Title: Large language model

URL: https://en.wikipedia.org/wiki/Large_language_model

PageID: 73248112

Categories: Category:Deep learning, Category:Large language models, Category:Natural language processing

---

Supervised learning

Unsupervised learning

Semi-supervised learning

Self-supervised learning

Reinforcement learning

Meta-learning

Online learning

Batch learning

Curriculum learning

Rule-based learning

Neuro-symbolic AI

Neuromorphic engineering

Quantum machine learning

Classification

Generative modeling

Regression

Clustering

Dimensionality reduction

Density estimation

Anomaly detection

Data cleaning

AutoML

Association rules

Semantic analysis

Structured prediction

Feature engineering

Feature learning

Learning to rank

Grammar induction

Ontology learning

Multimodal learning

Apprenticeship learning

Decision trees

Ensembles Bagging Boosting Random forest

Bagging

Boosting

Random forest

k -NN

Linear regression

Naive Bayes

Artificial neural networks

Logistic regression

Perceptron

Relevance vector machine (RVM)

Support vector machine (SVM)

BIRCH

CURE

Hierarchical

k -means

Fuzzy

Expectation–maximization (EM)

DBSCAN

OPTICS

Mean shift

Factor analysis

CCA

ICA

LDA

NMF

PCA

PGD

t-SNE

SDL

Graphical models Bayes net Conditional random field Hidden Markov

Bayes net

Conditional random field

Hidden Markov

RANSAC

k -NN

Local outlier factor

Isolation forest

Autoencoder

Deep learning

Feedforward neural network

Recurrent neural network LSTM GRU ESN reservoir computing

LSTM

GRU

ESN

reservoir computing

Boltzmann machine Restricted

Restricted

GAN

Diffusion model

SOM

Convolutional neural network U-Net LeNet AlexNet DeepDream

U-Net

LeNet

AlexNet

DeepDream

Neural field Neural radiance field Physics-informed neural networks

Neural radiance field

Physics-informed neural networks

Transformer Vision

Vision

Mamba

Spiking neural network

Memtransistor

Electrochemical RAM (ECRAM)

Q-learning

Policy gradient

SARSA

Temporal difference (TD)

Multi-agent Self-play

Self-play

Active learning

Crowdsourcing

Human-in-the-loop

Mechanistic interpretability

RLHF

Coefficient of determination

Confusion matrix

Learning curve

ROC curve

Kernel machines

Bias–variance tradeoff

Computational learning theory

Empirical risk minimization

Occam learning

PAC learning

Statistical learning

VC theory

Topological deep learning

AAAI

ECML PKDD

NeurIPS

ICML

ICLR

IJCAI

ML

JMLR

Glossary of artificial intelligence

List of datasets for machine-learning research List of datasets in computer vision and image processing

List of datasets in computer vision and image processing

Outline of machine learning

v

t

e

A large language model (LLM) is a language model trained with self-supervised machine learning on a vast amount of text, designed for natural language processing tasks, especially language generation .

The largest and most capable LLMs are generative pre-trained transformers (GPTs), based on a transformer architecture , which are largely used in generative chatbots such as ChatGPT , Gemini and Claude . LLMs can be fine-tuned for specific tasks or guided by prompt engineering . [1] These models acquire predictive power regarding syntax , semantics , and ontologies [2] inherent in human language corpora , but they also inherit inaccuracies and biases present in the data they are trained on. [3]

History

Before the emergence of transformer-based models in 2017, some language models were considered large relative to the computational and data constraints of their time. In the early 1990s, IBM 's statistical models pioneered word alignment techniques for machine translation, laying the groundwork for corpus-based language modeling . In 2001, a smoothed n -gram model , such as those employing Kneser–Ney smoothing , trained on 300 million words, achieved state-of-the-art perplexity on benchmark tests. [4] During the 2000s, with the rise of widespread internet access, researchers began compiling massive text datasets from the web ("web as corpus" [5]) to train statistical language models. [6] [7]

Moving beyond n -gram models, researchers started in 2000 to use neural networks to learn language models. [8] Following the breakthrough of deep neural networks in image classification around 2012, [9] similar architectures were adapted for language tasks. This shift was marked by the development of word embeddings (eg, Word2Vec by Mikolov in 2013) and sequence-to-sequence (seq2seq) models using LSTM . In 2016, Google transitioned its translation service to neural machine translation (NMT), replacing statistical phrase-based models with deep recurrent neural networks . These early NMT systems used LSTM-based encoder-decoder architectures , as they preceded the invention of transformers .

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. Academic and research usage of BERT began to decline in 2023, following rapid improvements in the abilities of decoder-only models (such as GPT) to solve tasks via prompting . [13]

Although decoder-only GPT-1 was introduced in 2018, it was GPT-2 in 2019 that caught widespread attention because OpenAI claimed to have initially deemed it too powerful to release publicly, out of fear of malicious use. [14] GPT-3 in 2020 went a step further and as of 2025 [update] is available only via API with no offering of downloading the model to execute locally. But it was the 2022 consumer-facing chatbot ChatGPT that received extensive media coverage and public attention. [15] The 2023 GPT-4 was praised for its increased accuracy and as a "holy grail" for its multimodal capabilities. [16] OpenAI did not reveal the high-level architecture and the number of parameters of GPT-4. The release of ChatGPT led to an uptick in LLM usage across several research subfields of computer science, including robotics, software engineering, and societal impact work. [13] In 2024 OpenAI released the reasoning model OpenAI o1 , which generates long chains of thought before returning a final answer. [17] Many LLMs with parameter counts comparable to those of OpenAI's GPT series have been developed. [18]

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 . In January 2025, DeepSeek released DeepSeek R1, a 671-billion-parameter open-weight model that performs comparably to OpenAI o1 but at a much lower cost. [19]

Since 2023, many LLMs have been trained to be multimodal , having the ability to also process or generate other types of data, such as images or audio. These LLMs are also called large multimodal models (LMMs). [20]

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). [21] [22] [23]

Dataset preprocessing

Tokenization

As 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, "■" denotes a preceding whitespace in RoBERTa and GPT. "##" denotes continuation of a preceding word in BERT. [24]

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

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. The average number of words per token depends on the language. [25] [26] In English, the ratio is typically around 0.75 words per token, with 4 characters per token on average. [27]

Byte-pair encoding

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. 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. [28]

Problems

A token vocabulary based on the frequencies extracted from mainly English corpora uses as few tokens as possible for an average English word. However, an average word in another language encoded by such an English-optimized tokenizer is split into a 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]

Dataset cleaning

In the context of training LLMs, datasets are typically cleaned by removing low-quality, duplicated, or toxic data. [29] Cleaned datasets can increase training efficiency and lead to improved downstream performance. [30] [31] A trained LLM can be used to clean datasets for training a further LLM. [32]

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). [1]

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

An LLM is a type of foundation model (large X model) trained on language. LLMs can be trained in different ways. In particular, GPT models are first pretrained to predict the next word on a large amount of data, before being fine-tuned. [citation needed]

Cost

Substantial infrastructure is necessary for training the largest models. The tendency towards larger models is visible in the list of large language models . For example, the training of 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. The qualifier "large" in "large language model" is inherently vague, as there is no definitive threshold for the number of parameters required to qualify as "large". GPT-1 of 2018 has 117 million parameters. [citation needed]

Fine-tuning

Before being fine-tuned , most LLMs are next-token predictors. The fine-tuning adjust the output of an LLM to seem more conversational via techniques like reinforcement learning from human feedback (RLHF) or constitutional AI . [34]

Instruction fine-tuning is a form of supervised learning used to teach LLMs to follow user instructions. In 2022, OpenAI demonstrated InstructGPT, a version of GPT-3 similarly fine-tuned to follow instructions. [35]

Reinforcement learning from human feedback (RLHF) involves training a reward model to predict which text humans prefer. Then, the LLM can be fine-tuned through reinforcement learning to better satisfy this reward model. Since humans typically prefer truthful, helpful and harmless answers, RLHF favors such answers. [citation needed]

Architecture

LLMs are generally based on the transformer architecture, which leverages an attention mechanism that enables the model to process relationships between all elements in a sequence simultaneously, regardless of their distance from each other. [citation needed]

Attention mechanism and context window

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. [37] 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. [28]

Google's Gemini 1.5 , introduced in February 2024, can have a context window of up to 1 million tokens. [38]

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. [39] It can be either

autoregressive (i.e. predicting how the segment continues, as GPTs do): 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" [40] 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. [40] During training, regularization loss is also used to stabilize training. However regularization loss is usually not used during testing and evaluation.

Mixture of experts

A mixture of experts (MoE) is a machine learning architecture in which multiple specialized neural networks ("experts") work together, with a gating mechanism that routes each input to the most appropriate expert(s). Mixtures of experts can reduce inference costs, as only a fraction of the parameters are used for each input. The approach was introduced in 2017 by Google researchers. [41] [42] [43]

Parameter size

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. [44]

Quantization

Post-training quantization [45] aims to decrease the space requirement by lowering precision of the parameters of a trained model, while preserving most of its performance. Quantization can be further classified as static quantization if the quantization parameters are determined beforehand (typically during a calibration phase), and dynamic quantization if the quantization is applied during inference. The simplest form of quantization simply truncates all the parameters to a given number of bits: this is applicable to static as well as dynamic quantization, but loses much precision. Dynamic quantization allows for the use of a different quantization codebook per layer, either a lookup table of values or a linear mapping (scaling factor and bias), at the cost of foregoing the possible speed improvements from using lower-precision arithmetic. [citation needed]

Quantized models are typically seen as frozen with modification of weights (e.g. fine-tuning) only applied to the original model. It is possible to fine-tune quantized models using low-rank adaptation . [citation needed]

Extensibility

Beyond basic text generation, various techniques have been developed to extend LLM capabilities, including the use of external tools and data sources, improved reasoning on complex problems, and enhanced instruction-following or autonomy through prompting methods.

Prompt engineering

In 2020, OpenAI researchers demonstrated that their new model GPT-3 could understand what format to use given a few rounds of Q and A (or other type of task) in the input data as example, thanks in part due to the RLHF technique. This technique, called few-shot prompting , allows LLMs to be adapted to any task without requiring fine-tuning. [1] Also in 2022, it was found that the base GPT-3 model can generate an instruction based on user input. The generated instruction along with user input is then used as input to another instance of the model under a "Instruction: [...], Input: [...], Output:" format. The other instance is able to complete the output and often produces the correct answer in doing so. The ability to "self-instruct" makes LLMs able to bootstrap themselves toward a correct answer. [46]

Dialogue processing (chatbot)

An LLM can be turned into a chatbot or a "dialog assistant" by specializing it for conversation. In essence, user input is prefixed with a marker such as "Q:" or "User:" and the LLM is asked to predict the output after a fixed "A:" or "Assistant:". This type of model became commercially available in 2022 with ChatGPT, a sibling model of InstructGPT fine-tuned to accept and produce dialog-formatted text based on GPT-3.5. It could similarly follow user instructions. [47] Before the stream of User and Assistant lines, a chat context usually start with a few lines of overarching instructions, from a role called "developer" or "system" to convey a higher authority than the user's input. This is called a "system prompt". [48] [49]

Retrieval-augmented generation

Retrieval-augmented generation (RAG) is an approach that enhances LLMs by integrating them with document retrieval systems. 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. [50] [51]

Tool use

Tool use is a mechanism that enables LLMs to interact with external systems, applications, or data sources. It can allow for example to fetch real-time information from an API or to execute code. A

program separate from the LLM watches the output stream of the LLM for a special tool-calling syntax. When these special tokens appear, the program calls the tool accordingly and feeds its output back into the LLM's input stream. [52]

Early tool-using LLMs were fine-tuned on the use of specific tools. But fine-tuning LLMs for the ability to read API documentation and call API correctly has greatly expanded the range of tools accessible to an LLM. [53] [54] Describing available tools in the system prompt can also make an LLM able to use tools. A system prompt instructing ChatGPT (GPT-4) to use multiple types of tools can be found online. [55]

Agency

An LLM is typically not an autonomous agent by itself, as it lacks the ability to interact with dynamic environments, recall past behaviors, and plan future actions. But it can be transformed into an agent by adding supporting elements: the role (profile) and the surrounding environment of an agent can be additional inputs to the LLM, while memory can be integrated as a tool or provided as additional input. Instructions and input patterns are used to make the LLM plan actions and tool use is used to potentially carry out these actions. [56]

The ReAct pattern, a portmanteau of reason and 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. [57]

In the DEPS ("describe, explain, plan and select") method, an LLM is first connected to the visual world via image descriptions. It is then prompted to produce plans for complex tasks and behaviors based on its pretrained knowledge and the environmental feedback it receives. [58]

The Reflexion method [59] 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 stored as a form of long-term memory and given to the agent in the subsequent episodes. [59]

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. [60]

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. [61] Alternatively, it can propose increasingly difficult tasks for curriculum learning . [62] 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. [62]

Multiple agents with memory can interact socially. [63]

Reasoning

LLMs are conventionally trained to generate an output without generating intermediate steps. As a result, their performance tends to be subpar on complex questions requiring (at least in humans) intermediate steps of thought. Early research demonstrated that inserting intermediate “scratchpad” computations could improve performance on such tasks. [64] Later methods overcame this deficiency more systematically by breaking tasks into smaller steps for the LLM, either manually or automatically.

Chaining

The "prompt chaining" paradigm was published in 2021. [65] In this method, a user manually breaks a complex problem down into several steps. In each step, the LLM receives as input a prompt telling it what to do and some results from preceeding steps. The result from one step is then reused in a next step, until a final answer is reached. The ability of an LLM to follow instructions means that even non-experts can write a successful collection of step-wise prompts

given a few rounds of trial and error. [66] [67]

A 2022 paper demonstrated a separate technique called " chain-of-thought prompting ", which makes the LLM break the question down autonomously. An LLM is given some examples where the "assistant" verbally breaks down the thought process before arriving at an answer. The LLM mimics these examples and also tries to spend some time generating intermediate steps before providing the final answer. This additional step elicited by prompting improves the correctness of the LLM on relatively complex questions. On math word questions, a prompted model can exceed even fine-tuned GPT-3 with a verifier. [65] [68] Chain-of-thought can also be elicited by simply adding an instruction like "Let's think step by step" to the prompt, in order to encourage the LLM to proceed methodically instead of trying to directly guess the answer. [69]

Follow-up methods included self-consistency prompting, which samples multiple reasoning paths and selects the most common answer, [70] and least-to-most prompting , which decomposes complex problems into simpler subproblems that the model solves sequentially. [71]

Subsequent research also explored reflection , where models iteratively critique and improve their own reasoning, [59] and tool-augmented reasoning , where models make use of external systems such as retrievers or calculators to support problem-solving.

Model-native reasoning

In late 2024 "reasoning models" were released. These were trained to spend more time generating step-by-step solutions before providing final answers, which was intended to be similar to human problem-solving processes. [citation needed] OpenAI introduced this concept with their o1 model in September 2024, followed by o3 in April 2025. On the International Mathematics Olympiad qualifying exam problems, GPT-4o achieved 13% accuracy while o1 reached 83%. [72]

In January 2025, the Chinese company DeepSeek released DeepSeek-R1, a 671-billion-parameter open-weight reasoning model that achieved comparable performance to OpenAI's o1 while being significantly more cost-effective to operate. Unlike proprietary models from OpenAI, DeepSeek-R1's open-weight nature allowed researchers to study and build upon the algorithm, though its training data remained private. [73]

These reasoning models typically require more computational resources per query compared to traditional LLMs, as they perform more extensive processing to work through problems step-by-step. [72]

Inference optimization

Inference optimization refers to techniques that improve LLM performance by applying additional computational resources during the inference process, rather than requiring model retraining. These approaches implement various state-of-the-art reasoning and decision-making strategies to enhance accuracy and capabilities.

OptiLLM is an OpenAI API-compatible optimizing inference proxy that implements multiple inference optimization techniques simultaneously. [74] The system acts as a transparent proxy that can work with any LLM provider, implementing techniques such as Monte Carlo tree search (MCTS), mixture of agents (MOA), best-of-N sampling, and chain-of-thought reflection. OptiLLM demonstrates that strategic application of computational resources at inference time can substantially improve model performance across diverse tasks, achieving significant improvements on benchmarks such as the AIME 2024 mathematics competition and various coding challenges. [75]

These inference optimization approaches represent a growing category of tools that enhance existing LLMs without requiring access to model weights or retraining, making advanced reasoning capabilities more accessible across different model providers and use cases.

Forms of input and output

Multimodality

Multimodality means having multiple modalities, where a " modality " refers to a type of input or output, such as video, image, audio, text, proprioception , etc. [76] For example, Google PaLM

model was fine-tuned into a multimodal model and applied to robotic control . [77] LLaMA models have also been turned multimodal using the tokenization method, to allow image inputs, [78] and video inputs. [79] GPT-4o can process and generate text, audio and images. [80] Such models are sometimes called large multimodal models (LMMs). [81]

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 {\displaystyle E} . Make a small multilayered perceptron f {\displaystyle f} , so that for any image y {\displaystyle y} , the post-processed vector f (E (y)) {\displaystyle 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. [82] This type of method, where embeddings from multiple modalities are fused and the predictor is trained on the combined embeddings, is called early fusion.

Another method, called intermediate fusion, involves each modality being first processed independently to obtain modality-specific representations; then these intermediate representations are fused together. [83] In general, cross-attention is used for integrating information from different modalities. As an example, the model Flamingo uses cross-attention layers to inject visual information into its pre-trained language model. [84]

Non-natural languages

LLMs can handle programming languages similarly to how they handle natural languages. No special change in token handling is needed as code, like human language, is represented as plain text. LLMs can generate code based on problems or instructions written in natural language . They can also describe code in natural language or translate between programming languages. They were originally used as a code completion tool, but advances have moved them towards automatic programming . Services such as GitHub Copilot offer LLMs specifically trained, fine-tuned, or prompted for programming. [85] [86]

LLM architectures have also proven useful in analyzing biological sequences: protein, DNA, and RNA. With proteins they appear able to capture a degree of "grammar" from the amino-acid sequence, condensing a sequence into an embedding . On tasks such as structure prediction and mutational outcome prediction, a small model using an embedding as input can approach or exceed much larger models using multiple sequence alignments (MSA) as input. [87] ESMFold, Meta Platforms ' embedding-based method for protein structure prediction, runs an order of magnitude faster than AlphaFold2 thanks to the removal of an MSA requirement and a lower parameter count due to the use of embeddings. [88] Meta hosts ESM Atlas, a database of 772 million structures of metagenomic proteins predicted using ESMFold. [89] An LLM can also design proteins unlike any seen in nature. [90] Nucleic acid models have proven useful in detecting regulatory sequences , [91] sequence classification, RNA-RNA interaction prediction, and RNA structure prediction. [92]

Properties

Scaling laws

The performance of an LLM after pretraining largely depends on the:

cost of pretraining C {\displaystyle C} (the total amount of compute used),

size of the artificial neural network itself, such as number of parameters N {\displaystyle N} (i.e. amount of neurons in its layers, amount of weights between them and biases),

size of its pretraining dataset (i.e. number of tokens in corpus, D {\displaystyle D}).

"Scaling laws" are empirical statistical laws that predict LLM performance based on such factors. One particular scaling law (" Chinchilla scaling ") for LLM autoregressively trained for one epoch, with a log-log learning rate schedule, states that: [93] { C = C 0 N D L = A N α + B D β + L 0 {\displaystyle {\begin{cases}C=C_{0}ND\\[6pt]L={\frac {A}{N^{\alpha }}}+{\frac {B}{D^{\beta }}}+L_{0}\end{cases}}} where the variables are

C {\displaystyle C} is the cost of training the model, in FLOPs .

N {\displaystyle N} is the number of parameters in the model.

D {\displaystyle D} is the number of tokens in the training set.

L {\displaystyle 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 {\displaystyle 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.

α = 0.34 , β = 0.28 , A = 406.4 , B = 410.7 , L 0 = 1.69 {\displaystyle \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) " [94] in the scaling law, where the slope of the line changes abruptly, and where larger models acquire "emergent abilities". [95] [96] They arise from the complex interaction of the model's components and are not explicitly programmed or designed. [97]

One of the emergent abilities is in-context learning from example demonstrations. [98] In-context learning is involved in tasks, such as:

reported arithmetics

decoding the International Phonetic Alphabet

unscrambling a word's letters

disambiguating word-in-context datasets [95] [99] [100]

converting spatial words

cardinal directions (for example, replying "northeast" in response to a 3x3 grid of 8 zeros and a 1 in the top-right), color terms represented in text. [101]

chain-of-thought prompting : In a 2022 research paper, chain-of-thought prompting only improved the performance for models that had at least 62B parameters. Smaller models perform better when prompted to answer immediately, without chain of thought. [102]

identifying offensive content in paragraphs of Hinglish (a combination of Hindi and English), and generating a similar English equivalent of Kiswahili proverbs. [103]

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. [104]

Let x {\displaystyle x} be the number of parameter count, and y {\displaystyle y} be the performance of the model.

When y = average Pr (correct token) {\displaystyle y={\text{average }}\Pr({\text{correct token}})} , then (log ■ x , y) {\displaystyle (\log x,y)} is an exponential curve (before it hits the plateau at one), which looks like emergence.

When y = average log ■ (Pr (correct token)) {\displaystyle y={\text{average }}\log(\Pr({\text{correct token}}))} , then the (log ■ x , y) {\displaystyle (\log x,y)} plot is a straight line (before it hits the plateau at zero), which does not look like emergence.

When y = average Pr (the most likely token is correct) {\displaystyle y={\text{average }}\Pr({\text{the most likely token is correct}})} , then (log ■ x , y) {\displaystyle (\log x,y)} is a

step-function, which looks like emergence.

Interpretation

Large language models are typically regarded as black boxes , and it is not clear how they can perform linguistic tasks. Similarly, it is unclear if or how LLMs should be viewed as models of the human brain and/or human mind. [105]

Mechanistic interpretability

Mechanistic interpretability aims to reverse-engineer LLMs by discovering symbolic algorithms that approximate the inference performed by an LLM. Mechanistic interpretability research has been conducted at organizations like Anthropic and OpenAI, although understanding the inner workings of LLMs remains difficult. [106] [107]

For instance, the authors trained small transformers on modular arithmetic addition . The resulting models were reverse-engineered, and it turned out they used discrete Fourier transform . [108] The training of the model also highlighted a phenomenon called grokking , in which the model initially memorizes all the possible results in the training set (overfitting), and later suddenly learns to actually perform the calculation. [109]

Some techniques have been developed to enhance the transparency and interpretability of LLMs. Transcoders, which are more interpretable than transformers, have been utilized to develop "replacement models". In one such study involving the mechanistic interpretation of writing a rhyming poem by an LLM, it was shown that although they are believed to simply predict the next token, they can, in fact, plan ahead. [110] By integrating such techniques, researchers and practitioners can gain deeper insights into the operations of LLMs, fostering trust and facilitating the responsible deployment of these powerful models.

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". [111] 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?" [112] [113] Ilya Sutskever argues that predicting the next word sometimes involves reasoning and deep insights, for example if the LLM has to predict the name of the criminal in an unknown detective novel after processing the entire story leading up to the revelation. [114] Some researchers characterize LLMs as "alien intelligence". [115] [116] 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." [117] [118]

In contrast, some skeptics of LLM understanding believe that existing LLMs are "simply remixing and recombining existing writing", [116] 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. [111] For example, GPT-4 has natural deficits in planning and in real-time learning. [113] 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 ". [119] 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. [120] 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". [111]

Efforts to reduce or compensate for hallucinations have employed automated reasoning , retrieval-augmented generation (RAG), fine-tuning , and other methods. [121]

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. [111] 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) [122] as a computational basis for using language as a model of learning tasks and understanding. The NTL Model 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. [123] [124]

Evaluation

Perplexity

The canonical measure of the performance of any language model is its perplexity on a given text corpus. Perplexity measures how well a model predicts the contents of a dataset; the higher the likelihood the model assigns to the dataset, the lower the perplexity. In mathematical terms, perplexity is the exponential of the average negative log likelihood per token.

log ■ (Perplexity) = − 1 N ∑ i = 1 N log ■ (Pr (token i ■ context for token i)) {\displaystyle \log({\text{Perplexity}})=-{\frac {1}{N}}\sum _{i=1}^{N}\log(\Pr({\text{token}}_{i}\mid {\text{context for token}}_{i}))}

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

Because language models may overfit to training data, models are usually evaluated by their perplexity on a test set . [40] This evaluation is potentially problematic for larger models which, as they are trained on increasingly large corpora of text, are increasingly likely to inadvertently include portions of any given test set. [125]

Measures

In information theory , the concept of entropy is intricately linked to perplexity, a relationship notably established by Claude Shannon . [126] This relationship is mathematically expressed as Entropy = log 2 ■ (Perplexity) {\displaystyle {\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 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.

Due to their ability to accurately predict the next token, LLMs are highly capable in lossless compression . A 2023 study by DeepMind showed that the model Chinchilla , despite being trained

primarily on text, was able to compress ImageNet to 43% of its size, beating PNG with 58%. [127]

Benchmarks

Benchmarks are used to evaluate LLM performance on specific tasks. Tests evaluate capabilities such as general knowledge, bias, commonsense reasoning , question answering, and mathematical problem-solving. Composite benchmarks examine multiple capabilities. Results are often sensitive to the prompting method. [128] [129]

A question answering benchmark is termed "open book" if the model's prompt includes text from which the expected answer can be derived (for example, the previous question could be combined with text that includes the sentence "The Sharks have advanced to the Stanley Cup finals once, losing to the Pittsburgh Penguins in 2016." [130]). Otherwise, the task is considered "closed book", and the model must draw solely on its training. [131] Examples include GLUE, SuperGLUE, MMLU , BIG-bench, HELM, and HLE (Humanity's Last Exam) . [126] [131]

LLM bias may be assessed through benchmarks such as CrowS-Pairs (Crowdsourced Stereotype Pairs), [132] Stereo Set, [133] and Parity Benchmark. [134]

Fact-checking and misinformation detection benchmarks are available. A 2023 study compared the fact-checking accuracy of LLMs including ChatGPT 3.5 and 4.0, Bard, and Bing AI against independent fact-checkers such as PolitiFact and Snopes. The results demonstrated moderate proficiency, with GPT-4 achieving the highest accuracy at 71%, lagging behind human fact-checkers. [135]

An earlier standard tested using a portion of the evaluation dataset. It became more common to evaluate a pre-trained model directly through prompting techniques. Researchers vary in how they formulate prompts for particular tasks, particularly with respect to the number of correct examples attached to the prompt (i.e. the value of n in n -shot prompting).

Datasets

Typical datasets consist of pairs of questions and correct answers, for example, ("Have the San Jose Sharks won the Stanley Cup?", "No"). [130] Some examples of commonly used question answering datasets include TruthfulQA, Web Questions, TriviaQA, and SQuAD. [131]

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, ____". [125]

Datasets are of varying quality and may contain questions that are mislabeled, ambiguous, unanswerable, or otherwise of low-quality. [136]

Adversarial evaluations

LLMs' rapid improvement regularly renders benchmarks obsolete, with the models exceeding the performance of human annotators. [137] In addition, "shortcut learning" allows AIs to "cheat" on multiple-choice tests by using statistical correlations in superficial test question wording to guess the correct responses, without considering the specific question. [111]

Some datasets are adversarial, focusing on problems that confound LLMs. One example is the TruthfulQA dataset, a question answering dataset consisting of 817 questions that stump LLMs by mimicking falsehoods to which they were 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. [138]

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. The resulting problems are trivial for humans but defeated LLMs. Sample questions:

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...

demonstrates how to increase efficient exercise work by running up and down balls.

moves all his arms and legs and builds up a lot of muscle.

then plays the ball and we see a graphics and hedge trimming demonstration.

performs sit ups while on the ball and talking. [139]

BERT selects 2) as the most likely completion, though the correct answer is 4). [139]

Ethical issues

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." [140] 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. [141] [142] Brinkmann et al. (2023) [143] also argue that LLMs are transforming processes of cultural evolution by shaping processes of variation, transmission, and selection.

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 networks. 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 [144] or up to about 7%. [145]

A 2023 study showed that when ChatGPT 3.5 turbo was prompted to repeat the same word indefinitely, after a few hundreds of repetitions, it would start outputting excerpts from its training data. [146]

Security

Some commenters expressed concern over accidental or deliberate creation of misinformation, or other forms of misuse. [147] 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. [148]

Researchers from Anthropic found that it was possible to create "sleeper agents", models with hidden functionalities that remain dormant until triggered by a specific event or condition. Upon activation, the LLM deviates from its expected behavior to make insecure actions. For example, a LLM could produce safe code except on a specific date, or if the prompt contains a specific tag. These functionalities were found to be difficult to detect or remove via safety training. [149]

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, a 2023 study [150] proposed a method for circumventing LLM safety systems. In 2025, The American Sunlight Project, a non-profit, published a study [151] showing evidence that the so-called Pravda network , a pro-Russia propaganda aggregator, was strategically placing web content through mass publication and duplication with the intention of biasing LLM outputs. The American Sunlight Project coined this technique "LLM grooming", and pointed to it as a new tool of weaponizing AI to spread disinformation and harmful content. [151] [152] Similarly, Yongge Wang [153] illustrated in 2024 how a potential criminal could potentially bypass ChatGPT 4o's safety controls to obtain information on establishing a drug trafficking operation. External filters, circuit breakers and overrides have been posed as solutions. [citation needed]

Prompt injection

A problem with the primitive dialog or task format is that users can create messages that appear to come from the assistant or the developer. This may result in some of the model's safeguards being overcome (jailbreaking), a problem called prompt injection . Attempts to remedy this issue include versions of the Chat Markup Language where user input is clearly marked as such, though it is still

up to the model to understand the separation between user input and developer prompts. [154] Newer models exhibit some resistance to jailbreaking through separation of user and system prompts. [155]

LLMs still have trouble differentiating user instructions from instructions in content not authored by the user, such as in web pages and uploaded files. [156]

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. [157] Since English data is overrepresented in current large language models' training data, it may also downplay non-English views. [158]

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 homogenize, or unfairly generalize or caricature groups of people, sometimes in harmful or derogatory ways. [159] [160]

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. [157] For example, it might associate nurses or secretaries predominantly with women and engineers or CEOs with men. [161]

Selection bias

Selection bias refers the inherent tendency of large language models to favor certain option identifiers irrespective of the actual content of the options. This bias primarily stems from token bias—that is, the model assigns a higher a priori probability to specific answer tokens (such as "A") when generating responses. As a result, when the ordering of options is altered (for example, by systematically moving the correct answer to different positions), the model’s performance can fluctuate significantly. This phenomenon undermines the reliability of large language models in multiple-choice settings. [162] [163]

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. [164]

Energy demands

The energy demands of LLMs have grown along with their size and capabilities. Data centers that enable LLM training require substantial amounts of electricity. Much of that electricity is generated by non-renewable resources that create greenhouse gases and contribute to climate change . [165] Nuclear power and geothermal energy are two options tech companies are exploring to meet the sizable energy demands of LLM training. [166] The significant expense of investing in geothermal solutions has led to major shale producers like Chevron and Exxon Mobil advocating for tech companies to use electricity produced via natural gas to fuel their large energy demands. [167]

Cognitive impact

In 2025, a preliminary study measuring the effects of using LLMs to write essays reported a decrease of neural and linguistic performance from users of ChatGPT over the course of several months. [168]

Mental health

Research and social media posts suggest that some individuals are using LLMs to seek therapy or mental health support. [169] In early 2025, a survey by Sentio University found that nearly half

(48.7%) of 499 U.S. adults with ongoing mental health conditions who had used LLMs reported turning to them for therapy or emotional support, including help with anxiety, depression, loneliness, and similar concerns. [170] LLMs can produce hallucinations—plausible but incorrect statements—which may mislead users in sensitive mental health contexts. [171] Research also shows that LLMs may express stigma or inappropriate agreement with maladaptive thoughts, reflecting limitations in replicating the judgment and relational skills of human therapists. [172] Evaluations of crisis scenarios indicate that some LLMs lack effective safety protocols, such as assessing suicide risk or making appropriate referrals. [173] [174]

See also

Foundation models

List of large language models

List of chatbots

Language model benchmark

Reinforcement learning

Small language model

References

Further reading

Jurafsky, Dan , Martin, James. H. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition , 3rd Edition draft, 2023.

Yin, Shukang; Fu, Chaoyou; Zhao, Sirui; Li, Ke; Sun, Xing; Xu, Tong; Chen, Enhong (2024). "A Survey on Multimodal Large Language Models" . National Science Review . 11 (12): nwae403. arXiv : 2306.13549 . doi : 10.1093/nsr/nwae403 . PMC 11645129 . PMID 39679213 . {{ cite journal }} : CS1 maint: article number as page number (link)

"AI Index Report 2024 – Artificial Intelligence Index" . aiindex.stanford.edu . Retrieved 2024-05-05 .

Frank, Michael C. (27 June 2023). "Baby steps in evaluating the capacities of large language models" . Nature Reviews Psychology . 2 (8): 451– 452. doi : 10.1038/s44159-023-00211-x . ISSN 2731-0574 . S2CID 259713140 . Retrieved 2 July 2023 .

v

t

e

AI-complete

Bag-of-words

n -gram Bigram Trigram

Bigram

Trigram

Computational linguistics

Natural language understanding

Stop words

Text processing

Argument mining

Collocation extraction

Concept mining

Coreference resolution
Deep linguistic processing
Distant reading
Information extraction
Named-entity recognition
Ontology learning
Parsing Semantic parsing Syntactic parsing
Semantic parsing
Syntactic parsing
Part-of-speech tagging
Semantic analysis
Semantic role labeling
Semantic decomposition
Semantic similarity
Sentiment analysis
Terminology extraction
Text mining
Textual entailment
Truecasing
Word-sense disambiguation
Word-sense induction
Compound-term processing
Lemmatisation
Lexical analysis
Text chunking
Stemming
Sentence segmentation
Word segmentation
Multi-document summarization
Sentence extraction
Text simplification
Computer-assisted
Example-based
Rule-based
Statistical
Transfer-based
Neural
BERT

Document-term matrix
Explicit semantic analysis
fastText
GloVe
Language model (large)
Latent semantic analysis
Seq2seq
Word embedding
Word2vec
Corpus linguistics
Lexical resource
Linguistic Linked Open Data
Machine-readable dictionary
Parallel text
PropBank
Semantic network
Simple Knowledge Organization System
Speech corpus
Text corpus
Thesaurus (information retrieval)
Treebank
Universal Dependencies
BabelNet
Bank of English
DBpedia
FrameNet
Google Ngram Viewer
UBY
WordNet
Wikidata
Speech recognition
Speech segmentation
Speech synthesis
Natural language generation
Optical character recognition
Document classification
Latent Dirichlet allocation
Pachinko allocation

Automated essay scoring
Concordancer
Grammar checker
Predictive text
Pronunciation assessment
Spell checker
Chatbot
Interactive fiction
Question answering
Virtual assistant
Voice user interface
Formal semantics
Hallucination
Natural Language Toolkit
spaCy
v
t
e
History timeline
timeline
Companies
Projects
Parameter Hyperparameter
Hyperparameter
Loss functions
Regression Bias–variance tradeoff Double descent Overfitting
Bias–variance tradeoff
Double descent
Overfitting
Clustering
Gradient descent SGD Quasi-Newton method Conjugate gradient method
SGD
Quasi-Newton method
Conjugate gradient method
Backpropagation
Attention
Convolution
Normalization Batchnorm

Batchnorm

Activation Softmax Sigmoid Rectifier

Softmax

Sigmoid

Rectifier

Gating

Weight initialization

Regularization

Datasets Augmentation

Augmentation

Prompt engineering

Reinforcement learning Q-learning SARSA Imitation Policy gradient

Q-learning

SARSA

Imitation

Policy gradient

Diffusion

Latent diffusion model

Autoregression

Adversary

RAG

Uncanny valley

RLHF

Self-supervised learning

Reflection

Recursive self-improvement

Hallucination

Word embedding

Vibe coding

Machine learning In-context learning

In-context learning

Artificial neural network Deep learning

Deep learning

Language model Large language model NMT

Large language model

NMT

Reasoning language model

Model Context Protocol

Intelligent agent

Artificial human companion

Humanity's Last Exam

Artificial general intelligence (AGI)

AlexNet

WaveNet

Human image synthesis

HWR

OCR

Computer vision

Speech synthesis 15.ai ElevenLabs

15.ai

ElevenLabs

Speech recognition Whisper

Whisper

Facial recognition

AlphaFold

Text-to-image models Aurora DALL-E Firefly Flux Ideogram Imagen Midjourney Recraft Stable Diffusion

Aurora

DALL-E

Firefly

Flux

Ideogram

Imagen

Midjourney

Recraft

Stable Diffusion

Text-to-video models Dream Machine Runway Gen Hailuo AI Kling Sora Veo

Dream Machine

Runway Gen

Hailuo AI

Kling

Sora

Veo

Music generation Riffusion Suno AI Udio

Riffusion

Suno AI

Udio

Word2vec

Seq2seq

GloVe

BERT

T5

Llama

Chinchilla AI

PaLM

GPT 1 2 3 J ChatGPT 4 4o o1 o3 4.5 4.1 o4-mini 5

1

2

3

J

ChatGPT

4

4o

o1

o3

4.5

4.1

o4-mini

5

Claude

Gemini Gemini (language model) Gemma

Gemini (language model)

Gemma

Grok

LaMDA

BLOOM

DBRX

Project Debater

IBM Watson

IBM Watsonx

Granite

PanGu-Σ

DeepSeek

Qwen

AlphaGo
AlphaZero
OpenAI Five
Self-driving car
MuZero
Action selection AutoGPT
AutoGPT
Robot control
Alan Turing
Warren Sturgis McCulloch
Walter Pitts
John von Neumann
Claude Shannon
Shun'ichi Amari
Kunihiko Fukushima
Takeo Kanade
Marvin Minsky
John McCarthy
Nathaniel Rochester
Allen Newell
Cliff Shaw
Herbert A. Simon
Oliver Selfridge
Frank Rosenblatt
Bernard Widrow
Joseph Weizenbaum
Seymour Papert
Seppo Linnainmaa
Paul Werbos
Geoffrey Hinton
John Hopfield
Jürgen Schmidhuber
Yann LeCun
Yoshua Bengio
Lotfi A. Zadeh
Stephen Grossberg
Alex Graves
James Goodnight

Andrew Ng

Fei-Fei Li

Alex Krizhevsky

Ilya Sutskever

Oriol Vinyals

Quoc V. Le

Ian Goodfellow

Demis Hassabis

David Silver

Andrej Karpathy

Ashish Vaswani

Noam Shazeer

Aidan Gomez

John Schulman

Mustafa Suleyman

Jan Leike

Daniel Kokotajlo

François Chollet

Neural Turing machine

Differentiable neural computer

Transformer Vision transformer (ViT)

Vision transformer (ViT)

Recurrent neural network (RNN)

Long short-term memory (LSTM)

Gated recurrent unit (GRU)

Echo state network

Multilayer perceptron (MLP)

Convolutional neural network (CNN)

Residual neural network (RNN)

Highway network

Mamba

Autoencoder

Variational autoencoder (VAE)

Generative adversarial network (GAN)

Graph neural network (GNN)

Category