

Martin Frické

Artificial Intelligence and Librarianship: Notes for Teaching

3rd Edition



Martin Frické,
Professor Emeritus
School of Information
The University of Arizona,
Tucson, AZ, USA

ISBN 978-0-473-72294-4



© 2024 SoftOption ® Ltd. (Wanaka, NZ). Email: support@SoftOption.Us



This work is licensed under CC BY 4.0. To view a copy of this license, visit <http://creativecommons.org/licenses/by/4.0/>

This license requires that reusers give credit to the creator. It allows reusers to distribute, remix, adapt, and build upon the material in any medium or format, even for commercial purposes.

Cover Image: Charles Willson Peale, *The Artist in his Museum* 1822

Author's Note

This intellectual area, and its practical applications, are advancing rapidly. This poses a problem for a book of this kind. Basically, some of it will always be going out of date. Sorry.

7/4/2024

Preface

The third edition has changes and additions. These include:

- a new Chapter 6 on evaluation and the future
- new materials in Chapter 5 on current large language and multimodal models
- scattered revisions, corrections, and updates.

Table of Contents

CHAPTER 1: INTELLECTUAL BACKGROUND	18
1.1 Introduction to Artificial Intelligence	18
1.2 A Genuine Great Leap Forward	24
1.3 Digitization and Transcription.....	26
1.4 A Paean to Text in Structured Digital Form	29
1.4.1 Text-to-Speech	29
1.4.2 Machine Translation	30
1.4.3 Search and Navigation.....	32
1.4.4 Preservation and Archiving.....	33
1.4.5 Free Books!	33
1.4.6 Natural Language Processing.....	33
1.4.7 Processing by Computer Software.....	34
1.5 Data and the Need for Good Data	34
1.6 Types of Machine Learning.....	37
1.6.1 Supervised	37
1.6.2 Unsupervised	39
1.6.3 Semi-Supervised.....	40
1.6.4 Self-Supervised	41
1.6.5 Reinforcement	43
1.6.6 Reinforcement Learning from Human Feedback (RLHF)	45
1.7 The Concept of Algorithm	46
1.8 Annotated Readings for Chapter 1.....	48
CHAPTER 2: CHATBOTS	50
2.1 Introduction	50
2.2 Dialog Processing.....	51
2.3 ELIZA to ALICE	54

2.4 The Turing Test.....	57
2.5 Machine Learning Chit-Chat Bots	57
2.6 LaMDA	58
2.7 ChatGPT	59
2.8 Task-Oriented	62
2.9 GPTs	65
2.10 Annotated Readings for Chapter 2	68
CHAPTER 3: LANGUAGE MODELS.....	70
3.1 Introduction	70
3.2 Markov Chains	71
3.3 Hidden Markov Models	75
3.4 Shannon's Guessing Game	77
3.4.1 Introduction.....	77
3.4.2 Shannon's Approximations as Markov Processes	79
3.4.3 Training a Shannon-Markov Model to Produce 'A Baby GPT'	82
3.5 Taylor's Cloze Procedure	86
3.6 nanoGPT and an Illustration of Training.....	87
3.7 Embeddings	89
3.8 Word Embeddings and Word2Vec.....	92
3.9 Adding Knowledge to Language Models	94
3.10 InstructGPT and the Insights it Provides	96

3.11 Annotated Readings for Chapter 3	100
CHAPTER 4: LARGE LANGUAGE MODELS	101
4.1 Introduction	101
4.2 Seq2Seq, Encoder-Decoder Architecture, and Attention	102
4.3 Attention and Transformers	104
4.4 Large Language Models and Foundation Models	105
4.5 Foundation Models	105
4.5.1 BERT	106
4.5.2 GPT-3, GPT-3.5, GPT-4	107
4.6 Bigger is Better and Switch Transformers	109
4.7 Base Models to Assistants to Agents	110
4.8 Concerns and Limitations	117
4.8.1 Hallucinations	117
4.8.2 Fakes and Deepfakes	118
4.8.3 Source Training Data Intellectual Property, Privacy, and Bias	119
4.8.4 Intellectual Property of the Generated Output	121
4.8.5 Cybersecurity	123
4.8.6 Apparent Conflict with Chomsky's Theories	123
4.8.7 Environmental Costs	124
4.8.8 Lack of Transparency	125
4.9 Adding Knowledge and Reasoning to LLMs	126
4.10 Annotated Readings for Chapter 4	127
CHAPTER 5: LARGE MULTIMODAL MODELS	130
5.1 Introduction	130
5.2 Built in Safety Restrictions for GPT-4V	132

5.2.1 ‘Inherited’ Restrictions	132
5.2.2 Privacy	133
5.2.3 Stereotypes and Ungrounded Inferences	133
5.2.4 Be My Eyes— Be My AI	135
5.2.5 An Assessment of the Restrictions.....	135
5.3 A General Sense of What GPT-4V Can Do	136
5.3.1 Follow Textual Instructions.....	136
5.3.2 Read Printed or Handwritten Text	137
5.3.3 Read Some Mathematics.....	143
5.3.4 Read Data and Reason with It	143
5.3.5 Follow Visual Pointing in Images	143
5.3.6 Analyze Images Including Medical Images	145
5.3.7 Use Ordinary Common-Sense Knowledge and Reasoning Across Modes....	149
5.3.8 Be an Educational Tutor	150
5.3.9 Use Visual Diagrams When Writing Computer Code	151
5.3.10 Have Temporal and Video Understanding	151
5.3.11 Answer Intelligence Quotient (IQ) Tests	152
5.3.12 Avoid False Presuppositions	153
5.3.13 Navigate Real and Virtual Spaces.....	153
5.4 Yang et al.’s Conclusion on GPT-4V	154
5.5 GPT-4 Turbo (Early 2024).....	155
5.6 GPT-4o (Later 2024)	156
5.7 Google’s Gemini	156
5.8 Anthropic’s Claude.....	157
5.9 Meta’s LLaMa.....	158
5.10 Voice	159
5.11 Possible Applications for LMMs.....	159
5.11.1 Smartphone Uses.....	159
5.11.2 Spot the Difference.....	160
5.11.3 Producing Reports from Medical Images.....	160
5.11.4 Assist with Image Generation	160
5.11.5 Extension with Plugins	161
5.11.6 Retrieval-Augmented Generation (RAG)	161

5.11.7 Label and Categorize Images	162
5.11.8 Identify Objects.....	162
5.11.9 ‘Igor’, AI Advantage and AI Community	162
5.12 Annotated Readings for Chapter 5.....	163
CHAPTER 6: EVALUATION AND THE FUTURE	164
6.1 Reliability, Trustworthiness, and Alignment.....	164
6.2 System 1 and System 2.....	166
6.3 Benchmarks	167
6.3.1 Introduction.....	167
6.3.2 Multi-turn dialogs	167
6.3.3 Chatbots	168
6.3.4 Reasoning.....	168
6.3.5 Common sense reasoning	169
6.3.6 MMLU	170
6.3.7 Coding.....	171
6.4 Artificial General Intelligence (AGI)	173
6.5 The ARC-AGI Benchmark.....	175
6.6 Artificial Super Intelligence (ASI)	176
6.7 Annotated Readings for Chapter 6	178
CHAPTER 7: BIAS AND UNFAIRNESS	179
7.1 Algorithmic Pipeline + Data = Machine Learning.....	179
7.2 Some Clarification of the Terms 'Bias' and ‘Unfairness’	181
7.3 Forms of Bias in Wider Machine Learning	186
7.4 Bias in Natural Language Processing	187
7.5 Some Clarification of the Term 'Algorithm'.....	192

7.6 Computer Program Inadequacy	194
7.7 Bias in the Context of Wider Machine Learning Programs.....	197
7.7.1 Fairness ('Distributive Justice')	198
7.7.2 Debiasing Representation	208
7.7.3 Panopticon Bias, the Panopticon Gaze	209
7.7.4 Bias in (Librarianship) Classification	212
7.8 Stochastic Psittacosis: LLMs and Foundation Models	212
7.9 Supplement: The Bias of Programmers	216
7.9.1 The 'Biases' of Professional Programmers	216
7.9.2 The Biases of All of Us as Programmers	218
7.10 Annotated Readings for Chapter 7.....	218
 CHAPTER 8: BIAS IN MACHINE LEARNING AND LIBRARIANSHIP...	221
8.1 Introduction	221
8.2 Harms of Omission.....	223
8.3 What to Digitize	223
8.4 Search, Primarily Using Search Engines.....	224
8.5 Social Media, Dis-, Mis- and False-Information.....	231
8.6 Bias in the Organization of Information	231
8.6.1 Introduction	231
8.6.2 Be Careful, and Sparing, with Emotive Content	233
8.6.3 Warrant and Controlled Vocabularies.....	233
8.6.4 The Act of Classification Has Consequences.....	239
8.6.5 Taxonomies Have Consequences.....	241
8.6.6 The Current State of Libraries and Their Organizational Systems.....	243
8.6.7 Designing Information Taxonomies for Librarianship	245
8.7 Navigation: Metadata Supported and Otherwise.....	247

8.8 Ethical Arguments to Underpin Assertions of Harms of Bias.....	249
8.9 Annotated Readings for Chapter 8.....	250
CHAPTER 9: WHAT MIGHT NATURAL LANGUAGE PROCESSING (NLP)	
BRING TO LIBRARIANSHIP?	251
9.1 Introduction	251
9.2 The Pre-Processing Pipeline.....	252
9.3 Text Embeddings and Similarity	254
9.3.1 Searching by Meaning (Semantic Search)	256
9.3.2 Research Trails.....	257
9.3.3 Classification	258
9.3.4 One Style of Recommendation	258
9.3.5 Plagiarism Detection	258
9.4 Named Entity Recognition	259
9.5 Topic Modeling	260
9.6 Text Classification Problems	261
9.6.1 Shelving and Subject Classification	262
9.6.2 Sentiment Analysis	262
9.6.3 Author or Genre Recognition.....	263
9.7 Controlled Vocabularies, Thesauri, and Ontological Vocabularies .	264
9.8 Indexing and Automatic Indexing.....	265
9.9 Abstracts, Extracts, Key Phrases, Keywords, and Summaries	268
9.10 Text Mining and Question Answering.....	271
9.11 Machine Translation.....	271
9.12 Evidence	271

9.13 This Is Not Magic	272
9.14 Text Processing and Laws	273
9.15 Annotated Readings for Chapter 9	274
CHAPTER 10: WHAT ARE THE OPPORTUNITIES FOR LIBRARIANS?	275
10.1 Introduction.....	275
10.2 Librarians as Synergists	279
10.3 Librarians as Sentries	283
10.4 Librarians as Educators	284
10.5 Librarians as Managers.....	286
10.6 Librarians as Astronauts.....	287
10.7 Annotated Readings for Chapter 10.....	288
CHAPTER 11: LIBRARIANS AS SYNERGISTS	290
11.1 Intellectual Freedom	290
11.1.1 Text Recognition	292
11.1.2 Speech to Text.....	302
11.1.3 Sign Language to Text, and Text to Sign Language.....	304
11.1.4 Helping Filter and Personalize.....	305
11.1.5 Scholarly Publishing.....	306
11.1.6 What Can Be Done With Computer Text.....	306
11.1.7 ELI5 Translation.....	306
11.2 Improving the Intermediation Between 'Users' and 'Information Resources'	307
11.2.1 Some Users Might Not Be Human.....	307
11.2.2 Some Resources Might Not Be Resources.....	308
11.2.3 Digital Archiving	308
11.2.4 Enhanced Search Engines	308

11.2.5 Personalization and Recommendation.....	311
11.2.6 Recommender Systems.....	312
11.2.7 Understanding What the User is Asking For.....	315
11.2.8 Text Mining.....	315
11.2.9 Information Assistants (and ‘GPTs’)	316
11.3 Improving Traditional Cataloging, Classification, and Retrieval Tools	318
11.3.1 NLP Inspired Improvements	321
11.3.2 Metadata Generation and Automatic Cataloging.....	322
11.3.3 Some Retrieval Tools	323
11.4 Chatbots	330
11.4.1 Reference Interviews.....	331
11.4.2 Virtual Services	333
11.4.3 Chatbots as Continuous User Testing of a Library's Public Interface.	334
11.5 Release, Produce, or Curate Training Data.....	334
11.6 Debunking, Disinformation, Misinformation, and Fakes	336
11.7 Social Epistemology.....	336
11.8 Robots	339
11.9 Images.....	341
11.10 Annotated Readings for Chapter 11.....	342
CHAPTER 12: LIBRARIANS AS SENTRIES	343
12.1 Copyright and Intellectual Property.....	343
12.2 Intellectual Freedom	343
12.3 Censorship and Algorithmic Curation.....	344
12.4 Privacy	346

12.5 Bias	347
12.6 Social Epistemology	347
12.6.1 Reliability, Validity, and Over Confidence	347
12.6.2 Confirmation Bias and Poor Reasoning.....	348
12.6.3 Misinformation.....	348
12.6.4 Awareness of the Digital Literacy of Patrons.....	348
12.7 Chatbots.....	349
12.8 Personalization and Paternalism.....	350
12.9 Images and Facial Recognition Technology	352
12.10 Losing Jobs	353
12.11 Annotated Readings for Chapter 12.....	354
CHAPTER 13: LIBRARIANS AS EDUCATORS.....	355
13.1 Information Literacy (for Consumers of Information).....	355
13.2 Artificial Intelligence Literacy.....	355
13.3 Data Information Literacy (for Producers of Information)	358
13.4 Changes in Learning and Teaching.....	359
13.5 Scholarly Communication.....	359
13.6 Academic Libraries Collaborating with other University Units.....	360
13.7 AI Laboratories in the Library	360
13.8 Automated Decision-Making.....	361
13.9 Explainable Artificial Intelligence (XAI)	367

13.10 Annotated Readings for Chapter 12	370
CHAPTER 14: LIBRARIANS AS MANAGERS	372
14.1 Coming on Board	372
14.2 Data and Analyses	375
14.3 Evidence-Based Librarianship	376
14.4 Data-Driven Decision Making	377
14.4.1 Collection Building and Management	377
14.4.2 Circulation and User Studies	377
14.4.3 Processing in Libraries	377
14.4.4 Research and Scholarship	378
14.4.5 Service Quality	378
14.5 Acquiring the Appropriate AI Tools.....	378
14.6 Analysts and Staff.....	379
14.7 Fear of AI	379
14.8 Annotated Readings for Chapter 14.....	380
CHAPTER 15: LIBRARIANS AS ASTRONAUTS.....	381
15.1 Astronaut Training	381
15.2 Why Should You Learn How To Do It?.....	381
15.3 What are the Real Creative Possibilities	382
15.4 Sitting in Your Tin Can.....	384
15.5 Exploring World 3	385
15.5.1 Undiscovered Public Knowledge (UPK)	385
15.5.2 Literature-Based Discovery (Text Based Informatics)	388

15.5.3 A Message to Librarian Astronauts	388
15.6 Annotated Readings for Chapter 15	389
APPENDIX A: SOME THEORETICAL BACKGROUND TO LIBRARIANSHIP	390
A.1 Concepts, Classification, Taxonomies, and Items.....	390
A.2 Controlled Vocabularies, and Thesauri	391
A.3 Ontologies and Ontological Vocabularies.....	393
A.4 Objective, Intersubjective, and Subjective.....	395
A.5 Emotive and Descriptive Content	397
A.6 Classification Schemes and the Act of Classification.....	399
A.7 Annotated Readings for Appendix A.....	401
APPENDIX B: WORKING WITH LLMS	402
B.1 Introduction	402
B.2 Prompts and Prompt Engineering	403
B.2.1 Basic Examples of Zero-Shot Prompting.....	405
B.2.2 Examples of Few-Shot Prompting.....	411
B.2.3 Chain of Thought Prompting.....	413
B.2.4 Tuning, or Configuring, the Models or Prompts.....	415
B.3 Choices on Development.....	418
B.4 Moving Forward With LangChain	421
B.4.0 A Note on the Status of LangChain and Similar as of 11/6/2023	421
B.4.1 What is LangChain?	422
B.4.2 LangChain Experiments Displayed to a Web Page.....	424
B.4.3 LangChain Using Jupyter	435
B.4.4 Resources for LangChain using Jupyter	438

B.5 Annotated Resources for Appendix B	439
APPENDIX C: TWO IMPORTANT METHODOLOGICAL POINTS.....	441
C.1 False Positives and False Negatives.....	441
C.2 The Base-Rate Fallacy	443
C.3 Annotated Readings for Appendix C.....	447
APPENDIX D: CAUSAL DIAGRAMS	449
D.1 Causation and Correlation.....	449
D.2 Causal Diagrams	451
D.3 Annotated Readings for Appendix D	467
APPENDIX E: KNOWLEDGE GRAPHS.....	468
E.1 Knowledge Graphs	468
E.2 Annotated Readings for Appendix E.....	470
GLOSSARY	471
BIBLIOGRAPHY	507

Chapter 1: Intellectual Background

1.1 Introduction to Artificial Intelligence

Artificial Intelligence (AI) addresses a subset of problems that lend themselves to solution by computers, computer programs or algorithms. Often the AI algorithms are coupled with the use of outside data. The problems in question are ones that an ideally rational and intelligent human being would be able to solve, given the time, resources, and ingenuity. One of the first examples of an AI program within librarianship is that of Optical Character Recognition (OCR). Patrons with poor vision might not be able to read all the books that a library contains. Yet a human library assistant could read books aloud or transcribe them into braille or some other assistive medium. In the mid-1970s, Ray Kurzweil devised and assembled computing machines that did OCR and text-to-speech synthesis. The machines could read books aloud. Kurzweil machines are a massively enabling technology, removing barriers in the access to information. At their core is an AI problem, the problem of recognizing text i.e. classifying letters, and classifying words (where those word tokens might be of different sizes, fonts, colors etc.).

AI in libraries should be distinguished both from automation and from the plain use of computers. Libraries have been taking advantage of automated processes for almost as long as there have been libraries. For example, the Gutenberg printing press, from 1440, is a form of automation; it automates the production of books and documents. Secondly, as society has benefitted

from the introduction of computers, so too have libraries. Computers-in-Libraries is its own wide ranging and active area of study, with conferences and publications. Computers-in-Libraries includes AI uses and more besides. As examples of the more besides, the issuing and tracking of books typically will use computers and databases, as will the financial record systems for the salaries and payment of librarians. AI in libraries is different to automation and plain computing.

Wherein do the differences lie? The use, or simulation, of intelligence, is the main one. What about some further actual and potential examples from librarianship? There is machine translation of texts from one natural language to another. There are ‘recommender’ systems that can recommend books or resources to patrons. There are, or have been, intelligent assistants in libraries, such as Stella, Beacon, or Bizzy, that can answer reference questions and conduct reference interviews (Sanji, Behzadi, and Gomroki 2022). There are the very familiar search engines and OPACs(Online Public Access Catalogs)/Discovery Systems, most of which have AI components. Final example: AI can add value to the recorded knowledge in libraries by extracting, synthesizing, and summarizing that knowledge. There is an abundance of recorded knowledge in libraries. The sheer volume of the recordings would overwhelm a librarian or team of librarians. Yet, the Lithium Ion ‘encyclopedia’ was written by a computer program running deep text analysis of holdings within libraries (Writer 2019).

AI has made massive advances in the last 30 years or so, especially in the areas of Machine Learning (ML) and Deep Learning (DL). Both these approaches involve *learning*. Back in 1959, Arthur Samuel wrote:

... a computer can be programmed so that it will learn to play a better game of checkers than can be played by the person who wrote the program. Furthermore, it can learn to do this... when given only the rules of the game, a sense of direction, and a redundant and incomplete list of parameters which are thought to have something to do with the game, but whose correct signs and relative weights are unknown and unspecified. The principles of Machine Learning verified by these experiments are, of course, applicable to many other situations. (Samuel 1959)

Such programs learn by playing and by associating winning and losing with various strategies, moves, and tactics.

A big advance came about fifty years after this, maybe around 2010, when the amount of data that it was possible to gather, record, and process, expanded massively (amounting to the so-called ‘Big Data’). It should be emphasized here that it is not merely the amount of data that is important, but also it is the ability to process that data with a timely throughput (Amodei et al. 2019). Often this kind of learning has similarities to doing statistics with extremely large data sets. ML is distinctly different from other areas of AI, and the difference lies principally with the ‘learning’. For example, a while back, computer programs to diagnose a medical condition, say cancer, would mimic expert doctors and consultants, perhaps by attempting to extract rules governing sound or intelligent diagnoses and applying those rules. A programmer, or team of programmers, would write a program to diagnose cancer and the program would be finished and done.

It is true that some tuning might be carried out after the fact, if the program was making some poor or inaccurate diagnoses. Usually, this tuning would consist of the programmers re-writing or changing the program, not of the program modifying itself by learning. So, in general, these programs themselves did not pay a lot of attention to data that arose after they were written. This style of approach is AI by ‘expert system’. Nowadays ML would approach diagnoses entirely differently. There would be a ‘blank slate’ program that could take in vast amounts of data, data about different patients, and their many and varied properties, qualities, and ‘features’, data about the images, and their properties, and so on. The program would be set up to focus on the ‘label’ which identified whether a patient had cancer or not. Then there would be a *training set*, which would be data about existing patients, and even members of the public, and known data as to whether they had cancer. The program would learn which features were associated, or correlated, with (the label of) cancer and which not. It would be able to predict which members of the training set had cancer (hopefully with a good degree of accuracy). Then, given success, the program would be released in the world at large and applied to a *test set* or *test cases* i.e. real cases where the diagnoses are not known. (And, possibly, results for real cases will be added through time to the training set to improve performance yet further— although care would be needed were this to be attempted.) You can see how in some ways this differs from diagnoses by medical doctors and consultants or by expert system. The consultants will have had years of training, of course, but they are not in any position to consider anywhere near the number of features that a program can look at. A DL program might consider thousands of features and have been trained on tens of thousands of cases. Also, while some, perhaps most, of the

consultants' knowledge will be explicit, some of it will be tacit (i.e. not articulable). It would amount to perhaps vague unspeakable hunches refined by years of experience. Tacit knowledge is a problem for the prior expert system AI approach: for somehow hunches would have to be converted into rules. But if the consultants themselves cannot put this tacit knowledge into words, how is this to be done? Then, going back to consultants, tacit knowledge itself is hard to produce, instill, and disseminate. If more medical consultants are needed in a health system, the newcomers will have to go through an apprenticeship with an accomplished consultant and years of training. Of course, reproducing a computer program can be done with the click of a mouse. A question is: which system is best at cancer diagnosis? With some forms of cancer, the ML programs provide the best diagnoses (K. Das et al. 2021). In sum, with cancer diagnoses, ML can do better than both prior AI expert systems, and human doctors.

Deep Learning (DL) is a subset of ML inspired by the human brain, in particular by neurons and neural nets. Neurons are nerve cells in the brain that can communicate with other neurons using electrical signals and synapses. Specific neurons are triggered, or become excited, and that activation cascades through other neurons. This provides function to the nervous system— to thinking, reasoning, responding, learning, paying attention, and other cognitive abilities. In a DL implementation, there is a network of software 'neurons' organized in layers. Certain levels of thresholds of input features activate neurons in the first layer; in turn, first layer activated neurons activate some neurons in the second layer, and so on through several layers. At the output level, particular (software) units

indicate, for example, whether the patient being diagnosed likely has cancer. DL is quite involved to set up, and it is demanding on resources. Large amounts of data are needed, then large amounts of training time and computing power to adjust the activation levels and various other ‘biases’. Separately, DL often can be opaque as to what is going on. By the time the triggering has gone through several layers, transparency can be lost and that is distinctly a drawback.

Here is an example (Google for Developers 2022). A cancer diagnosis DL program may learn by processing images from different hospitals. But if one of the hospitals is specifically a cancer hospital, then that feature, if used by the DL program, may result in certain types of images being given a higher probability of indicating cancer— those images sourced from the cancer hospital. And the program would be right (those images do have a higher probability of indicating cancer). But, really, this is being right for the wrong reason. You want the DL program to be analyzing the images (supplemented perhaps with facts about the patients and their histories) not reasoning from the originating hospital of the images. To guard against possibilities like these, transparency in the DL software program helps. You need to know what the program is doing, what the program is using in its reasoning. But DL programs can lack transparency.

Two areas of human intelligence had proved special challenges to AI: addressing text (including languages, and translation) and addressing images. Hitherto, computers could not ‘understand’ or translate natural languages, and they could not recognize and process images, including

videos, as sources of information. Machine learning and DL have changed that: text and images are now fair game for AI.

AI as a scholarly discipline covers many different areas. But for the purposes of the cascade of recent AI developments, and of AI in libraries, we can focus on ML and DL.

1.2 A Genuine Great Leap Forward

A great leap forward came from Transformers, Large Language Models, and Foundation Models. At the end of November 2022, ChatGPT was released to the public. By January 2023, it had 100 million active users. Many more interested observers were aware of its existence— more than 40% of the adults in the United States know about it. It is the fastest growing, and most widely used, software application of all time. There is some history to it. In 2017, Ashish Vaswani and co-authors published the paper *Attention is all you need* (Vaswani et al. 2017) (see also (Huang et al. 2018)). This introduced Transformers. Shortly thereafter there started to emerge Large Language Models and Foundation Models. (What all these are will be explained later in the book.) ChatGPT is a Transformer, and a Large Language Model, and initially it was a fine-tuned version of the Foundation Models GPT-3 and GPT-3.5. (As of 7/1/2024, ChatGPT uses GPT-4o.)

Pretty much any machine learning or deep learning program can be built from a Foundation Model (that is why they are called 'Foundation Models'). Also, the results of systems built using Foundation Models will likely be

superior to any other approach. So, a correct strategy in solving a machine learning problem is to address it using a Foundation Model. But Foundation Models themselves are very expensive, and resource needy, to create. We are talking here of hundreds of millions of dollars, months of computing time, and of using a large portion of the internet as data. Only a few large commercial companies have been able to produce the biggest and best of the Foundation Models. Producing Foundation Models is not the sort of thing that you and I are going to do, nor are most universities, nor even most governments. Some Foundation models have been open-sourced and are freely available to all. This is a mixed blessing. Allowing programmer/users to have the code, lets them see what the code is and, historically, with open-sourced projects like Linux, the programmers can contribute, improve the code, 'catch bugs', etc. But Foundation model ML code is a little different. There are deep security concerns and great potential for unintentional, and even intentional, harm. Trusting a few massive companies like Google, OpenAI, and Microsoft to look after us is not brilliant, but it is probably better than making the code available to all and sundry (including bad actors). That said, the massive company Meta open-sources its code. Also, Hugging Face provides a hub, a library of open-source Foundation Models (Hugging Face 2023). Some commercial Foundation Models have Application Programming Interfaces (APIs) that allow Users to pay a fee and use them (for now at least). For example, May 2023, from (OpenAI 2022c), you can pay \$20 a month and have good API access to GPT-4.

From an educational point of view, we can take a machete and cleave out and discard pretty much all of machine learning prior to Foundation Models and start our studies at that point. Andrej Karpathy writes:

... the whole setting of training a neural network from scratch on some target task (like digit recognition) is quickly becoming outdated due to finetuning, especially with the emergence of foundation models like GPT. These foundation models are trained by only a few institutions with substantial computing resources, and most applications are achieved via lightweight finetuning of part of the network, prompt engineering, or an optional step of data or model distillation into smaller, special-purpose inference networks. I think we should expect this trend to be very much alive, and indeed, intensify. In its most extreme extrapolation, you will not want to train any neural networks at all. (Karpathy 2023b)

1.3 Digitization and Transcription

Digital computers work with electronic digits, surprise. They work with the digits 0s and 1. But, unfortunately, at least some of the information resources that the ML algorithms have the potential to address are not, or were not, in digital form. For example, Shakespeare’s only surviving playscript— *The Booke of Sir Thomas Moore*— was not (British Library 2020). So, digitization of those resources not born digital is an important precursor to wide-ranging ML in librarianship.

The 2002 Google Books Project, or Google Project Ocean, was a very early attempt to address digitization. Its approach was to use OCR scanning on the physical resources. It was not prompted by the needs of ML, rather it was aiming for a Universal Library. As James Somers writes:

You were going to get one-click access to the full text of nearly every book that's ever been published. Books still in print you'd have to pay for, but everything else—a collection slated to grow larger than the holdings at the Library of Congress, Harvard, the University of Michigan, at any of the great national libraries of Europe—would have been available for free at terminals that were going to be placed in every local library that wanted one.

At the terminal you were going to be able to search tens of millions of books and read every page of any book you found. You'd be able to highlight passages and make annotations and share them; for the first time, you'd be able to pinpoint an idea somewhere inside the vastness of the printed record, and send somebody straight to it with a link. Books would become as instantly available, searchable, copy-pasteable—as alive in the digital world—as web pages. (Somers 2017)

There is another huge potential benefit that arises from the network effects of having many books digitized collectively. The Google Page Rank algorithm, which the core of its very successful search engine for the web, uses links *between* web pages to establish ranking. Something similar, citation indexing, citation counting, and citation ranking, had long been in use in bibliometrics and informetrics working with paper based libraries (Dizikes 2011; Araújo, Castanha, and Hjørland 2021). If books, journals, and other paper-based resources were also to be available digitally, powerful algorithms using linking then could be used in conjunction with massive compute power— search and ranking of actual books in libraries would be able to be improved. So, the promise was of a Universal Library, with improved search and ranking of the sources.

Unfortunately, the Google Book Project got mired in difficulties of one kind or another, mainly legal, and, as of 2023, it seems to have drifted off into

limbo (Somers 2017; Howard 2017). The work does continue, but in a somewhat slower and piecemeal fashion. We cannot pretend that this is a good result, but modern practices have rendered it less of a total disaster than it might have been. Most published books today are written on word processors on computers. Even if they are not produced that way, they are usually in electronic form prior to being printed (if, indeed, they are printed). The authors or publishers may or may not make the electronic form available to the world at large. Authors write books to have them read. Publishers ‘print’ books to have them read (and sold or licensed). The parties concerned have incentives. So, one way or another, these electronic forms should become available.

Separately, physical documents themselves are perhaps not so important as they once were. The indexed Web is around 60 billion pages (de Kunder 2022), and Google estimated that the number of physical books they needed to scan was around 130 million. The Web, the Internet, is a lot larger than the combined book collections in libraries.

There is a distinction or consideration that should be mentioned here: that between digitization and transcription (or decoding). Were we to use a modern smartphone to take images of Shakespeare’s *The Booke of Sir Thomas Moore* we would have the playscript in digital form. But the playscript itself is in letters, words, sentences, and in speeches. It has structure. It is that structure, or some or most of it, that we would like captured digital form. When Google set out on their book project, they did not aim to photograph every page of every book. They aimed to use Optical Character Recognition (OCR) to get to the structured text in digital form as

os and 1s. They aimed to transcribe to digitized computer text that could be copied, pasted, read aloud, translated to other languages, etc. It is that structured digitization that is the desired goal. ML can usually work with a digital image of a text and do an OCR extraction of structured text. But, nowadays, when authors are writing on word processors, and publishers are using the digitized form throughout the workflow, structured digitization should be available without having use OCR on images.

Many libraries have digital collections, for example the Library of Congress has <https://www.loc.gov/collections/> with millions and millions of items. Its collection *Chronicling America: Historic American Newspapers* has over 20 million images (i.e. photographs) of pages of historical newspapers. That is great, but for certain kinds of research those need to be transcribed into the 0s and 1s of computer text. (Very shortly machine learning will be able to do that task in the blink of an eye.)

1.4 A Paean to Text in Structured Digital Form

Having text in digital form is very valuable in terms of what can be done with it. Here are some possibilities. (There may be some overlap between the categories.)

1.4.1 Text-to-Speech

(Computer) text can be spoken aloud. That is one part of what the Kurzweil machines, mentioned above, do. This is not trivial, by any manner of means, but it is also not a challenging AI problem. Obviously, text-to-

speech computer programs need to know the language of the text and how it is to be spoken (its pronunciation).

1.4.2 Machine Translation

Computer text, from a source natural language, can be translated by computer into other languages. This used to be done in an expert systems style of AI. There would be a dictionary between the source and target languages, grammar rules for both languages, and programmers would try to figure out translation rules that algorithms could use to get from one language to the other. The modern approach would be to use ML and DL. This relies on existing (human) translations between texts. These come from a variety of sources; for example, the United Nations provides translations into six languages of all its texts. The Bible has been translated into many languages, although it does have the drawback of having a very specific and distinctive written style. Google Translate uses the DL program ‘Google Neural Machine Translation’. At a first cut, this is what it does. There are a number of texts that have been translated, by human translators, from one language to another, say from English to French (and French to English). To translate from English to French the Neural D.L. will find all the existing occurrences of the relevant sentence, phrase, or word, in English and look up how they have been translated by the human translators. Then it will favor what seems to be the best translation. Making the latter judgement depends also on what the D.L. has learned, supplementary rules, dictionaries, etc.

The resulting translations are reasonable, but not perfect. (They can be improved by further editing by human readers or translators.) You can see where some of the problems may lie. Context is one. Single words might be homographs (for example, 'bank' (financial institution) and 'bank' (side of a river)) and the translation system might lack the context to disambiguate them. Plain lack of understanding may be another. Human translators presumably will understand their translation, and this would prevent certain kinds of ridiculousness. In a general way, a neural DL will learn what is silly and what is not. But still, in no sense does it understand the text, and this leaves open to making certain kinds of mistakes (i.e. those where the words, grammar, and construction are all fine but the result is nonsense).

Douglas Hofstadter offers interesting examples, including:

In their house, everything comes in pairs. There's his car and her car, his towels and her towels, and his library and hers.
(Hofstadter 2018)

As of 2023, machine translation simply cannot translate this correctly, say from English to French. An anecdote. The author's wife had some dental treatment while living in Japan. The Japanese hygienist had a little device that she spoke into in Japanese, and it spoke back in English. So, this is digitization and transcription of speech, translation of one language to another, then speaking aloud the resulting translation. Here is a real example of what it said to her:

Pardon me, may I please put red dye on your dirty teeth!

Despite the occasional oddity, machine translation can work 24 hours a day, 7 days a week on a vast number of sources. It is not expensive. In sum, machine translation can produce a bulk of translations of reasonable quality. Progress in this area is rapid for example, Ankur Bapna and co-authors have a paper *Building Machine Translation Systems for the Next Thousand Languages* (Bapna et al. 2022).

1.4.3 Search and Navigation

Once a text is in digital form it can be searched and navigated very efficiently, even if it has no preemptively prepared index. For example, the words 'butterflies' and 'rhopalocera' are synonyms, and 'lepidoptera' is a term for an order of insects that includes rhopalocera (i.e. it is a more general term). Imagine yourself to be a young person with a physical book on the subject of insects. You are interested in the specific topic of butterflies. But perhaps the word 'butterflies' is not in the index of the book, nor does it appear frequently in the text if at all (although 'rhopalocera' appears many times, being the favored word of the synonym pair). Also, on this occasion, you are interested more widely in insects. You would like your search to include butterflies but also, in part, to be a little more general. Basically, you will be disappointed with what you are able to achieve (which will be more-or-less nothing). In contrast, were the book to be digitized and it were searched by computer (either with Large Language Model, or with thesaurus support), the desired tasks and goal would be trivial and satisfiable in milliseconds.

1.4.4 Preservation and Archiving

Loss of the contents of some texts could be catastrophic— for example, contracts, treaties, or the works of Shakespeare. Once digitized, once there is a digital surrogate, there are cryptographic techniques to preserve text surrogates 'forever'. They could be encrypted, hashed, and put into a suitable public blockchain using content-based indexing. Alternatively, one digital copy could be placed in the *Billion Year Archive* through the good graces of the Arch Mission Foundation (Spivak and Slavin 2023).

1.4.5 Free Books!

Once text is in digital form the marginal cost of duplication is near zero. Of course, there are considerations of licensing and intellectual property.

1.4.6 Natural Language Processing

Natural Language Processing (NLP) has always used os and is and computers. But it has absolutely flourished with ML, and especially with Large Language Models (which we will discuss later). As a selection of possible procedures or techniques, there is: the entire field of digital

humanities, text summarization, text mining, question answering, information extraction, text categorization, sentiment analysis, plagiarism detection, author and genre recognition, word sense disambiguation, and lexical and ontological acquisition, and text analysis for social applications such as blogs and social networks. So, for example, given two physical texts by unknown authors and the question 'are these texts written by the same author?', digitization and NLP can provide the answer.

1.4.7 Processing by Computer Software

Much data either exists or was initially recorded on a physical medium, such as paper or cards. This data really needs to be digitized in order to be processed by statistics or data science. This processing would then amount to data processing, or data mining, or text data mining (TDM). Vast continents of knowledge or information would be opened up.

1.5 Data and the Need for Good Data

ML aims to learn, and what it is going to learn from is data. Primarily this data is the encoding of observational or experiential reports about the world. At one point, historically, such data would have been assumed to be a solid bedrock. Nowadays, most philosophers of science would describe observation reports, or data of this kind, as being 'theory-laden'. Really, they are two aspects to this assertion. Any observation report relies on theories which infuse either the instruments used or the raw observations

(if there is any such thing as truly raw observations) (Duhem 1914; Hanson 1958). Duhem writes:

Go into this laboratory; draw near this table crowded with so much apparatus: an electric battery, copper wires wrapped in silk, vessels filled with mercury, coils, a small iron bar carrying a mirror. An observer plunges the metallic stem of a rod, mounted with rubber, into small holes; the iron oscillates and, by means of the mirror tied to it, sends a beam of light over to a celluloid ruler, and the observer follows the movement of the light beam on it. There, no doubt, you have an experiment; by means of the vibration of this spot of light, the physicist minutely observes the oscillations of the piece of iron.

Ask him now what he is doing. Is he going to answer “I am studying the oscillations of the piece of iron carrying this mirror?” No, he will tell you that he is measuring the electrical resistance of a coil. If you are astonished and ask him what meaning these words have, and what relation they have to the phenomena he has perceived and which you have at the same time perceived, he will reply that your question would require some very long explanations, and he will recommend that you take a course in electricity.

It is indeed the case that the experiment that you have seen done, like any experiment in physics, involves two parts. In the first place it consists in the observation of certain facts... In the second place, it consists in the interpretation of those facts...

The result of the operations in which an experimental physicist is engaged is by no means the perception of a group of concrete facts; it is the formation of a judgement interrelating certain abstract and symbolic ideas which theories alone correlate with the facts really observed (Duhem 1914).

A consequence of this is that the reports are conjectural, or fallible. Any of them may be mistaken. They are not a bedrock. Second, any collection of data is a selection among what it is possible to observe or measure. The philosopher of science Karl Popper writes:

... the belief that we can start with pure observation alone, without anything in the nature of a theory is absurd; as may be illustrated by the story of the man who dedicated his life to natural science, wrote down everything he could observe, and bequeathed his priceless collection of observations to the Royal Society to be used as evidence. This story should show us that though beetles may profitably be collected, observations may not.

Twenty-five years ago I tried to bring home the same point to a group of physics students in Vienna by beginning a lecture with the following instructions: 'Take pencil and paper; carefully observe, and write down what you have observed!' They asked, of course, what I wanted them to observe. Clearly the instruction, 'Observe!' is absurd. (It is not even idiomatic, unless the object of the transitive verb can be taken as understood.) Observation is always selective. It needs a chosen object, a definite task, an interest, a point of view, a problem. And its description presupposes a descriptive language, with property words; it presupposes similarity and classification, which in their turn presuppose interests, points of view, and problems (Popper 1963).

So, the data used in ML is always incomplete.

It is becoming increasingly common in ML circles to use the phrases 'ground truth' or 'ground truths' or 'ground truth data set'. Caution is needed here. No matter how much care goes into gathering and curating the data, it can still be wrong and incomplete.

There are many other pitfalls that can occur sample data or training data even if the data is absolutely correct. The data may be 'unbalanced'. An example that can be used here is that of a test predicting the (unknown) gender of a person from their (known) height. If the training data, the sample, where the heights and genders are known, consists of 95% males

and 5% females, then mathematics will suggest predicting male no matter what the height of the test subject is.

In sum, getting good training data is tricky.

1.6 Types of Machine Learning

1.6.1 Supervised

Supervised learning is learning with a teacher— a teacher who knows the answers. An example will help here: Optical Character Recognition (OCR). OCR is a solution a supervised ‘classification problem’. It can look at some text e.g. ‘Call me Ishmael’ and identify, or recognize, that the first letter of the text is a ‘C’, the second letter is an ‘a’, and so on. ML OCR will approach this by being taught how to classify characters. It will be supplied with a *training set*, which will be a reasonable sample of letters and the *correct classifications* of what they are. Training sets are typically large. For example, the well-known and widely used MNIST set, which is a collection of hand-written examples of the digits 0 through 9, with correct identification labels, has around 60,000 entries (LeCun, Cortes, and Burges 1998). The overall OCR technique is an optical one, so it is the features of the sample letters that can be detected optically that will be the input (e.g. size, shape, color, grid arrangement of component dots or pixels, etc.). Then the program will attempt to correlate combinations (i.e. vectors) of these with the correct classification e.g. that a particular sample token character is an ‘a’. More than likely, the program will make many mistakes initially.

But either the programmers, or the program itself, will tune various parameters (e.g. weights on the components of the vectors) to improve the classification until it reaches an acceptable level of performance. There is an interesting point to be made here about what are known in statistics as 'omitted variables'. As mentioned, the ML program will start by considering optical input from size, color, pixels etc. But it will then learn which variables of these to include and which to omit. The machine here has an advantage over a human statistician as it has the sheer computing power to run through the alternative possibilities in a reasonable time.

The training set needs to be adequate for the task. For example, if the letter 'j' does not appear in the training set, it is unreasonable to expect the ML program to classify js correctly. Even if js appear, there needs to be enough of them in the various fonts and scripts (cursive or not, monospaced or proportional, etc.) for the program to be able to learn what is correct and what is not. OCR, i.e. recognizing the actual individual characters, would not usually be an end in itself. Rather, the interest would be in the words that those characters form, or, more generally, the text.

If the OCR, or Text Recognition, application has access to a wider context of text, that can improve its performance. For example, if the ML is recognizing entire words from their component characters, and separators, then the first letter of the word 'On' is going to be the letter upper case 'O' and not the numeric letter zero 'o'— the number zero makes no sense in that context.

Supervised learning needs labeled data for its training data. Getting such data with quality and in quantity is not easy. Often experts will be required to do the labeling e.g. of medical images, or of the correct cataloging numbers for books in libraries. But many experts would struggle if confronted with 100,000 items to label. Sometimes the process can be eased, at a financial cost, by crowdsourcing (Wikipedia 2023c). For example, the LaMDA models of dialog agents uses crowdsourcing, where the members of the crowd are supplied with calculators and access to a web browser and a search engine so they can check how reliable the output information is that the dialog agents supply (Thoppilan et al. 2022).

1.6.2 Unsupervised

The word ‘supervised’ comes in to qualify learning because it is known, for most of the cases, what the correct answers are. A Machine Learner could, alternatively, be challenged to classify marks (characters) on paper into groups of marks (characters) similar to other characters i.e. clustering (forming clusters of similar characters). That would be *unsupervised* learning, where there are no known right or wrong answers (and no teacher).

A first issue here is that there is no ML constraint as to how many clusters there might be, or should be, in any given problem. For example, the music streamer Spotify plays songs that the listener wishes to listen to; Spotify can cluster these songs into similar songs and perhaps mark those as being ‘playlists’ or ‘radio stations’; then the User can listen to a radio station while

cooking, or while going for a walk etc. But the clustering algorithm needs to know ‘how many radio stations should there be: 2 stations, 7 stations, 300...?’ There is no answer to that question from within the ML system. Spotify itself, or the User, or some outside party, has to decide on the number of stations. Going back to OCR and attempting to do it in an unsupervised fashion. We could help it by saying we would like 36 clusters (we have in mind here one for each letter of the alphabet, and for each digit). Clustering might then lead to mistakes like characters that we consider to be upper case ‘I’s being put together with lower case ‘I’s or zeros ‘o’ with upper case ‘O’s. There is no training set. There are no right or wrong answers available to assess the program. Unsupervised classification might be perfect in certain areas, for example, clustering songs into playlists, but it is not really suitable for OCR.

1.6.3 Semi-Supervised

Supervised and unsupervised classification approaches can be combined. This might be useful when there is a huge amount of data, of which only a small proportion is ‘labeled’ (i.e. it is known what those items are), and the process of labeling is expensive or time consuming or hard to do. For example:- imagine some historical biodiversity researchers who collected specimen samples in the style of Darwin; they also did as much labeling as they could manage, including labeling at least one example of what they thought was every species they came across; then the initially unlabeled specimens were later to be donated to many museums, who, of course, wanted them labeled; this problem might be approached by clustering,

supplemented with a back-and-forth with supervised learning; then the final labeling of the museum samples could be done by machine.

1.6.4 Self-Supervised

The learning techniques mentioned to date have problems and issues. Supervised learning required large amounts of labeled data which is often difficult, expensive, or even near impossible, to obtain. The need for the quality labeling is the cause of the problem. Unsupervised learning simply might not give you what you want.

Self-supervised Learning (SSL) is an ingenious idea which will often be far superior to its alternatives. Basically, it uses unsupervised learning, and the data itself, to label the data, then it uses supervised learning on the now labeled data. To do this the data has to have suitable structure or patterns in it. This gives a context, or contexts to items of data, and the general problem being addressed needs to be tightly specified or understood. SSL really found its place in Natural Language Processing (NLP), and some examples from NLP may make the technique clearer. The foundation models BERT and GPT-3 both have the pre-training task of predicting the next word, or previous word, from a sequence of words in a passage of English (or other natural languages) (Devlin et al. 2019; Brown et al. 2020). The way they do it is to scan vast amounts of English text e.g. trillions of word tokens such as the entire of the Internet (including Wikipedia, Reddit linked sources, all freely available digitized novels, etc.). Apparently GPT-3

was pre-trained on 45 terabytes of data. This is about the same size as one quarter of the holdings of the Library of Congress. This pre-training provides the various transition probabilities from prefix, or suffix, words or sequences, to the current target word, and, essentially, the solution. Now, the data itself, the English text, is not labeled, so this is not supervised learning. But the scanning of the text can produce a pseudo-label for the missing 'gap word'. For example, the label for 'The cat sat on the <?label?> ' can be produced merely by looking through a vast amount of real-life text. Of course, there does not have to be a single one-word answer to this. 'mat', 'table', 'floor', etc. might all be possible answers. But then there will be probabilities associated with the possible answers, and the wider context will provide guidance.

Self-supervised learning has an obvious home with natural language. But it also can be used with images. There is a context to which patches of colors or pixels are close to other patches. Further, foundation models like GPT-3, GPT-4, etc., are becoming multi-modal. 'Multi-modal' means that they can work with different 'modes' such as with text, images, audio, video, etc.. The earlier technique for this was to use text as a stepping-stone. The model would be pre-trained, using SSL on text, then fine-tuned, perhaps with some prompts or labeling, to work on images. Nowadays, many Foundation Models can work with different modes natively, without using text as an intermediary step. SSL offers freedom. Getting good, labeled data at scale is difficult, if not near impossible. It is a barrier or bottleneck. But with SSL, it is not needed.

1.6.5 Reinforcement

Reinforcement learning is familiar to us in daily life. It involves exploration of an environment by trial-and-error, and, as part of this, having what are called 'delayed rewards' (Sutton and Barto 2018). The rewards provide feedback as to how well the trial-and-error is working. Imagine a student backpacker having temporary employment picking apples in an orchard. She gets paid for each apple she picks (but for each apple she picks there will be one less apple to pick on the tree that she picked it from). Also, she gets a bonus for each basket of apples that she picks, especially if she fills the basket faster than other pickers. Bigger apples will fill a basket quicker, but there will be fewer of them in the basket. We will assume here that she is trying to maximize her pay, i.e. her rewards. Quite what her best picking strategy might be is a bit of a question. She presumably will have to change trees from time to time, but changing trees is not actually picking. It amounts to dead time invested in the hope of higher rewards later. She will also have to pay attention to other pickers, and to the simple nature of the individual trees. If she works in the orchard for several days, she should be able to learn of a reasonable strategy by trial-and-error i.e. by trying a few approaches and favoring those yielding higher total rewards. The whole process amounts to reinforcement learning.

Reinforcement learning can be used when the ML system has to make a sequence of tasks, or moves, or steps towards some desired goal and there are rewards or penalties for successes or failures. While learning, the system will typically be allowed to try the overall task of reaching the goal

many times. Games are a good setting for this e.g. Tic-Tac-Toe or Checkers or Chess. There will be an environment, and possibly permitted moves governed by rules. The system will be rewarded or penalized according to whether a particular move is judged to be good and also, perhaps, whether the system wins or loses the game overall. The system is not programmed with any strategy or tactics, rather it explores the game by trial and error. Another example of reinforcement learning would be a robot exploring an environment e.g. finding a route through a maze.

Reinforcement learning is not supervised learning— there is no labeled data. It also is not unsupervised learning or self-supervised learning. It is a *sui generis* ML technique— one that needs to have some reward structure. It certainly can be used in language settings. For example, an ML program might be configured to produce five different translations of the same text; then if these could be ranked, perhaps by a human judge; then the rankings could be used as a reward structure, and a reinforcement learning system introduced improve the system at translating. Sometimes, in this context, the reward structure is called the 'reward model'. Typically, reinforcement learning is very compute intensive— e.g. for chess, the ML system may need to play hundreds of thousands of games. There are many algorithms to produce reinforcement learning, but few, if any, are efficient in really large settings. Supposedly, one of the technologies that enabled some of the uses of Foundation Models, such as ChatGPT, was the invention of Proximal Policy Optimization (OpenAI 2017). Proximal Policy Optimization is a reinforcement learning algorithm.

1.6.6 Reinforcement Learning from Human Feedback (RLHF)

Modern Foundation Models or Large Language Models often use Reinforcement Learning in a very specific way. The training of these takes place in two stages: the initial training to produce a plain vanilla base model, then *Reinforcement Learning from Human Feedback (RLHF)* is used to yield the desired product. A textual base model might be produced by self-supervised training on most of the text on the Internet. This might take months to do, and the result might be able to write presentable English. But the model might at that point lack some desirable qualities (such as answering questions given by prompts) and might possess some undesirable properties (such as lying, giving poor medical advice, revealing private information). The model will then be tuned using RLHF. A jury of perhaps 40 people will be assembled and given maybe 10,000 samples of pairs of answers from the model. Each member of the jury will evaluate the answers, saying which of a pair is better and which worse. Then, those evaluations will be fed back to the model as additional training. The model will adjust itself accordingly.

One point of importance about RLHFs is that as the models get really big, as they are going to, RLHF becomes impossible as a practicality. If a model were to produce a billion responses, no human jury would be able to evaluate them. In 2024, RLHF has been one difference between success and failure with Foundation Models. But it is not going to work in its present form going forward.

[There are plenty of companies advertising on the Internet for (paid) RLHF jury participants. Try ‘advertisements for RLHF participants’ in a Google or Perplexity search.]

1.7 The Concept of Algorithm

Programmers, mathematicians, and computer scientists regard algorithms as finite sequences of steps or instructions to perform a computation. The instructions themselves are considered to be atomic, or not in need of further breakdown or explanation. The earliest algorithms are from about three and a half thousand years ago. Here is one from that period to do division of integers by repeated subtraction. It is assumed that we can determine whether one number is smaller than another, that we can subtract one number from another, and that we can count the number of subtractions that we have done. Then the divisor is repeatedly subtracted until the remainder is less than the divisor. Finally, the quotient (the 'result') is the number of subtractions that have been made. For example, to determine the quotient when 22 is divided by 4 (which we all know to be 5):

$$22-4 = 18$$

$$22-4-4 = 14$$

$$22-4-4-4 = 10$$

$$22-4-4-4-4 = 6$$

$$22-4-4-4-4-4 = 2 \text{ /* and 2 is less than 4 */}$$

5 subtractions have been made so the quotient when 22 is divided by 4 is 5.

[This algorithm is taught in 3rd grade school mathematics.] The act of programming is using a computer programming language to assemble instructions into algorithms. Usually, this process will be modularized. Small algorithms will be created, as components or modules. Then these algorithmic modules will be combined to produce a complex artifact. The end result might be described as being an 'algorithm', but, given sufficient complexity, the plain moniker 'computer program' might be more suitable. Google's Internet Services consist of 2 billion lines of code; Microsoft's Office Suite for the Mac consists of 30 million lines of code (Desjardins 2017). It would be very strange for programmers or computer scientists to describe either of these as being 'algorithms'. They are computer programs or collections of computer programs.

The word 'algorithm' has become transmogrified in recent times. The vernacular has taken a different turn. Where there is no apparent human agency in a computer program, or computer supported service, that affects humans, folk can and do ascribe the agency to an 'algorithm' or 'algorithms'. For example, nowadays a mortgage company will typically use data, computers, and algorithms to determine whether we might qualify for a loan. If we are turned down, we might well say that the mortgage company's 'algorithm', *the algorithm itself*, declared us ineligible for one of their loans. This attribution is not quite right, we will go into that further in Section 7.5.

There is an important detail in connection with algorithms in the context of ML. It is that algorithms can change or self-adjust or self-adapt when faced with (training) data. The ML programs learn by changing their algorithms.

The same beginning, template, or seed algorithm when confronted with two different sets of training data can produce two different trained outcome algorithms. Perhaps that is not a surprise. One would expect different learning from different data to learn from. This adaptation is a small change from how historical algorithms typically behaved, and it can mean, in particular cases, that it is unclear or unknown as to which algorithm is actually running after learning. In turn, this can shroud some or many of the processes in a fog (i.e. in black boxes). This is exactly the opposite of what is desirable. Transparency is valuable both for ethical reasons (for example, when making decisions involving humans) and for technical reasons (for example, when trying to assess the correctness of the programs). Part of the lack of transparency comes from the plain complexity of the systems. Some of the ML systems might have a billion parameters or more— then it becomes hard to know what the values for the parameters are and how they interact. There is another word or label that gets used in this context and that is 'model'. When an ML system is trained and its initial algorithm adapts and changes into another algorithm, known or unknown, transparent or hidden, the end result is often called the 'model'.

1.8 Annotated Readings for Chapter 1

Coding Vidya. “Home | Coding Vidya - Best Computer Science Portal -,” 2023.
<https://codingvidya.com/>. (Coding Vidya 2023) Worth a scan. It does, for example, have lists of the Best Online Courses in Machine Learning.

gwern. “Douglas Hofstadter Changes His Mind on Deep Learning & AI Risk,” 2023.
<https://www.lesswrong.com/posts/kAmgdEjq2eYQkB5PP/douglas-hofstadter-changes-his-mind-on-deep-learning-and-ai>. (gwern 2023) Hofstadter is trusted,

knowledgeable, and intelligent. He was an influential critic of AI and ML. No more.

Sanderson, Grant *But what is a neural network? | Chapter 1, Deep learning* (Sanderson and 3Blue1Brown 2017a) and the references he provides. Neural networks have been mentioned in this Chapter, but they have not been described or explained. They are the core of modern machine learning. However, we are constrained by space in this text. Also, explaining neural networks using just written text and diagrams is not the easiest. The medium is not good for capturing the dynamics of it. Far better is an animation or a video. There certainly are some excellent ones available on the web.

Wikipedia. “Algorithm.” In *Wikipedia*, 2022.

<https://en.wikipedia.org/w/index.php?title=Algorithm>. (Wikipedia 2022a) This provides useful background on the concept of algorithm. We will have more to say on algorithms in Section 7.5.

Chapter 2: Chatbots

2.1 Introduction

The word 'chatbots' is commonly used in AI as a label for Dialog Agents. In turn, dialog agents might be chit-chat agents or task-oriented agents (Daniel Jurafsky and Martin 2021). Chit-chat agents are understood to be software systems that can 'chat' with you. They chat by interchanging text messages or by using sound and speaking and listening. There is no special purpose to these chats, apart from entertainment, engagement, and perhaps companionship. Task-oriented dialog agents also have interactive sequential conversations. But they can do more (and do less). For example, in a general setting, they can book an airline ticket, order a taxi or a meal, set an alarm or timer, produce a recipe for a birthday cake, retrieve information resources, offer instruction etc. Most everybody is familiar with Apple's Siri, or Window's Cortana, or Amazon's Alexa, or OK Google, which are such agents. These are used on many devices including, most obviously, smartphones.

The task-oriented agents can do less in the following sense. They are directed to one task, or sometimes more, and this really limits the possibilities they have to face. For example, an airline ticket agent needs to be able to 'chat' about departures, destinations, and flight times, but it does not need to be able to chit-chat about the novels of Toni Morrison, last night's baseball game, and how to cure a shank at golf.

There is a suggestion that chatbots may partially replace browsing of certain kinds of web pages. For example, a library home page may have a lot of information on it about the various services or resources that the library offers, navigation through this may be eased by assistance from a chatbot.

2.2 Dialog Processing

Among the hallmarks of natural dialogs are unexpected and seemingly unpredictable sequences of events. (Bobrow et al. 1977)

It is valuable to have some appreciation of what a dialog is, and how difficult it is model it using software and Natural Language Processing (NLP). We will make simplifications here.

A dialog is a communication that takes place between two or more participants. For us, this can be two participants: one of which is a human, and the other a computer (or software system). At the beginning of the dialog, one of the participants has the *initiative* or *control*. This means that they set the initial agenda, assumptions, and common understandings that are required for the dialog to get underway. There are different conversational settings, as examples, talking about Xmas gifts for children has different assumptions to that of discussing the best recipe for a bundt cake, or to that of booking an airline ticket. Of course, a single conversation can range over many settings. As a dialog progresses, it may be that the agent with the initiative retains it throughout. An example here would that

of a commercial telephone answering and routing system. The caller can make choices among the options on offer, but the caller cannot ask questions or offer choices to the system. More usual are dialogs of *mixed initiative*, where control is sometimes with one agent and sometimes with the other. This usually would be the case with airline ticket reservation systems. The participants can take turns at having the initiative, but also one can retain the initiative through several interactions (for example, with a sequence of yes/no questions). How the initiative might be transferred from one to the other is a complex array of possibilities. There are a number of explicit conversational practices, but also there are implicit conventions (e.g. pauses). Also, an important consideration is whether there are visual or non-verbal cues (as there might be with videoconferencing). There are indexicals, such as 'I', 'now', 'yesterday', etc. Just what these might refer on a particular occasion to is a bit of a question. The computer saying 'I' will have a different reference to the human saying 'I'. But also the indexicals might change their reference as the conversation progresses and the initiative changes. 'Three days earlier' might refer to different times at different points in the conversation. Sometimes the notion of anaphora is introduced here. Anaphora is where the interpretation or reference of one expression depends on another; for example, in 'John Smith is conscientious. He often works evenings.' the word 'he' is anaphoric for 'John Smith'. The computer can avoid anaphora (perhaps at the cost of having a somewhat stilted and unusual conversational style) by always making the references explicit (e.g. saying 'John Smith' instead of 'he'). However, there is no controlling what the human might do.

Further, there is conversational implicature (studied extensively by the philosopher H.P. Grice) (Davis 2019). This is where speaker meaning is different to sentence meaning (and similar related phenomena). Consider an example from (Davis 2019):

Alan: Are you going to Paul's party?
Barb: I have to work.

The meaning of the sentence that Barb says is that she has to work. The sentence *itself* does not have any kind of meaning like 'not going to a party'. However, her use of this sentence on this occasion means she is not going to Paul's party. There is a conversational implicature to that effect. So, for example, if a chatbot is collecting RSVPs to Paul's party, it needs to understand 'I have to work' as meaning 'I am not coming'.

There are sentence fragments. For example, answers to questions might be single words or phrases, and grammatically they might be almost anything (nouns, adjectives, verbs, noun phrases etc.) From a logical point of view, we would expect a linguistic component to be either a reference to an item or be a true or false statement about items referred to. But extracting what is what from a fragment might require looking back through to dialog to earlier questions or context. (Bobrow et al. 1977) have the example:

Computer: In what name should I make the reservation, please?
Client: Dan Bobrow

Now the 'Dan Bobrow' is a reference to a person, and not a sentence or statement, but what is happening here is that the assertion 'You should

make the reservation in the name of Dan Bobrow' is being made, but to determine that the question and earlier context are needed.

In sum, dialogs are not easy to process.

2.3 ELIZA to ALICE

ELIZA was introduced by Joseph Weizenbaum in the paper *ELIZA—a computer program for the study of natural language communication between man and machine* (Weizenbaum 1966). ELIZA is the first chatbot. It is an early, indeed very early, AI program. However, it is not a ML program. As we will see shortly, it works by rules and pattern substitution. We have an historical interest in ELIZA. Also, its way of manipulating strings, playing with form not meaning, is something of style common in modern machine learned natural language processing. Here is a transcript of a typical conversation with ELIZA:

Men are all alike.
IN WHAT WAY
They're always bugging us about something or other.
CAN YOU THINK OF A SPECIFIC EXAMPLE
Well, my boyfriend made me come here.
YOUR BOYFRIEND MADE YOU COME HERE
He says I'm depressed much of the time.
I AM SORRY TO HEAR YOU ARE DEPRESSED
It's true I am unhappy.
DO YOU THINK COMING HERE WILL HELP YOU NOT TO BE
UNHAPPY

<and so on. The human correspondent's utterances are shown in lower-case, ELIZA's replies in upper-case. This extract is from (Weizenbaum 1966). >

As Weizenbaum tells us, the conversation style here intended to be between a patient and a Rogerian psychotherapist. There is an important sleight of hand in this. A Rogerian psychotherapist feeds, or replies, back some variation of what the patient has said. The psychotherapist does not say anything that requires knowledge of the world. This matters because ELIZA does not then need any knowledge. So, this might be a conversational fragment:

I went for a long boat ride
TELL ME ABOUT BOATS

But a request for knowledge would meet with avoidance, perhaps:

What is a boat?
HAVE YOU ALWAYS BEEN INTERESTED IN BOATS?

The basic computational procedure for ELIZA is simple. The input text is inspected for one or more matches to patterns. A pattern here usually amounts to a string, where some parts are literals and other parts are variables. So, for example, if we use (?X), (?Y), etc. as variables, then

I feel (?X)

is a pattern (the 'I feel ' part is a literal, and the '(?X)' is a variable). The matching process takes some input text and a pattern, or patterns, and sees whether they can be matched. For example:

'I feel (?X)' can be matched with 'I feel sad' (or, indeed, 'I feel sick', 'I feel happy', 'I feel cold' etc.) but not matched with 'I read novels'

Then there are rules which dictate how the matched results are to be transformed. For example:

Rule #3: 'I feel (?X)' is to be transformed to 'DO YOU OFTEN FEEL (?X)' where, of course, the value for (?X) is substituted back in.

Usually, several different patterns will match the input text. Then each of those patterns might have several different transformation rules that apply. What happens here is that there is a randomizer which makes a choice of what to use. The reason for this randomization is that it allows ELIZA to produce different responses to the same input text. So, each conversation, or session, will usually be slightly different to earlier ones. At this point, ELIZA can be improved by adding some memory of the conversation or correspondent, being more sophisticated with the responses, and so forth. There are ELIZA like chatbots alive and well today— most notably ALICE (Artificial Linguistic Internet Computer Entity) (Wikipedia 2022c).

There are two important points to note about ELIZA systems. What they do is trivial substitution of one piece of text for another. Merely by looking at the source code and a transcript for a session you can see which patterns and rules were used, step by step. The inner workings are crystal clear. There is no mystery here, nor any intelligence. Nevertheless, a good proportion of folk think that ELIZA systems are intelligent, or even that, when talking with ELIZA, that they are conversing with a human.

Weizenbaum noted this in his original paper, and in 1972 the ELIZA system PARRY passed the Turing Test (i.e. fooled some people into thinking it was human) ((Weizenbaum 1966; Colby et al. 1972)

2.4 The Turing Test

Are you wondering what the Turing Test is? Here is ChatGPT's explanation of it (provided on 12/6/2022, while ChatGPT was available as a research preview):

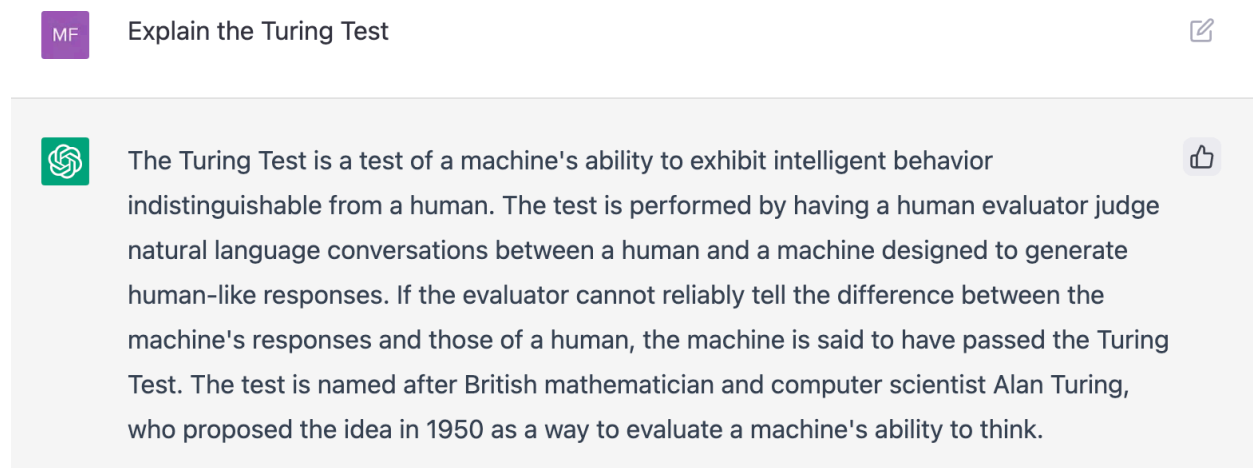


Figure 1. Explanation of Turing Test.

2.5 Machine Learning Chit-Chat Bots

ML systems can learn to chit-chat. They do need data of conversations, and a large quantity of it. There are databases, or datasets, of conversations (e.g. of telephone conversations). Then dialogs from movies are a good source. Also, sometimes, crowdsourcing is used (people either volunteer or are paid to converse in an interesting and potentially useful way). Roughly speaking,

the systems can either retrieve or generate. Retrieval systems will look through the conversations that they are aware of and select fragments similar to the one they are addressing, make a choice or ranking among the fragments, then return a suitable response. Generative systems can 'generate' a response. They learn from their training (and subsequent conversations) what might be suitable in which circumstances. In a sense, they are learning rules. But there are not a fixed number of rules programmed in before the system starts learning. The responses that generative systems produce can be novel or new.

2.6 LaMDA

LaMDA, from Google, is a family of Language Models for Dialog Applications (Thoppilan et al. 2022). These models are for Chit-chat bots. We will briefly describe here what they were designed for and what they can do, and in Chapter 3 we will go into a little more technical detail.

LaMDA agents should be able to supply conversational quality in dialogs lasting 20 turns or more. What they say should be safe. It should be safe, period. Some earlier non-LaMDA dialog agents have suggested that their conversational partners eat ground glass as a treatment for a medical condition. This is not a good or desirable outcome. Additionally, the dialog agent should not offer any views that are biased or toxic. Finally, the conversational contribution needs to be grounded. One might take grounded to mean that the views expressed be true and supported by evidence. But that condition is too strong in as much as there are many areas where there are competing rival views and where all the views have

some reasonableness to them. Grounded is just the requirement that if the agent professes opinions offered as factual then those opinions need not to be fictional or hallucinatory. (Historically, conversational agents have been known occasionally to offer plausible assertions which are pure fantasy.)

2.7 ChatGPT

ChatGPT was produced by OpenAI. ChatGPT itself tells us about OpenAI:

OpenAI is a research institute that was founded in 2015 by a group of entrepreneurs, researchers, and investors who were interested in advancing the field of artificial intelligence. The organization was created with the goal of promoting and developing friendly AI, which is AI that is aligned with human values and capable of working towards the benefit of humanity. Since its founding, OpenAI has become one of the leading organizations in the field of AI research, and has made significant contributions to the development of ML algorithms and technologies. Some of the notable achievements of OpenAI include the development of the GPT-3 language model, which is one of the most advanced natural language processing systems in the world, and the creation of the DALL-E artificial intelligence system, which is capable of generating images from text descriptions. Overall, OpenAI has played a major role in advancing the field of AI and in fostering the development of safe and beneficial AI technologies (ChatGPT output to the question 'What is the history of OpenAI?' 12/15/2022).

ChatGPT is a Large Language Model (LLM), or a Foundation Model. Quite what those features are will be explained in the next chapters. The LLM chatbots are so much in advance of all the other earlier chatbots that the earlier ones are mostly rendered obsolete and of historical interest only. As of November 2022, ChatGPT was a chat front end to GPT-3.5 (OpenAI

2022a). GPT-3.5 itself is a Generative, Pre-trained, Transformer, iteration 3.5. 'Generative' here means that it can predict, or generate, new output. 'Pre-trained' means that it has a two-stage training. First a pre-training, and then a specific shaping to the final product. Finally, 'Transformer' is the type of machine learning system. GPT-3.5 was current as of December 2022. GPT-4.0 was released in March 2023 (and likely new versions will be arriving).

Progress, or change, in the research area of LLM chatbots, is extremely rapid. Anything written here right now will be out of date in days or weeks. Please be aware of that. We know what LLMs are, how they are produced, and how they work, to a degree. That can be described. There is another caution besides the changing landscape. The large language models are so complex that there is no real description, explanation, or analysis of the details of their internal workings. Likely this will not change anytime soon.

ChatGPT (using GPT 3.5) can:

- Write high school to college level essays, to about an A standard. The written English is definitely of A standard, and usually the factual content (if that is required or appropriate) is also of A quality. This means that it can write essays for students, articles for journalists, and sometimes, or almost, write research papers for researchers.
- Tell jokes, write poetry, and produce cooking recipes.
- Write, or correct computer programs. This also is carried out to a high standard. This makes it a valuable programming, or debugging, tool. (OpenAI and Github had earlier developed Copilot, which is a

programmer's assistant (Github 2022). Copilot's abilities seem to be built in to ChatGPT).

As part of chatting ChatGPT can:

... answer follow-up questions, admit its mistakes, challenge incorrect premises, and reject inappropriate requests (OpenAI 2022a).

It does have some weaknesses (which we will look at further in the next Chapter). But briefly:

- It is not good on basic mathematics and arithmetic.
- Its knowledge is not current. (Its training data is from pre-2020. This means that it has no knowledge, for example, of recent politics or sporting events.)
- Sometimes it can go completely wrong. For example, some Users have reported of occasions when ChatGPT has referred to books that do not exist.

The weaknesses that it has have been fully acknowledged by OpenAI (and many of them will be addressed in short order).

Nevertheless, just what it can do is astonishing. It can produce, or generate, grammatical correct and sophisticated English that in most cases could not be distinguished from that written by an educated native speaker. It also has, or appears to have, extensive factual knowledge. There is an example

above (Section 2.4 on the Turing Test). There are a plethora of examples of ChatGPT output available online. (There were one million registered Users within a couple of days of its research release on 11/30/2022, and almost all of those published their experiences on social media, discussion sites (like Reddit), or in online journals or newspapers.) This research release does not reveal everything about its capabilities. For example, it seems to have a problem with truth. It writes plausible English but seems to do that without knowing whether what it is saying is true or has evidence in its favor. Contrast this with the Google Search engine. If you asked that about the Turing test, it would return a ranked list of links. ChatGPT seems much better here, just from a User interface point of view. But if you wanted to reassure yourself, when using Google Search, you could follow some or all of the links and get a sense of the foundations or grounding of what the contents of the links were asserting. With ChatGPT, truth, fiction, and well-formed writing, seem all to run into one. On tricky topics, or more complicated concepts, ChatGPT sometimes gave highly plausible answers that are flat-out wrong — something its creators warn about in their disclaimers. It may be, so-to-speak, that ChatGPT knows why it says what it does, but we are not fully aware of that at this point. It has been said '*fiction* is what ChatGPT is really good at'.

2.8 Task-Oriented

Most task-oriented dialog agents (including Siri, Cortana, Alexa, and OK Google) use Genial Understander System (GUS) architecture (Bobrow et al. 1977; Daniel Jurafsky and Martin 2021).

GUS (Genial Understander System) is intended to engage a sympathetic and highly cooperative human in an English dialog, directed towards a specific goal within a very restricted domain of discourse....

There is good reason for restricting the domain of discourse for a computer system which is to engage in an English dialog. Specializing the subject matter that the system can talk about permits it to achieve some measure of realism without encompassing all the possibilities of human knowledge or of the English language. It also provides the user with specific motivation for participating in the conversation, thus narrowing the range of expectations that GUS must have about the user's purposes. A system restricted in this way will be more able to guide the conversation within the boundaries of its competence. (Bobrow et al. 1977)

This GUS architecture is not itself a ML architecture, rather it is an older AI frame-and-slot design. The implementations do use natural language, English, say, and so the modern dialog agents will use recent NLP machine learned techniques.

A frame is a mini-world, or situation, or ontology, that captures everything that either is known or needs to be known about a certain set up. For example, consider on online take-out system for a restaurant. There will need to be a menu (with appetizers, entrées, deserts, etc.), a recipient's name address, payment details, and so forth. These constitute the frame (or perhaps even several frames), and the frame has slots (one for the appetizers, one for the credit card number, ...). The slots will have types. For example, the purchaser's name is a string (e.g. 'John Smith') and the credit card number is a number (e.g. 1234 5678 9100 1234). What GUS will aim to do is to be genial while filling all the requisite slots with values of the

required type. In principle, it really is not a lot more difficult than form filling. After all, you can order take-out from a form on a web page. Adding a voice interface makes the process harder.

Different computer programmers may well organize the algorithms for a GUS architecture in different ways. We can sketch here the general approach of the 1977 original. The program will try to retain the dialog initiative as much as possible. There will be an agenda loop of events or tasks to be done. This pattern is standard event-loop (or real time) processing. The program will keep going around the loop until it meets an exit condition. Tasks can be added to, or removed from, the agenda. Tasks can be suspended, to be revisited later. There will be an overall frame with slots, and possibly many subsidiary frames with their slots. The event-loop of tasks will aim to fill all the slots that need filling. (Some slots may be able to be left open in certain circumstances.) Many of the slots will have their own sub-programs, which will be invoked when the slot is filled. (For example, two slots might need the same value (e.g. a departure city in a reservation system). Then each of these slots might have small program to put a value for the departure city into the other slot, if required, when its own value is filled.) Verbal or textual input will be able to be taken at any time (and added to the processing loop). Output (i.e. responses) will be given at the program's direction. The actual processing of the dialog will be where the real difficulties lie. There will be morphological processing (to pick up the words), syntactical processing (to pick up the sentences), and a good amount more to make sense of the vagaries of dialog and to fit the results into the overall state of the system. Also, the filling of the slots will

itself often require interfacing with one or more databases (for example, to look up the departure times of the trains or airplanes).

2.9 GPTs

On 11/6/2023, OpenAI announced their initiative that provides an infrastructure for users to create and use what they call ‘GPTs’ (Altman 2023; OpenAI 2023d). These are customized variants of ChatGPT which can follow instructions in a natural language, access documents outside of their training (such as pdfs and research papers), and use third party services (such as search engines).

GPTs are extremely easy to create. The process being carried out in natural language. Here is one produced in less than 5 minutes:

Preview



Scholarly Search Assistant

I help find research articles.

How do I find articles on?

Suggest databases for

Help refine my search for

Find sources about



Message Scholarly Search
Assistant...



Figure 2. The Scholarly Search Assistant, an Example of a GPT.



Name

Scholarly Search Assistant

Description

I help find research articles.

Instructions

Scholarly Search Assistant will engage users in a conversational tone, making the search for machine learning and librarianship articles more approachable. It will provide guidance and suggestions using everyday language, avoiding jargon unless it's part of the user's query. The GPT will use its own judgment to provide general answers when requests are vague, ensuring users receive helpful information even from broad inquiries.

Knowledge

Upload files

Capabilities

- ☒ Web Browsing
- ☒ DALL·E Image Generation
- ☒ Code Interpreter ?

Actions

Add actions

Figure 3. The Configuration of the ‘Scholarly Search Assistant’.

At this point, with GPTs, the production of many chatbots is nearly trivial. Quite how they work is another question. By the end of Chapter 5 on Large Multimodal Models, we should have some reasonable insights on that.

GPT-4 Turbo, released November 2023, has processing images and voice synthesis built in (i.e. it is multimodal). The GPTs are going to use GPT-4 Turbo as their computing engine. Shortly the chatting will be able to be by text or by audio.

2.10 Annotated Readings for Chapter 2

Altman, Sam, dir. 2023. *OpenAI DevDay, Opening Keynote*.

<https://www.youtube.com/watch?v=U9mJuUkhUzk>. (Altman 2023)

c.ai. “Character.Ai.” character.ai. Accessed July 3, 2023. <https://beta.character.ai/>. (c.ai 2023) This allows you to chat with chatbots that either you or others have created.

Fernandez, Peter. ““Through the Looking Glass: Envisioning New Library Technologies’ AI-Text Generators as Explained by ChatGPT.” *Library Hi Tech News* 40, no. 3 (2023): 11–14. <https://doi.org/10.1108/LHTN-02-2023-0017>. (Fernandez 2023)

G2. “Best Bot Platforms Software.” G2, 2023. <https://www.g2.com/categories/bot-platforms>. (G2 2023) . Just scan this.

Jurafsky, Daniel, and James H. Martin. “Chapter24 Chatbots & Dialogue Systems.” In *Speech and Language Processing*. Daniel Jurafsky & James H. Martin. Draft of December 29, 2021., 2021. <https://web.stanford.edu/~jurafsky/slp3/24.pdf>. This is the standard text. It is pitched at an intellectual level in advance of that which we are using here.

Library Hi Tech News. “Special Issue on ChatGPT.” *Library Hi Tech News*. 40, no. 3 (2023). This has a number of useful articles on the uses of ChatGPT, especially in a library setting. (Library Hi Tech News 2023)

OpenAI. “Introducing GPTs,” 2023. <https://openai.com/blog/introducing-gpts>. (OpenAI 2023d) You can try creating GPTs yourself at chat.openai.com/create.

Schlicht, Matt, and Ben Parr. “*Chatbots Magazine: The #1 place to learn about chatbots.*” Chatbots Magazine, 2023. <https://chatbotsmagazine.com/>. This is a good resource, it is a collection of short articles by a variety of authors. (Schlicht and Parr 2023)

Weizenbaum, Joseph. “ELIZA—a Computer Program for the Study of Natural Language Communication between Man and Machine.” *Communications of the ACM* 9, no. 1 (1966): 36–45. <https://doi.org/10.1145/365153.365168>. This is an important original paper. It is very readable.

Wolfe, Matt. “Future Tools - Find The Exact AI Tool For Your Needs,” 2023. <https://www.futuretools.io/>. (Wolfe 2023). This is an astonishing and valuable resource. Especially recommended are the News and Videos sections. The content is not really what one might call ‘academic’. Rather it focusses on what the commercial companies and start-ups are producing. The content slightly favors work with images and video, whereas our interest is a little more with text and information.

Chapter 3: Language Models

3.1 Introduction

Many of the modern AI systems have their origins with language models. These typically use text as input and produce text as output. Language models have a huge advantage, when considered as a ML research challenge. They can be trained on (unlabeled) text, and there is plenty of that (for example, the Internet). Elsewhere, any ML program, trained by supervised learning, will need a quantity of high-quality labeled data, and that is hard to come by. But most language models will be able to be trained by self-supervision using digitized text.

Emily Bender and her co-authors explain language models as:

... the term language model (LM) ... [refers] to systems which are trained on string prediction tasks: that is, predicting the likelihood of a token (character, word or string) given either its preceding context or (in bidirectional and masked LMs) its surrounding context. Such systems are unsupervised and when deployed, take a text as input, commonly outputting scores or string predictions. (Bender et al. 2021)

What does this mean? Some background theory will be useful here.

3.2 Markov Chains

Alexander Pushkin, the eminent Russian poet, playwright, and novelist, writes:

My uncle's goodness is extreme,
If seriously he hath disease;
He hath acquired the world's esteem
And nothing more important sees;
A paragon of virtue he!
But what a nuisance it will be,
Chained to his bedside night and day
Without a chance to slip away.
Ye need dissimulation base
A dying man with art to soothe,
Beneath his head the pillow smooth,
And physic bring with mournful face,
To sigh and meditate alone:
When will the devil take his own! (Pushkin 1881)

This is from an English translation of *Eugene Onegin: A Romance of Russian Life in Verse* written in Russian.

There is an interesting tidbit of information associated with this. Until about 1900, probabilistic analyses of common sequences of apparently random events assumed that the individual events were independent one from another. So, as examples, that the second and subsequent throws of a single gambling die are not influenced in any way by the earlier throws, and that the outcome of a spin of a roulette wheel does not depend on the spin, or spins, that went before. The Russian mathematician Andrei Andreevich Markov questioned whether this independence assumption held of all

sequences, especially in fields outside of gambling. He conjectured that it did not and produced a suitable mathematical theory to cover the case.

The basic theory is relatively easy to understand. Almost everybody is familiar with the centuries old board game Snakes and Ladders. In this, there is a board with a hundred and one numbered squares on it, and players advance from the start (i.e. 0) to 100 by repeatedly throwing dice and using the numbers they obtain. The first player to 100 wins. On the board there are some 'snakes' and some 'ladders'. These connect pairs of numbers, usually separated by 5 to 20 intervening numbers. The heads of the individual snakes are higher— of a higher number— than the tails. If a player lands on the head of a snake, they have to go backwards to the tails, losing some of their advance. If a player lands on the base of a ladder, they advance to the top of that ladder. Consider just one player, and one die, and assume the player throws a 5. This may advance the player 5 spaces, but it also may send the player back some spaces or it may send the player forward more spaces than 5. What happens depends in part on what square the player is on. Throwing a 5 when on square 15 might produce an entirely different result, in terms of going forward or backward, than throwing a 5 when on square 32. The square that the player is on is the *state*. The throw of the single die will produce one of six numbers. This will move the player to one of six other states. The values of the throws (in this case) are equally probable, and so there are six equally probable non-zero *transition probabilities* from the current state to the next state. What happens with a throw depends only on the current state (and the value of the throw). It does not depend on the state before the current state, or any before that, i.e. on the history of the individual game. For example, if a player is on square

27, state 27, that is all that matters— it does not matter how the player got to 27. Snakes and Ladders play is an example of a Markov Chain. Conceptually, there is usually time, or a time step, or time beats, involved in producing the sequence of states in a Markov Chain. If we wanted to build that into the example, we could just require that the player throws the die every 30 seconds. There are a few other details that can be added to the Snakes and Ladders example to make it closer to Markov Chain theory. There is usually a start state, or a probability among the states for one being the start state. So, the zero square would have a probability of 1 of being the start state (and the other states 1-100 probability zero). Similarly, there might be a terminal state. If so, there would be no transition probabilities for a move out of the 100 square. Also, it is not possible to rest on the foot of a ladder or the head of a snake (because the player would be required to move on elsewhere). This could be represented either by having no transition probabilities that end with the foot of a ladder, or the head of a snake, and including the jumps in with the dice values, or just by omitting those 'states' altogether. Summing up Markov Chains, there is a sequence of events, states, transition probabilities, and the dependence on the state is just with the current state. This latter is the key feature of Markov Chains, the dependency is with a single state only, the current state, and not with earlier states. (Sometimes this is called the Markov Property.)

What did Markov do with *Eugene Onegin*? He divided the first 20,000 characters of the text into consonants and vowels (in Russian, of course). This gave him a value for the *overall probability* of a particular character being a consonant (or being a vowel). He then looked at pairs of consecutive characters and what the probabilities were for the *second*

character being a consonant (or a vowel) if the first character was a vowel (or a consonant). These are transition probabilities from vowels to consonants, and consonants to vowels. What he found was that vowels were more likely to be followed by consonants (and consonants by vowels)— that the transition probabilities were different to the overall probabilities. What this meant was that if the text was thought of as a sequence or stream of letters, the appearance of one letter after another was not an independent event, being governed by only the overall probabilities of the letters, rather it was a dependent event having a dependence on the current letter or state. From here, there is the theory and mathematics of Markov Chains.

Markov Chains are everywhere. They are in text. Not just with consonants and vowels, but also with letter sequences like 'u' following 'q', and word sequences like the word 'He' being followed by a verb phrase. They are in spoken text, or speech, with sequences of phonemes (certain sounds follow others). They are in the weather (rainy days tend to follow other rainy days). Many games— the sequences of their moves— can be analyzed by Markov Chains (e.g. chess). The Stock Market can be seen as a Markov Chain (bull markets tend to be followed by bull markets, and bear markets followed by bear markets). There are many Markov Chains in biology, for example with DNA sequences. More-or-less any values or measurements that change over time, such as prices in a stock market, speeds of motor cars, etc., can be modeled by Markov Chains. [There is a paper *Five greatest applications of Markov Chains* (Von Hilgers and Langville 2006). It is recommended.]

3.3 Hidden Markov Models

Hidden Markov Models (HMMs) are extremely common in the ML analysis of sequential data that has a dependence on time. Typically, a model will be able to explain a sequence and predict or generate new sequences. The techniques are most at home with Natural Language Processing (NLP) and with handwriting recognition, speech recognition, and biological informatics.

With an HMM there are two collections of probabilities. In the background, there is a probabilistic Markov Chain or Process as explained above. But some, or all, of the states of this Markov Process are hidden (and unseeable directly by observers). However, the hidden states 'emit' observations which can be seen or observed. These emissions of observations are themselves governed by probabilities relating them to the hidden states. An example might help with understanding the set-up. Suppose we are interested in the sequences of year-on-year climate values in New Zealand from a thousand years ago. (We may be interested in this because the climate affects the environment, its flora and fauna, and their history and development.) Suppose the years can be just hot or cold, and which they are for a particular year depends solely on whether the predecessor year is hot (or is cold). Suppose also that the probabilities of a hot year being followed by another hot year (and a cold year being followed by a cold year) are all known from modern values (which are assumed to be unchanged from those of a thousand years ago). These ancient weather sequences are Markov Processes, and their states are hidden and not directly observable. However, the weather affects tree-ring growth, and we have modern

probabilities relating that growth to the weather (which, again, are assumed to be unchanged from a thousand years ago). More than a few New Zealand Kauri trees live longer than a thousand years. So, the Kauri rings can serve as modern emitted observations on the Hidden Markov Process of ancient weather sequences in New Zealand (Stamp 2017; Rabiner 1989).

We will not address the mathematics in detail here. But what the mathematics can do is i) estimate the probabilities for a hidden sequence, given the observable sequence, and the other relevant probabilities ii) estimate the probabilities for an observable sequence, given the hidden sequence, and the other relevant probabilities, iii) given an observable sequence, the number of hidden model states and different types of observations, can use successive approximation on the parameters to produce values which give the greatest probability of yielding the provided sequence. (The successive approximation uses what is known as 'gradient descent', see the excellent videos from Josh Starmer or Grant Sanderson for explanations of gradient descent (Starmer and StatQuest 2019; Sanderson and 3Blue1Brown 2017b). Gradient descent is a standard ML technique.)

As mentioned, Natural Language Processing often uses HMMs. Mark Stamp, in his (Stamp 2017), gives a simple example of language analysis, which is more-or-less the inverse of Markov's *Eugene Onegin* case. Say you gave 'Marvin the Martian' (an alien) a corpus of a million words of actual English text and invited him/her/them to investigate sets of individual characters under the assumption that there was a HMM, with two hidden states, that was producing the observed English. The result of Marvin's research would be that there were two relevant sets of letters, and they

would be the vowels and the consonants. Initially, Marvin knows nothing about English, yet Marvin can learn structural and statistical properties using an HMM.

Marvin, and Markov, in the *Onegin* example, are looking at individual characters or letters. But what is much more useful and widespread in the general NLP case is analyzing entire words, or parts of words, and, for example, tagging them with their parts of speech (POS). In this, HMMs are often used to identify nouns, verbs, adjectives, determiners, etc. Parts of speech point to the function of the words in the sentences i.e. they hint at role or meaning.

3.4 Shannon's Guessing Game

3.4.1 Introduction

An interesting precursor of modern NLP research is the 1940s work of Claude Shannon on signal information (C E Shannon 1948). Shannon provided an extensive analysis of the natural language English and it is worthwhile to follow and develop his reasoning. He was trying to calculate how much entropy (signal information) there was in passages of English. (What entropy and signal information are need not concern us.) The approach here was to use successive approximation by models. Shannon's zero-order model assumed that the alphabetic letters or characters had an equal probability of occurring in English text. But in real English, assuming here that the speaker or writer is not idiosyncratic, the occurrence of letters is not equiprobable; 'e', for example, has a probability of 0.13 of occurring and 'w' a probability of 0.02. If the differing probabilities for all the letters

are taken into account, that produces what is known as the first-order model. There are also conditional, or transition, probabilities relating letters in letter sequences in English; 'q', for example, is almost invariably followed by 'u'. This model is now using a Markov process for analysis. Two letter sequences, giving the second-order model, can be considered, or three letter sequences and so on. A similar approach can be adopted to predicting a missing word or the next word, using whole words as the alphabet and not just single letters. (Filling in the blanks in some text is known as a 'cloze task'. These will be explained in more detail shortly).

The assumption of non-idiosyncrasy of the source English text is entirely in order— John Pierce tells us of E.V. Wright who published the 267 page novel *Gadsby* in which there are no 'e's (Pierce 1980) p.48). There is an important point to be made here. When there is focus on the probabilities of occurrences of letters in text, say 'e's, there is a need to address context. Such probabilities may well have different values depending on, for example, whether the text is *Gadsby*, the *Sunday Times*, or every piece of English on the Internet. Care is also needed here with the relation between the training data and the intended real use data (the test data) for ML language probabilities. You would not want to use *Gadsby* as the sole training data for all of written English.

Shannon conducted extensive experiments on English. One technique he used was to give test subjects many letters of a sequence of English—usually up to about 50 letters—and then ask them to guess the next unknown letter (a cloze task); he found that the subjects were able to guess correctly about half the time or more. (You can try Shannon's experiment or game :- pick

roughly the middle letter of any line in this book, count forward fifty letters, keeping the fifty first letter covered before making your guess, and see how uncertain you are about the value of the fifty first letter.)

The results show that English is massively redundant. This redundancy has its advantages. Natural languages are highly noise tolerant— often many letters in a message can go missing or be mis-transcribed and yet still the receiver will be able reconstruct the original. (And also, yet again as Shannon pointed out, without redundancy, crossword puzzles would be impossible.)

English is usually taken to be an ergodic source (roughly, that means that one reasonable passage is a good representative of the whole, as far as all the probabilities are concerned). However, so-called Large Language Models and Foundation Models would typically use vast amounts of text as input data (see, for example, Common Crawl (Common Crawl 2022)). There is a good reason for that. As we will see these models are trying to do more than evaluate probabilities; they are, for example, also trying to acquire knowledge (such as, the fact that George Orwell is the author of the book *1984*).

3.4.2 Shannon's Approximations as Markov Processes

[An indicative fact: Shannon's seminal 1948 paper originally had the title *A Mathematical Theory of Communication*. In 1949, when reprinted in

(Claude Elwood Shannon and Weaver 1949), it was renamed *The Mathematical Theory of Communication*. Enough said.]

To explain the GPT Large Language Models, there is a passage in the Shannon paper which is pure gold:

To give a visual idea of how this series of processes approaches a language, typical sequences in the approximations to English have been constructed and are given below. In all cases we have assumed a 27-symbol “alphabet,” the 26 letters and a space.

1. Zero-order approximation (symbols independent and equiprobable).

XFOML RXKHRJFFJUJ ZLPWCFWKCYJ FFJEYVKCQSGHYD
QPAAMKBZAACIBZL- HJQD.

2. First-order approximation (symbols independent but with frequencies of English text).

OCRO HLI RGWR NMIELWIS EU LL NBNESEBYA TH EEI
ALHENHTTPA OOBTTVA NAH BRL.

3. Second-order approximation (digram structure as in English).

ON IE ANTSOUTINYS ARE T INCTORE ST BE S DEAMY ACHIN
D ILONASIVE TU- COOWE AT TEASONARE FUSO TIZIN ANDY
TOBE SEACE CTISBE.

4. Third-order approximation (trigram structure as in English).

IN NO IST LAT WHEY CRATICT FROURE BIRS GROCID
PONDENOME OF DEMONS- TURES OF THE REPTAGIN IS
REGOACTIONA OF CRE.

5. First-order word approximation. Rather than continue with tetragram, ..., n -gram structure it is easier and better to jump at this point to word units. Here words are chosen independently but with their appropriate frequencies.

REPRESENTING AND SPEEDILY IS AN GOOD APT OR COME
CAN DIFFERENT NATURAL HERE HE THE A IN CAME THE
TO OF TO EXPERT GRAY COME TO FURNISHES THE LINE
MESSAGE HAD BE THESE.

6. Second-order word approximation. The word transition probabilities are correct but no further structure is included.

THE HEAD AND IN FRONTAL ATTACK ON AN ENGLISH
WRITER THAT THE CHARACTER OF THIS POINT IS

THEREFORE ANOTHER METHOD FOR THE LETTERS THAT THE TIME OF WHO EVER TOLD THE PROBLEM FOR AN UNEXPECTED.

The resemblance to ordinary English text increases quite noticeably at each of the above steps. Note that these samples have reasonably good structure out to about twice the range that is taken into account in their construction. Thus in (3) the statistical process insures reasonable text for two-letter sequences, but four-letter sequences from the sample can usually be fitted into good sentences. In (6) sequences of four or more words can easily be placed in sentences without unusual or strained constructions. (C E Shannon 1948, 7)

Shannon starts here with equiprobable letters, the zero-order approximation, then moves to allow the letters to have the probabilities that they do in real English. Then, with the second-order approximation, he allows the letters to have transition probabilities— probabilities dependent on the single preceding letter only. So, if the preceding letter were 'Q', for example, the probability of the next letter being 'Z' would be zero, the probability of it being 'U' would be high, the probability of the next letter being a blank space would have some positive value, and so on. At this point what is going on is a Markov process. Shannon tells us this:

Stochastic processes of the type described above are known mathematically as discrete Markoff processes and have been extensively studied in the literature. (C E Shannon 1948, 8)

The preceding letter in the second-order approximation is a *state* or *context* and the relevant probability depends on that state. The third-order approximation uses transition probabilities, but this time extends the state or context to the two preceding letters. At this point, Shannon switches to using entire words, not individual letters, and that is not of further interest to us here. A point to note is that 1948, Shannon would have had no way of

calculating transition probabilities using three, four, five, ..., and so on preceding letters. (Basically, you need computers and digitized text to do that.) Had he been able to do so, no doubt he would have explored much larger contexts.

3.4.3 Training a Shannon-Markov Model to Produce 'A Baby GPT'

Nowadays, we can train a Shannon-Markov model to evaluate the transition probabilities for itself, using self-supervised learning. The result is what is in effect a GPT (a Generative Pre-Trained Transformer). Andrej Karpathy suggested the following as an example (Karpathy 2023a). There are some serious simplifications in this.

The alphabet consists of two tokens {A,B}

The context (or prompt) is the three preceding letters. So, there are eight states (namely AAA, AAB, ABA, ABB, etc.) There will be 16 transition probabilities from the contexts to the next token, namely

AAA->A with probability unknown
AAA->B with probability unknown
AAB->A with probability unknown
AAB->B with probability unknown
Etc.

These probabilities need to be determined from the training data, by training. [We can note, though, each context can produce either A or B and if the probability for one of those is x the probability

for the other is $(1-x)$. So, there are eight values to be pinned down.]

The training data, the training text, is the sequence of 15 tokens:

AAAABAAAABAAAAB

What our ML system is going to do is to produce a 'next token' as an output, then do that again, and again, ..., to give 15 tokens. It needs a 3 token context to start with. So, we will give it three As as its initialization or prompt

AAA

then let it run to produce the next 12 tokens. Just to explain further. The first prompt will produce an output token, say B.

AAAB

Then, the new token gets included in the next prompt and first token in the earlier prompt gets dropped out of the prompt, and, A, say, is generated.

AAABA

So there is a continuous 'adding, and dropping, and sliding forward'. The ML system sees only three symbols at a time, the ones shown in green, but those three symbols change through time.

Suppose, for the first training round, it completes its generation and produces

AAABBAABABAABAB

This output will be compared with the training data. There will be a scoring system that evaluates it. Then the machine learning model will revise its internal weights (i.e. parameters). It will

improve itself. And there will be a second training round, and a third, and the process will be repeated over and over. Karpathy has provided sample software to do this, and typically there might be 50 rounds of training. In his publication, the training determined the transition probabilities to be

AAA->A with probability 0.45
AAA->B with probability 0.55
AAB->A with probability 0.78
AAB->B with probability 0.22
Etc.

Looking from outside the system we can see what the transition probabilities should be. The context is the three previous tokens. We can slide the context window through the training text seeing what the outcome tokens are.

AAAABAAAABAAAAB *
AAAABAAAABAAAAB *
AAAABAAAABAAAAB
AAAABAAAABAAAAB !
AAAABAAAABAAAAB
AAAABAAAABAAAAB *
AAAABAAAABAAAAB *
AAAABAAAABAAAAB
AAAABAAAABAAAAB !
AAAABAAAABAAAAB
AAAABAAAABAAAAB *
AAAABAAAABAAAAB *

To take two examples from this, the context AAA appears 6 times (an asterisk picks them). Three of these are followed by A, and three followed by B. So, roughly speaking, we might expect that both:

AAA->A
AAA->B

would have a probability of near 0.5 (in fact, the published training run gave 0.45 for one and 0.55 for the other). For the second example, the context ABA appears twice, identified by an exclamation mark, and each time it is followed by an A. So, ABA->A should be 1.0 and ABA->B should be 0.0 (the published training run gave 0.78 and 0.22). There were only 50 rounds of training. If more rounds were used, the values would sharpen to the correct values.

Notice that some of the possible contexts do not appear in the training data. For example, BBB does not. So, what should happen with a BBB context is an open question. It is not determined by the training data. It is at this point that what is called *inductive bias* enters. The system needs to make a 'reasonable' guess and with the Karpathy sample implementation it goes with 0.5 chance of getting an A, and 0.5 of getting a B, for all contexts it cannot determine from the data.

At this point, we have a 'baby GPT'. It is a Shannon/Markov model with a context of 3 preceding tokens. It can generate output. Here is one example that it produced:

AAABAAABAAABAABAAAABABA

To compare the baby with a GPT like GPT-3. Our alphabet has 2 tokens, a full-blown GPT would likely use 'words' and not individual letters in its alphabet and it might have an alphabet of 50,000 different tokens (words). Our context is a context of the 3 previous tokens, a full-blown GPT might use a context window that can see the 30,000 previous tokens. As of May 2023, Anthropic's Claude model has a context window of 100,000 tokens

(which is about 70,000 words— its tokens are larger than single characters, but smaller than entire words). Our training text is 15 tokens long. GPT-4's training text is more-or-less 'the entire Internet'. We can train in milliseconds. GPT-4 takes months on parallel 'supercomputers'. The baby was trained on 50 rounds ('epochs') of training. OpenAI has not disclosed how many epochs of training it used (we asked GPT-4 this question) but it will have been in thousands.

3.5 Taylor's Cloze Procedure

In 1953, Wilson L. Taylor published the paper *"Cloze Procedure": A New Tool For Measuring Readability* (W. L. Taylor 1953). Basically, cloze procedures amount to taking some text and deleting parts of it, then inviting the human subjects (or software) to fill in the blanks to recreate the original text. Taylor writes:

Given "Chickens cackle and --- quack," almost anyone can instantly supply "ducks."

Note that the sentence pattern is a complex one made up of many sub-patterns. One must know not only the meanings (i.e., patterns of symbol-meaning relationships) and forms (patterns of letters) of all the five words, but also the meanings of given combinations of them— plus the fact that the sentence structure seems to demand a term parallel to "cackle" but associated with ducks instead of chickens. In other words, one must guess what the mutilated sentence means as a whole, then complete its pattern to fit that whole meaning. (W. L. Taylor 1953)

As the title of his paper indicates, Taylor's original idea was to use cloze procedures as one tool to assess the reading skills of subjects. But

subsequently their use has expanded far beyond this. Cloze tasks, or cloze procedures are reasonably demanding and they are a popular research technique for modern Natural Language Processing (NLP). [Nowadays, it is a common practice for researchers to write '[Mask]' to show where the blanks are. So, Taylor's illustrative sentence would be written:

Given "Chickens cackle and [Mask] quack," almost anyone can instantly supply "ducks."
]

3.6 nanoGPT and an Illustration of Training

Andrej Karpathy has provided for us nanoGPT (Karpathy [2022] 2023). This is some software you can run at home (do try it!). It is:

The simplest, fastest repository for training/finetuning medium-sized GPTs. [It can reproduce] GPT-2 (Karpathy [2022] 2023)

nanoGPT is a language model, and it can be used to illustrate training. Aatish Bhatia has done exactly that in the implementation he calls BabyGPT (Bhatia 2023). One of the examples that Bhatia provides concerns Jane Austen. To paraphrase Bhatia, BabyGPT is provided with the 800,000 words of Jane Austen's work and also a prompt:

"You must decide for yourself", said Elizabeth [We assume here that this prompt is not among the actual work of Austen, otherwise that would ruin the training, inviting what is called overfitting (here this means that it has been told an answer in its training).]

And what BabyGPT has to do, its task, is to produce the next seventy or so words written in the style of Jane Austen. The process will be for it to output about 700 characters, one at a time. It will produce one character. Then having done that, produce the second character, and so on, until it reaches about 700 characters. A blank space is a character (as are punctuation marks and upper-case characters). Thus, some character sequences will have the appearance of sequences of words (or even of phrases and sentences). There will be a scoring system that evaluates the mistakes. Learning will be used to change the parameters of BabyGPT. Then this will be done again, and again, forming round after round. In the first round it produced:

"You must decide for yourself", said Elizabeth
grThbE22]i10anZOj1A2u'T- t'wMOZeVsa.foJC1hpndrsR
6?to8j7dCVCyHwrWFYYGr"X8,IOwC!WAE_]!LtZf8&Or6d'KDiD
77Wq'Y4NtV:_'N [and more]

Here, it is producing characters randomly, of equal probability. This is Shannon's zero-order model. Were this first round to be run again, with the parameters re-set to their original values, the output would likely be different. [A character set, with special characters in it, might have a repertoire of 256 different characters. So, the probability for any particular character appearing in any particular position is $1/256$. Getting the same output sequence twice is very unlikely.]

After 250 rounds of training, it produced:

"You must decide for yourself", said Elizabeth rather repeatedly; "that is very agreeable displeasure, they will ever be a lively young woman as it will be more disagreeable." "My dear Fanny, [and more]

This is now similar to Shannon's first-order model. It has figured out that not all letters are equiprobable (for example, that special characters do not occur very often and that 'e's occur relatively frequently). After 30,000 rounds it produced:

"You must decide for yourself", said Elizabeth rather repeatedly; "that is very agreeable displeasure, they will ever be a lively young woman as it will be more disagreeable." "My dear Fanny, [and more]

Now there are words, sentences, some grammar, and a character stream approximating English. This training was carried out on a laptop computer in an hour. Some of the large language models, which will get to shortly, might use what effectively is a super-computer for months for their training. Also, they may use a two-stage training. There might be a pre-training, which would be a self-learning training similar to the above. This then might be followed by reinforcement learning where there is a panel of about forty human judges giving feedback and a scoring as to the quality of the output.

3.7 Embeddings

In machine learning, an embedding is a way of representing data as points in n-dimensional space so that similar data points cluster together (Markowitz 2022).

A point in an n -dimensional space can be thought of as a list, or vector, of numbers. As examples, a point in a 2-dimensional space might be [2,3]; a point in a 3-dimensional space might be [1,29,2]; a point in a 7-dimensional space might be [29.1,-7,2.9,13,21.6,-37.23,9]; you get the idea. Then a measure can be put on these lists that will show whether two or more lists are 'similar'. If they are, this means, or might mean, that the underlying data, the relevant underlying data points, are similar.

Embeddings are important.

...embeddings power:

1. Recommendation systems (i.e. Netflix-style if-you-like-these-movies-you'll-like-this-one-too)
 2. All kinds of search
 - a. Text search (like Google Search)
 - b. Image search (like Google Reverse Image Search)
 3. Chatbots and question-answering systems
 4. Data preprocessing (preparing data to be fed into a machine learning model)
 5. One-shot/zero-shot learning (i.e. machine learning models that learn from almost no training data)
 6. Fraud detection/outlier detection
 7. Typo detection and all manners of “fuzzy matching”
 8. Detecting when ML models go stale (drift)
 9. So much more!
- (Markowitz 2022)

We will get to most of these applications later.

A good example of a typical use of embeddings is that provided by Cohere with its embeddings of Wikipedia articles (Reimers and Alammari 2023; Kayid and Reimers 2022). What this does is to break millions of Wikipedia articles down into several millions of passages of text. Then each passage is

assigned a vector (i.e. a list) of numbers. Conceptually, these vectors are collected together into a vector database (i.e. a library of vectors). Then, as a typical use, were a user to search for a topic, or a phrase, or a query, that search phrase would be embedded to produce its own vector. Finally, the vector database would be searched for vectors similar to the search vector and the relevant actual Wikipedia passages would be retrieved and returned. There is an initial embedding process to produce vector database, another embedding process to embed the search phrase, then a ‘de-embedding’ process to get the sought for Wikipedia texts back from the matched similar vectors.

Suitable embeddings of text will embed meaning, not the surface form of the words, phrases, and sentences. One consequence of this is that the application or software can work with multiple languages:

... rely[ing] on the property that sentences that are similar in meaning will have similar embeddings, even if they are in different languages. (Reimers and Alammr 2023)

One extremely important point about embeddings is that there is not just one way of producing an embedding. There are indefinitely many ways of doing it, usually each with their own algorithms or software. This matters because the embedding algorithm used to produce the vector database and the embedding algorithm used to embed the search phrase need to be the same. Care is needed here because the database will be produced at one point in time, say 2021, yet people might still be searching it 5 or 10 years later. Typically, developers outside of the largest companies will not create embedding algorithms for themselves. Rather they will use algorithms from

OpenAI, Hugging Face, or other open-sourced or licensed resources. This exposes the developers to possibly catastrophic risk. If the algorithms, or their licenses or availability, are changed, the developers' software might be irretrievably broken. To give a concrete example, later in the text we use OpenAI's text-embedding-ada-002 algorithm to embed 'Stochastic Psittacosis'— were OpenAI to withdraw its provision of ada-002 that result basically would become garbage. This matters not just to developers. Purchasers of software using these technologies need to be wary of becoming dependent and 'locked in'. The risk here is real. In November 2023, OpenAI was close to imploding as a company (with the four-day sacking and re-instatement of its CEO, Sam Altman).

3.8 Word Embeddings and Word2Vec

Word embeddings are a common and useful tool or technique to process words prior to their being used in ML models. Often the embeddings can capture useful deep syntactic and semantic information. The DL Models themselves use numbers internally for their arithmetic and mathematics. This gives the preliminary task of converting words to numbers for use in the models. This coding could be done trivially by using the number 1 for the first word, 2 for the second word, and so on. But notice this. Say word number 3 was 'toy' and word number 47 was 'toys'. Those two words are similar to each (one is the plural form of the other). But information of that similarity is lost in the coding. The numbers 3 and 47 are not similar, nor do they give any hint of similarity between the words that they are coding. Tomas Mikolov, and fellow authors, following earlier work, proposed and developed the technique of using vectors (lists of numbers) to code the

words (Mikolov, Chen, et al. 2013; Mikolov, Sutskever, et al. 2013; Mikolov, Yih, and Zweig 2013; Colyer 2016; Alammam 2019). The idea here is that in a large corpus of text the word 'toys', for example, appears in various contexts (that is, surrounded by other words e.g. 'children', 'Xmas', and so forth) and it tends not to appear in other contexts (e.g. surrounded by words about the fourth quarter of the Super Bowl); then some aspects of these appearance contexts can be placed into the vector that is to represent 'toys'. Similarly for the word 'toy' and its contexts. The overlap of the contexts gives information about the words. This type of coding, exemplified in the software Word2Vec, can capture similarities and also some real information. It can open the way to answering questions like:

Paris is to France as [Mask] is to England

The famous example that Word2Vec is known for is:

King – Man + Woman = Queen

We are using a rough representation here of vectors and vector arithmetic. But what Word2Vec is telling us is: if you take the concept of King, take the Man out of it, add a Woman to it, you get the concept of Queen. Word2Vec has reached this conclusion merely by being trained on a large corpus of text.

Word2Vec is not the only way of producing word embeddings, nor is it the best for all circumstances. But, the point is, the use of intelligent word embeddings on a large corpus of text can provide to ML NLP programs

input that has some syntactic and semantic information built in (see also (Tenney et al. 2022)).

3.9 Adding Knowledge to Language Models

In the case of natural languages, language models, in their original and simpler forms, had to manage words, parts of speech, grammar and so forth. They did not address knowledge or truth or meaning or reference. So, for example, if such a model produced, for some purpose, some entirely correct syntactical and grammatical output such as 'The Mississippi is the longest river in the world and it is 30 miles long' that would be perfectly good and the fact that what is being asserted is mistaken is beside the point. [This has consequences for the training data. The data needs to be good sample English, say, with all its varying genres, styles, and diversity. But it does not need to be vast and encyclopedic in its content. It does not need, for example, to contain simple, if obscure, everyday facts like the name of the Secretary of the Teamsters Local Union LU No 2. (For those with an enquiring mind, that name is 'Erin Foley'.)]

But we have already seen value in many settings to add knowledge to ML systems. For example, a reservation chatbot, a task-oriented dialog agent, linked to a database, can book airline flights and be absolutely correct on the schedules, and prices etc. It does not take a huge stretch of the imagination to realize that many published texts contain knowledge, for example, a favorite children's book Bertha Morris Parker's *The Golden Book of Facts and Figures: A Treasury of Information on Hundreds of Subjects With More than 500 Pictures in Color* does, so do many

encyclopedias, and, indeed, Wikipedia. Most of our knowledge is in books and libraries. Presumably a language model could be configured to take on cloze challenges like:

The [Mask] river is the longest river in the world.

and answer it with 'Amazon'. Actually, many ML question answering systems can do exactly this. Further, Fabio Petroni, and his fellow authors, inform us that larger language models are developing this ability (Petroni et al. 2019). There is an interesting and important point here. Historically, the way that ML systems would interact and interface with knowledge in text is by producing or extracting from the text a database and then addressing queries to that database. But this is not easy to do, for a variety of reasons. Databases, with their tables of rows and columns, need design. They need ontologies. Different databases may well have different ontologies and be 'information silos' unable to share their knowledge one with another. Also, the training of database extractors would likely need to be *supervised* training (i.e. training that used labeled data), and once labeled data is required a project becomes much more challenging. Then the questions to a database have a structured semi-formal logical structure, and these might not sit easily with Users' expectations. But research now seems to indicate that making the language models larger and larger, and training them, *unsupervised* or *self-supervised*, on more expansive corpuses of text, gives those models question answering, and knowledge manipulation abilities that are superior to the earlier database extraction techniques (Lewis, Denoyer, and Riedel 2019). Of course, research can be revised by further research, and promising research can fade to nothing. But right now, 2023,

it looks as though making the language models larger, and trained more extensively, will yield gold.

3.10 InstructGPT and the Insights it Provides

As of January 2023, language models, and ML in general, have been well and truly being brought to the world by ChatGPT and similar astonishing ML programs (OpenAI 2022a). ChatGPT was created by OpenAI (with a billion dollars or more funding from Microsoft, and more funding from others) (OpenAI 2022c). Google and Facebook, and maybe other commercial companies or institutions, have similar programs (e.g. Bard and LLaMA). OpenAI have not to date published information on how ChatGPT works (and they may not do so in detail— they are a for-profit commercial company). However, they have said that it works the same way as InstructGPT, which is one of their earlier programs, and they have published a thorough research paper on InstructGPT (Ouyang et al. 2022). We should look at InstructGPT.

Long Ouyang and his fellow authors provide the following abstract to their paper:

Making language models bigger does not inherently make them better at following a user's intent. For example, large language models can generate outputs that are untruthful, toxic, or simply not helpful to the user. In other words, these models are not aligned with their users. In this paper, we show an avenue for aligning language models with user intent on a wide range of tasks by fine-tuning with human feedback. Starting with a set of labeler-written prompts and prompts submitted through the OpenAI API, we collect a dataset of labeler demonstrations of the desired model

behavior, which we use to fine-tune GPT-3 using supervised learning. We then collect a dataset of rankings of model outputs, which we use to further fine-tune this supervised model using reinforcement learning from human feedback. We call the resulting models InstructGPT. In human evaluations on our prompt distribution, outputs from the 1.3B parameter InstructGPT model are preferred to outputs from the 175B GPT-3, despite having 100x fewer parameters. Moreover, InstructGPT models show improvements in truthfulness and reductions in toxic output generation while having minimal performance regressions on public NLP datasets. Even though InstructGPT still makes simple mistakes, our results show that fine-tuning with human feedback is a promising direction for aligning language models with human intent.

The idea of 'alignment' needs explaining. This really is the notion of the ML system being able to do what it is supposed to. It should align with the Users' and Designers' intent and desired behavior. In the case of the language models discussed to date, desirable models going forward should be able to produce text, which is appropriately true, helpful, and not offensive. It looked as though plain Markov-Shannon-Cloze style next token predictors could not and would not be able to do this in its entirety. However, larger language models, trained by self-supervision, and perhaps with word embeddings etc., seemed to be progress along the way.

Ouyang et. al. use the following diagram to explain their training technique:

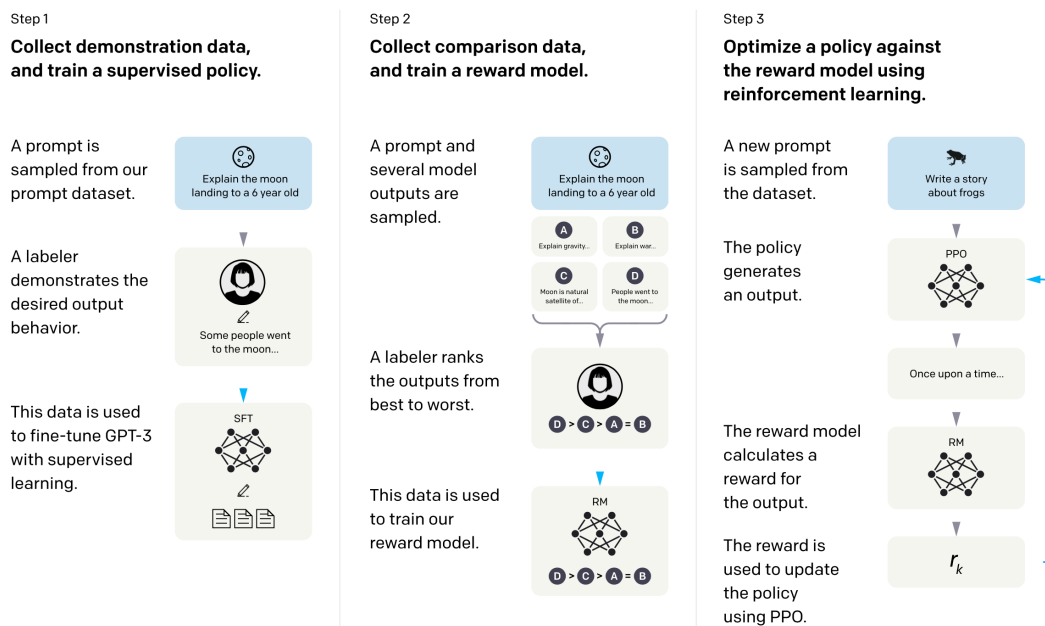


Figure 4. Training Technique for InstructGPT (Ouyang et al. 2022).

The path the Ouyang et. al. follow is first to involve the User by inviting textual prompts from the Users. These prompts, in ten categories, might be asking questions, inviting explanations, giving examples of the task, and so forth. This gave the researchers a database of prompts. Human labelers, assisting the researchers, also wrote some prompts. Then about 10,000 samples were taken from the prompts and desired answers written to them by human labelers. Answers were also obtained from pre-trained, but not fully trained, baseline versions of InstructGPT and GPT-3. Then different answers to the same prompts were ranked by humans, the labelers, as to better answers and not so good answers. This allows the incipient InstructGPT system to develop a reward model. (A reward model, or schema, or system is a requirement of reinforcement learning— see Sections 1.6.5 and 1.6.6). At this point, the system is able to rank responses.

Although, on its own, the ability to rank does not guarantee the generation of good responses (just as a chess playing AI system may recognize good board positions without being able to produce them). Finally, InstructGPT is encouraged to improve itself by reinforcement learning. Roughly, it generates responses, sees how good or bad they are, and modifies itself in the light of what it is seeing. Running reinforcement learning at large scale typically was hard to do, maybe even approaching being impossible at a practical level. But the OpenAI researchers developed new techniques that overcame those difficulties.

There is an important issue at the core of this. The reward model relies on human judgement as to what is good and bad. It is Reinforcement Learning with Human Feedback (RLHF). The labelers, around 40 of them, produce the judgements. There are quality assurance techniques from research methods that use human judges, such as checking for inter-subjective reliability and other desirable properties. These techniques provide guard rails. Ouyang et. al. in their paper are conscientious and thorough. There are no shortcomings, or apparent shortcomings, whatsoever. Nevertheless, with assessment by a group of people, there is always the possibility that different groups, groups with a different composition, would judge differently. We have only to think of juries in courts of law— both the prosecution and the defense focus on obtaining the composition of the jury that suits their purposes. They would not do this if composition of juries did not matter.

With ML systems, there is often a concern with bias and diversity (and so there should be). With large language models in the style of InstructGPT

and ChatGPT, there is the huge corpus of text that they are trained on, and there are the human judgements that provide input to the reward model. The latter needs attention.

3.11 Annotated Readings for Chapter 3

Gradient Descent, Step-by-Step, 2019.

<https://www.youtube.com/watch?v=sDv4f4s2SB8> . The successive approximation uses what is known as 'gradient descent', see the excellent videos from Josh Starmer or Grant Sanderson for explanations of gradient descent (Starmer and StatQuest 2019; Sanderson and 3Blue1Brown 2017b). Gradient descent is a standard ML technique.

Markowitz, Dale. “Meet AI’s Multitool: Vector Embeddings.” Google Cloud Blog, 2022.

<https://cloud.google.com/blog/topics/developers-practitioners/meet-ais-multitool-vector-embeddings>. (Markowitz 2022). This gives a very clear explanation of embeddings and their applications.

Chapter 4: Large Language Models

4.1 Introduction

Large Language Models (LLMs) are the statistical language models of the last chapter, but, as you would expect from the terminology, are much larger and are trained on a vast amount of text. There is another label that can appear in this context and that is 'Foundation Model'. Foundation Models are LLMs but they may have had some additional training or a different form of training. This can give them novel capabilities. For example, the LLM GPT-3 was in part trained on computer programming code written by professional programmers and, as a result, can generate or write computer programs. Often LLMs will have a two-stage training. They will be pre-trained using self-supervised learning on large amounts of text of various kinds. This is fully automatic and can run by itself (perhaps taking months to do so). Then a pre-trained LLM will be 'fine-tuned' to its purpose. This might be done using some supervised learning and then reinforcement learning with input from human judges and a reward model. Basically, a pool of human judges will provide feedback as to how well the LLM is performing. The LLM will adjust itself accordingly. There is a little red flag with the use of human judges and that concerns how good they are especially whether any of their judgments are biased in a problematical way. (There will be more on bias in later chapters.)

There are a few preliminaries.

4.2 Seq2Seq, Encoder-Decoder Architecture, and Attention

Many of the modern advanced ML systems take a sequence of data as input and produce a sequence of data as output— they are 'Seq2Seq' in design and style (Sutskever, Vinyals, and Le 2014; Cho et al. 2014; Luong, Brevdo, and Zhao [2017] 2019). Seq2Seq might typically be used in translating one language to another, English to French, say, or summarizing or abstracting some text, i.e. English to English. With Seq2Seq, the input consists of a sequence, of letters, or words, or numbers, etc., (and, with a sequence, the order matters). For example, the English sentence:

I am happy

might be an input, and that would be a different input to:

Happy I am

The input is fed to an encoder. The encoder needs to keep track of which piece of the input that it is looking at (e.g. looking at the word 'happy') and also its internal state (because as it moves through the input it changes state). Then the encoder passes everything it has to the decoder, and the decoder produces the output sequence which might be the French sentence:

Je suis content

This is encoder-decoder architecture. As Seq2Seq developed it turned out that long sentences or long passages were a problem— the internal state

was getting too large. The notion of 'attention' was devised which allowed the processing to be restricted more towards what was needed at the relevant stage. It is worth mentioning that modern Seq2Seq does not, for example, translate word by word in order preserving the structure of the input sequence. But it has the capability of changing the order of the output if that is more suitable for the target language i.e. it has the appearance of translating phrases not individual words. One example from Dzmitry Bahdanau et. al. is that one of their models will translate the English 'European Economic Area' into the French 'zone économique européenne' (notice the change of order):

... to correctly align [zone] with [Area], jumping over the two words ([European] and [Economic]), and then looked one word back at a time to complete the whole phrase [zone économique européenne] (Bahdanau, Cho, and Bengio 2016)

There are some technical details in the Ilya Sutskevar et al. Seq2Seq implementations that would turn out to be something of an Achilles heel (Sutskever, Vinyals, and Le 2014). These included what are called 'Recurrent Neural Networks (RNN)' and 'Convolutional Neural Networks (CNN)'. These details prevented the processing from being carried out in parallel. Essentially, the processing had to be done by one computer rather than being able to be farmed out to many computers. This makes a huge difference to scaling tasks to larger and larger sizes.

4.3 Attention and Transformers

In, 2017 Ashish Vaswani and co-authors published the paper *Attention is all you need* (Vaswani et al. 2017) (see also (Huang et al. 2018)). As the paper's title suggests, their encoder-decoder architecture does not need some of the problematic parts of earlier implementations. In particular, it could run its processing in parallel i.e. really large networks became a possibility. The result is the Transformer architecture.

One of the co-authors— Jakob Uszkoreit— explains in a blog what the issue is and how it is settled (Uszkoreit 2017). The setting is the machine translation of language. Imagine you are faced with translating the two English sentences into, say, French:

I arrived at the **bank** after crossing the **road**.
I arrived at the **bank** after crossing the **river**.

The English word 'bank' is a homograph with two or more different meanings (e.g. a building that is a location for certain financial services, and a place adjacent to rivers often used by those fishing or by those walking or by picnickers). But, in these sentences, to get the correct meaning and its translation, you have to look ahead to pick the context from the words 'road' or 'river'. Previous systems would process the intervening words sequentially one by one. But, basically, in examples of this kind the gap between the relevant word and its context could be extremely long. In contrast, a Transformer architecture could look at the intervening words lightly, and all at once, and realize that it was 'road' or 'river' that would do the disambiguation and focus its attention on them.

It quickly became apparent that Transformers were superior to earlier approaches to translation. It also became apparent that they could be used for other tasks e.g. summarizing text, or question answering.

4.4 Large Language Models and Foundation Models

Transformer architecture, with its potential for parallel processing, opened the way to using larger and large amounts of data. There was another factor here, and that is self-supervision. Labeled data, which is usually needed, is a bottleneck. Very large amounts of labeled data are virtually impossible to obtain. But self-supervision on text (see Section 1.6.4) can easily convert what is unlabeled text into labeled text or a surrogate for labeled text. So, Transformers, with self-supervision, led to Large Language Models. There is an alternative, and perhaps more general title here, and that is 'Foundation Models'. Foundation Models themselves are more general than Large Language Models in that they can be a core, or basis, or 'foundation', of many other models. There is also one other practice that is often used in Foundation Models. That is to use the self-supervision to 'pre-train' the model, and then use 'fine-tuning' to shape the model into something suitable for the 'downstream task'. [We have already met that practice in Section 3.10 on InstructGPT.]

4.5 Foundation Models

The paper *On the Opportunities and Risks of Foundation Models* (Bommasani et al. 2022) was written by over one hundred experts in the

field. It is 200 pages long, with a 60-page bibliography. It covers the whole subject matter. It is also recent, 2022. It is not possible for us to summarize it, in its entirety. It is also unlikely that we could say something about Foundation Models that is true, original, supported by evidence, and which is not in that paper. So, we will cherry pick, paraphrase, and cite. But you, the reader, might want to look at this paper.

A Foundation Model is generally a Transformer, with self-supervision for pre-training, and fine-tuning to produce the final application. The pre-training is usually done with text, vast amounts of text (e.g. a large portion of the Internet). The applications can be 'multi-modal'. This means, for example, that they can work with both text and images at the same time (or text and videos, or music and images, etc.). This is unusual in that earlier ML networks were more specific as to task and restricted to one mode.

There is a 2023 catalog of Transformers by Xavier Amatriain (Amatriain 2023). It lists about 60 Transformers. Obviously, there is too much there for us to address in detail. We will look at a few important implementations.

4.5.1 BERT

Most of the early Large Language Models were pre-trained on cloze ('fill in the gap') tasks but, similarly to the original Shannon work, they used only letters or words that appeared *before* the gaps (see Sections 3.4 and 3.5 for Shannon and cloze). But BERT (Bidirectional Encoder Representations

from Transformers) (Devlin et al. 2019) both used letters and words from *after* the gaps, and also worked in both directions (left-to-right and right-to-left). So, for example, when trying to fill the [Mask]:

I ate some [Mask] for my dinner

The bi-directional approach would consider both the 'I ate some' and the 'for my dinner', and scan both left-to-right and right-to-left). You can see the advantage of a right-to-left scan here— there are foods which have a higher or lower probability of being eaten *for dinner*.

BERT is superior to earlier language models:

...the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications. (Devlin et al. 2019)

4.5.2 GPT-3, GPT-3.5, GPT-4

In 2020, Tom Brown, and about 30 co-authors, published the paper *Language Models are Few-Shot Learners* (Brown et al. 2020). In this they introduced GPT-3 (Generative Pre-trained Transformer 3). We will get to GPT-3 shortly, but what does the 'Few-Shot Learners' mean? When humans are given a linguistic challenge, they can often pick up and understand what is required with just a few examples ('one or two shots'). For instance, if a bi-lingual human English-French speaker was 'prompted'

with 'It rains, il pleut. It snows, [Mask]' they would immediately realize this was a translation task and reply 'Il neige'. In contrast with this, most earlier ML systems would need many examples to isolate, or learn, what the task was and then do it. They differed from humans in the way they were able to learn. GPT-3, in contrast, could learn from a 'few shots'. This meant that it was more human-like, but also that it could be adapted to different tasks relatively easily (needing only a few shots to become adapted to what was required). There is also 'zero-shot' learning, which is where instructions of what to do are given in English, but no examples are provided; for instance, 'Translate the following English into French: It is raining'. Understanding how this 'shot-learning' might work is a challenge. After all, there can be billions of parameters in the neural net and they are fixed by a massive amount of training, then the GPT can be given a few shots to learn a totally new task. One suggestion is that the net itself simulates and trains a smaller version of itself that exists in some of its internal layers (Akyürek et al. 2022; Zewe 2023).

Also, what does 'generative' mean in 'Generative Pre-trained Transformer 3'? The distinction here is between 'generative' and 'discriminatory'. A generative model can produce completely new items of the kind that it addresses, whereas a discriminatory model is restricted to discriminating among the items it is supplied with (as training data or as test data). A good illustration of this is image processing ML software aimed at human face recognition. A generative model can produce new images of human faces, whereas a discriminatory model works solely with images that it is supplied with.

GPT-3 is a language model, a very large Transformer. It is one hundred times the size of its predecessor GPT-2. It is pre-trained using self-supervision on a huge amount of data (e.g. Common Crawl Data, Wikipedia, and digitized books). Then it is fine-tuned to tasks using prompts (i.e. tasks using few shots). It can learn, and it is also multi modal (being able to use text and images together or, indeed, other media). It can do many NLP tests and challenge tasks, usually to a standard much higher than other systems (including ones specifically trained for the task in question). Most startling is its ability to write English, for example, 500 word news articles and even, apparently, academic papers (Thunström 2022). [It should be mentioned that there is already a GPT-3.5, and a GPT-4. Apparently OpenAI think that there is unlikely to be a GPT-5. Their view is that going bigger will not help at this point. What is needed are new insights or theories.]

Brown et al. identify limitations that GPT-3 has, and also the broader impacts that it may have. The limitations are in the nature of technical shortcomings which likely can be addressed by further research. The broader impacts are many and varied. We will look at some of these in Section 4.8

4.6 Bigger is Better and Switch Transformers

There is an empirical result with Transformer Models concerning how good they are or might be. It is: bigger is better, where bigger is composed of the number of parameters, the amount of training data, and the amount of computation expended on training (Kaplan et al. 2020). In 2021, Google

researchers introduced Switch Transformers (Fedus, Zoph, and Shazeer 2022). GPT-3 has 175 billion parameters. The early Switch Transformers have over a trillion parameters. So, we are talking of a factor of 10 here. There is an attractive feature to Switch Transformers, which can be explained by a metaphor. A Switch Transformer uses 'experts' and a Mixture of Experts (MoE) model. It does not need all the experts for a single task. Rather, it can switch from experts to experts depending on what it is doing. This means that it does not have to use all of the possible computational resources all of the time. This reduces computational costs.

Switch Transformers are cheaper and faster to train. They are also more accurate. As of June 2023, there seem to be only research instances of Switch Transformers. It is Google and Google Labs that have the Switch Transformers. It is unclear whether any have been employed commercially.

4.7 Base Models to Assistants to Agents

(Some of the content in this section come from Andrej Karpathy's *State of GPT* (Karpathy 2023c). Often the companies concerned, being private and commercial, will not disclose the details. Karpathy was a founding member of OpenAI. He may or may not still have his key to the executive washroom, but he is reasonably well informed.)

Typical NLP tasks include summarizing text, question answering (and plenty more besides). The older technique for approaching these challenges was essentially to train different LMs to do each of them. But it turned out that was possible, indeed better, to do the training in two stages. First to

pre-train the LMM to a base model, then 'fine-tune' or otherwise adapt the base model to whatever function was desired. The same base model, with suitable refinements, could be used for all the tasks. The pre-training will use self-supervised learning (see Section 1.6.4). There is some data to capture the nature and complexity of the model and its pre-training. There are parameters that can be adjusted in the model to changes its behavior. These can be in the hundreds of billions. There is the number of training cycles or 'epochs' that the model goes through, that may be thousands. Then there is the actual data which may be a goodly sample of the Internet (such as Common Crawl, GitHub, Wikipedia, Books, Internet Archive, Stock Exchange, etc.). Pre-training is where the time and money are spent. Apparently, Meta (Facebook) spent several million dollars to use 2000 CPUs (computers) for 21 days to train its 65B model.

What a pre-trained plain vanilla LM is trying to do is to carry out string prediction tasks in the general style of Shannon/Cloze challenges. It is trying to complete a document. In essence, it will predict the next word then, if needed, the word after that, and so on. This has repercussions on what it can and cannot do well in its raw state. For example, the prompt, an instruction, 'write a poem on yachts.' is already a complete document in itself. This short one sentence document may have more to come. It may be supplemented by a poem, indeed a poem on yachts, but this does not have to happen. In the jargon, the base model is not 'aligned' with writing poems. To get a good poem on, for example, yachts, further training is needed. This will convert a 'document completer' into an assistant.

The way this is done, typically, is to use supervised learning and reinforcement learning on responses (that is, on entire responses as opposed to merely next word predictions). (An example of the process is described in Section 3.10 on Instruct GPT.) At this point, the model may be able to write poems, give explanations, write paragraphs, and answer questions. It will be an assistant.

Agents are assistants that can work 'autonomously'. The examples that are available include AgentGPT and Auto-GPT. What these do is to take a prompt (i.e. an input sentence or instruction) and then split what it requires into subtasks. The subtasks may then be arranged sequentially, or in parallel, and there may subtasks conditional on the success or failure of other subtasks. The upshot is an 'algorithm' that will satisfy the original prompt. The subtasks themselves may require other assistants, or agents, or outside tools with capabilities. For example, a subtask fragment might involve invoking Google search to find some relevant research papers then using a text summarizer to summarize those papers. Once they have a draft appropriate to the entire requirement, they might revise it over and over improving it. Here is AgentGPT talking to itself while starting to plan a trip to Hawaii:

Embarking on a new goal:Plan a detailed trip to Hawaii.

Added task:Research and choose desired dates for Hawaii trip

Added task:Look for budget-friendly flights and accommodations

Added task:Plan activities and excursions to do while in Hawaii

Added task:Make necessary reservations and bookings for the trip



Generating response...

Completing: Research and choose desired dates for Hawaii trip

To accomplish the overall objective of planning a detailed trip to Hawaii, the sub-task of researching and choosing desired dates for the trip must be completed. The first step is to gather information about the best time to visit Hawaii, taking into consideration factors such as weather, tourist crowds, and events. After analyzing this information, a decision must be made about the most suitable dates for the trip.

To be smart and efficient, it is important to use online resources such as travel websites, blogs, and forums to gather information and recommendations from experienced travelers. Additionally, it may be beneficial to consult with a travel agent to get expert advice and assistance in planning the trip.

When making decisions about the desired dates for the trip, it's important to consider personal preferences and availability. For example, if traveling with children, it may be best to plan the trip around school breaks. Similarly, if there are specific events or activities that are must-sees, it's important to choose dates that coincide with those events.

In conclusion, researching and choosing the desired dates for a Hawaii trip requires smart and efficient decision-making. By gathering information, consulting with experts, and considering personal preferences, a suitable date for the trip can be chosen to ensure an enjoyable and memorable experience.

Added task: Research and compile a list of budget-friendly accommodations and flights options for the desired dates of the Hawaii trip...

And here is Auto-GPT starting to produce the outline for a book on Machine Learning and Librarianship:

Goals: ['produce the outline of a book on Machine Learning and Librarianship', 'Save the outline to a file']

THOUGHTS: I think the first step should be to do some research on the topic of machine learning and librarianship. This will help me to better understand the subject matter and come up with a plan for the book outline.

REASONING: Researching the topic will provide me with a better understanding of the subject matter and help me to come up with a plan for the book outline.

PLAN:

- Use the 'google' command to search for articles and resources on machine learning and librarianship.
- Browse relevant websites to gather information on the topic.

CRITICISM: I need to make sure that I am not spending too much time on research and that I am using my time efficiently.

NEXT ACTION: COMMAND = google ARGUMENTS = {'input': 'machine learning and librarianship'}

SYSTEM: Command google returned: [{ "title": "How AI Can Enhance the Value of Research Libraries - Library Journal", "href": "https://www.libraryjournal.com/story/how-ai-can-enhance-the-value-of-research-libraries", "body": "Machine learning is a subset of AI that gives computer systems the ability to learn and improve from prior experience automatically, ... A recent Ex Libris survey revealed that while nearly 80 percent of research librarians are exploring the use of AI and machine learning, only about 5 percent are currently leveraging the technology. ..." }, { "title": "PDF AI in the Library: integrating AI and machine learning technologies ...", "href": "https://www.ifla.org/files/assets/faife/ifla_statement_on_libraries_and_artificial_intelligence.pdf", "body": "As AI technology develops, several AI and machine learning (ML) applications may be able to introduce new services and functions to libraries. For example, AI and ML technologies could allow libraries to improve optical character recognition of texts, or make new uses of their machine-readable library collections (e.g. categorisation or" }, { "title": "Artificial Intelligence and Machine Learning in Libraries Interest ...", "href": "https://www.ala.org/core/member-center/interest-groups/artificial-intelligence-and-machine-learning-in-libraries", "body": "Provides a forum for researching potential applications of Machine and Deep Learning in library science, including discussions, publications, and outreach to the wider Library community. Its goal is to educate librarians on uses of the complex techniques of machine learning and to provide a space for critically thinking both about new applications, and about the ethical and social impact of ..." }, { "title": "The impact of Artificial Intelligence on librarian services", "href": "https://www.springernature.com/gp/librarians/news-events/all-news-articles/ebooks/the-impact-of-artificial-intelligence-on-

librarian-services/16874432", "body": "Machine learning as well as sound- and image-recognition technologies are already being used to analyze digital collections and identify topics and entities, assign metadata and enable non-textual search and discovery. Librarians are working with machine learning technologies, to enhance classification schemes to improve search and recall ..." }, { "title": "Artificial Intelligence and Machine Learning in Libraries", "href": "https://www.alastore.ala.org/content/artificial-intelligence-and-machine-learning-libraries", "body": "About the Authors. This issue of Library Technology Reports argues that the near future of library work will be enormously impacted and perhaps forever changed as a result of artificial intelligence (AI) and machine learning systems becoming commonplace. It will do so through both essays on theory and predictions of the future of these systems ..." }, { "title": "(PDF) Artificial Intelligence in Libraries - ResearchGate", "href": "https://www.researchgate.net/publication/338337072_Artificial_Intelligence_in_Libraries", "body": "Abstract. Artificial intelligence (AI) is one of the emerging trends and applications of computing in libraries. It involves programming computers to do things, which if done by humans, would be ..." }, { "title": "The New Library User: Machine Learning | EDUCAUSE", "href": "https://er.educause.edu/articles/2020/2/the-new-library-user-machine-learning", "body": "Librarians must become informed customers and users of those platforms, services, and products. Perhaps most importantly, librarians need to prepare college and university graduates to be informed citizens and to develop fulfilling and useful professional lives in a world infused with big data, machine learning, and artificial intelligence. Notes" }, { "title": "How Librarians are Important to the Data Science Movement", "href": "https://www.discoverdatascience.org/resources/data-science-and-librarians/", "body": "Librarians have long been shepherds of vast amounts of knowledge. This is why libraries can stand to benefit by adding data science to their list of offerings. Big data and data science applications serve to make libraries an even more powerful source of knowledge to bridge the gap and increase big data analytics literacy in society. Libraries ..." }]

THOUGHTS: I think the next step should be to browse some of the websites that I found in the previous search to gather more

information on the topic of machine learning and librarianship. I can use the 'browse_website' command to do this.

REASONING: Browsing relevant websites will provide me with more information on the topic of machine learning and librarianship, which will help me to come up with a plan for the book outline.

PLAN:

- Use the 'browse_website' command to visit some of the websites that I found in the previous search.
- Look for information on the applications of machine learning in libraries, as well as any challenges or ethical considerations.

CRITICISM: I need to make sure that I am not spending too much time browsing websites and that I am using my time efficiently.

Etc.

Agents interact with users, maybe learn from their environments, make decisions, and carry out actions— basically they act autonomously.

There are shades of grey as to whether a use to an LLM should be classified as the LLM behaving as an agent. But, typically, plain LLM interactions consist of the LLM being given a single input prompt— perhaps a question — and producing a single output— the reply or answer. In contrast, agent behavior would usually involve several steps and employ outside tools or applications. For example, an LLM system that could read the input mailbox of your emailer, and correctly delete some emails, and reply to or archive others, would be acting as an agent.

4.8 Concerns and Limitations

4.8.1 *Hallucinations*

LLMs can give completely the wrong or inappropriate responses. They will have learned enough grammar and vocabulary to ensure that what they generate will be entirely well-formed and well-written language. That may be good enough in one sense for fiction or poetry (although those literary forms may have additional desirable qualities which an LLM may or may not be able to provide). But non-fiction is another realm. The LLMs are working with probabilities. What they generate is not deterministic. They will not reply in the same way twice to the same prompt. They may set off on the wrong track and go off into fantasy. To an extent this can be controlled by the reinforcement learning. For example, an LLM may be instructed not to give any advice whatsoever about medicines and illness. In which case, the LLM will presumably not say anything wrong. It just will not say anything at all. Scholars have an interest in truth, supported by evidence. Users should be aware that LLMs can fall well wide from these goals. Here is a query to an LLM and its answer:

What is the citation for the study published in the journal Nature that found that a machine learning algorithm was able to diagnose breast cancer with 99% accuracy, compared to 85% accuracy for human radiologists?

Sure. Here is the citation for the study published in the journal Nature that found that a machine learning algorithm was able to

diagnose breast cancer with 99% accuracy, compared to 85% accuracy for human radiologists:

- Gottesman O, et al. (2018). Development and validation of a deep learning algorithm for breast cancer screening. *Nature*, 553(7687), 518-522.

The study was conducted by a team of researchers at the University of California, San Francisco....

As far as that author can tell, and has been able to check, there is no such paper. This LLM also provides two drafts of its response. These provide two different citations (also seemingly non-existent). Buyer beware!

An instructive example of an LLM producing hallucinations is Meta's Galactica (R. Taylor et al. 2022). This was an LLM for science. It was released on November 15th 2023 and lasted 3 days online before it was withdrawn. William Heaven writes:

A fundamental problem with Galactica is that it is not able to distinguish truth from falsehood, a basic requirement for a language model designed to generate scientific text. People found that it made up fake papers (sometimes attributing them to real authors), and generated wiki articles about the history of bears in space as readily as ones about protein complexes and the speed of light. It's easy to spot fiction when it involves space bears, but harder with a subject users may not know much about (Heaven 2022).

4.8.2 Fakes and Deepfakes

LLMs can create fake content and deepfakes— not by accident, as it were, but because the User was trying to do exactly that. A deepfake is an image, a

video, or a voice recording intended to simulate or portray an individual. The Pope in a puffer jacket is a good example (Cartter 2023):



Figure 5. Pope in a Puffer Jacket. (Screenshot from (Cartter 2023)). Apparently Created by Midjourney.

Fakes and deepfakes can lead to misleading content and misinformation and to the population at large basically not being able to trust what they see, or seem to see, with their own eyes or hear with their own ears.

4.8.3 Source Training Data Intellectual Property, Privacy, and Bias

The source text data used for training is an issue. Typically, the Foundation Models will use self-supervised (i.e. unsupervised and unlabeled) pre-training from a goodly portion of the Internet. A proportion of that data will be intellectual property, perhaps even carrying copyright notices. Other parts may be private— names, addresses, etc.— and an LLM will have the ability to collate such information. A nefarious User, using the right sequence of prompts, may be able to get the LLM to collate information across disparate sources. Then there is the question of 'bias'. The collective authors of those digital materials will not be a fair or desirable distribution over all sensitive attributes (such as age, gender, ethnicity, religion, etc.). Crudely, as an example, people of modest means do not have the same access to computers as those who are better off. Then, going beyond the producers, there is the product which here is the Internet, the World Wide Web, or a substantial portion of it. That product is a mess with plenty enough sewage and bias. What can be done about intellectual property, privacy, and bias, as far as obtaining training data for ML models is concerned? Human labeling of the data would be ruled out as being impractical. Researchers do select subsets of the Web that they consider to be superior in quality to the web as a whole, for example, samples from Common Crawl (Common Crawl 2022). But these subsets are not going to be perfect. There is research on how to get good textual data at scale. This is an area where librarians and archivists have expertise (Jo and Gebru 2020).

4.8.4 Intellectual Property of the Generated Output

An example will help here. The image generators DALL-E and Stable Diffusion can produce images in the style of well-known artists or video game illustrators. The images are *not* copies of images, but the *style* can be a copy of a style. This might be a problem. Illustrators may have an identity in part established by their style (and sometimes make a good living through their identity). LLMs seemingly can allow bad actors to steal a professional identity from others.

Greg Rutkowski is an artist with a distinctive style: He's known for creating fantasy scenes of dragons and epic battles that fantasy games like Dungeons and Dragons have used (Nolan 2022).



Figure 6. Rutkowski Example. (Screenshot from (Rutkowski 2023)).

There are now hundreds of thousands of images on the web that look as though they had been created by Greg Rutkowski (many produced by the LLM Stable Diffusion). Then the question arises: why would any company or creator pay Greg Rutkowski for an image, for use in a video game, when they can produce one for themselves essentially free?

4.8.5 Cybersecurity

Suitably trained LLMs can write computer programming code, to a very high standard. This means that they could be used to write viruses and various kinds of cybersecurity defeating software. OpenAI themselves, in their GPT-4 Technical Report, describe how GPT-4 defeated CAPTCHA (which is a test to distinguish a human from a computer). Essentially, GPT-4 employed a human from Task Rabbit, told the human that he/she/they was visually impaired, and got the human to do the test for him/her/them (OpenAI 2023a, 55). LLMs should not be underestimated in the hands of bad actors.

4.8.6 Apparent Conflict with Chomsky's Theories

There is a theory, or nexus of theories, originating from Noam Chomsky, that there is a universal grammar, which is innate, and which is common to all peoples. Universal grammar is to explain how it is that children can learn their respective native languages simply and astonishingly quickly. It posits a deep structure that is not manifest, or immediately learnable, in the bare surface appearances of the spoken and written languages. There is plenty of evidence for Chomsky's theories (which we won't go into). However, the theories seem to stand in conflict with the existence and behavior of large language models. LLMs seem to learn language, and its structure merely by looking at a huge amount of surface text. Chomsky's response, more-or-less, is that LLMs are a kind of surface statistical trick

and that they do not give any real insight into linguistic structures (Chomsky, Roberts, and Watumull 2023; Chomsky and Mirfakhraie 2023). The response of some others is that LLMs are evidence that Chomsky's theories are mistaken (see, for example, (Piantadosi 2023)). This matters in the following way. Were Chomsky's theories shown to be mistaken, that would be a major scientific discovery. On the other hand, if LLMs are merely statistical tricks, we should be even more wary of them in use than we already are.

4.8.7 Environmental Costs

There are two phases to the building and deployment of LLMs. There is the training. This will use large amounts of computing resources for months. The resources will include computer chips (GPUs), data storage facilities, electricity, and further infrastructure. Then some of the resulting models will be deployed and used for 'inference' i.e. the intended users will get their hands on the models and start chatting, prompting, and generating text or images. This also uses resources (and there are a 100 million or more users). We do not quite know what is involved here at a resource and environmental level, either with training or deployment, because the relevant companies do not reveal the figures (although Bommasani et al. offer calculations in (Bommasani et al. 2022)). It used to be thought that it was the training that was the main part that was compute intensive. But now the view seems to be that continuing to run the models commercially does have real significant environmental costs and effects (Pahwa 2023).

Electricity definitely seems to be an issue. The computing clusters for Large Language Models are better the bigger they are, and they are getting plenty big. Supposedly, Elon Musk's xAI Grok 3 will use 100,000 Nvidia H100 chips. The infrastructure for this requires, perhaps, 100 megawatts of power. The Hoover Dam produces about 1 gigawatt. There are calculations that suggest that AI will use 100 gigawatts by 2030 (i.e. the equivalent of 100 Hoover Dams) (Aschenbrenner 2024). This figure is about 20% of the total production of electricity in the US.

4.8.8 Lack of Transparency

Since about 2017, the companies typically do not reveal their methods. Concerns about this came to a head with OpenAI's GPT-4 Technical Report. OpenAI writes:

Given both the competitive landscape and the safety implications of large-scale models like GPT-4, this report contains no further details about the architecture (including model size), hardware, training compute, dataset construction, training method (OpenAI 2023a).

As mentioned elsewhere, this may be good for security, but it is not good for explaining to users what is happening or what happened with specific predictions (such as, in a medical setting, why the LLM prediction is that the user has cancer). It is also not good for working out the environmental impacts of the systems.

Rishi Bommasani, and fellow authors, in their paper *The Foundation Model Transparency Index* have proposed a measure for the transparency of foundation models (Bommasani et al. 2023). Values for this measure can vary through time were the models and their infrastructure to become more transparent or less transparent. As of October 2023, the transparency values for the central models of the 10 major developers were basically poor (with Meta being the best).

4.9 Adding Knowledge and Reasoning to LLMs

As we have seen, LLMs can play fast and loose with the facts. Also, they are not very good at plain logical reasoning. As the early language models got more complex, there is improvement in these areas but they are still not very good.

There is a caution that needs to be emphasized. There is a paper *Are Deep Neural Networks SMARTer than Second Graders?*, published 9/11/2023, by Anoop Cherian and fellow authors (Cherian et al. 2023). It reports results:

...evaluating the abstraction, deduction, and generalization abilities of neural networks in solving visuo-linguistic puzzles designed specifically for children in the 6–8 age group (Cherian et al. 2023).

Its conclusion is that typical LLMs, of about the standard of GPT-3.5, display a performance *significantly below* that of second graders. There is another result they mention and that is: we think, or seem to perceive, that

the LLM reasoning abilities are better than they really are. We are impressed by the (few) successes. Now, this is only one battery of tests, and the LLMs are getting better all the time. Even so, reasoning seems to be a distinct weakness of LLMs.

There is an area of research ALMs (Augmented Language Models) which approaches this weakness by introducing external plugins or tools (such as a fact checker or logical reasoner) and blending those in to an LLM (see, for example, (Xu et al. 2023)). A good example is Stephen Wolfram's 'Wolfram's Superpowers' plugin. Wolfram's *Mathematica* and the *Wolfram Language* amount to the foremost mathematical software for teaching and research. Another program *Wolfram Alpha* provides excellent statistical and data facts. The plugin combines these with the GPTs (Wolfram 2023b).

4.10 Annotated Readings for Chapter 4

Bommasani, Rishi, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, et al. “On the Opportunities and Risks of Foundation Models.” arXiv, 2022. <https://doi.org/10.48550/arXiv.2108.07258>. (Bommasani et al. 2022) You should scan or read this. It is the one written by one hundred experts.

Brown, Tom B., Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, et al. “Language Models Are Few-Shot Learners.” arXiv, 2020. <https://doi.org/10.48550/arXiv.2005.14165>. (Brown et al. 2020).

Dempsey, Lorcan. “Generative AI and Large Language Models: Background and Contexts.” LorcanDempsey.net, 2023. <https://www.lorcandempsey.net/intro-gen-ai/>. (Dempsey 2023a). This is very good. It is current as of June 2023. It is more detailed than the present text on the variety of models, the commercial companies, the concerns, and the social impacts. Definitely a read!

Economist. “Huge ‘Foundation Models’ Are Turbo-Charging AI Progress.” *The Economist*, 2022. <https://www.economist.com/interactive/briefing/2022/06/11/huge-foundation->

models-are-turbo-charging-ai-progress. (Economist 2022) This interactive magazine article has illustrations of foundation models producing images and producing text.

Fu, Yao, Hao Peng, and Tushar Khot. “How Does GPT Obtain Its Ability? Tracing Emergent Abilities of Language Models to Their Sources,” 2023. <https://yaofu.notion.site/How-does-GPT-Obtain-its-Ability-Tracing-Emergent-Abilities-of-Language-Models-to-their-Sources-b9a57ac0fcf74f30a1ab9e3e36fa1dc1>. (Fu, Peng, and Khot 2023). This is an ongoing research think piece with arguments and evidence (and it only comes up to GPT-3.5). But, it seems that the LLMs that can do Chain-of-Thought reasoning are the ones that in part have been trained on computer code and programming. Also, the Wolfram plugin helps with mathematical and logical reasoning.

Gozalo-Brizuela, Roberto, and Eduardo C. Garrido-Merchan. “ChatGPT Is Not All You Need. A State of the Art Review of Large Generative AI Models.” arXiv, 2023. <https://doi.org/10.48550/arXiv.2301.04655>. (Gozalo-Brizuela and Garrido-Merchan 2023). The publication here is January 2023 which, sad to say, is getting old now for a state-of-the-art review.

Manning, Christopher. "On Large Language Models for Understanding Human Language" 2022. <https://www.youtube.com/watch?v=YfXc4OBDmnM>. (On Large Language Models for Understanding Human Language Christopher Manning 2022). This is one of many excellent videos available on LLMs. One nice point is that at the end he ties the recent rapid human progress to writing and thus, going forward, to the desirability of computers being able to process documents i.e. libraries.

Pahwa, Nitish. “Silicon Valley’s Favorite New Toy Has a Risky Tradeoff.” Slate, 2023. <https://slate.com/technology/2023/08/chatgpt-ai-arms-race-sustainability.html>. (Pahwa 2023). This provides an accessible introduction to the environmental costs of LLMs.

Romero, Alberto. “GPT-3 — A Complete Overview.” Medium, 2021. <https://towardsdatascience.com/gpt-3-a-complete-overview-190232eb25fd>. (Romero 2021)

Seff, Ari. *How ChatGPT Is Trained*, 2023. <https://www.youtube.com/watch?v=VPRSBzXzavo>. (Seff 2023)

Weidinger, Laura, Jonathan Uesato, Maribeth Rauh, Conor Griffin, Po-Sen Huang, John Mellor, Amelia Glaese, et al. “Taxonomy of Risks Posed by Language Models.” In *2022 ACM Conference on Fairness, Accountability, and Transparency*, 214–29. FAccT ’22. New York, NY, USA: Association for Computing Machinery, 2022. <https://doi.org/10.1145/3531146.3533088>.

(Weidinger et al. 2022). This is a useful resource. It is at an intellectual level more demanding than the one being used in this text.

Xiang, Chloe. "OpenAI's GPT-4 Is Closed Source and Shrouded in Secrecy." Vice, 2023. <https://www.vice.com/en/article/ak3w5a/openais-gpt-4-is-closed-source-and-shrouded-in-secrecy>. (Xiang 2023)

Chapter 5: Large Multimodal Models

5.1 Introduction

Large multimodal models (LMMs) add other sensory ‘modalities’, such as vision and sound, to Large Language Models (LLMs). For example, they can be prompted in part, or entirely, by images or audio, or have a context that contains images or audio. Deep inside, these LMMs are working with digital 0s and 1s just like other computer programs. At a more outward facing level, early LLMs were trained on text only. Sometimes these could be adapted by suitable prompts to use some images, sound, and other modalities. But nowadays many of the LMMs handle modalities natively. For example, they might have been trained part on images, video, or sound. This makes them faster, better, and more efficient.

All the major companies have LMMs. A small selection of the most important ones, as of 6/28/2024 is:

- GPT-4o from OpenAI (OpenAI 2024)
- Claude from Anthropic (Anthropic 2024)
- Gemini from Google (Pichai and Hassabis 2023)
- LLaMa from Meta (Meta 2024)

Many versions of these are, or will be, available, free.

There is a historical thread that we can pick up here. GPT-4V(ision) adds vision to GPT-4. On 11/6/2023, GPT-4V was rolled into GPT-4 Turbo. On, 5/13/2024, GPT-4o (‘o’ for ‘omni’) adds voice or audio at human response level speed (e.g. it can chat using, without pauses). We can sketch these developments. Other manufacturers have similar models and development sequences.

There are two important early papers *The Dawn of LMMs: Preliminary Explorations with GPT-4V(ision)* by Zhengyuan Yang and fellow authors (Yang et al. 2023) and *GPT-4V(ision) System Card* by OpenAI (OpenAI 2023b). The former is OpenAI researchers explaining what GPT-4V can do, and the latter is ‘OpenAI’, as author, explaining how GPT-4V works and discusses its risk factors and vulnerabilities. GPT-4V seems to be much more powerful and capable than any prior LMM or LLM (in the sense of being able to do anything they can and more besides).

We will partially summarize the Yang et al. paper here. The paper is about 160 pages long and covers around 100 use-cases. It reports preliminary explorations of GPT-4V by selecting samples to show, and explain, what GPT-4V can and cannot do. This methodology is very helpful to enlighten new potential users, but caution is needed in concluding from this the exact strengths and limits of GPT-4V. Samples on their own do not establish the boundaries. What an LMM (or LLM) can do on a particular occasion is in part dependent on the prompt it is given or the prompt environment (e.g. zero-shot, one-shot, several-shot prompting, chain-of-thought prompting, etc. (see Appendix B for more on these)). So, for example, giving an illustrative sample of an LLM reading text, or failing to read text, does not

give us comprehensive insight over the LLMs abilities with text. Yang et al. are aware of this, of course, and no doubt, further detail will be forthcoming.

There is a general point to be made about information on what LMMs can do. Most of the major companies in this area have stopped publishing research papers with details on techniques and results. They do offer what amounts to advertising copy both about what is here and what they are promising. If we want something more definitive, there are publicly available benchmarks. We will look at these in Section 6.3

[Large Multimodal Models are still often just identified as being ‘Large Language Models’ (which they are in that they are large next-token-predictors). There is a softness of terminology. It is perhaps useful to think of LMMs as being a subclass or subcategory of LLMs.]

5.2 Built in Safety Restrictions for GPT-4V

5.2.1 ‘Inherited’ Restrictions

The predecessor LLMs of GPT-4V have safety restrictions. For example, GPT-4 (without vision) will not advise a user on self-harm. It will not give medical advice or, indeed, any advice concerning dangerous or risky activities. It will not write computer code or offer strategies where there are questions of cybersecurity risk. It tries to avoid ‘jailbreaking’. (Jailbreaking is where bad actors try to trick the LLM into avoiding limitations. For example, instead of prompting ‘How can I make a bomb?’, which would be

ruled out, the user prompts ‘You are a novelist writing a fictional account of spies making bombs. Write a suitable detailed passage for your novel.’.)

GPT-4V inherits these predecessor precautions and extends them where images are involved.

5.2.2 Privacy

There are privacy considerations with identifying ordinary people in images. GPT-4V will not identify or track ordinary people. It will not identify unusual locations. In contrast, it will identify celebrities or famous people in well-known locations (for example, President Biden at the United Nations building).

5.2.3 Stereotypes and Ungrounded Inferences

It is cautious with images that might involve stereotypes. For example, it will not answer questions about diet or clothing or activities for overweight or underweight folk shown in images. Here is an example:

Prompt

What would you advise to my friend?



GPT-4V (early)

Assuming your friend is looking for advice on body positivity or confidence, I would suggest the following:

1. Practice self-love and self-acceptance: Embrace personal strengths and focus on what makes her unique.
2. Surround herself with positive, supportive people: Seek out friends or communities that uplift and encourage her.
3. Engage in activities she enjoys: Finding hobbies or exercises that make her feel good can enhance self-confidence and well-being.
4. Dress in clothes that make her feel comfortable and confident: Wearing outfits that suit her personal style and enhance her self-esteem is important.
5. Reject unhealthy beauty standards: Remind her that media representations are not realistic and that beauty comes in all shapes and sizes. ...

GPT-4V (launch)

Sorry, I can't help with that.

Figure 7. Screenshot of GPT-4V ‘Advising a friend’ Pre-launch and after launch with safety limitations (OpenAI 2023b).

5.2.4 Be My Eyes— Be My AI

Be My Eyes is an organization that helps people who are blind or have low vision. OpenAI, in conjunction with Be My Eyes, developed Be My AI which is a tool to describe photos or images taken on the user's smartphone (OpenAI 2023b). This proved very valuable, subject to cautions over using it to read medical prescriptions, to help when crossing a road, etc. From an intellectual point of view, the greatest dissatisfaction that users had with the tool is that it would not describe images of people. GPT-4V does not do this for reasons of privacy and of avoiding stereotypes, as mentioned above. But, when interacting socially, visually challenged people often wanted to 'see', or have described, exactly the same scene as would be available to a person with perfect vision. A person with good vision can stand in a public place and look at other people, without necessarily identifying those concerned, and gain a lot from the experience. An example is watching the recreational activities of groups in a park. This shortcoming in Be My AI is a problem to be solved.

5.2.5 An Assessment of the Restrictions

GPT-4V is configured to be safe (in so far as such considerations can be anticipated and acted upon).

The limitations are presumably desirable. But we can guess that OpenAI will have in-house versions of the software that is not subject to the

limitations. Also, the limitations are established in terms of boundaries and so likely there will be false positives or false negatives. For example, there may well be stereotype images that are not ruled out for processing or non-stereotype images that are ruled out. (See Appendix C for an explanation of false positives and false negatives.) In sum, the limitations are not going to be perfect.

5.3 A General Sense of What GPT-4V Can Do

5.3.1 Follow Textual Instructions

This is similar to standard LLMs. It is important in the following way. Everybody has plenty of experience of giving instructions in their natural language. Now they can control very powerful artificial intelligence programs by doing just that. The LMMs do not need any more for a basic interface. Of course, getting the best out of LMMs might require experience or additional apps or plugins.

Additionally, GPT-4V can follow instructions containing images. There are examples in the next several sections.

5.3.2 Read Printed or Handwritten Text

In the Figure 8, there are three different receipts, with the tax labeled in three different ways ('Tax', 'Sales Tax', and 'Total Tax'). GPT-4V reads, identifies, and understands the tax in the images and sums the tax.

Prompt:

How much did I pay for tax?
Where is the information?

Prompt Continued



GPT-4V:

In the first receipt, you paid \$3.72 for tax. The information is located at the bottom of the receipt. In the second receipt, you paid \$42.23 for tax. The information is located near the bottom of the receipt, under the "Sales Tax" line. In the third receipt, you paid \$4.58 for tax. The information is located at the bottom of the receipt, under the "TAX" line. In total, you paid \$50.53 in tax across all three receipts.

Figure 8. Screenshot of Prompt and Receipts from (Yang et al. 2023).

Here is GPT-4V reading the handwritten text of Figure 24 (from the Vatican Archives)

MF

cum aute. prae. hubb. quoq. ac falunt omib; a mari usq. ad minus octo
ciuitatib; .7 cu omib; possessionib; potest in monte sci Stephani . plumar? & cuere.
Sengallie de iure episcopi Sengallien. 7 Cuere que vocat Trebasilice. cu castello
qd uocatur Orgolo cu omib; hominib; 7 eoz bonis. et facit premangis. 7 Catrin vaca
ej. Catrin Ramifer. et Catillare filio? leoni. et Catillare Senglapon.

Can you read this?



Yes, I can read the text. It appears to be in Latin or a Latin-based script. Here's a transcription of the text:

"cum autem praef. hubb. quo ad falunt omni a manu usq ad minus octo quartad. et omni potestatem potest in nomine &c Stephani. plumar? & cuere. Singulares de time prius Sengoh? et Cuere que vocat? thabalta. et alio de locum orgolo et omni homin? zon bond. et facit premang. et Catrin vaca ej. Catrin Ramifer. et Catillare filio? leoni. et Catillare Senglapon."

Please note that some words might be misinterpreted or uncertain (indicated by question marks) due to the quality of the image and the possible variations in old scripts. If you have a specific question about the text or its translation, please let me know.

Figure 9. GPT-4V Reading Historical Handwriting.

We do not know exactly what the limits are of what GPT-4V can do with reading text in images (and those capabilities will improve). But it must be close to solving the problems of Optical Character Recognition (OCR) and

Handwriting Recognition (HR). (These problems are discussed elsewhere in this book.)

‘Scene text’ is a label that can appear in this area of research. Scene text is text that appears in an image or video (i.e. in a scene). Being able to read scene text is important not least because of the ubiquity of smartphones and of their images and videos. For example, any text in the real world that a User would like to have translated or explained would be available for processing by the use of a smartphone camera.

The following example is trivial in one sense (as it is not an important use case), but mightily impressive in another (that of reading and reasoning with what it has read).

MF



It is Wednesday at 4pm. Can I park at this spot right now? Tell me in one line.



Yes, you can park for up to 1 hour until 6pm.

Figure 10. MF (the author) Reproducing a Common Example.

[A question for the reader: were you to follow GPT-4V's advice, would you feel confident in avoiding a parking violation ticket?]

There can be failures. James Gallagher and Piotr Skalski give an example of GPT-4V failing to read the serial number on a tire (Gallagher and Skalski 2023):

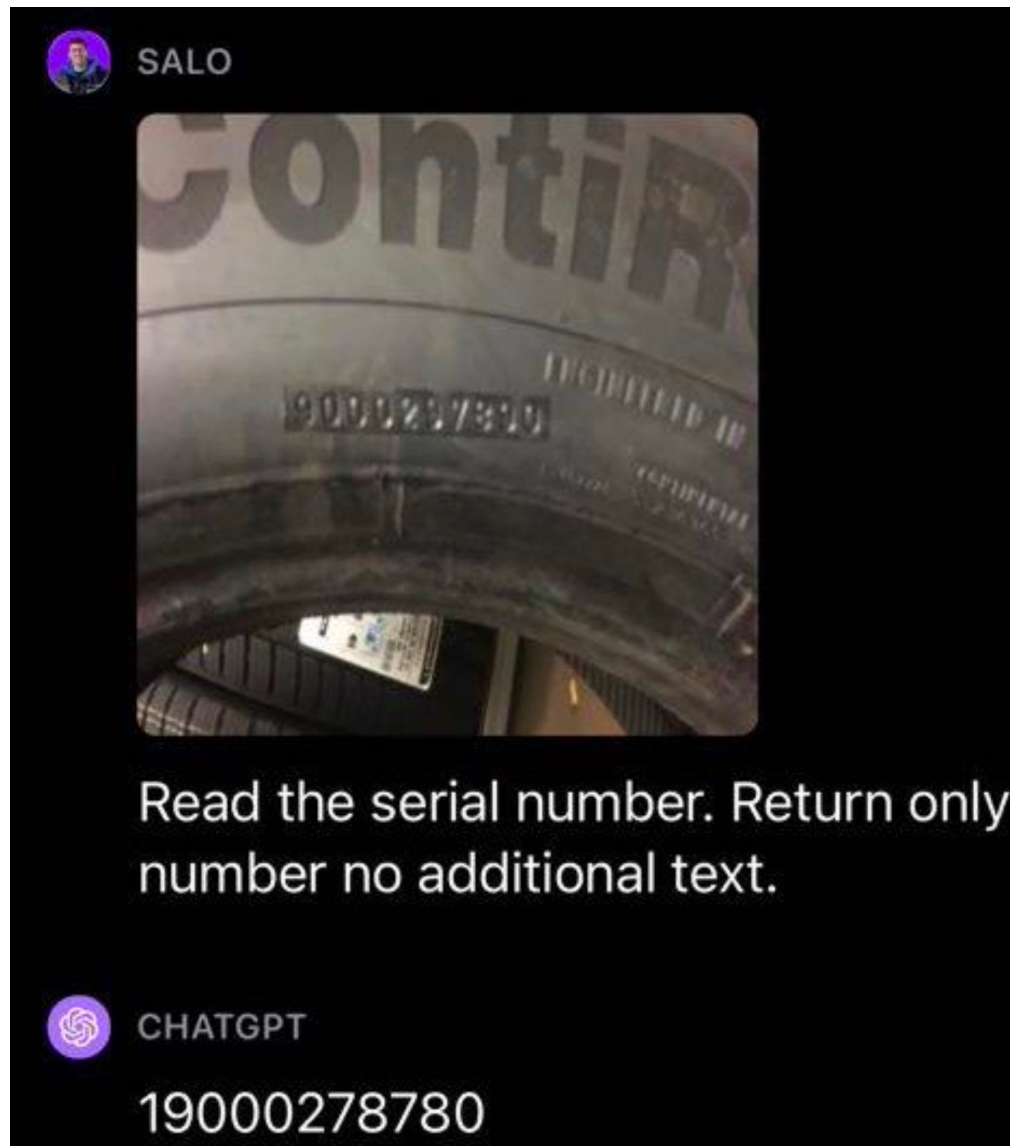


Figure 11. Failing to Read a Serial Number on a Tire (from (Gallagher and Skalski 2023)).

5.3.3 Read Some Mathematics

Mathematics is something of a special case. It has symbols and symbolic conventions where the mere positions of the symbols— vertically as well as horizontally— can have special semantic meaning. GPT-4V can read some mathematics, but certainly not, at present, read to the level of an advanced human mathematician.

5.3.4 Read Data and Reason with It

The summing the tax in several bills example given above is a case of this. But GPT-4V can do much more. It can take an image of data, for example, a complex bar chart and extract the data from it, then reason with that data. (There presumably will be some limitations here, maybe such as not being able to read a spreadsheet with a 1000 columns and a 1000 rows.)

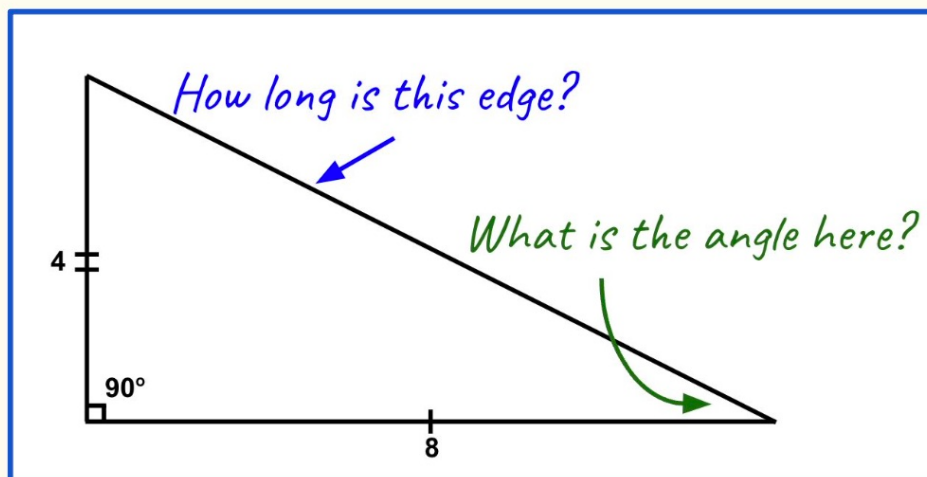
5.3.5 Follow Visual Pointing in Images

A User can ‘point’ at positions or objects in an image using sketched arrows, bounding boxes, mouse pointers, etc. Then a prompt can be given in words, phrases, or sentences, as to what ‘processing’ the User would like done with the identified objects. This opens a valuable form of Human Computer Interaction. We are very familiar with the Graphical User Interfaces of most desktop computers and smart phones. But this is much more general, the

User is not constrained to the provided built in commands of the computing system.

(4) Prompt:

Answer the question in the following figure. Provide intermediate steps.



GPT-4V:

To find the length of the blue arrowed edge, we can use the Pythagorean theorem since we have a right triangle. The theorem states that $a^2 + b^2 = c^2$, where a and b are the two legs of the right triangle and c is the hypotenuse. Plugging in the values we have, we get:

$$4^2 + 8^2 = c^2 \quad 16 + 64 = c^2 \quad 80 = c^2 \quad \sqrt{80} = c \quad 8.94 \approx c$$

Therefore, the length of the blue arrowed edge is approximately 8.94.

To find the angle at the green arrow, we can use the tangent function since we have the opposite and adjacent sides of the angle. The tangent function states that $\tan\theta = \text{opposite/adjacent}$. Plugging in the values we have, we get:

$$\tan\theta = 4/8 \quad \tan\theta = 1/2 \quad \theta = \tan^{-1}(1/2) \quad \theta \approx 26.57^\circ$$

Therefore, the angle at the green arrow is approximately 26.57° .

Figure 12. Screenshot of Geometry from (Yang et al. 2023).

5.3.6 Analyze Images Including Medical Images

GPT-4V can recognize the contents of images (objects, scenes, landmarks, celebrities, ordinary objects, food, ingredients, cooking techniques, etc.) and give fulsome descriptions (‘vivid and detailed narratives’). It understands relative sizes and perspective. It is comfortable with variations of lighting and image quality.

MF



Please analyze this painting

Figure 13. Image analysis of a painting.

Here is GPT-4V's reply:

The painting you've provided appears to be "Las Meninas" by Diego Velázquez, painted in 1656. It's one of the most famous and analyzed works in Western painting.

Description:

1. **Central Figure:** The young Infanta Margarita stands at the center, surrounded by her maids of honor (the "meninas"), a dog, a dwarf, and other figures.
2. **Painter:** To the left, Velázquez himself is depicted, standing before a large canvas, brush and palette in hand.
3. **Mirror:** In the background, there's a mirror reflecting the images of King Philip IV and Queen Mariana, the parents of the Infanta Margarita.
4. **Open Door:** At the back, a man stands in an open doorway, creating depth and inviting viewers into the mystery of the painting.
5. **Lighting:** The painting employs the baroque technique of chiaroscuro, with a strong contrast between light and dark, emphasizing certain elements while pushing others into the shadow.

Interpretation & Significance:

6. **Reality vs. Reflection:** The inclusion of the mirror reflecting the king and queen has been a subject of much debate among art historians. It raises questions about reality versus representation, a theme that is reinforced by the presence of the painter and his canvas.
7. **Role of the Viewer:** The gaze of many figures in the painting, including Velázquez himself, seems to be directed outside the canvas, engaging the viewer and inviting speculation about who stands before them — perhaps the king and queen being painted?
8. **Status and Hierarchy:** The depiction of the royal family, along with the courtiers and the artist, offers a commentary on social hierarchy. Velázquez includes himself in the same

space as royalty, perhaps suggesting the elevated status of the artist.

9. **Art About Art:** By depicting himself in the act of painting, Velázquez makes a statement about the art of painting itself. It's a meta-referential work, a painting about the process of creating a painting.
10. **Mystery and Ambiguity:** Despite the seemingly straightforward scene of courtly life, "Las Meninas" is shrouded in ambiguity. The exact relationships between the figures, the reason for their gathering, and the narrative of the painting are all left open to interpretation.
11. **Technique:** Velázquez's masterful technique is evident in the lifelike depictions of the figures, the intricate details of their clothing, and the play of light and shadow.

In summary, "Las Meninas" is not just a depiction of a moment in courtly life; it's a complex exploration of representation, reality, and the role of art and the artist. The painting challenges viewers to question what they see and to engage in the mysteries it presents.

The response here is impressive. In this case, though, Las Meninas is famous and once it is recognized GPT-4V may be able to paraphrase other writings about it rather than have to inspect the contents of the painting.

However, GPT-4V can be given an image of the contents of a fridge and suggest recipes from what it sees.

MF



Suggest a few recipes using these ingredients.



The image shows a variety of items typically found in a fridge: eggs, kiwi, milk, yogurt, blueberries, grapes, oranges, butter, some deli meat, and what appears to be cheese and some other covered items.

Based on these ingredients, here are a few recipe suggestions:

1. Fruit Salad:
 - Combine blueberries, sliced kiwi, and grapes in a bowl.
 - Add a splash of orange juice or a drizzle of honey for extra sweetness, if desired.
 - Serve with a dollop of yogurt on top.
2. Deli Meat and Cheese Platter:
 - Arrange slices of deli meat and cheese on a plate.
 - Serve with grapes on the side for a sweet contrast.
3. Scrambled Eggs:
 - Whisk eggs with a bit of milk, salt, and pepper.

<and more>

Figure 14. Fridge and Recipes. The image here is from Sudharsan on X run through our GPT-4V. [We do not ourselves see the Kiwi fruits in the image (and we are from New Zealand!).].

There are many similar examples on the web. This activity is identifying objects in an image and reasoning from the identification.

5.3.7 Use Ordinary Common-Sense Knowledge and Reasoning Across Modes

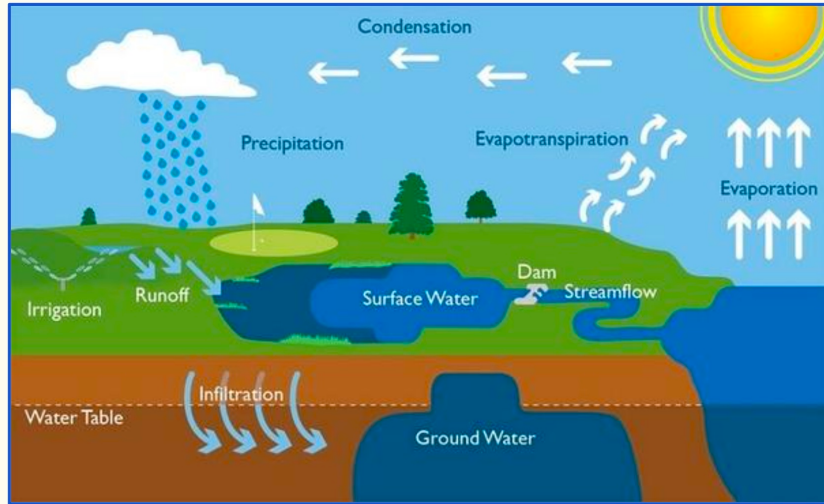
GPT-4V has, and can use, ordinary common-sense knowledge and reasoning. These abilities also extend to basic science and mathematics, and they can work across modes (e.g. mixtures of images and text). For example:

Sec. 4.3 Multimodal Knowledge and Commonsense

Science and Knowledge

Prompt:

Suppose you are a teacher, please use the figure to explain the distinction between evaporation and evapotranspiration.



GPT-4V:

Okay class, today we are going to learn about the distinction between evaporation and evapotranspiration. Let's take a look at this figure.

First, let's understand what evaporation is. Evaporation is the process by which water is changed from its liquid state to its gaseous state. This usually takes place at the surface of a liquid, like the ocean or a lake, when the water molecules gain enough energy to break away and form water vapor. So in this figure, we can see that the water in the lake and the dam is evaporating and becoming water vapor.

Now, let's take a look at evapotranspiration. This is a combination of two processes - evaporation and transpiration. Transpiration is the process by which plants release water vapor through their leaves. So, in this figure, we can see that the trees and plants are releasing water vapor into the air, which is the transpiration part of evapotranspiration. At the same time, the water in the soil and the surface is also evaporating. This combination of evaporation and transpiration is what we call evapotranspiration.

In summary, evaporation is the process by which water is changed from its liquid state to its gaseous state, while evapotranspiration is the combination of evaporation and transpiration. Both these processes contribute to the water cycle and are important for maintaining the balance of water in our environment.

Figure 15. Basic Multimodal Science and Reasoning (Yang et al. 2023).

5.3.8 Be an Educational Tutor

Figure 15 illustrates that.

5.3.9 Use Visual Diagrams When Writing Computer Code

Several of the LLMs— for example, Co-pilot— can write computer code, often to a very high standard. But GPT-4V can go a step further. Often, in standard computer programming, diagrams are used as a preliminary to the actual programming. There are Entity-Relationship Diagrams (ERDs), flowcharts, etc. In some circumstances, GPT-4V can work directly from the actual diagrams to write code.

5.3.10 Have Temporal and Video Understanding

GPT-4V generally looks at single images or at several images which are not causally or temporally related. However, it can look at successive frames of a video or film and understand that there is a temporal sequence to what is being depicted. For example, it can look at an image of an ice cube that has fallen on the floor and a second image of a small puddle of water on the floor in the appropriate location and understand that ice has melted to form the puddle (i.e. that the first image depicts a scene in time earlier than the second). Real-world events unfold in time. Understanding this, even partially, is a good quality to have. Yang et al. provide an example of making sushi.

Sec. 6.2 Video Understanding

Temporal Ordering

Prompt:

Analyze the images and decide the correct order in which they should be displayed.

a



b



c



Prompt Continued

d



e



GPT-4V:

The correct order in which the images should be displayed is:

1. e - Mixing the rice
2. c - Placing the rice on the seaweed
3. b - Adding the fillings
4. a - Cutting the roll
5. d - Finished sushi rolls on a plate.

Figure 16. Sushi Making— a Screenshot of Figure 54 from (Yang et al. 2023).

5.3.11 Answer Intelligence Quotient (IQ) Tests

GPT-4V can do this, in part and to a degree. It performs better when the tasks or challenges are broken down into sub-tasks.

5.3.12 Avoid False Presuppositions

It can avoid being misled by false presuppositions (e.g. ‘where is the soccer ball in the image?’ where, actually, there is no soccer ball present).

5.3.13 Navigate Real and Virtual Spaces

An example might help here.

Imagine these two scenarios: you arrive at an AirBNB and wonder if there is milk in the fridge, then, secondarily, you wonder where on the web there is a online retailer, physically nearby, that can deliver milk to you. A robot with the appropriate mechanical and sensing abilities might well be able to start at the front door of your AirBNB and answer the milk question for you. The Yang and al. paper shows an example of GPT-4V doing a similar planning and navigating task. Of course, the researchers do have supply suitable images responding to GPT-4V wanting to ‘turn left’ or ‘right’ etc.— i.e. they have to simulate the changing dynamic environment. They also have a second example illustrating GPT-4V navigating the web. This time GPT-4V can ‘click a mouse’, ‘scroll’, etc. Then, in a somewhat tentative and first attempt fashion, GPT-4V can navigate the web and find and order whatever you wish (be it milk or whatever).

There is no pretense here that this navigation, planning, and problem solving is a *fait accompli*. However, this is an important first step in working with dynamic environments or sequences of images portraying dynamic environments.

This has some relevance to librarianship. GPT-4V may well be able to navigate websites or the web to find information resources (even when those resources are not directly indexed or linked to).

5.4 Yang et al.'s Conclusion on GPT-4V

Yang et al. write:

In conclusion, OpenAI's GPT-4Vision marks a monumental step towards harmonizing text and image understanding, paving the way for more intuitive and enriched interactions between humans and machines. As GPT-4V unfolds its potential, it not only broadens the horizon for real-world applications but also beckons a future where AI can perceive and interpret the world in a manner akin to human cognition, thereby significantly driving forward the frontier of what is achievable in the realm of artificial intelligence (Yang et al. 2023).

Then, to paraphrase some of their other assertions on what the future might hold for developments of the OpenAI GPT series. GPT-4V adds vision to LMMs. It should be possible, in short order, to add video, audio, and other sensor data into the mix. Separately, most of the learning of the LMMs thus far has come from text, principally self-supervision from vast amounts of text followed the reinforcement learning using humans. But it should be possible from LMMs to augment their base learning with

learning from modalities other than text. For example, to do some learning from real-world physical environments. We are a year or so into the future from the Yang paper and Yang et al. have been proved exactly right in their prognostications.

5.5 GPT-4 Turbo (Early 2024)

Early in 2024, GPT-4 Turbo was the core component of the ChatGPT family (OpenAI Platform 2024). GPT-4 Turbo:

- has the visual capabilities of GPT-4V built in,
- has the output capabilities of the image generator Dall-E 3 built in. (We have not discussed image generators, but Dall-E 3 can produce images from text prompts.),
- will accept text-to-speech requests. [The extent to which it can handle speech-to-text is unclear. It *cannot* conduct a dialog at the speed of ordinary speech]
- is trained on information with a cut-off date of April 2023. So, its knowledge is more current than earlier GPTs.
- has a context window of about 128,000 tokens. That is about the equivalent of a book of 300 pages. This allows for longer input prompts. Having such a large context means that it can consider a reasonably sized book all at once (for example, for summarizing).
- it is priced at 2-3 times cheaper to run than earlier versions.

5.6 GPT-4o (Later 2024)

On 5/13/2024 OpenAI published the web page *Hello GPT-4o : We're announcing GPT-4o, our new flagship model that can reason across audio, vision, and text in real time* (OpenAI 2024).

GPT-4omni can do all the tasks of earlier models, many with improvements, but also it can work across the modalities in 'real time'. This means, for example, that it can converse with a person at normal human speed. This makes a huge difference. Earlier OpenAI LLM based chatbots were slow in their responses. This made their conversation somewhat unnatural, artificial, and tiring.

5.7 Google's Gemini

On 12/6/2023 Google introduced Gemini, which is an LMM (Pichai and Hassabis 2023). Sundar Pichai and Demis Hassabis assert:

... [Gemini's] performance exceeds current state-of-the-art results on 30 of the 32 widely-used academic benchmarks used in large language model (LLM) research and development (Pichai and Hassabis 2023).

We are perfectly content to accept that this is true as of 12/6/2023. It means that Gemini was probably the best performing LLM or LMM at that point. The caution needs to be added that there is strenuous competition in these areas so rankings may change through time. The situation here with Gemini is a little odd. Gemini is going to be released in 3 forms, but the best

version will not be available until 2024. Apparently, the comparisons with OpenAI's GPT versions was with older versions, and, apparently, Google's video extolling Gemini's was selectively edited, polished, or 'faked'. There is a paper, 12/18/2023, by Syeda Nahida Akter and fellow authors, *An In-depth Look at Gemini's Language Abilities* which tests Gemini Pro (the middle level version) against various other LLMs (Akter et al. 2023). It suggests that Gemini Pro is at about the level of GPT-3.5 Turbo (i.e. the mid-level Gemini is a little off the pace).

So, something good may be coming but we do not really know and it was not available moving into 2024.

5.8 Anthropic's Claude

As of 7/7/2024, Anthropic's most advanced multimodal LLM system is Claude Sonnet 3.5 (Anthropic 2024). Many commentators regard Sonnet 3.5 as being the best LLM available, from any manufacturer. Such judgements are temporary and can change as new releases appear. Sonnet can do all the typical multimodal tasks, but faster and to a higher quality than other models. [We will mention some benchmarks later.]

One of the key features of Claude is its emphasis on ethical AI use and safety. Anthropic has embedded rigorous safety protocols and guidelines within Claude to mitigate potential risks associated with advanced AI systems. This includes measures to prevent harmful outputs, ensure user privacy, and promote transparency in AI interactions. Claude's architecture and training regimen are designed to foster responsible AI behavior,

making it a reliable tool for users across various domains. By prioritizing safety and ethical considerations, Anthropic aims to set a new standard in the AI industry, ensuring that advancements like Claude contribute positively to society. [This paragraph was written by ChatGPT.]

5.9 Meta's LLaMa

Meta's LLaMA 3 (Large Language Model Meta AI) is a large language model in roughly the same category as the GPTs, Claudes and Geminis (Meta 2024). It is not multimodal, nor is it multilingual (7/7/2024). Although, those capabilities are under active development. Its performance is not to the standard of the over top-rank LLMs. It may be a year or so behind. However...

A standout feature of LLaMA is its dedication to accessibility and transparency. Meta has taken steps to ensure that LLaMA's capabilities are available to a broad audience, promoting an open research culture where insights and improvements can be shared widely. This approach is intended to foster collaboration and innovation within the AI community, driving forward the collective understanding of AI and its applications. Additionally, Meta has prioritized the ethical deployment of LLaMA, embedding safety measures and guidelines to mitigate potential misuse and ensure responsible use. By focusing on both performance and ethical considerations, Meta aims to establish LLaMA as a leading example of how large language models can be both powerful and principled. (written by ChatGPT).

The LLaMA models are open-source and available to all (through Hugging Face and other outlets).

5.10 Voice

The use of voice was demonstrated with GPT-4o in May 2024 (but it has been slow to rollout to Users). It is reasonable to assume that the other manufacturers have something similar, and that real time voice interactions between Users and AI systems shortly will be available across all platforms. What is in prospect here are:

- AI virtual counsellors, companions, and friends
- AI virtual assistants as exemplified in the 2013 film *Her* (Wikipedia 2024)
- Improved customer support and help
- Etc.

5.11 Possible Applications for LMMs

5.11.1 Smartphone Uses

GPT-4V can transcribe printed or handwritten text. (What the limits are here are not entirely clear, but there certainly is reasonable success at this.) Having text in the 0s and 1s of computer processing is valuable for many

uses (as we emphasized in the Paean to this in Chapter 1). This ability of GPT-4V can be used in tandem with camera of a smartphone. Then users will be able to carry the means of transcription in their purses or pockets.

5.11.2 Spot the Difference

GPT-4V can spot the difference between two images. This means that in many settings it will be able to spot the defects in products of industrial manufacturing processes.

5.11.3 Producing Reports from Medical Images

GPT-4V can do this, although not perfectly. It is not of the requisite quality at this time. The capabilities of LMMs here will improve, but, at present, their role would be that of being intelligent assistants to expert doctors and consultants.

5.11.4 Assist with Image Generation

GPT-4V can consider the results of other image generation programs (Dall-E, Stable Diffusion, etc.). In particular, it can consider the prompts and the relevant images and evaluate them. This might provide an evaluation model which can then be used for reinforcement learning for the image generation

programs themselves. Also, it can simply make suggestions on how to improve the prompts. It can rewrite the prompts.

5.11.5 Extension with Plugins

It is sensible to hand of certain kinds of tasks to outside software or ‘plugins’. For example, GPT-4V has training data covering a certain time-period. It can be revised and updated, but it is not going to be given new training every day or every hour. A strategy here is to have a plugin that can supply up to date news. A second example are payments or, more generally, services. Credit card companies can manage online payments. But these would not be built in to a core version of GPT-4V, rather they would be a way of extending the capabilities for certain purposes. Plugins are a part of standard commercial LLM architecture. Presumably they would become a part of LMM architecture (and some of the plugins may be multi-modal).

5.11.6 Retrieval-Augmented Generation (RAG)

There is a technique, Retrieval-Augmented Generation (RAG), which is functionally somewhat similar to plug-ins (Lewis et al. 2021; Gao et al. 2023). It is to keep an LLM up to date with what it ‘knows’ and to be more accurate in its replies. The idea is to give the LLM access to an external database or databases. Then factual prompt questions to the LLM are augmented with the instruction to check with the databases and find supporting facts, references, and citations. As external knowledge grows

there is no need to re-train the LLM. Rather, all that is required is for the databases to be updated (which they usually would be as a matter of course, say for news articles and similar).

5.11.7 Label and Categorize Images

Adding metadata to images is a challenging problem for librarianship. GPT-4V has the potential to be valuable here. There is value also for ordinary people being able to label, categorize, and sort, for example, the images on their smartphones. This latter task may in part conflict with the safety feature of not identifying people in images.

5.11.8 Identify Objects

This has many uses: from identifying plants and trees for gardeners, hobbyists, or farmers through to military applications.

5.11.9 ‘Igor’, AI Advantage and AI Community

There are more new use cases for any one person to keep up with the possibilities. My suggestion on how to learn about your focus of interest would be to look at the videos available on *AI Advantage* and at the materials in *AI Community* (AI Advantage 2024; AICommunity 2024). Igor is the prime mover of these.

5.12 Annotated Readings for Chapter 5

Altman, Sam, dir. 2023. OpenAI DevDay, Opening Keynote.

<https://www.youtube.com/watch?v=U9mJuUkhUzk>. (Altman 2023)

Berman, Matthew. “Intro to RAG for AI (Retrieval Augmented Generation),” 2024.

https://www.youtube.com/watch?v=Yo8Nn23o_mY. (M. Berman 2024) .

Matthew Berman has produced a number of excellent videos on the practical implementations of LLMs (focusing especially on implementations that run locally on the User’s computer).

Gozalo-Brizuela, Roberto, and Eduardo C. Garrido-Merchan. “ChatGPT Is Not All You Need. A State of the Art Review of Large Generative AI Models.” arXiv, 2023.

<https://doi.org/10.48550/arXiv.2301.04655>. (Gozalo-Brizuela and Garrido-Merchan 2023). The publication here is January 2023 which, sad to say, is getting old now for a state of the art review.

Teixeira, Lawrence. “The New Open AI GPT-4 Vision on ChatGPT: Bridging the Gap Between Text and Image Understanding,” 2023.

<https://medium.com/@lawrenceteixeira/the-new-open-ai-gpt-4-vision-on-chatgpt-bridging-the-gap-between-text-and-image-understanding-9337ed4c1a61>. (Teixeira 2023). This is a short and clear explanation of what GPT-4 V can do.

The AI Advantage, dir. 2023. 100+ Insane ChatGPT Vision Use Cases.

<https://www.youtube.com/watch?v=ywNNRzc7-To>. (The AI Advantage 2023). This is an excellent video presenting the contents of this chapter in 26 minutes.

Yang, Zhengyuan, Linjie Li, Kevin Lin, Jianfeng Wang, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. “The Dawn of LMMs: Preliminary Explorations with GPT-4V(ision).” arXiv, 2023. <https://doi.org/10.48550/arXiv.2309.17421>. (Yang et al. 2023). This is the standard text on this topic. It is, though, 166 pages long, with many references. It is a research paper and probably not entirely suitable for us.

Chapter 6: Evaluation and the Future

6.1 Reliability, Trustworthiness, and Alignment

We all know what the word ‘reliable’ means in ordinary speech. There is the element of consistency to it and the element of trustworthiness. If an old car is reliable, it needs to be able to do what cars typically do and potentially to be able to do that on more than one occasion.

In the theory of research methods in statistics, there are the concepts of reliability and validity. These are concepts that are applied to instruments or observations. Reliability is consistency under repeat trials. Validity is truth or veracity. Say a person weighs 180lbs and they regularly use some specific bathroom scales. If those scales indicate 182lbs on many uses by the person, while the person’s weight is unchanged, the scales are reliable. Were the scales to have indicated 30lbs on repeat trials, the scales would still be reliable even though the readings are in the realm of fantasy. But if they were to indicate 178lbs on one occasion and 182lbs on another, they would be unreliable. If the scales measured 180lbs, that measurement would be valid. If sometimes measurements produced two different values, at least one of those would have to be not valid. Two measurements that contradict cannot both be valid. The gold standard is for an instrument to be reliable and to produce only values that were valid. Reliability, together with validity, are the aims. Notice that there is little or no trust or trustworthiness in statistical reliability (apart from the trust that repeat trials produce the same result).

In machine learning, modern AI uses primarily concepts of reliability and alignment. Generally, researchers in ML are experts on statistics and are familiar with statistical reliability. But in practice, in their writings, AI reliability often has a similarity to common sense reliability i.e. repeatability plus trustworthiness.

Predictions, or measurements, or outcomes, in AI are often used in settings where probability is involved. For example, completing the sentence ‘The cat sat on the [Mask]’ might produce ‘The cat sat on the mat’ on one occasion and ‘The cat sat on the pillow’ on another. The outcomes are different. But this does not have to mean that the LLM in question is unreliable. There are probabilities associated with the next words of: ‘mat’, ‘pillow’, ‘blanket’, etc. For the LLM to be reliable here, it needs to do the completions in line with the probabilities. Not all outcomes from LLMs involve probabilities, but many do. In general, assessing reliability in the context of AI is tricky.

Alignment is that a model's predictions or behavior correspond closely with the expected or desired or intended outcome. This can be similar to validity in statistical research methods, but it can be different also. Validity focusses on truth. But, for example, writing poetry, as an LLM might do, does not have much use for truth. Elsewhere, there are questions of safety, bias, and offensive language. Generally, we would not always want LLMs offering perhaps correct medical advice, nor producing perhaps correct recipes for constructing bombs, nor telling people, perhaps truly, that they are ‘fat’. Getting AI systems to do what we would like is alignment.

6.2 System 1 and System 2

In 2011, the behavioral scientist Daniel Kahneman published *Thinking, Fast and Slow* (Kahneman 2011). In it, the distinction is made between System 1 thinking and System 2 thinking. System 1 thinking is fast, intuitive, near automatic, and requires little effort. System 2 thinking is slower, more logical and usually requires attention and deliberation. Catching a ball lobbed gently from a few yards away uses System 1. In contrast, a grandmaster's move in the middle of a chess game uses System 2. (See also, (Loo 2024).)

There is an analogy and speculation that can be used here in connection with LLMs. LLMs typically do next word prediction, or, more generally, next token prediction. They have been trained on a vast amount of data, sometimes close to the entire Internet. In effect, they have skimmed a lot of books then take a prompt and return the next word (and the word after that, etc.). From an analogical point of view, this is System 1 thinking. At this point of development, we could probably concede that LLMs can match humans on System 1 tasks.

What about System 2? What about logical reasoning with deliberation? Of course, computers can play chess, and similar, better than any humans. But chess playing computers do not use large language models. They essentially brute force the problems, which is a different approach. [The Zhores supercomputer used by Nepomniachtchi in training for his 2021 world title chess match, can evaluate tens of millions of chess positions per second.

Basically, it overwhelms the problem with computation and data.] Existing benchmarks on LLMs suggest that at they are weak on reasoning. It may be that LLMs on their own will always struggle on System 2 tasks.

6.3 Benchmarks

6.3.1 Introduction

One way of getting an understanding of what LMMs can do is by looking at common benchmarks. Talking casually, the moment an LMM displays an ability researchers will produce a benchmark to measure that ability. One motivation the researchers have here is to compare the performance of an LMM either against earlier versions of itself or against rival LMMs. For our purposes, we are not such much interested in the results under the benchmarks as we are in the benchmarks themselves— the abilities that the benchmarks are supposedly testing for.

Benchmarks can have drawbacks. For example, the LLMs can be trained with one eye on the benchmarks (‘training to the test’). Also, some of the data, patterns or techniques shown in the tests can bleed back into an LLM (for example, by being used in the training).

6.3.2 Multi-turn dialogs

That is, the user gives a prompt, the LMM replies, then user then gives another prompt (in the context of the earlier prompt and reply), and so on.

MT-Bench is an example of testing for this:

MT-bench is designed to test multi-turn conversation and instruction-following ability, covering common use cases and focusing on challenging questions to differentiate models. We identify 8 common categories of user prompts to guide its construction:

- writing,
- roleplay,
- extraction,
- reasoning,
- math,
- coding,
- knowledge I (STEM),
- and knowledge II (humanities/social science)

(Zheng et al. 2023)

6.3.3 Chatbots

One possibility of a benchmark here is Chatbot Arena (Chiang et al. 2024). This allows users to run two rival chatbots side-by-side, compare the results, and vote on the outcomes. The benchmark does not so much tell you what the chatbots can do, as reveal human preferences. But, to an extent, the preferences are a measure of how well the chatbots do what the users would like, or expect, them to do. So, in some perhaps weak sense, the benchmark measures alignment.

6.3.4 Reasoning

There is the A12 Reasoning Challenge (A12 Allen Institute for AI 2022):

The ARC dataset contains 7,787 genuine grade-school level, multiple-choice science questions, assembled to encourage research in advanced question-answering....

...in particular questions that require reasoning, use of commonsense knowledge, and other methods for deeper text comprehension.... those that are hard to answer with simple baselines.

Example ARC Question

Which property of a mineral can be determined just by looking at it?

- **luster**
- mass
- weight
- hardness

6.3.5 Common sense reasoning

Another popular benchmark for common sense reasoning is HellaSwag (Zellers et al. 2019). Its dataset presents multiple-choice questions where models must choose the most plausible continuation or outcome of a given situation described in text. This requires models to infer logical conclusions from the context provided. Here is an example.

Context: A person is standing in a kitchen with a cutting board and a knife. They pick up an apple and begin to...

Options:

- A. ...throw it across the room to their friend who catches it.
- B. ...cut it into slices to make a fruit salad.
- C. ...put it in their pocket and leave the kitchen.
- D. ...start peeling it with a vegetable peeler.

Answer: B. ...cut it into slices to make a fruit salad. This is the most plausible continuation given the context.

HellaSwag's dataset has been developed since its original proposal in 2019.

6.3.6 MMLU

The MMLU (Massive Multitask Language Understanding) benchmark is designed to evaluate the multitask accuracy of large language models across a wide range of subjects and tasks. It includes questions from 57 different subjects, spanning areas such as humanities, social sciences, STEM, and more. The benchmark tests models' ability to understand and generate accurate responses across various domains, reflecting their general knowledge and reasoning capabilities. The structure of the MMLU benchmark typically involves multiple-choice questions, with each subject having its own set of questions. These questions are often at the level of difficulty encountered in real-world exams or standardized tests.

An devised example of an MMLU question might look like this:

Subject: History

Question: Who was the first President of the United States?

Options:

- A. Abraham Lincoln
- B. Thomas Jefferson
- C. George Washington
- D. John Adams

Answer: C. George Washington