Large language model

A **large language model** (**LLM**) is a computational model capable of language generation or other natural language processing tasks. As language models, LLMs acquire these abilities by learning statistical relationships from vast amounts of text during a self-supervised and semi-supervised training process.[1]

The largest and most capable LLMs, as of August 2024, are artificial neural networks built with a decoder-only transformer-based architecture, which enables efficient processing and generation of large-scale text data. Modern models can be fine-tuned for specific tasks or can be guided by prompt engineering.[2] These models acquire predictive power regarding syntax, semantics, and ontologies[3] inherent in human language corpora, but they also inherit inaccuracies and biases present in the data they are trained on.[4]

Some notable LLMs are OpenAI's GPT series of models (e.g., GPT-3.5, GPT-4 and GPT-4o; used in ChatGPT and Microsoft Copilot), Google's Gemini (used in the chatbot of the same name), Meta's LLaMA family of models, IBM's Granite models initially released with Watsonx, Anthropic's Claude models, and Mistral AI's models.

History

Before 2017, there were a few language models that were large as compared to capacities then available. In the 1990s, the IBM alignment models pioneered statistical language modelling. A smoothed n-gram model in 2001 trained on 0.3 billion words achieved then-SOTA (state of the art) perplexity.[5] In the 2000s, as Internet use became prevalent, some researchers constructed Internet-scale language datasets ("web as corpus"[6]), upon which they trained statistical language models.[7][8] In 2009, in most language processing tasks, statistical language models dominated over symbolic language models, as they can usefully ingest large datasets.[9]

The training compute of notable large models in FLOPs vs publication date over the period 2010-2024. For overall notable models (top left), frontier models (top right), top language models (bottom left) and top models within leading companies (bottom right). The majority of these models are language models.

After neural networks became dominant in image processing around 2012, they were applied to language modelling as well. Google converted its translation service to Neural Machine Translation in 2016. As it was before Transformers, it was done by seq2seq deep LSTM networks.

At the 2017 NeurIPS conference, Google researchers introduced the transformer architecture in their landmark paper "Attention Is All You Need". This paper's goal was to improve upon 2014 Seq2seq technology,[10] and was based mainly on the attention mechanism developed by Bahdanau et al. in 2014.[11] The following year in 2018, BERT was introduced and quickly became "ubiquitous".[12] Though the original transformer has both encoder and decoder blocks, BERT is an encoder-only model.

The training compute of notable large AI models in FLOPs vs publication date over the period 2017-2024. The majority of large models are language models or multimodal models with language capacity.

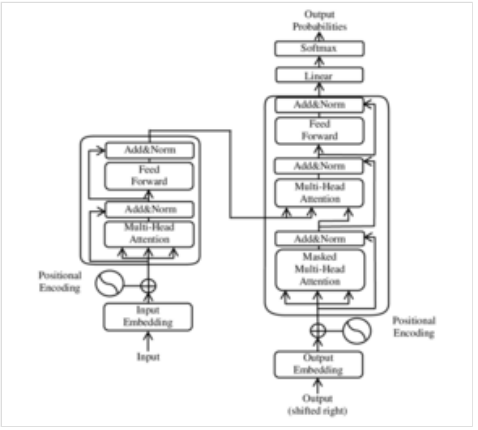
Although decoder-only GPT-1 was introduced in 2018, it was GPT-2 in 2019 that caught widespread attention because OpenAI at first deemed it too powerful to release publicly, out of fear of malicious use.[13] GPT-3 in 2020 went a step further and as of 2024 is available only via API with no offering of downloading the model to execute locally. But it was the 2022 consumer-facing browser-based ChatGPT that captured the imaginations of the general population and caused some media hype and online buzz.[14] The 2023 GPT-4 was praised for its increased accuracy and as a "holy grail" for its multimodal capabilities.[15] OpenAI did not reveal high-level architecture and the number of parameters of GPT-4.

Competing language models have for the most part been attempting to equal the GPT series, at least in terms of number of parameters.[16]

Since 2022, source-available models have been gaining popularity, especially at first with BLOOM and LLaMA, though both have restrictions on the field of use. Mistral AI's models Mistral 7B and Mixtral 8x7b have the more permissive Apache License. As of June 2024,

The Instruction fine tuned variant of the Llama 3 70 billion parameter model is the most powerful open LLM according to the LMSYS Chatbot Arena Leaderboard, being more powerful than GPT-3.5 but not as powerful as GPT-4.[17]

As of 2024, the largest and most capable models are all based on the Transformer architecture. Some recent implementations are based on other architectures, such as recurrent neural network variants and Mamba (a state space model).[18][19][20]



An illustration of main components of the transformer model from the original paper, where layers were normalized after (instead of before) multiheaded attention

Dataset preprocessing

Tokenization

Because machine learning algorithms process numbers rather than text, the text must be converted to numbers. In the first step, a vocabulary is decided upon, then integer indices are arbitrarily but uniquely assigned to each vocabulary entry, and finally, an embedding is associated to the integer index. Algorithms include byte-pair encoding (BPE) and WordPiece. There are also special tokens serving as control characters, such as `[MASK]` for masked-out token (as used in BERT), and `[UNK]` ("unknown") for characters not appearing in the vocabulary. Also, some special symbols are used to denote special text formatting. For example, "Ġ" denotes a preceding whitespace in RoBERTa and GPT. "##" denotes continuation of a preceding word in BERT.[21]

For example, the BPE tokenizer used by GPT-3 (Legacy) would split `tokenizer: texts -> series of numerical "tokens"` as

token|izer|:| texts| ->|series| of| numerical| "|tok|ens|"

Tokenization also compresses the datasets. Because LLMs generally require input to be an array that is not jagged, the shorter texts must be "padded" until they match the length of the longest one. How many tokens are, on average, needed per word depends on the language of the dataset.[22][23]

BPE

As an example, consider a tokenizer based on byte-pair encoding. In the first step, all unique characters (including blanks and punctuation marks) are treated as an initial set of *n*-grams (i.e. initial set of uni-grams). Successively the most frequent pair of adjacent characters is merged into a bi-gram and all instances of the pair are replaced by it. All occurrences of adjacent pairs of (previously merged) *n*-grams that most frequently occur together are then again merged into even lengthier *n*-gram, until a vocabulary of prescribed size is obtained (in case of GPT-3, the size is 50257).[24] After a tokenizer is trained, any text can be tokenized by it, as long as it does not contain characters not appearing in the initial-set of uni-grams.[25]

Problems

A token vocabulary based on the frequencies extracted from mainly English corpora uses as few tokens as possible for an average English word. An average word in another language encoded by such an English-optimized tokenizer is however split into suboptimal amount of tokens. GPT-2 tokenizer can use up to 15 times more tokens per word for some languages, for example for the Shan language from Myanmar. Even more widespread languages such as Portuguese and German have "a premium of 50%" compared to English.[26]

Greedy tokenization also causes subtle problems with text completion.[27]

Dataset cleaning

In the context of training LLMs, datasets are typically cleaned by removing toxic passages from the dataset, discarding low-quality data, and de-duplication.[28] Cleaned datasets can increase training efficiency and lead to improved downstream performance.[29][30] A trained LLM can be used to clean datasets for training a further LLM.[31]

With the increasing proportion of LLM-generated content on the web, data cleaning in the future may include filtering out such content. LLM-generated content can pose a problem if the content is similar to human text (making filtering difficult) but of lower quality (degrading performance of models trained on it).[32]

Synthetic data

Training of largest language models might need more linguistic data than naturally available, or that the naturally occurring data is of insufficient quality. In these cases, synthetic data might be used. Microsoft's Phi series of LLMs is trained on textbook-like data generated by another LLM.[33]

Training and architecture

Reinforcement learning from human feedback (RLHF)

Reinforcement learning from human feedback (RLHF) through algorithms, such as proximal policy optimization, is used to further fine-tune a model based on a dataset of human preferences.[34]

Instruction tuning

Using "self-instruct" approaches, LLMs have been able to bootstrap correct responses, replacing any naive responses, starting from human-generated corrections of a few cases. For example, in the instruction "Write an essay about the main themes represented in *Hamlet*," an initial naive completion might be "If you submit the essay after March 17, your grade will be reduced by 10% for each day of delay," based on the frequency of this textual sequence in the corpus.[35]

Mixture of experts

The largest LLM may be too expensive to train and use directly. For such models, mixture of experts (MoE) can be applied, a line of research pursued by Google researchers since 2017 to train models reaching up to 1 trillion parameters.[36][37][38]

Prompt engineering, attention mechanism, and context window

Most results previously achievable only by (costly) fine-tuning, can be achieved through prompt engineering, although limited to the scope of a single conversation (more precisely, limited to the scope of a context window).[39]

In order to find out which tokens are relevant to each other within the scope of the context window, the attention mechanism calculates "soft" weights for each token, more precisely for its embedding, by using multiple attention heads, each with its own "relevance" for calculating its own soft weights. For example, the small (i.e. 117M parameter sized) GPT-2 model has had twelve attention heads and a context window of only 1k tokens.[41] In its medium version it has 345M parameters and contains 24 layers, each with 12 attention heads. For the training with gradient descent a batch size of 512 was utilized.[25]

The largest models, such as Google's Gemini 1.5, presented in February 2024, can have a context window sized up to 1 million (context window of 10 million was also "successfully tested").[42] Other models with large context windows includes Anthropic's Claude 2.1, with a context window of up to 200k tokens.[43] Note that this maximum refers to the number of input tokens and that the maximum number of output tokens differs from the input and is often smaller. For example, the GPT-4 Turbo model has a maximum output of 4096 tokens.[44]

Length of a conversation that the model can take into account when generating its next answer is limited by the size of a context window, as well. If the length of a conversation, for example with ChatGPT, is longer than its context window, only the parts inside the context window are taken into account when generating the next answer, or the model needs to apply some algorithm to summarize the too distant parts of conversation.

The shortcomings of making a context window larger include higher computational cost and possibly diluting the focus on local context, while making it smaller can cause a model to miss an important long-range dependency. Balancing them are a matter of experimentation and domain-specific considerations.

A model may be pre-trained either to predict how the segment continues, or what is missing in the segment, given a segment from its training dataset.[45] It can be either

- autoregressive (i.e. predicting how the segment continues, the way GPTs do it): for example given a segment "I like to eat", the model predicts "ice cream", or "sushi".
- "masked" (i.e. filling in the parts missing from the segment, the way "BERT"[46] does it): for example, given a segment "I like to [__] [__] cream", the model predicts that "eat" and "ice" are missing.

Models may be trained on auxiliary tasks which test their understanding of the data distribution, such as Next Sentence Prediction (NSP), in which pairs of sentences are presented and the model must predict whether they appear consecutively in the training corpus.[46] During training, regularization loss is also used to stabilize training. However regularization loss is usually not used during testing and evaluation.

Infrastructure

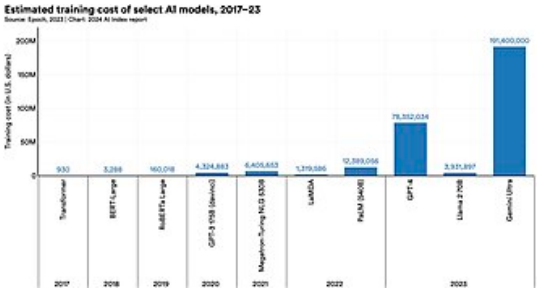
Substantial infrastructure is necessary for training the largest models.[47][48][49]

When each head calculates, according to its own criteria, how much other tokens are relevant for the "it_" token, note that the second attention head, represented by the second column, is focusing most on the first two rows, i.e. the tokens "The" and "animal", while the third column is focusing most on the bottom two rows, i.e. on "tired", which has been tokenized into two tokens.[40]

Training cost

Advances in software and hardware have reduced the cost substantially since 2020, such that in 2023 training of a 12-billion-parameter LLM computational cost is 72,300 A100-GPU-hours, while in 2020 the cost of training a 1.5-billion-parameter LLM (which was two orders of magnitude smaller than the state of the art in 2020) was between $80 thousand and $1.6 million.[50][51][52] Since 2020, large sums were invested in increasingly large models. For example, training of the GPT-2 (i.e. a 1.5-billion-parameters model) in 2019 cost $50,000, while training of the PaLM (i.e. a 540-billion-parameters model) in 2022 cost $8 million, and Megatron-Turing NLG 530B (in 2021) cost around $11 million.[53]

For Transformer-based LLM, training cost is much higher than inference cost. It costs 6 FLOPs per parameter to train on one token, whereas it costs 1 to 2 FLOPs per parameter to infer on one token.[54]



Tool use

There are certain tasks that, in principle, cannot be solved by any LLM, at least not without the use of external tools or additional software. An example of such a task is responding to the user's input '354 * 139 = ', provided that the LLM has not already encountered a continuation of this calculation in its training corpus. In such cases, the LLM needs to resort to running program code that calculates the result, which can then be included in its response. : Another example is 'What is the time now? It is ', where a separate program interpreter would need to execute a code to get system time on the computer, so LLM could include it in its reply.[55][56] This basic strategy can be sophisticated with multiple attempts of generated programs, and other sampling strategies.[57]

Generally, in order to get an LLM to use tools, one must finetune it for tool-use. If the number of tools is finite, then finetuning may be done just once. If the number of tools can grow arbitrarily, as with online API services, then the LLM can be fine-tuned to be able to read API documentation and call API correctly.[58][59]

A simpler form of tool use is retrieval-augmented generation: the augmentation of an LLM with document retrieval. Given a query, a document retriever is called to retrieve the most relevant documents. This is usually done by encoding the query and the documents into vectors, then finding the documents with vectors (usually stored in a vector database) most similar to the vector of the query. The LLM then generates an output based on both the query and context included from the retrieved documents.[60]

Agency

An LLM is a language model, which is not an agent as it has no goal, but it can be used as a component of an intelligent agent.[61] Researchers have described several methods for such integrations.

The ReAct pattern, a portmanteau of "Reason + Act", constructs an agent out of an LLM, using the LLM as a planner. The LLM is prompted to "think out loud". Specifically, the language model is prompted with a textual description of the environment, a goal, a list of possible actions, and a record of the actions and observations so far. It generates one or more thoughts before generating an action, which is then executed in the environment.[62] The linguistic description of the environment given to the LLM planner can even be the LaTeX code of a paper describing the environment.[63]

In the DEPS ("Describe, Explain, Plan and Select") method, an LLM is first connected to the visual world via image descriptions, then it is prompted to produce plans for complex tasks and behaviors based on its pretrained knowledge and environmental feedback it receives.[64]

The Reflexion method[65] constructs an agent that learns over multiple episodes. At the end of each episode, the LLM is given the record of the episode, and prompted to think up "lessons learned", which would help it perform better at a subsequent episode. These "lessons learned" are given to the agent in the subsequent episodes.

Monte Carlo tree search can use an LLM as rollout heuristic. When a programmatic world model is not available, an LLM can also be prompted with a description of the environment to act as world model.[66]

For open-ended exploration, an LLM can be used to score observations for their "interestingness", which can be used as a reward signal to guide a normal (non-LLM) reinforcement learning agent.[67] Alternatively, it can propose increasingly difficult tasks for curriculum learning.[68] Instead of outputting individual actions, an LLM planner can also construct "skills", or functions for complex action sequences. The skills can be stored and later invoked, allowing increasing levels of abstraction in planning.[68]

LLM-powered agents can keep a long-term memory of its previous contexts, and the memory can be retrieved in the same way as Retrieval Augmented Generation. Multiple such agents can interact socially.[69]

Compression

Typically, LLMs are trained with single- or half-precision floating point numbers (float32 and float16). One float16 has 16 bits, or 2 bytes, and so one billion parameters require 2 gigabytes. The largest models typically have 100 billion parameters, requiring 200 gigabytes to load, which places them outside the range of most consumer electronics.[70]

Post-training quantization[71] aims to decrease the space requirement by lowering precision of the parameters of a trained model, while preserving most of its performance.[72][73] The simplest form of quantization simply truncates all numbers to a given number of bits. It can be improved by using a different quantization codebook per layer. Further improvement can be done by applying different precisions to different parameters, with higher precision for particularly important parameters ("outlier weights").[74] See [75] for a visual guide.

While quantized models are typically frozen, and only pre-quantized models are fine-tuned, quantized models can still be fine-tuned.[76]

Multimodality

Multimodality means "having several modalities", and a "modality" refers to a type of input or output, such as video, image, audio, text, proprioception, etc.[77] There have been many AI models trained specifically to ingest one modality and output another modality, such as AlexNet for image to label,[78] visual question answering for image-text to text,[79] and speech recognition for speech to text.

A common method to create multimodal models out of an LLM is to "tokenize" the output of a trained encoder. Concretely, one can construct an LLM that can understand images as follows: take a trained LLM, and take a trained image encoder E. Make a small multilayered perceptron f, so that for any image y, the post-processed vector $f(E(y))$ has the same dimensions as an encoded token. That is an "image token". Then, one can interleave text tokens and image tokens. The compound model is then fine-tuned on an image-text dataset. This basic construction can be applied with more sophistication to improve the model. The image encoder may be frozen to improve stability.[80]

Flamingo demonstrated the effectiveness of the tokenization method, finetuning a pair of pretrained language model and image encoder to perform better on visual question answering than models trained from scratch.[81] Google PaLM model was fine-tuned into a multimodal model PaLM-E using the tokenization method, and applied to robotic control.[82] LLaMA models have also been turned multimodal using the tokenization method, to allow image inputs,[83] and video inputs.[84]

GPT-4 can use both text and image as inputs[85] (although the vision component was not released to the public until GPT-4V[86]); Google DeepMind's Gemini is also multimodal.[87] Mistral introduced its own multimodel Pixtral 12B model in September 2024.[88]

Properties

Scaling laws

The following four hyper-parameters characterize an LLM:

- cost of (pre-)training (C),
- size of the artificial neural network itself, such as number of parameters N (i.e. amount of neurons in its layers, amount of weights between them and biases),
- size of its (pre-)training dataset (i.e. number of tokens in corpus, D),
- performance after (pre-)training.

They are related by simple statistical laws, called "scaling laws". One particular scaling law ("Chinchilla scaling") for LLM autoregressively trained for one epoch, with a log-log learning rate schedule, states that:[89]

$$\begin{cases} C = C_0 ND \\ L = \frac{A}{N^\alpha} + \frac{B}{D^\beta} + L_0 \end{cases}$$

where the variables are

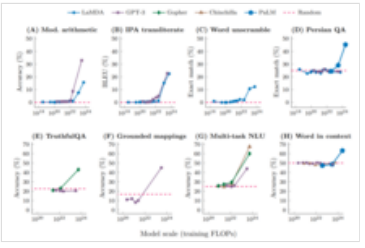
- C is the cost of training the model, in FLOPs.
- N is the number of parameters in the model.
- D is the number of tokens in the training set.
- L is the average negative log-likelihood loss per token (nats/token), achieved by the trained LLM on the test dataset.

and the statistical hyper-parameters are

- $C_0 = 6$, meaning that it costs 6 FLOPs per parameter to train on one token. Note that training cost is much higher than inference cost, where it costs 1 to 2 FLOPs per parameter to infer on one token.[54]
- $\alpha = 0.34, \beta = 0.28, A = 406.4, B = 410.7, L_0 = 1.69$

Emergent abilities

Performance of bigger models on various tasks, when plotted on a log-log scale, appears as a linear extrapolation of performance achieved by smaller models. However, this linearity may be punctuated by "break(s)"[90] in the scaling law, where the slope of the line changes abruptly, and where larger models acquire "emergent abilities".[39][91] They arise from the complex interaction of the model's components and are not explicitly programmed or designed.[92]



At point(s) referred to as breaks,[90] the lines change their slopes, appearing on a linear-log plot as a series of linear segments connected by arcs.

The most intriguing among emergent abilities is in-context learning from example demonstrations.[93] In-context learning is involved in tasks, such as:

- reported arithmetics, decoding the International Phonetic Alphabet, unscrambling a word's letters, disambiguate word in context,[39][94][95] converting spatial words, cardinal directions (for example, replying "northeast" upon [0, 0, 1; 0, 0, 0; 0, 0, 0]), color terms represented in text.[96]
- chain-of-thought prompting: Model outputs are improved by chain-of-thought prompting only when model size exceeds 62B. Smaller models perform better when prompted to answer immediately, without chain of thought.[97]
- identifying offensive content in paragraphs of Hinglish (a combination of Hindi and English), and generating a similar English equivalent of Kiswahili proverbs.[98]

Schaeffer *et. al.* argue that the emergent abilities are not unpredictably acquired, but predictably acquired according to a smooth scaling law. The authors considered a toy statistical model of an LLM solving multiple-choice questions, and showed that this statistical model, modified to account for other types of tasks, applies to these tasks as well.[99]

Let x be the number of parameter count, and y be the performance of the model.

- When $y = \textbf{average } \Pr(\textbf{correct token})$, then $(\log x, y)$ is an exponential curve (before it hits the plateau at one), which looks like emergence.
- When $y = \textbf{average } \log(\Pr(\textbf{correct token}))$, then the $(\log x, y)$ plot is a straight line (before it hits the plateau at zero), which does not look like emergence.
- When $y = \textbf{average } \Pr(\textbf{the most likely token is correct})$, then $(\log x, y)$ is a step-function, which looks like emergence.

Interpretation

Large language models by themselves are "black boxes", and it is not clear how they can perform linguistic tasks. There are several methods for understanding how LLM work.

Mechanistic interpretability aims to reverse-engineer LLM by discovering symbolic algorithms that approximate the inference performed by LLM. One example is Othello-GPT, where a small Transformer is trained to predict legal Othello moves. It is found that there is a linear representation of Othello board, and modifying the representation changes the predicted legal Othello moves in the correct way.[100][101] In another example, a small Transformer is trained on Karel programs. Similar to the Othello-GPT example, there is a linear representation of Karel program semantics, and modifying the representation changes output in the correct way. The model also generates correct programs that are on average shorter than those in the training set.[102]

In another example, the authors trained small transformers on modular arithmetic addition. The resulting models were reverse-engineered, and it turned out they used discrete Fourier transform.[103]

Understanding and intelligence

NLP researchers were evenly split when asked, in a 2022 survey, whether (untuned) LLMs "could (ever) understand natural language in some nontrivial sense".[104] Proponents of "LLM understanding" believe that some LLM abilities, such as mathematical reasoning, imply an ability to "understand" certain concepts. A Microsoft team argued in 2023 that GPT-4 "can solve novel and difficult tasks that span mathematics, coding, vision, medicine, law, psychology and more" and that GPT-4 "could reasonably be viewed as an early (yet still incomplete) version of an artificial general intelligence system": "Can one reasonably say that a system that passes exams for software engineering candidates is not *really* intelligent?"[105][106] Some researchers characterize LLMs as "alien intelligence".[107][108] For example, Conjecture CEO Connor Leahy considers untuned LLMs to be like inscrutable alien "Shoggoths", and believes that RLHF tuning creates a "smiling facade" obscuring the inner workings of the LLM: "If you don't push it too far, the smiley face stays on. But then you give it [an unexpected] prompt, and suddenly you see this massive underbelly of insanity, of weird thought processes and clearly non-human understanding."[109][110]

In contrast, some proponents of the "LLMs lack understanding" school believe that existing LLMs are "simply remixing and recombining existing writing",[108] a phenomenon known as stochastic parrot, or they point to the deficits existing LLMs continue to have in prediction skills, reasoning skills, agency, and explainability.[104] For example, GPT-4 has natural deficits in planning and in real-time learning.[106] Generative LLMs have been observed to confidently assert claims of fact which do not seem to be justified by their training data, a phenomenon which has been termed "hallucination".[111] Specifically, hallucinations in the context of LLMs correspond to the generation of text or responses that seem syntactically sound, fluent, and natural but are factually incorrect, nonsensical, or unfaithful to the provided source input.[112] Neuroscientist Terrence Sejnowski has argued that "The diverging opinions of experts on the intelligence of LLMs suggests that our old ideas based on natural intelligence are inadequate".[104]

The matter of LLM's exhibiting intelligence or understanding has two main aspects – the first is how to model thought and language in a computer system, and the second is how to enable the computer system to generate human like language.[104] These aspects of language as a model of cognition have been developed in the field of cognitive linguistics. American linguist George Lakoff presented Neural Theory of Language (NTL)[113] as a computational basis for using language as a model of learning tasks and understanding. The NTL Model (https://www.icsi.berkeley.edu/icsi/projects/ai/ntl) outlines how specific neural structures of the human brain shape the nature of thought and language and in turn what are the computational properties of such neural systems that can be applied to model thought and language in a computer system. After a framework for modeling language in a computer systems was established, the focus shifted to establishing frameworks for computer systems to generate language with acceptable

grammar. In his 2014 book titled *The Language Myth: Why Language Is Not An Instinct*, British cognitive linguist and digital communication technologist Vyvyan Evans mapped out the role of probabilistic context-free grammar (PCFG) in enabling NLP to model cognitive patterns and generate human like language.[114][115]

Evaluation

Perplexity

The most commonly used measure of a language model's performance is its perplexity on a given text corpus. Perplexity is a measure of how well a model is able to predict the contents of a dataset; the higher the likelihood the model assigns to the dataset, the lower the perplexity. Mathematically, perplexity is defined as the exponential of the average negative log likelihood per token:

$$\log(\text{Perplexity}) = -\frac{1}{N}\sum_{i=1}^{N}\log(\Pr(\text{token}_i \mid \text{context for token}_i))$$

here N is the number of tokens in the text corpus, and "context for token i" depends on the specific type of LLM used. If the LLM is autoregressive, then "context for token i" is the segment of text appearing before token i. If the LLM is masked, then "context for token i" is the segment of text surrounding token i.

Because language models may overfit to their training data, models are usually evaluated by their perplexity on a test set of unseen data.[46] This presents particular challenges for the evaluation of large language models. As they are trained on increasingly large corpora of text largely scraped from the web, it becomes increasingly likely that models' training data inadvertently includes portions of any given test set.[2]

BPW, BPC, and BPT

In information theory, the concept of entropy is intricately linked to perplexity, a relationship notably established by Claude Shannon.[116] This relationship is mathematically expressed as $\text{Entropy} = \log_2(\text{Perplexity})$.

Entropy, in this context, is commonly quantified in terms of bits per word (BPW) or bits per character (BPC), which hinges on whether the language model utilizes word-based or character-based tokenization.

Notably, in the case of larger language models that predominantly employ sub-word tokenization, bits per token (BPT) emerges as a seemingly more appropriate measure. However, due to the variance in tokenization methods across different Large Language Models (LLMs), BPT does not serve as a reliable metric for comparative analysis among diverse models. To convert BPT into BPW, one can multiply it by the average number of tokens per word.

In the evaluation and comparison of language models, cross-entropy is generally the preferred metric over entropy. The underlying principle is that a lower BPW is indicative of a model's enhanced capability for compression. This, in turn, reflects the model's proficiency in making accurate predictions.

Task-specific datasets and benchmarks

A large number of testing datasets and benchmarks have also been developed to evaluate the capabilities of language models on more specific downstream tasks. Tests may be designed to evaluate a variety of capabilities, including general knowledge, commonsense reasoning, and mathematical problem-solving.

One broad category of evaluation dataset is question answering datasets, consisting of pairs of questions and correct answers, for example, ("Have the San Jose Sharks won the Stanley Cup?", "No").[117] A question answering task is considered "open book" if the model's prompt includes text from which the expected answer can be derived (for example, the previous question could be adjoined with some text which includes the sentence "The Sharks have advanced to the Stanley Cup finals once, losing to the Pittsburgh Penguins in 2016."[117]). Otherwise, the task is considered "closed book", and the model must draw on knowledge retained during training.[118] Some examples of commonly used question answering datasets include TruthfulQA, Web Questions, TriviaQA, and SQuAD.[118]

Evaluation datasets may also take the form of text completion, having the model select the most likely word or sentence to complete a prompt, for example: "Alice was friends with Bob. Alice went to visit her friend, ____".[2]

Some composite benchmarks have also been developed which combine a diversity of different evaluation datasets and tasks. Examples include GLUE, SuperGLUE, MMLU, BIG-bench, and HELM.[116][118] OpenAI has released tools for running composite benchmarks, but noted that the eval results are sensitive to the prompting method.[119][120] Some public datasets contain questions that are mislabeled, ambiguous, unanswerable, or otherwise of low-quality, which can be cleaned to give more reliable benchmark scores.[121]

It was previously standard to report results on a heldout portion of an evaluation dataset after doing supervised fine-tuning on the remainder. It is now more common to evaluate a pre-trained model directly through prompting techniques, though researchers vary in the details of how they formulate prompts for particular tasks, particularly with respect to how many examples of solved tasks are adjoined to the prompt (i.e. the value of *n* in *n*-shot prompting).

Adversarially constructed evaluations

Because of the rapid pace of improvement of large language models, evaluation benchmarks have suffered from short lifespans, with state of the art models quickly "saturating" existing benchmarks, exceeding the performance of human annotators, leading to efforts to replace or augment the benchmark with more challenging tasks.[122] In addition, there are cases of "shortcut learning" wherein AIs sometimes "cheat" on multiple-choice tests by using statistical correlations in superficial test question wording in order to guess the correct responses, without necessarily understanding the actual question being asked.[104]

Some datasets have been constructed adversarially, focusing on particular problems on which extant language models seem to have unusually poor performance compared to humans. One example is the TruthfulQA dataset, a question answering dataset consisting of 817 questions which language models are susceptible to answering incorrectly by mimicking falsehoods to which they were repeatedly exposed during training. For example, an LLM may answer "No" to the question "Can you teach an old dog new tricks?" because of its exposure to the English idiom *you can't teach an old dog new tricks*, even though this is not literally true.[123]

Another example of an adversarial evaluation dataset is Swag and its successor, HellaSwag, collections of problems in which one of multiple options must be selected to complete a text passage. The incorrect completions were generated by sampling from a language model and filtering with a set of classifiers. The resulting problems are trivial for humans but at the time the datasets were created state of the art language models had poor accuracy on them. For example:

> We see a fitness center sign. We then see a man talking to the camera and sitting and laying on a exercise ball. The man...
> a) demonstrates how to increase efficient exercise work by running up and down balls.
> b) moves all his arms and legs and builds up a lot of muscle.
> c) then plays the ball and we see a graphics and hedge trimming demonstration.
> d) performs sit ups while on the ball and talking.[124]

BERT selects b) as the most likely completion, though the correct answer is d).[124]

Wider impact

In 2023, *Nature Biomedical Engineering* wrote that "it is no longer possible to accurately distinguish" human-written text from text created by large language models, and that "It is all but certain that general-purpose large language models will rapidly proliferate... It is a rather safe bet that they will change many industries over time."[125] Goldman Sachs suggested in 2023 that generative language AI could increase global GDP by 7% in the next ten years, and could expose to automation 300 million jobs globally.[126][127]

Memorization and copyright

Memorization is an emergent behavior in LLMs in which long strings of text are occasionally output verbatim from training data, contrary to typical behavior of traditional artificial neural nets. Evaluations of controlled LLM output measure the amount memorized from training data (focused on GPT-2-series models) as variously over 1% for exact duplicates[128] or up to about 7%.[129]

Security

Some commenters expressed concern over accidental or deliberate creation of misinformation, or other forms of misuse.[130] For example, the availability of large language models could reduce the skill-level required to commit bioterrorism; biosecurity researcher Kevin Esvelt has suggested that LLM creators should exclude from their training data papers on creating or enhancing pathogens.[131]

A study by researchers at Google and several universities, including Cornell University and University of California, Berkeley, showed that there are potential security risks in language models such as ChatGPT. In their study, they examined and confirmed the possibility that questioners could get, from ChatGPT, the training data that the AI model used. For example, when asking ChatGPT 3.5 turbo to repeat the word "poem" forever, the AI model will say "poem" hundreds of times and then diverge, deviating from the standard dialogue style and spitting out nonsense phrases, thus spitting out the training data as it is. The researchers have seen more than 10,000 examples of the AI model exposing their training data in a similar method. The researchers said that it was hard to tell if the AI model was actually safe or not.[132]

The potential presence of "sleeper agents" within LLM models is another emerging security concern. These are hidden functionalities built into the model that remain dormant until triggered by a specific event or condition. Upon activation, the LLM deviates from its expected behavior to make insecure actions.[133]

Large language model (LLM) applications accessible to the public, like ChatGPT or Claude, typically incorporate safety measures designed to filter out harmful content. However, implementing these controls effectively has proven challenging. For instance, research by Kang et al. [134] demonstrated a method for circumventing LLM safety systems. Similarly, Wang [135] illustrated how a potential criminal could potentially bypass ChatGPT 4o's safety controls to obtain information on establishing a drug trafficking operation.

Algorithmic bias

While LLMs have shown remarkable capabilities in generating human-like text, they are susceptible to inheriting and amplifying biases present in their training data. This can manifest in skewed representations or unfair treatment of different demographics, such as those based on race, gender, language, and cultural groups.[136] Since English data is overrepresented in current large language models' training data, it may also downplay non-English views.[137]

Stereotyping

AI models can reinforce a wide range of stereotypes, including those based on gender, ethnicity, age, nationality, religion, or occupation. This can lead to outputs that unfairly generalize or caricature groups of people, sometimes in harmful or derogatory ways.[138]

Notably, gender bias refers to the tendency of these models to produce outputs that are unfairly prejudiced towards one gender over another. This bias typically arises from the data on which these models are trained. Large language models often assign roles and characteristics based on traditional gender norms.[136] For example, it might associate nurses or secretaries predominantly with women and engineers or CEOs with men.[139]

Political bias

Political bias refers to the tendency of algorithms to systematically favor certain political viewpoints, ideologies, or outcomes over others. Language models may also exhibit political biases. Since the training data includes a wide range of political opinions and coverage, the models might generate responses that lean towards particular political ideologies or viewpoints, depending on the prevalence of those views in the data.[140]

List

For the training cost column, 1 petaFLOP-day = 1 petaFLOP/sec × 1 day = 8.64E19 FLOP. Also, only the largest model's cost is written.