds)	Prompt Evaluation Count	Prompt Evaluation Duration (seconds)	Total Duration (seconds)	Tokens per Second
MAC OS	51.613	720.62	0.025	51.48	0.571	52.211	14.07
UBUNTU	115.176	717	0.043	67.58	5.190	120.413	6.3

Figure 8. A performance of Llama2:7b.



	Evaluation Time (seconds)	Evaluation Count (units)	Load Duration Time (seconds)	Prompt Evaluation Count	Prompt Evaluation Duration (seconds)	Total Duration (seconds)	Tokens per Second
MAC OS	24.759	350.17	0.037	52.8	0.588	25.387	14.38
UBUNTU	74.431	470.83	0.045	96.26	6.955	81.434	6.53

Figure 9. A performance of Orca2:7b.

Across all models, several trends are evident (Figures 7–9). UBUNTU generally shows longer *evaluation times*, indicating slower processing capabilities compared to MAC OS. *Evaluation counts* are relatively comparable, suggesting that the number of operations conducted within a given timeframe is similar across hardware configurations. *Load duration times* are consistently longer on UBUNTU, affecting readiness and response times negatively. UBUNTU tends to conduct more *prompt evaluation count*, but also takes significantly longer, which exposes efficiency issues. UBUNTU experiences longer *total durations* for all tasks, reinforcing the trend of slower overall performance. MAC OS demonstrates higher *tokens per second* across all models, indicating more efficient data processing capabilities.

The performance indicators (Table 1) suggest that across all models, the evaluation time on the Mac M1 system is significantly less than on the Ubuntu system with Xeon processors, indicating faster overall performance. In terms of tokens per second, the Mac M1 also performs better, suggesting it is more efficient at processing information regardless of having fewer CPU cores and less RAM.

Table 1. Comparative performance metrics of Llama2:7b, Mistral:7b, and Orca2:7b LLMs on macOS M1 and Ubuntu Xeon Systems (w/o GPU).

	Llama2:7b		**Mistral:7b**		**Orca2:7b**	
Metric	**macOS/M1**	**Ubuntu/Xeon**	**macOS/M1**	**Ubuntu/Xeon**	**macOS/M1**	**Ubuntu/Xeon**
Evaluation time (s)	Faster (51,613)	Slower (115,176)	Faster (35,864)	Slower (45,325)	Fastest (24,759)	Slowest (74,431)
Evaluation count (units)	Slightly Higher (720)	Comparable (717)	Higher (496)	Lower (284)	Lower (350)	Higher (471)
Load duration time (s)	Faster (0.025)	Slower (0.043)	Fastest (0.016)	Slower (0.039)	Similar (0.037)	Similar (0.045)
Prompt evaluation count	Lower (51)	Higher (68)	Lower (47)	Higher (54)	Lower (53)	Highest (96)
Prompt evaluation duration (s)	Shorter (0.571)	Longer (5.190)	Shorter (0.557)	Longer (4.488)	Shorter (0.588)	Longest (6.955)
Total duration (s)	Shorter (52,211)	Longer (120,413)	Shorter (36,440)	Longer (49,856)	Shortest (25,387)	Longer (81,434)
Tokens/second	Higher (14.07)	Lower (6.3)	Higher (13.91)	Lower (6.36)	Highest (14.38)	Lower (6.53)

Despite having a higher core count and more RAM, the evaluation time is longer, and the tokens per second rate are lower on the Ubuntu system. This suggests that the hardware advantages of the Xeon system are not translating into performance gains for these particular models. Notably, the Ubuntu system shows a higher prompt evaluation count for Orca2:7b, which might be leveraging the greater number of CPU cores to handle more prompts simultaneously.

Orca2:7b has the lowest evaluation time on the Mac M1 system, showcasing the most efficient utilization of that hardware. Llama2:7b shows a significant difference in performance between the two systems, indicating it may be more sensitive to hardware and operating system optimizations. Mistral:7b has a comparatively closer performance between the two systems, suggesting it may be more adaptable to different hardware configurations.

The table suggests that the Mac M1's architecture provides a significant performance advantage for these language models over the Ubuntu system equipped with a Xeon processor. This could be due to several factors, including but not limited to the efficiency of the M1 chip, the optimization of the language models for the specific architectures, and the potential use of the M1's GPU in processing.

4.2. RAG Q&A Score Test Results

In this section, the performance of the Mistral:7b, Llama2:7b, and Orca2:7b models is assessed through several key metrics: METEOR, ROUGE-1, ROUGE-L, BLEU, perplexity, cosine similarity, Pearson correlation coefficient, and F1 score. A summary of the average metrics is presented in Table 2.

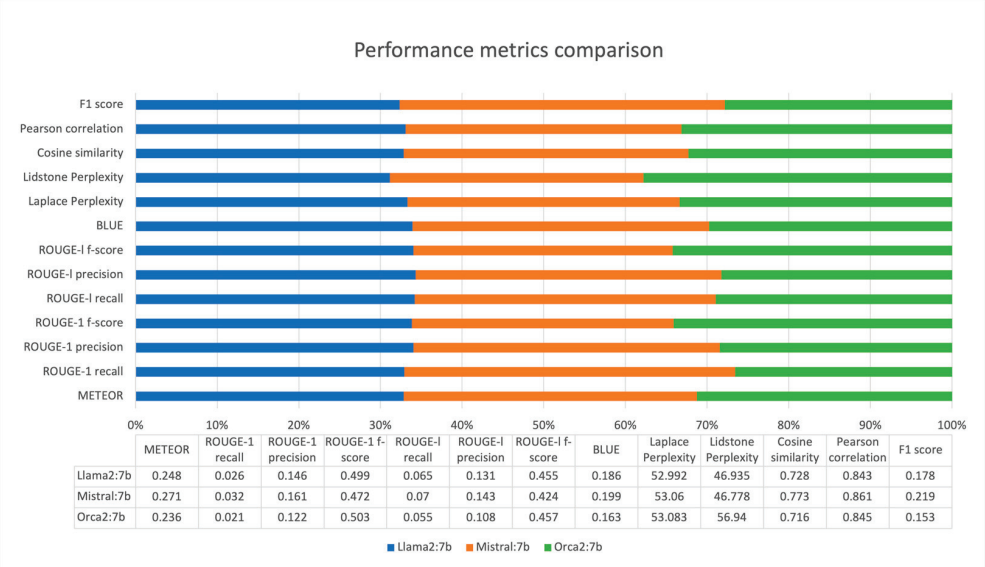
Table 2. A summary of performance metrics using RAG Q&A chat.

Metric	Llama2:7b	Mistral:7b	Orca2:7b	Best Model	Metric in Text Generation and Summarization Tasks
METEOR	0.248	**0.271**	0.236	Mistral:7b	Assesses fluency and adequacy of generated text response, considering synonymy and paraphrase.
ROUGE-1 recall	0.026	**0.032**	0.021	Mistral:7b	Measures the extent to which a generated summary captures key points from a source text, indicating coverage.
ROUGE-1 precision	0.146	**0.161**	0.122	Mistral:7b	Evaluates the fraction of content in the generated summary that is relevant to the source text, implying conciseness.
ROUGE-1 f-score	0.499	0.472	**0.503**	Orca2:7b	Provides a balance between recall and precision for assessing the overall quality of a generated summary.
ROUGE-l recall	0.065	**0.07**	0.055	Mistral:7b	Reflects the degree to which a generated lowercase summary encompasses the content of a reference lowercase summary.
ROUGE-l precision	0.131	**0.143**	0.108	Mistral:7b	Measures the accuracy of a generated lowercase summary in replicating the significant elements of the source text.
ROUGE-l f-score	0.455	0.424	**0.457**	Orca2:7b	Integrates precision and recall to evaluate the quality of a generated lowercase summary holistically.
BLUE	0.186	**0.199**	0.163	Mistral:7b	Quantifies the similarity of the generated text to reference texts by comparing n-grams, which is useful for machine translation and summarization.
Laplace perplexity	**52.992**	53.06	53.083	Llama2:7b	Estimates the likelihood of a sequence in generated text, indicating how well the text generation model predicts sample sequences.
Lidstone perplexity	**46.935**	46.778	56.94	Mistral:7b	Assesses the smoothness and predictability of a text generation model by evaluating the likelihood of sequence occurrence with small probability adjustments.
Cosine similarity	**0.728**	0.773	0.716	Mistral:7b	Determines the semantic similarity between the vector representations of generated text and reference texts.
Pearson correlation	**0.843**	0.861	0.845	Mistral:7b	Quantifies the linear correspondence between generated text scores and human-evaluated scores, indicating model predictability and reliability.
F1 score	**0.178**	0.219	0.153	Mistral:7b	Combines the precision and recall of the generated text in summarization tasks, providing a singular measure of its informational quality.

For a more straightforward interpretation of the results, the ranges of values of the different metrics are briefly described below.

The *ideal METEOR score is 1*. It indicates a perfect match between the machine-generated text and the reference translations, encompassing both semantic and syntactic accuracy. For ROUGE metrics (*ROUGE-1 recall, precision, f-score, ROUGE-l recall, precision, f-score), the best possible value is 1*. This value denotes a perfect overlap between the content generated by the model and the reference content, indicating high levels of relevance and

precision in the captured information. *The BLEU score's maximum is also 1* (or 100 when expressed in percentage terms), representing an exact match between the machine's output and the reference texts, reflecting high coherence and context accuracy. *For perplexity, the lower the value, the better the model's predictive performance*. The best perplexity score would technically approach 1, indicating the model's predictions are highly accurate with minimal uncertainty. *The cosine similarity of 1 signifies maximum similarity between the generated output and the reference*. *A Pearson correlation of 1 is ideal*, signifying a perfect positive linear relationship between the model's outputs and the reference data, indicating high reliability of the model's performance. *An F1 score reaches its best at 1*, representing perfect precision and recall, meaning the model has no false positives or false negatives in its output. For a better comparison of the models, Figure 10 is presented.



	METEOR	ROUGE-1 recall	ROUGE-1 precision	ROUGE-1 f-score	ROUGE-l recall	ROUGE-l precision	ROUGE-l f-score	BLUE	Laplace Perplexity	Lidstone Perplexity	Cosine similarity	Pearson correlation	F1 score
Llama2:7b	0.248	0.026	0.146	0.499	0.065	0.131	0.455	0.186	52.992	46.935	0.728	0.843	0.178
Mistral:7b	0.271	0.032	0.161	0.472	0.07	0.143	0.424	0.199	53.06	46.778	0.773	0.861	0.219
Orca2:7b	0.236	0.021	0.122	0.503	0.055	0.108	0.457	0.163	53.083	56.94	0.716	0.845	0.153

Figure 10. A comparison of performance metrics.

The presented metrics provide a picture of the performance of the models on text generation and summarization tasks. The analysis for each metric is as follows.

METEOR evaluates the quality of translation by aligning the model output to reference translations when considering precision and recall. *Mistral:7b scores highest, suggesting its translations or generated text are the most accurate*.

ROUGE-1 recall measures the overlap of unigrams between the generated summary and the reference. A higher score indicates more content overlap. *Mistral:7b leads, which implies it includes more of the reference content in its summaries or generated text*.

ROUGE-1 precision (the unigram precision). *Mistral:7b has the highest score, indicating that its content is more relevant and has fewer irrelevant inclusions.*

ROUGE-1 F-score is the harmonic mean of precision and recall. *Orca2:7b leads slightly, indicating a balanced trade-off between precision and recall in its content generation.*

ROUGE-L recall measures the longest common subsequence and is good at evaluating sentence-level structure similarity. *Mistral:7b scores the highest, showing it is better at capturing longer sequences from the reference text.*

ROUGE-L precision. *Mistral:7b again scores highest, indicating it includes longer, relevant sequences in its summaries or generated text without much irrelevant information.*

ROUGE-L F-Score. Orca2:7b has a marginally higher score, suggesting a balance in precision and recall for longer content blocks.

BLEU assesses the quality of machine-generated translation. *Mistral:7b outperforms the others, indicating its translations may be more coherent and contextually appropriate.*

Laplace perplexity. For perplexity, a lower score is better as it indicates a model's predictions are more certain. *Llama2:7b has the lowest score, suggesting the best predictability under Laplace smoothing conditions.*

Lidstone perplexity—Mistral:7b has the lowest score, indicating it is slightly more predictable under Lidstone smoothing conditions.

Cosine similarity measures the cosine of the angle between two vectors. A higher score indicates greater semantic similarity. *Mistral:7b has the highest score, suggesting its generated text is most similar to the reference text in terms of meaning*.

Pearson correlation measures the linear correlation between two variables. A score of 1 indicates perfect correlation. *Mistral:7b has the highest score, showing its outputs have a stronger linear relationship with the reference data.*

F1 score balances precision and recall. *Mistral:7b has the highest F1 score, indicating the best balance between recall and precision in its outputs.*

Based on the above analysis, the following summary can be concluded. For text generation and summarization, Mistral:7b appears to be the best-performing model in most metrics, particularly those related to semantic quality and relevance. Orca2:7b shows strength in balancing precision and recall, especially for longer content sequences. Llama2:7b demonstrates the best predictive capability under Laplace smoothing conditions, which may be beneficial in certain predictive text generation tasks. The selection of the best model would depend on the specific requirements of the text generation or summarization task.

4.3. Blockchain RAM Resource Evaluation

The integration of blockchain technologies to record test results aims to increase the confidence and transparency of the results obtained, support their traceability, and facilitate their documentation. This can also be implemented using classical databases (e.g., Mongo DB, Postgres, Maria DB, or others), but as the amount of data stored in the blockchain is relatively small (on average 1 kB per test), there is no obstacle to exploit the advantages of this technology.

In terms of blockchain performance, the platform used (Antelope) is characterized as one of the fastest—over 7000 transactions per second. Since the main time for running the tests is due to the generation of answers (with or without the use of RAG), it is multiple times longer than the time for recording the results in the blockchain (for example, about 1 min for generating an answer, and about 0.5 s for recording the results in the blockchain).

PaSSER App uses a private, permissionless blockchain network. Such a blockchain network can be installed by anyone with the required technical knowledge. Instructions for installing the Antelope blockchain can be found on GitHub at: "https://github.com/scpdxtest/PaSSER/blob/main/#%20Installation%20Instructions.md (accessed on 1 April 2024)". There are no plans to implement a public blockchain network in this project.

The blockchain (Antelope) must use a system token. The symbol (SYS), precision, and supply are set when the network is installed. All system resources (RAM, CPU, NET) are evaluated in SYS. Similar to other platforms (e.g., Bitcoin, Ethereum), to execute transactions and build blocks (to prevent denial of service attacks), Antelope charges a fee in SYS tokens.

The NET is used to measure the amount of data that can be sent per transaction as an average consumption in bytes over the last 3 days (72 h). It is consumed temporarily for each action/transaction.

The CPU limits the maximum execution time of a transaction and is measured as average consumption in microseconds, also for the last 3 days. It is also temporarily consumed when an action or transaction is sent. NET and CPU are together called bandwidth. In public networks, users can use the bandwidth resources if available, i.e., that are not mortgaged by other accounts. The price of mortgaged bandwidth (NET and CPU) resources varies, depending on how many SYS tokens are staked in total.

The RAM is the information that is accessible from application logic (order books, account balances). It limits the maximum space that can be occupied to store permanent

data (in the block producers' RAM). An account can exchange NET and CPU by mortgaging SYS tokens, but must buy RAM. RAM is not freed automatically. The only way to free up memory is to delete the data that is using the account (multi-index tables). Freed unused RAM can be sold and purchased at the market price, which is determined by the Bancor algorithm [43]. In detail, the entire RAM, CPU, NET evaluation procedure that applies here is described in [11].

In our case, the RAM price is extracted from the '*rammarket*' table of the *eosio system account* by executing the *cleos* command from the command line with the appropriate parameters (date 23 March 2024, 14:05 h).

```
"rows": [{
"supply": "10000000000.0000 RAMCORE",
"base": {
"balance": "68660625616 RAM",
"weight": "0.50000000000000000"
},
"quote": {
"balance": "1000857.1307 SYS",
"weight": "0.50000000000000000"
}
}
]
```

In order to apply the Bancor algorithm for RAM pricing to our private network, the following clarifications should be made. The Antelope blockchain network has a so-called RAM token. The PaSSER uses our private Antelope blockchain network, whose system token is SYS. In the context of the Bancor algorithm, RAM and SYS should be considered *Smart Tokens*. The *Smart Token* is a token that has one or more connectors with other tokens in the network. The connector, in this case, is a SYS token, and it establishes a relationship between SYS and RAM. Using the Bancor algorithm [43,44] could be presented as follows:

$$\begin{gathered} RAMPrice = cb/(STos \times CW), CW = cb/STtv, => \\ RAMPrice = cb/(STos \times cb/STtv) = STtv/STos \end{gathered} \tag{21}$$

where: *cb* is for connector balance, *Stos* is for a Smart Token's Outstanding Supply = base.balance, and *STtv* is for a Smart Token's total value = Connector Balance = quote.balance.

The cost evaluation of RAM, CPU, and NET resources in SYS tokens during a test execution occurs as follows. The PaSSER App uses SYS tokens. The CPU price is measured in (SYS token/ms/Day) and is valid for a specific account on the network. The NET Price is measured in (SYS token/KiB/Day) and is valid for a specific account on the network.

This study only considers the cost of the RAM resources required to run the tests.

Data on the current market price of the RAM resource is retrieved from an oracle [16] every 60 min that runs within the SCPDx platform whose blockchain infrastructure is being used.

The current price of RAM is 0.01503345 SYS/kB as of 23 March 2024, 2:05 PM. Assuming the price of 1 SYS is equal to the price of 1 EOS, it is possible to compare the price of the RAM used if the tests are run on the public Antelope blockchain because there is a quote of the EOS RAM Token in USD and it does not depend on the account used. The quote is available at "https://coinmarketcap.com/ (accessed on 23 March 2024)".

Table 3 shows that in terms of RAM usage and the associated costs in SYS and USD, the score tests require more resources than the timing tests. The total cost of using blockchain resources for these tests is less than 50 USD. This gives reason to assume that using blockchain to manage and document test results has promise. The RAM price, measured in SYS per kilobyte (kB), remains constant across different tests in the blockchain network.

Table 3. Blockchain test costs.

Operation	Bytes	RAM Price (SYS/kB)	EOS Price (USD)	RAM Cost (SYS)	Equivalent Cost (USD)
Time tests *	402,300	0.01503345	1.01	5.91	5.97
Score tests **	2,896,560	0.01503345	1.01	42.52	42.95
Total	3,298,860			48.43	48.92

* Six series of 446 blockchain transactions. ** Three series of 446 blockchain BC transactions.

This means that blockchain developers and users can anticipate and plan for the costs associated with their blockchain operations. The blockchain resource pricing model is designed to maintain a predictable and reliable cost structure. This predictability matters for the long-term sustainability and scalability of blockchain projects as it allows for accurate cost estimation and resource allocation. However, the real value of implementing blockchain must also consider the benefits of increased transparency, security and traceability against these costs.

5. Discussion

The PaSSER App testing observations reveal several aspects that affect the acquired results and the performance and can be managed. These are data cleaning and pre-processing, chunk sizes, GPU usage, and RAM size.

Data cleaning and pre-processing cannot be fully automated. In addition to removing special characters and hyperlinks, it is also necessary to remove non-essential or erroneous information, standardize formats, and correct errors from the primary data. This is done manually. At this stage, the PaSSER App processes only textual information; therefore, the normalization of data, handling of missing data, and detection and removal of deviations are not considered.

Selecting documents with current, validated, and accurate data is pivotal, yet this process cannot be entirely automated. What can be achieved is to ensure traceability and record the updates and origins of both primary and processed data, along with their secure storage. Blockchain and distributed file systems can be used for this purpose. Here, this objective is partially implemented since blockchain is used solely to record the testing results.

The second aspect is chunk sizes when creating and using vectorstores. Smaller chunks require less memory and computational resources. This is at the expense of increased iterations and overall execution time, which is balanced by greater concurrency in query processing. On the other hand, larger chunks provide more context but may be more demanding on resources and potentially slow down processes if not managed efficiently.

Adjusting the chunk size affects both the recall and precision of the results. Adequate chunk size is essential to ensure a balance between the retrieval and generation tasks in RAG, as over- or undersized chunks can negatively impact one or both components. In the tests, 500-character chunk sizes were found to give the best results. In this particular implementation, no metadata added (document type or section labels) is used in the vectorstore creation process, which would facilitate more targeted processing when using smaller chunks.

GPU usage and RAM size obviously affect the performance of the models. It is evident from the results that hardware configurations that do not use the GPU perform significantly slower on the text generation and summarization tasks. Models with fewer parameters (up to 13b) can run reasonably well on 16 GB RAM configurations. Larger models need more resources, both in terms of RAM and GPU resources. This is the reason, in this particular implementation, to use the selected small LLMs, which are suitable for standard and commonly available hardware configurations and platforms.

It is important to note that the choice of a model may vary depending on the specific requirements of a given task, including the desired balance between creativity and accuracy,

the importance of fluency versus content fidelity, and the computational resources available. Therefore, while Mistral:7b appears to be the most versatile and capable model based on the provided metrics, the selection of a model should be guided by the specific objectives and constraints of the application in question.

While promising, the use of RAG-equipped LLMs requires caution regarding data accuracy, privacy concerns, and ethical implications. This is of particular importance in the healthcare domain, where the goal is to assist medical professionals and researchers by accessing the latest medical research, clinical guidelines, and patient data, as well as assisting diagnostic processes, treatment planning, and medical education. Pre-trained open-source models can be found on Huggingface [45]. For example, TheBloke/medicine-LLM-13B-GPTQ is used for medical question answering, patient record summarization, aiding medical diagnosis, and general health Q&A [46]. Another model is m42-health/med42-70b [47]. However, this application requires measures to ensure accuracy, privacy, and compliance with health regulations.

6. Conclusions

This paper presented the development, integration, and use of the PaSSER web application, designed to leverage RAG technology with LLMs for enhanced document retrieval and analysis. Despite the explicit focus on smart agriculture as the chosen specific domain, the application can be used in other areas.

The web application integrates Mistral:7b, Llama2:7b, and Orca2:7b LLMs, selected for their performance and compatibility with medium computational capacity hardware. It has built-in testing modules that evaluate the performance of the LLMs in real-time by a set of 13 evaluation metrics (ROUGE-1 recall, precision, f-score; ROUGE-l recall, precision, f-score; BLUE, Laplace perplexity, Lidstone perplexity, cosine similarity, Pearson correlation, F1 score).

The LLMs were tested via the ***'Q&A Time LLM Test'*** and ***'RAG Q&A score test'*** functionalities of the PaSSER App. The ***'Q&A Time LLM Test'*** was focused on assessing LLMs across two hardware configurations. From the results of the ***'Q&A Time LLM Test'***, it can be concluded that even when working with 7b models, the presence of GPUs is crucial for text generation speed. The lowest total duration times were shown by Orca2:7b on the Mac M1 system. From the results of the ***'RAG Q&A Score Test'*** applied to the selected metrics over the dataset of 446 question–answer pairs, the Mistral:7b model exhibited superior performance.

The PaSSER App leverages a private, permissionless Antelope blockchain network for documenting and verifying results from LLMs' testing. The system operates on a token-based economy (SYS) to manage RAM, CPU, and NET resources. RAM usage and associated costs, measured in SYS and USD, indicate that the total cost for blockchain resources for conducted tests remains below 50 USD. This pricing model guarantees reliability and predictability by facilitating accurate cost estimations and efficient resource distribution. Beyond the monetary aspects, the value of implementing blockchain encompasses increased transparency, security, and traceability, highlighting its benefits.

Future development will focus on leveraging other pre-trained open-source LLMs (over 40b) LLMs, exploring fine-tuning approaches, and further integration in the existing Antelope blockchain/IPFS infrastructure of the SCPDx platform.

Author Contributions: Conceptualization, I.R. and I.P.; methodology, I.R. and I.P.; software, I.R. and M.D.; validation, I.R. and M.D.; formal analysis, I.P. and I.R.; investigation, L.D.; resources, L.D.; data curation, I.R. and M.D.; writing—original draft preparation, I.R. and I.P.; writing—review and editing, I.R., I.P. and M.D.; visualization, I.R. and M.D.; supervision, I.P.; project administration, L.D.; funding acquisition, I.R, I.P. and L.D. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Bulgarian Ministry of Education and Science under the National Research Program "Smart crop production" approved by the Ministry Council No. 866/26.11.2020.

Data Availability Statement: All data and source codes are available at: "https://github.com/scpdxtest/PaSSER (accessed on 1 April 2024). Git Structure: *'README.md'*—information about the project and instructions on how to use it; *'package.json'*- the list of project dependencies and other metadata; *'src'*- all the source code for the project; *'src/components'*—all the React components for the project; *'src/components/configuration.json'*—various configuration options for the app; *'src/App.js'*—the main React component that represents the entire app; *'src/index.js'*—JavaScript entry point file; *'public'*—static files like the 'index.html' file; *'scripts'*—Python backend scripts; *'Installation Instructions.md'*—contains instructions on how to install and set up the project.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Howard, J.; Ruder, S. Universal Language Model Fine-Tuning for Text Classification. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Melbourne, Australia, 1 July 2018. [CrossRef]
2. Houlsby, N.; Giurgiu, A.; Jastrzebski, S.; Morrone, B.; De Laroussilhe, Q.; Gesmundo, A.; Attariyan, M.; Gelly, S. Parameter-Efficient Transfer Learning for NLP. No. 97. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; Chaudhuri, K., Salakhutdinov, R., Eds.; pp. 2790–2799.
3. Brown, T.B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language Models Are Few-Shot Learners. *arXiv* **2005**, arXiv:2005.14165. Available online: https://arxiv.org/abs/2005.14165v4 (accessed on 26 March 2024).
4. Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.; Rocktäschel, T.; et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. *arXiv* **2005**, arXiv:2005.11401. Available online: http://arxiv.org/abs/2005.11401 (accessed on 2 February 2024).
5. Gao, Y.; Xiong, Y.; Gao, X.; Jia, K.; Pan, J.; Bi, Y.; Dai, Y.; Sun, J.; Guo, Q.; Wang, M.; et al. Retrieval-Augmented Generation for Large Language Models: A Survey. *arXiv* **2023**, arXiv:2312.10997. Available online: http://arxiv.org/abs/2312.10997 (accessed on 18 February 2024).
6. Karpukhin, V.; Oguz, B.; Min, S.; Lewis, P.; Wu, L.; Edunov, S.; Chen, D.; Yih, W. Dense Passage Retrieval for Open-Domain Question Answering. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online, 1 November 2020. [CrossRef]
7. Guu, K.; Lee, K.; Tung, Z.; Pasupat, P.; Chang, M. Retrieval Augmented Language Model Pre-Training. *Proc. Mach. Learn. Res.* **2020**, *119*, 3929–3938.
8. Izacard, G.; Grave, E. Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering. In Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, Online, 20 April 2021. [CrossRef]
9. GitHub. GitHub—Scpdxtest/PaSSER. Available online: https://github.com/scpdxtest/PaSSER (accessed on 8 March 2024).
10. Popchev, I.; Doukovska, L.; Radeva, I. A Framework of Blockchain/IPFS-Based Platform for Smart Crop Production. In Proceedings of the ICAI'22, Varna, Bulgaria, 6–8 October 2022. [CrossRef]
11. Popchev, I.; Doukovska, L.; Radeva, I. A Prototype of Blockchain/Distributed File System Platform. In Proceedings of the IEEE International Conference on Intelligent Systems IS'22, Warsaw, Poland, 12–14 October 2022. [CrossRef]
12. IPFS Docs. IPFS Documentation. Available online: https://docs.ipfs.tech/ (accessed on 25 March 2024).
13. GitHub. Antelope. Available online: https://github.com/AntelopeIO (accessed on 11 January 2024).
14. Ilieva, G.; Yankova, T.; Radeva, I.; Popchev, I. Blockchain Software Selection as a Fuzzy Multi-Criteria Problem. *Computers* **2021**, *10*, 120. [CrossRef]
15. Radeva, I.; Popchev, I. Blockchain-Enabled Supply-Chain in Crop Production Framework. *Cybern. Inf. Technol.* **2022**, *22*, 151–170. [CrossRef]
16. Popchev, I.; Radeva, I.; Doukovska, L. Oracles Integration in Blockchain-Based Platform for Smart Crop Production Data Exchange. *Electronics* **2023**, *12*, 2244. [CrossRef]
17. Ollama. Available online: https://ollama.com. (accessed on 25 March 2024).
18. GitHub. GitHub—Chroma-Core/Chroma: The AI-Native Open-Source Embedding Database. Available online: https://github.com/chroma-core/chroma (accessed on 26 February 2024).
19. PrimeReact. React UI Component Library. Available online: https://primereact.org (accessed on 25 March 2024).
20. WharfKit. Available online: https://wharfkit.com/ (accessed on 25 March 2024).
21. LangChain. Available online: https://www.langchain.com/ (accessed on 25 March 2024).
22. NLTK: Natural Language Toolkit. Available online: https://www.nltk.org/ (accessed on 26 February 2024).

23. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 8024–8035.
24. NumPy Documentation—NumPy v1.26 Manual. Available online: https://numpy.org/doc/stable/ (accessed on 26 February 2024).
25. Paul Tardy. Rouge: Full Python ROUGE Score Implementation (Not a Wrapper). Available online: https://github.com/pltrdy/rouge (accessed on 1 April 2024).
26. Contributors. T. H. F. Team (Past and Future) with the Help of All Our. Transformers: State-of-the-Art Machine Learning for JAX, PyTorch and TensorFlow. Available online: https://github.com/huggingface/transformers (accessed on 1 April 2024).
27. SciPy Documentation—SciPy v1.12.0 Manual. Available online: https://docs.scipy.org/doc/scipy/ (accessed on 26 February 2024).
28. Pyntelope. PyPI. Available online: https://pypi.org/project/pyntelope/ (accessed on 27 February 2024).
29. Rastogi, R. Papers Explained: Mistral 7B. DAIR.AI. Available online: https://medium.com/dair-ai/papers-explained-mistral-7b-b9632dedf580 (accessed on 24 October 2023).
30. ar5iv. Mistral 7B. Available online: https://ar5iv.labs.arxiv.org/html/2310.06825 (accessed on 6 March 2024).
31. The Cloudflare Blog. Workers AI Update: Hello, Mistral 7B! Available online: https://blog.cloudflare.com/workers-ai-update-hello-mistral-7b (accessed on 6 March 2024).
32. Hugging Face. Meta-Llama/Llama-2-7b. Available online: https://huggingface.co/meta-llama/Llama-2-7b (accessed on 6 March 2024).
33. Mitra, A.; Corro, L.D.; Mahajan, S.; Codas, A.; Ribeiro, C.S.; Agrawal, S.; Chen, X.; Razdaibiedina, A.; Jones, E.; Aggarwal, K.; et al. Orca-2: Teaching Small Language Models How to Reason. *arXiv* **2023**, arXiv:2311.11045.
34. Popchev, I.; Radeva, I.; Dimitrova, M. Towards Blockchain Wallets Classification and Implementation. In Proceedings of the 2023 International Conference Automatics and Informatics (ICAI), Varna, Bulgaria, 5–7 October 2023. [CrossRef]
35. Chen, J.; Lin, H.; Han, X.; Sun, L. Benchmarking Large Language Models in Retrieval-Augmented Generation. *arXiv* **2023**, arXiv:2309.01431. [CrossRef]
36. Banerjee, S.; Lavie, A. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization, Ann Arbor, MI, USA, 22 June 2005.
37. Lin, C.-Y. *ROUGE: A Package for Automatic Evaluation of Summaries*; Association for Computational Linguistics: Barcelona, Spain, 2004.
38. Papineni, K.; Roukos, S.; Ward, T.; Zhu, W.-J. Bleu: A Method for Automatic Evaluation of Machine Translation. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, PA, USA, 6–12 July 2002. [CrossRef]
39. Arora, K.; Rangarajan, A. Contrastive Entropy: A New Evaluation Metric for Unnormalized Language Models. *arXiv* **2016**, arXiv:1601.00248. Available online: https://arxiv.org/abs/1601.00248v2 (accessed on 2 February 2024).
40. Jurafsky, D.; Martin, J.H. Speech and Language Processing. Available online: https://web.stanford.edu/~jurafsky/slp3/ (accessed on 8 February 2024).
41. Li, B.; Han, L. *Distance Weighted Cosine Similarity Measure for Text Classification*; Springer: Berlin/Heidelberg, Germany, 2013.
42. Sokolova, M.; Japkowicz, N.; Szpakowicz, S. Beyond Accuracy, F-Score and ROC: A Family of Discriminant Measures for Performance Evaluation. *Adv. Artif. Intell.* **2006**, *4304*, 1015–1021.
43. issuu. Bancor Protocol Whitepaper En. Available online: https://issuu.com/readthewhitepaper/docs/bancor_protocol_whitepaper_en (accessed on 24 March 2024).
44. Medium; Binesh, A. EOS Resource Usage. Available online: https://medium.com/shyft-network/eos-resource-usage-f0a8098827d7 (accessed on 24 March 2024).
45. Hugging Face. Models. Available online: https://huggingface.co/models (accessed on 23 March 2024).
46. Cheng, D.; Huang, S.; Wei, F. Adapting Large Language Models via Reading Comprehension. *arXiv* **2024**, arXiv:2309.09530. [CrossRef]
47. Hugging Face. M42-Health/Med42-70b. Available online: https://huggingface.co/m42-health/med42-70b (accessed on 26 March 2024).

electronics

MDPI

Article

Generative Adversarial Network Models for Augmenting Digit and Character Datasets Embedded in Standard Markings on Ship Bodies

Abdulkabir Abdulraheem, Jamiu T. Suleiman and Im Y. Jung *

School of Electronic and Electrical Engineering, Kyungpook National University, Daegu 41566, Republic of Korea; aaoabdul@gmail.com (A.A.); jamiu.suleiman111@gmail.com (J.T.S.)
* Correspondence: iyjung@ee.knu.ac.kr

Abstract: Accurate recognition of characters imprinted on ship bodies is essential for ensuring operational efficiency, safety, and security in the maritime industry. However, the limited availability of datasets of specialized digits and characters poses a challenge. To overcome this challenge, we propose a generative adversarial network (GAN) model for augmenting the limited dataset of special digits and characters in ship markings. We evaluated the performance of various GAN models, and the Wasserstein GAN with Gradient Penalty (WGAN-GP) and Wasserstein GAN with divergence (WGANDIV) models demonstrated exceptional performance in generating high-quality synthetic images that closely resemble the original imprinted characters required for augmenting the limited datasets. And the evaluation metric, Fréchet inception distance, further validated the outstanding performance of the WGAN-GP and WGANDIV models, establishing them as optimal choices for dataset augmentation to enhance the accuracy and reliability of recognition systems.

Keywords: data augmentation; generative adversarial networks; Fréchet inception distance; ship-marking characters and digits

1. Introduction

The recognition of characters and digits imprinted on ship bodies is of significant importance owing to their distinctive features and role in conveying crucial information. Ship markings, governed by standardized regulations, serve as identifiers offering crucial operational details [1–4]. Automatic recognition of these characters is essential for various reasons. It facilitates efficient ship documentation and tracking, enhancing maritime safety and security. Moreover, it aids in identifying vessels involved in incidents or illegal activities, supporting investigations and law enforcement. In the unfortunate event of accidents, accurate character identification provides insights for investigations and reconstructions. Imprinted character images from damaged components reveal origins and potential contributing factors. Precision in identification is vital for precise tracking, maintenance, and replacement of ship parts.

Figure 1 illustrates old ship markings, representative of our dataset sources. The dataset comprises cropped images of degraded numbers and letters from ship imprints. These real-world scenarios require recognition models resilient to degraded imprints. Unlike larger and well-maintained ships that often repaint their identification markings, older or smaller vessels face corrosion and fading, posing challenges to recognition systems. Seawater and environmental factors worsen the degradation.

Figure 1. Old ship markings from sources similar to those of our dataset.

In our work, we adopt data augmentation, utilizing cutting-edge generative adversarial networks (GANs), as a promising approach, to enhance the automatic identification of imprinted characters on ships. By specifically employing GANs for data augmentation, we present a pragmatic solution tailored to scenarios where traditional augmentation methods might fall short. This is especially relevant when machine learning-based algorithms are utilized for recognition tasks. Our study thus bridges the gap between data scarcity and machine learning, offering a new perspective that can have far-reaching implications. Additionally, the significance of our work extends beyond the maritime sector. Our proposed methodology, which revolves around selecting the most suitable GAN model for a specific dataset and evaluating its performance for data augmentation, has the potential to address data scarcity challenges across diverse domains in the field of Information Technology. For instance, in domains such as medical imaging, where acquiring large datasets can be a challenge due to ethical or logistical constraints, our approach could be adapted to enhance the quality and diversity of available data [5–7].

Figure 2 depicts a whole-system-architecture chart that illustrates the information flow and operations of a system that utilizes data augmentation for enhancing the accuracy of ship-character identification and retrieval. The ship-character recognition system comprises several key components, including input data, a data augmentation sub-system, an augmented dataset, a machine learning model for classification, and a retrieval client interface. The input data comprise special alphanumeric characters found on ship components. The data augmentation module employs state-of-the-art GAN techniques to create variations of the original images. The augmented dataset comprises the synthetic images produced by the GAN. The machine learning model learns to identify the imprinted characters on the ship components using the augmented dataset. Furthermore, the retrieval system retrieves relevant ship-component information, such as part numbers, specifications, and maintenance history based on the identified characters and digits.

The subsequent sections of this paper are structured as follows: First, we review and analyze previous studies on augmentation techniques in Section 2 for extending object detection in maritime images. Then, we present our methodology in Section 3, outlining the experimental setup and procedures employed to evaluate the performance of cutting-edge GAN models on ship-marking-character and -digit datasets. Following that, we provide a comprehensive evaluation of our results, analyzing the outcomes and comparing the generated character and digit images. We also discuss the limitations encountered during the study, shedding light on potential constraints and areas for improvement. Furthermore, we present the potential direction of future research suggesting enhancements in the proposed approach. Lastly, we conclude this article by summarizing our findings and highlighting the significance of our research in advancing the augmentation of ship characters.

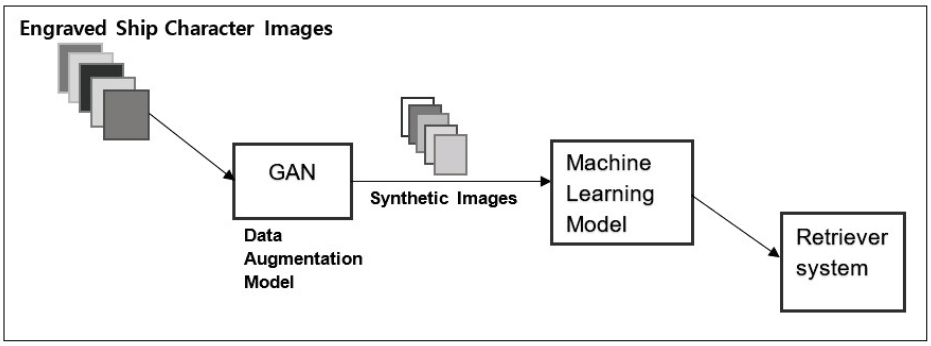


Figure 2. Ship-character image recognition process.

2. Related Work

Extensive research focused on achieving reliable detection and recognition of ship images utilizing different techniques has been performed. While considerable progress has been made, our work contributes to this research area by providing an extensive and diverse dataset for training recognition models. We reviewed previous studies on ship identification and data augmentation with the aim to gain valuable insights into the effectiveness of data augmentation approaches and their effect on improving ship-character identification. Some of the works mentioned here explore the use of convolutional neural networks (CNNs) and GANs in the ship application domains.

In [8], the authors addressed the important issue of ship-type identification in maritime surveillance. They highlighted the challenges in building large-scale marine-environment datasets, where data collection and security concerns limit the availability of comprehensive data. To overcome these limitations, the authors proposed a novel approach utilizing GANs for data augmentation. By augmenting a small number of real ship images, they improved fine-grained ship classification performance and demonstrated the effectiveness of augmented data in training ship classification networks. This research demonstrates the potential of augmented data for enhancing ship identification and classification for maritime surveillance. Ref. [9] proposed a data augmentation method for extending object detection datasets in maritime images. Their approach involved extracting the mask of the foreground object and combining it with a new background to generate location information and additional data. This technique aimed to enhance the learning process by incorporating diverse and high-quality data features. Further, experimental evaluation demonstrated the effectiveness of their method in improving the performance and robustness of object detection models specifically tailored to maritime imagery. Ref. [10] introduced BoxPaste, a powerful data augmentation method tailored for ship detection in Synthetic Aperture Radar (SAR) imagery. Their approach involves pasting ship objects from one SAR image onto another, thereby achieving considerable performance improvements in the SAR ship-detection dataset compared with baseline methods. They also proposed a principle for designing SAR ship detectors, emphasizing the potential benefits of lighter models. The integration of their data augmentation scheme with RetinaNet [11] and Adaptive Training Sample Selection (ATSS) [12] further demonstrates its effectiveness, resulting in impressive performance gains.

In [13], the authors proposed a modification to the Faster R-CNN object detection network to tackle the challenge of multiscale ships in SAR images. By incorporating the constant false-alarm-rate algorithm and re-evaluating low-scoring bounding boxes, the proposed method achieved improved detection performance. This work contributes to the advancement of SAR ship detection using deep learning methods and provides valuable insights for addressing the multiscale-ship-detection problem. Ref. [14] proposed a densely connected multiscale neural network based on the faster R-CNN framework for multiscale and multiscene SAR ship detection. Their method addressed the challenges in detecting small-scale ships and handling complex backgrounds in SAR images. By densely connecting feature maps and introducing a training strategy to focus on hard examples,

their approach achieved excellent performance in multiscale SAR ship detection across various scenes.

Data augmentation across multiple domains and the use of simple models trained on large datasets can be highly beneficial for the performance improvement of object detection applications. The effectiveness of this approach in improving the accuracy and robustness of object detection algorithms was demonstrated in [15–17]. By augmenting available data with various transformations, such as rotations, translations, and noise addition, the models can better generalize and exhibit improved detection capabilities across various scenarios and variations in the input data. Ref. [18] addressed the challenge of training deep learning models that require a large number of images by employing data augmentation as a preprocessing step. They resized dataset images to a uniform size of 256 $\times$ 256 pixels and applied various augmentation techniques, such as right shift, image flipping, and left shift. These methods increased dataset diversity and improved the performance of the classifier. Image flipping, in particular, allowed for modifications in the pixel location of pixels, enhancing the variations within the dataset and enabling more robust model training. The authors in [19] addressed the problem of insufficient data by utilizing Conditional Wasserstein GAN-Gradient Penalty (CWGAN-GP), and DenseNet and ResNet. They employed GANs to generate underwater sonar images and expanded the dataset. GANs have gained considerable attention due to their ability to learn complex data distributions in high-dimensional spaces. By employing CWGAN-GP&DR, the authors addressed the overfitting issue and successfully expanded the dataset for improved model training. Ref. [20] introduced several data augmentation techniques for improving palimpsest character recognition using deep neural networks. Palimpsests are manuscripts with overlaid text that makes recognizing the underlying characters challenging. The authors proposed four augmentation methods—random mask overlay, random rotation, random scaling, and random noise addition. These methods were evaluated on a palimpsest dataset, and they observed that the random mask overlay method achieved the best performance, improving character recognition accuracy by up to 10%. These findings highlight the effectiveness of data augmentation in enhancing the performance of character recognition algorithms for palimpsest manuscripts.

In [21], the authors proposed a generative and discriminative model-based approach for information retrieval. Their minimax game model generates text queries that are relevant to a given document and learns to discriminate between relevant and irrelevant documents. The model achieved state-of-the-art results in various information retrieval tasks.

While different from our approach, the methods in these papers provide valuable insights into scene text detection, recognition, and segmentation. Furthermore, these studies showcase the progress made in understanding and processing textual information in complex visual environments, contributing to our understanding of character recognition and augmenting our knowledge in the domain of ship-character data augmentation. Our approach emphasizes the importance of equipping recognition models with abundant data from diverse character datasets collected from multiple ships and ship-body images. This extensive dataset encompasses various ship markings, allowing the recognition model to effectively learn and generalize across different ship types and marking variations. By leveraging this rich dataset, we aim to enhance the accuracy and robustness of ship-character recognition, enabling more effective ship identification mechanisms in real-world scenarios.

3. Dataset and Data Augmentation Methods

This section provides an overview of the datasets, the state-of-the-art GAN techniques, and the evaluation techniques employed in this work and offers valuable insights into the intricacies of the methodology and its relevance in the field of ship-character recognition. The presentation of datasets and the demonstration of the utilization of advanced GAN

techniques enhance the understanding of the nuances of our methodology and emphasize its importance in ship-character analysis and recognition.

3.1. Datasets

The dataset used in the experiments comprises ship-character images (0–9 digits and 13 letters (A, C, D, E, I, L, M, N, O, P, R, S, and T)) obtained from old or poorly maintained ships. These images were carefully selected to support ship-character identification and retrieval systems. These images exhibit various characteristics to capture the diverse ship markings found on different parts of body and engine components, as depicted in Figure 3.

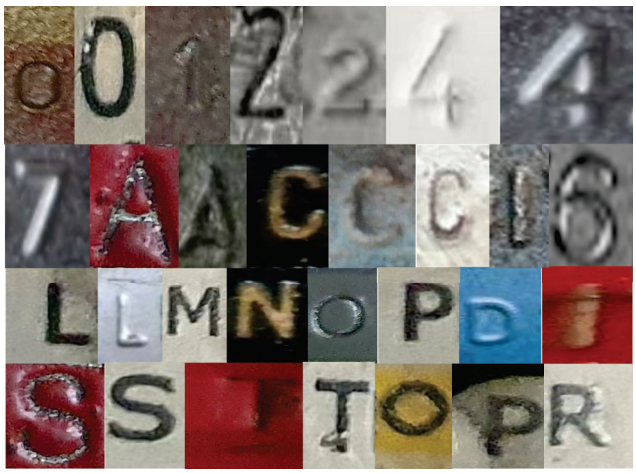


Figure 3. Collection of character and digit samples from the source dataset.

The images in the dataset exhibit variations in size and color, but they were preprocessed to ensure consistency during training and analysis. Specifically, they were normalized to grayscale and resized to 56 × 56 pixels in width and height. The images are stored in standard formats such as JPEG or PNG. The dataset encompasses various engraving styles commonly found on ships, including embossed, engraved, and painted characters, either individually or in combination. These characters represent ship identification numbers, hull markings, engine-component identifiers, and other characters relevant to ship operations. This curated dataset serves as the foundation for evaluating and comparing the performance of GAN models in generating synthetic ship-character images, thereby enabling the development of accurate and reliable ship technology applications.

3.2. State-of-the-Art GANs

We selected specific GAN models for comparison based on their application areas and their ability to generate high-quality images in a shorter time frame compared with photorealistic GAN models. Our selection took into account the practicality and efficiency of generating synthetic images for our research goals. Augmenting data using GANs and then using them for network training is a considerably useful method for avoiding infringement of personal information and security problems in data [8]. Our experiment was performed using the following GAN models:

- GAN [22]: GAN is a fundamental model in which a generator and discriminator are trained in an adversarial manner. The generator aims to produce synthetic samples, while the discriminator distinguishes between real and fake samples. GANs have demonstrated their ability to generate realistic data across various domains.
- Auxiliary Classifier GAN (AC-GAN) [23]: AC-GAN extends the conditional GAN framework by having the discriminator predict the class label of an image instead of receiving it as input. This approach stabilizes training, allows the generation of large, high-quality images, and promotes a latent space representation independent of the class label.

- Boundary-Seeking GAN (BGAN) [24]: BGAN focuses on learning the manifold boundary of the real data distribution by minimizing the classification error of the discriminator near the decision boundary. This encourages the generator to generate samples that lie on the data manifold, resulting in higher-quality and more realistic generated samples.
- Boundary Equilibrium GAN (BEGAN) [25]: BEGAN optimizes a lower bound of the Wasserstein distance using an autoencoder as the discriminator. It maintains equilibrium between generator and discriminator using an additional hyperparameter.
- Deep Convolutional GAN (DCGAN) [26]: DCGAN utilizes CNNs as the generator and discriminator. It introduces architectural constraints to ensure the stable training of CNN-based GANs and demonstrates competitive performance in image classification tasks.
- Wasserstein Generative Adversarial Network (WGAN) [15]: WGAN utilizes the Wasserstein distance to measure the discrepancy between real and generated data distributions. It introduces a critic network and focuses on optimizing the Wasserstein distance for stable training.
- WGAN with GP (WGAN-GP) [27]: WGAN-GP proposes a GP to enforce the Lipschitz constraint in the discriminator, replacing the weight clipping used in WGAN. This penalty improves stability, prevents issues such as mode collapse, and eliminates the need for batch normalization.
- Wasserstein divergence (WGANDIV) [28]: WGANDIV approximates Wasserstein divergence; exhibits stability in training, including progressive growing training; and has demonstrated superior quantitative and qualitative results.
- Deep Regret Analytic GAN (DRAGAN) [29]: DRAGAN applies a GP similar to WGAN-GP but with a focus on real data manifold. Even though DRAGAN is similar to WGAN-GP, it exhibits slightly less stability compared with WGAN-GP.
- Energy-based GAN (EBGAN) [30]: EBGAN models the discriminator as an energy function that assigns low energies to regions near the data manifold. It focuses on capturing regions close to the data distribution.
- FisherGAN [31]: FisherGAN introduces GAN loss based on the Fisher information matrix, maximizing Fisher information to encourage diverse and high-quality sample generation. It improves mode coverage and sample quality, enhancing the performance of GANs in generating realistic and varied data.
- InfoGAN [32]: InfoGAN extends the GAN framework by introducing an additional latent variable that captures the interpretable factors of variation in the generated data. By maximizing the mutual information between this latent variable and the generated samples, InfoGAN enables explicit control over specific attributes of the generated data. It promotes disentangled representations and targeted generation.
- Least-squares GAN (LSGAN) [33]: LSGAN addresses the vanishing gradient problem using the least-squares (L2) loss function instead of cross-entropy. It stabilizes the training process and produces visuals that closely resemble real data.
- MMGAN and Non-Saturating GAN (NSGAN) [34]: NSGAN simultaneously trains the generator (G) and discriminator (D) models. The objective is to maximize the probability of D making a mistake. NSGAN differs from MMGAN in its generator loss. Furthermore, the output of G can be interpreted as a probability.
- RELATIVISTIC GAN (REL-GAN) [35]: It introduces a relativistic discriminator that compares real and generated samples in a balanced manner by considering their relative ordering. This approach reduces bias toward either real or fake samples, resulting in improved training stability and generation quality.
- SGAN [36]: SGAN maintains statistical independence between multiple adversarial pairs, addresses limitations in representational capability, and exhibits improved stability and performance compared with standard methods. SGAN is suitable for various applications and produces a single generator. Future extensions can explore diversity between pairs and consider multiplayer game theory.

We implemented all of the GANs listed above by adapting and optimizing them for our dataset, executed them for our dataset, evaluated the results using well-established metrics, and then selected the GAN models that produced diverse and high-quality images of the imprinted characters. The selected models were used to generate additional images of imprinted characters. During the training phase, our focus was on characters (specifically, the letters A, C, D, E, I, L, M, N, O, P, R, S, and T) and digits (0–9). These specific characters and digits were chosen because of their easy availability and the convenience of collecting them from various sources. These characters and digits were grouped into classes according to each character or digit, and each class was trained for a specific number of epochs—20,000 and then 50,000 epochs. During each epoch, the GAN models processed each training example in the dataset, calculated the loss, and updated the model parameters using the chosen optimization algorithm. The number of epochs determined the frequency at which the model iteratively updated its parameters to learn from the data. An epoch was considered complete when all the training examples had been used once for parameter updates. Employing multiple epochs is a common practice for improving the performance of the model by allowing it to learn from the data multiple times. To assess the performance of the GAN models and provide a rationale for their selection, we performed visual inspections of the generated images [37–39]. We implemented each GAN model based on the original design proposed by the respective authors, with minimal hyperparameter tuning. Our objective was to identify the most suitable and efficient GAN model for our specific use case. For the training process, we trained each GAN model for 20,000 and 50,000 epochs. A relatively high number of epochs was selected to ensure adequate learning and convergence of the models. Throughout the training process, image outputs at the 50th and 100th iterations were generated to monitor the progress and visually assess the generated samples. Consistency with the original recommendations was maintained as closely as possible. However, certain GAN models had unique parameters and architectural variations. Our focus was to maintain consistency with the recommended settings and focus on evaluating the overall performance and quality of the generated images across the different GAN models.

3.3. Evaluation Metrics

We evaluated the results using well-established metrics, and the GAN models that exhibited both diversity and high-quality images of the imprinted characters were selected. Subsequently, the chosen models were utilized to generate additional images of imprinted characters.

Evaluating the quality and fidelity of generated images in a GAN presents unique challenges due to the absence of a universal discriminator for fair comparisons. When assessing GAN performance, two primary properties—fidelity and diversity—must be considered. Fidelity refers to the realism and visual quality of the generated images, taking into account factors such as image clarity and resemblance to real samples, whereas diversity measures the range and variety of images produced by the generator, ensuring that it captures the entire scope of the training data or desired modeling class. Evaluating fidelity involves comparing generated samples to their closest real counterparts and analyzing the overall distribution of fake versus real images. Diversity evaluations require the evaluation of the ability of the GAN model to generate diverse images rather than producing a single realistic but limited output. Striking a balance between fidelity and diversity is crucial, as a successful GAN should consistently generate high-quality images while covering a wide range of possibilities. However, accurately quantifying these properties remains a challenge, particularly without relying on memorizing the training dataset. By considering fidelity and diversity, evaluators can gain valuable insights into the performance of the GAN model and its capability to generate convincing and varied fake images.

Visual examination of samples is one of the most common and intuitive ways to evaluate GANs. However, it has several limitations, including the reviewer's biases toward the model, its configuration, and the project objectives [40]. In addition, visual examination

requires knowledge of what is realistic and what is not for the target domain, and it is limited to the number of images that can be reviewed in a reasonable time.

The evaluation of GAN models encompasses various methods, such as Fréchet inception distance (FID) [41], Inception Score (IS) [37], and precision and recall [42]. These metrics provide valuable insights into different aspects of the generated images, including their quality, diversity, and resemblance to real data.

The IS is a commonly used metric for evaluating the quality and diversity of generated images [37]. A higher score indicates better performance with low entropy in the conditional probability distribution and high entropy in the marginal probability distribution. However, the IS has several limitations. It can be easily manipulated or exploited to achieve high scores by generating one real image per classifier class, resulting in a lack of diversity. Furthermore, it solely considers the generated samples and does not compare them to real images. The proxy statistics used in the calculation may not accurately reflect real-world performance and are dependent on the tasks and capabilities of the classifier. Additionally, the IS may not provide precise results when dealing with images containing multiple objects, as it is trained on the ImageNet dataset, which focuses on single-object classification. Given that our imprinted digit dataset does not align with the ImageNet classes, the IS metric may not offer meaningful insights into the quality and diversity of the generated images [43].

The FID is commonly used to evaluate GANs by measuring the similarity between real and generated images based on their embeddings.

$$FID(x, g) = ||\mu_x - \mu_g||_2^2 + Tr(\Sigma_x + \Sigma_g - 2(\Sigma_x \Sigma_g)^{\frac{1}{2}}) \quad (1)$$

As shown in Equation (1), $FID(x, g)$ for a 'multivariate' normal distribution calculates the Fréchet distance and aims for a lower value, indicating better performance [39,41,44,45]. x and g are the real and fake embeddings (activation from the Inception model) assumed to be two multivariate normal distributions. μ_x and μ_g are the magnitudes of vectors x and g. Tr is the trace of the matrix, and Σ_x and Σ_g are the covariance matrices of the vectors. We assessed the quality, fidelity, and overall resemblance of the generated images to the original characters and digits. Additionally, we used the FID as an objective evaluation metric, which allowed us to quantitatively measure the proximity of the generated images to the real images, thereby providing further validation and justification for selecting the preferred GAN model [39,41,44].

A high precision value indicates that the discriminator correctly identifies a large proportion of the generated samples as fake, minimizing false positives. On the other hand, a high recall value indicates that the discriminator correctly identifies a large proportion of the real samples as genuine, minimizing false negatives [42]. These metrics further contributed to the evaluation of GAN performance and the assessment of the ability of discriminators to distinguish between real and generated images.

The IS primarily focuses on the diversity of the generated images and does not consider the efficiency of the generator in approximating the real image distribution. Thus, it is limited in its ability to measure the fidelity to real images. On the other hand, the FID can detect intra-class mode dropping and provides a more comprehensive evaluation metric by considering the quality and diversity of the generated samples. However, the IS and the FID have limitations in detecting overfitting. Precision and recall metrics are impractical for real images as the underlying data manifold is usually unknown, making them only suitable for evaluations on synthetic data, where the ground truth is available. Thus, although precision and recall are relevant metrics in certain scenarios, they are not widely applicable or suitable for assessing the performance of generative models on real-world image datasets [44]. Despite these limitations, the FID is widely used for evaluating generative models due to its robustness, reliability, and consistency in comparing model performance, particularly when dealing with large sample sizes [44]. Particularly, its consistency in relative model comparisons makes it a preferred choice among researchers and practitioners in the field.

We chose the GAN models that demonstrated low FID values and exhibited promising results in terms of generating high-quality images. Moreover, we considered the average training time of the models, taking into account computational efficiency and the practicality of generating images within a reasonable time frame. Owing to the combination of visual inspection and quantitative evaluation metrics, the selected GAN models not only produced visually appealing images but also met the desired efficiency and performance requirements for our research.

To identify the most suitable GAN model for our GAN augmentation task, we performed an extensive evaluation of FID scores for various GANs, including ACGAN, BGAN, BEGAN, DCGAN, DRAGAN, EBGAN, FISHERGAN, GAN, INFOGAN, LSGAN, MMGAN, NSGAN, RELATIVISTIC GAN, SGAN, WGAN, WGAN-GP, and WGANDIV. Each GAN model was trained, and its generated images were subjected to FID analysis. The FID score, which measures the dissimilarity between the feature distributions of real and fake images, was calculated for each GAN. The lower the FID score, the closer the resemblance between the generated and real images, indicating higher image quality [45].

4. Results and Evaluation

In our evaluation, we focused on generating the 13 available letters (A, C, D, E, I, L, M, N, O, P, R, S, and T) and digits (0–9). To achieve optimal performance, we carefully tuned the hyperparameters of the GAN models. The Learning Rate (LR) determines the step size at which the model updates its parameters during training. A lower LR results in more stable training but slower convergence. The batch size refers to the number of samples processed in each iteration. A larger batch size can accelerate training but may require more memory. Dropout is a regularization technique that randomly drops out a fraction of the units of the model during training to prevent overfitting and promote generalization. The number of epochs determines how many times the model will iteratively update its parameters to learn from the data. An epoch is completed when all training examples have been used once for parameter updates. We introduced random noise as input vectors for the generator across all GANs to ensure randomness and diversity. We chose a batch size of 32 to prevent overfitting, considering the size of our dataset. We extended the epoch count from 20,000 to 50,000 in all experiments. We also generated sample outputs periodically for visual assessment. We evaluated the performance of all models using four different learning rates: 0.001, 0.002, 0.0001, and 0.0002. However, we used the Adam optimizer with a learning rate of 0.0002 for both generators and discriminators for WGAN-DIV and WGAN-GP. These adjustments were made based on empirical evaluations to determine the optimal values that result in improved GAN performance. After training and evaluating various models, WGANDIV, WGAN-GP, GAN, and BGAN emerged as the top-performing models, exhibiting exceptional visual appeal in their generated outputs. Upon closer examination, WGAN-GP and WGANDIV exhibited similar characteristics and were visually comparable and better. These two GAN models demonstrated superior performance after being trained for 50,000 epochs, and the optimal output could be achieved around 12,500 epochs.

Tables 1 and 2 present the FID scores achieved by each GAN model when applied to the character and digit datasets, respectively. A lower FID score indicates higher similarity and better quality of the generated samples, reflecting the effectiveness of the GAN model in capturing the underlying characteristics of the dataset. Upon analyzing the results, it was observed that WGANDIV exhibited the lowest FID score among all the evaluated GAN models, indicating its superior image quality. This exceptional performance establishes WGANDIV as the preferred GAN model for our data augmentation task because it excels at generating highly realistic images. Additionally, WGAN-GP also exhibited commendable performance, indicating its effectiveness in producing high-fidelity images. Thus, WGANDIV and WGAN-GP were the top-performing GAN models in our evaluation.

Table 1. FID values for the evaluated GANs.

GAN	A	C	D	E	I	L	M	N	O	P	R	S	T
ACGAN	412.8	482.6	469.5	496.6	414.3	442.6	453.5	455.1	419.7	481.0	407.9	570.5	470.2
BGAN	299.8	298.2	242.0	229.1	311.1	296.9	272.1	294.7	312.5	340.5	268.3	287.6	297.3
BEGAN	402.6	434.3	569.3	398.9	365.9	315.4	453.2	438.6	328.1	304.8	249.2	465.0	320.9
DRAGAN	375.5	472.6	416.4	442.0	512.7	450.8	485.5	469.1	430.9	437.1	472.9	467.9	462.7
EBGAN	581.3	434.9	470.3	400.4	457.3	384.8	434.6	408.9	383.9	401.7	381.0	514.4	436.1
F-GAN	525.7	499.1	441.7	497.1	462.5	530.0	553.3	498.6	447.2	491.7	502.6	551.3	526.2
GAN	242.6	316.3	258.4	218.0	316.8	273.8	280.0	260.9	300.5	232.9	245.8	295.7	229.4
INFOGAN	479.1	481.7	443.7	493.0	529.9	515.2	522.9	484.5	438.9	449.9	492.8	505.5	527.6
LSGAN	393.5	409.8	415.1	411.6	393.8	361.1	377.9	400.8	419.6	356.8	384.4	472.1	463.4
MMGAN	455.9	486.2	419.2	408.4	538.5	495.5	526.4	464.5	466.2	410.1	486.2	488.6	512.5
NSGAN	449.1	478.7	416.0	408.9	519.7	453.6	486.6	459.4	432.4	436.9	463.1	485.8	478.1
REL-GAN	300.1	359.1	383.4	393.7	368.4	385.2	376.5	432.8	434.5	391.9	373.8	468.1	421.3
SGAN	350.4	407.9	424.3	419.5	365.6	372.8	411.3	381.8	349.6	429.4	406.2	493.2	389.0
WGAN	380.3	313.5	349.5	258.5	293.2	288.5	369.9	332.6	324.9	249.7	368.6	337.1	391.5
WGAN-GP	261.7	271.8	188.6	197.8	268.6	236.5	247.8	294.2	230.0	211.2	258.8	307.8	290.3
WGANDIV	231.6	224.8	210.2	215.8	279.1	241.0	252.1	245.5	283.3	213.4	255.0	290.4	285.4

Table 2. FID values for the evaluated GANs.

GAN	0	1	2	3	4	5	6	7	8	9
ACGAN	352.2	442.7	350.9	339.7	381.0	419.9	400.543	383.9	320.8	387.5
BGAN	250.4	278.6	282.0	269.1	314.9	220.4	286.6	293.3	292.9	291.5
BEGAN	312.9	379.8	423.1	367.5	365.9	368.1	404.4	463.9	318.1	346.7
DRAGAN	320.5	355.6	394.0	377.4	375.83	357.3	359.8	357.4	383.4	368.3
EBGAN	321.8	370.4	390.0	377.4	373.4	351.3	443.6	411.7	398.0	373.9
FISHERGAN	417.3	502.9	487.3	477.9	519.3	395.5	421.6	408.1	465.4	522.7
GAN	273.9	251.1	262.2	262.1	281.3	266.8	253.0	284.1	286.1	263.2
INFOGAN	301.1	363.6	393.2	412.8	405.5	406.5	349.7	390.9	412.8	386.4
LSGAN	327.6	372.2	291.4	339.0	356.4	339.4	355.3	349.4	366.8	395.7
MMGAN	438.4	339.8	351.8	349.0	366.3	390.6	322.3	344.1	368.5	412.3
NSGAN	307.5	401.4	400.2	418.8	390.7	419.8	330.4	373.6	388.6	402.7
REL-GAN	333.1	356.1	286.6	340.3	321.5	335.1	411.5	384.3	394.4	414.6
SGAN	322.6	389.7	342.3	362.1	374.5	374.3	389.7	356.5	370.7	391.5
WGAN	247.8	305.2	318.4	353.0	342.0	339.9	383.0	383.0	395.4	330.6
WGAN-GP	224.0	305.0	236.9	229.3	246.9	240.3	231.2	256.0	262.3	289.5
WGANDIV	235.6	289.5	239.9	223.2	358.9	220.7	256.3	270.6	239.5	299.7

The success of WGAN-GP and WGANDIV can be attributed to several factors. First, both models utilize Wasserstein distance as a loss function, which helps address the mode-collapse issue commonly encountered in GAN training. WGAN-GP employs the gradient penalty technique to enforce Lipschitz continuity, promoting stable training and preventing mode collapse. On the other hand, WGANDIV incorporates an additional divergence term that encourages the generator to produce more diverse samples, resulting in improved quality. To further support our evaluation, we performed a detailed visual inspection of the generated samples from the four best GAN models. Based on the original images provided in Figure 4, Figures 5–8 provide representative images showcasing the outputs from BGAN, GAN, WGAN-GP, and WGANDIV, respectively. Upon visual examination, it is evident that the samples generated by WGAN-GP and WGANDIV shown in Figures 7 and 8 exhibit superior quality in terms of capturing the intricate details and characteristics of the imprinted characters and digits. The images from these models demonstrate sharper edges, more pronounced textures, and enhanced overall fidelity compared with the other models with very-low-quality images, as shown in Figures 9–13.

During the evaluation process, we observed variations in the FID scores across various numbers and characters, regardless of the GAN models used. This discrepancy in FID

values can be attributed to the inherent complexity and diversity of the imprinted-character and -digit datasets. Certain digits/characters inherently possess more distinctive features and complex shapes, making their accurate generation challenging. Consequently, the generated samples for these digits/characters may exhibit higher FID values, indicating a greater dissimilarity with respect to the real data distribution. Conversely, numbers/characters with simpler shapes and fewer intricate details may yield lower FID values, indicating better alignment with the real data distribution. Despite the visual similarity between the generated and original images, the high FID score can be attributed to factors such as sensitivity to intra-class mode dropping, smaller sample size, and dataset characteristics [44].

Figure 4. Input samples: 'A', '2', and '0', selected from the original dataset shown in Figure 3.

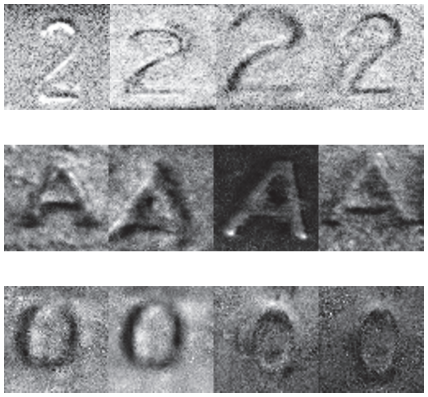


Figure 5. BGAN output samples.

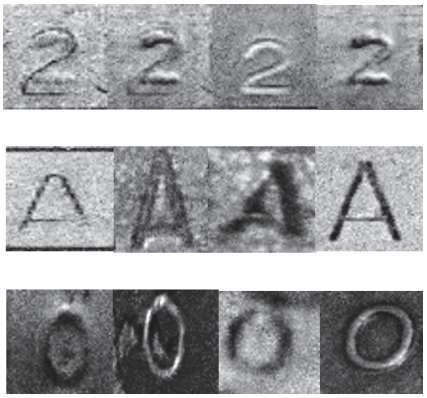


Figure 6. GAN output samples.

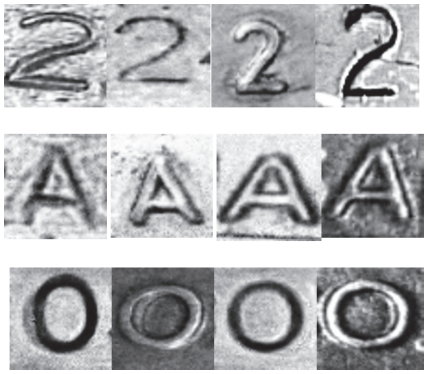


Figure 7. WGAN-GP output samples.

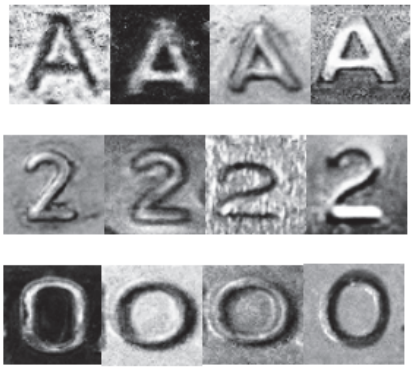


Figure 8. WGANDIV output samples.

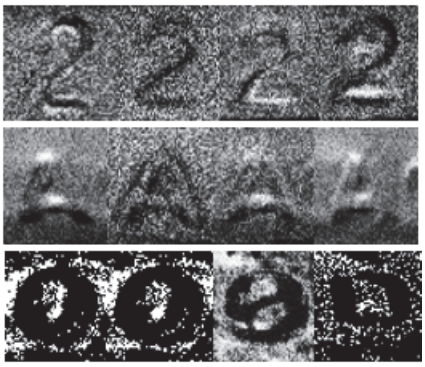


Figure 9. InfoGAN output samples.

Figure 10. LSGAN output samples.

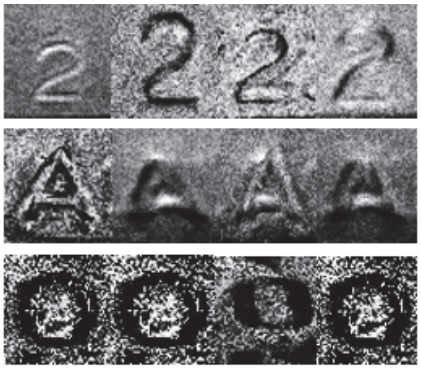


Figure 11. NSGAN output samples.

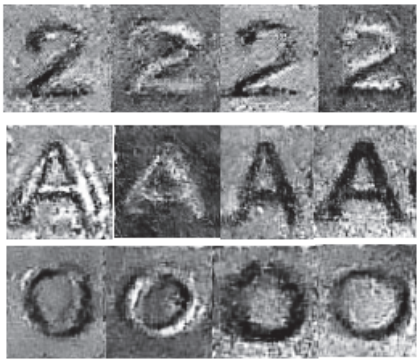


Figure 12. REL-GAN output samples.

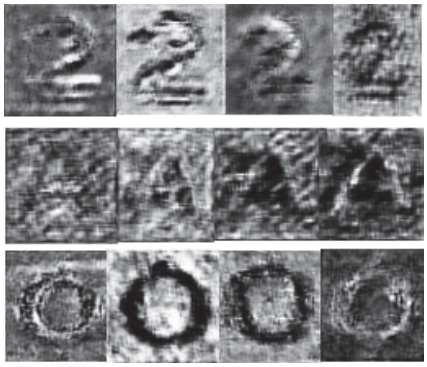


Figure 13. EBGAN output samples.

According to [44], the performance of each model is considerably influenced by the dataset, and there is no model that strictly outperforms the others. Figure 14 shows the original images from which Figures 15 and 16 are generated from. We observed that compared with other letters, certain letters, such as E, L, and I, did not exhibit considerable variations in the generated images across the top-performing GANs. This can be attributed to their simple shapes and low representation of variations in the dataset. The letters E, L, and I possess straightforward and uncomplicated structures, while other letters may have more complex curves and details. The GAN models can easily and accurately generate simpler shapes, resulting in less variation in the generated images for these letters. This finding suggests that the style or structure of each character can influence the diversity of the generated images. Figure 17 shows the selected data samples of the other letters and digits.

Figure 14. Input samples: 'I', 'L', and 'E', selected from the original dataset shown in Figure 3.

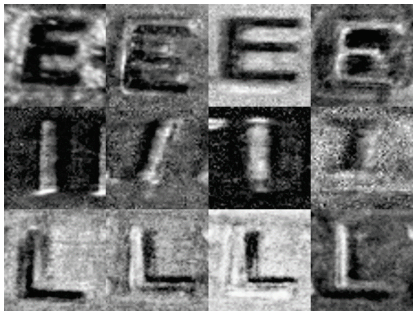


Figure 15. Images of letters I, E, and L from WGANDIV model.

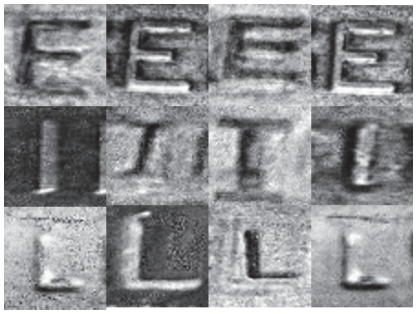


Figure 16. Images of letters I, E, and L from WGAN-GP model.

(**a**) Original letter images selected

(**b**) Selected letter images generated by WGANDIV

Figure 17. *Cont*.

(**c**) Selected letter images generated by WGAN-GP

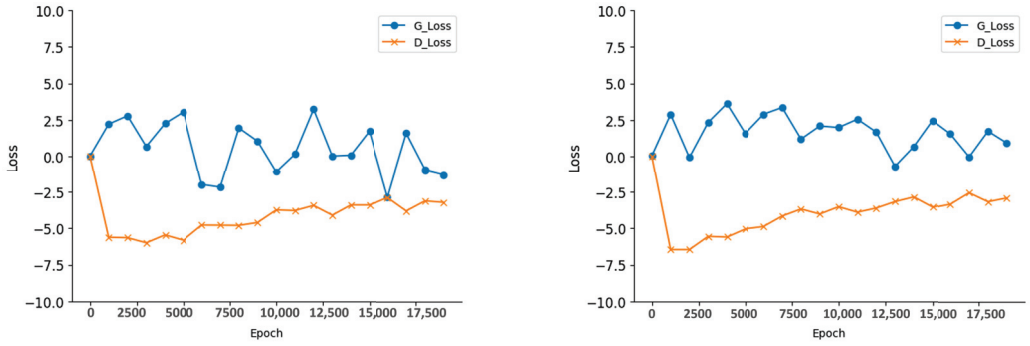
(**d**) Original digit images selected

(**e**) Selected digit images generated by WGANDIV

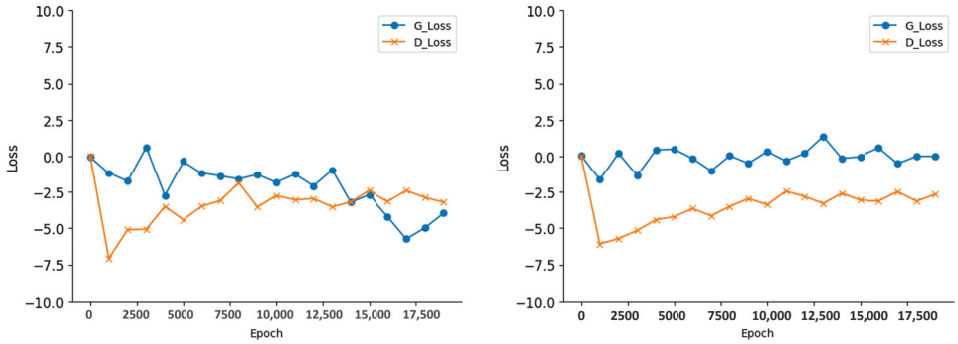
(**f**) Selected digit images generated by WGAN-GP

Figure 17. Selected data samples of the other letters and digits.

Monitoring and analyzing the sampled loss graphs in Figures 18a–d and 19a–d (WGANDIV for '0', WGANDIV for '2', WGANDIV for 'D', WGANDIV for 'A'; WGAN-GP for '1', WGAN-GP for '5', WGAN-GP for 'D', and WGAN-GP for 'R') during training provided valuable insights into the learning process of the GAN models. They revealed how the model adapted and improved over time. By analyzing these plots, we can gain a deeper understanding of the training dynamics and convergence of GAN models. The loss plots of the discriminator and generator depicting changes in the respective losses with the iterative training of the models provide valuable insights into the optimization process of GAN models. At the beginning of training, the discriminator loss is typically high due to the random initialization of the discriminator network. As the generator produces initial samples that lack resemblance to real samples, the discriminator easily distinguishes them as fake, resulting in a high discriminator loss. Simultaneously, the generator loss is also high because the generated samples fail to effectively deceive the discriminator. As training progresses, the discriminator gradually improves its discriminatory capabilities and becomes more proficient at accurately classifying real and generated samples, resulting in a decrease in the discriminator loss. The learning of the discriminator can be observed as a decrease in the slope of the loss curve, indicating the increased ability of the model to differentiate between real and generated samples. Conversely, the generator loss initially decreases as the generator learns to produce more plausible samples that can better deceive the discriminator. With backpropagation, the generator refines its parameters and adjusts its output to generate samples that progressively resemble real samples. Consequently, it becomes increasingly challenging for the discriminator to distinguish between real and generated samples, resulting in a decrease in the generator loss. The convergence of the losses indicates the optimization progress of the GAN models. Ideally, successful training can yield low values for discriminator and generator losses, indicating that the discriminator accurately classifies samples and the generator produces samples that closely resemble real ones. The convergence of the loss curves indicates that the models have reached a stable equilibrium, where the generator effectively captures the underlying data distribution.

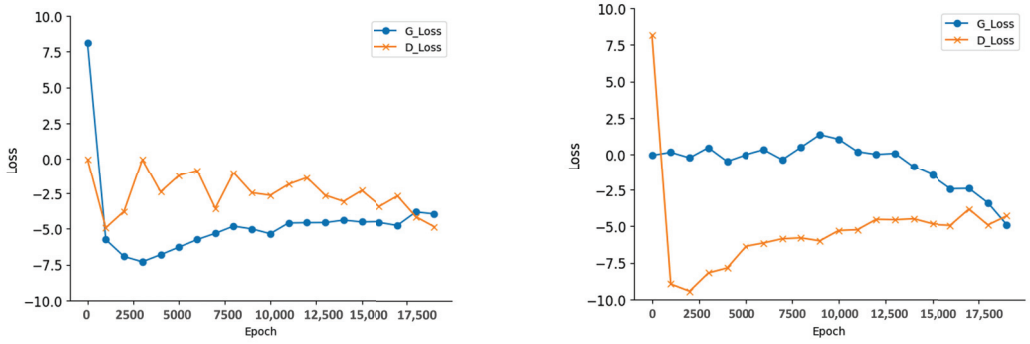


(**a**) WGANDIV-0 loss graph (**b**) WGANDIV-2 loss graph

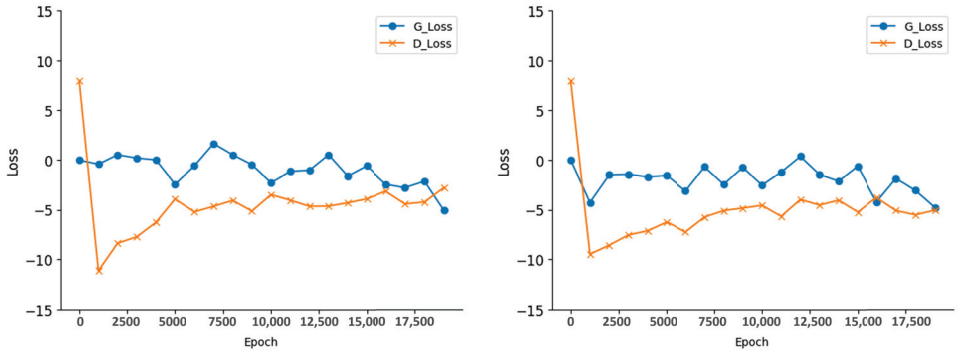


(**c**) WGANDIV-D loss graph (**d**) WGANDIV-A loss graph

Figure 18. Loss graphs of WGANDIV.



(**a**) WGAN-GP-1 loss graph (**b**) WGAN-GP-5 loss graph



(**c**) WGAN-GP-D loss graph (**d**) WGAN-GP-R loss graph

Figure 19. Loss graphs of WGAN-GP.

Based on our analysis, we determined that the optimal output for the evaluated WGAN-GP and WGANDIV models can be achieved around 12,500 epochs. At this point, we observed the convergence of the discriminator and generator. Notably, the GANs already exhibited promising results at approximately 12,500 epochs, indicating considerable improvements in image quality within a relatively shorter training duration. As part of our experimental evaluation, we analyzed the training duration for each of the assessed GAN models on our Ubuntu server equipped with two NVIDIA TITAN RTX GPUs. The average training time, measured in minutes, for 50,000 epochs varied across the models: WGAN (26.62), WGAN-GP (33.18), WGAN-DIV (37.84), GAN (27.42), MMGAN (24.43), NSGAN (28.21), BGAN (66.15), BEGAN (39.21), EBGAN (94.176), DRAGAN (100.56), SGAN (108.18), LSGAN (104.73), INFOGAN (96.24), and REL-GAN (107.59). These differences in training time can be attributed to several factors. Model architecture complexity, the number of parameters, and the convergence behavior are key influencers. Models with more intricate architectures and larger parameter spaces often require longer training times to achieve convergence. Additionally, GANs that introduce unique regularization techniques or novel loss functions might also require more iterations to reach an optimal balance between the generator and discriminator networks. Moreover, variations in training time can also be influenced by the computational resources available, such as the processing power of the machine used for training. These varying training durations, influenced by architectural complexity, convergence characteristics, and available computational resources, provide insights into the different demands of the evaluated GAN models.

To assess the training stability and quality of our GAN models for generating imprinted digits, monitoring the loss curves played a crucial role. With our evaluation, we closely examined the loss plots of the discriminator and generator to gain insights into the learning process and ensured effective convergence of the models. Initially, we observed high discriminator and generator losses because the models were randomly initialized and the generated samples did not closely resemble real imprinted digits. However, as training progressed, we observed a gradual decrease in the discriminator loss. This revealed that the discriminator improved its ability to accurately classify real imprinted digits from the generated ones. Simultaneously, the generator loss exhibited a downward trend, indicating that the generator was learning to produce imprinted-digit samples that closely resembled the real digits. This improvement was evident as the discriminator found it increasingly challenging to distinguish between the real imprinted digits and the generated ones. The convergence of the loss curves served as a crucial indicator of the optimization progress of our imprinted-digit GAN models. As the losses approached lower values and the curves exhibited stability, we inferred that the models were reaching a stable equilibrium. This suggested that the generator successfully captured the intricate details and style of the imprinted digits, while the discriminator became highly accurate in differentiating real imprinted digits from the generated ones. By monitoring the loss curves, we identified the potential issues during training, such as fluctuations, sudden spikes, or plateaus. These observations enabled us to address problems such as mode collapse, instability, or inadequate training. Careful analysis of the loss plots guided our decisions regarding hyperparameter tuning, regularization techniques, and architectural modifications. This iterative process helped in improving convergence and generating high-quality imprinted-digit samples that faithfully replicated the intricate details of the original engravings.

5. Limitations and Future Work

A notable limitation in our study is the technical challenge associated with mode collapse, a phenomenon wherein a generative model, such as a GAN, fails to capture the full diversity of the real data distribution, leading to reduced variety in the generated samples. Within the context of our research, the presence of a relatively small dataset encompassing only a few alphabets introduces the potential risk of exacerbating mode collapse. This concern informed our strategic decision to concentrate on a specific subset of characters (13 of 26 alphabet characters). By doing so, we aimed to ensure both diversity

and realism in the augmentation process, thereby mitigating the likelihood of mode collapse and its consequential negative impact on the quality and generalizability of the generated samples. Despite this limitation, we hold a strong belief in the broader applicability of our results and conclusions. Our evaluation methodology encompassed a thorough assessment of visual quality, quantitative metrics, and the practical implications of our findings. These evaluations consistently underscored the efficacy of our chosen approach in enhancing the recognition of engraved characters. Consequently, the techniques applied to the selected subset of characters offer promising avenues for broader application, substantiating the generalizability of our approach beyond the specific character set considered in this study. Future work could explore the incorporation of domain-specific knowledge into GAN models, which could significantly improve the applicability of generated images. The applicability of generated synthetic images goes beyond the maritime field. The ability to incorporate environmental and contextual factors into GAN models could potentially be applied to other industries that rely on image recognition, such as outdoor robotics, agricultural monitoring, and infrastructure maintenance.

6. Conclusions

In conclusion, our research has demonstrated the efficacy of GAN models in augmenting limited datasets of imprinted digits and characters for ship-character recognition. The WGAN-GP and WGANDIV models were able to generate diverse yet realistic digit images that are seamlessly aligned with ship-related engravings. The significance of these findings lies in their potential to significantly enhance maritime safety, operational efficiency, and security by bolstering character recognition capabilities. Our study has made significant progress in addressing data scarcity challenges in ship-character recognition. However, there are still many unexplored possibilities. Future work could explore the incorporation of domain-specific knowledge into GAN models, which could significantly improve the applicability of generated images. The applicability of generated synthetic images goes beyond the maritime field.

Author Contributions: A.A., J.T.S. and I.Y.J. conceived and designed the experiments; A.A. and J.T.S. performed the experiments; A.A., J.T.S., and I.Y.J. analyzed the data; A.A. and J.T.S. wrote the paper. I.Y.J. re-organized and corrected the paper. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (No. 2021R1F1A1064345) and by the BK21 FOUR project funded by the Ministry of Education, Korea (No. 4199990113966).

Data Availability Statement: https://github.com/k-sumtin/ShipMarkingsCharacterDataset.git, accessed on 23 July 2023

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Western Central Atlantic Fishery Commission. *The Marking and Identification of Fishing Vessels*; Food and Agriculture Organization of the United Nations: Rome, Italy, 2017.
2. Joseph, A.; Dalaklis, D. The international convention for the safety of life at sea: highlighting interrelations of measures towards effective risk mitigation. *J. Int. Marit. Saf. Environ. Aff. Shipp.* **2021**, *5*, 1–11. [CrossRef]
3. IMO. *International Convention for the Safety of Life at Sea: Consolidated Text of the 1974 SOLAS Convention, the 1978 SOLAS Protocol, the 1981 and 1983 SOLAS Amendments*; IMO: London, UK, 1986.
4. Wawrzyniak, N.; Hyla, T.; Bodus-Olkowska, I. Vessel identification based on automatic hull inscriptions recognition. *PLoS ONE* **2022**, *17*, e0270575. [CrossRef] [PubMed]
5. Wei, K.; Li, T.; Huang, F.; Chen, J.; He, Z. Cancer classification with data augmentation based on generative adversarial networks. *Front. Comput. Sci.* **2022**, *16*, 1–11. [CrossRef]
6. Kiyoiti dos Santos Tanaka, F.H.; Aranha, C. Data Augmentation Using GANs. *arXiv* **2019**, arXiv:1904.09135.
7. Wickramaratne, S.D.; Mahmud, M.S. Conditional-GAN based data augmentation for deep learning task classifier improvement using fNIRS data. *Front. Big Data* **2021**, *4*, 659146. [CrossRef]

8. Moon, S.; Lee, J.; Lee, J.; Oh, A.R.; Nam, D.; Yoo, W. A Study on the Improvement of Fine-grained Ship Classification through Data Augmentation Using Generative Adversarial Networks. In Proceedings of the 2021 International Conference on Information and Communication Technology Convergence (ICTC), Jeju-do, Republic of Korea, 20–22 October 2021; pp. 1230–1232. [CrossRef]
9. Shin, H.C.; Lee, K.I.; Lee, C.E. Data Augmentation Method of Object Detection for Deep Learning in Maritime Image. In Proceedings of the 2020 IEEE International Conference on Big Data and Smart Computing (BigComp), Busan, Republic of Korea, 19–22 February 2020; pp. 463–466. [CrossRef]
10. Suo, Z.; Zhao, Y.; Chen, S.; Hu, Y. BoxPaste: An Effective Data Augmentation Method for SAR Ship Detection. *Remote Sens.* **2022**, *14*, 5761. [CrossRef]
11. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal loss for dense object detection. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2980–2988.
12. Zhang, S.; Chi, C.; Yao, Y.; Lei, Z.; Li, S.Z. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 9759–9768.
13. Kang, M.; Leng, X.; Lin, Z.; Ji, K. A modified faster R-CNN based on CFAR algorithm for SAR ship detection. In Proceedings of the 2017 International Workshop on Remote Sensing with Intelligent Processing (RSIP), Shanghai, China, 19–21 May 2017; pp. 1–4. [CrossRef]
14. Jiao, J.; Zhang, Y.; Sun, H.; Yang, X.; Gao, X.; Hong, W.; Fu, K.; Sun, X. A Densely Connected End-to-End Neural Network for Multiscale and Multiscene SAR Ship Detection. *IEEE Access* **2018**, *6*, 20881–20892. [CrossRef]
15. Arjovsky, M.; Chintala, S.; Bottou, L. Wasserstein generative adversarial networks. In Proceedings of the International Conference on Machine Learning, PMLR, Sydney, Australia, 6–11 August 2017; pp. 214–223.
16. You, A.; Kim, J.K.; Ryu, I.H.; Yoo, T.K. Application of generative adversarial networks (GAN) for ophthalmology image domains: A survey. *Eye Vis.* **2022**, *9*, 1–19. [CrossRef]
17. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
18. Escorcia-Gutierrez, J.; Gamarra, M.; Beleño, K.; Soto, C.; Mansour, R.F. Intelligent deep learning-enabled autonomous small ship detection and classification model. *Comput. Electr. Eng.* **2022**, *100*, 107871. [CrossRef]
19. Xu, Y.; Wang, X.; Wang, K.; Shi, J.; Sun, W. Underwater sonar image classification using generative adversarial network and convolutional neural network. *IET Image Process.* **2020**, *14*, 2819–2825. [CrossRef]
20. Starynska, A.; Easton, R.L., Jr.; Messinger, D. Methods of data augmentation for palimpsest character recognition with deep neural network. In Proceedings of the 4th International Workshop on Historical Document Imaging and Processing, Kyoto, Japan, 10–11 November 2017; pp. 54–58.
21. Wang, J.; Yu, L.; Zhang, W.; Gong, Y.; Xu, Y.; Wang, B.; Zhang, P.; Zhang, D. Irgan: A minimax game for unifying generative and discriminative information retrieval models. In Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval, Tokyo, Japan, 7–11 August 2017; pp. 515–524.
22. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial networks. *Commun. ACM* **2020**, *63*, 139–144. [CrossRef]
23. Odena, A.; Olah, C.; Shlens, J. Conditional image synthesis with auxiliary classifier gans. In Proceedings of the International Conference on Machine Learning, PMLR, Sydney, Australia, 6–11 August 2017; pp. 2642–2651.
24. Devon Hjelm, R.; Jacob, A.P.; Che, T.; Trischler, A.; Cho, K.; Bengio, Y. Boundary-Seeking Generative Adversarial Networks. *arXiv* **2017**, arXiv:1702.08431.
25. Berthelot, D.; Schumm, T.; Metz, L. Began: Boundary equilibrium generative adversarial networks. *arXiv* **2017**, arXiv:1703.10717.
26. Radford, A.; Metz, L.; Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv* **2015**, arXiv:1511.06434.
27. Gulrajani, I.; Ahmed, F.; Arjovsky, M.; Dumoulin, V.; Courville, A. Improved Training of Wasserstein GANs. *arXiv* **2017**, arXiv:1704.00028.
28. Wu, J.; Huang, Z.; Thoma, J.; Acharya, D.; Van Gool, L. Wasserstein divergence for gans. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 653–668.
29. Kodali, N.; Abernethy, J.; Hays, J.; Kira, Z. On convergence and stability of gans. *arXiv* **2017**, arXiv:1705.07215.
30. Zhao, J.; Mathieu, M.; LeCun, Y. Energy-based generative adversarial network. *arXiv* **2016**, arXiv:1609.03126.
31. Mroueh, Y.; Sercu, T. Fisher GAN. *arXiv* **2017**, arXiv:1705.09675.
32. Chen, X.; Duan, Y.; Houthooft, R.; Schulman, J.; Sutskever, I.; Abbeel, P. InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets. *arXiv* **2016**, arXiv:1606.03657.
33. Mao, X.; Li, Q.; Xie, H.; Lau, R.Y.; Wang, Z.; Paul Smolley, S. Least squares generative adversarial networks. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2794–2802.
34. Goodfellow, I. Nips 2016 tutorial: Generative adversarial networks. *arXiv* **2016**, arXiv:1701.00160.
35. Jolicoeur-Martineau, A. The relativistic discriminator: A key element missing from standard GAN. *arXiv* **2018**, arXiv:1807.00734.
36. Chavdarova, T.; Fleuret, F. Sgan: An alternative training of generative adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 9407–9415.
37. Borji, A. Pros and Cons of GAN Evaluation Measures. *CoRR* **2018**, *179*, 41–65. [CrossRef]

38. Denton, E.L.; Chintala, S.; Fergus, R. Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks. *arXiv* **2015**, arXiv:1506.05751.
39. Shmelkov, K.; Schmid, C.; Alahari, K. How good is my GAN? In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 213–229.
40. Gerhard, H.E.; Wichmann, F.A.; Bethge, M. How sensitive is the human visual system to the local statistics of natural images? *PLoS Comput. Biol.* **2013**, *9*, e1002873. [CrossRef] [PubMed]
41. Zhu, X.; Vondrick, C.; Fowlkes, C.C.; Ramanan, D. Do we need more training data? *Int. J. Comput. Vis.* **2016**, *119*, 76–92. [CrossRef]
42. Sajjadi, M.S.; Bachem, O.; Lucic, M.; Bousquet, O.; Gelly, S. Assessing generative models via precision and recall. *arXiv* **2018**, arXiv:1806.00035.
43. Barratt, S.; Sharma, R. A note on the inception score. *arXiv* **2018**, arXiv:1801.01973.
44. Lucic, M.; Kurach, K.; Michalski, M.; Gelly, S.; Bousquet, O. Are GANs created equal? A large-scale study. *arXiv* **2018**, arXiv:1711.10337.
45. Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *arXiv* **2017**, arXiv:1706.08500.

electronics

MDPI

Article

Generative AI-Driven Data Augmentation for Crack Detection in Physical Structures

Jinwook Kim, Joonho Seon, Soohyun Kim, Youngghyu Sun, Seongwoo Lee, Jeongho Kim, Byungsun Hwang and Jinyoung Kim *

Department of Electronic Convergence Engineering, Kwangwoon University, Seoul 01897, Republic of Korea; yoonlight12@kw.ac.kr (J.K.); dimlight13@kw.ac.kr (J.S.); kimsoogus@kw.ac.kr (S.K.); yakrkr@kw.ac.kr (Y.S.); swoo1467@kw.ac.kr (S.L.); jh980828@kw.ac.kr (J.K.); hbsun1225@kw.ac.kr (B.H.)
* Correspondence: jinyoung@kw.ac.kr; Tel.: +82-2-940-5567

Abstract: The accurate segmentation of cracks in structural materials is crucial for assessing the safety and durability of infrastructure. Although conventional segmentation models based on deep learning techniques have shown impressive detection capabilities in these tasks, their performance can be restricted by small amounts of training data. Data augmentation techniques have been proposed to mitigate the data availability issue; however, these systems often have limitations in texture diversity, scalability over multiple physical structures, and the need for manual annotation. In this paper, a novel generative artificial intelligence (GAI)-driven data augmentation framework is proposed to overcome these limitations by integrating a projected generative adversarial network (ProjectedGAN) and a multi-crack texture transfer generative adversarial network (MCT2GAN). Additionally, a novel metric is proposed to evaluate the quality of the generated data. The proposed method is evaluated using three datasets: the bridge crack library (BCL), DeepCrack, and Volker. From the simulation results, it is confirmed that the segmentation performance can be improved by the proposed method in terms of intersection over union (IoU) and Dice scores across three datasets.

Keywords: crack detection; data augmentation; generative artificial intelligence; generative adversarial network; semantic segmentation

1. Introduction

Cracks are a common problem in physical structures and can cause the deterioration of their durability. The prompt detection and repair of cracks are essential for preventing accidents that may lead to structural collapse [1–3]. Regular crack inspection has been used to assess structural stability, but conventional visual inspection is labor-intensive and relies on the subjective judgment of the inspector. This reliance can lead to inconsistencies in the assessment process and compromise the structure's safety. Image processing methods based on thresholding [4] and edge detection [5] techniques have been proposed to address these inconsistencies. Nevertheless, the performance of these techniques can be degraded by unexpected noise similar to cracks. In addition, their dependency on predefined thresholding [4] and linear filtering [5] limits their generalization ability to new crack images.

Remarkable achievements in deep learning for computer vision have influenced its application to segmentation methodologies. Specifically, convolutional neural network (CNN)-based methods can more effectively identify crack patterns compared to previous image processing techniques thanks to their ability to approximate intricate nonlinear functions [6–8]. Among the CNN-based approaches, semantic segmentation has emerged as an important research area in crack detection due to its capability in detecting cracks and providing attribute information. Several CNN-based approaches have been proposed to perform crack segmentation [9–11]. Fully connected networks (FCNs) have been proposed

to extract information on the detailed attributes of cracks, such as width, height, and location [9]. Based on its skip connections and symmetric structure, U-Net can reduce false positives caused by unexpected noise [10]. Furthermore, DeepCrack has been proposed to address the discontinuity of crack detection by combining low-level and high-level features [11]. However, CNN-based crack detection methodologies are highly dependent on the quality and variety of the training data, which can limit their generalization in practical applications [12–15].

Because various crack shapes and surface characteristics (texture, rugometry, brightness, glow, etc.) can appear in real-world physical structures, collecting diverse images is crucial to enhance the generalization performance of CNN-based crack detection methods [16,17]. However, when capturing crack images of physical structures, the availability of datasets is often limited due to the cost of data collection. Generative artificial intelligence (GAI), specifically data augmentation techniques based on generative adversarial networks (GANs), has been proposed to address this data availability challenge. GAN-based approaches can complement the scarcity of label data thanks to their ability to generate plausible and diverse images [18]. A data augmentation method based on deep convolutional GAN (DCGAN) [19] has been proposed to augment small crack datasets. However, DCGANs may suffer instability issues because the discriminator converges faster than the generator. The automated pavement crack GANs (APC-GANs) method has been developed to address this convergence problem by incorporating Gaussian noise into the discriminator to reduce its convergence speed [20].

While GAN models have advantages, conventional methods [19,20] require manual annotation, which leads to additional costs. Therefore, the automation of manual annotation can be crucial to mitigate the costs. Researchers have proposed a method to transfer crack textures between different structures while preserving crack patterns using CycleGAN-based techniques to automate this process [21]. However, CycleGAN has limitations in changing geometric shapes, which may limit the texture diversity of generated crack images. A framework integration of DCGAN and pixel-to-pixel (Pix2Pix) has been proposed to address the diversity issue [22]. While the DCGAN-with-Pix2Pix method can be used to improve crack diversity, it remains dependent on human visual inspection for quality assurance. Therefore, there is a need to automate quality assurance inspection while increasing diversity.

In summary, the data availability issue has been mitigated using data augmentation in previous methods. However, manual annotation can incur costs for conventional data augmentation methods. While recent studies have been proposed to reduce the annotation cost, previous studies encountered challenges related to texture diversity, scalability over multiple physical structures, and the need for manual inspection. In this paper, a novel GAI-driven data augmentation framework is proposed to address these issues by incorporating projected GAN (ProjectedGAN) and multi-crack texture transfer GAN (MCT2GAN). The proposed framework can mitigate the constraints of texture diversity and scalability over multiple physical structures by employing ProjectedGAN and MCT2GAN, respectively, while reducing the costs of manual annotation via a novel evaluation process. The main contributions of this paper can be summarized as follows:

- A novel dataset augmentation framework is proposed to address the limitations of texture diversity and data availability issues in small crack datasets. The proposed framework operates in two stages: a label generator and a crack image generator. By implementing this framework to generate a diverse texture crack dataset, the segmentation accuracy in crack detection systems can be improved in environments with small amounts of available data.
- A modified loss function and ReMix method are integrated into StarGANv2; MCT2GAN is proposed to preserve the label image's crack shape and improve GAN's learning stability. The proposed GAN model that can preserve the crack shape can improve the segmentation performance in crack detection systems with small amounts of crack data.

- A novel performance metric called the "crack representation balance (CRB) score" is proposed to assess the representation of crack patterns in generated images by considering global and local perspectives. The structure of the cracks and the detailed patterns of each can be qualified with global and local perspectives, respectively. Therefore, this approach can provide a perspective to evaluate the quality of the generated crack segmentation dataset.

The rest of this paper is organized as follows: In Section 2, a dataset synthesis framework is proposed. In Section 3, the proposed framework is validated through experiments. Section 4 concludes the paper by discussing this study's findings and suggesting areas for future research.

2. Framework for Crack Data Augmentation

The process of generating synthetic crack datasets in the proposed dataset synthesis framework is illustrated in Figure 1. This generation process consists of three stages: label generation, crack generation, and quality assessment. In the initial stage, a set of synthetic crack label images is generated by the ProjectedGAN model. A feedback loop is applied to the output to reduce the need for manual evaluation by incorporating a discriminator. The synthesized images are used to train the MCT2GAN model. Then, a style encoder is used to adjust the crack pattern to match the surface texture so that the cracks match the unique properties of each texture. The quality of the generated images can be evaluated with performance metrics to ensure that the best images are selected for inclusion. Finally, the synthesized label images are combined with the corresponding crack images to generate a diverse dataset suitable for training and testing the semantic segmentation model.

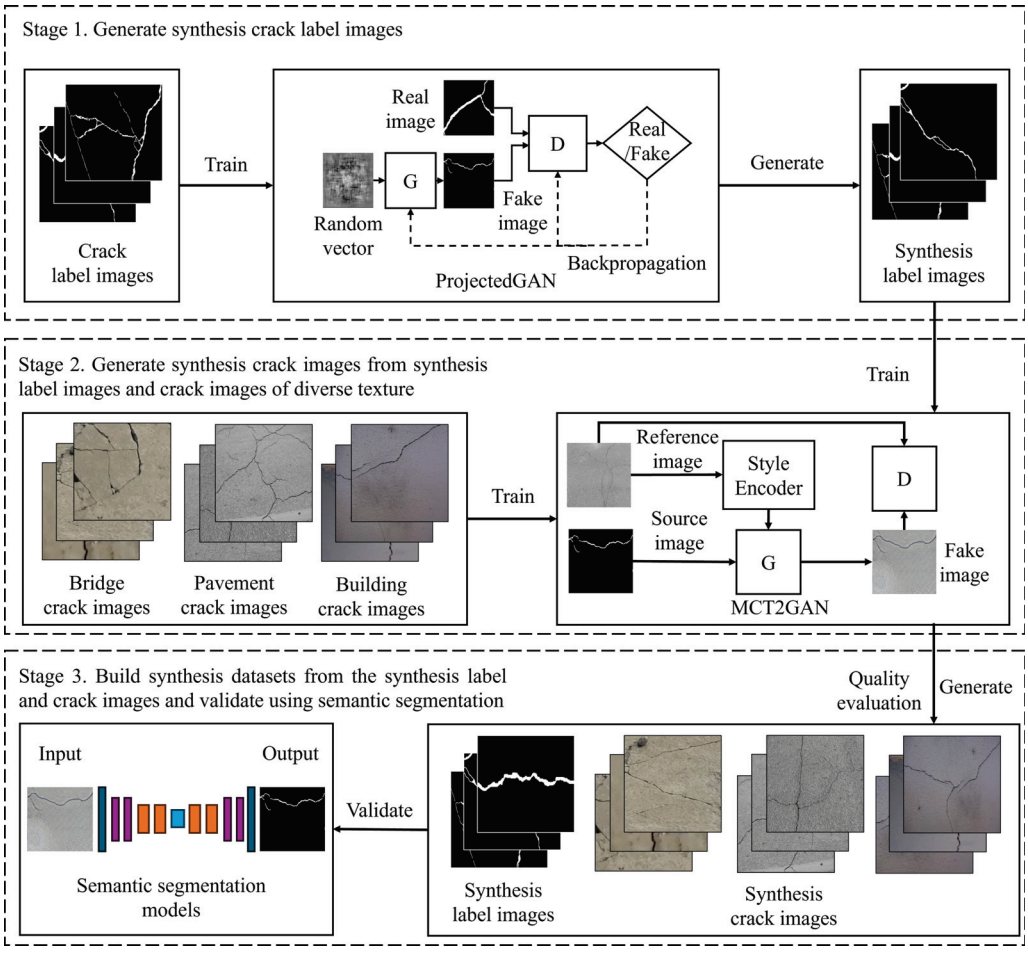


Figure 1. Process of generating a synthesized crack dataset in the proposed dataset synthesis framework.

2.1. Label Generator

ProjectedGAN [23] is a promising candidate for implementing a label generator, as it can generate diverse images. As shown in Figure 2, ProjectedGAN consists of a generator, a pre-trained feature network, a random projection, and multiscale discriminators. The generator G produces fake labeled images $G(z)$ from a random vector z. Features of a real label image x and a fake label image $G(z)$ are extracted by pre-trained feature extraction networks to disentangle causal generative factors. However, according to [23], the discriminator may not fully exploit the features extracted from the feature extraction networks. In Figure 2, random projection is employed to enhance the discriminator's capacity to fully utilize features extracted from the pre-trained networks, facilitating the mixing of features across channels and resolutions through 1×1 and 3×3 convolutions. Then, the multiscale discriminators D_l receive the projected features $P_l(\cdot)$ as inputs and determine whether they are real or fake. These multiscale discriminators provide feedback, encouraging the generator to create finely detailed images. Finally, the generator and discriminator weights are updated based on the feedback from the discriminators. The ProjectedGAN optimization process can be expressed as follows:

$$\min_{G} \max_{D_l} \sum_{l=1}^{4} \Big(\mathbb{E}_x[\log D_l(P_l(x))] + \mathbb{E}_z[\log(1 - D_l(P_l(G(z))))] \Big), \tag{1}$$

where the index l from 1 to 4 is used to represent the projected features P_l and discriminators D_l. Only D_l and G are updated during optimization, while P_l remains unchanged. P_l consists of pre-trained feature networks, cross-channel mixing (CCM), and cross-scale mixing (CSM).

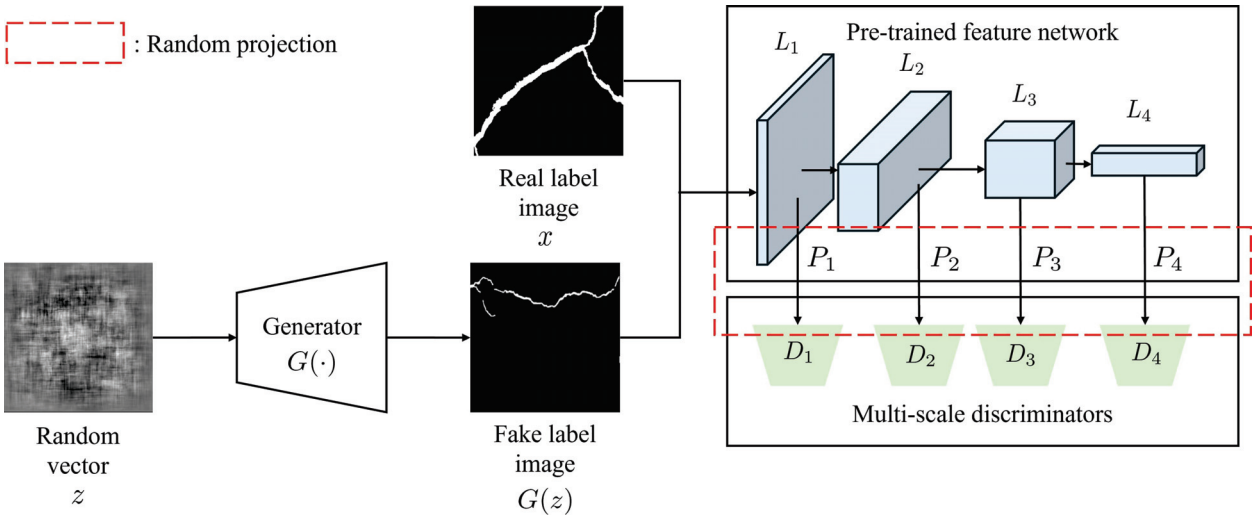


Figure 2. Structure of ProjectedGAN.

2.2. Crack Generator with Various Textures

MCT2GAN is an image-to-image translation model that can train multiple domains with a single generator, reducing the need for a re-training process for different data types. This ability to handle multiple domains can provide it with an advantage in scalability over multiple physical structures compared to the other GAN models trained on a specific structure. As shown in Figure 3, the MCT2GAN model is used to produce a wide range of textural crack images using both synthetic label and real crack datasets. Initially, the source and reference images are selected for translation. The reference image is passed through a style encoder, generating a style code, which the generator uses to transform the source image. Finally, the discriminator determines whether the transformed image is real or fake.

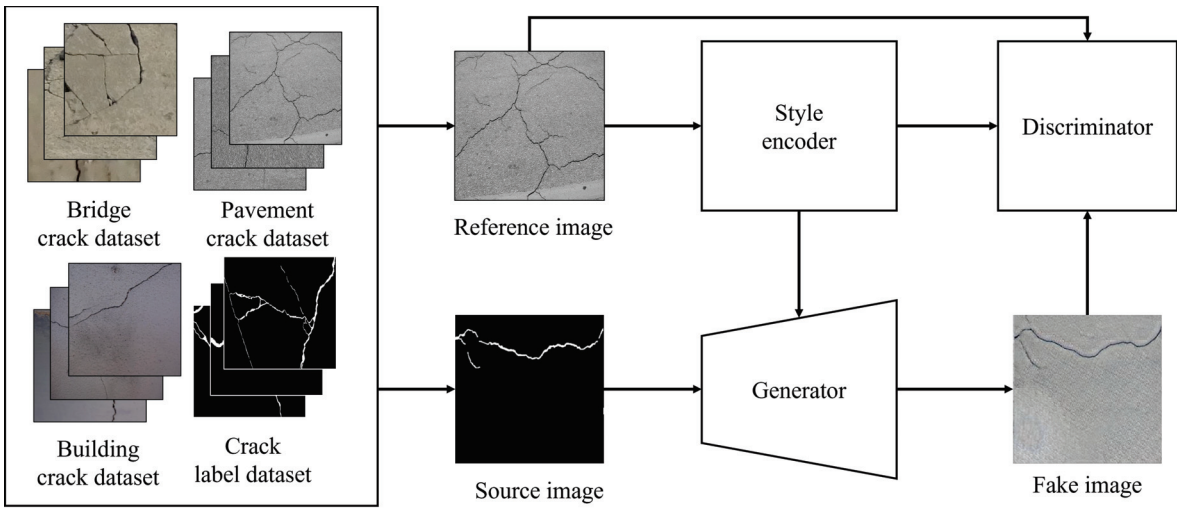


Figure 3. Structure of MCT2GAN.

The loss function of MCT2GAN consists of adversarial loss, style reconstruction, diversity-sensitive loss, and cycle consistency loss. The generator is designed to perform bi-directions, from image to label and from label to image. However, since the focus is on translating images from labels, the description will be limited to this process. In this context, the source images $\mathbf{x}_{src}$ and the reference images $\mathbf{x}_{ref}$ are assumed to be the label and crack images, respectively. y is the domain vector for each structure and consists of a building, pavement, and bridge.

In adversarial loss, the generator learns to generate a realistic crack images $G(\mathbf{x}_{src}, \tilde{\mathbf{s}})$ from the label images $\mathbf{x}_{src}$ and style code $\tilde{\mathbf{s}}$, making it indistinguishable from a crack image. Hinge loss [24] is used to implement adversarial loss to improve the training stability of MCT2GAN. The equation is represented as

$$\mathcal{L}_{adv,D} = \mathbb{E}_{\mathbf{x}_{src},y}[\max(0, 1 - D_y(\mathbf{x}_{src}))] + \mathbb{E}_{\mathbf{x}_{src},\mathbf{x}_{ref},\tilde{y}}[\max(0, 1 + D_{\tilde{y}}(G(\mathbf{x}_{src}, \tilde{\mathbf{s}})))], \quad (2)$$

$$\mathcal{L}_{adv,G} = \mathbb{E}_{\mathbf{x}_{src},\mathbf{x}_{ref},\tilde{y}}[D_{\tilde{y}}(G(\mathbf{x}_{src}, \tilde{\mathbf{s}}))], \quad (3)$$

where $\mathbf{x}_{src}$ and $\mathbf{x}_{ref}$ denote sets of label images and crack images, respectively, $D_y(\cdot)$ is the output of discriminator corresponding to domain y, and $\tilde{\mathbf{s}} = E(\mathbf{x}_{ref})$ is style code extracted by style encoder E from crack image $\mathbf{x}_{ref}$ corresponding to y.

The generator's objective in style reconstruction loss is to accurately capture the style of the structural domain $\tilde{y}$. The loss function of style reconstruction is expressed as

$$\mathcal{L}_{sty} = \mathbb{E}_{\mathbf{x}_{src},\tilde{y},\mathbf{x}_{ref}}[\|\tilde{\mathbf{s}} - E_{\tilde{y}}(G(\mathbf{x}_{src}, \tilde{\mathbf{s}}))\|_1], \quad (4)$$

where $E_{\tilde{y}}$ is the style encoder for the target domain $\tilde{y}$. During training, the generator can learn the style information by minimizing the difference between the encoded feature and the style code from the generated data.

The diversity-sensitive loss enables the generator to produce various texture crack images. This loss function encourages the generator to produce diverse outputs when provided with different style codes, even if the input label image is unchanged. The loss function is formulated as

$$\mathcal{L}_{ds} = \mathbb{E}_{\mathbf{x},\tilde{y},\mathbf{x}_{ref_1},\mathbf{x}_{ref_2}}[\|G(\mathbf{x}_{src}, \tilde{\mathbf{s}}_1) - G(\mathbf{x}_{src}, \tilde{\mathbf{s}}_2)\|_1], \quad (5)$$

where $\tilde{\mathbf{s}}_1 = E(\mathbf{x}_{ref_1})$ and $\tilde{\mathbf{s}}_2 = E(\mathbf{x}_{ref_2})$ are the style codes extracted from the reference images $\mathbf{x}_{ref_1}$ and $\mathbf{x}_{ref_2}$, respectively. By incorporating this loss, the generator is guided to avoid producing similar outputs and instead generates diverse texture patterns.

Cycle consistency loss is used to preserve the characteristics of the label image. In this study, the label image and the crack image must be paired in terms of the shape and localization of the crack to be utilized in semantic segmentation. Therefore, this loss plays an important role in generating an accurate crack image corresponding to the label image. This loss is defined as

$$\mathcal{L}_{cyc} = \mathbb{E}_{\mathbf{x},y,\tilde{y},\mathbf{z}}[\|\mathbf{x}_{src} - G(G(\mathbf{x}_{src}, \tilde{\mathbf{s}}), \hat{\mathbf{s}})\|_1], \tag{6}$$

where $\hat{\mathbf{s}} = E_y(\mathbf{x})$ represents the style code of the label crack image estimated by the style encoder.

By combining $\mathcal{L}_{sty}$, $\mathcal{L}_{ds}$, and $\mathcal{L}_{cyc}$ with their corresponding hyperparameters λ_{sty}, λ_{ds}, and λ_{cyc}, respectively, the loss function of MCT2GAN is formulated as follows:

$$\min_{G,E} \max_{D} \mathcal{L}_{adv} + \lambda_{sty}\mathcal{L}_{sty} - \lambda_{ds}\mathcal{L}_{ds} + \lambda_{cyc}\mathcal{L}_{cyc}. \tag{7}$$

This objective function enables the generator to produce realistic and diverse crack images across multiple domains.

2.3. Quality Evaluation for Generated Data

The Fréchet inception distance (FID) [25] is used to evaluate the fidelity of generated image data through the Fréchet distance between the fake image distribution P_f and the real image distribution P_r using InceptionNet. The formula of FID is as follows:

$$\mathrm{FID}(P_r, P_f) = \|\mu_r - \mu_f\|_2^2 + \mathrm{Tr}\Big(C_r + C_f - 2(C_r C_f)^{1/2}\Big), \tag{8}$$

where the mean vectors are denoted as μ_r and μ_f, and the covariance matrices are denoted as C_r and C_f for real and fake images, respectively. $\mathrm{Tr}(\cdot)$ means the sum of the diagonal elements of a matrix.

Quantifying whether the generated image represents the crack pattern in a specific label image can be used to evaluate the generator's performance. Evaluating the generator consists of three steps: generation, prediction, and comparison. First, the generator generates a crack image from the label image, a reference image. Then, image cracks are predicted by using the FCN model. The quality of the generated images can be evaluated by comparing the predicted images with the label images of the generated crack image and calculating the Dice, Hausdorff distance (HD) [26], and crack representation balance (CRB) scores. Lastly, the evaluated images are used to augment the original dataset. The intersection over union (IoU) and Dice scores are used to evaluate the augmented crack datasets' segmentation performance. The IoU and Dice scores are calculated as follows:

$$\mathrm{IoU} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}, \tag{9}$$

$$\mathrm{Dice} = \frac{2 \times \mathrm{TP}}{2 \times \mathrm{TP} + \mathrm{FP} + \mathrm{FN}}, \tag{10}$$

where TP, FP, and FN are the true positive, false positive, and false negative calculated between the predicted image and the label image.

The process for assessing the quality of the generated crack images is depicted in Figure 4. Herein, the Dice score is used to evaluate the degree to which the generated crack represents the size of the label crack. The HD score can be used to evaluate the accuracy of the generated crack image in representing the shape of the labeled crack. The HD score is expressed as

$$\mathrm{HD}(A, B) = \max[h(A, B), h(B, A)], \tag{11}$$

where A and B are the sets of label images and generated crack images, respectively. $h(A, B) = \max_{a \in A} \min_{b \in B} \|a - b\|$ is the directed Hausdorff distance. a and b are points of the sets.

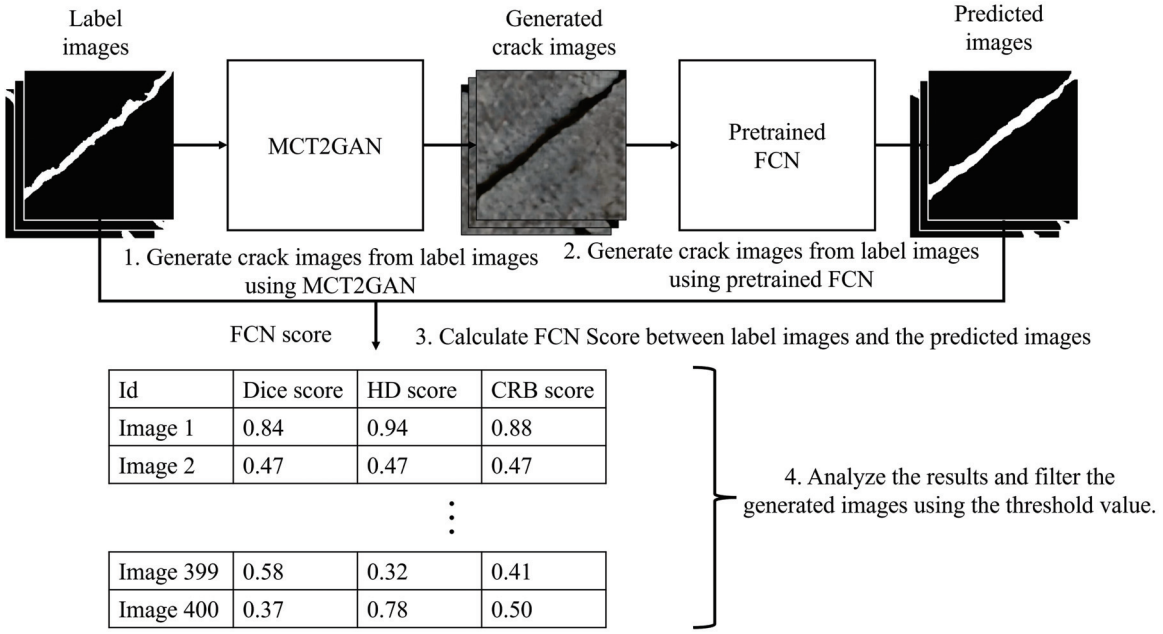


Figure 4. Process of quality evaluation.

The Dice and HD scores can measure the ability of a model to represent the size and patterns of cracks, respectively. These two metrics need to be considered simultaneously to evaluate crack representation. Therefore, the CRB score is proposed to consider the performance of both metrics simultaneously through the harmonic mean of the two metrics. The CRB score can be described using the following formulas:

$$\text{CRB-score} = 2 \times \frac{\text{Dice} \times \text{HD}}{\text{Dice} + \text{HD}}. \tag{12}$$

3. Simulation Results

3.1. Simulation Settings

The proposed framework was validated using three datasets: the Bridge Crack Library (BCL), DeepCrack, and Volker. These datasets were selected due to their diversity in crack types and environmental conditions. The BCL dataset consists of 11,000 images with a resolution of 256 × 256, which are collected from concrete bridges [27]. The number of images was selected to total 500 via random sampling to prevent generating crack images with a biased texture. The DeepCrack dataset contains 537 crack images with a resolution of 544 × 384, obtained using a line-array camera on the surface of the concrete and asphalt pavement [11]. The Volker dataset consists of 427 images with a resolution of 512 × 512 [28] in concrete building structures. All images were resized to a 256 × 256 resolution for consistency in the input dimensions. In this paper, datasets of small size are employed to exploit the advantage of the augmentation method. It is expected that the impact of augmentation will become smaller with decreasing data size. In other words, the gap in performance enhancement by the augmentation method may be decreased in large datasets. Therefore, small datasets could be more suitable for clearly demonstrating the impact of augmentation [29]. An overview of the datasets is given in Table 1.

Table 1. Overview of datasets.

Dataset	Structure	Materials	Dataset Size	Image Size (H × W)
BCL [27]	Bridge	Concrete	500	256 × 256
DeepCrack [11]	Pavement	Concrete and asphalt	537	544 × 384
Volker [28]	Building	Concrete	427	512 × 512

The label and crack generators were implemented using PyTorch 2.0.1, CUDA 11.7, and Python 3.10.11 on Ubuntu 20.04.5. To meet the computational requirements of the GAN model, two NVIDIA GeForce RTX 4090 GPUs were used for the simulations.

For the hyperparameter settings of ProjectedGAN, the number of iterations was empirically set to 20,000. The optimizer was configured as Adam, with a $\beta 1$ of 0.5 and a $\beta 2$ of 0.999. The batch size was set to 8, balancing GPU memory usage and training stability. The learning rate was set to 0.0002, and to smooth the learning process, an exponential moving average (EMA) with a decay rate of 0.999 was applied.

For MCT2GAN's hyperparameter settings, the weights for R1 regularization, cyclic consistency loss, style reconstruction loss, and diversity-sensitive losses were set to 1, 1, 10, and 2, respectively. Adam was used to optimize the generator and discriminator with a $\beta 1$ of 0 and a $\beta 2$ of 0.99. The batch size was set to 8. Both the learning rate and weight decay were set to 0.0001. EMA was applied with a decay rate of 0.999. The ReMix method [30] was used to prevent overfitting in the training process. The training iteration was empirically set to stop at 20,000.

3.2. Performance Evaluation and Discussion

The augmentation and stylization capabilities of the proposed model were evaluated by comparing them with conventional methods for data augmentation [21,22]. CycleGAN and Pix2Pix were selected as representative methods among the existing augmentation methods. Since CycleGAN and Pix2Pix cannot perform texture transfer, these models were trained using individual datasets, while MCT2GAN was trained with multiple datasets simultaneously.

The texture diversity and quality of the images generated by each GAN model were evaluated by visualizing the crack images in Figure 5. The images generated by CycleGAN can preserve crack patterns in the visual evaluation, but cannot describe detailed textures. This limitation may be due to the loss function of CycleGAN, which only preserves the crack patterns. The images generated by Pix2Pix can preserve both textures and crack patterns. However, it was confirmed that Px2Pix cannot represent various texture styles because the generated images tend to have similar textures. As shown in Figure 5, the proposed model can produce diverse texture crack images compared with the other models. In addition, it was confirmed that the crack patterns generated by the proposed model have a higher resolution than those generated by CycleGAN and Pix2Pix. Consequently, these results imply that the proposed model can generate more diverse textures and higher-quality crack images than comparative models.

A comparison based on FID, described in Table 2, was performed to compare the quality of images generated by conventional GAN-based and the proposed models. In Table 2, it is confirmed that the MCT2GAN model shows superior FID values compared with the other models on all datasets. Typically, it is known that learning multiple textures can degrade performance. However, the simulation results, which show that the FID scores of MCT2GAN obtained after learning three textures were higher than those of the comparative models, may imply that the proposed model can learn complex distributions in multiple structures.

Table 2. FID of GAN-based methods.

Method	BCL	DeepCrack	Volker
CycleGAN [21]	212.46	245.79	240.05
Pix2Pix [22]	101.02	142.64	139.33
MCT2GAN	**86.17**	**132.68**	**126.41**

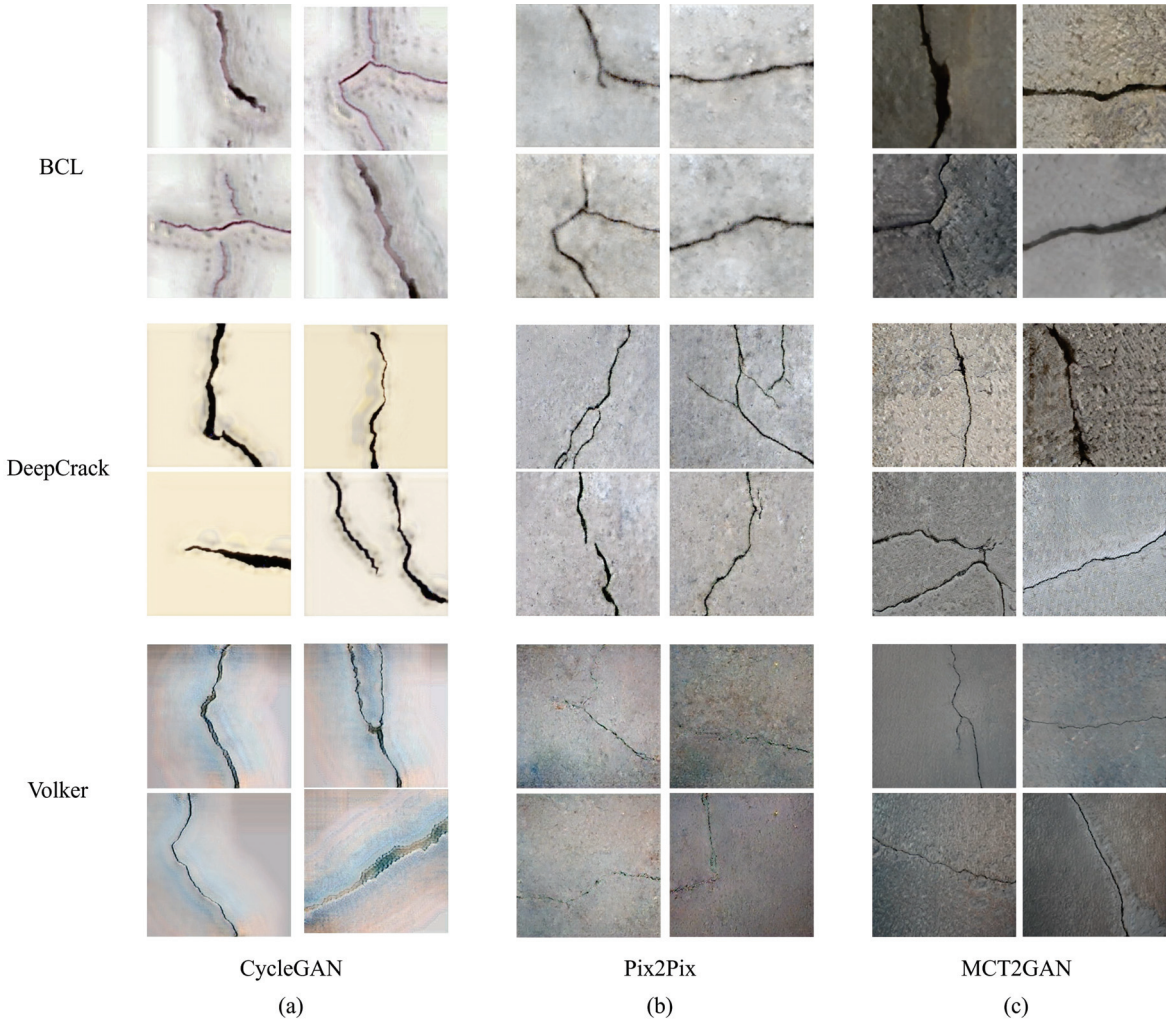


Figure 5. Crack images generated from GAN-based methods.

The quality metrics and images generated by the crack generator are described in Figures 6 and 7. After 2000 iterations, the HD score increased, indicating that the shape of the generated crack can be closer to the shape of the label image. However, both the Dice and CRB scores decreased. As shown in Figure 7a, this change was confirmed to be due to a reduction in overlapping areas between the label crack and the synthesized crack due to the crack being generated in the wrong location compared to the image in 1000 iterations. As a result, the CRB score effectively penalizes the synthesized crack's localization error, reflecting the positional discrepancy despite the improved shape accuracy. The Dice score barely changed between 6000 and 7000 iterations, as illustrated in Figure 6, while the HD and the CRB scores increased. In the overlapped image at 6000 and 7000 iterations, as shown in Figure 7b, the region of blue color (FP) below the crack decreased from 7000. It was confirmed that the HD score penalizes this exaggerated appearance of cracks. In this case, the CRB score effectively captures this penalty, reflecting the shape correction and overall crack quality. The HD score was maintained from 17,000 to 18,000 iterations, as illustrated in Figure 6, but both the Dice and the CRB scores increased. As shown in Figure 7c, the reduction in the blue region could indicate that the size of the synthesized crack changed. Here, the CRB score captures these changes effectively, reflecting the synthesized crack's size and shape. Consequently, the CRB score can be employed to assess the overall quality of synthesized crack images by simultaneously evaluating the images' localization, size, and shape.

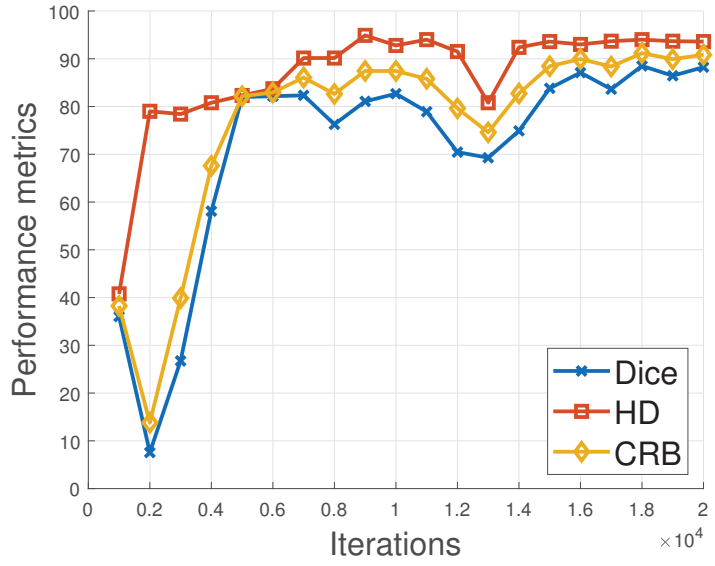


Figure 6. Performance metrics of generated crack image sample in training process.

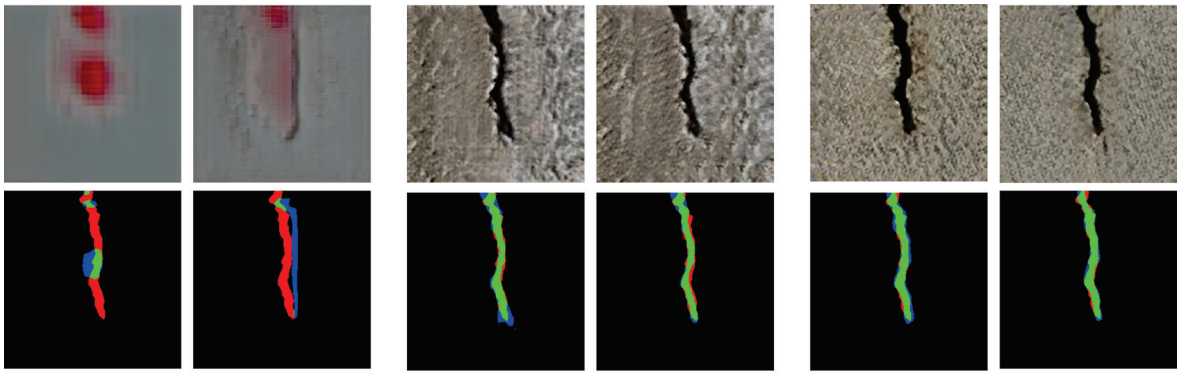


Figure 7. Samples of the generated crack images in the training process. The images in the first row are the synthesized crack images. The images in the second row are the overlapped images between the ground truth and the predicted mask of the synthesized image from the pre-trained FCN model.

An ablation study was performed to confirm the effectiveness of the label generator in data augmentation, and the results are summarized in Table 3. According to Table 3, increasing the cycle consistency loss λ_{cyc} can improve the Dice scores by preserving the shape of the crack. The ReMix method improved the Dice scores of BCL and DeepCrack, while Volker's performance suffered. By changing the adversarial loss from BCE to hinge loss, the variance in the Dice scores in the three datasets decreased. It was confirmed that hinge loss improves training stability by unscrewing the distribution of the three datasets. However, using hinge loss without the ReMix method decreased performance in all datasets. The model with hinge loss is typically trained to perform generalization for stable training. In environments with small amounts of data, hinge loss can lead to overfitting issues. ReMix, on the other hand, is a method that enhances the diversity of data based on augmentation techniques, thereby preventing overfitting. Therefore, upon integrating the ReMix method and hinge loss, they may act as complementary solutions to each other's weaknesses. In addition, a label generator can enhance the performance of MCT2GAN by increasing the amount of data.

Table 3. Ablation study of crack generator in terms of Dice score.

Method	BCL	DeepCrack	Volker
StarGANv2 [31]	39.66	36.67	33.86
StarGANv2 + λ_{cyc} 10	50.70	40.18	45.66
StarGANv2 + λ_{cyc} 10 + ReMix	58.79	45.86	37.21
StarGANv2 + λ_{cyc} 10 + Hinge Loss	35.81	26.77	33.33
StarGANv2 + λ_{cyc} 10 + ReMix + Hinge Loss (MCT2GAN)	60.36	52.57	51.71
MCT2GAN + Data of Label Generator (Proposed method)	**62.98**	**53.90**	**59.36**

The IoU and Dice scores of the original dataset and the dataset augmented by StarGANv2, MCT2GAN, and the proposed method are presented in Table 4. The results in Table 4 indicate that the U-Net trained on the dataset augmented by the proposed method outperformed the one trained on the original dataset and the dataset augmented by StarGANv2 and MCT2GAN regarding IoU and Dice score. For the BCL dataset, it was demonstrated that the IoU and Dice scores were up to 3.91% and 2.18% higher, respectively, than those of the original dataset. Then, for the DeepCrack dataset, it was demonstrated that the IoU and Dice scores improved by up to 5.00% and 3.23%, respectively. Lastly, for the Volker dataset, it was demonstrated that the IoU and Dice scores were up to 1.94% and 1.50% higher, respectively, than those of the original dataset. Based on the IoU and Dice scores, it is confirmed that the proposed method could enhance the crack segmentation task with small amounts of crack data.

Table 4. Crack dataset segmentation results.

Dataset	Original Dataset		Augmented Dataset					
			StarGANv2		MCT2GAN		Proposed Method	
	IoU	Dice	IoU	Dice	IoU	Dice	IoU	Dice
BCL	51.96	68.59	43.61	60.73	52.44	68.81	53.99	70.09
DeepCrack	68.37	81.04	58.83	74.08	69.87	82.26	71.79	83.66
Volker	42.78	59.64	33.34	50.01	38.40	55.49	43.61	60.54

4. Conclusions

In this paper, a novel generative artificial intelligence (GAI)-driven framework for effective data augmentation in environments with small amounts of data is proposed. By integrating a projected generative adversarial network (ProjectedGAN) and a multi-crack texture transfer generative adversarial network (MCT2GAN), the framework can address challenges related to texture diversity and scalability over multiple physical structures. In addition, the crack representation balance (CRB) score was introduced as a novel performance metric to evaluate the quality of generated crack images. From the simulation results, it was demonstrated that the proposed method can improve the performance of crack detection systems while reducing the costs associated with data collection and annotation on different datasets. The framework of the proposed method can be refined into an end-to-end system in future work to improve the efficiency of generating crack and label images. The style encoder can also be expanded into a local texture encoder to account for rugometry, brightness, and glow, thereby enhancing generalization to real-world physical structures. The proposed method may be applicable to medical datasets, which face challenges similar to those of crack datasets, including small amounts of data and a lack of diversity.

Author Contributions: Conceptualization, J.K. (Jinwook Kim), J.S., S.K. and S.L.; methodology, J.K. (Jinwook Kim), J.S., S.K. and S.L.; formal analysis, J.K. (Jinwook Kim), J.S., S.K., Y.S. and S.L.; investigation, J.K. (Jinwook Kim), B.H. and J.K. (Jeongho Kim); resources, J.K. (Jinwook Kim), B.H., and J.K. (Jeongho Kim); writing—original draft preparation, J.K. (Jinwook Kim), J.S., S.K., Y.S. and S.L.;

writing—review and editing, J.K. (Jinwook Kim), J.S., S.K., Y.S., S.L., B.H., J.K. (Jeongho Kim), and J.K. (Jinyoung Kim); visualization, J.K. (Jinwook Kim), B.H., J.K. (Jeongho Kim), and J.K. (Jinyoung Kim); supervision, J.K. (Jinyoung Kim); project administration, J.K. (Jinyoung Kim). All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data is contained within the article.

Acknowledgments: This work was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2024-2020-0-01846) supervised by the IITP (Institute for Information and Communications Technology Planning and Evaluation), and in part by a Research Grant from Kwangwoon University in 2024.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Wardhana, K.; Hadipriono, F.C. Analysis of Recent Bridge Failures in the United States. *J. Perform. Constr. Facil.* **2003**, *17*, 144–150. [CrossRef]
2. Zhao, X.H.; Cheng, W.C.; Shen, J.S.; Arulrajah, A. Platform collapse incident of a power plant in Jiangxi, China. *Nat. Hazards* **2017**, *87*, 1259–1265. [CrossRef]
3. Tan, J.S.; Elbaz, K.; Wang, Z.F.; Shen, J.S.; Chen, J. Lessons learnt from bridge collapse: A view of sustainable management. *Sustainability* **2020**, *12*, 1205. [CrossRef]
4. Tang, J.; Gu, Y. Automatic crack detection and segmentation using a hybrid algorithm for road distress analysis. In Proceedings of the 2013 IEEE International Conference on Systems, Man, and Cybernetics, Manchester, UK, 13–16 October 2013; pp. 3026–3030. [CrossRef]
5. Lim, R.S.; La, H.M.; Sheng, W. A robotic crack inspection and mapping system for bridge deck maintenance. *IEEE Trans. Autom. Sci. Eng.* **2014**, *11*, 367–378. [CrossRef]
6. Zhang, L.; Yang, F.; Daniel Zhang, Y.; Zhu, Y.J. Road crack detection using deep convolutional neural network. In Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 25–28 September 2016; pp. 3708–3712. [CrossRef]
7. Cha, Y.J.; Choi, W.; Büyüköztürk, O. Deep learning-based crack damage detection using convolutional neural networks. *Comput.-Aided Civ. Infrastruct. Eng.* **2017**, *32*, 361–378. [CrossRef]
8. Chen, F.C.; Jahanshahi, M.R. NB-CNN: Deep learning-based crack detection using convolutional neural network and Naïve Bayes data fusion. *IEEE Trans. Ind. Electron.* **2018**, *65*, 4392–4400. [CrossRef]
9. Yang, X.; Li, H.; Yu, Y.; Luo, X.; Huang, T.; Yang, X. Automatic pixel-level crack detection and measurement using fully convolutional network. *Comput.-Aided Civ. Infrastruct. Eng.* **2018**, *33*, 1090–1109. [CrossRef]
10. Huyan, J.; Li, W.; Tighe, S.; Xu, Z.; Zhai, J. CrackU-net: A novel deep convolutional neural network for pixelwise pavement crack detection. *Struct. Control Health Monit.* **2020**, *27*, e2551. [CrossRef]
11. Zou, Q.; Zhang, Z.; Li, Q.; Qi, X.; Wang, Q.; Wang, S. DeepCrack: Learning hierarchical convolutional features for crack detection. *IEEE Trans. Image Process.* **2019**, *28*, 1498–1512. [CrossRef]
12. Li, S.; Zhao, X.; Zhou, G. Automatic pixel-level multiple damage detection of concrete structure using fully convolutional network. *Comput.-Aided Civ. Infrastruct. Eng.* **2019**, *34*, 616–634. [CrossRef]
13. Gao, Y.; Zhai, P.; Mosalam, K.M. Balanced semisupervised generative adversarial network for damage assessment from low-data imbalanced-class regime. *Comput.-Aided Civ. Infrastruct. Eng.* **2021**, *36*, 1094–1113. [CrossRef]
14. Zheng, Y.; Gao, Y.; Lu, S.; Mosalam, K.M. Multistage semisupervised active learning framework for crack identification, segmentation, and measurement of bridges. *Comput.-Aided Civ. Infrastruct. Eng.* **2022**, *37*, 1089–1108. [CrossRef]
15. Chen, J.; Lu, W.; Lou, J. Automatic concrete defect detection and reconstruction by aligning aerial images onto semantic-rich building information model. *Comput.-Aided Civ. Infrastruct. Eng.* **2023**, *38*, 1079–1098. [CrossRef]
16. Zou, Q.; Cao, Y.; Li, Q.; Mao, Q.; Wang, S. CrackTree: Automatic crack detection from pavement images. *Pattern Recognit. Lett.* **2012**, *33*, 227–238. [CrossRef]
17. Fan, L.; Li, S.; Li, Y.; Li, B.; Cao, D.; Wang, F.Y. Pavement cracks coupled with shadows: A new shadow-crack dataset and a shadow-removal-oriented crack detection approach. *IEEE/CAA J. Autom. Sin.* **2023**, *10*, 1593–1607. [CrossRef]
18. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; Volume 27, pp. 2672–2680.
19. Xu, B.; Liu, C. Pavement crack detection algorithm based on generative adversarial network and convolutional neural network under small samples. *Measurement* **2022**, *196*, 111219. [CrossRef]
20. Zhang, T.; Wang, D.; Mullins, A.; Lu, Y. Integrated APC-GAN and AttuNet framework for automated pavement crack pixel-level segmentation: A new solution to small training datasets. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 4474–4481. [CrossRef]

21. Branikas, E.; Murray, P.; West, G. A novel data augmentation method for improved visual crack detection using generative adversarial networks. *IEEE Access* **2023**, *11*, 22051–22059. [CrossRef]
22. Jin, T.; Ye, X.W.; Li, Z.X. Establishment and evaluation of conditional GAN-based image dataset for semantic segmentation of structural cracks. *Eng. Struct.* **2023**, *285*, 116058. [CrossRef]
23. Sauer, A.; Chitta, K.; Müller, J.; Geiger, A. Projected GANs converge faster. In *Proceedings of the Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2021; Volume 34, pp. 17480–17492.
24. Tran, D.; Ranganath, R.; Blei, D. Hierarchical implicit models and likelihood-free variational inference. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Volume 30, pp. 5523–5533.
25. Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; Hochreiter, S. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Volume 30, pp. 6626–6637.
26. Hsieh, Y.A.; Tsai, Y.J. Machine learning for crack detection: Review and model performance comparison. *J. Comput. Civ. Eng.* **2020**, *34*, 04020038. [CrossRef]
27. Ye, X.W.; Jin, T.; Li, Z.X.; Ma, S.Y.; Ding, Y.; Ou, Y.H. Structural crack detection from benchmark data sets using pruned fully convolutional networks. *J. Struct. Eng.* **2021**, *147*, 04721008. [CrossRef]
28. Pak, M.; Kim, S. Crack detection using fully convolutional network in wall-climbing robot. In *Proceedings of the Advances in Computer Science and Ubiquitous Computing*; Springer: Singapore, 2021; pp. 267–272. [CrossRef]
29. Shijie, J.; Ping, W.; Peiyi, J.; Siping, H. Research on data augmentation for image classification based on convolution neural networks. In Proceedings of the 2017 Chinese Automation Congress (CAC), Jinan, China, 20–22 October 2017; pp. 4165–4170. [CrossRef]
30. Cao, J.; Hou, L.; Yang, M.H.; He, R.; Sun, Z. ReMix: Towards image-to-image translation with limited data. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 19–25 June 2021; pp. 15013–15022. [CrossRef]
31. Choi, Y.; Uh, Y.; Yoo, J.; Ha, J.W. StarGAN v2: Diverse image synthesis for multiple domains. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 8185–8194. [CrossRef]

electronics

MDPI

Article

Illumination and Shadows in Head Rotation: Experiments with Denoising Diffusion Models

Andrea Asperti *, Gabriele Colasuonno and Antonio Guerra

Department of Informatics-Science and Engineering (DISI), University of Bologna, 40126 Bologna, Italy; gabriele.colasuonno@studio.unibo.it (G.C.); guerra.antonio98@gmail.com (A.G.)
* Correspondence: andrea.asperti@unibo.it

Abstract: Accurately modeling the effects of illumination and shadows during head rotation is critical in computer vision for enhancing image realism and reducing artifacts. This study delves into the latent space of denoising diffusion models to identify compelling trajectories that can express continuous head rotation under varying lighting conditions. A key contribution of our work is the generation of additional labels from the CelebA dataset, categorizing images into three groups based on prevalent illumination direction: left, center, and right. These labels play a crucial role in our approach, enabling more precise manipulations and improved handling of lighting variations. Leveraging a recent embedding technique for Denoising Diffusion Implicit Models (DDIM), our method achieves noteworthy manipulations, encompassing a wide rotation angle of $\pm 30^{\circ}$. while preserving individual distinct characteristics even under challenging illumination conditions. Our methodology involves computing trajectories that approximate clouds of latent representations of dataset samples with different yaw rotations through linear regression. Specific trajectories are obtained by analyzing subsets of data that share significant attributes with the source image, including light direction. Notably, our approach does not require any specific training of the generative model for the task of rotation; we merely compute and follow specific trajectories in the latent space of a pre-trained face generation model. This article showcases the potential of our approach and its current limitations through a qualitative discussion of notable examples. This study contributes to the ongoing advancements in representation learning and the semantic investigation of the latent space of generative models.

Keywords: diffusion models; latent space; embedding; representation learning; semantic trajectories; editing; head rotation

1. Introduction

The possibility of manipulating images acting on their latent representation, which is typical of generative models, has always exerted a particular fascination for researchers. Understanding the effect a tiny modification has on the encoding of a generated sample helps us to better understand the properties of latent space and the disentanglement of the different features. This is strictly related to editing, since understanding semantically meaningful directions (such as color, pose, and shape) can be utilized to modify an image to include certain desired features.

The field of deep generative modeling has recently witnessed a significant shift with the emergence of Denoising Diffusion Models (DDM) [1], which are rapidly establishing themselves as the new state-of-the-art technology [2,3]. These models are likely poised to surpass the long-standing Generative Adversarial Networks (GANs) [4] by providing an excellent generative quality with high sample diversity, simple and stable training, and a solid probabilistic foundation. They have achieved impressive results in a wide range of diverse domains comprising, e.g., medical imaging [5], healthcare [6], protein synthesis [7], and weather forecasting [8].

While DDMs have shown remarkable capabilities in generating realistic samples, the exploration of latent space and the manipulation of generated samples to edit specific attributes remains a complex task. This is partly due to the high dimensionality of latent space, which poses challenges in navigating and understanding the underlying semantics, but also to the complexity of embedding data into the latent space, computing the internal encoding of a given sample. In the case of GANs that have been explored so far, most of the known techniques for semantic editing [9–12] are in fact based on the preliminary definition of a "recoder" [13–15], inverting the generative process and essentially providing a functionality similar to encoders for Variational Autoencoders [16,17].

The embedding problem for the particular but important case of Denoising Diffusion Implicit Models [18] has been recently investigated in [19]. A crucial difference in the latent space of denoising models is that it appears to be organized as a foliation, with a different slice for each data point. These slices correspond to the set of all noisy points in the space that will collapse onto the given data point during the denoising process. Slices are typically very large, occupying significant portions of the input space. As a result, the embedding problem is inherently multimodal and underconstrained. Embedding techniques, such as the one described in [19], typically select a point in the slice based on criteria that are difficult to control and decipher. Consequently, there is no evidence that we can organize the latent points extracted from the embedding network along meaningful trajectories. This is precisely the problem we aim to address in this work.

Unlike many works in the literature that focus on one-step modifications of the input (e.g., changing a color or adding or removing elements), we are interested in continuous modifications of the input image. The case of head rotation is particularly appealing for our study for several reasons. Firstly, face generation is a well-investigated domain, and the rotation problem is recognized as one of the most complex and intriguing editing operations. The challenge with head rotation is that it requires preserving the distinctive features of the person while applying significant transformations that cannot be defined in terms of texture, color, shapes, or other similar information associated with segmentation areas. Another significant point for considering rotation is the availability of good open-source libraries that can automatically measure the pose of the head. These can be used both to guide and to test the effectiveness of the operation.

By employing the DDIM embedding technique, we have been able to achieve remarkable manipulations in head orientation, spanning a large rotation angle of $\pm 30^{\circ}$ along the yaw direction. Some examples are given in Figure 1.

This seems to testify to the fact that compulsory trajectories can be defined in the latent space of diffusion models in spite of the intrinsically multimodal nature of the embedding function.

Our methodology involves utilizing a pre-trained generative latent model for face generation and computing in its latent space trajectories composed of rectilinear segments, thus simulating the rotation effect. The direction of each segment is computed by linear regression, fitting through clouds of latent representations of dataset samples with varying yaw rotations. Each segment is then translated to the correct and known source location. To obtain trajectories tailored to a specific source image, we restrict the analysis to subsets of data sharing significant attributes with it; this is usually sufficient to ensure that the essential characteristics of the face are preserved throughout the manipulation process. We tested several attributes and the most significant ones appear to be gender, expression (smiling/not smiling), age (young/old), and illumination source (left/center/right). This last attribute is not a traditional attribute of the CelebA datasets; we created such labeling in recent years through the collaboration of many students, following a methodology briefly described in Section 5.

Figure 1. Rotation examples. The sources are images no. 114, no. 16,399, and no. 98,018 of CelebA (central image).

The analysis and comparison of the attributes, highlighting the importance of considering the source of illumination to achieve good rotation effects, is the main contribution of our work in the specific domain of face editing.

In this article, we present our methodology and showcase some preliminary, experimental results of the manipulations performed using the DDIM embedding technique. We do not yet have a quantitative evaluation of our work, due to the difficulty in identifying proper metrics; this is left as a subject for further investigation. Nevertheless, our findings demonstrate the potential of DDMs in enabling intricate editing operations while maintaining the fidelity of generated samples. The insights gained from this research contribute to advancing the field of deep generative modeling and provide a valuable foundation for future developments in latent space exploration and attribute manipulation.

The article is structured in the following way. In Section 2 we discuss related works and clarify the scope of our research, which aims to understand the dynamic of head movement in the latent space of Diffusion Models. Section 3 briefly presents the theory of this class of generative models; this section does not contain original material and can be skipped by readers knowledgeable in the area. In Section 4, we discuss the architecture of the neural models used for our work. Section 5 introduces the CelebA dataset, its attributes, and our original labeling relative to the illumination source. In Section 6, we explain our methodology. Preprocessing operations (cropping and background removal) and postprocessing ones (super resolution and color correction) are discussed in Section 7. Numerous examples are given in Section 9 and in Appendix A. Finally, concluding remarks and ideas for future research directions are given in Section 10.

2. Related Works

The task of head rotation holds significant importance in computer vision, finding extensive applications in various domains like security, entertainment, and healthcare.

Before the rise of deep learning, facial rotation methods primarily revolved around applying the traits of an input face image onto a 3D face model and then rotating it to create the desired rotated version. Examples of this approach can be found in [20,21]. In [22], the rotation problem was tackled using a 3D transformation matrix, which mapped each point from a 2D face image to its corresponding point on a 3D face model. Although these

techniques could generate rotated face images, they were constrained by distortion and blurring issues that arose during the conversion of 2D images into 3D models.

The progress of deep learning has significantly expedited advances in facial rotation techniques, especially those leveraging generative adversarial networks (GANs). A typical application is face frontalization, aiming to improve face recognition accuracy by synthesizing a frontal face image from a side-view facial image.

Popular techniques in this category include DR-GAN [23], TP-GAN [24], CAPG-GAN [25], and FNM [26]. DR-GAN isolates the input image's features and angle to generate a frontal image, whereas TP-GAN separately learns the overall outline features and detailed features to synthesize the frontal face. CAPG-GAN utilizes a heat map to frontalize an input face and FNM leverages both labeled and unlabeled data to improve learning efficiency.

All these methods face challenges in producing convincing results for input images taken from near-side angles or angles that are not from frontal views.

Several 3D geometry-based approaches have been devised to tackle head rotation challenges by combining traditional techniques with GANs. Relevant methods in this domain include FF-GAN [27], UV-GAN [28], HF-PIM [29], and Rotate-and-Render [30]. These techniques leverage the strengths of both 3D modeling and GANs to achieve more realistic and accurate rotations, overcoming some limitations of purely 2D or GAN-only approaches.

In contrast to reconstruction-based techniques, 3D geometry-based methods produce more realistic results for side-facing images. However, the need to handle detailed geometrical data, perform extensive rendering calculations, and integrate multiple complex processes makes 3D geometry-based methods more resource-intensive compared to other generative techniques.

Neural Radiance Fields (NeRF) [31] is an advanced method for representing intricate 3D scenes by means of neural networks. NeRF models the radiance and volume density of a scene as a continuous function. This function is parameterized by a neural network that receives a 3D coordinate and a viewing direction as inputs. The scene's appearance is rendered by integrating the radiance along each camera ray. In FENeRF [32], the authors utilize NeRF to forecast a 3D representation of a given face with a particular rotation. This representation can then be further manipulated to edit the facial attributes. All these approaches are sensibly different from our work, since they aim to train conditional models, taking into account the geometric or textural information constraining the generation. In our case, we simply start from an unconstrained generative model, already containing in its latent space the source and target image, and try to identify the path leading from the source to the target. The purpose of this research is to investigate the structure of the latent space of generative models to better understand the learned representation and the properties of the encodings.

Several works have been done in this direction in the case of GANs, aiming to manipulate and govern the attributes of generated faces through a latent space-based approach. These techniques enable control over various attributes, including the age, eyeglasses, gender, expression, and rotation angles of the synthesized faces. Different methods have been developed, including PCA analysis to extract important latent directions [10], semantic analysis to control various attributes [9], and composing a new latent vector to control multiple attributes [33]. The most recent research mostly focused on text-guided image editing [34,35], frequently exploiting segmentation masks to drive generation [36].

All methods address rotation as a single shot operation, failing to provide evidence of a smooth and continuous modification of the source along a given trajectory.

Similar works have been done in the case of Variational Autoencoders (VAEs). In this context, due to the Gaussian-like shape of the latent space induced by the Kullback–Leibler regularization, more principled approaches to the computation of trajectories can be considered, for instance considering geodesic paths [37–39]. In the case of DDMI, the source space is indeed also Gaussian, but this is just the source noisy space rapidly collapsing, after a few iterations of the denoising process, towards the actual manifold of the data. So, there is

no evidence that following a geodesic path could be beneficial, and our investigation seems to suggest that this is not the case (see Section 8).

In the specific case of Diffusion Models, there are several recent investigations on text-guided generation [40–43], but we are aware of no work focused on trajectories for continuous transformations, as the ones addressed by our research.

3. Denoising Diffusion Models

This section provides a fairly self-contained theoretical introduction to diffusion models. It has been added for the sake of completeness, and to introduce the terminology. It does not contain original material and it can be skipped by people with knowledge in the domain. We refer the readers to these excellent textbooks for additional information [44,45].

3.1. Diffusion and Reverse Diffusion

In order to have data distributed according to some probability distribution, $x_0 \sim q(x_0)$. We thus consider a forward process which gradually adds noise to the data, producing noised samples $x_1, \ldots, x_T$, for some time horizon $T > 0$. Specifically, the diffusion model $q(x_{0:T})$ is supposed to be a Markov chain with the following shape:

$$q(x_t|x_{t-1}) = \mathcal{N}\left(x_t \middle| \sqrt{\frac{\alpha_t}{\alpha_{t-1}}}x_{t-1}; \left(1 - \frac{\alpha_t}{\alpha_{t-1}}\right)\cdot I\right) \tag{1}$$

with $\{\alpha_t\}_{t\in[0,T]}$ being a decreasing sequence in the interval $[0, 1]$.

Considering the fact that the composition of Gaussian distributions is still Gaussian, in order to sample $x_t \sim q(x_t|x_0)$ we do not need to go through an iterative process. If we define $\alpha_t = 1 - \beta_t$ and $\overline{\alpha}_t = \prod_{s=0}^{t} \alpha_s$, then

$$\begin{aligned} q(x_t|x_0) &= \mathcal{N}(x_t; \sqrt{\overline{\alpha}_t}x_0, (1 - \overline{\alpha}_t)\mathbf{I}) \\ &= \sqrt{\overline{\alpha}_t}x_0 + \epsilon\sqrt{1 - \overline{\alpha}_t} \end{aligned}$$

for $\epsilon \sim \mathcal{N}(0, \mathbf{I})$. In these equations, $1 - \overline{\alpha}_t$ is the variance of the noise for an arbitrary time step t and $\overline{\alpha}_t$ could be equivalently used instead of β_t to define the schedule of the noising process.

The idea behind denoising generative models is to reverse the above process, addressing the distribution $q(x_{t-1}|x_t)$. If we know how to sample from $q(x_{t-1}|x_t)$, then we can generate a sample starting from a Gaussian noise input $x_T \sim \mathcal{N}(0, \mathbf{I})$. In general, the distribution $q(x_{t-1}|x_t)$ cannot be expressed in closed form and it will be approximated using a neural network. In [46] it was observed that $q(x_{t-1}|x_t)$ approaches a diagonal Gaussian distribution when T is large and $\beta_t \to 0$, so in order to learn the distribution it suffices to train a neural network predicting the mean μ_θ and the diagonal covariance matrix Σ_θ:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$$

The whole reverse process is thus:

$$p_\theta(x_{0:T}) = p_\theta(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t) \tag{2}$$

where $p_\theta(x_T) = \mathcal{N}(0, \mathbf{I})$.

For training, we can use a variational lower bound on the negative log likelihood:

$$\begin{aligned}
&-\log p_\theta(\mathbf{x}_0) \\
&\quad \leq -\log p_\theta(x_0) + D_{\mathrm{KL}}(q(x_{1:T}|x_0)\|p_\theta(x_{1:T}|x_0)) \\
&\quad = -\log p_\theta(x_0) + \mathbb{E}_{x_{1:T}\sim q(x_{1:T}|x_0)}\Big[\log \tfrac{q(x_{1:T}|x_0)}{p_\theta(x_{0:T})/p_\theta(x_0)}\Big] \\
&\quad = -\log p_\theta(x_0) + \mathbb{E}_q\Big[\log \tfrac{q(x_{1:T}|x_0)}{p_\theta(x_{0:T})} + \log p_\theta(x_0)\Big] \\
&\quad = \mathbb{E}_q\Big[\log \tfrac{q(x_{1:T}|x_0)}{p_\theta(x_{0:T})}\Big] \\
&\quad = \mathbb{E}_q\Big[-\log p(x_T) - \textstyle\sum_{t\geq 1}\log \tfrac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}\Big] = \mathcal{L}(\theta)
\end{aligned}$$

This can be further refined expressing L_θ as the sum of the following terms [46]:

$$L_\theta = L_T + L_{t-1} + \cdots + L_0 \tag{3}$$

where

$$\begin{aligned}
&L_T = D_{\mathrm{KL}}(q(x_T|x_0) \parallel p_\theta(x_T)) \\
&L_t = D_{\mathrm{KL}}(q(x_t|x_{t+1}, x_0)\|p_\theta(x_t|x_{t+1})) \text{ for } 1 \leq t \leq T-1 \\
&L_0 = -\log p_\theta(x_0|x_1)
\end{aligned}$$

The advantage of this formulation is that the forward process posterior $q(x_t|x_{t+1}, x_0)$ becomes tractable when conditioned on x_0 and assumes a Gaussian distribution:

$$q(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}|\tilde{\mu}(x_t, x_0); \tilde{\beta}_t\mathbf{I}) \tag{4}$$

where

$$\tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 \tag{5}$$

and

$$\tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\cdot\beta_t. \tag{6}$$

As a consequence of this, the KL divergences in Equation (3) are comparisons between Gaussians and they can be calculated in a Rao–Blackwellized fashion with closed-form expressions.

After a few manipulations, we get:

$$\mathcal{L}(\theta) = \sum_{t=1}^{T}\gamma_t\mathbb{E}_{q(x_t|x_0)}\Big[\|\mu_\theta(x_t, \alpha_t) - \tilde{\mu}(x_t, x_0)\|_2^2\Big] \tag{7}$$

which is just a weighted mean squared error between the image produced from $p_\theta(x_t|x_0)$ and the true image given by the reverse diffusion process $q(x_{t-1}|x_t, x_0)$ for each time t.

In [18], they use a slightly different approach based on predicting the noise $\epsilon_\theta(x_t, t)$ in a given image x_t instead of denoising it.

Recall that the purpose of the training network is to approximate the conditioned probability distributions of the reverse diffusion process:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$$

Our goal is to train the network to predict $\tilde{\mu}$ of Equation (5). Since $x_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_t)$, we have

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}\Big(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)\Big) \tag{8}$$

and

$$x_{t-1} \sim \mathcal{N}(\mathbf{x}_{t-1}; \frac{1}{\sqrt{\alpha_t}}\Big(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x}_t, t)\Big), \Sigma_\theta(\mathbf{x}_t, t)) \tag{9}$$

The network can be simply trained to minimize the quadratic distance between the actual and the predicted error. Ignoring weighting terms, this seems to be irrelevant if not harmful in practice, thus the loss is:

$$L_t^{\text{simple}} = \mathbb{E}_{t\sim[1,T],x_0,\epsilon_t}\left[\|\epsilon_t - \epsilon_\theta(x_t,t)\|^2\right] \tag{10}$$

L_t^{simple} does not give any learning signal for $\Sigma_\theta(x_t,t)$. In [18], the authors preferred to fix it to a constant, testing both $\beta_t\mathbf{I}$ and $\tilde{\beta}_t\mathbf{I}$, with no sensible difference between the two alternatives.

3.2. Pseudocode

With the above setting, the algorithms for training and sampling are very simple. The network $\epsilon_\theta(x_t,t)$ inputs a noisy image x_t and time step t, and it is supposed to return the noise contained in the image. Suppose that you have a given noise scheduling $(\alpha_T,\ldots,\alpha_1)$. We can train the network in a supervised way, sampling a true image x_0, creating a noisy version of it $x_t = \sqrt{\alpha_t}x_0 + \sqrt{1-\alpha_t}\epsilon$ where $\epsilon \sim \mathcal{N}(0;I)$ and instructing the network to guess ϵ. Note that we only have a single network that is parametric in the time step t (or, since it is equivalent, parametric in α_t). The procedure is schematically described in Algorithm 1.

Algorithm 1: Training

1: Fix a noise scheduling $(\alpha_T,\ldots,\alpha_1)$
2: **repeat**
3: $x_0 \sim P_{DATA}$
4: $t \sim$Uniform(1, ..., T)
5: $\epsilon \sim \mathcal{N}(0;I)$
6: $x_t = \sqrt{\alpha_t}x_0 + \sqrt{1-\alpha_t}\epsilon$
7: Take gradient descent step on $||\epsilon - \epsilon_\theta(x_t,\alpha_t)||^2$
8: **until** converged

Generative sampling is performed through an iterative loop: we start with a purely noisy image x_T and progressively remove noise by means of the denoising network (see Algorithm 2). The denoised version of the image at time step t is obtained using Equation (9).

Algorithm 2: Sampling

1: $x_T \sim \mathcal{N}(0,I)$
2: **for** $t = T,\ldots,1$ **do**
3: $z \sim \mathcal{N}(0;I)$ if $t > 1$ else $z = 0$
4: $x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t,t)\right) + \sigma_t z$
5: **end for**

Several improvements can be made to this technique.

An important primary point concerns the noise scheduling $\{\alpha_t\}_{t=1}^T$. In [1], the authors used linear or quadratic schedules. This typically results in a very steep decrease during the initial time steps, which could be problematic for generation. In order to address this issue, alternative scheduling functions that incorporate a more gradual decrease, such as the 'cosine' or 'continuous cosine' schedule, have been proposed in the literature [47]. The precise choice of the scheduling function does not seem to matter, provided it shows a nearly linear behaviour in the middle of the generative process and smoother changes around the beginning and the end of the scheduling.

Another major issue concerns the speedup of the sampling process, which in the original approach was up to one or a few thousand steps. Since the generative model ap-

proximates the reverse of the inference process, in order to reduce the number of iterations required by the generative model, it could be worth rethinking the inference process. This investigation motivated the definition of Denoising Deterministic Implicit Models, which is explained in the following section.

3.3. Denoising Deterministic Implicit Models

Denoising Deterministic Implicit Models (DDIMs) [18] are a variation of the previous approach exploiting a non-Markovian noising process which has the same forward marginals as DDPM, but allows for better tuning of the variance of the reverse noise.

We start by defining the $q(x_{t-1}|x_t, \mathbf{x}_0)$ parametric with respect to a desired standard deviation σ:

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}x_0 + \sqrt{1-\bar{\alpha}_{t-1}}\epsilon_{t-1} \tag{11}$$

$$= \sqrt{\bar{\alpha}_{t-1}}x_0 + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\epsilon_t + \sigma_t\epsilon \tag{12}$$

$$= \sqrt{\bar{\alpha}_{t-1}}x_0 + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\frac{x_t - \sqrt{\bar{\alpha}_t}x_0}{\sqrt{1-\bar{\alpha}_t}} + \sigma\epsilon \tag{13}$$

So,

$$q_\sigma(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}; \mu_{\sigma_t}(x_0, \alpha_{t-1}), \sigma_t^2\mathbf{I}) \tag{14}$$

with

$$\mu_{\sigma_t}(x_0, \alpha_{t-1}) = \sqrt{\bar{\alpha}_{t-1}}x_0 + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\frac{x_t - \sqrt{\bar{\alpha}_t}x_0}{\sqrt{1-\bar{\alpha}_t}} \tag{15}$$

According to this approach, the forward process is no longer Markovian, but it depends both on the starting point x_0 and on x_{t-1}. However, it can be easily proved that the marginal distribution $q_\sigma(x_t|x_0) = \mathcal{N}(x_t|\sqrt{\bar{\alpha}_t}x_0; (1-\bar{\alpha}_t)\cdot I)$ recovers the same marginals as in DDPM. As a result, x_t can be diffused from x_0 and α_t by generating a realization of normally distributed noise $\epsilon_t \sim \mathcal{N}(\epsilon_t|0; I)$.

We can set $\sigma_t^2 = \eta \cdot \tilde{\beta}_t$, where η is a control parameter that can be used to tune sampling stochasticity. In the special case where $\eta = 0$, the sampling process becomes completely deterministic.

The sampling procedure in case of DDIM is slightly different from the case of DDPM. In order to sample x_{t-1} according to Equation (14), we need x_0, which is obviously unknown at the time of generation. However, since we are guessing the amount of noise $\epsilon(x_t, t)$ in x_t at each time point, we can generate a denoised observation $\tilde{x}_0$, which is a prediction of x_0 given that x_t:

$$\tilde{x}_0 = (x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(x_t, t)/\sqrt{\bar{\alpha}_t})$$

The full generative procedure is summarized in the pseudocode of Algorithm 3:

Algorithm 3: Sampling (DDIM)

1: $x_T \sim \mathcal{N}(0, I)$
2: **for** $t = T, ..., 1$ **do**
3: $\quad \epsilon = \epsilon_\theta(x_t, \bar{\alpha}_t)$
4: $\quad \tilde{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \frac{1-\bar{\alpha}_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon)$
5: $\quad x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\tilde{x}_0 + \sqrt{1-\bar{\alpha}_{t-1}}\epsilon$
6: **end for**

An interesting aspect of DDIM, which is frequently exploited in the literature, is that due to its deterministic nature, it also defines an implicit latent space, which opens the way for a very interesting operation comprising latent space interpolation, or the exploration of

interesting trajectories for editing purposes. The latent space can be obtained by integrating an ODE in the forward direction and then reverse the process to get the latent encodings that produce a given real image [4]. In [19], it was shown that a deep neural network can also be trained to perform this embedding operation, sensibly reducing its cost. We shall provide details on the embedding network in Section 4.1.

4. Model Architecture

As made clear in Section 3, the main component of a diffusion model is a denoising network, which inputs noise variance $\bar{\alpha}_t$, an image x_t corrupted with a corresponding amount of noise, and tries to guess the actual noise $\epsilon_\theta(x_t, \bar{\alpha}_t)$ present in the image. Starting from an image x_0 of the data distribution, we can generate a random noise $\epsilon \sim \mathcal{N}(0; I)$ and produce a corrupted version $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$. Then, the network is trained to minimize the distance between the actual noise ϵ and the predicted noise $\epsilon_\theta(x_t, \bar{\alpha}_t)$:

$$Loss = \|\epsilon - \epsilon_\theta(x_t, \bar{\alpha}_t)\|^2 \tag{16}$$

The architecture of this network is traditionally based on the U-net [48], a well-known convolutional neural network introduced, at its origin, for image segmentation tasks.

The U-net (see Figure 2) features an encoder–decoder structure, incorporating skip connections between layers of the encoder and decoder with corresponding spatial dimensions. We worked on images with an initial resolution of 64×64 [19].

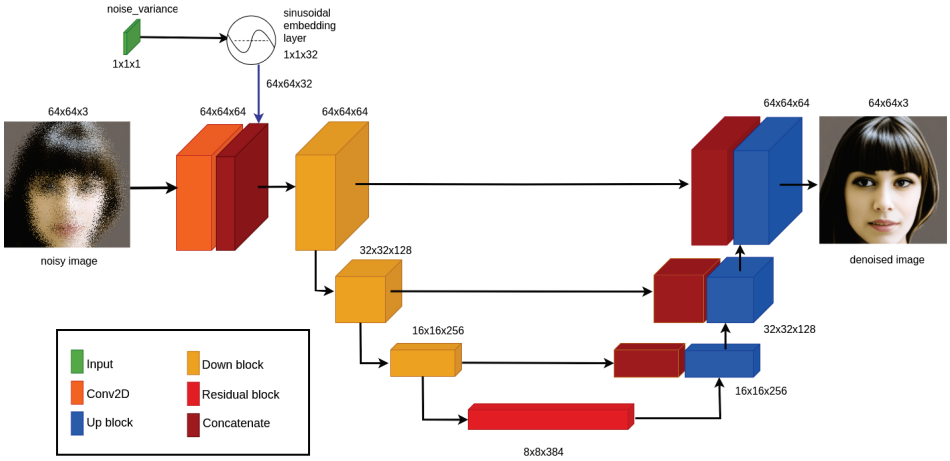


Figure 2. The U-net architecture of our denoising model.

The input relative to the noise variance α_t is typically embedded using sinusoidal position embeddings. Then, this information is vectorized and concatenated to the initial one. The detailed structures of the various modules is given in Figure 3.

4.1. The Embedding Network

In DDIM, the sampling process is deterministic given the initial noise x_T, so it is natural to try to address the reverse problem, computing x_T from x_0. The operation is not obvious, however, since the problem is clearly underconditioned, and we may have many different points in the latent space generating the same output.

The embedding problem for Denoising Models can be addressed in several different ways: by gradient ascent, integrating the ordinary differential equations (ODE) defining the forward direction, and then running the process in reverse to get the latent representations, or by directly trying to train a network to approximate the embedding task [19]. The advantage of the latter approach is that, once training is completed, the computation of the latent encoding is particularly fast.

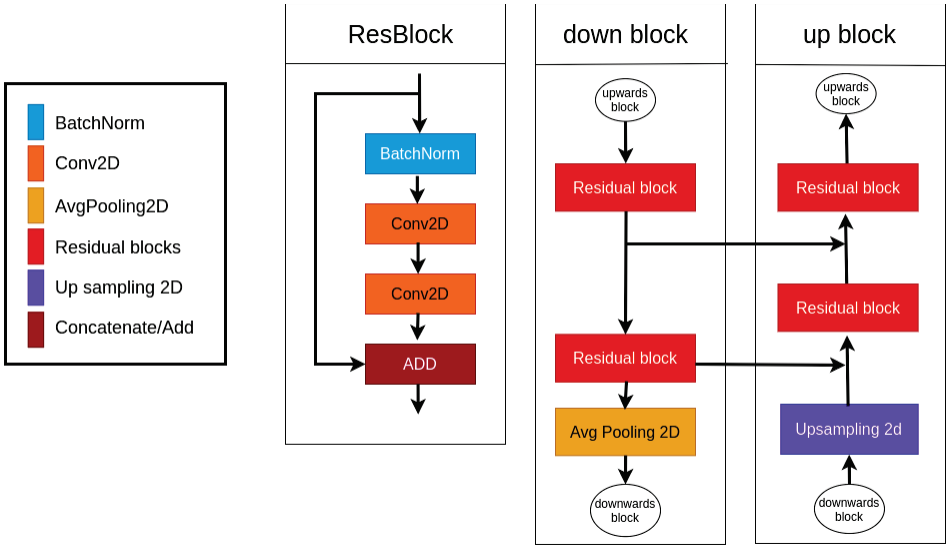


Figure 3. Main architectural modules, including the residual block, down block, and up block.

The embedding network $Emb(x)$ inputs an image and tries to compute its embedding. It is trained in a completely supervised way: given some noise x_T, we generate a sample x_0 and train the network to synthesize x_T from x_0, using as the loss the distance between x_T and $Emb(x_0)$. Modern neural computation environments enable us to effortlessly backpropagate gradients through the iterative loop of the reverse diffusion process.

Many different models of the Embedding Network were compared in [19]; the best results were obtained by the U-Net, essentially identical to the denoising network. The big difference with respect to generation is that we compute the latent representation with a single pass through the network. Reconstruction has an MSE around 0.0012 in the case of CelebA, which is definitely good.

5. CelebFaces Attributes Dataset

The CelebFaces Attributes dataset (CelebA) [49] consists of over 200,000 celebrity images, with a rich set of annotations. In the past, it has been widely used in the fields of computer vision and machine learning for tasks like attribute prediction, facial recognition, and generation. Each image is equipped with 40 binary attributes, covering characteristics like gender, age, hair color, and so on. Additionally, bounding box annotations for the faces are provided.

CelebA exhibits a wide diversity in image quality, resolution, and sources, capturing a broad spectrum of ethnicities, age groups, and genders. To facilitate the deep learning models' development, an aligned version of the dataset is offered, where faces are centered in a common coordinate system, ensuring consistent size and orientation.

CelebAMask-HQ [50] is a recent derivative of CelebA. It contains 30,000 high-resolution images, manually annotated with segmentation masks relative to 19 different facial components and accessories. This dataset serves as a valuable resource for training and evaluating face parsing, recognition, generation, and editing.

5.1. Analysis of the Dataset

Since a generative model is designed to model the distribution of data, it is always good practice to begin the study of a model with an analysis of the underlying dataset. Any bias in the data, such as an imbalance between different classes, will likely be reflected in the model, potentially leading to unexpected generative behaviors. For example, in the CelebA dataset, there is a noticeable gender imbalance, with a majority of female images over male images. Consequently, when editing faces, this bias can cause the model to more frequently transform male faces into female ones.

We compared and selected the relevant attributes for our investigation using the methodology described in Section 8. This approach aimed to assess the impact of each attribute in the direction of the trajectory. Ultimately, we focused on a subset of attributes that included gender, age, smiling, and "mouth slightly open".

In this section, we investigate the distribution of these attributes in CelebA; results are summarized in Figure 4.

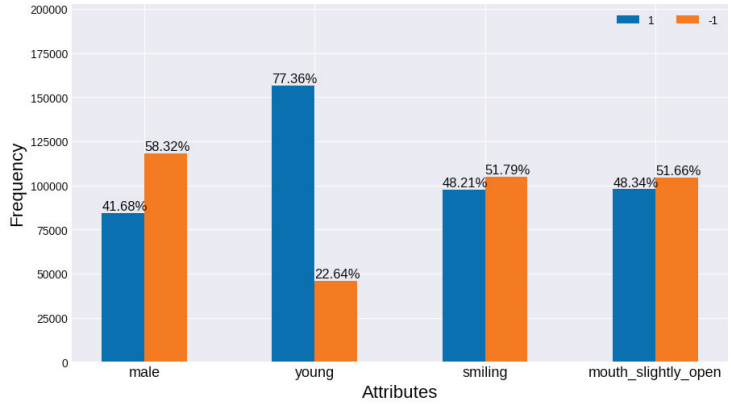


Figure 4. Distribution of CelebA attributes. In CelebA, each attribute is annotated with either -1 or 1. For example, for gender, "male = 1" stands for male, and "male = -1" stands for female.

With respect to some of the attributes, the dataset is highly unbalanced: approximately 58% of the images in the dataset depict females and around 77% feature young people. This imbalance could adversely affect the generative process, particularly when dealing with male and older people as compared to female and younger ones.

In the following sections, we provide further information relative to face orientation and illumination source not covered by the CelebA attributes.

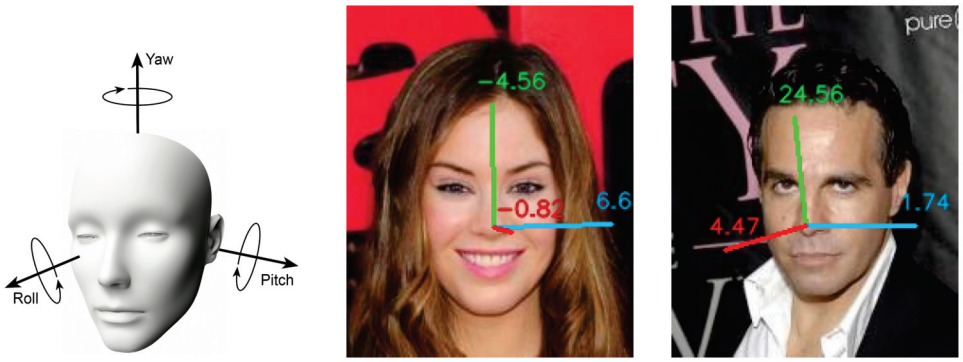
5.2. Face Orientation

To guide the DDIM generation process in producing faces with different orientations, information about head orientation from the CelebA dataset is needed. This information is not included in the standard annotations of the CelebA-aligned dataset. However, Head Pose Estimation is a well-investigated topic [51], and a large number of libraries are available for this purpose. In its usual formulation, the task includes expressing a person's head orientation in a three-dimensional space by calculating three rotation angles called yaw, pitch, and roll. Specifically, yaw denotes the rotation around the vertical axis, pitch is the rotation around the horizontal axis, and roll is the rotation around an axis perpendicular to the other two (see Figure 5a).

For the straightforward task of recognizing the orientation of nearly frontal faces, as is the case with the majority of faces in CelebA, there are open-source libraries that perform excellently. Specifically, we used the cv2.solvePnP() function from the OpenCV library [52], employing the same technique described in [53] and which is readily accessible in the public repository. In the same repositories, we also provide direct access to pre-computed angles for all images in the CelebA dataset.

An example of the kind of obtainable annotations is given in Figure 5.

More interestingly, we can examine the distribution of CelebA images concerning orientations, particularly yaw, as it represents the most significant rotation in the dataset: see Figure 6. CelebA is an aligned dataset: as expected, over 40% of the images have yaw within the $[-10, +10]$ degree range. Moreover, only 4.48% of the images have yaw outside the $[-40, +40]$ degree range. The limited number of images with high yaw values restricts the generative power of the model. Typically, rotations need to be confined within a region of the data with statistical significance, such as yaw in the $[-30, +30]$ degree range.



(**a**) (**b**)

Figure 5. (**a**) Yaw, Pitch, and Roll angles in HPE. (**b**) Examples of head pose estimation for CelebA images: yaw is in green, pitch in blue, and roll in red.

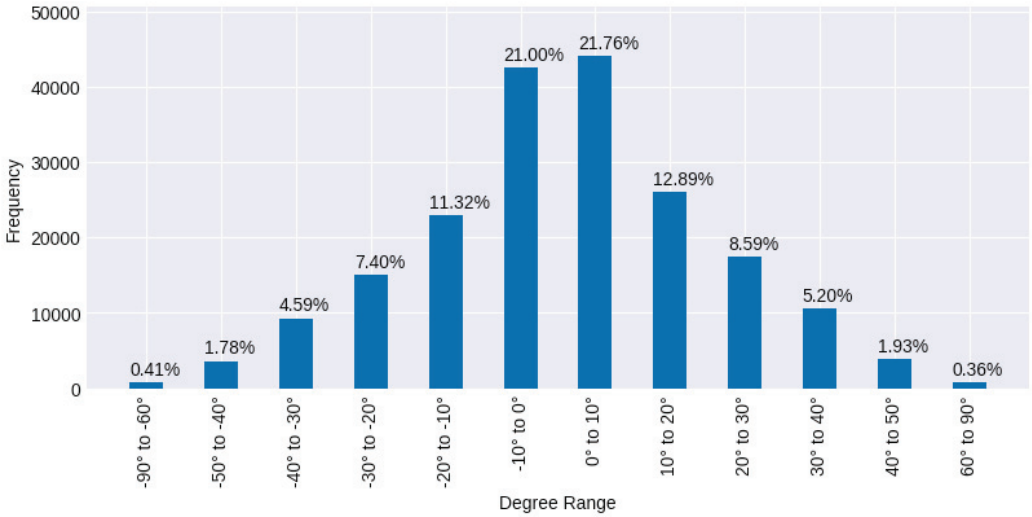


Figure 6. Yaw distribution in the CelebA dataset.

5.3. Light Direction Analysis

When rotating a face, it is crucial to preserve the right shadows produced by the lighting conditions. Unfortunately, no attributes are available relative to the source of illumination in the case of CelebA, and to our knowledge there is no open source software able to correctly identify lighting directions in an automatic way.

An important byproduct of our work is the provision of labels for CelebA, categorizing images into three major groups based on their main source of illumination: left, center, and right. The labeling process was carried out in a semi-supervised way over the last few years with the collaboration of many students. The basic procedure involved manually annotating a large portion of the data, developing and training classification models, cross-validating the data using different models, manually revising the classifications, and repeating the process until no further critical issues emerged.

Nevertheless, this labeling process has proven to be a valuable tool for our methodology and we hope it can serve as a significant asset for further investigations into face processing tasks. The labeling can be freely accessed through the code in the github repository. We also provide pre-computed yaw, pitch, and poll angles for each CelebA image.

In Figure 7, we summarize the outcome of our labeling and the complex interplay between illumination and orientation by showing the mean faces corresponding to different light sources and poses.

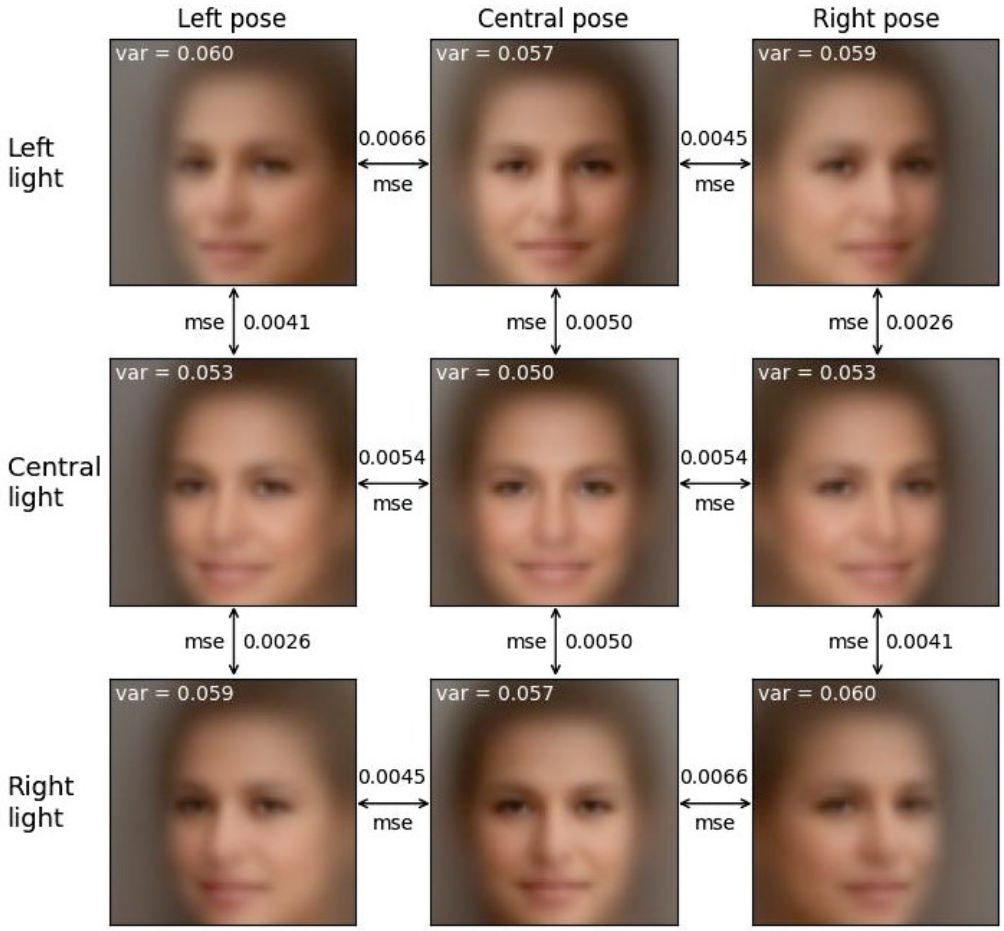


Figure 7. Illumination–Pose centroids. The different figures visualize the mean faces relative to different light sources and poses, considering three major orientation classes. We also report the variance in each class (mean of variances of the pixels) and the mean square error (MSE) between class centers.

In the picture, we also show the variance in each class (mean of variances of the pixels) and the squared Euclidean distance (MSE) between class centroids. We can observe the following points: (1) the different provenance of the light is still clearly recognizable in the mean faces, implicitly testifying to the quality of our labeling; (2) from the point of view of the position and shape of the shadows over the phase, the illumination and pose are strictly interconnected; and (3) the variance of each class is an order of magnitude larger than the distance between their centers, hinting to the complexity of the classification problem.

The second point is particularly important for our work. Investigations into the most relevant variables in the latent representation of images, including faces, have revealed that much of the information is conveyed by variables that explain macroattributes of the source image, such as colors and intensities of large regions (e.g., light/dark backgrounds, light/dark hair) [54]. The intensity and positions of dark/light regions on a face are strongly influenced by the source of illumination. Therefore, it is natural to expect that this information significantly impacts the latent encoding, and indeed, as is testified by this work, it does.

6. Methodology

The problem is finding trajectories in latent spaces corresponding to left/right rotations of the head.

Our starting point is a large dataset of head images enriched with information related to the rotation of the head and additional attributes such as lighting source, gender, age, and expression (smiling/not smiling). The selection of these attributes is discussed in Section 8. Images are preprocessed to remove the background as described in Section 7.

We also assume that there is a pre-trained generative model for the above dataset, along with an embedding tool capable of mapping an arbitrary sample of the dataset to its internal representation in the latent space of the generative model.

The other input is the image of the head to be rotated, let us call it X. Let Θ_X be its current rotation and let Z_X be its latent representation. This image can be one of the images in the dataset or a completely new one. In the latter case, its current rotation and its other attributes must be pre-computed and passed to the rotation model.

The methodology has the following steps:

- Filtering. We restrict the investigation to a subset of the dataset sharing the selected attributes with X. So, if X is a young, smiling blond man with a frontal illumination, we shall restrict the analysis to images sharing the same attributes.
- Clustering. Starting from Θ_X we create clusters of images with a rotation around $\Theta_X + \Delta$ for increasing values of Δ encompassing an overall rotation of $\pm 30^\circ$.
- Embedding and centroids computation. We embed the clusters in the latent space and compute their centroids. Each centroid conceptually corresponds to the latent representation of a "generic" person with the given attributes and rotations.
- Rotation trajectories. We defined rotation trajectories by fitting linear lines through the centroids using linear regression. We experimented with different spline segmentations, but ultimately obtained the best results by splitting the problem into two directions; one for right rotation and another for left rotation.
- Re-sourcing. The final step involves applying the trajectory vector corresponding to the rotation starting from Z_X. We sample points along this trajectory and generate the corresponding images in the visible space.

In order to improve the quality of the final image we post-process it for super resolution and color correction.

For a fixed rotation movement (left or right), the approach is schematically described in Figure 8. Our attempts to split the rotation in a larger number of linear steps have been hindered by the progressive loss of the key facial characteristics of the source image.

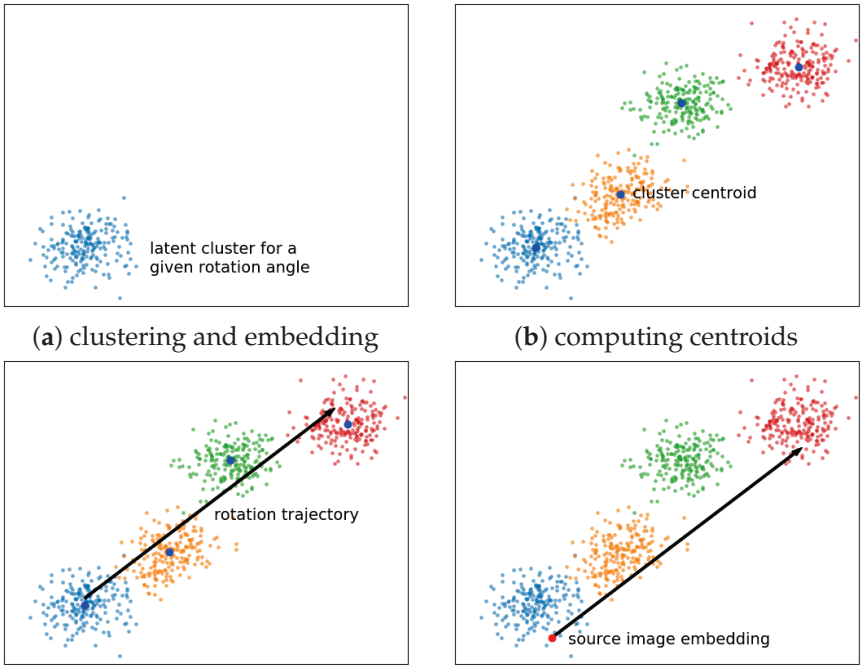
The clustering phase is not an essential part of the algorithm, since we could directly apply regression on the cloud of embedded points. We compute centroids mostly for debugging purposes to visualize and compare "generic" faces for a given set of attributes and rotations.

The number of angles relative to centroids and their intervals can be easily customized by the user. Using a step size that is too small typically reduces the number of images retrieved from the dataset that matches that specific orientation, thereby diminishing the statistical significance of the cluster. We usually work with a step size of 10°.

In the nest sections we provide additional details on some of the main steps of our methodology.

Filtering CelebA Images

In our first attempts, we selected images from the dataset just using the rotation. However, it is important to select images that have at least a rough similarity with the source image we want to act on. To this aim, we use a basic set of attributes comprising lighting source, gender, age, and expression (smiling/not smiling). In Figure 9 we show the different mean value of CelebA data relative to the different configurations of some of these attributes.



(**a**) clustering and embedding (**b**) computing centroids

(**c**) fitting trajectories (**d**) re-sourcing to the input

Figure 8. Overall methodology. All pictures refer to the latent space of the generative model, schematically represented in two dimensions. We also suppose that the images were pre-filtered along the relevant attributes. (**a**) We use the embedder to compute clusters of latent points corresponding to specific rotation angles, expressed by different colors; (**b**) we compute the centroids of the clusters; (**c**) we fit a line through the centroids to compute a rotation trajectory; and (**d**) we move along this direction starting from the specific embedding of the source images we want to rotate.

In order to obtain a sufficiently representative number of candidates, we typically enlarge the dataset via a flipping operation, consistently inverting the relevant attributes (yaw and lighting source). For example, if we are looking for images with a yaw of +30° and a light direction of 'RIGHT' we can also take into account images with a yaw of $-30°$ and a light direction of 'LEFT', provided we flip them. We aim to retrieve sets composed of at least 1K images, starting from a relatively narrow tolerance interval $[\Theta - \Delta, \Theta + \Delta]$ around the desired angle Θ and possibly enlarging Δ if required.

The relevance of exploiting the attributes is exemplified in Figure 10, where we compare rotations obtained selecting clouds of images according to rotations (first row), with the case where we refine the selection with relevant attributes (second row).

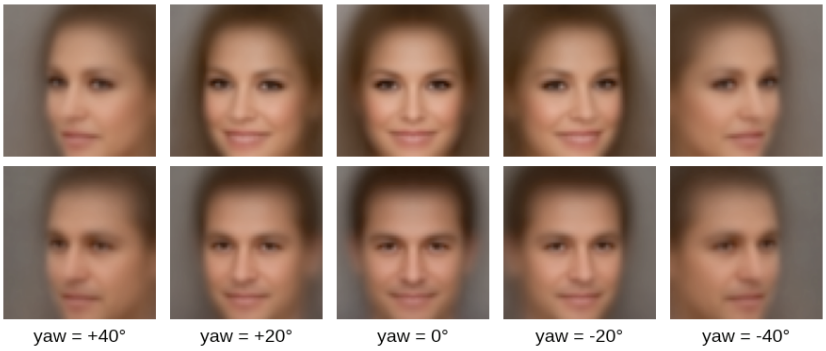


Figure 9. Mean images for specified yaw angles for females (top row) and males (bottom row).

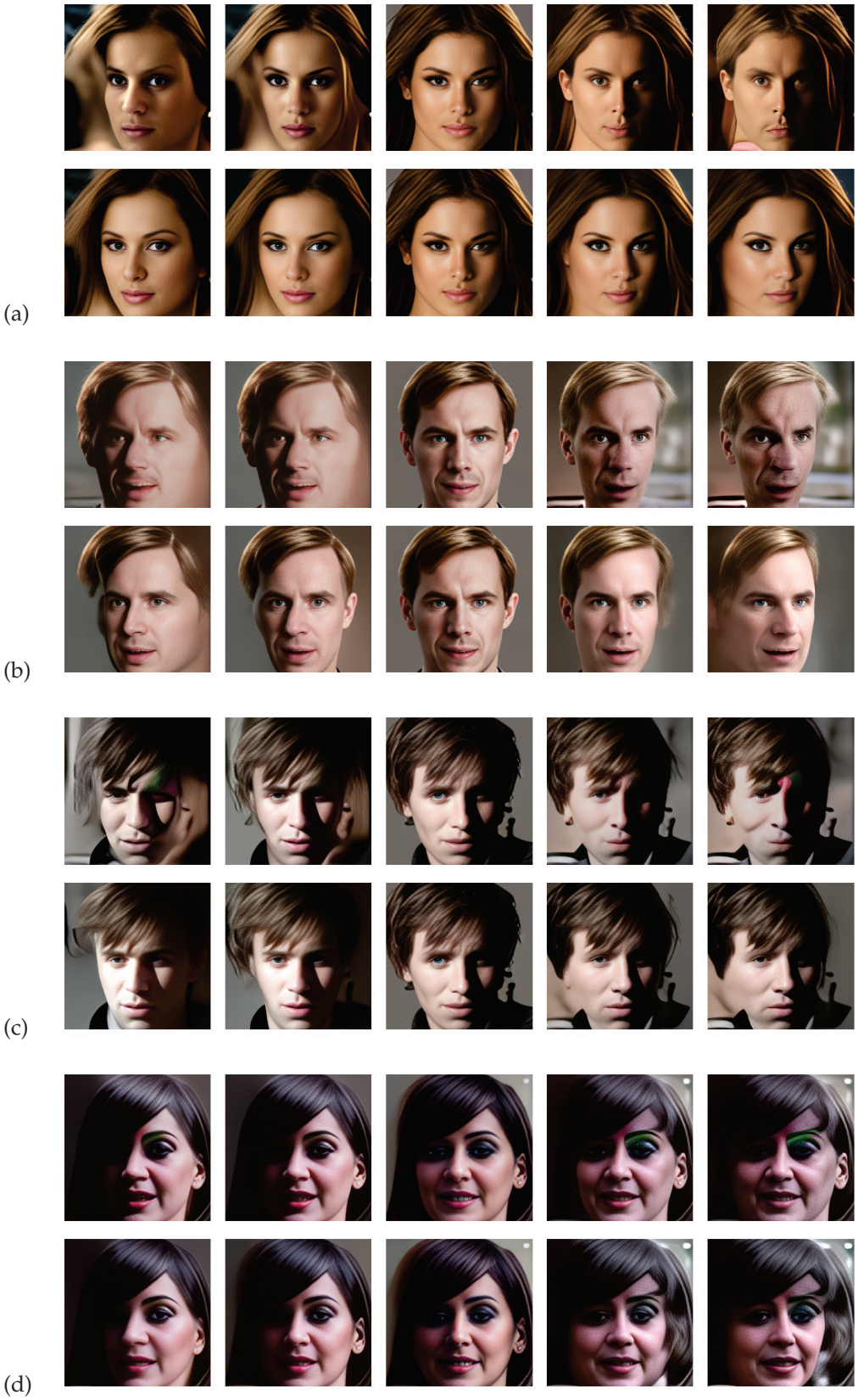


Figure 10. Relevance of attributes. For all images (**a**–**d**), the rotation in the first row corresponds to a trajectory computed considering angles, while the second one is relative to a trajectory taking attributes into account. These are examples of complex rotations due to the strong shadows over the face.

In Section 8, we provide a more technical comparison of the different trajectories in terms of their cosine similarity. We also used this metric as a way to select the most relevant attributes. There is a delicate balance between the specificity provided by attributes and the statistical relevance of the images retrieved from the dataset, which is essential for the regression phase. More details can be found in [55].

7. Preprocessing and Postprocessing

The deployment of the previous technique requires a few preprocessing and postprocessing steps, discussed in this section. Preprocessing is aimed at preparing inputs in a format suitable for the DDIM embedder, while post-processing is devoted to enhancing the quality of the result.

7.1. Preprocessing

In this article, we restrict the input to aligned CelebA images. We were able to generalize the approach to an arbitrary image provided by the user, as we did in [53], but the scientific added value is negligible.

Since the input image is already aligned we work with a central crop with a dimension of 128×128, which is frequently used in the literature [56], and then resized to a dimension of 64×64.

The main step of the preprocessing phase is background removal, since we experimentally observed that this operation facilitates rotation. To this aim, we trained a U-Net model on the CelebAMask-HQ dataset, which includes high-quality, manually annotated face masks. All masks were combined and treated as a binary segmentation problem, focusing on background/foreground separation (see Figure 11).

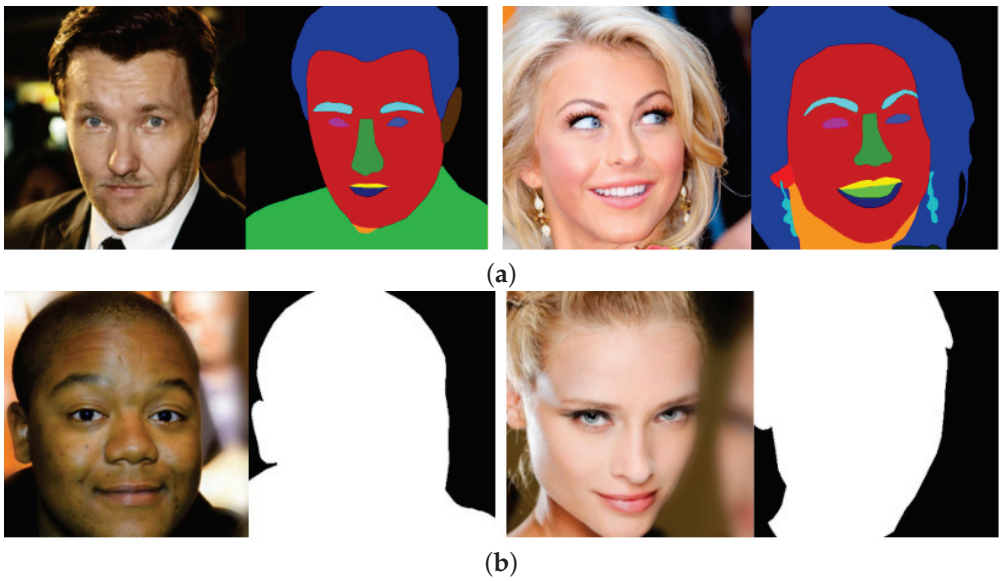


Figure 11. (**a**) Examples of CelebaMask-HQ segmentations and (**b**) cropped version with unified masks used to train the model on background removal tasks.

This approach allowed us to obtain a fairly precise segmentation of the facial region, with a precision of 96.78% and a recall of 97.60%.

7.2. Postprocessing

To enhance the final results, we crafted a postprocessing pipeline featuring two additional steps: super-resolution and color correction. This meticulous process ensures sharper details and more accurate colors.

7.2.1. Super-Resolution

The initial output, generated at a resolution of 64 × 64, is enhanced to 256 × 256 using CodeFormer [57], a recently introduced model known for its proficiency in Super-Resolution and Blind Face Restoration. CodeFormer amalgamates the strengths of transformers and codebooks to achieve remarkable results. Transformers have gained widespread popularity and application in natural language processing and computer vision tasks. On the other hand, codebooks serve as a method to quantize and represent data more efficiently in a compact form. The codebook is learned via self-reconstruction of the HQ faces using a vector-quantized autoencoder, which embeds the image into a representation capturing the rich HQ details required for face restoration.

The key advantage of employing Codebook Lookup Transformers for face restoration lies in their ability to capture and exploit the structural and semantic characteristics of facial images. By employing a pre-defined codebook that encapsulates facial features, the model proficiently restores high-quality face images from low-quality or degraded inputs, effectively handling various types of noise, artifacts, and occlusions.

7.2.2. Color Correction

The final step of the post-processing phase involves applying a color correction technique to reduce color discrepancies between the generated faces and their corresponding source images. This technique is essential for enhancing the overall visual coherence of the final result.

The color correction process leverages the Lab color space to match the color statistics of the two images. It begins by converting both images to the Lab color space. Then, the Lab channels of the target image are adjusted by normalizing them according to the mean and standard deviation of the source image. Finally, the target image is converted back to the RGB color space, ensuring that the colors of the generated face closely match those of the original source image.

Once the trajectory is identified, we move along it for a specified number of steps, checking the rotation after each iteration. If the generated image does not show the expected rotation, we try to dynamically increase the number of steps.

In case of images with a large initial yaw, we also apply a preliminary face frontalization phase.

8. Analysis of the Trajectory Slopes

In this section, we contrast the trajectory slopes within the latent space of Diffusion Models acquired through distinct attribute selections. We utilize cosine similarity as a synthetic metric to gauge the correlation between these trajectories.

We recall that we approximate trajectories using linear steps derived from linear regression conducted over the centroids of diverse clusters of data point embeddings. The selection of data points is based on yaw angles and various attributes, comprising the source of the illumination.

Figure 12 provides a visual representation of the variation of the trajectory slopes across varying ranges of rotation degrees. A heatmap is employed to graphically portray the level of resemblance between the trajectories, where distinct colors denote the magnitude of similarity.

As depicted in the figure, altering the degree ranges employed for cluster creation by 10° leads to a consistently diminishing cosine similarity between the slopes. The most significant disparity is observable between the intervals $[-40^\circ, 0^\circ]$ and $[0^\circ, +40^\circ]$. This means that the direction of the trajectory required to turn a face in the range $(-40^\circ, 0^\circ)$ is very different from the direction required to turn it in the range $(0^\circ, +40^\circ)$.

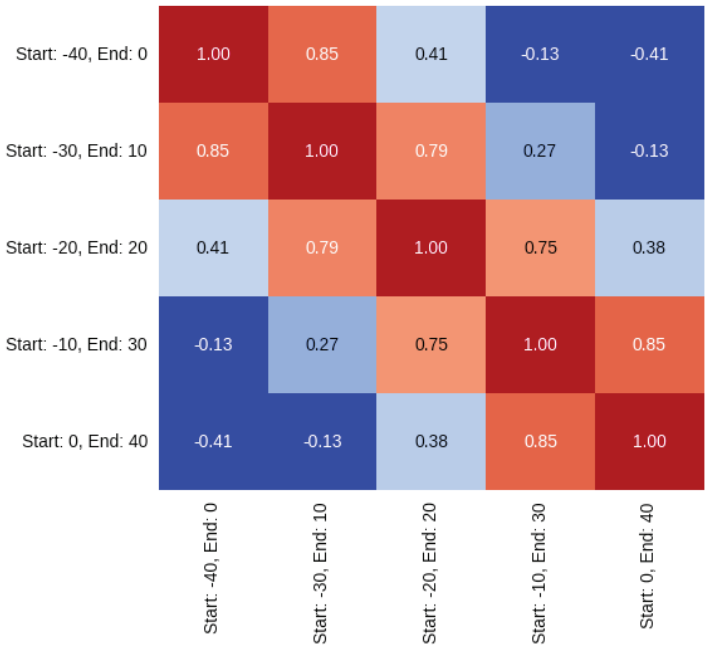


Figure 12. Cosine similarities between trajectory slopes. The different trajectories are obtained from data whose rotation yaw is comprised in the specified range.

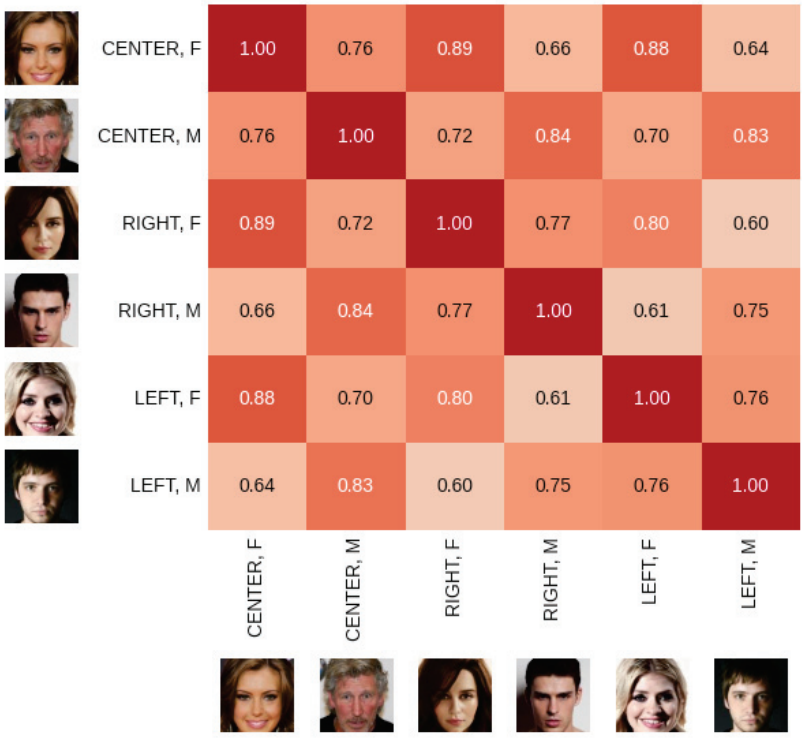
Based on the preceding analysis, it might appear that an incremental rotation approach involving frequent slope recalculations holds an advantage. Nevertheless, following the slope computation, it becomes necessary to translate the trajectory from centroids to the latent representation of the current image and move away from it. With each step, there is usually a gradual erosion of individual facial attributes. Thus, a trade-off arises: fewer steps could result in a relatively less precise rotations but better preservation of identity traits, while a greater number of steps could yield the opposite outcome.

According to our experimental findings, we obtained the best outcomes by just using two trajectories: the rightward trajectory $[0°, -40°]$ and the leftward trajectory $[0°, +40°]$. This is typically preceded by a frontalization step when required.

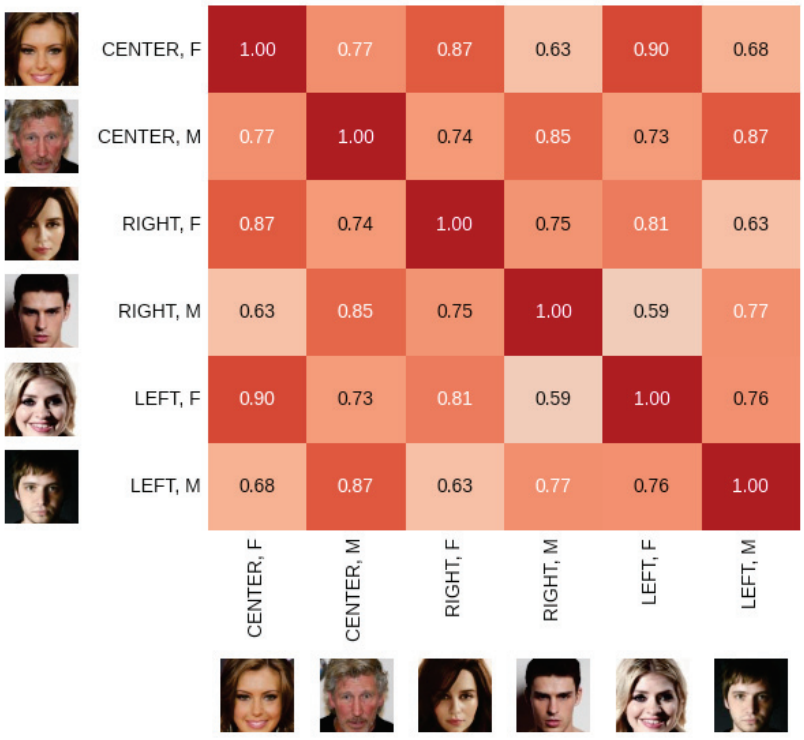
The remainder of this section is dedicated to evaluating the influence of auxiliary attributes on trajectory definition: does rotating a male head yield the same results as rotating a female one? What implications arise from factors such as age or face illumination?

To this aim, we fix an initial central pose as well as two fixed trajectories, rightward and leftward, and compare the slopes obtained selecting data points for centroids according to different attributes. Specifically, in Figure 13 we focus on gender and illumination, in Figure 14 we focus on illumination and age, and finally in Figure 15 we focus on gender, age, and smile. In all the Figures, the individual faces are merely meant to be representatives of their corresponding class of attributes and have no specific influence on the slope of the associated trajectory.

Figure 13 presents a heatmap that illustrates the cosine similarities between the left and right slopes computed on clusters of images taking into account only two attributes: light direction (center, left, or right) and gender (male or female). Gender is the most important factor of variation, more relevant than illumination source. Still, images with disparate light directions results in sensibly different trajectories.



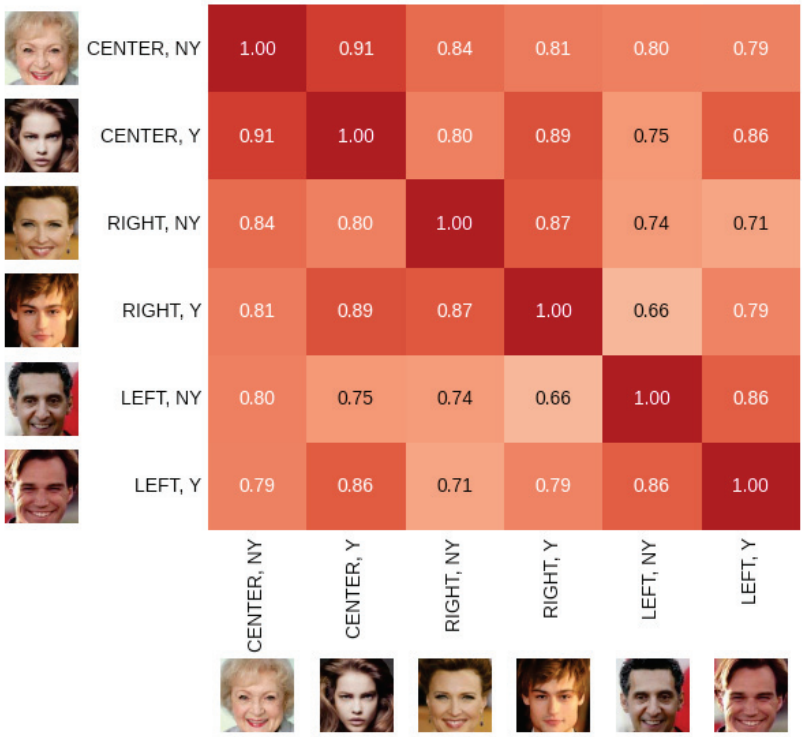
(**a**)



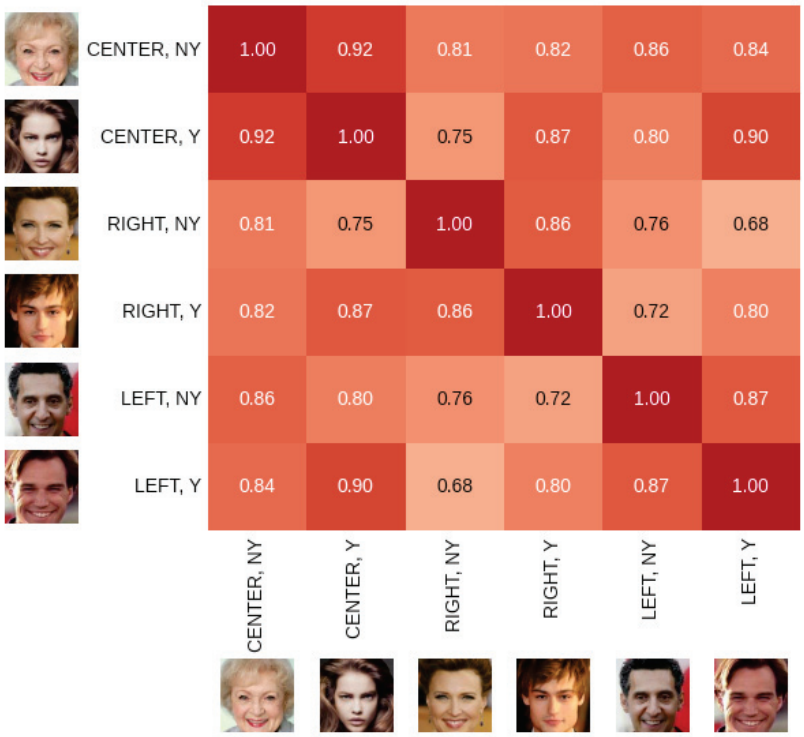
(**b**)

Figure 13. Heatmap of Cosine similarities between left (**a**) and right (**b**) slopes, computed using only light direction (CENTER, LEFT, or RIGHT) and gender (M or F) as attributes. Each subset of attributes is represented by a corresponding sample from CelebA.

These results suggest that both light direction and gender are important attributes for enhancing accuracy in trajectory calculation.



(**a**)

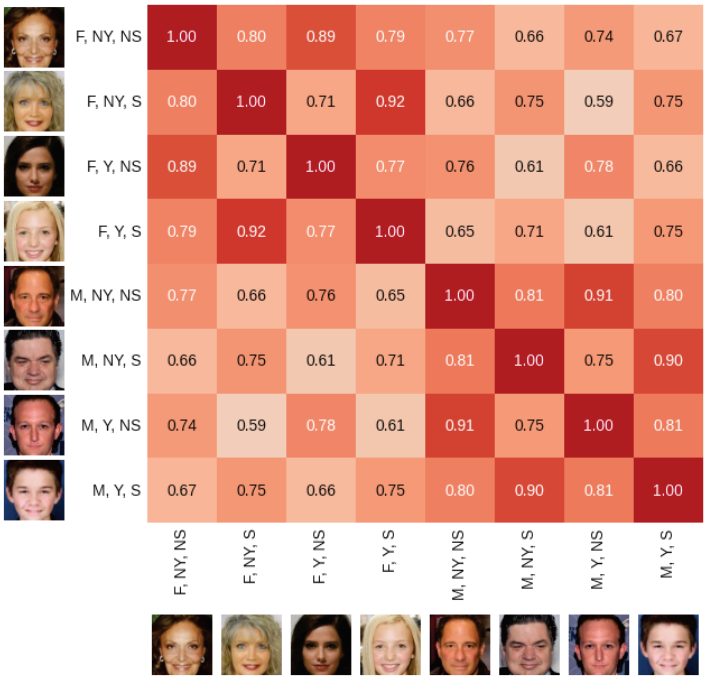


(**b**)

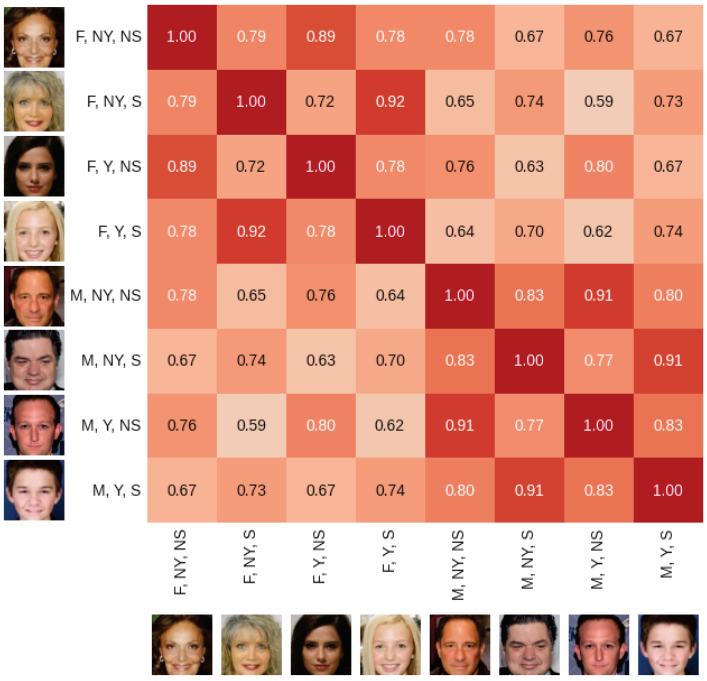
Figure 14. Heatmap of cosine similarities between left (**a**) and right (**b**) slopes, computed using only light direction (CENTER, LEFT, or RIGHT) and youthfulness (Y or NY) as attributes.

Figure 14 parallels the previous concept, differing solely in the replacement of gender with age (young or not young) as the attribute under consideration.

As depicted by the figure, a consistent trend emerges: diminished similarity values occur when the light direction is in contrast, and the same trend applies to the age parameter. Nevertheless, it is noteworthy that age has a relatively lower impact on trajectory definition.



(**a**)



(**b**)

Figure 15. Heatmap of cosine similarities between left (**a**) and right (**b**) slopes, computed with light direction fixed to center and using only gender (M or F), youthfulness (Y or NY), and smiling (S or NS) as attributes.

Figure 15 embarks on a more intricate exploration of trajectory attributes. In this context, we delve into a refined set of attributes: gender (M or F), youthfulness (Y or NY), and smiling (S or NS).

Similar to previous instances, the lowest similarity emerges when all attributes stand distinct from each other. For instance, a comparison between a female who is not young and not smiling ([F, NY, NS]) and a male who is young and not smiling ([M, Y, NS]) represents the scenario with the least similarity. Furthermore, upon closer examination of the image, it becomes apparent that gender exerts the most significant impact on similarity, closely trailed by the "smiling" attribute.

In our work, we consistently employed the aforementioned technique to methodically select the most pertinent attributes for delineating trajectories within the latent space.

9. Results and Troubleshooting

Measuring the quality of the generative systems is a notoriously difficult task due to the lack of a ground truth to use as a comparison. This is particularly difficult in the case of the rotation operation, where we must assess both the model's capacity to obtain the desired orientation and the fidelity of the target to the source sample. Traditional metrics used in the field of generative modeling, like the Fréchet Inception Distance (FID), cannot be used in this context, since they are designed to compare distributions of data, not individual samples. In our case, the rotation measured on the generated sample is a parameter used to control the short iterative loop governing the computation of the trajectory; so, apart from a few cases where the algorithm fails to achieve the desired rotation and is forcibly stopped, the rotation of the result is the one expected.

The difficult task is to quantify the similaritiesxsc z of the individual features of the target with those of the source. We are currently doing experiments with the Feature Similarity Index (FSIM) [58], the Identity Preservation Metric [59], ArcFace's Additive Angular Margin Loss [60], and the Learned Perceptual Image Patch Similarity (LPIPS) [61]. All of them are valuable, but they also suffer from well-known limitations: FSIM and similar metrics like SSIM are based on local patterns and luminance but may not adequately capture global context or the perceptual importance of different image regions; the Identity Preservation Metric heavily depends on the facial recognition or feature extraction model used, while the Learned Perceptual Image Patch Similarity (LPIPS) can be significantly influenced by the diversity and representativeness of the training dataset used for the neural networks that underpin this metric. We shall report on these quantitative analyses in a forthcoming paper.

Also, a qualitative comparison with similar GAN-based architectures is problematic. As observed in [54], state-of-the-art GANs, especially when trained on CelebA-HQ, seem to have serious generative deficiencies: many images from CelebA seem to lie outside their generative range. This means that it is not always possible to embed a generic face in the latent space and reconstruct an image with sufficient similarities.

In this preliminary report, we shall just showcase the promising potential of our approach through some examples; the reader is also invited to test the system, freely available on GitHub at https://github.com/asperti/Head-Rotation.

Some examples of rotations are given in Figure 16. More examples are given in Appendix A. In general, the technique still suffers from of a few notable problems, and we shall devote the remaining part of this section to their discussion.

Figure 16. Examples of rotations.

9.1. Loss of Individual's Features

Preserving facial features while rotating the image of a large angle poses a significant challenge. This becomes especially problematic when employing a latent-based approach, where the essential traits of an individual are solely captured in the source point coordinates. Consequently, there is a risk of losing these key characteristics while following a given path, often resulting in more generic and less distinct features. Figure 17 illustrates this phenomenon. While the right rotation (from the observer's point of view) appears reasonably accurate, the left rotation exhibits a gradual loss of the individual's features.

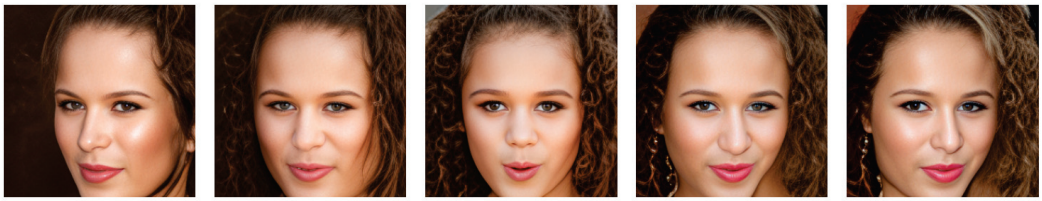


Figure 17. Troubleshooting: difficulty in preserving facial features.

9.2. High Pitch and Roll

In the presence of head poses with high pitch or roll (not very frequent in CelebA), the technique can encounter serious troubles, as exemplified in Figure 18.

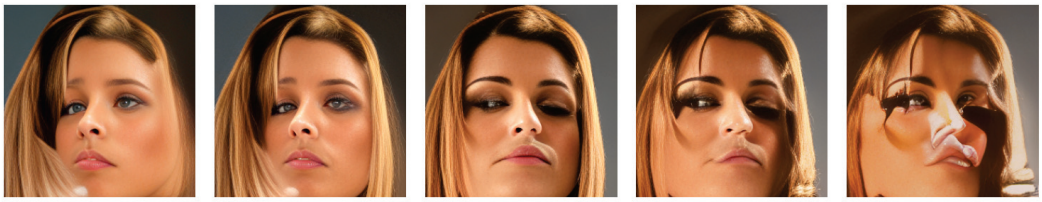


Figure 18. Troubleshooting: problems with high pitch or rolls.

Usually, the method tries to either correct the anomalous angles, as seen along the left rotation, or simply gets lost, as seen along the right rotation.

9.3. Hats and Other Artifacts

The generative model does not seem to have enough semantic information to handle situations involving the presence of artifacts such as microphones, hats, or any kind of headgear (see Figure 19).

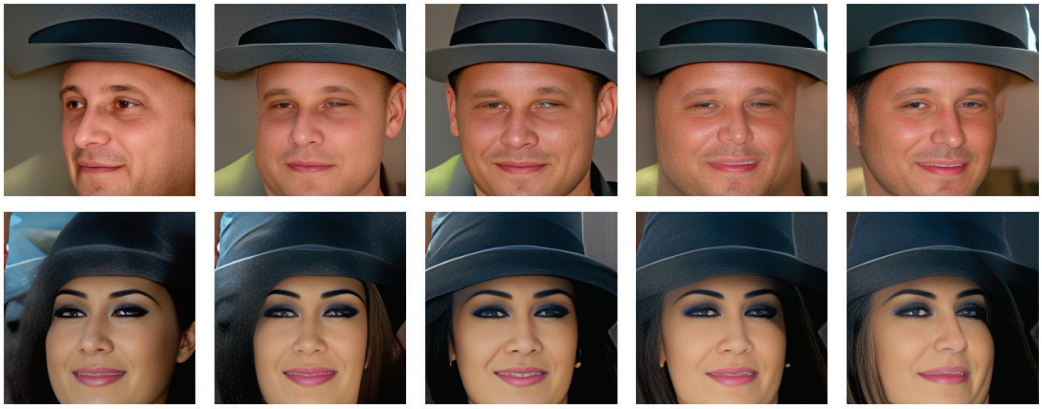


Figure 19. Troubleshooting: problems with hats and other artifacts.

Also, glasses over the head are usually a problem, as exemplified in Figure 20.

Figure 20. Troubleshooting: cannot rotate glasses over the head.

In the same figure, you may also observe the progressive loss of identity and change in expression along the left rotation. Glasses over the eyes may sometimes be lost during rotation, but otherwise they are handled correctly. Several examples are given in the Appendix A.

9.4. Deformation and Loss of Contours

Rotation may sometimes introduce anomalous deformations in the shape of the head; additionally, the technique is frequently unable to define precise contours for the face under extreme yaw angles. Both phenomena are evident in Figure 21.

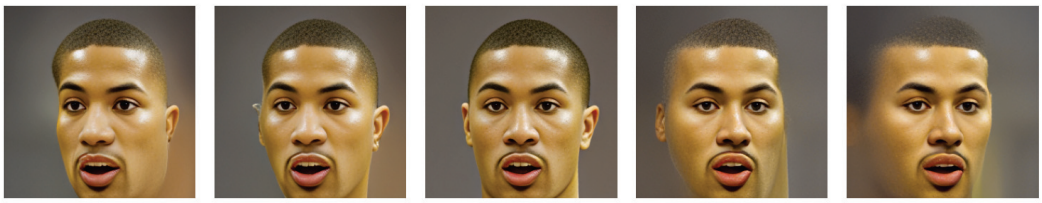


Figure 21. Troubleshooting: deformation and loss of contours.

9.5. Difficulty in Rotating Neck and Ears

The technique sometimes has trouble correctly rotating the neck or ears of subjects. They may get detached from the actual figure, remaining in the "background". This is illustrated in Figure 22.

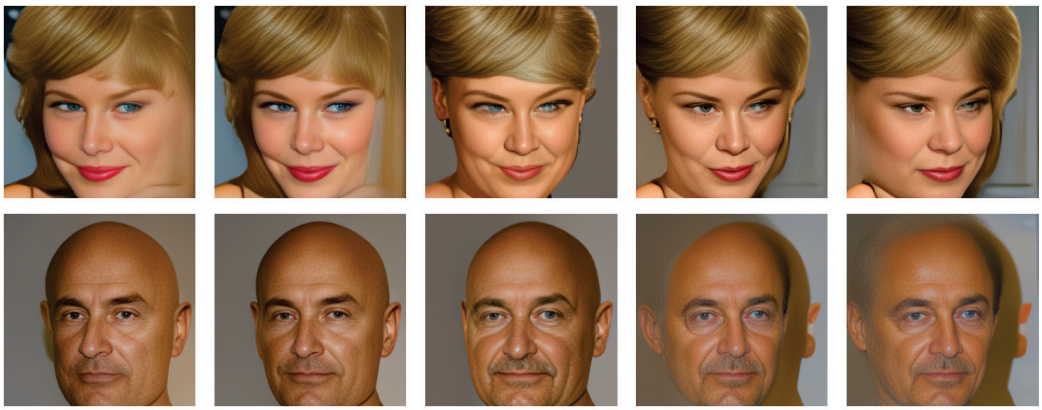


Figure 22. Troubleshooting: problems with neck and ears.

Sometimes, a similar situation also happens with hair.

10. Conclusions

This work contributes to the investigation of trajectories in the latent space of generative models, with particular attention to editing operations not easily expressible in terms of the texture, color, or shapes of well-identified segmentation areas that require holistic manipulation of the image. Head rotation, especially intended as a continuous transformation, is a typical example of these kinds of manipulations. Our investigation

suggests that identifying compelling trajectories relies on recognizing relevant attributes of the source image that can guide the statistical search in latent space. Among these relevant attributes, in the case of head rotation, the direction of the illumination plays a crucial role, creating complex shadowing effects on the face that are difficult to manage during rotation. Emphasizing the importance of lighting conditions for achieving realistic generative results in head rotation is one of the contributions of this work.

As a side result of our research, we created additional labels for the CelebA dataset, categorizing images into three groups based on the prevalent illumination direction: left, right, or center. Since CelebA-HQ is a well-known subset of CelebA, our labeling can be easily extended to the former dataset.

Our work is at a preliminary stage, and many aspects deserve further investigation.

Firstly, the current version lacks a robust quantitative evaluation and a thorough comparison with alternative techniques. Secondly, continuous movements can be better investigated in a video setting, a research field that has undergone remarkable achievements in recent years, mostly thanks to stable diffusion techniques [62–64]. Specifically, exploiting the spatio-temporal coherence of adjacent frames could help in understanding the global structure and 3D perspective, which become particularly useful when dealing with artifacts such as hats, earrings, or eyeglasses.

In the context of video generation, our work could contribute to extracting a dataset of difficult cases, especially in terms of light conditions that could pose interesting challenges for generative models. This dataset would be valuable for testing and improving the robustness of generative models in handling complex scenarios, thereby advancing the field of video-based generative modeling.

Furthermore, our preliminary findings indicate that incorporating detailed lighting information into the generative process significantly enhances the realism of generated images. Future work should focus on developing more sophisticated methods for capturing and utilizing lighting attributes in the latent space. This includes exploring the use of advanced neural network architectures and loss functions specifically designed to preserve lighting consistency during image manipulation.

Additionally, we plan to extend our investigation to other types of holistic image manipulations beyond head rotation, such as changing facial expressions or body poses, which also require careful consideration of lighting and other contextual factors. By addressing these challenges, we contribute to providing a comprehensive framework for holistic image manipulation in generative models.

Author Contributions: Conceptualization, A.A.; Methodology, A.A., G.C. and A.G.; Software, G.C. and A.G.; Validation, G.C.; Formal analysis, A.G.; Data curation, G.C.; Writing—original draft, A.A., G.C. and A.G.; Supervision, A.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially funded by the Future AI Research (FAIR) project of the National Recovery and Resilience Plan (NRRP), Mission 4 Component 2 Investment 1.3 funded from the European Union-NextGenerationEU.

Data Availability Statement: The application described in this paper is open source. The software can be downloaded from the following github repository: https://github.com/asperti/Head-Rotation.

Acknowledgments: We would like to thank the many students who helped in the annotation of CelebA for illumination orientation, and in particular L. Bugo, D. Filippini and A. Rossolino.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Additional Rotation Examples

In this appendix we provide a short list of additional examples of rotations obtained by means of our model.

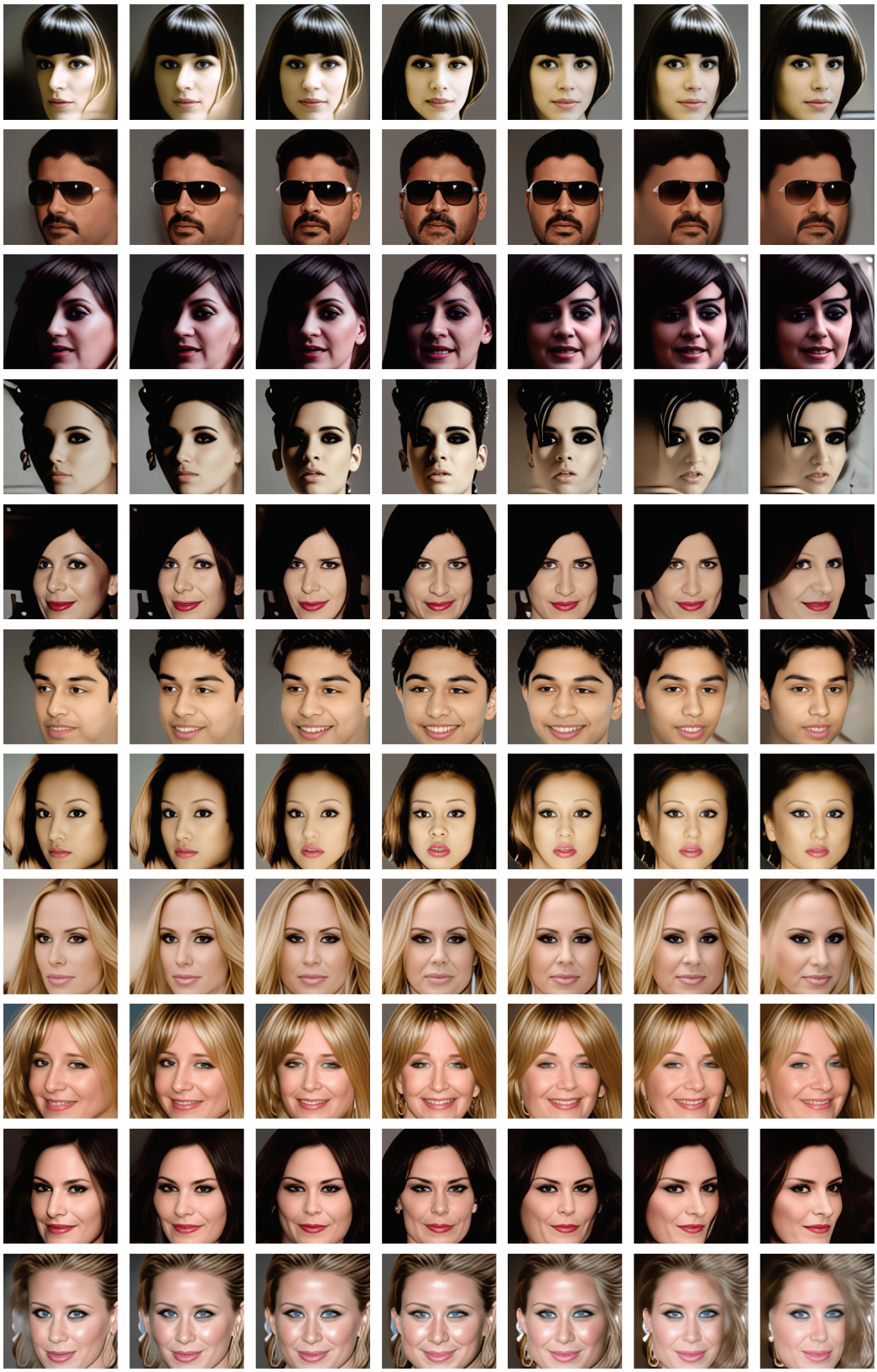


Figure A1. Examples of rotations.

Figure A2. Examples of rotations.

References

1. Ho, J.; Jain, A.; Abbeel, P. Denoising Diffusion Probabilistic Models. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 6840–6851.
2. Choi, J.; Kim, S.; Jeong, Y.; Gwon, Y.; Yoon, S. ILVR: Conditioning Method for Denoising Diffusion Probabilistic Models. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, 10–17 October 2021; pp. 14347–14356. [CrossRef]
3. Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; Ommer, B. High-resolution image synthesis with latent diffusion models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 10684–10695.
4. Dhariwal, P.; Nichol, A.Q. Diffusion Models Beat GANs on Image Synthesis. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 8780–8794.
5. Eschweiler, D.; Yilmaz, R.; Baumann, M.; Laube, I.; Roy, R.; Jose, A.; Brückner, D.; Stegmaier, J. Denoising diffusion probabilistic models for generation of realistic fully-annotated microscopy image datasets. *PLoS Comput. Biol.* **2024**, *20*, e1011890. [CrossRef]
6. Shokrollahi, Y.; Yarmohammadtoosky, S.; Nikahd, M.M.; Dong, P.; Li, X.; Gu, L. A Comprehensive Review of Generative AI in Healthcare. *arXiv* **2023**, arXiv:2310.00795.
7. Trippe, B.L.; Yim, J.; Tischer, D.; Baker, D.; Broderick, T.; Barzilay, R.; Jaakkola, T.S. Diffusion Probabilistic Modeling of Protein Backbones in 3D for the motif-scaffolding problem. In Proceedings of the the Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, 1–5 May 2023.
8. Zhao, Z.; Dong, X.; Wang, Y.; Hu, C. Advancing Realistic Precipitation Nowcasting With a Spatiotemporal Transformer-Based Denoising Diffusion Model. *IEEE Trans. Geosci. Remote Sens.* **2024**, *62*, 1–15. [CrossRef]

9. Shen, Y.; Gu, J.; Tang, X.; Zhou, B. Interpreting the Latent Space of GANs for Semantic Face Editing. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, 13–19 June 2020; Computer Vision Foundation/IEEE: Piscataway, NJ, USA, 2020; pp. 9240–9249. [CrossRef]
10. Härkönen, E.; Hertzmann, A.; Lehtinen, J.; Paris, S. GANSpace: Discovering Interpretable GAN Controls. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 9841–9850.
11. Li, Z.; Tao, R.; Wang, J.; Li, F.; Niu, H.; Yue, M.; Li, B. Interpreting the Latent Space of GANs via Measuring Decoupling. *IEEE Trans. Artif. Intell.* **2021**, *2*, 58–70. [CrossRef]
12. Shen, Y.; Yang, C.; Tang, X.; Zhou, B. InterFaceGAN: Interpreting the Disentangled Face Representation Learned by GANs. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *44*, 2004–2018. [CrossRef] [PubMed]
13. Creswell, A.; Bharath, A.A. Inverting the Generator of a Generative Adversarial Network. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 1967–1974. [CrossRef] [PubMed]
14. Alaluf, Y.; Tov, O.; Mokady, R.; Gal, R.; Bermano, A. Hyperstyle: Stylegan inversion with hypernetworks for real image editing. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 18511–18521.
15. Xia, W.; Zhang, Y.; Yang, Y.; Xue, J.H.; Zhou, B.; Yang, M.H. Gan inversion: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *45*, 3121–3138. [CrossRef]
16. Kingma, D.P.; Welling, M. An Introduction to Variational Autoencoders. *Found. Trends Mach. Learn.* **2019**, *12*, 307–392. [CrossRef]
17. Asperti, A.; Evangelista, D.; Piccolomini, E.L. A Survey on Variational Autoencoders from a Green AI Perspective. *SN Comput. Sci.* **2021**, *2*, 301. [CrossRef]
18. Song, J.; Meng, C.; Ermon, S. Denoising Diffusion Implicit Models. In Proceedings of the 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, 3–7 May 2021.
19. Asperti, A.; Evangelista, D.; Marro, S.; Merizzi, F. Image Embedding for Denoising Generative Models. *Artif. Intell. Rev.* **2023**, *56*, 14511–14533. [CrossRef]
20. Hassner, T.; Harel, S.; Paz, E.; Enbar, R. Effective face frontalization in unconstrained images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 4295–4304.
21. Zhu, X.; Lei, Z.; Yan, J.; Yi, D.; Li, S.Z. High-fidelity pose and expression normalization for face recognition in the wild. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 787–796.
22. Moniz, J.R.A.; Beckham, C.; Rajotte, S.; Honari, S.; Pal, C. Unsupervised depth estimation, 3d face rotation and replacement. In Proceedings of the Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, Montréal, QC, Canada, 3–8 December 2018; pp. 9759–9769
23. Tran, L.; Yin, X.; Liu, X. Disentangled representation learning gan for pose-invariant face recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–27 July 2017; pp. 1415–1424.
24. Huang, R.; Zhang, S.; Li, T.; He, R. Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2439–2448.
25. Hu, Y.; Wu, X.; Yu, B.; He, R.; Sun, Z. Pose-guided photorealistic face rotation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8398–8406.
26. Qian, Y.; Deng, W.; Hu, J. Unsupervised face normalization with extreme pose and expression in the wild. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 9851–9858.
27. Yin, X.; Yu, X.; Sohn, K.; Liu, X.; Chandraker, M. Towards large-pose face frontalization in the wild. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 3990–3999.
28. Deng, J.; Cheng, S.; Xue, N.; Zhou, Y.; Zafeiriou, S. Uv-gan: Adversarial facial uv map completion for pose-invariant face recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7093–7102.
29. Cao, J.; Hu, Y.; Zhang, H.; He, R.; Sun, Z. Learning a high fidelity pose invariant model for high-resolution face frontalization. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 2872–2882.
30. Zhou, H.; Liu, J.; Liu, Z.; Liu, Y.; Wang, X. Rotate-and-render: Unsupervised photorealistic face rotation from single-view images. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 5911–5920.
31. Mildenhall, B.; Srinivasan, P.P.; Tancik, M.; Barron, J.T.; Ramamoorthi, R.; Ng, R. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In Proceedings of the ECCV, Online, 23–28 August 2020.
32. Sun, J.; Wang, X.; Zhang, Y.; Li, X.; Zhang, Q.; Liu, Y.; Wang, J. FENeRF: Face Editing in Neural Radiance Fields In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, 18–24 June 2022; pp. 7662–7672.
33. Abdal, R.; Zhu, P.; Mitra, N.J.; Wonka, P. Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. *ACM Trans. Graph. (ToG)* **2021**, *40*, 1–21. [CrossRef]
34. Gal, R.; Alaluf, Y.; Atzmon, Y.; Patashnik, O.; Bermano, A.H.; Chechik, G.; Cohen-Or, D. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv* **2022**, arXiv:2208.01618.

35. Gal, R.; Patashnik, O.; Maron, H.; Bermano, A.H.; Chechik, G.; Cohen-Or, D. StyleGAN-NADA: CLIP-guided domain adaptation of image generators. *ACM Trans. Graph.* **2022**, *41*, 141:1–141:13. [CrossRef]
36. Morita, R.; Zhang, Z.; Ho, M.M.; Zhou, J. Interactive Image Manipulation with Complex Text Instructions. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2023, Waikoloa, HI, USA, 2–7 January 2023; pp. 1053–1062. [CrossRef]
37. Kalatzis, D.; Eklund, D.; Arvanitidis, G.; Hauberg, S. Variational autoencoders with riemannian brownian motion priors. *arXiv* **2020**, arXiv:2002.05227.
38. Chadebec, C.; Allassonnière, S. A geometric perspective on variational autoencoders. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 19618–19630.
39. Shamsolmoali, P.; Zareapoor, M.; Zhou, H.; Tao, D.; Li, X. Vtae: Variational transformer autoencoder with manifolds learning. *IEEE Trans. Image Process.* **2023**, *32*, 4486–4500. [CrossRef]
40. Hertz, A.; Mokady, R.; Tenenbaum, J.; Aberman, K.; Pritch, Y.; Cohen-Or, D. Prompt-to-Prompt Image Editing with Cross-Attention Control. In Proceedings of the The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, 1–5 May 2023.
41. Zhang, Z.; Han, L.; Ghosh, A.; Metaxas, D.N.; Ren, J. SINE: SINgle Image Editing with Text-to-Image Diffusion Models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, 17–24 June 2023; pp. 6027–6037. [CrossRef]
42. Kawar, B.; Zada, S.; Lang, O.; Tov, O.; Chang, H.; Dekel, T.; Mosseri, I.; Irani, M. Imagic: Text-Based Real Image Editing with Diffusion Models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, 17–24 June 2023; pp. 6007–6017. [CrossRef]
43. Couairon, G.; Verbeek, J.; Schwenk, H.; Cord, M. DiffEdit: Diffusion-based semantic image editing with mask guidance. In Proceedings of the The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, 1–5 May 2023.
44. Sanseviero, O.; Cuenca, P.; Passos, A.; Whitaker, J. *Hands-On Generative AI with Transformers and Diffusion Models*; O'Reilly Media, Inc.: Sebastopol, CA, USA, 2024.
45. Bishop, C.M.; Bishop, H. Diffusion Models. In *Deep Learning: Foundations and Concepts*; Springer: Berlin/Heidelberg, Germany, 2023; pp. 581–607.
46. Sohl-Dickstein, J.; Weiss, E.A.; Maheswaranathan, N.; Ganguli, S. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. In *JMLR Workshop and Conference Proceedings, Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6–11 July 2015*; JMLR: New York, NY, USA, 2015; Volume 37, pp. 2256–2265.
47. Nichol, A.Q.; Dhariwal, P. Improved denoising diffusion probabilistic models. In Proceedings of the International Conference on Machine Learning, Virtual, 18–24 July 2021; PMLR: New York, NY, USA, 2021; pp. 8162–8171.
48. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, Munich, Germany, 5–9 October 2015; Springer: Berlin/Heidelberg, Germany, 2015; pp. 234–241.
49. Liu, Z.; Luo, P.; Wang, X.; Tang, X. Deep Learning Face Attributes in the Wild. In Proceedings of the International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 3730–3738.
50. Lee, C.H.; Liu, Z.; Wu, L.; Luo, P. Maskgan: Towards diverse and interactive facial image manipulation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 5549–5558.
51. Asperti, A.; Filippini, D. Deep Learning for Head Pose Estimation: A Survey. *SN Comput. Sci.* **2023**, *4*, 349. [CrossRef]
52. Bradski, G. The OpenCV Library. *Dr. Dobb's J. Softw. Tools* **2000**.
53. Asperti, A.; Colasuonno, G.; Guerra, A. Portrait Reification with Generative Diffusion Models. *Appl. Sci.* **2023**, *13*, 6487. [CrossRef]
54. Asperti, A.; Tonelli, V. Comparing the latent space of generative models. *Neural Comput. Appl.* **2023**, *35*, 3155–3172. [CrossRef]
55. Guerra, A. Exploring Latent Embeddings in Diffusion Models for Face Orientation Conditioning. Master's Thesis, University of Bologna, Bologna, Italy, 2023.
56. Dai, B.; Wipf, D.P. Diagnosing and enhancing VAE models. In Proceedings of the Seventh International Conference on Learning Representations (ICLR 2019), New Orleans, LA, USA, 6–9 May 2019.
57. Zhou, S.; Chan, K.C.K.; Li, C.; Loy, C.C. Towards Robust Blind Face Restoration with Codebook Lookup Transformer. In Proceedings of the NeurIPS, New Orleans, LA, USA, 28 November–9 December 2022.
58. Zhang, L.; Zhang, L.; Mou, X.; Zhang, D. FSIM: A Feature Similarity Index for Image Quality Assessment. *IEEE Trans. Image Process.* **2011**, *20*, 2378–2386. [CrossRef] [PubMed]
59. Taigman, Y.; Yang, M.; Ranzato, M.; Wolf, L. DeepFace: Closing the Gap to Human-Level Performance in Face Verification. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, 23–28 June 2014; IEEE Computer Society: Piscataway, NJ, USA, 2014; pp. 1701–1708. [CrossRef]
60. Deng, J.; Guo, J.; Xue, N.; Zafeiriou, S. ArcFace: Additive Angular Margin Loss for Deep Face Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, 16–20 June 2019; pp. 4690–4699. [CrossRef]

61. Zhang, R.; Isola, P.; Efros, A.A.; Shechtman, E.; Wang, O. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, 18–22 June 2018; Computer Vision Foundation/IEEE Computer Society: Piscataway, NJ, USA, 2018; pp. 586–595. [CrossRef]
62. Ho, J.; Salimans, T.; Gritsenko, A.A.; Chan, W.; Norouzi, M.; Fleet, D.J. Video Diffusion Models. In Proceedings of the NeurIPS, New Orleans, LA, USA, 28 November–9 December 2022.
63. Ho, J.; Chan, W.; Saharia, C.; Whang, J.; Gao, R.; Gritsenko, A.A.; Kingma, D.P.; Poole, B.; Norouzi, M.; Fleet, D.J.; et al. Imagen Video: High Definition Video Generation with Diffusion Models. *arXiv* **2022**, arXiv:2210.02303.
64. Brooks, T.; Peebles, B.; Holmes, C.; DePue, W.; Guo, Y.; Jing, L.; Schnurr, D.; Taylor, J.; Luhman, T.; Luhman, E.; et al. Video Generation Models as World Simulators guides. *OpenAI* **2024**.

electronics

MDPI

Article

Novel Learning Framework with Generative AI X-Ray Images for Deep Neural Network-Based X-Ray Security Inspection of Prohibited Items Detection with You Only Look Once

Dongsik Kim and Jinho Kang *

School of Electronic Engineering, Gyeongsang National University, Jinju-si 52828, Republic of Korea; ehdtlr9724@gnu.ac.kr
* Correspondence: jinhokang@gnu.ac.kr

Abstract: As the rapid expansion of future mobility systems increases, along with the demand for fast and accurate X-ray security inspections, deep neural network (DNN)-based systems have gained significant attention for detecting prohibited items by constructing high-quality datasets and enhancing detection performance. While Generative AI has been widely explored across various fields, its application in DNN-based X-ray security inspection remains largely underexplored. The accessibility of commercial Generative AI raises safety concerns about the creation of new prohibited items, highlighting the need to integrate synthetic X-ray images into DNN training to improve detection performance, adapt to emerging threats, and investigate its impact on object detection. To address this, we propose a novel machine learning framework that enhances DNN-based X-ray security inspection by integrating real-world X-ray images with Generative AI images utilizing a commercial text-to-image model, improving dataset diversity and detection accuracy. Our proposed framework provides an effective solution to mitigate potential security threats posed by Generative AI, significantly improving the reliability of DNN-based X-ray security inspection systems, as verified through comprehensive evaluations.

Keywords: X-ray security inspection; prohibited items; machine learning; deep neural network; object detection; generative AI; copy-paste augmentation; novel framework; image generation; YOLO

1. Introduction

Security inspections are essential for ensuring passenger safety in public transportation, including airplanes, ships, and future mobility systems [1–3]. Currently, X-ray detection systems rely on manual inspection by security personnel to identify prohibited items. However, this task can cause significant fatigue and stress, potentially affecting the accuracy of prohibited item detection. Additionally, personnel-based systems are incompatible with future autonomous mobility systems. Recently, deep neural network (DNN)-based accurate and fast automatic systems used to detect prohibited items in X-ray security inspection have received great attention in recent years.

Specifically, object detection techniques within computer vision that can individually identify prohibited items on images have been widely studied for X-ray security inspection [4–8]. However, X-ray images, produced using high-energy radiation, often appear as overlapping objects due to their transmission properties, making detection in security inspections more challenging. These overlap issues can occur between objects and the back-

ground or among multiple objects [4]. In addition, it is known that the manual collection and annotation of X-ray images is labor-intensive and costly [9].

To resolve these challenges, constructing high-quality datasets and DNN models have been widely studied to improve detection performance in X-ray security inspection. Benchmark datasets have been extensively studied for X-ray security inspection in real-world scenarios, such as GDXray [10], SIXray [11], OPIXray [4], HIXray [5], CLCXray [6], PIDray [7], LSIray [12], DvXray [13], and LDXray [14]. These datasets have been used to improve the detection performance of X-ray prohibited items by addressing issues such as lack of data, positive data imbalances, overlap between objects, and image building from multiple directions. In addition, adding or improving additional modules in DNN models was studied to enhance the detection performance based on benchmark datasets [4–6,8,11–17]. In addition, the framework for data augmentation of X-ray security inspection images with generative adversarial networks (GANs) was proposed to improve detection performance [9,18].

On the other hand, with OpenAI's launch of ChatGPT in 2020, Generative AI (Gen AI) has received great worldwide attention in various fields. It autonomously generates new data such as text, images, video, etc., thereby automating tasks traditionally performed by humans [19,20]. With the availability of various commercial Generative AIs (Gen AIs) such as ChatGPT, Bard, DALL·E, and so on, an environment has been created where anyone can easily generate text or high-quality images simply by providing prompt input [19–24]. In this regard, research on the utility and impact of Gen AIs, including several concerns of their exploiting capabilities, has been widely studied in various applications [25,26]. Specifically, images created by Gen AI have become so sophisticated that distinguishing them from real images has become increasingly challenging [27]. Accordingly, the potential risks associated with Gen AI technologies, such as deepfakes, have become more prominent [26]. However, to the best of our knowledge, research on the utility and impact of Gen AIs in machine learning-based X-ray security inspection systems has not been studied, despite its tremendous utility and concerns.

In particular, the widespread availability of commercial Gen AIs, which can generate highly realistic images with just a few lines of input, raises concerns about the potential creation of new prohibited items, posing significant safety threats. Since the performance of machine learning-based X-ray security inspection systems relies heavily on the training dataset [28], there is a concern that systems trained solely on traditional benchmark datasets may fail to detect newly created prohibited items generated by commercial Gen AIs. This limitation could lead to serious security risks in public transportation and future mobility systems. Hence, to accurately detect prohibited items, including newly emerging ones, synthetic X-ray images generated by commercial Gen AIs should be incorporated into the training of DNNs for X-ray security inspection systems to investigate their impact on the object detection performance of the prohibited items.

In this paper, we propose a novel machine learning framework that integrates real-world X-ray images with Gen AI X-ray images to enhance dataset diversity and improve the detection of prohibited items in DNN-based X-ray security inspection systems. To support our framework, we establish a newly developed Gen AI X-ray image dataset using a commercial text-to-image model, DALL·E 3, providing a valuable resource for training and evaluating DNN models. To improve model robustness and generalization, we leverage copy-paste augmentation, which enhances object overlap, occlusions, and cluttered environments. We systematically evaluate the impact of Gen AI images on detection accuracy by training the model with varying numbers of real-world and synthetic Gen AI X-ray images, assessing its ability to generalize and identify potential limitations. Our evaluations demonstrate that YOLOv8, when trained solely on real-world X-ray images,

struggles to detect prohibited items in Gen AI X-ray images, highlighting the necessity of synthetic data for improved generalization. In contrast, the proposed framework, which incorporates copy-paste-augmented Gen AI X-ray images, significantly enhances detection performance without compromising real-world accuracy in terms of Precision, Recall, and mAP50-95.

2. Proposed Approach

2.1. Framework on Learning with Generative AI X-Ray Images

Our proposed framework is illustrated in Figure 1. We devised a novel approach that enables the automatic security inspection system to learn to detect well-known prohibited items from real-world X-ray image datasets, as well as prohibited items from synthetic X-ray images generated by commercial Generative AI (Gen AI). This approach addresses the limitations of traditional X-ray datasets by enhancing dataset diversity. By reflecting real-world environments and use cases, text-based prompts are used to generate synthetic X-ray images.

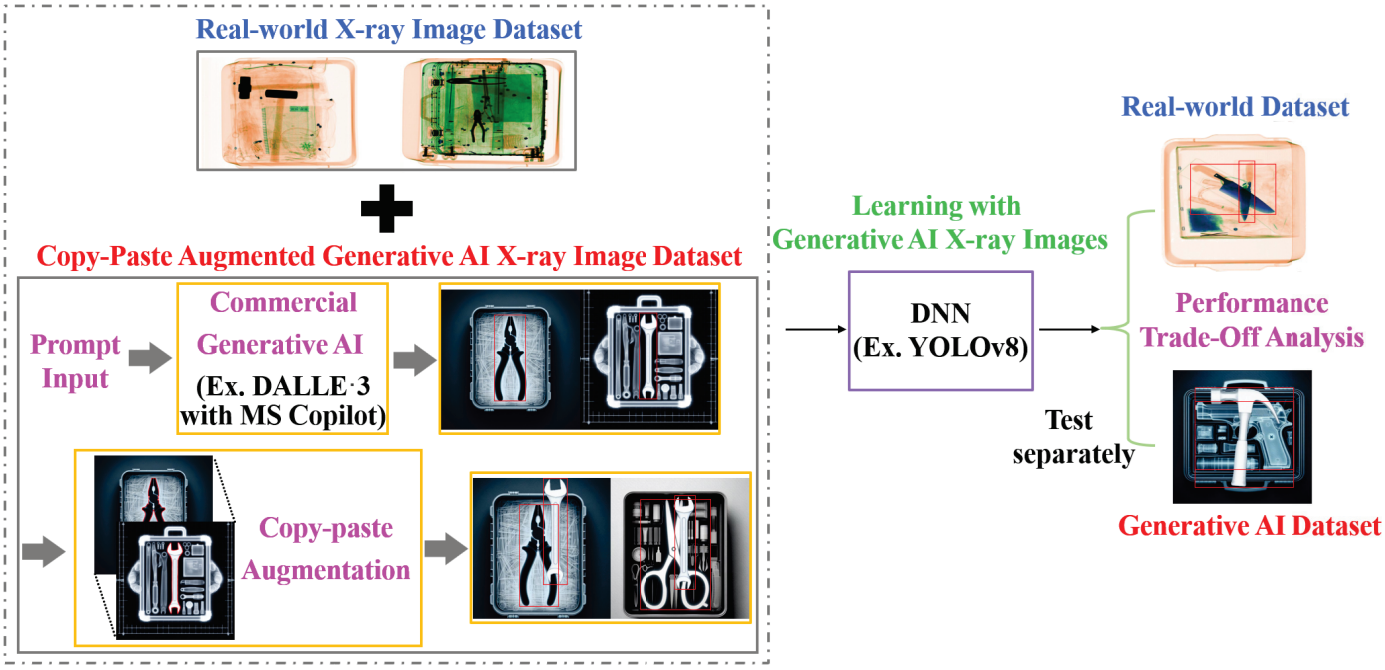


Figure 1. Illustration of our proposed framework.

In addition, to develop a more robust and accurate system while preventing the commercial Gen AI dataset from consisting of simplistic scenarios with a lack of object overlap, we leverage the copy-paste augmentation technique [29], in which individual objects extracted from Gen AI X-ray images are superimposed onto real X-ray detection systems. This augmentation technique exposes the DNN model to a wider range of object placements, occlusions, and cluttered environments, enhancing its ability to detect concealed or partially visible prohibited items. By integrating copy-paste-augmented Gen AI X-ray images into training, our framework enables the ability to improve the model's generalization ability, enhancing detection performance on Gen AI X-ray images while maintaining accuracy in real-world scenarios. The augmented dataset enables the model to learn a broader spectrum of object interactions, occlusions, and spatial complexities, ultimately improving detection accuracy across both domains. The process of generating copy-paste-augmented Gen AI X-ray images is described in the next subsection.

The DNN model is trained using an enriched hybrid dataset that integrates real-world X-ray images and Gen AI X-ray images, enabling the model to generalize across both domains [30]. In this paper, we employed YOLOv8 as our chosen DNN model [31]. YOLOv8 incorporates an efficient architecture that utilizes the self-attention mechanism and deformable convolution, reducing memory requirements while simultaneously providing high detection accuracy and fast processing speed [31,32]. Additionally, YOLOv8 supports built-in data augmentation functions such as scaling, flipping, and cropping, enhancing user convenience and contributing to its widespread adoption across various industries [12,33–35]. Although YOLOv10, the latest version of the YOLO model, has been recently released with advancements and optimizations [36], we chose to utilize YOLOv8 due to its deployment-friendly nature and suitability for industrial applications requiring fast detection.

To evaluate its effectiveness, we conducted a performance tradeoff analysis focusing on the following aspects:

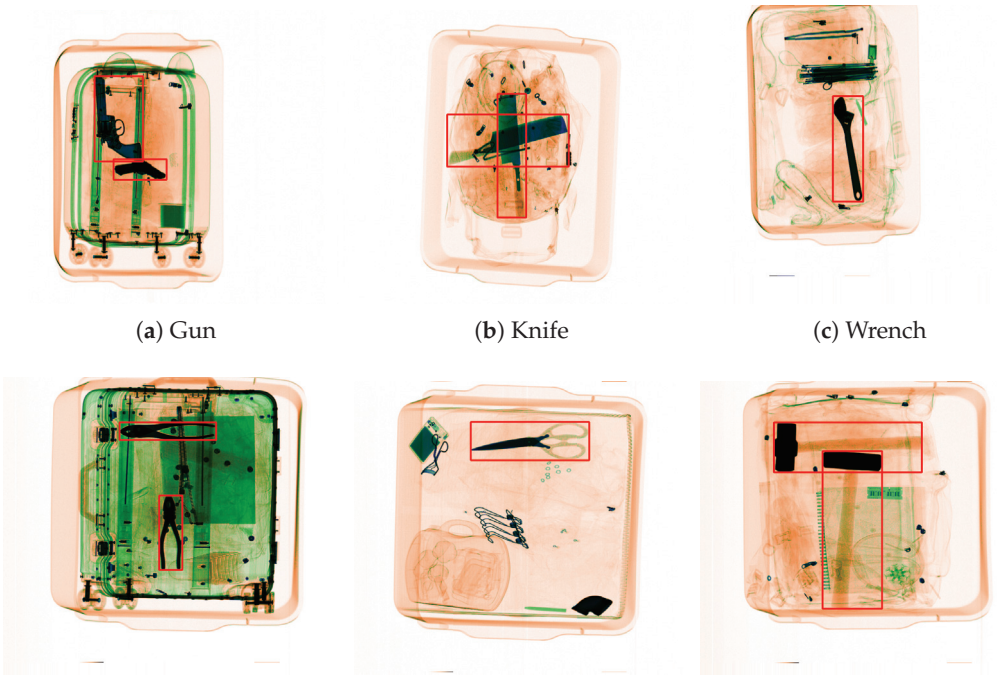
- The impact of Gen AI images on detection accuracy.
- The model's ability to generalize across real-world and synthetic Gen AI X-ray images.

To systematically analyze these tradeoffs, we trained the DNN model with varying numbers of real-world and Gen AI X-ray images in the training dataset. The trained models were then evaluated separately using distinct test datasets. This approach enabled us to determine whether synthetic data enhances model accuracy or introduces limitations, ultimately refining the effectiveness of Gen AI-enhanced datasets. This approach allowed us to determine whether synthetic data improves model accuracy or introduces limitations, ultimately enhancing the effectiveness of Gen AI-enhanced datasets.

2.2. Real-World X-Ray Image Dataset for Prohibited Items

In this paper, we utilized "X-ray multi-object detection data" as the real-world X-ray images of prohibited items, which were provided by AI Hub [37]. In this subsection, we briefly introduce this dataset. AI Hub, supported by MSIT (Ministry of Science and ICT) and NIA (National Information Society Agency) of South Korea, is a platform that provides the infrastructure needed for the development of AI technologies and services, along with various datasets applicable. The "X-ray multi-object detection data" broadly categorizes various types of items in X-ray images into three categories: (1) "Prohibited Items", (2) "Information Storage Devices", and (3) "General Items", where the dataset consists of a total of 541,260 images across 317 classes. The details on the selected dataset that we utilized in the training and evaluation for the prohibited items are describied in [37].

Figure 2 shows examples of real-world X-ray images that we utilized. The original dataset of prohibited items was classified into six categories (Gun, Knife, Wrench, Pliers, Scissors, and Hammer) based on the SIXRAY dataset [11] for practical X-ray security inspection. Various subcategories were merged, and unrelated or mislabeled data were excluded. The final dataset consisted of 51,210 real-world X-ray images, which were divided into Train, Validation, and Test sets.



(**a**) Gun (**b**) Knife (**c**) Wrench

(**d**) Pliers (**e**) Scissors (**f**) Hammer

Figure 2. Examples of real-world X-ray images for prohibited items where the red boxes indicate the prohibited items that have been labeled.

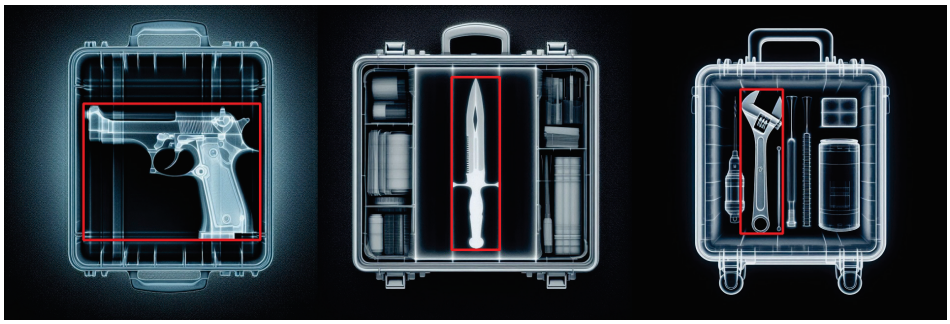
2.3. Copy-Paste-Augmented Generative AI X-Ray Image Dataset for Prohibited Items

In order to leverage the Gen AI X-ray image dataset for prohibited items, we produced X-ray images for the prohibited items created by the commercial Gen AI. Among the several text-to-image Gen AI models, we adopted the DALL·E 3 [38], which is developed by Open AI, thanks to its accessibility and availablity in ChatGPT4 [20] as well as Microsoft Copilot [39]. To create the images, we input the text prompt into the uncustomized original DALL·E 3 through Microsoft Copilot (we leveraged Microsoft Copilot due to its utility, where several images corresponding to the prompt are generated at once), without any modifications or additional fine-tuning, as follows:

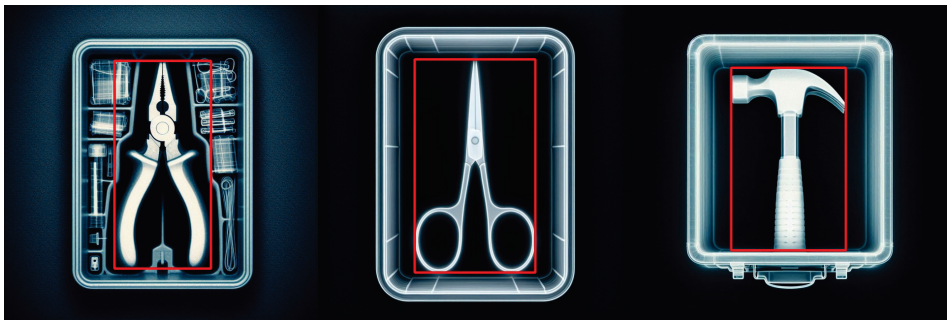
> "X-ray image of a box containing a *Prohibited Item*. This image is in the style typically seen by airport security personnel."

In the above prompt, a *Prohibited Item* means one of Gun, Knife, Wrench, Pliers, Scissors, or Hammer so that it would generate the images with the same classes as the real-world X-ray images. Then, we manually labeled each generated image by utilizing the "LabelImg" program, which supports labeling with YOLO and Pascal VOC formats [40], since the generated images lack information on object location, quantity, and type.

Figure 3 shows examples of Gen AI X-ray images for prohibited items created from DALL·E 3. Most images contain one prohibited item, but some images also contain several prohibited items. In addition, due to creative generation on some images, we excluded the generated images that were unclear to identify objects. To match the default input image size of YOLOv8, the size of each Gen AI image was adjusted 640×640 from the initial size of 1024×1024.



(**a**) Gun (**b**) Knife (**c**) Wrench



(**d**) Pliers (**e**) Scissors (**f**) Hammer

Figure 3. Examples of Gen AI X-ray images for prohibited items that we created, where the red boxes indicate the prohibited items that have been labeled.

Next, we produced augmented images based on the created Gen AI X-ray images using a separate copy-paste Python (version 3.12.2) script [41] instead of using YOLO's built-in augmentation features. This study focuses on images generated by commercial Gen AI. To address the limitations of existing Gen AI X-ray images, which often feature simple compositions and lack object overlap, we applied the copy-paste augmentation technique while preserving other image properties such as color and size. This approach enhances individual images by incorporating additional object information and introducing object occlusion, thereby creating more realistic training data.

Figure 4 illustrates examples of the copy-paste-augmented Gen AI X-ray images for prohibited items that we created. For copy-paste augmentation with instance segmentation labels, we exploited a commonly used labeling tool that supports labeling tasks for various image segmentation and augmentation techniques. In this study, augmentation was primarily performed using the copy-paste technique. However, exploring additional performance improvements by integrating it with various augmentation techniques remains as an on-going topic of research.

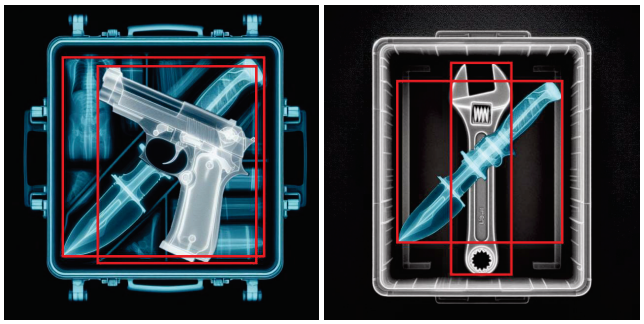


Figure 4. Examples of copy-paste-augmented Gen AI images for prohibited items that we created, where the red boxes indicate the prohibited items that have been labeled.

3. Results and Discussion

In this section, our goal is to evaluate and analyze the performance of our proposed framework by training YOLOv8 while varying the number of real-world X-ray training images and copy-paste-augmented Gen AI X-ray images.

3.1. Performance Metrics and Setup

To evaluate our proposed framework, we considered the performance metrics that are widely adopted in object detection based on the confusion matrix of Table 1 as follows [42]:

Table 1. Confusion matrix.

		Predicted Class	
		Positive	**Negative**
Actuall class	Positive	TP	FN
	Negative	FP	TF

- Precision: Proportion of predicted positive classes that are actually positive, which is defined as $\frac{TP}{TP+FP}$.
- Recall: Proportion of actual positive instances that the model correctly identifies as positive, which is defined as $\frac{TP}{TP+FN}$.
- mAP50-95: A metric that averages the mean Average Precision (mAP) calculated at Intersection over Union (IoU) thresholds ranging from 0.5 to 0.95 in increments of 0.05. This metric includes higher IoU criteria compared to mAP50, thus requiring more accurate bounding boxes.

The environment that we adopted for evaluation is described in Table 2. We utilized YOLOv8 for the DNN model thanks to its lightweight, fast, and high-performance characteristics [31]. The pretrained weights from the COCO dataset were applied to our YOLOv8 model to improve its generalization ability in object detection, which has been widely adopted for benchmarking object detection models [43].

Table 2. Environment for evaluation.

Component	**Specification**
OS	Windows 11
CPU	Intel i9 139006 (Santa Clara, CA, USA)
RAM	64 GB
Graphics	NVIDIA RTX 4080 SUPER (Santa Clara, CA, USA)
VRAM	16 GB
DNN Model	YOLOv8.2.42

The hyperparameters considered for model training are described as follows. The model was trained for 20 epochs using the AdamW optimizer with an initial learning rate of 0.01, a momentum of 0.937, a weight decay of 0.0005, and a batch size of 16. In addition, the input image size was set to 640 $\times$ 640. We considered the default hyperparameter values for YOLOv8 to maintain a consistent experimental environment without the influence of tuning on performance differences. Although it is known that enhancing the YOLOv8 structure and tuning its hyperparameters can improve performance, this study focused on performance analysis based on the inclusion of Gen AI images. Therefore, the default configuration was retained to enable a clearer comparison. Extending the proposed framework to incorporate

an improved YOLOv8 by optimizing its structure and hyperparameter configurations remains an ongoing area of research.

To evaluate our proposed framework through performance tradeoff analysis, we trained six separate YOLOv8 models, each with varying numbers of real-world X-ray images and Gen AI X-ray images. Table 3 shows the total number of images for each set, where non-copy-paste-augmented Gen AI (NC-Gen AI) X-ray images mean Gen AI images without copy-paste augmentation technique such as the images in Figure 3. For performance comparison based on the copy-paste augmentation technique, Models 1–3 were trained using NC-Gen AI X-ray images, while Models 4–6 were trained using copy-paste-augmented Gen AI (C-Gen AI) X-ray images such as the images in Figure 4.

Table 3. Total number of X-ray images.

Number of Images	Train	Valid	Test
Real-World	35,850	7680	7680
Non-Copy-Paste Augmented Gen AI	300	60	60
Copy-Paste Augmented Gen AI	300	60	60

More specifically, Table 4 provides the number of training images for Models 1–3 and Models 4–6, respectively, where $\alpha \in [0, 300]$ represents the number of Gen AI X-ray images used in the training set. For example, Model 1 was trained with 1000 real-world X-ray images and α NC-Gen AI X-ray images, while Model 4 was trained with 1000 real-world X-ray images and α C-Gen AI X-ray images. During the evaluations, α increased from 0 to 300 so that each model was trained with an increasing number of Gen AI X-ray images. In this case, a uniform distribution was used to sample the number of images corresponding to the value of α among all training datasets containing 300 Gen AI X-ray images. Also, for Models 1 and 4 and Models 2 and 5, 1000 and 10,000 images, respectively, were also randomly sampled from a real-world training dataset containing 35,850 images.

Table 4. The number of training images for Models 1–6.

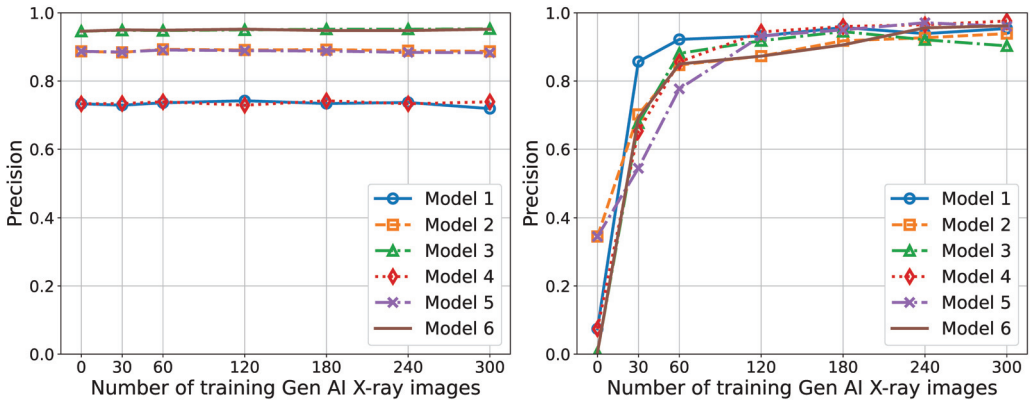
YOLOv8	Real-World	NC-Gen AI/C-Gen AI
Model 1/4	1000	α/α
Model 2/5	10,000	α/α
Model 3/6	35,850 (All)	α/α

For a performance tradeoff analysis, each trained model was separately evaluated on a real-world X-ray image test set, an NC-Gen AI X-ray image test set, and a C-Gen AI image test set. This approach aimed to assess the model's existing detection capability while determining its effectiveness in identifying new threat elements in complex environments. Consequently, the impact of the Gen AI dataset on model performance was quantitatively analyzed. Each performance metric was calculated as the average of five experiments conducted with different sampled images for a fixed number of images, i.e., 1000 or 10,000 sampled real-world images and α sampled Gen AI images.

3.2. Evaluation and Performance Analysis

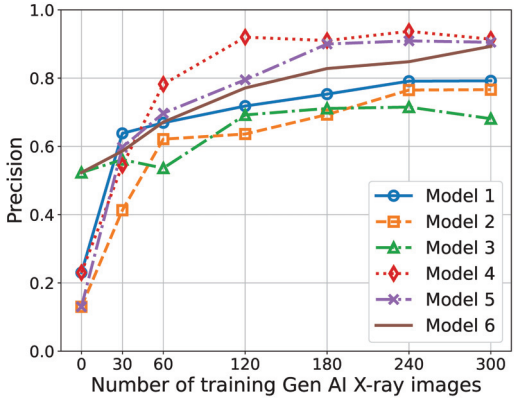
In this subsection, Figures 5–7 illustrate the performance of each model in terms of Precision, Recall, and mAP50-95 with respect to the number of training Gen AI X-ray images, i.e., α, evaluated respectively by (a) a real-world X-ray image test set, (b) an NC-

Gen AI X-ray image test set, and (c) a C-Gen AI X-ray X-ray image test set. In addition, the detailed values associated with each figure are provided in Tables 5–7, respectively.



(**a**) Real-world X-ray image test set

(**b**) NC-Gen AI X-ray image test set



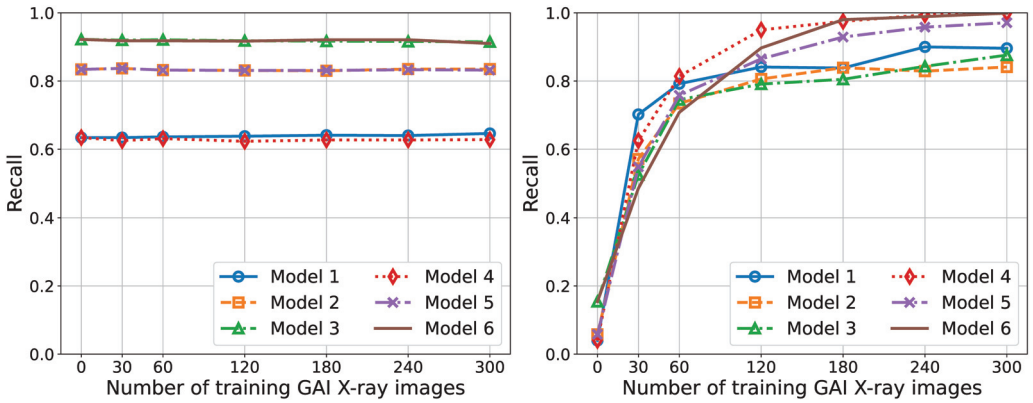
(**c**) C-Gen AI X-ray image test set

Figure 5. Precision with respect to the number of training Gen AI X-ray images evaluated by (**a**) real-world X-ray image test set, (**b**) NC-Gen AI X-ray image test set, (**c**) C-Gen AI X-ray image test set.

Table 5. Precision with respect to the number of training Gen AI X-ray images evaluated by real-world X-ray image test set, NC-Gen AI X-ray image test set, and C-Gen AI X-ray image test set.

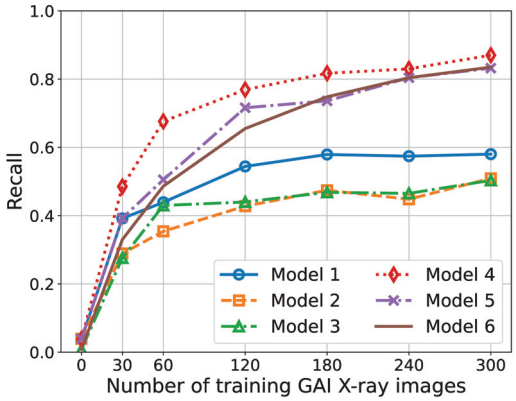
Number of Training Gen AI X-Ray Images		**0**	**30**	**60**	**120**	**180**	**240**	**300**
Real-world X-ray image test	Model 1	0.733	0.729	0.736	0.742	0.734	0.737	0.719
	Model 2	0.887	0.884	0.893	0.891	0.892	0.889	0.887
	Model 3	0.946	0.949	0.948	0.950	0.952	0.952	0.953
	Model 4	0.733	0.734	0.739	0.729	0.742	0.733	0.739
	Model 5	0.887	0.885	0.891	0.889	0.888	0.884	0.883
	Model 6	0.946	0.950	0.949	0.952	0.948	0.948	0.952
NC-Gen AI X-ray image test	Model 1	0.074	0.857	0.922	0.932	0.957	0.939	0.954
	Model 2	0.345	0.702	0.847	0.873	0.918	0.927	0.939
	Model 3	0.004	0.677	0.881	0.918	0.945	0.921	0.903
	Model 4	0.074	0.653	0.856	0.944	0.960	0.965	0.976
	Model 5	0.345	0.544	0.777	0.932	0.950	0.971	0.960
	Model 6	0.004	0.692	0.850	0.873	0.906	0.956	0.962
C-Gen AI X-ray image test	Model 1	0.230	0.638	0.669	0.718	0.753	0.791	0.792
	Model 2	0.130	0.413	0.621	0.636	0.693	0.765	0.766
	Model 3	0.523	0.561	0.536	0.692	0.711	0.715	0.681
	Model 4	0.230	0.544	0.782	0.920	0.910	0.937	0.914
	Model 5	0.130	0.597	0.697	0.795	0.900	0.909	0.905
	Model 6	0.523	0.586	0.671	0.771	0.828	0.848	0.893

First, Figure 5a–c represent the Precision performance, and the corresponding detailed values are provided in Table 5. In Figure 5a, the Precision remained consistently high for all models, regardless of α. Specifically, the copy-paste augmentation or not did not affect the Precision performance for real-world images. As expected, Models 3 and 6 exhibited the highest performance, while Models 1 and 4 showed the lowest performance. Figure 5b shows that the Precision considerably increased as α increased for all models. The models trained without the Gen AI X-ray images (i.e., $\alpha = 0$) yielded Precision values smaller than 0.4, while the models trained with 120 images achieved Precision values higher than 0.8. Models 4–6 slightly outperformed Models 1–3 and achieved near-perfect Precision values with 300 C-Gen AI images. Figure 5c demonstrates the Precision performance evaluated for the C-Gen AI test set. Similarly, increasing α led to improved performance. On the other hand, Models 4–6 considerably outperformed Models 1–3 for all Gen AI image levels and achieved a 0.9 Precision value with 300 C-Gen AI training images.



(**a**) Real-world X-ray image test set

(**b**) NC-Gen AI X-ray image test set



(**c**) C-Gen AI X-ray image test set

Figure 6. Recall with respect to the number of training Gen AI X-ray images evaluated by (**a**) real-world X-ray image test set, (**b**) NC-Gen AI X-ray image test set, (**c**) C-Gen AI X-ray image test set.

Next, Figure 6a–c show the Recall performance, and the corresponding detailed values are provided in Table 6. In Figure 6a, the Recall remained consistently high for all models regardless of α, and the copy-paste augmentation or not had little to no effect on the Recall performance for real-world images, where Models 3 and 6 exhibited the highest performance. Figure 6b,c exhibit that the Recall considerably increased as α increased on both the NC-Gen AI and C-Gen AI test sets. In particular, the models trained without the Gen AI X-ray images (i.e., $\alpha = 0$) yielded Recall values smaller than 0.2. Moreover, Models 4–6 outperformed Models 1–3 on both test sets after 120 images, with a larger performance gap observed in the C-Gen AI test set. Models 4–6 achieved near-perfect Recall values on

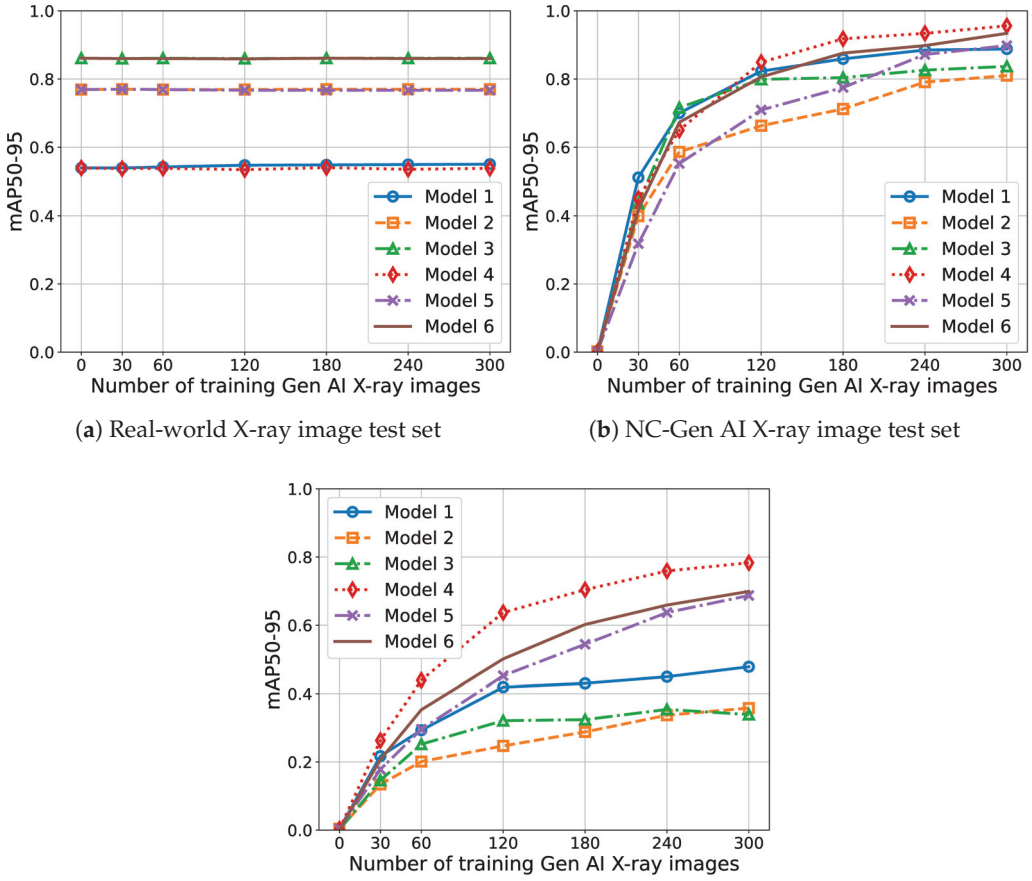
the NC-Gen AI test set and approximately 0.85 in their Recall values on the C-Gen AI test set when trained with 300 C-Gen AI images. In addition, Model 4 demonstrated the highest performance on the C-Gen AI test set.

Table 6. Recall with respect to the number of training Gen AI X-ray images evaluated by real-world X-ray image test set, NC-Gen AI X-ray image test set, and C-Gen AI X-ray image test set.

Number of Training Gen AI X-Ray Images		0	30	60	120	180	240	300
Real-world X-ray image test	Model 1	0.634	0.634	0.636	0.638	0.641	0.640	0.646
	Model 2	0.834	0.837	0.832	0.831	0.830	0.835	0.835
	Model 3	0.922	0.920	0.921	0.918	0.917	0.916	0.915
	Model 4	0.634	0.625	0.631	0.623	0.627	0.627	0.628
	Model 5	0.834	0.837	0.832	0.831	0.831	0.833	0.832
	Model 6	0.922	0.918	0.918	0.917	0.921	0.921	0.910
NC-Gen AI X-ray image test	Model 1	0.040	0.702	0.792	0.841	0.838	0.900	0.896
	Model 2	0.058	0.571	0.734	0.806	0.839	0.829	0.841
	Model 3	0.154	0.525	0.745	0.791	0.805	0.843	0.876
	Model 4	0.040	0.625	0.814	0.950	0.975	0.994	0.999
	Model 5	0.058	0.548	0.758	0.865	0.929	0.958	0.971
	Model 6	0.154	0.485	0.708	0.897	0.980	0.989	0.999
C-Gen AI X-ray image test	Model 1	0.038	0.392	0.439	0.544	0.579	0.574	0.580
	Model 2	0.039	0.289	0.354	0.428	0.474	0.448	0.509
	Model 3	0.008	0.276	0.430	0.440	0.468	0.465	0.503
	Model 4	0.038	0.485	0.676	0.769	0.817	0.830	0.870
	Model 5	0.039	0.393	0.505	0.716	0.736	0.804	0.832
	Model 6	0.008	0.330	0.486	0.655	0.748	0.804	0.835

Table 7. mAP50-95 with respect to the number of training Gen AI X-ray images evaluated by real-world X-ray image test set, NC-Gen AI X-ray image test set, and C-Gen AI X-ray image test set.

Number of Training Gen AI X-Ray Images		0	30	60	120	180	240	300
Real-world X-ray image test	Model 1	0.539	0.539	0.542	0.547	0.548	0.549	0.550
	Model 2	0.769	0.770	0.769	0.769	0.770	0.770	0.770
	Model 3	0.861	0.860	0.861	0.860	0.861	0.861	0.861
	Model 4	0.539	0.537	0.538	0.534	0.540	0.535	0.538
	Model 5	0.769	0.770	0.769	0.767	0.767	0.767	0.767
	Model 6	0.861	0.860	0.860	0.859	0.861	0.860	0.860
NC-Gen AI X-ray image test	Model 1	0.001	0.511	0.700	0.823	0.859	0.885	0.888
	Model 2	0.002	0.399	0.586	0.663	0.712	0.791	0.810
	Model 3	0.001	0.436	0.716	0.799	0.804	0.826	0.837
	Model 4	0.001	0.449	0.650	0.849	0.918	0.934	0.956
	Model 5	0.002	0.318	0.552	0.709	0.775	0.872	0.898
	Model 6	0.001	0.417	0.673	0.806	0.876	0.898	0.934
C-Gen AI X-ray image test	Model 1	0.003	0.217	0.293	0.419	0.430	0.449	0.478
	Model 2	0.004	0.134	0.201	0.247	0.288	0.337	0.358
	Model 3	0.001	0.146	0.252	0.321	0.324	0.354	0.339
	Model 4	0.003	0.263	0.439	0.637	0.704	0.759	0.783
	Model 5	0.004	0.178	0.296	0.452	0.544	0.637	0.687
	Model 6	0.001	0.207	0.353	0.501	0.602	0.659	0.699



(**a**) Real-world X-ray image test set

(**b**) NC-Gen AI X-ray image test set

(**c**) C-Gen AI X-ray image test set

Figure 7. mAP50-95 with respect to the number of training Gen AI X-ray images evaluated by (**a**) real-world X-ray image test set, (**b**) NC-Gen AI X-ray image test set, (**c**) C-Gen AI X-ray image test set.

Figure 7a–c show the mAP50-95 performance, and the corresponding detailed values are provided in Table 7. Figure 7a demonstrates the similar performance to the Precision and Recall results with respect to α. Also, Figure 7b,c demonstrate that the mAP50-95 significantly increased as α increased on both the NC-Gen AI and C-Gen AI test sets. Models 4–6 outperformed Models 1–3 on both test sets, with a larger performance gap observed in the C-Gen AI test set. Specifically, Models 4 and 6 achieved 0.9 in their mAP50-95 values on both NC-Gen AI test sets with 300 C-Gen AI training images. On the other hand, Model 4 demonstrated the highest mAP50-95 performance on the C-Gen AI test set and achieved near 0.8 in its mAP50-95 value with 300 C-Gen AI training images, while Model 6 yielded near 0.7 value.

3.3. Discussion

In this subsection, we analyze and discuss the performance of each model individually from the experimental results. First, Model 1 consistently exhibited the lowest performance in the real-world X-ray image test set, regardless of the increase in Gen AI X-ray images. However, in the NC-Gen AI X-ray image test set, its performance improved as the number of Gen AI images increased, exceeding the average. In the C-Gen AI X-ray image test set, Model 1 achieved the highest performance among Models 1–3 as the number of Gen AI images increased, but the performance gap between Model 1 and Models 4–6 widened progressively.

Model 2 maintained stable performance in the real-world X-ray image test set, even as the number of Gen AI X-ray images increased. In the NC-Gen AI X-ray test set, it demonstrated excellent Precision as the number of Gen AI images increased, but the Recall and mAP50-95 values remained relatively low. In the C-Gen AI X-ray image test set, Model 2 exhibited a significant performance gap compared to Models 4–6. Among Models 1–3, it showed strong performance in terms of Precision, but the Recall and mAP50-95 values remained low.

Model 3 achieved the highest performance in the real-world X-ray image test set, regardless of the increase in Gen AI X-ray images. In the NC-Gen AI X-ray test set, the Precision was strong, but the Recall and mAP50-95 were relatively poor. In the C-Gen AI X-ray image test set, Model 3 recorded the lowest performance overall.

Model 4 exhibited a performance trend similar to Model 1 in the real-world X-ray image test set. However, in both the NC-Gen AI and C-Gen AI X-ray image test sets, Model 4 achieved the highest performance as the number of Gen AI images increased.

Model 5 showed performance nearly identical to Model 2 in the real-world X-ray image test set. It performed well in the NC-Gen AI X-ray image test set and exhibited high performance across all metrics in the C-Gen AI X-ray image test set. However, its mAP50-95 score was approximately 0.1 lower than that of Model 4.

Finally, Model 6 demonstrated performance similar to Model 3, achieving the highest scores in the real-world X-ray image test set. It also performed well in the NC-Gen AI X-ray test set. In the C-Gen AI X-ray image test set, it achieved high performance, but its mAP50-95 score was approximately 0.1 lower than that of Model 4, similar to Model 5.

The results conclude that YOLOv8, when trained solely on real-world X-ray images, exhibits limited detection ability on Gen AI images. However, incorporating Gen AI images into training enhances performance on synthetic data while maintaining real-world accuracy. Additionally, copy-paste augmentation further improves the overall detection capabilities.

4. Conclusions

This paper studied a novel machine learning framework for detecting prohibited items in DNN-based X-ray security inspection systems. As Generative AI technology advances, allowing anyone to create desired images using natural language without requiring specialized AI knowledge, security systems trained solely on real-world images may fail to detect newly generated prohibited items. This limitation could pose a security vulnerability and potential risk. To address this, a new Gen AI X-ray image dataset was created using a commercial text-to-image Gen AI model and was used to train and evaluate YOLOv8. Experimental results indicate that when the YOLOv8 model was trained solely on real-world X-ray images, it failed to detect prohibited items in Gen AI X-ray images, suggesting the necessity of supplementary training using Generative AI data. In contrast, models incorporating copy-paste Gen AI X-ray images significantly improved detection performance without compromising real-world detection accuracy. These findings suggest that leveraging commercial Gen AI can enhance the accuracy of DNN-based X-ray security inspection systems. Furthermore, incorporating just a small number of Gen AI images into the training set can provide a powerful solution to mitigate the security risks associated with Gen AI.

To verify whether the generated X-ray images truly reflect real-world X-ray images, our ongoing research focuses on implementing Human Expert Verification to enhance the quality and reliability of the generated AI X-ray image dataset [44–46]. This approach will strengthen our proposed framework, making it more robust and accurate. Moreover, our findings demonstrate that training with Generative AI data can enhance performance. However, they often introduce bias, generate unrealistic objects, and may lead to model

collapse, creating a tradeoff [44–46]. To address these challenges, future research will focus on developing strategies to maximize performance while minimizing drawbacks. This includes constructing additional Generative AI datasets and adjusting the ratio of real to generated data. Furthermore, expert validation of the generated images will be conducted to ensure data quality and reliability, as well as to explore strategies for mitigating potential model bias.

Author Contributions: Conceptualization, D.K. and J.K.; Methodology, D.K. and J.K.; Software, D.K.; Validation, D.K. and J.K.; Formal analysis, D.K. and J.K.; Investigation, D.K. and J.K.; Resources, J.K.; Writing—original draft preparation, D.K.; writing—review and editing, J.K. Visualization, D.K. and J.K.; Supervision J.K.; Project administration, J.K.; Funding acquisition, J.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the NRF(National Research Foundation of Korea) grant funded by the Korea government (Ministry of Science and ICT) (RS-2023-00214142), and in part by the IITP(Institute of Information & Coummunications Technology Planning & Evaluation)-ICAN(ICT Challenge and Advanced Network of HRD) grant funded by the Korea government (Ministry of Science and ICT) (IITP-2025-RS-2022-00156409).

Data Availability Statement: The original contributions presented in the study are included in the article; further inquiries can be directed to the corresponding author. For the real-world X-ray image dataset, this paper used datasets from 'The Open AI Dataset Project (AI Hub, S. Korea)', where all data information can be accessed through 'AI Hub (https://www.aihub.or.kr, accessed on 31 January 2024)'.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Akçay, S.; Kundegorski, M.E.; Devereux, M.; Breckon, T.P. Transfer learning using convolutional neural networks for object classification within X-ray baggage security imagery. In Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 25–28 September 2016; pp. 1057–1061.
2. Saavedra, D.; Banerjee, S.; Mery, D. Detection of threat objects in baggage inspection with X-ray images using deep learning. *Neural Comput. Appl.* **2021**, *33*, 7803–7819.
3. Lee, J.N.; Cho, H.C. Development of artificial intelligence system for dangerous object recognition in X-ray baggage images. *Trans. Korean Inst. Electr. Eng.* **2020**, *69*, 1067–1072.
4. Wei, Y.; Tao, R.; Wu, Z.; Ma, Y.; Zhang, L.; Liu, X. Occluded prohibited items detection: An X-ray security inspection benchmark and de-occlusion attention module. In Proceedings of the 28th ACM International Conference on Multimedia, Seattle, WA, USA, 12–16 October 2020; pp. 138–146.
5. Tao, R.; Wei, Y.; Jiang, X.; Li, H.; Qin, H.; Wang, J.; Ma, Y.; Zhang, L.; Liu, X. Towards real-world X-ray security inspection: A high-quality benchmark and lateral inhibition module for prohibited items detection. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 10–17 October 2021; pp. 10923–10932.
6. Zhao, C.; Zhu, L.; Dou, S.; Deng, W.; Wang, L. Detecting overlapped objects in X-ray security imagery by a label-aware mechanism. *IEEE Trans. Inf. Forensics Secur.* **2022**, *17*, 998–1009.
7. Zhang, L.; Jiang, L.; Ji, R.; Fan, H. Pidray: A large-scale X-ray benchmark for real-world prohibited item detection *Int. J. Comput. Vis.* **2023**, *131*, 3170–3192.
8. Kim, E.; Lee, J.; Jo, H.; Na, K.; Moon, E.; Gweon, G.; Yoo, B.; Kyung, Y. SHOMY: Detection of Small Hazardous Objects using the You Only Look Once Algorithm. *KSII Trans. Internet Inf. Syst. (TIIS)* **2022**, *16*, 2688–2703.
9. Liu, J.; Lin, T.H. A framework for the synthesis of X-ray security inspection images based on generative adversarial networks. *IEEE Access* **2023**, *11*, 63751–63760.
10. Mery, D.; Riffo, V.; Zscherpel, U.; Mondragón, G.; Lillo, I.; Zuccar, I.; Lobel, H.; Carrasco, M. GDXray: The database of X-ray images for nondestructive testing. *J. Nondestruct. Eval.* **2015**, *34*, 42.
11. Miao, C.; Xie, L.; Wan, F.; Su, C.; Liu, H.; Jiao, J.; Ye, Q. Sixray: A large-scale security inspection X-ray benchmark for prohibited item discovery in overlapping images. In Proceedings of the 2021 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 2119–2128.

12. Han, L.; Ma, C.; Liu, Y.; Jia, J.; Sun, J. SC-YOLOv8: A security check model for the inspection of prohibited items in X-ray images. *Electronics* **2023**, *12*, 4208. [CrossRef]
13. Ma, B.; Jia, T.; Li, M.; Wu, S.; Wang, H.; Chen, D. Towards dual-view X-ray baggage inspection: A large-scale benchmark and adaptive hierarchical cross refinement for prohibited item discovery *IEEE Trans. Inf. Forensics Secur.* **2024**, *19*, 3866–3878.
14. Tao, R.; Wang, H.; Guo, Y.; Chen, H.; Zhang, L.; Liu, X.; Wei, Y.; Zhao, Y. Dual-view X-ray detection: Can AI detect prohibited items from dual-view X-ray images like humans? *arXiv* **2024**, arXiv:2411.18082.
15. Li, Y.; Zhang, C.; Sun, S.; Yang, G. X-ray detection of prohibited item method based on dual attention mechanism. *Electronics* **2023**, *12*, 3934. [CrossRef]
16. Jing, B.; Duan, P.; Chen, L.; Du, Y. EM-YOLO: An X-ray prohibited-item-detection method based on edge and material information fusion. *Sensors* **2023**, *23*, 8555. [CrossRef] [PubMed]
17. Zhang, H.; Zhao, Z.; Yang, J. Attention-based prohibited item detection in X-ray images during security checking. *IET Image Process.* **2024**, *18*, 1119–1131. [CrossRef]
18. Zhu, Y.; Zhang, Y.; Zhang, H.; Yang, J.; Zhao, Z. Data augmentation of X-ray images in baggage inspection based on generative adversarial networks. *IEEE Access* **2020**, *8*, 86536–86544.
19. OpenAI. ChatGPT. 2024. Available online: https://chat.openai.com (accessed on 4 April 2024).
20. Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F.L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. GPT-4 technical report. *arXiv* **2023**, arXiv:2303.08774.
21. Anil, R.; Dai, A.M.; Firat, O.; Johnson, M.; Lepikhin, D.; Passos, A.; Shakeri, S.; Taropa, E.; Bailey, P.; Chen, Z.; et al. Palm 2 technical report. *arXiv* **2023**, arXiv:2305.10403.
22. Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Roziere, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. Llama: Open and efficient foundation language models. *arXiv* **2023**, arXiv:2302.13971.
23. Ramesh, A.; Pavlov, M.; Goh, G.; Gray, S.; Voss, C.; Radford, A.; Chen, M.; Sutskever, I. Zero-shot text-to-image generation. In Proceedings of the 38th International Conference on Machine Learning, Virtual Event, 18–24 July 2021; pp. 8821–8831.
24. Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; Ommer, B. High-resolution image synthesis with latent diffusion models. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 10684–10695.
25. Andreoni, M.; Lunardi, W.T.; Lawton, G.; Thakkar, S. Enhancing autonomous system security and resilience with generative AI: A comprehensive survey. *IEEE Access* **2024**, *12*, 109470–109493. [CrossRef]
26. Golda, A.; Mekonen, K.; Pandey, A.; Singh, A.; Hassija, V.; Chamola, V.; Sikdar, B. Privacy and security concerns in generative AI: A comprehensive survey. *IEEE Access* **2024**, *12*, 48126–48144. [CrossRef]
27. Bird, J.J.; Lotfi, A. CIFAKE: Image Classification and Explainable Identification of AI-Generated Synthetic Images. *IEEE Access* **2024**, *12*, 15642–15650.
28. Lee, Y.; Kang, J. Performance Analysis by the Number of Learning Images on Anti-Drone Object Detection System with YOLO. *J. Korean Inst. Commun. Inf. Sci.* **2024**, *49*, 356–360
29. Ghiasi, G.; Cui, Y.; Srinivas, A.; Qian, R.; Lin, T.-Y.; Cubuk, E.D.; Le, Q.V.; Zoph, B. Simple copy-paste is a strong data augmentation method for instance segmentation. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 2918–2928.
30. Terven, J.; Córdova-Esparza, D.M.; Romero-González, J.A. A comprehensive review of YOLO architectures in computer vision: From YOLOv1 to YOLOv8 and YOLO-NAS. *Mach. Learn. Knowl. Extr.* **2023**, *5*, 1680–1716. [CrossRef]
31. Jocher, G.; Chaurasia, A.; Qiu, J. Ultralytics YOLO. 2024. Available online: https://github.com/ultralytics/ultralytics (accessed on 31 January 2024).
32. Sohan, M.; SaiRam, T.; RamiReddy, V.C. A review on yolov8 and its advancements. In Proceedings of the International Conference on Data Intelligence and Cognitive Informatics, Tirunelveli, India, 18–20 November 2024; pp 529–45
33. Wang, A.; Yuan, P.; Wu, H.; Iwahori, Y.; Liu, Y. Improved YOLOv8 for Dangerous Goods Detection in X-ray Security Images. *Electronics* **2024**, *13*, 3238. [CrossRef]
34. Fan, J.; Haji Salam, M.S.B.; Rong, X.; Han, Y.; Yang, J.; Zhang, J. Peach Fruit Thinning Image Detection Based on Improved YOLOv8 and Data Enhancement Techniques. *IEEE Access* **2024**, *12*, 191199–191218. [CrossRef]
35. Zhang, L.; Wu, X.; Liu, Z.; Yu, P.; Yang, M. ESD-YOLOv8: An Efficient Solar Cell Fault Detection Model Based on YOLOv8. *IEEE Access* **2024**, *12*, 138801–138815.
36. Mao, M.; Lee, A.; Hong, M. Efficient Fabric Classification and Object Detection Using YOLOv10. *Electronics* **2024**, *13*, 3840. [CrossRef]
37. AI Hub. Available online: https://aihub.or.kr (accessed on 31 January 2024).
38. OpenAI. DALL · E 3 System Card. 2023. Available online: https://openai.com/research/dall-e-3-system-card (accessed on 4 April 2024).
39. Microsoft Copilot. Copilot. 2023. Available online: https://copilot.microsoft.com/ (accessed on 4 April 2024).

40. Tzutalin. LabelImg. 2015. Available online: https://github.com/HumanSignal/labelImg (accessed on 4 April 2024).
41. Conrad, R. Copy-Paste-Aug. 2020. Available online: https://github.com/conradry/copy-paste-aug (accessed on 5 December 2024).
42. Fawcett, T. An introduction to ROC analysis. *Pattern Recognit. Lett.* **2006**, *27*, 861–874. [CrossRef]
43. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common objects in context. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; pp. 740–755.
44. Thielen, N.; Rachinger, B.; Schröder, F.; Preitschaft, A.; Meier, S.; Seidel, R. Comparative Study on Different Methods to Generate Synthetic Data for the Classification of THT Solder Joints. In Proceedings of the 1st International Conference on Production Technologies and Systems for E-Mobility (EPTS), Bamberg, Germany, 5–6 June 2024; pp. 1–6.
45. Singh, K.; Navaratnam, T.; Holmer, J.; Schaub-Meyer, S.; Roth, S. Is synthetic data all we need? benchmarking the robustness of models trained with synthetic images. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 16–22 June 2024; pp. 2505-2515.
46. Paproki, A.; Salvado, O.; Fookes, C. Synthetic data for deep learning in computer vision & medical imaging: A means to reduce data bias. *ACM Comput. Surv.* **2024**, *56*, 271.

MDPI AG
Grosspeteranlage 5
4052 Basel
Switzerland
Tel.: +41 61 683 77 34

Electronics Editorial Office
E-mail: electronics@mdpi.com
www.mdpi.com/journal/electronics

Academic Open
Access Publishing

mdpi.com

ISBN 978-3-7258-4186-8