A Brief Explanation of Large Language Models

by Foster Provost* with Adam Brandenburger†

Version 09/18/23

1. Introduction

An AI system such as ChatGPT has several components, including one or more large language models (LLMs), a mechanism for managing dialog with users, and possibly other components, such as web search and other features that go beyond the "classic" ChatGPT.

2. Logistic Regression

At heart, an LLM works by having estimated the coefficients of a giant non-linear multinomial logistic regression from the corpus on which it is trained. In other words, it is an equation that starts with many free parameters, namely, the coefficients, and applies a machine learning procedure to fix those coefficients to useful values. By "giant", we mean that the equation contains many billions of coefficients — even over a trillion coefficients.

The resulting equation is then used to make a prediction. Given an input consisting of a sequence of words, the output is a probability distribution over the next word to follow the input.

3. Definition of Tokens

In reality, the objects used are not words but, rather, what are called tokens. The set of tokens is smaller than the set of all words (in English, say). In principle, the set of tokens could be as small as 26 characters—as in the number of letters in English—plus some additional symbols. But that would be too small a set to work well. The sets of tokens used in LLMs are much larger than this and comprise many words, pieces of words (like common syllables), letters, and punctuation, as well as useful components of other textual applications, such as computer programming.

Note that these tokens are exactly the values that the "features" — that is, the input variables, or what we think of as the x-variables — can take in the regression equation. The target or y-variable is the probability of each possible output token.

Of course, the corpus used for training need not only be text. For example, it could be GitHub code or a database of images. For simplicity, we will talk in terms of words in what follows, but we really mean tokens representing either text chunks or other objects.

4. Continuation Protocol

When the AI system (such as ChatGPT) is queried, it calls on the regression equation to output a probability distribution over all possible continuation words, given the input. It then selects a particular continuation word according to some protocol. For example, the protocol might first

---

* NYU Stern School of Business, NYU Center for Data Science. Provost thanks Ira Rennert for financial support and the NYU Stern Fubon Center for supporting improved understanding of AI and Data Analytics for business.

† NYU Stern School of Business, NYU Tandon School of Engineering, NYU Shanghai. Brandenburger thanks NYU Stern, NYU Shanghai, and J.P. Valles for financial support.

select the n continuation words with the highest estimated probability (for some integer n) and then sample a word from this subset, say, in accordance with these probabilities.

The second continuation word is generated by feeding the string consisting of the original input text together with the first continuation word as suffix back into the regression equation. The same process as before generates the second output word. This procedure iterates until a stopping point (for example, the process chooses a "finish" token as output) and the resulting output text is then fed by the AI system to the user. That resulting output text (a sequence of words, not just individual words) is what we would see as the output of ChatGPT for example.

5. Training Data

The data used to train an LLM — that is, to determine the regression coefficients — are obtained in different ways for different LLMs. As we understand it, for OpenAI's GPT LLMs, the training data were collected essentially by starting with the entire corpus of what has ever been written (and digitized), obtained largely by crawling the web. Then, certain subcorpora will have been more heavily weighted in training. For example, data from a more authoritative source such as Wikipedia would have been significantly overweighted.

6. Supervised Learning

LLMs are trained via what is called supervised machine learning. This refers to the procedure where the training data are labeled in such a way that a "correct" response to each training input is already encoded. (In a classic example, images of animals in the data are labeled as "cat" or "non-cat.") For training LLMs, an "instance" of an input is some contiguous sequence of text drawn from the training corpus we mentioned above. The label is then the actual next word — that is, the word following that sequence. This is why the word "correct" is in quotes above. The response is correct in the context of the source data, that is, in terms of what was already written in the training corpus. The actual word that follows a given text sequence is de facto correct. This backward-looking notion needs to be distinguished from the target task, which is to create a new chat response to some user input. What was written before does not necessarily represent the correct target response — more precisely, the next word in the correct target response. However, this "pretraining" on a massive volume of prior text turns out to provide remarkably good "proxy labeling" for training.

The "P" in ChatGPT stands precisely for "pretrained". The "G" stands for "generative," highlighting that the LLM is generating text. The "T" stands for "transformer," which is the particular neural network architecture used (more on this below).

In addition, LLMs can be "post-trained" to be better chatters than a pre-trained LLM. ChatGPT uses such an LLM. There are different methods of post-training. An example is when a tester (a person) generates multiple outputs to a given input and makes a selection of the best-sounding output.

To recap, for pre-training LLMs, the data are already labeled, because the "correct" label for an input appearing in the source data is precisely the next word in that source text. This is the key to why machine learning works so well for LLMs. Machine learning of massive models such as LLMs requires massive amounts of labeled data, which is naturally available for these LLMs.

7. Universal Approximation

We can ask why the particular computer architecture used to estimate the logistic regression in an LLM — namely, a neural network — should be able to perform this task. The fundamental answer comes from the Universal Approximation Theorem (which has various versions) of neural networks. This piece of math says that any function (within some very large family of

functions) can be approximated as closely as one wants via the input-output mapping of a suitably chosen neural network.

But these theorems also caution us to be aware of the danger of overfitting. Indeed, complexity control is a key issue in the design of all large neural networks. Too much complexity will allow the machine to fit the training data too well, which will then lead to poor performance on subsequent inputs. This is the overfitting problem. Of course, too little complexity will be suboptimal, too. One of the best ways to limit overfitting is to have access to massive amounts of labeled training data, just as we said is the case for LLMs.