Generative AI - a concise primer for non-experts

Claudius Gros[†], Daniel Gros[*]
[†]Institute for Theoretical Physics, Goethe University Frankfurt a.M., Germany
[*]Institute for European Policymaking at Bocconi,
Bocconi University, Milan, Italy

June 16, 2023

Abstract

A concise introduction into the workings of large language models is presented. We start with an introduction to the attention mechanism, the core of the transformer architecture, which is then followed by a discussion of the steps needed to engineer the base model, a generatively pretrained transfromers (GPT), to a working chatbot like ChatGPT.

How do large language models work?

Information processing vs. information routing. Before the advent of transformers, deep learning architectures were based on neuronal networks. A prominent example are convolution nets [1], which see widespread use for image classification. Neural networks process information using a non-linear transformation of their combined input.

Transformer are based in contrast on a mechanism denoted 'attention' [2], which was introduced in the form of 'self-attention' in the by-now-famous article from 2017 *"Attention is all you need"* [3]. With attention, information is routed, in addition to being processed. Focusing attention to certain parts of upstream layers means that only the information originating from these parts are processed further, but not the information streams arriving from other parts.

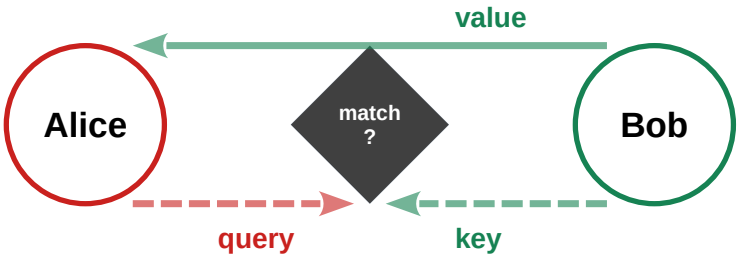

Figure 1: **The Query-Key-Value attention mechanism.** Tokens, viz words, are the constituent units of transformers, with individual layers containing thousands of tokens. Consider two tokens, say Alice and Bob, with the question being whether Bob has information that is relevant to Alice. In order to find out, Alice sends a query to Bob, which Bob compares with its key. If there is a match, Alice receives Bob's value object, viz an information package. Alice is said to 'pay attention' to Bob.

Transformer architectures are hence qualitatively different from the neural nets used in classical deep learning approaches. ChatGPT and most modern language processing software are based on various versions of the attention mechanism. Indeed, GPT stands for 'Generative Pretrained Transformer'.

Query, Key and Value. Every representation of a word, the token, generates three objects, denoted Query, Key and Value[1]. On a basic level, attention is akin to a database query. Consider two tokens, Alice and Bob, situated somewhere within the transformer architecture. First, Alice sends her query object to Bob, with Bob comparing the query received with its key. If a match is found, Bbb sends something back to Alice, namely his value object. The whole process is illustrated in Fig. 1.

Value objects correspond to the information associated with a token at a given time, which means that Bob's value object will be processed by Alice only when query and key match. This is the core of information routing within transformer architectures. During training, the parameters used to generate the Query, Value and Key objects are adapted.

It is not important which concrete implementation is used for the definition of the matching condition, only that the criterion used is never changed. The simplest version, the dot product between two vectors, does the job. The outcome of the query process is in practice a graded process, and not the all-or-nothing routing used here for illustrational purposes[2].

Base model. Transformer models are given texts to process, lots of text, with

[1]Mathematically, each token is associated with an activity vector and Q/K/V matrices. The vector resulting form the product of the respective matrix with the activity the token in question is denoted here 'object'.

[2]In reality, Alice receive Bob's value object together with a measure of how good the match between query and key has been.

the only task to predict the next word – at every single step. One speaks of 'self-supervised' learning. The different versions of GPT, namely GPT-1, GPT-2, GPT-3 and GPT-4, have increasing numbers of internal parameters, of the order of 10^8 / 10^9 / 10^{11} and 10^{12}. The majority of these parameters are used for the encoding of the Q/K/V objects, which are present in large numbers[3] In addition, there is a corresponding number of linear connections. During training, the content of the training texts is encoded implicitly, mostly within the Q/K/V objects.

What is a question? The base model, the generatively pretrained transformer, does not know about 'questions' and 'answers'. When presented with a prompt (the text input), it will merely try to complete the input word by word. For most applications, the output generated will have nothing to do with an informative answer.

As a first step, one therefore needs to teach the base model the notion of 'questions' and 'answers'. To do so, the model is presented $10^4 - 10^5$ handcrafted (by humans) high-quality examples of question-answer pairs.[4] Once trained with this comparatively small number of question-answer examples, the model is capable to produce meaningful reponses to a near infinite number of possible prompts.

Value alignment. Once the model is able to generate meaningful responses, quality testing is performed. At this step, denoted RLHF (reinforcement learning from human feedback), humans evaluate the quality of responses generated to a wide range of prompts, both with respect to the information content and with respect to a set of value requirements (like political correctness). Such a datbase of sored prompt-response pairs is generated. This database is used to train a second transformer, with the second transformer training subsequently the chatbot in the making. The outcome is a 'foundation model', viz a value-aligned chatbot like ChatGPT.

Learning from few examples. To be further useful, the foundation model can be supplemented with application modules. The additional circuits, called 'downstream tasks' in machine learning, can be suitably adapted classical neural networks. An alternative is Lora (low-rank adaptation) [4], which consists of adding a comparatively small number of additional parameters to the foundational model, which is otherwise kept frozen.[5] The key point is that application modules can be trained in general very fast, using only a few representative examples. This is possible because downstream tasks are designed to make use of the huge amount of causally structured information stored implicitly within the foundation model. One speaks of 'few-shot learning' [5].

[3] An order-or-magnitude estimate: Tokens may have $2^9 = 512$-dimenional actitivity vectors. The QKV and L matrices (L is an additional linear projection) then contain $4 \times (512 \times 512) \approx 10^5$ parameters. A single layer with about 10^4 tokens has therefore $10^9 = 10^4 \times 10^5$ parameters, a number which is increased to 10^{11} for a transformer with 100 layers.

[4] Like: Question: *"Give my a summary of this or that topic"*. Answer: *"An informatiive and well-written summary."*.

[5] A model is frozen if the parameters are not updated.

Prompt engineering. Using suitably structured prompts one can adapt the foundation model to a wide range of tasks without the need of additional application modules. This approach is called 'prompt engineering' [6].

An example for structured prommpting would be to present the foundation model several typical sentences together with suitable solutions or answers. Say that that we want the chatbot to generate Twitter hashtags and that there are three hashtags, #first, #second and #third, at disposal. Teaching via prompting works by presenting the system a few typical Twitter posts for each of the three hashtags. This should be enough for the model to be able to generate the appropriate hashtags for further posts.

AI psychology. A remarkable property of large language models is that they tend to produce better results when asked to list their reasoning, viz the intermediate steps. This approach can be taken a step further by asking the system to solve a certain problem, e.g. a logical reasoning problem, several times, listing each time the intermediate chain of thoughs [7]. Next one asks the chatbot to rank its own thinking, viz the several versions of intermediate reasonings produced during the invidual problem solving attempts. In general the system will select the best thought, improving performance in part drastically. This approach, denoted 'tree of thoughts' [7], can be regarded to represent a first step towards AI psychology. Motivating the system to reflect about itself tend to improve performance.

Acknowledgements

We thank Elena Porfidia from Bocconi University for research assistance.

References

[1] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv*, 2015.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[4] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry,

Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[6] What is Prompt Engineering?

[7] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*, 2023.