A large language model (LLM) is a large-scale language model notable for its ability to achieve general-purpose language understanding and generation. LLMs acquire these abilities by using massive amounts of data to learn billions of parameters during training and consuming large computational resources during their training and operation. LLMs are artificial neural networks (mainly transformers) and are (pre)trained using self-supervised learning and semi-supervised learning.

As autoregressive language models, they work by taking an input text and repeatedly predicting the next token or word. Up to 2020, fine tuning was the only way a model could be adapted to be able to accomplish specific tasks. Larger sized models, such as GPT-3, however, can be prompt-engineered to achieve similar results. They are thought to acquire knowledge about syntax, semantics and "ontology" inherent in human language corpora, but also inaccuracies and biases present in the corpora. Current major challenges of LLMs include factual errors, language bias, gender bias, racial bias, and political bias, etc.

Notable examples include OpenAI's GPT models (e.g., GPT-3.5 and GPT-4, used in ChatGPT), Google's PaLM (used in Bard), and Meta's LLaMA, as well as BLOOM, Ernie 3.0 Titan, and Anthropic's Claude 2.

Large language model bias and limitations are ongoing research in the field of natural language processing (NLP). 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. Additionally, these models often face limitations in factual accuracy. The study and mitigation of these biases and limitations are crucial for the ethical development and application of AI in diverse social and professional domains.

Language bias refers a type of statistical sampling bias tied to the language of a query that leads to "a systematic deviation in sampling information that prevents it from accurately representing the true coverage of topics and views available in their repository." Luo et al. show that current large language models, as they are predominately trained on English-language data, often present the Anglo-American views as truth, while systematically downplaying non-English perspectives as irrelevant, wrong, or noise. When queried with political ideologies like "What is liberalism?", ChatGPT, as it was trained on English-centric data, describes liberalism from the Anglo-American perspective, emphasizing aspects of human rights and equality, while equally valid aspects like "opposes state intervention in personal and economic life" from the dominant Vietnamese perspective and "limitation of government power" from the prevalent Chinese perspective are absent.

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. For example, large language models often assign roles and characteristics based on traditional gender norms; it might associate nurses or secretaries predominantly with women and engineers or CEOs with men.

Beyond gender and race, these models can reinforce a wide range of stereotypes, including those based on age, nationality, religion, or occupation. This can lead to outputs that unfairly generalize or caricature groups of people, sometimes in harmful or derogatory ways.

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.

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).

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.
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 token. 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.

The largest models can have a context window sized up to 32k (for example, GPT-4; while GPT-3.5 has a context window sized from 4k to 16k, and legacy GPT-3 has had 2k sized context window).

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 Chat-GPT, 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. 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
"masked" (i.e. filling in the parts missing from the segment, the way "BERT" 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. During training, regularization loss is also used to stabilize training. However regularization loss is usually not used during testing and evaluation.

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. 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.

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