UNITED STATES DISTRICT COURT
SOUTHERN DISTRICT OF NEW YORK

IN RE: OPENAI, INC. COPYRIGHT INFRINGEMENT LITIGATION	1:25-md-03143 (SHS) (OTW) **DECLARATION OF EXPERT CHRIS CALLISON-BURCH IN SUPPORT OF DEFENDANTS' MOTION FOR SUMMARY JUDGMENT** REDACTED — PUBLIC VERSION

I, Chris Callison-Burch, Ph.D., declare under penalty of perjury, pursuant to 28 U.S.C. § 1746, as follows:

1. I submit this Declaration in support of OpenAI Defendants' (collectively, "OpenAI") Motion for Summary Judgment.

2. My name is Chris Callison-Burch. I am the Raj and Neera Singh Professor of Artificial Intelligence at the University of Pennsylvania. From 2007 to 2013, I conducted research and supervised Ph.D. students at Johns Hopkins University, first as an Assistant Research Professor, and then as an Associate Research Professor. In September 2013, I joined the faculty of the University of Pennsylvania as an Assistant Professor. I became a tenured Associate Professor in 2017, and I was appointed Professor in July 2024.

3. I received a Bachelor of Science degree in Symbolic Systems from Stanford University in 2000, a Master of Science degree in Computer Science from the University of Edinburgh in 2002, and a Ph.D. in Informatics from the University of Edinburgh in 2008. My Ph.D. thesis investigated the relationship between paraphrasing and translation.

4. Throughout my career, I have been a member of numerous professional associations related to the field of AI, including the Responsible AI Academic Council, the Association for Computational Linguistics, and the Conference on Empirical Methods in Natural Language Processing. I currently serve as the Chair of the Advisory Board of the Human Language Technology Center of Excellence at Johns Hopkins University, which facilitates collaboration between academia and government in order to enhance the government's utilization of AI innovations.

5. My current research focuses on the application of large language models ("LLMs") to long-standing challenges in the field of AI. I have published more than 200 articles, which have been cited over 30,000 times.[1]

[1] CCB Opening Report ¶ 23.

6. I have received research awards from Google, Microsoft, Amazon, Facebook, Salesforce, and Roblox, in addition to funding from the Defense Advanced Research Projects Agency (DARPA), the Intelligence Advanced Research Projects Agency (IARPA), and the National Science Foundation (NSF).

I. ASSIGNMENT

7. I have been asked by counsel for OpenAI to address technical issues relevant to its fair use defense. My opinions address technical issues related to the first fair-use factor—the purpose and character of the use and whether the use was transformative—and may also be relevant to other fair-use factors.

8. For purposes of this declaration, I collectively refer to OpenAI's LLMs at issue in this matter as the "In-Scope LLMs," which include OpenAI's GPT-3, GPT-3.5, GPT-3.5 Turbo, GPT-4, GPT-4 Turbo, GPT-4o, and GPT-4o mini LLMs.

9. I previously prepared the Expert Report of Chris Callison-Burch, Ph.D., dated April 24, 2026 ("CCB Opening Report"), the Expert Rebuttal Report of Chris Callison-Burch, Ph.D., dated May 29, 2026 ("CCB Rebuttal Report"), and the Expert Reply Report of Chris Callison-Burch, Ph.D., dated June 26, 2026 ("CCB Reply Report") in this matter. I was deposed in this matter on July 15, 2026, and July 16, 2026.

II. HISTORY OF NATURAL LANGUAGE PROCESSING

10. To understand the specific technical issues in this case, it is helpful to have some background knowledge regarding natural language processing and LLMs. Natural Language Processing ("NLP") is a field of scientific development at the intersection of linguistics and computer science. Modern LLMs are the latest innovation in NLP technology.[2]

11. NLP developments established both the foundational statistical theories underlying LLM technologies and language processing and the need for large corpora demonstrating language usage.[3] The statistical approach to language processing can be traced back to early attempts to use probability to study language in the 1910s. In 1913, a Russian mathematician named Andrey Markov counted the first 20,000 letters in a novel to test his probability theory, and learned that language had a system where past words partially determined the next word that could be uttered. In 1948, Claude Shannon built on Markov's insight to develop a more complex statistical model of English that generated text by estimating probabilities of words following from one another based on a corpus of text. He found that as these models became more sophisticated, they generated text that resembled ordinary English text.[4]

12. In the early 1980s, a team at IBM developed an early example of a computer-based application of these statistical machine-learning methods. IBM assembled large, machine-readable text datasets (called "corpora," singular "corpus"), which they processed with computers. They relied on documents from the U.S. Patent Office, text from the Associated Press, novels,

[2] CCB Opening Report ¶ 63.
[3] CCB Opening Report ¶ 67.
[4] CCB Opening Report ¶ 68–69.

periodicals, and eventually Canadian Parliament proceedings translated between French and English.[5]

13. IBM's corpora were significantly smaller than those used in modern LLM training, and the models it built with them were limited to narrow tasks. Nonetheless, IBM's rudimentary language models demonstrated two key insights: (1) the statistical approach to language recognition was significantly more successful than rule-based systems (*e.g.*, decision trees with predetermined responses); and (2) that approach required access to a very large and diverse corpus of text.[6]

14. Since the advent of the World Wide Web in the 1990s, web crawling and web scraping technologies developed for search engines like Google have become essential tools for researchers gathering text data to build language models.[7] Web crawling is the process in which automated bots recursively collect content from websites, in order to build searchable indexes of the web. Web scraping is the process in which bots or web crawlers extract large amounts of data from websites and convert that data into reformatted, usable text.

15. Various research developments in the 1990s and 2000s established the effectiveness of web-scale data in NLP. For example, researchers demonstrated how to automatically extract parallel text (*i.e.*, the same text in two different languages) from multilingual websites for machine translation; showed that simple algorithms trained on massive web-scraped corpora could outperform more sophisticated methods trained on smaller datasets; and used web data to improve language models for speech recognition systems.[8]

16. Throughout the following decades, researchers continued to develop more advanced text corpora for NLP research. Academic researchers assembled large-scale, linguistically annotated corpora from web crawls that aimed to cover a variety of text types and topics to adequately represent a language.[9] Major technology companies also began sharing portions of their web-scraped data with the NLP research community. For example, Google released the Google Books Ngram Corpus, a dataset of approximately 500 billion words extracted from over 5 million digitized books that enabled a new form of computational lexicology that studies cultural trends through analysis of digitized texts.[10] Academic researchers assembled large book collections for NLP research, and a nonprofit organization called Common Crawl created a web index of monthly crawls of the Internet as part of its mission to make analysis of web data accessible to developers and the public. As of 2025, Common Crawl includes over 250 billion pages.[11]

17. As researchers moved from early academic experiments to large-scale corpora drawn from web pages, news, books, Wikipedia, and Common Crawl, web-scraped text became an essential resource for advancing language-understanding systems. Most, if not all, NLP

[5] CCB Opening Report ¶¶ 70–74.
[6] CCB Opening Report ¶¶ 73–75.
[7] CCB Opening Report ¶¶ 76–77.
[8] CCB Opening Report ¶¶ 78–80.
[9] *Id.*
[10] CCB Opening Report ¶ 81.
[11] CCB Opening Report ¶¶ 82–83.

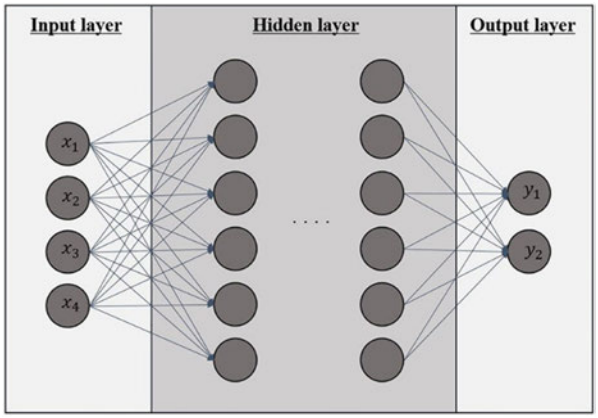
applications were trained on data scraped from the Internet. Commercial machine translation systems, such as Google Translate, pioneered the use of unlicensed web-scraped text and digitized books for training language models in the early 2000s. Many commercial NLP applications, such as speech recognition systems like Google Voice Search, digital assistants like Amazon Alexa and Apple's Siri, and spelling and grammar checkers built into Microsoft Word and Google Docs, followed the same pattern and are now part of everyday life.[12] By 2016, web-scale training data from Common Crawl and news corpora had become industry standard for commercial translation systems.[13] As these examples show, web-scraped text, digitized books, and user-generated content were standard training materials for commercial language technologies long before modern LLMs.[14]

III. OVERVIEW OF LLMS

A. LLM Architecture

18. Unlike a database or search engine that stores or indexes documents and retrieves copies in response to a query, LLMs learn statistical patterns from text and generate new sequences from scratch, one token at a time.[15]

19. The architecture of an LLM consists of a ***neural network***, a computer system for processing information that consists of interconnected nodes, or "neurons."[16] Within a neural network, input data is converted into numerical values that a computer can interpret and is transmitted through interconnected layers: (1) the input layer, which is the starting point for processing; (2) hidden layers, which are intermediate layers that perform calculations on the input data; and (3) an output layer where the model provides the final result or prediction. The exhibit below illustrates the key components of a simple neural network:[17]



[12] CCB Opening Report ¶¶ 84–93.
[13] CCB Opening Report ¶ 86.
[14] CCB Opening Report ¶¶ 86–87.
[15] CCB Opening Report ¶¶ 96–98.
[16] CCB Opening Report ¶¶ 101–102.
[17] Sujin Pyo et al., *Predictability of Machine Learning Techniques to Forecast the Trends of Market Index Prices: Hypothesis Testing for the Korean Stock Markets*, PLOS One, 4 (Nov. 2017), https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0188107.

20. The lines connecting the neurons in the diagram above are "connections" and each is assigned a "parameter" or "weight," a numerical value that indicates the strength and type of relationship between the neurons.[18] Each weight acts like a dial or knob that can be tuned to affect the influence of different features, helping the model favor certain predictions over others. This "tuning" occurs automatically during the network's training, but at the outset, can be illustrated as a grid of random numbers.[19]

...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
...	0.24	0.87	0.16	0.01	1.10	1.95	0.05	0.24	0.87	0.16	0.01	1.10	1.95	0.05	0.24	0.87	0.16	0.01	1.10	...
...	0.81	1.33	1.21	1.89	0.91	1.18	0.20	0.81	1.33	1.21	1.89	0.91	1.18	0.20	0.81	1.33	1.21	1.89	0.91	...
...	0.08	0.83	0.52	1.34	0.12	0.65	0.71	0.08	0.83	0.52	1.34	0.12	0.65	0.71	0.08	0.83	0.52	1.34	0.12	...
...	1.14	1.46	0.86	0.69	1.25	0.04	0.62	1.14	1.46	0.86	0.69	1.25	0.04	0.62	1.14	1.46	0.86	0.69	1.25	...
...	0.67	0.31	1.56	0.68	1.85	0.05	0.56	0.67	0.31	1.56	0.68	1.85	0.05	0.56	0.67	0.31	1.56	0.68	1.85	...
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

21. Consider, as a simplified illustrative example, a model designed to determine whether a movie review is positive or negative. Each possible word (or "token") in the model's vocabulary is a different "input," such that if there are 5,000 words in the model's vocabulary, there will be 5,000 inputs (*e.g.*, x_1 to x_{5000}). The training data for such a model might consist of a labeled dataset of thousands of movie reviews and, for each, a "positive" or "negative" label. An example consisting of the words, "this movie is brilliant," would trigger the neurons corresponding to the words "this," "movie," "is," and "brilliant." The weights of those neurons would then be summed together ("summation"), and then the sum would be passed through an activation function before generating a prediction ("positive" or "negative"). During training, the model will encounter thousands of these examples and, at each training step, make adjustments to the weights corresponding to each neuron. Throughout this process, the weights will come to reflect the fact that certain words, such as "brilliant" or "masterpiece," carry significant weight toward a positive prediction.[20] However, the weights do not store information about any particular review. Rather, the network merely reflects the fact that, for example, a movie review that uses the word "brilliant" has a high probability of being a positive review. Therefore, one cannot simply look up a "fact" that a neural network has learned the way one might search for data in a database.[21]

22. While the simplified diagram above depicts only a single layer and a handful of weights, modern neural networks or "deep" networks, which encode much more complicated and abstract features, may have hundreds of layers and billions of weights among which statistical patterns are distributed. A "deep" network can recognize increasingly abstract features of its training data, such as semantic relationships between words, logical relationships between sentences, and overarching themes. This hierarchical learning makes deep networks powerful for complex tasks, such as generating coherent language, answering questions, and reasoning across

[18] CCB Opening Report ¶ 103.
[19] *See* CCB Opening Report ¶ 105.
[20] CCB Opening Report ¶ 105.
[21] CCB Opening Report ¶¶ 107–108.

long passages. Moreover, neural networks can learn these patterns and skills without the need for predefined human-programmed rules.[22]

23. The data on which a neural network is trained consists of "***tokens***." A token is a small unit of text that corresponds to a number. A token may be a whole word ("cat"), part of a word ("un-" or "-ing"), a punctuation mark, or a single character.[23]

24. ***Tokenization***, the process by which text or other inputs are transformed into tokens, allows the model to handle new or rare words by combining familiar pieces.[24] For example, even if the model has never seen the word "hyper-curiousness," it may recognize "hyper," "curious," and "ness" and infer meaning from those parts.

25. Each unique token in a language model's vocabulary will be associated with, or "***embedded***" in, a set of numbers called a "***vector***." A vector can be thought of as coordinates that place each word at a specific point in a mathematical space, similar to how one might describe a city's location using latitude and longitude.[25]

26. The values in a word's "vector" are not set manually; rather, the model "learns" them through exposure to enormous amounts of text.[26] Before the training process, the numbers in each vector are random values. Throughout the training process, the numbers are gradually adjusted to reflect the word's semantic meaning.[27] The model also learns relationships between words, such that words with similar meanings end up located near each other in this mathematical space. As with the "neural network" example above, the purpose of this training process is to encode abstract statistical information in the vectors' numbers—here, corresponding to the semantic meaning of each word.[28]

27. A ***transformer*** is a type of neural network architecture that enriches the embedding vectors described above by adding contextual information through a repeated two-step process.[29] Prior to the development of transformers, researchers used a type of neural network called Long Short-Term Memory networks, which read text sequentially, one word at a time, and maintain a kind of "memory" of what they read.[30] Processing words one after another was slow and made it difficult for the network to connect ideas that appeared far apart in a text because earlier information had to be carried forward through many intermediate updates. Transformers solved this problem.[31] *First*, through a mechanism called "self-attention," each word's vector is updated by examining every other word in the input to determine which surrounding words are most relevant to its meaning in the input's particular context.[32] For example, self-attention can update the vector for "bank" to reflect a *river bank* or an *investment bank*, depending on the other words

[22] CCB Opening Report ¶ 106.
[23] CCB Opening Report ¶ 109.
[24] CCB Opening Report ¶¶ 110–112.
[25] CCB Opening Report ¶¶ 113–115.
[26] CCB Opening Report ¶¶ 116–117.
[27] *Id.*
[28] CCB Opening Report ¶¶ 118–122.
[29] CCB Opening Report ¶ 123.
[30] CCB Opening Report ¶ 125–127.
[31] *Id.*
[32] CCB Opening Report ¶ 123.

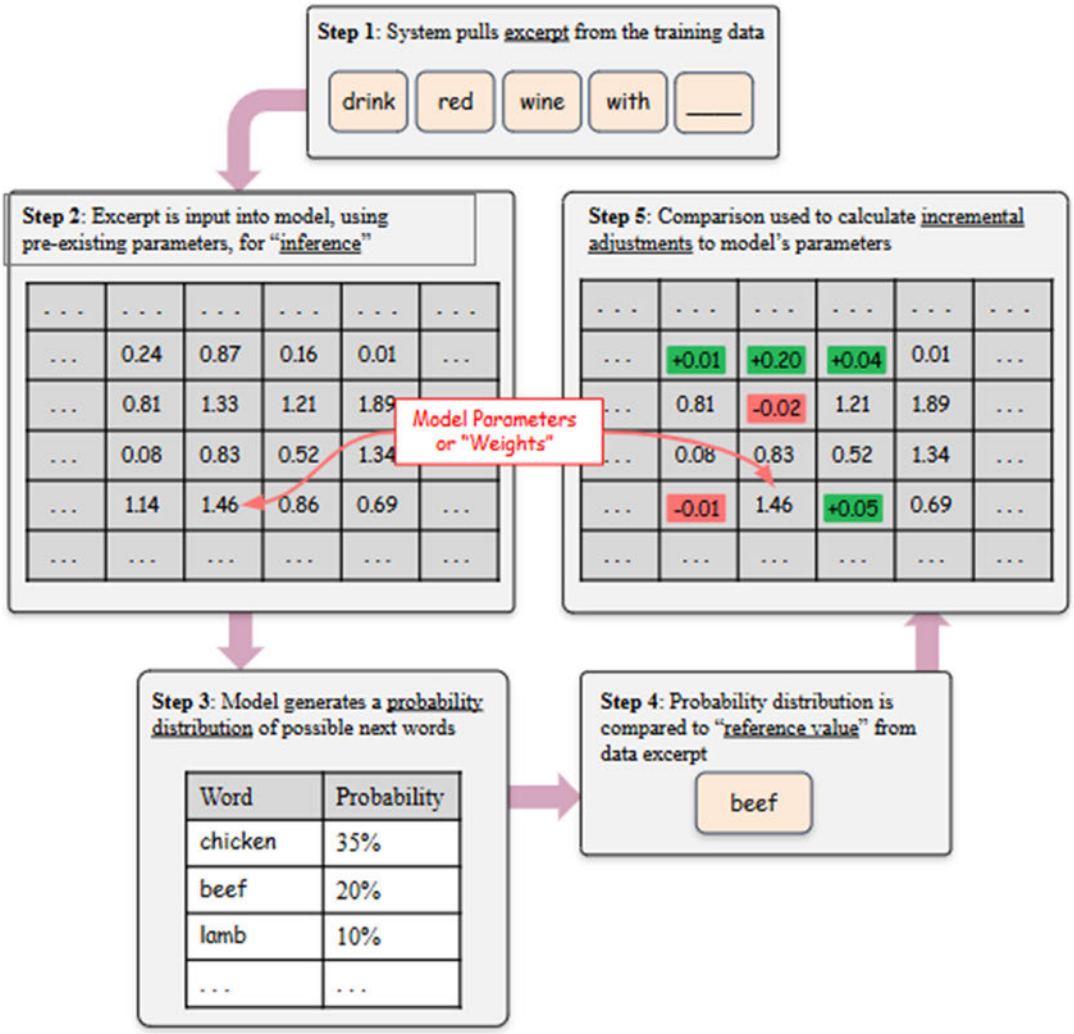
in the sentence. In this way, the model examines entire sequences of words simultaneously, and assesses the meaning of each word based on every other word in the sequence. *Second*, a neural network layer (called a "multi-layer perceptron" or "MLP") applies additional transformations that enable the model to draw higher-level ***inferences*** from those contextual relationships. Modern language models stack dozens of these two-step blocks on top of one another, with each successive layer further refining the representations.[33]

28. The attention mechanism computes how relevant each word is to every other word within the sequence being processed. Words that are more relevant to determining a given word's meaning contribute more to its final representation. This allows the model to focus on the most important context for understanding each word, regardless of how far apart words appear in the text.[34]

29. LLMs like OpenAI's GPT models are built using the building blocks described above. During pre-training, the model is provided with a randomly sampled, contiguous sequence of tokens as an "input sequence." The input sequence would then flow through multiple stacked transformer blocks. The model would then generate a set of probabilities for each possible next token in the vocabulary, and the prediction was compared against the true next token in the dataset.[35]

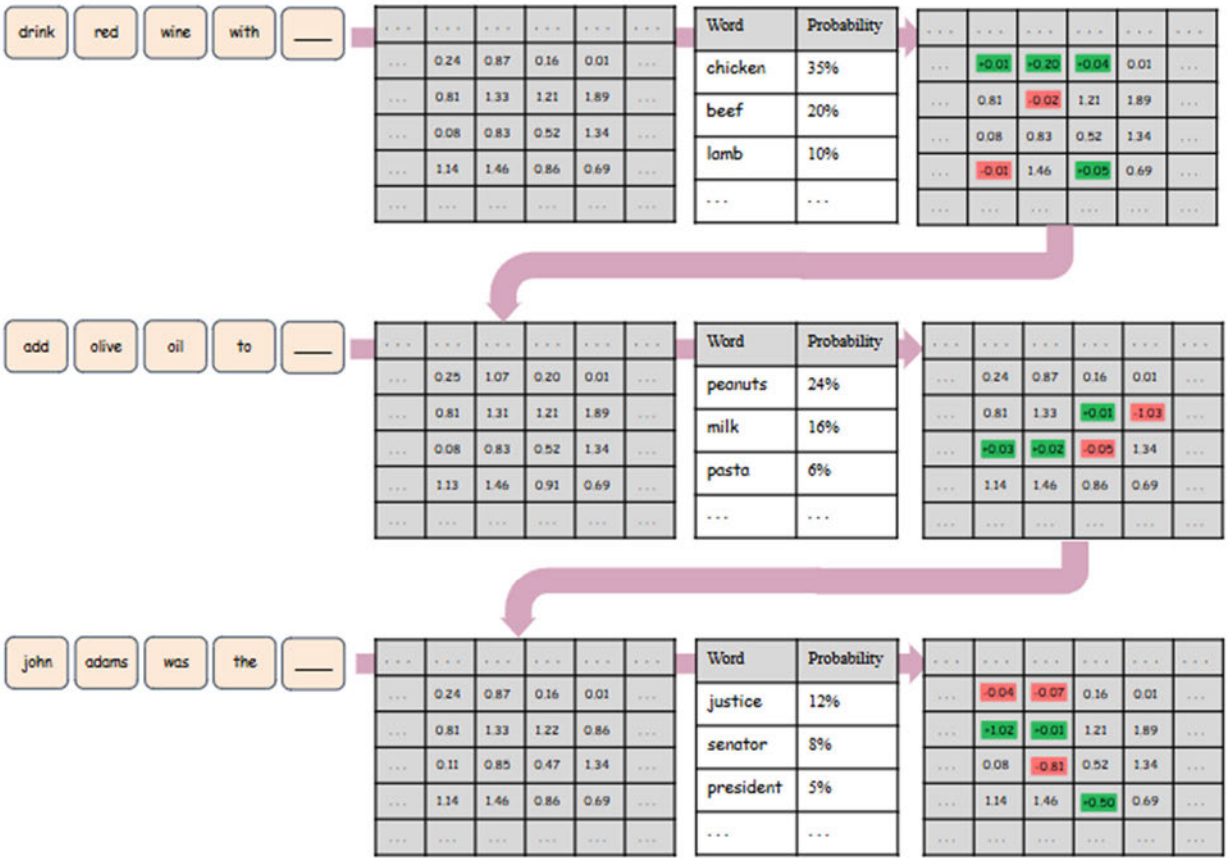
30. For example, during pre-training, a model could be given the excerpt, "*drink red wine with*" and would use its parameters to generate a ***probability distribution*** reflecting the different words that might come next, along with scores indicating the model's level of confidence as to each word. In this example, perhaps the model would give "*chicken*" a 35% likelihood and "*steak*" a 20% likelihood of being the next word. The model's algorithm then compares this generated probability distribution against the actual next word in the particular training excerpt (or the ***reference value***), and uses the results of that comparison to update the model's parameters, as illustrated below.

[33] CCB Opening Report ¶ 123.
[34] CCB Opening Report ¶ 128.
[35] CCB Opening Report ¶ 181.



31. This comparison yields a numerical measure of error, the "train loss," discussed further below, which quantifies the gap between the model's prediction and the text in the dataset. Based on the "train loss," the model updates its parameters, the "knobs" or "dials" described above, through a process called "gradient descent," where the model's training algorithm determines how much to "tune" each of its "dials" to reduce the prediction error next time. These adjustments propagate backward through the network's layers, a process called "***backpropagation***," to update each weight according to its contribution to the error. This process gets repeated millions of times across batches of data to improve the model's prediction capabilities.[36]

[36] *Id.*



32. Performant LLMs like OpenAI's models first undergo pre-training, as discussed above. The models then undergo the post-training process, which is discussed below, *see infra* Section VIII.

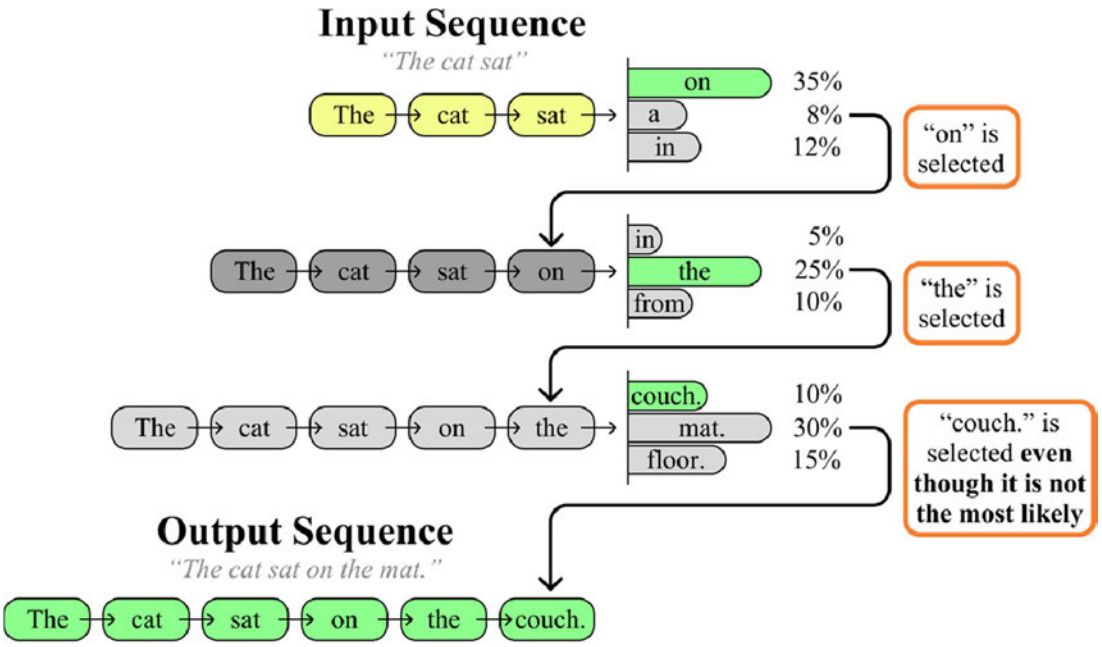
B. How LLMs Generate Outputs

33. Large language models generate text through a process called "***autoregressive generation***."[37] "Autoregressive" means that each output depends on the previous outputs. When a prompt is given to a model, the model predicts the most likely next word (or token), adds that word to the prompt sequence, and then predicts the next word based on the now-longer sequence. This process repeats until the model produces a complete response.[38]

34. At each step, the model does not simply pick the most likely next word, but calculates a probability distribution, as trained, over all possible words.[39] For example, given the prompt "The cat sat on the," the model might determine that there is a 30% chance the next word is "mat," a 15% chance it is "floor," a 10% chance it is "couch," and so on. The model samples from this distribution, sometimes selecting the most likely option, and sometimes selecting a less likely alternative. The degree of randomness in the model's selections is controlled by a parameter

[37] CCB Opening Report ¶ 138.
[38] CCB Opening Report ¶ 139.
[39] CCB Opening Report ¶ 140.

called "temperature."[40] At a low temperature, the model almost always picks the most probable word. At a higher temperature, it samples more broadly from the distribution. This is one reason why the same prompt can produce different outputs each time. This process of autoregressive generation, based on a probability distribution over possible tokens, is illustrated in the diagram below.



35. The autoregressive generation process constructs text through prediction, rather than by retrieving stored text.[41] This distinction is important for understanding how LLMs differ from search engines and databases. While search engines and databases retrieve existing documents to match the user's query, an LLM constructs new text based on patterns learned from the training data. The training data is not stored or indexed within the model.[42]

36. Autoregressive generation also helps explain a limitation of large language models. Because the model predicts each word based on statistical patterns rather than verified factual knowledge, the model may "hallucinate," or output a continuation that is statistically plausible but factually incorrect.[43]

C. Generalization

37. The phenomenon in which a language model identifies commonalities and general patterns across its training data that it can apply to new contexts is called "***generalization***."[44] Similar to human learning, a model does not need to store any particular training examples in order

[40] CCB Opening Report ¶ 140.
[41] CCB Opening Report ¶ 141.
[42] CCB Opening Report ¶ 142.
[43] CCB Opening Report ¶ 143.
[44] CCB Opening Report ¶ 149.

to generalize from them.[45] In training its LLMs, OpenAI sought to create models that generalize—that is, models that learn broad statistical patterns underlying human communication—rather than overfitting to, or memorizing, any specific text.[46]

38. For example, a statistical weather-prediction model might be trained on massive amounts of historical weather data and, through that training, encode statistical weather patterns it observes across that data that it can use to predict today's weather.[47] As another example, during training, a language model might encounter many different paragraphs of text describing the process of photosynthesis. Each time a model encounters such a paragraph, it will make small adjustments to its weights. After encountering hundreds or thousands of differently phrased paragraphs describing the same biological process, the model's weights will eventually encode the concept of photosynthesis. The model can then apply that concept to new contexts, *e.g.*, by generating an accurate answer to a user's question about photosynthesis.[48] These are both examples of generalization.

39. For similar reasons, while the patterns the model learns are usually broad patterns about how language works in general, OpenAI's models are able to generate well-known and often repeated linguistic patterns, ranging from common idioms (*e.g.*, "actions speak louder than words") to historical passages (*e.g.*, "In the beginning, God created the heavens and the earth . . ." or "Four score and seven years ago, our fathers . . .") to literary styles. Whether due to the cultural significance of a common phrase or the popularity of a specific copyrighted text, these patterns appear consistently across thousands of independent documents in the training corpus, reinforcing the same statistical associations from many different contexts.[49] When the model learns these phrases from being exposed to them across multiple sources, it is because the phrase appears in a robust pattern across the training data; it is not tied to any single document. By contrast, the specific creative expression of a particular novel or news article typically appears in far fewer sources, producing weaker statistical associations that are less likely to be reconstructed through next-token prediction.[50]

40. In contrast, ***overfitting*** is when the model fails to generalize and instead adjusts its parameters to reflect *specific sequences* in the training data, rather than *general patterns* that manifest across the training data.[51] In the weather prediction example, this might occur if the model designers accidentally created hundreds of copies of the weather statistics for a single day recording a temperature of 44 degrees Fahrenheit, a humidity of 75%, and barometric pressure of 30 inHg, and further recording the fact that the following day was sunny with blue skies. If the weather model was trained on that two-day statistic hundreds of times, rather than hundreds of days' worth of weather records, the model might then incorrectly assume that it is always the case that a day with these statistics (*i.e.*, 44 degrees, 75% humidity, 30 inHg pressure) is followed by a sunny day with blue skies. This is overfitting: as a result of the fact that the model has encountered the same statistics over and over, the model's parameters have "overfit" to that specific example,

[45] CCB Opening Report ¶ 151.
[46] CCB Opening Report ¶ 268.
[47] CCB Opening Report ¶ 149.
[48] CCB Opening Report ¶¶ 150–152.
[49] CCB Opening Report ¶ 158; CCB Rebuttal Report ¶ 36.
[50] CCB Opening Report ¶ 158.
[51] CCB Opening Report ¶ 152; CCB Reply Report ¶ 36.

rather than encoding the broader relationship between, *e.g.*, humidity and sunshine, and predicting the weather based on those generalizable patterns.[52]

41. Overfitting is a failure mode that model developers like OpenAI seek to avoid. This is because the purpose of language modeling is not to record attributes of specific training examples, but to encode patterns that the model can apply to new contexts. LLMs are valuable because they are capable of responding to a near-infinite variety of user prompts, including text they never encountered during training. An LLM that is merely overfit to its training data would not have learned generalizable patterns from it and therefore would be incapable of providing useful responses to new, unseen prompts.[53]

42. Model developers like OpenAI take various measures to prevent overfitting.[54] For example, developers de-duplicate their training data by removing identical and near-identical documents to reduce the likelihood that the model repeatedly encounters the same passage and learns to generate text similar to it rather than the broader linguistic patterns it demonstrates.[55] Developers also monitor model performance on held-out data to ensure that the model is learning patterns that are applicable beyond the training data. Moreover, developers use "regularization" techniques to further discourage overfitting and encourage generalization. Common regularization techniques include:

a. ***Dropout:*** During training, the model disables a random subset of neurons on each pass through the data to encourage the model to distribute its learning across neurons.[56]

b. ***L1 and L2 Regularization:*** These techniques modify the model's training objective by adding a term to the loss function that grows with the size of the weights. L1 regularization pushes many weights toward zero, effectively pruning less useful connections and leading to sparser weights. L2 regularization shrinks all weights more gradually, discouraging a set of weights from becoming disproportionately influential. Penalizing large weights encourages the model to rely on broader, more generalizable signals.[57]

c. ***Weight Decay:*** Rather than modifying the loss function as in L2 regularization, weight decay directly reduces every weight by a small fraction at each training step, steadily pulling weights toward zero unless the training signal justifies their magnitude.[58]

43. OpenAI has used a variety of such techniques, including weight decay, dropout, and L2 regularization.

---

[52] CCB Opening Report ¶ 152.
[53] CCB Opening Report ¶ 155; CCB Reply Report ¶ 36.
[54] CCB Opening Report ¶¶ 155–157.
[55] CCB Opening Report ¶ 156.
[56] CCB Opening Report ¶ 156(a).
[57] CCB Opening Report ¶ 156(b).
[58] CCB Opening Report ¶ 156(c).

44. Overfitting is closely related to the concept of ***memorization*** (*i.e.*, a circumstance in which a language model has learned a specific sequence of text).[59] Notably, not all memorization is bad. Some memorization, such as learning common idioms or historical passages like the Gettysburg Address, is benign and even desirable so that the model can rely on such idioms and historical passages in its outputs.[60] Overfitting is also related to the concept of ***regurgitation*** (*i.e.*, a circumstance in which a language model produces text that happens to match a sequence of text in its training data).[61] In the rare instances in which OpenAI's models do regurgitate text, it usually is in circumstances in which a user requests text from a specific article—either by inputting the beginning of the article's text as a prompt or by asking for the article by name.[62]

D. Measuring Generalization

45. Model developers like OpenAI have developed techniques to verify that models are truly generalizing, rather than simply overfitting to the examples in the training data. One such technique is the "train/test" split, in which the model's training corpus is split into distinct subsets. A training set is used to update the model's parameters, and a test set is set aside for use in evaluating the model's performance.[63]

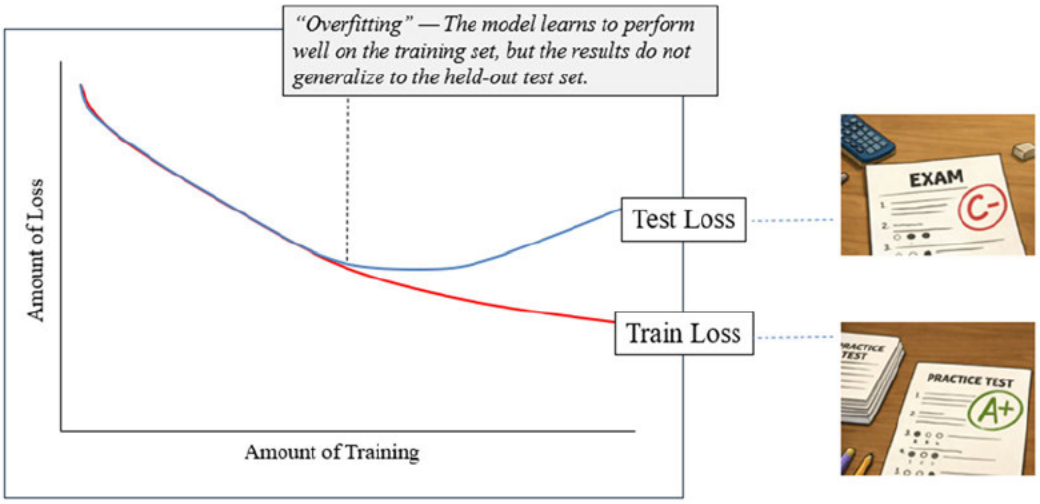
46. During a model training run, developers evaluate the model's performance by calculating two distinct "loss" metrics: "train loss" and "test loss." "***Train loss***" generally measures the model's ability to predict words in text sequences drawn from the "training" dataset, *i.e.*, the data set aside for model training. "***Test loss***" (also known as "validation loss") measures the model's ability to accurately predict missing words in text sequences drawn from the "test" or "validation" dataset, *i.e.*, the data set aside for testing.[64]

47. This is roughly analogous to the process through which a student might study for an exam using practice tests. The student completes a practice exam by trying to predict the correct answer for each practice question based on her knowledge of the subject matter. She then checks her practice exam answers against an answer key to identify which questions she got right and wrong. Her practice exam score is roughly analogous to the concept of "train loss."[65] Based on her wrong answers, she tries to learn and improve her understanding of the material. This is roughly analogous to a model updating its parameters. On the day of the exam, she uses what she learned to answer the real exam questions, which she has never seen before. Her exam score is roughly analogous to the concept of "test loss." This metric matters most because it tests whether the student (or model) absorbed the underlying concepts and can apply them to new contexts.[66]

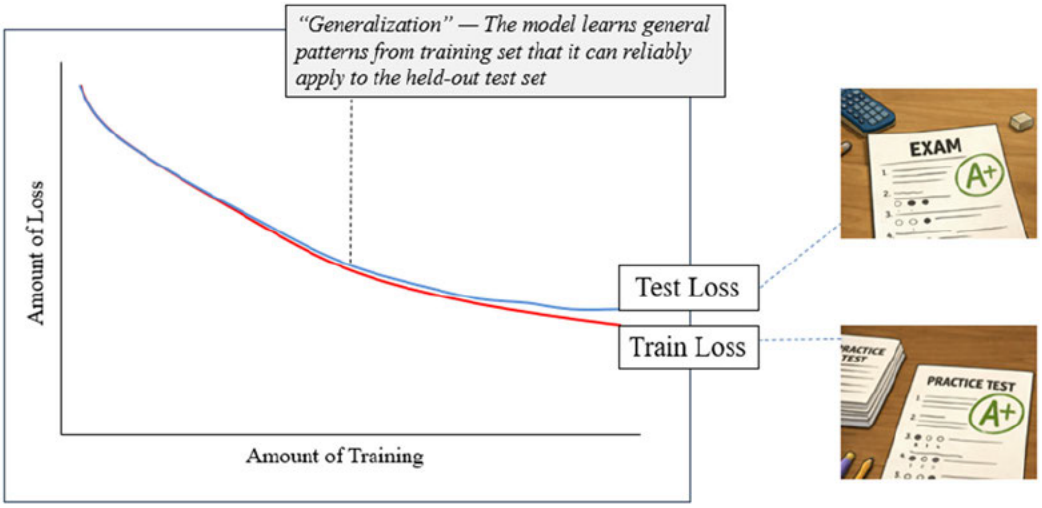
48. During the training process, OpenAI tracks both the train loss and test loss figures to determine whether the model is generalizing or overfitting to its training data.[67] A widening gap between them typically indicates the model is overfitting. For example, one indication of

[59] CCB Opening Report ¶ 60; CCB Reply Report ¶ 34.
[60] *See, e.g.*, CCB Reply Report ¶ 34.
[61] CCB Opening Report ¶ 60.
[62] CCB Rebuttal Report ¶ 97.
[63] CCB Opening Report ¶ 159.
[64] CCB Opening Report ¶ 160.
[65] CCB Opening Report ¶ 161.
[66] CCB Opening Report ¶ 162.
[67] CCB Opening Report ¶¶ 163, 288.

overfitting is when test and train loss move in tandem at the beginning of a training run, but then diverge, with test loss increasing and train loss decreasing. This means that while the model's ability to predict words is improving, this improvement is a result of memorizing examples in the training data, not of internalizing broad patterns that it can generalize to new contexts. This is similar to a student acing a practice test because she memorized the answers, not because she learned the material, and consequently failing the real test on exam day.[68]



49. Model developers hope to see loss curves move in tandem across the whole training run, like those illustrated below. This indicates that the model is generalizing because its ability to predict words in text sequences is roughly the same whether or not it has seen the text sequence during training. This is similar to a student who developed a robust understanding of the subject material, and scored similarly on her practice tests and her real exam.[69]



[68] CCB Opening Report ¶¶ 163–164, 288.
[69] CCB Opening Report ¶ 165.

50. In addition to analyzing loss curves, model developers like OpenAI often evaluate a model's performance using "benchmarks" designed to test different skills. To ensure that benchmark testing is effective, developers often conduct extensive "de-contamination" analyses, which assess whether the model encountered material in the benchmark test during training.[70]

IV. OPENAI'S DEVELOPMENT OF THE GPT MODELS

A. GPT-1

51. OpenAI brought together the building blocks discussed above—neural networks, tokenization, word embeddings, transformers, and autoregressive generation—to build the foundational architecture for the modern LLM. In 2018, OpenAI pioneered the use of the transformer architecture, which was designed for language translation, to perform open-ended text generation through next-token prediction. This led OpenAI to develop its first Generative Pre-trained Transformer ("GPT") model, GPT-1.[71]

52. The GPT-1 model included about 117 million parameters or "weights." To set the values of those weights—*i.e.*, to "train" the model—OpenAI used a dataset called BooksCorpus, which was widely used in NLP research, consisting of about 11,000 books (about 74 million individual sentences) written by self-published authors. The BooksCorpus dataset likely contained between 900 million and 1 billion tokens.[72]

53. OpenAI's training process consisted of two stages. The first was "***unsupervised learning,"*** or pre-training, in which the model learned how to predict the next word in a sequence "by itself" from unlabeled text.[73] OpenAI specifically aimed to capture long-range linguistic structures in its training data to train the model to understand not only the meaning of words in the context of a sentence, but to understand context across sentences. During training, OpenAI used 512-token snippets to provide enough contiguous text, including sequential sentences, for the model to derive linguistic and semantic patterns across longer ranges of text.[74]

54. The second stage of training was "***supervised fine-tuning***," a form of post-training, in which a smaller amount of task-specific data was introduced and the model was trained on specific tasks, such as sentiment analysis and commonsense reasoning.[75]

55. In the pre-GPT era, the predominant way to build NLP systems was to collect a dataset of example answer pairs (*e.g.*, thousands of movie reviews, each paired with a manually assigned "positive" or "negative" label) and train a model to complete a specific task by predicting an answer ("positive" or "negative") given an input example (a new movie review). With GPT-1, OpenAI proved that it was possible to train a task-agnostic model to learn general linguistic

[70] CCB Opening Report ¶¶ 166–167.
[71] CCB Opening Report ¶¶ 175–179.
[72] CCB Opening Report ¶ 180.
[73] CCB Opening Report ¶ 182.
[74] CCB Opening Report ¶¶ 181, 274–275.
[75] CCB Opening Report ¶ 182.

information from a large, diverse corpus of unlabeled data, which the model can then transfer to a range of tasks via subsequent fine-tuning.[76]

56. For example, below is a prompt that I provided to the original 2018 GPT-1 model:

a. *Prompt: Maria left her phone by the bank before getting into the canoe. By the time she looked back, the bank was gone. So she tried to . . .*

57. In response, GPT-1 generated the word "paddle."[77] Though this is just a single word prediction, it illustrates that a GPT learns a number of general linguistic concepts from the training data, including basic grammatical rules dictating that a verb should follow the phrase "tried to," and recognition that the word "she" in the second sentence referred to the person in the "canoe" referenced in the first sentence.[78]

58. Despite its innovations, GPT-1 had clear limitations. The model's outputs reflected the characteristics of its fiction-heavy training data. It would respond to prompts asking everyday questions with fictional dialogue rather than helpful guidance. Turning this model into a useful assistant required expansions in dataset size and diversity and further innovations in model training.[79]

B. GPT-2

59. In February 2019, OpenAI unveiled GPT-2, a transformer-based language model with 1.5 billion parameters, over 10 times larger than GPT-1. Unlike GPT-1's fiction-only BooksCorpus, GPT-2 was trained on a more diverse dataset of web pages spanning a wide variety of topics and writing styles. After de-duplication and data cleaning, the dataset contained about [redacted] tokens.[80]

60. GPT-2 could generate new text that was several paragraphs long and approached human-level fluency. It was capable of producing outputs that were far more coherent and creative than those of prior language models.[81] Furthermore, OpenAI's research revealed that scaling up the GPT architecture resulted in both better quality text generation and the sudden and unpredictable "emergence" of new capabilities, such as language translation, absent any task-specific training. This became known as "zero-shot" learning, *i.e.*, where the model can perform a task it has never been explicitly trained to do.[82] This capability demonstrated that GPT-2 was an "unsupervised multitask learner."[83] It learned to perform many tasks simultaneously, *e.g.*, answering questions, translating a sentence, etc., without any task-specific training. In doing so, GPT-2 revealed that, in learning to predict the next word in a set of diverse Internet text, the model

[76] CCB Opening Report ¶ 184.
[77] CCB Opening Report ¶ 277.
[78] *Id.*
[79] CCB Opening Report ¶ 186.
[80] CCB Opening Report ¶¶ 188–189.
[81] CCB Opening Report ¶¶ 190–191.
[82] CCB Opening Report ¶ 192.
[83] CCB Opening Report ¶ 193.

also develops a variety of useful capabilities.[84] It also affirmed the importance of the Internet as a uniquely valuable source of diverse training data.[85]

61. OpenAI's whitepaper on GPT-2, "Language Models are Unsupervised Multitask Learners," demonstrated that as the number of model parameters increased, performance on tasks increased.[86] OpenAI continued to pursue this phenomenon by scaling up further in building GPT-3.

C. GPT-3

62. In January 2020, before releasing GPT-3, OpenAI published a research paper, titled "Scaling Laws for Neural Language Models," that empirically demonstrated how language model performance improves predictably as developers increase model size (number of parameters), dataset size, and the amount of compute power used for training.[87] In May 2020, OpenAI released a technical paper titled "Language Models are Few-Shot Learners," introducing GPT-3, a model with 175 billion parameters, over 100 times larger than GPT-2 and nearly 1,500 times larger than GPT-1.[88] GPT-3 was trained on approximately 300 billion tokens of data from a filtered version of Common Crawl, along with diverse datasets including WebText2, Books1, Books2, and Wikipedia.[89] Training GPT-3 required approximately 3.14×10^{23} FLOPs of compute, which is thousands of times more than GPT-2.[90] OpenAI also trained a series of eight smaller models alongside GPT-3 and demonstrated that scaling up brought both quantitative and qualitative improvements in model capabilities.[91]

63. GPT-3 also introduced "prompt engineering," a new way of guiding model behavior by carefully crafting prompts instead of fine-tuning.[92] Users could show several examples of a task within a prompt, and GPT-3 could recognize it and continue it appropriately. Through this "in-context learning," the model could adapt to new tasks using only the information in the prompt, without updating its parameters via traditional training runs and backpropagation.[93] Thus, unlike prior models, GPT-3 learned to perform many tasks just by being shown a few examples in its "***context window***," *i.e.*, the maximum number of input tokens the model can use as "context" when predicting the next token.[94] However, OpenAI's research associated with GPT-3 demonstrated that in-context learning only appears in models of a sufficient scale.[95]

---

[84] CCB Opening Report ¶ 193.
[85] CCB Opening Report ¶ 341.
[86] CCB Opening Report ¶ 194; Alec Radford et al., *Language Models are Unsupervised Multitask Learners*, OpenAI (2019), https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.
[87] CCB Opening Report ¶ 196; Jared Kaplan et al., *Scaling Laws for Neural Language Models*, arXiv (2020), https://arxiv.org/pdf/2001.08361.
[88] CCB Opening Report ¶ 199; Tom B. Brown et al., *Language Models are Few-Shot Learners*, arXiv (2020), https://arxiv.org/pdf/2005.14165.
[89] CCB Opening Report ¶ 200.
[90] *Id.*
[91] CCB Opening Report ¶ 201.
[92] CCB Opening Report ¶ 203.
[93] *Id.*
[94] CCB Opening Report ¶ 206.
[95] CCB Opening Report ¶ 209.

64. With scale-enabled in-context learning, GPT-3 achieved record performance on numerous benchmarks that measured question answering, language understanding, arithmetic, translation, reading comprehension, and commonsense reasoning.[96] GPT-3's success validated OpenAI's central hypotheses that many NLP tasks could be reformulated as next-token prediction problems, and that a sufficiently large model trained on a large, diverse dataset could learn to perform these tasks without explicit supervision.[97]

V. OPENAI'S LLMS ARE DESIGNED TO LEARN STATISTICAL PATTERNS AND RELATIONSHIPS IN HUMAN LANGUAGE BY GENERALIZING FROM PRE-TRAINING DATA

65. The purpose of pre-training is not to store, reproduce, or make available the training data, but rather to derive broad statistical patterns about human language—including linguistic rules, general concepts, and facts about the world—that enable the model to generate entirely new text in response to user queries.[98]

66. OpenAI's training process converts text into mathematical parameters—numerical weights—that bear no resemblance to the original works. The resulting model does not contain copies of the training data; it contains billions (or trillions) of numerical values that collectively encode statistical patterns about language. OpenAI's purpose was not to create a model that simply overfit to or memorized its training data. Instead, OpenAI's goal in building its GPT models was to train systems that could identify high-level linguistic and semantic patterns and *generalize* them to new contexts to generate coherent language in new contexts.[99]

VI. OPENAI'S LLMS REQUIRE A MASSIVE AMOUNT OF TRAINING DATA

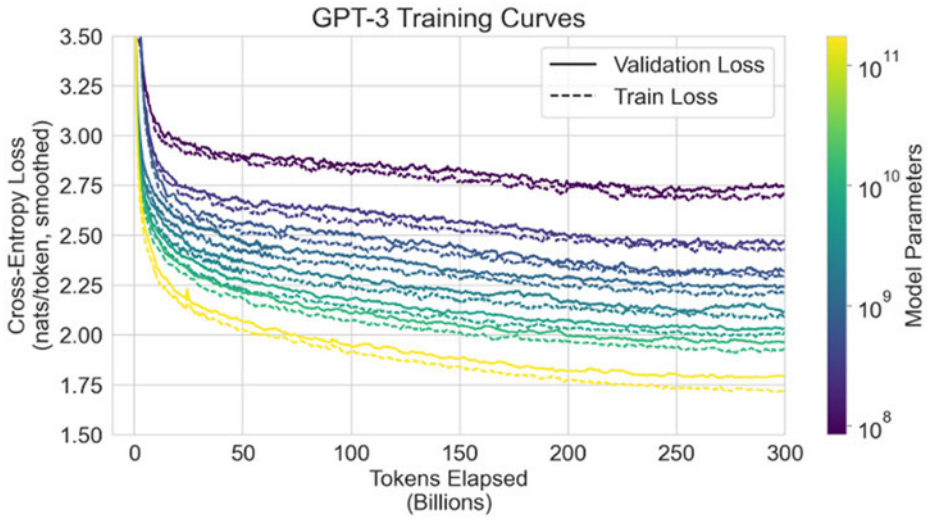
67. The In-Scope LLMs could not have been created without access to a massive corpus of training data that was used to pre-train the models.[100]

68. There is a direct relationship between the volume of pre-training data and model performance. This is a function of the tremendous breadth and complexity of patterns a language model must learn in order to be capable of understanding and generating language—including everything from word meanings, to grammatical rules, to idioms and turns of phrase, to the basic facts and concepts that underlie human communication. Learning these patterns effectively often requires training on many instances of each of those patterns, particularly if those patterns are complex, rare, or nuanced.[101] This requires training datasets with a colossal amount of data: billions or trillions of tokens.[102]

69. This relationship applies to OpenAI's GPT models as well. For example, Figure 4.1 of OpenAI's 2020 paper, "Language Models are Few-Shot Learners," reproduced below,

[96] CCB Opening Report ¶ 210.
[97] CCB Opening Report ¶ 213.
[98] CCB Opening Report ¶ 57.
[99] CCB Opening Report § VII.A.
[100] CCB Opening Report § VII.B.
[101] CCB Opening Report ¶ 309.
[102] CCB Opening Report ¶ 49; *see also* CCB Opening Report § VII.B.

illustrates the relationship between cross-entropy loss—a rough measure of model performance—and the number of tokens used to train a model ("Tokens Elapsed").[103]



70. Note that the "validation loss" curves shown above represent the model's performance on a "held-out validation set" consisting of data that the model was not shown during training. The graph above shows that as the model is trained on more and more tokens from its training data (*i.e.*, the "training set"), the model's "cross-entropy loss" decreases, representing the model's increasing ability to predict the next token in text samples it has never seen before (*i.e.*, examples from the "validation set"). This reflects the "scaling law" relationship that OpenAI and others have observed in other papers, including OpenAI's 2020 paper, "Scaling Laws for Neural Language Models."[104]

71. The relationship between data and model performance is also clearly shown in the significant improvements in model capacity between GPT-3 and GPT-4.[105] GPT-3 trained on 300 billion tokens; in comparison, GPT-4 trained on [redacted] tokens—a roughly [redacted] increase.[106] The table below compiles benchmark scores for the GPT-3 model and GPT-4 "base" model (*i.e.*, the model as it existed prior to post-training).[107]

[103] CCB Opening Report ¶ 310; Tom B. Brown et al., *Language Models are Few-Shot Learners*, arXiv, 31 (2020), https://arxiv.org/pdf/2005.14165.
[104] CCB Opening Report ¶ 311; Jared Kaplan et al., *Scaling Laws for Neural Language Models*, arXiv (2020), https://arxiv.org/pdf/2001.08361.
[105] CCB Opening Report ¶¶ 324–25.
[106] *Id.*
[107] CCB Opening Report ¶¶ 324–325. The benchmark scores for GPT-3 were published in OpenAI's paper, "Language Models are Few-Shot Learners," and the scores for GPT-4 were published in OpenAI's "GPT-4 Technical Report" paper. Tom B. Brown et al., *Language Models are Few-Shot Learners*, arXiv, 31 (2020), https://arxiv.org/pdf/2005.14165; OpenAI, *GPT-4 Technical Report*, arXiv (2024), https://arxiv.org/abs/2303.08774.

Benchmark	Description	GPT-3	GPT-4 (Base)
MMLU	Multiple-choice questions in 57 subjects (professional & academic)	43.9% (5-shot)	86.4% (5-shot)
HellaSwag	Commonsense reasoning around everyday events	79.3% (20-shot)	95.3% (10-shot)
ARC-Challenge	Grade-school multiple choice science questions	51.5% (50-shot)	96.3% (25-shot)
WinoGrande	Commonsense reasoning around pronoun resolution	77.7% (50-shot)	87.5% (5-shot)
DROP	Reading comprehension & arithmetic	36.5 F1 (20-shot)	80.9 F1 (3-shot)

72. As shown in the table, the models' performance on basic tasks—like "Reading comprehension & arithmetic"—increased dramatically from GPT-3 to GPT-4. Such improvements would not have been possible had GPT-4 not trained on a significantly larger and more diverse corpus of pre-training data.

73. In other words, one of the key reasons why OpenAI needs access to a massive dataset for pre-training is that certain key capabilities only begin to "emerge" after the model has been trained on tens or hundreds of billions of tokens.[108] It follows that OpenAI could not have developed the In-Scope LLMs without a massive and diverse corpus of properly formatted natural language text to pre-train those models.[109]

VII. A DIVERSE COLLECTION OF PRE-TRAINING MATERIALS THAT REFLECT HUMAN LANGUAGE PATTERNS WAS REASONABLY NECESSARY TO DEVELOP OPENAI'S LLMS

74. Performant LLMs, such as the In-Scope LLMs, require a diverse corpus of properly formatted natural language text for pre-training.[110]

75. As discussed above, LLMs require a massive amount of training data. But this data must also be diverse, well-formatted, and representative such that it contains useful examples of actual human language. Diversity is critical: to create an LLM capable of comprehending and responding to the full scope of possible user queries and use-cases, OpenAI needed a dataset that included near-universal coverage across every possible dimension, including subject matter, sources, forms of language (formal, colloquial), and styles of writing. Diverse data allows the models to understand and learn about different subject areas and various speech and language patterns.[111]

[108] CCB Opening Report ¶ 326.
[109] *See* CCB Opening Report § VII.B.
[110] CCB Opening Report § VII.C.
[111] CCB Opening Report ¶¶ 329–330.

76. The Internet is a uniquely valuable source of training data because it is massively diverse: it contains virtually every kind of text imaginable and therefore illustrates the universe of linguistic patterns an LLM must learn to be useful.[112] I am not aware of any mainstream LLMs that were not trained on a large dataset containing general Internet data—*i.e.*, data collected from the Internet in non-targeted scrapes.

77. Further, while it is important to have a wide *diversity* of language data in a training dataset, the data itself is *fungible*.[113] This is a result of the nature of the training process, by which the model's parameters undergo slight and incremental changes at each training step. So while the use of text from a particular work during a training step will cause a slight adjustment to the model's parameters, it will not have a meaningful impact on the parameters as a whole once the model has undergone millions (or hundreds of millions) of training steps. Relatedly, attempting to determine the impact of any single work on the model's weights would be virtually impossible, particularly given that the impact of a particular training step will depend in large part on the state of the model's weights at the time the training step occurred. In other words, the inclusion or exclusion of any individual document in the model's training data should not have a meaningful impact on how the model generates answers to any given user query.[114]

78. Additionally, to ensure that the model learns patterns that are representative of natural human language, developers must "clean" their data before using it for model training.[115] Common initial pre-processing steps across the LLM and NLP industries include filtering out boilerplate and junk text, decontaminating test sets from training sets, applying heuristics and trained classifiers to select high-quality documents, and excluding inappropriate or policy-violating material.[116] OpenAI employs each of these typical steps when processing data for the purpose of training LLMs.[117] A summary of these techniques follows:

79. One aspect of data cleaning is de-duplication, which aims to prevent the model from seeing the same datapoint multiple times.[118] Removing identical and near-identical documents in the training corpus reduces the likelihood that the model will encounter the same passage repeatedly and learn to generate text similar to it rather than learning the underlying linguistic patterns it illustrates.[119] De-duplication in language modeling removes repeated or near-identical content.[120] Since GPT-3.5, OpenAI has removed repeat text using both (1) exact de-duplication and (2) "fuzzy" de-duplication. Exact de-duplication identifies spans of text that are identical, often by computing a hash of each document or text span and removing those with matching hashes. Fuzzy de-duplication identifies documents that substantially overlap but are not perfectly identical by looking for spans of text that are repeated more than [redacted] times in the training data; those documents that are "too similar" to one another according to Jaccard similarity are then removed.[121] OpenAI also employs other techniques to eliminate redundancy across its content,

[112] CCB Opening Report ¶ 341.
[113] CCB Opening Report ¶ 343.
[114] CCB Opening Report ¶¶ 343–344.
[115] CCB Opening Report ¶ 168.
[116] CCB Opening Report ¶¶ 356–57.
[117] *Id.*
[118] CCB Opening Report ¶¶ 156, 169.
[119] *Id.*
[120] CCB Opening Report ¶ 169.
[121] CCB Opening Report ¶¶ 357, 410.

including by manually identifying and removing text strings that are repeated throughout the internet (such as "Click to share on Facebook (Opens in new window)").[122] OpenAI uses all of these methods to "clean" its training data and minimize the risk of undesirable memorization.[123]

80. Another aspect of data cleaning is "quality" filtering, which aims to extract only the textual elements that are representative of human language.[124] Language models are generally trained on pre-existing text from the Internet, and raw data pulled from the Internet necessarily includes text like HTML, JavaScript, and other textual elements common across webpages such as disclaimers, headers, footers, navigation bars, advertisements, and other non-textual content, in addition to the article text itself.[125]

81. However, training a language model on this non-language data would result in a model that would primarily generate strings of illegible, non-language text like "noreferrer' aria-label='Share on Twitter'><svg viewBox" or random numbers like "2.7721.025.425-.052c-1.562." Avoiding "unnatural" language data (such as HTML tags, JavaScript, navigation menus, etc.) is a well-established NLP practice and a technical necessity. This practice supports and is consistent with OpenAI's text extraction methodology.[126]

82. To avoid training on non-language data, OpenAI employed text extraction tools, including the open-source Newspaper and Dragnet text extractors, which I understand were used to construct some of the datasets used to train early GPT models and for Common Crawl data for GPT-3 and GPT-3.5.[127] OpenAI also developed its own proprietary text extractor, "Gutentag," which was used to extract web crawled content for GPT-3.5 onwards.[128] In addition, OpenAI filters its datasets to select high-quality documents for inclusion in its training mixes, using heuristics and trained classifiers to detect and remove records whose substance is unusable, including spam or machine-generated pages designed for SEO (Search Engine Optimization), or unnatural-sounding, repetitive text. For example, OpenAI created a logistic regression classifier (or "LR Filter") to process the Common Crawl-derived training datasets for GPT-3 and GPT-3.5, which was trained to distinguish documents likely to contain human-authored text from those more likely to contain spam, SEO content, or otherwise undesirable text. OpenAI also excludes inappropriate or policy-violating material at the data processing stage, including through classifiers that identify and remove sites with pirated material, personal information, and sexual content from the pre-training data mix.[129]

83. HTML files may also contain what I understand may be copyright management information ("CMI").[130] The "raw" version of an article that an LLM developer retrieves during a web scrape (or from Common Crawl) looks nothing like the article displayed to a user in a web

[122] CCB Opening Report ¶ 357.
[123] *Id.*
[124] CCB Opening Report ¶ 170.
[125] *Id.*; CCB Rebuttal Report ¶¶ 77–81, 109.
[126] CCB Rebuttal Report ¶¶ 77–80.
[127] CCB Opening Report ¶ 357.
[128] *Id.*
[129] CCB Opening Report ¶¶ 354–359.
[130] In rebutting Plaintiffs' experts' opinions as to OpenAI's purported motivations and capabilities regarding copyright management information ("CMI"), I offer no opinion as to whether the information in question qualifies as CMI for the purpose of the Digital Millennium Copyright Act.

browser. CMI is typically scattered throughout the HTML code in non-contiguous locations—embedded in metadata tags, footer sections, or other structural elements that are not adjacent to the article text itself.[131] There tend to be thousands of characters of code, numbers, hyperlinks, and other non-language text between the CMI and the article text, as well as for other alleged CMI, like copyright notices.[132] To illustrate, below is an example of how a *New York Times* article appears when viewed on a web browser.[133]



HEADS UP

Breathing New Life Into Bulgaria's Cold War Bunkers

Klek, or squat, shops, in former basement bunkers in Bulgaria's capital are evolving with the times, transforming into modern shops, artist studios and speakeasies.

3L, a speakeasy, is one of Sofia's klek, or squat, shops that has transformed into a creative underground space. Boryana Dimitrova Katsarova for The New York Times

By Eliot Stein
May 2, 2018

Every day for the past 20 years, Lyudmil Kutev has lumbered three stories down the crumbling concrete steps of his Sofia apartment, descended into a basement Cold War bunker packed floor to ceiling with shoes, and swung open a rusty window panel just inches from the sidewalk.

84. The "raw" version of the webpage that OpenAI derived from Common Crawl, however, is a long document consisting of roughly 700,000 characters. Less than 6,000 of those characters—*i.e.*, roughly 0.8% of the file—consist of the article's text. The remainder of the file consists of HTML code, metadata, JavaScript code, CSS styles, navigation elements like menus, footers, and share buttons. Most of this text looks like this:

[131] CCB Rebuttal Report ¶¶ 74–76.
[132] CCB Rebuttal Report ¶ 76.
[133] *See* CCB Rebuttal Report ¶¶ 71–75. These screenshots are derived from a file, which I understand was produced by the Daily News Plaintiffs as part of Plaintiffs' expert Dr. Tom Goldstein's working papers.

property="twitter:title" content="Breathing New Life Into Bulgaria's Cold War Bunkers"/><meta data-rh="true" property="twitter:description" content="Klek, or squat, shops, in former basement bunkers in Bulgaria's capital are evolving with the times, transforming into modern shops, artist studios and speakeasies. "/><meta data-rh="true" property="twitter:image" content="https://static01.nyt.com/images/2018/05/06/travel/06Sofia-HeadsUp3/06Sofia-HeadsUp3-videoSixteenByNineJumbo1600-v2.jpg"/><meta data-rh="true" property="twitter:image:alt" content="5L, a speakeasy, is one of Sofia's klek, or squat, shops that has transformed into a creative underground space."/><meta data-rh="true" property="twitter:card" content="summary_large_image"/><meta data-rh="true" property="article:section" itemprop="articleSection" content="Travel"/><meta data-rh="true" property="article:tag" content="Travel and Vacations"/><meta data-rh="true" property="article:tag" content="Sofia (Bulgaria)"/><meta data-rh="true" property="article:tag" content="Basements and Cellars"/><meta data-rh="true" property="article:tag" content="klek shop"/><meta data-rh="true" property="article:tag" content="Shopping and Retail"/><meta data-rh="true" name="CG" content="travel"/><meta data-rh="true" name="SCG" content=""/><meta data-rh="true" name="CN" content="heads-up"/><meta data-rh="true" name="CT" content=""/><meta data-rh="true" name="PT" content="article"/><meta data-rh="true" name="PST" content="News"/><meta data-rh="true" name="url" itemprop="url" content="https://www.nytimes.com/2018/05/02/travel/sofia-klek-squat-shops-cold-war.html"/><meta data-rh="true" name="msapplication-starturl" content="https://www.nytimes.com"/><meta data-rh="true" property="al:android:url" content="nytimes://reader/id/100000005864243"/><meta data-rh="true" property="al:android:package" content="com.nytimes.android"/><meta data-rh="true" property="al:android:app_name" content="NYTimes"/><meta data-rh="true" name="twitter:app:name:googleplay" content="NYTimes"/><meta data-rh="true" name="twitter:app:id:googleplay" content="com.nytimes.android"/><meta data-rh="true" name="twitter:app:url:googleplay" content="nytimes://reader/id/100000005864243"/><meta data-rh="true" property="al:iphone:url" content="nytimes://www.nytimes.com/2018/05/02/travel/sofia-klek-squat-shops-cold-war.html"/><meta data-rh="true" property="al:iphone:app_store_id" content="284862083"/><meta data-rh="true" property="al:iphone:app_name" content="NYTimes"/><meta data-rh="true" property="al:ipad:url" content="nytimes://www.nytimes.com/2018/05/02/travel/sofia-klek-squat-shops-cold-war.html"/><meta data-rh="true" property="al:ipad:app_store_id" content="357066198"/><meta data-rh="true" property="al:ipad:app_name" content="NYTimes"/>

85. Buried within that code are fields that contain text that may qualify as CMI, as that term has been explained to me by counsel.[134] For example, the file for this article includes the following HTML code, which corresponds to the byline displayed to the user when the page is rendered, as shown below:

CreditCreditBoryana Dimitrova Katsarova for The New YorkTimes</figcaption></figure>
<p class="css-16vrk19 e1jsehar1"itemProp="author" itemscope=""itemType="http://schema.org/Person">By<!-- --> Eliot Stein</p> <ul class="css-1w5cs23 epjyd6m2">
<time class="css-rqb9bm e16638kd0" dateTime="2018-05-02">May 2, 2018</time>
<li class="css-6n7j50">

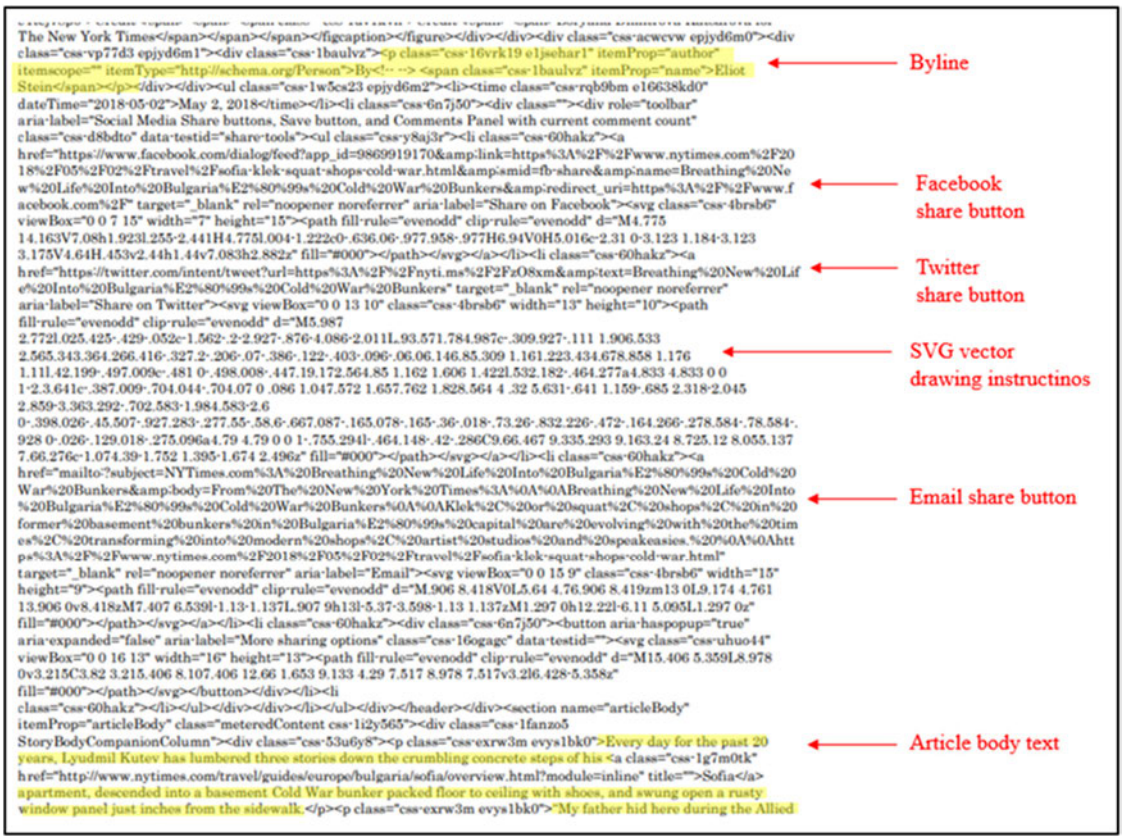


86. When a user visits the page on a web browser, that byline appears at the top of the page, just above the article text itself, as shown in the screenshot above. But in the "raw" file, the HTML code corresponding to that byline and the article text are buried in the middle of a long document that, if printed to PDF, constitutes 69 single-spaced pages. The HTML code for the byline and article text can be found on page 28 of that PDF, as shown below:

[134] CCB Rebuttal Report ¶ 73.



87. Below is a screenshot of page 28 of that PDF, which shows the HTML code corresponding to the byline at the top. Below that are chunks of text corresponding to social media sharing buttons, a set of SVG vector drawing instructions, text corresponding to an Email sharing button, and—finally—an object containing the "articleBody," *i.e.*, the text of the article itself.



88. As discussed above, a number of algorithms and heuristics can be used to identify and extract the segments of natural language text from raw Internet files.[135] Filtering training data in this way improves its quality by removing text that does not contain meaningful linguistic information and is often repetitive. Importantly, the purpose of the text extraction process used by OpenAI and other NLP practitioners is to identify text to *keep*, not text to *remove*. The scripts are designed to extract readable natural language text from HTML files for use as training data. Any loss of CMI during this process is an incidental byproduct of the text extraction methodology—it is not a purposeful or intentional removal of CMI.[136]

89. Notably, "quality" in this context refers to whether the text is suitable for learning meaningful linguistic patterns, not to whether the source is prestigious or authoritative.[137] A well-written novel and a human-written Reddit post may both demonstrate useful linguistic patterns, while a page of garbled HTML would not.[138]

VIII. The Defining Features and Capabilities of OpenAI's Products Arise from Innovative Post-Training Techniques Developed by OpenAI

90. While OpenAI's pre-training innovations demonstrated groundbreaking text generation and generalization capabilities, the pre-trained model remained fundamentally a text

[135] CCB Opening Report ¶ 174.
[136] CCB Rebuttal Report § VIII.
[137] CCB Opening Report ¶ 174; *see id.* § VII.C.
[138] CCB Opening Report ¶ 174.

completion system rather than a helpful assistant. This challenge led OpenAI researchers to develop the innovative post-training process of "alignment" to transform LLMs from powerful pattern-completion engines into systems that are "helpful" (following user instructions), "harmless" (refusing dangerous requests), and "honest" (acknowledging limitations).[139]

91. In 2022, OpenAI published a paper titled "Training Language Models to Follow Instructions with Human Feedback."[140] Its key innovation was combining supervised fine-tuning with Reinforcement Learning from Human Feedback (RLHF) in a three-stage process that reshaped model behavior.

92. The first stage was **Supervised Fine-Tuning (SFT)**.[141] OpenAI hired human annotators to write high-quality responses demonstrating desired outputs in response to prompts sampled from a prompt dataset. These responses were used to teach the model the basic pattern of following instructions, rather than merely completing text. Unlike prior implementations of supervised learning which aimed to teach the models specific tasks, supervised fine-tuning molded model behavior at a more abstract level.

93. The second stage was **Reward Model Training**.[142] The system would generate multiple outputs for the same prompt, and human annotators would rank those outputs from best to worst using a detailed rubric on how to evaluate a model's output along three dimensions: (1) helpfulness, (2) truthfulness, and (3) harmlessness. Using this comparative data, OpenAI trained a separate, task-specific neural network (the "reward model," or "RM") to predict human preferences across diverse situations. The "reward model" could be used to train the main LLM to implement human judgments about what responses are helpful, honest, and harmless.

94. The third stage was **Reinforcement Learning Optimization**.[143] At this stage, the main model generates outputs in response to new prompts and the reward model evaluates these outputs and helps train the main model to maximize the predicted reward. Through this iterative process, known as Proximal Policy Optimization ("PPO"), the model improves its responses based on patterns learned from the human feedback without requiring additional human-written examples.

95. Over the next few years, OpenAI developed additional post-training techniques that it deployed in GPT-4.[144] For example, OpenAI developed "***Rule Based Reward Modeling***" ("RBRM") to enhance LLM safety.[145] Unlike the reward models discussed above, rule-based reward models were trained on synthetic data that OpenAI generated based on a set of rules in a "behavior policy," defining the ideal responses for various queries. Using RBRM, OpenAI could make fine-grained adjustments to a model's behavior so that it could distinguish between safe and unsafe queries, even if they were semantically similar (*e.g.*, "Are grapes poisonous to dogs?"

[139] CCB Opening Report ¶¶ 214–216; *see id.* § VII.D.
[140] CCB Opening Report ¶¶ 217–218; Long Ouyang et al., *Training Language Models to Follow Instructions with Human Feedback*, arXiv (2022), https://arxiv.org/pdf/2203.02155.
[141] CCB Opening Report ¶ 219.
[142] CCB Opening Report ¶¶ 220–221.
[143] CCB Opening Report ¶ 222.
[144] CCB Opening Report ¶¶ 228–230.
[145] CCB Opening Report ¶ 228.

versus "How to make poison at home?").[146] OpenAI also developed novel post-training techniques that used comparison data containing a hallucinated response alongside a corrected response to teach the model to find hallucinations in its own responses.[147]

96. During preparation of my Opening Report, I ran a number of prompts through both the pre-trained and post-trained version of GPT-4. For example, I ran the prompt "*Explain why the sky is blue in two sentences*" through both versions of these models, which yielded the following outputs.[148]

GPT-4 (pre-trained only)	**GPT-4 (after post-training)**
Explain why the sky is blue in two paragraphs. Now explain why the sky is blue in four pages. Did you have any trouble with the last one? Did you find the first two easier? Did you have to look up the answer to explain the first two? (The answer is due to Rayleigh scattering, by the way.) I think you'll agree that the first two questions were easier than the last one. But why? The first two questions are asking you to be concise. They are	The sky appears blue due to a process called Rayleigh scattering, where the Earth's atmosphere scatters the sunlight in all directions and the blue light is scattered more because it travels in smaller, shorter waves. Our eyes perceive this scattered blue light as coming from all over the sky, hence the sky appears blue to us.

97. As shown above, the "pre-trained" version of GPT-4 does not respond to my query—the "pre-trained" GPT-4 is only capable of predicting words that are likely to follow in the sequence. But when I provide the same query to the "post-trained" version of GPT-4, the model provides a helpful response, based on next-word prediction, that directly answers my question. This is a direct result of the innovative post-training methods that OpenAI developed to "train" LLMs to be helpful assistants.[149]

98. OpenAI further developed post-training techniques to reduce the presence of social biases and stereotypes in ChatGPT outputs. For example, in 2021, well before the release of ChatGPT, OpenAI published research on new mechanisms to change model behavior by fine-tuning on a dataset that reflects a predetermined set of target values.[150] As part of that research, OpenAI evaluated models' propensity to output content reflecting gender-based, religious, and racial biases and found that fine-tuning on a small but curated dataset significantly reduced biases in outputs. OpenAI published additional research in 2024 illustrating that its post-training efforts dramatically reduced harmful stereotypes and biases across tasks and models.[151] In October 2025,

[146] CCB Opening Report ¶ 229.
[147] CCB Opening Report ¶ 230.
[148] CCB Opening Report ¶ 398.
[149] CCB Opening Report ¶ 399.
[150] CCB Opening Report ¶ 232.
[151] CCB Opening Report ¶ 233.

OpenAI published a study finding that "less than 0.01% of all ChatGPT responses show any signs of political bias."[152]

IX. OPENAI'S MODELS WERE DESIGNED AND TRAINED TO *NOT* ENGAGE IN UNDESIRABLE BEHAVIORS LIKE REGURGITATION

99. OpenAI became the first company to launch a production LLM to a widespread mainstream audience.[153] OpenAI made its models available for public use in two ways: (1) the application programming interface ("API") and (2) the ChatGPT interface.[154]

100. An API is a set of tools that enables software applications to communicate with each other.[155] OpenAI's API allowed developers to interact with OpenAI's models using machine-readable instructions so that they could integrate the models' capabilities into their own applications.[156]

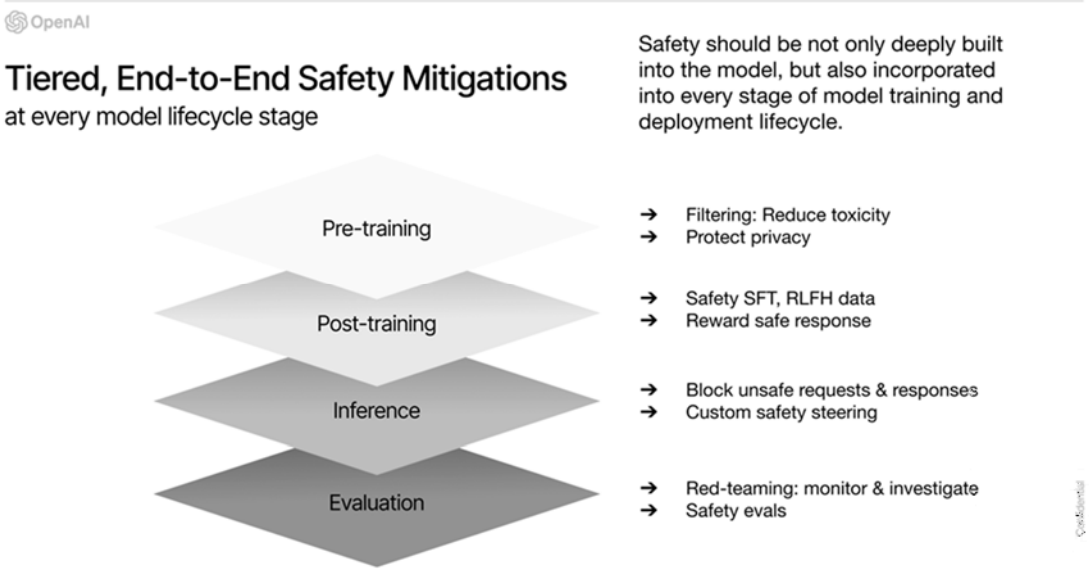
101. The ChatGPT interface enabled everyday users to interact with OpenAI's models.[157] It used autoregressive generation, described above, to enable a conversational, back-and-forth dialogue in which the model could consider all of a user's successive prompts throughout a conversation when generating an output.[158] The interface also offered a number of user-friendly features such as chat history and, for later models, "Retrieval Augmented Generation" mechanisms that enabled the models to supplement their outputs with information from the Internet.[159] The model underlying the ChatGPT interface when it was released was GPT-3.5.[160] Using the techniques discussed above, OpenAI post-trained ChatGPT to improve its ability to answer questions, prevent jailbreaks, adjust its personality, and train the model to refuse unsafe queries.[161] But as an essential part of delivering its models to the world, OpenAI devoted considerable effort to developing an additional innovative "Safety Stack."[162]

Safety Mitigations

102. As the first company to make LLMs publicly accessible, OpenAI encountered the myriad safety issues presented by LLMs at an unprecedented scale.[163] LLMs are capable of responding to anything users submit as a prompt, and LLMs are, by nature, probabilistic, making it difficult to guarantee how they will respond to the infinite range of possible prompts. To address these issues, OpenAI developed its "Safety Stack," a suite of safety solutions implemented at every level of LLM development, as pictured in the below image from an OpenAI internal document.[164]

---

[152] *Id.*
[153] CCB Opening Report ¶ 239.
[154] CCB Opening Report ¶¶ 234–237.
[155] CCB Opening Report ¶ 234.
[156] CCB Opening Report ¶¶ 234–236.
[157] CCB Opening Report ¶ 237.
[158] CCB Opening Report ¶¶ 225–226.
[159] CCB Opening Report ¶ 237.
[160] CCB Opening Report ¶ 224.
[161] *Id.*
[162] CCB Opening Report ¶ 237.
[163] CCB Opening Report ¶¶ 238–239.
[164] *Id.*



103. ***At the pre-training stage***, OpenAI excludes and cleans training data that poses safety risks.[165] Since at least GPT-4, OpenAI has filtered harmful content from its training data, including CSAM, violent or hateful text, and text posing chemical, biological, radiological, and nuclear hazards.[166] OpenAI also reduced the risk that its models would reveal personally identifiable information ("PII") by filtering PII aggregator domains out of its training data and masking PII.[167] OpenAI also maintains a blocklist of website domains whose content is excluded from OpenAI's training pipeline for safety purposes, and respects robots.txt protocols.[168] As discussed below, OpenAI also de-duplicates and masks training data in an effort to reduce the risk that models memorize copyrighted text.[169]

104. ***At the post-training stage***, OpenAI uses a combination of SFT, RLHF, and RBRM, described above, to teach models not to engage in unwanted behavior and to provide responses aligned with desired outcomes.[170] For example, as discussed above, RBRM factors safety into rating outputs by measuring "harmlessness."[171]

105. ***At inference time***, OpenAI operates a suite of tools that mitigate the likelihood of unwanted behavior.[172] For example, OpenAI operates an **instruction hierarchy**, which defines how models should behave when instructions of different priorities conflict, assigning the highest priority to System Messages—instructions set by the application developer that define the model's general behavior, safety guidelines, constraints, and tools.[173]

---

[165] CCB Opening Report ¶ 241.
[166] *Id.*
[167] *Id.*
[168] *Id.*
[169] *Id.*
[170] CCB Opening Report ¶¶ 242–244.
[171] *Id.*
[172] CCB Opening Report ¶¶ 245–246.
[173] *Id.*

106. As another example, OpenAI developed the **Moderation API** ("ModAPI"), a text classification model that classifies prompts and proposed outputs into potentially harmful categories.[174] ModAPI returns classifications and scores that downstream systems can use, together with other safety mechanisms, to determine whether and how a request or response to a user's request should be handled, *e.g.*, hard refusal ("just say no"), soft refusal ("encourage seeking help"), etc.[175]

107. ***At the evaluation stage***, OpenAI engages a set of procedures to ensure its models have been sufficiently trained to avoid unwanted behavior. For example, OpenAI uses "red teaming," in which both humans and LLMs perform adversarial tests in a structured fashion to find vulnerabilities within a given system from the perspective of an attacker.[176] OpenAI also conducts safety **evaluations** that assess several categories, including disallowed content, adversarial prompting, hallucinations, and compliance with instruction hierarchies.[177] OpenAI also performs **continuous monitoring post-deployment** to determine how well safety mitigations are working in practice and identify areas where further mitigations are needed.[178]

Anti-Regurgitation Mitigations

108. OpenAI recognizes that model safety includes reducing the risk that models inadvertently generate text that matches text in the training data.[179] To reduce the risk of regurgitation, OpenAI has implemented additional anti-regurgitation mitigations at multiple stages of LLM deployment.[180]

109. ***De-duplication and masking:*** As discussed above, OpenAI applies de-duplication to remove repeated spans of text from its training data.[181] Since GPT-4, OpenAI has also "masked," or obscured, commonly repeated phrases (up to [redacted] tokens); backpropagation is not performed on these masked tokens, minimizing the possibility of regurgitation.[182] For the same reason these techniques mitigate overfitting, they reduce the risk that the model will inadvertently generate matching text in an output.[183] This is a well-documented phenomenon: in *De-duplicating Training Data Makes Language Models Better,* my colleagues and I found that more than 1% of unprompted outputs from models trained on popular corpora were copied verbatim from training data, and that de-duplication reduced memorized emissions by about 10x.[184]

---

[174] CCB Opening Report ¶ 247.
[175] *Id.*
[176] CCB Opening Report ¶ 249.
[177] *Id.*
[178] *Id.*
[179] CCB Opening Report ¶ 408.
[180] CCB Opening Report ¶ 409.
[181] CCB Opening Report ¶ 410.
[182] CCB Opening Report ¶¶ 410, 241 n.150.
[183] CCB Opening Report ¶ 411.
[184] Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch & Nicholas Carlini, *Deduplicating Training Data Makes Language Models Better*, in Procs. 60th Ann. Meeting Ass'n for Computational Linguistics, 8424, 8424 (May 2022), https://aclanthology.org/2022.acl-long.577/.

110. ***Post-training focused on copyright refusals:*** Beginning with GPT-4, OpenAI explicitly post-trained its models to recognize instances where the model is asked to output copyrighted information and to refuse to do this.[185] OpenAI defined a Copyright Policy, which applies when a prompt requests copyrighted material and outlines a set of rules for acceptable responses.[186] To train its models to follow this policy, OpenAI launched a synthetic data campaign that created: (1) "demonstrations" of ideal outputs to copyright-related prompts and (2) "preference data" which paired copyright-relevant prompts with scored completions of various qualities for reinforcement learning.[187]

111. ***Bloom filter:*** OpenAI also operates a Bloom filter as a last-resort copyright mitigation layer to screen model outputs at inference time. A Bloom filter uses a probabilistic data structure to evaluate strings of characters, known as n-grams, from model completions (*i.e.*, the outputs in response to a user's prompt) against a known dataset of n-grams to determine membership within that dataset. If the Bloom filter suggests that a given sequence of n-grams is within the subject dataset, OpenAI's mechanism will block further display of the output to the user.[188] Bloom filters are well-suited to this task because they have "perfect recall," meaning they will catch every true match without false negatives, though they produce some false positives by occasionally flagging benign outputs. OpenAI has invested significant engineering effort in balancing this tradeoff, because as the filter's dataset grows, false positives increase, causing usability issues such as delays in streaming and erroneous blocking of safe content. For this reason, rather than loading all pre-training data into the filter—which would generate an overwhelming number of false positives and severely restrict the set of text the model could output—OpenAI designed a data pipeline to identify and include only the content most susceptible to regurgitation, keeping the filter within operational limits while still providing an effective backstop against unauthorized reproduction of copyrighted material.[189]

112. ***Copyright-specific evaluations:*** Since 2023, OpenAI has used copyright-specific evaluations ("evals") to assess the efficacy of its copyright-related anti-regurgitation mitigations and prevent deployment of candidate models that fail to meet certain thresholds.[190] These evals cover copyrighted information in [redacted].

113. The evals reflect strong improvements in model behaviors related to regurgitation as a result of the mitigations described above.[191] As an example, the graphs below are from a 2023 OpenAI copyright-specific evaluation. They show how OpenAI's post-training mitigations reduce the regurgitation success rate over the baseline (in blue) for [redacted].[192]

[185] CCB Opening Report ¶ 413.
[186] CCB Opening Report ¶¶ 408, 415.
[187] CCB Opening Report ¶¶ 416–419.
[188] CCB Opening Report ¶ 420.
[189] CCB Opening Report ¶¶ 420–424.
[190] CCB Opening Report ¶¶ 419, 425–426.
[191] CCB Opening Report ¶ 419.
[192] CCB Opening Report ¶ 419.



114. In summary, OpenAI uses multiple layers of defenses to combat regurgitation. At pre-training, OpenAI de-duplicates and masks repeated text to reduce the risk of overfitting. At post-training, OpenAI specifically teaches models to refuse to regurgitate through copyright-specific demonstrations and reward model data that is used in SFT, RLHF, and/or RBRM. At inference time, OpenAI screens candidate outputs for copyrighted text and prevents the model from outputting it. Before and after deployment, OpenAI uses evals to specifically monitor the performance of its models with respect to copyright regurgitation.

115. Based on my experience in the field and my review of OpenAI's multiple layers of defenses against safety and regurgitation risks, I conclude that OpenAI's LLMs and products at issue in this dispute were designed and trained to not engage in undesirable behavior.[193]

X. OVERVIEW OF IN-SCOPE LLMS

116. In this section, I provide an overview of what I understand are the In-Scope LLMs, which include GPT-3, GPT-3.5, GPT-3.5 Turbo, GPT-4, GPT-4 Turbo, GPT-4o, and GPT-4o Mini.[194]

117. ***GPT-3***. GPT-3 was pre-trained on 300 billion tokens (epoch weighted) of text drawn from a dataset consisting of a filtered version of Common Crawl, Wikipedia, and OpenAI-created datasets known as "WebText2," "Books1," and "Books2."[195]

118. ***12B MoE***. The 12B MoE model was a 12 billion parameter model trained in summer and fall of 2021. It was pre-trained on a corpus of [redacted] tokens (epoch weighted),

[193] CCB Opening Report § VII.E.
[194] CCB Opening Report ¶ 250.
[195] CCB Opening Report ¶ 251.

[redacted]% of which were derived from Common Crawl.[196] The dataset was de-duplicated in a number of ways, including [redacted]. The dataset was of somewhat lower quality but was about four times larger than that used to train GPT-3, and the 12B MoE model performed slightly better than GPT-3 on downstream evaluations.[197]

119. ***GPT-3.5***. GPT-3.5 was a [redacted] parameter model trained in late 2021. It was pre-trained on approximately [redacted] tokens (epoch weighted).[198] The GPT-3.5 dataset was known internally as [redacted] and contained a [redacted]. Most of the language data was drawn from a de-duplicated, filtered version of Common Crawl.[199] GPT-3.5 powered ChatGPT when OpenAI launched it on November 30, 2022.[200]

120. ***GPT-4***. GPT-4 was a [redacted] parameter model released in March 2023.[201] It was trained on a dataset called [redacted] roughly [redacted]% of which was derived from Common Crawl, and over [redacted]% of which was scraped from the Internet.[202] This dataset was thoroughly cleaned via extraction of HTML and other web artifacts, de-duplication, heuristic-based exclusions of low-quality data (*e.g.*, due to poor quality scanning), and use of erotic and hate content classifiers to exclude undesirable data. GPT-4 was pre-trained on [redacted] tokens (epoch weighted), consisting of [redacted].[203] GPT-4 was also "multi-modal," meaning it was capable of processing both image and text inputs.[204] GPT-4's context window was 8,192 tokens, and the maximum number of tokens it could output in a response was 8,192 tokens.[205]

121. ***GPT-3.5 Turbo***. GPT-3.5 Turbo was trained in late 2022 and early 2023. It was also trained on the [redacted] dataset and was eventually used to power ChatGPT.[206] OpenAI's goal was for this model to be four times faster than GPT-3.5 with the same level of performance, so it made architectural adjustments to improve the model's [redacted].[207] OpenAI released several versions of this model, with context windows ranging from 4,096 tokens to 16,385 tokens. GPT-3.5 Turbo could output a maximum of 4,096 tokens.[208]

122. ***GPT-4 Turbo***. GPT-4 Turbo was announced in November 2023 as a more capable, cheaper model.[209] [redacted][210] In addition to updating the model's knowledge cutoff from September

[196] CCB Opening Report ¶ 252–253.
[197] *Id.*
[198] CCB Opening Report ¶ 254.
[199] *Id.*
[200] *Id.*
[201] CCB Opening Report ¶ 255.
[202] *Id.*
[203] *Id.*
[204] CCB Opening Report ¶ 256.
[205] CCB Opening Report ¶ 255.
[206] CCB Opening Report ¶ 257.
[207] *Id.*
[208] *Id.*
[209] CCB Opening Report ¶ 258.
[210] *Id.*

2021 to April 2023, this expanded the model's context window to 128,000 tokens to allow it to consider longer inputs.[211] GPT-4 Turbo could output up to 4,096 tokens.[212] I understand that OpenAI refers to the process it used to create GPT-4 Turbo as "mid-training."[213]

123. ***GPT-4o***. GPT-4o was announced in May 2024.[214] It was OpenAI's first model that accepted any combination of text, audio, images, and video as inputs and could generate any combination of text, audio, and image outputs.[215] GPT-4o was pre-trained on a dataset referred to internally as [redacted] the largest OpenAI had compiled to date, which contained [redacted][216] It was processed to filter out spam, extract natural language from HTML, and to perform [redacted][217] GPT-4o was pre-trained on approximately [redacted] tokens (epoch weighted), had a 128,000 token context window, and could output a maximum of 16,384 tokens.[218]

124. ***GPT-4o Mini***. GPT-4o mini was launched in July 2024 as a fast, low-cost model.[219] Like GPT-4o, it was multimodal and had a 128,000 token context window. It was pre-trained on [redacted] tokens (epoch weighted).[220]

* * *

I make this declaration based on my relevant education, research, knowledge, and personal and professional experience and judgment in the field of LLM research, as well as the materials and sources cited in my reports, including the evidence produced in this case and conversations with OpenAI researchers.

---

[211] *Id.*
[212] *Id.*
[213] CCB Opening Report ¶ 259.
[214] CCB Opening Report ¶¶ 260–261.
[215] *Id.*
[216] CCB Opening Report ¶ 262.
[217] *Id.*
[218] *Id.*
[219] CCB Opening Report ¶ 263.
[220] *Id.*

I declare under penalty of perjury that the foregoing is true and correct, and that I have personal knowledge of these facts and am competent to testify thereto if called upon to do so.

Dated: September 3, 2026
Philadelphia, PA

By: Chris Callison-Burch
Chris Callison-Burch