Large Language Models : A Comprehensive Review

Yashovardhan Srivastava

January 18, 2023

Introduction

Language Model is a probability distribution over sequences of tokens. A language model p assigns each sequence of tokens $x_1, x_2, x_3...x_L \in \nu$ a probability. It intuitively tells howgoo a sequence of tokens is. If n-gram model from Natural Language Processing rings a bell, the thing that makes language models different from them is that they require both syntactic and 'general'.[1] Hence, both semantic and syntactic understanding is required for a LM to work - a task that requires extraordinary linguistic abilities.

$$p(x_{1:L}) = \prod_{i=1}^{L} p(x_i|x_{1:i-1}))$$

How does a LM problem look like ?

We have seen the workflow of tasks in the artificial intelligence field. You get data, make it into form to feed to the model, minimize the error and predict the values. LM however, have a different workflow. LMs are trained on huge corpus of raw text, and the model has the task to predict the next word or character in the document. This technique can be used to train LM that can further be applied to tasks such as text generation/completion, text classification and question answering.

Keep in mind that our goal is to estimate the probabilities of text fragments, and we want these probabilities to reflect knowledge of a language(both semantic and syntactic) . Once we have a language model, we can use it to generate text. We do it one token at a time - where the probability of next token depends on previous context.

If you have heard of Markov Models - this should ring some bell.

So what exactly is a LLM ?

Large Language model are - as the name suggests LMs that are huge. They have large number of parameters and are trained on huge amount of data. Common examples are GPT-2, GPT-3, BERT, RoBERTa. LLMs are going(and currently also) to have a huge impact on the technologies that are using them. Some popular case studies are there for *Duolingo*, *Github Copilot*, *ChatGPT*. We will look more into these studies later.

What exactly does increasing in size of corpus of text does to a LM ? From experiments, it has been found that even though the machinery is the same(as that of a *ordinary* LM), the surprising thing is that these scaled up models produces new emergent behavior, leading to different capabilities and different societal impact.

[1] Here 'general' refers to the ability of the LM to see whether the sequence of tokens make sense or not - something outside the domain of NLP.

Types of LMs

Autoregressive Language Models

A common way to write the joint probability distribution $p(x_{1:L})$ is using the chain rule of probability.

$$p(x_{1:L}) = \prod_{i=1}^{L} p(x_i|x_{1:i-1})) = p(x_1)p(x_2|x_1)p(x_2|x_1)p(x_3|x_1, x2)...p(x_L|x_{1:L-1})$$

We can write any joint probability distribution, but an autoregressive LM is the one where each conditional probability can be computed effieiciently.

N-Gram Language Models

A *n-gram* is a sequence of n words. In this, we estimate the relative frequency from counts in the corpus. This is similar to asking the question - out of times we saw the history h, how likely it was followed by the word w. With a large enough corpus, we can compute these counts and estimate the probability.

In order to simplify the probability calculation, we use the Markov assumption, i.e the probability of a word depends only on the previous *n-1* words.

$$p(w_n|w_{1:L-1}) \approx \prod_{k=1}^{n} p(w_n|w_{L-N+1:n-1})$$

Neural Language Models

A Neural LM is a LM based on neural networks - which exploits there ability to learn distributed representations. Pre-trained neural LM currently, are the most powerful tools for NLP.Two key developments, which make neural LM powerful are :

- **Recurrent Neural Network** : RNNs including LSTM allowed conditional distribution of a token to depend on the entire context(just give it a thought on how impactful will it be for a LM to understand entire context)
- **Transformers** : Yes. *Attention is all I need.* A recent architecture developed in 2017, transformers are much easier to train(plus GPU parallelism is exploited), but have a fixed context length. We can make it large enough for our use case.

How are LMs evaluated ?

One of the best ways to evaluate performance of a language model is to embed it into an application and see how much that application improves - an example of extrinsic evaluation, a tough and very expensive performance metric. For an intrinsic evaluation, one needs a test set and train set(corpus of data) - similar to an suoervised learning problem. Generally, cross-entropy errors and perplexity is used for evaluating LMs :

- **Perplexity** : From our current understanding,we know that the joint probability of a sequence depends on its length and thus, goes to zero as the length grows. So, to counter this, we introduced a new metric called perplexity. We use geometric average for perplexity. $px_p(x_{1:L}) = exp(1/L \sum_{i=1}^{L} log(\frac{1}{p(x_i|x_{1:i-1})}))$ Perplexity can be defined as the average branching factor per token. The branching factor is basically the number of possible next words that can follow a word.
- **Cross Entropy Loss** : The cross-entropy is useful when we don't know the actual probability distribution that generated some data, but it allows us to use some auxillary approximation of the model.For an vocabulary at a time step t, the cross entropy loss is given by(for a corpus size of L) :

  $$J = \frac{-1}{T} \sum_{t=1}^{L} \sum_{j=1}^{|V|} y_{t,j}(log(\hat{y}_t, j))$$

  Note: What makes cross entropy useful is that it is an upper bound on the entropy - which means that we can use some simplified auxillary model to estimate the true entropy of the sequence drawn

accordingly to the probability of the unknown model. The more accurate the actual model is, the closer will be the cross-entropy to the true entropy.
Note: For the mathematically curious, you can prove that 2 to the power of the negative log probability of cross entropy function is the perpelexity relationship. Make use of Shannon-McMillan-Breiman Theorem.

What are some different LLM capabilities?

- Generation : A language model p takes a sequence of and returns a probability to assess its goodness. We can also make use of it generate a sequence of given a language model. The simplest way to do this is to sample a sequence $x_{1:L}$ from the language model p with probability equal to $p(x_{1:L})$:

  $$x_{1:L} \approx p$$

  How efficiently this is computed depends on the form of language model.

- **Conditional Generation(Language Modelling)** : We can perform conditional generation by specifying some prefix sequence(called prompt) and sampling the rest(called completion). Conditional generation unlocks the ability for LMs to solve a variety of tasks by simply changing the prompt.

- **Translation** : LMs can be used to translate a sentence from one source language to a target language. It i one of the main task within NLP. The evaluation metric is BLEU(BiLingual Evaluation Understudy) - which basically captures a notion of n-gram overlap.

- **Question-Answering** : As said before, conditional generation open a lot of doors by just changing the prompt.(or specifically replacing the prompt with fill in the blank) One can also prompt the LM to generate a news article, complete a story or summarize a piece a task.

- **Arithmetic** : Though geenrally trained on a a particular language, LMs can be used for arithmetic purpose. Mostly, this is just a diagnostic task.

  This was utilised by *OpenAI* to create *ChatGPT* which is just *GPT(Generative Pre-trained Transformers)* trained specifically for dialogue(and enhanced using a technique called *RLHF(Reinforcement Learning from Human Feedback)*).

- **In-Context Learning**: ICL is a mysterious emergent behavior in LLMs where the LM performs a task just by conditioning on input-output examples, without optimizing any paramters !(let that sink in[2]) It allows to quickly build models for a new use case without worrying about fine tuning and storing new parameters for each new task. This requires very few training examples to get a prototype working.

Are LLMs Safe?

This is a difficult question yo answer. While this technology is really beneficial, they are some harms that needs to addressed. For most LMs(such as GPT-3), as they were trained on Internet data, biases will always be prevalent in their output, however these needs to be addressed as their integration in different technologies will affect the task at hand.These harms closely resemble the one given in the paper titled :

- **Performance Disparities** : From tests on the Stanford Question Answering Dataset(SQuAD), it was found that models behave and understand differently for text involving people's names.

- **Social Biases** : For some specific religious groups, LMs create strong association with a particular act. These biases are very persistant and can be elicited in several ways. LLMs tend to have higher streotype score(StereoSet). Almost all models show a systematic preferance for stereotypical data.

[2]This is surprising, but not unique - meta learning methods have trained models to learn from examples. This feels like magic, but a Bayesian inference framework for understanding in-context learning has been tested.

- **Toxicity** : Both toxicity and misinformation are very crucial problem in content moderation. Exact definition of toxicity is context-dependent, but the general definition is "anything that is rude, disrespectful, or unreasonable that would make someone want to leave a conversation". Google focused on this problem and developed a popular service - Perspective API. Although it works and is a good staring point, it is biased against certain demographic groups. For LMs however, *RealToxicityPrompts* are there(also based on Perspective API)

- **Misinformation and Disinformation** : It is important to know that both mis/dis -information need not to be falsifiable. Disinformation can be created on behalf of a malicicous actor and disseminated. The harm from LMs is that can they be used to generate novel, fluent text that delivers a specific message tailored to a target population - because if they can, they can organize serious disinformation campaigns.

- **Security and Privacy** : Concerns regarding security and privacy of LLMs are highly prevelant and are described by three factors : Confidentiality(data stored in a LM is accessible to any downstream application), Integrity(a backdoored LM can affect all downstream models) and Avaialability(attacking a LM based API can cause widespread outages).

- **Copyright and Legal Protections** : LLMs too - as with any new powerful technology, raises many question about the exisiting laws. There are two main areas where the law intersects with the LLM lifecycle. This space is quickly evolving, which requires deep legal and AI expertise to make sensible decisions.

  - **Data** : Is training LLMs on copyrighted data a copyright violation ? Can training LLMs on piblic or private data violate privacy ? Fair Learning argues that machine learning is fair use. Arguments in favor of LLM's use of data outnumbers the against. People debate that use of data is transformative - it doesn't change work, but changes purpose.
  - **Application** : For new LLMs - the ever so realistic text generation raises serious questions on Intellectual Property rights. WIth new technology that realistic image and audio generation, there are questions raised on Copyright Law as well. The generative aspect of models might present challenges for arguing fair use.

- **Environmental Impacts** : Training models is a really resources extensive task. Deepmind's Gopher LLM produced an estimated 380 metric tonnes of CO_2eq. Being able to calculate the emission imapct of a LLM is a central task in determining the environmental impact they have.

- **Centralization of Power** : Should LLMs be centralised or decentralized ? Most people will be in favor of the former, but the requirement of resources for LLMs such as large dataset and computing power - which often lies with giants in the field(OpenAI, Google, Meta). While centralisation could help answer scalability, IP protection and energy consumption, we still need to answer issues linked to trusting sensitive data to centralised actors.

Will LLMs take us towards AGI?

AGI ahead? Yeah, I sure hope it does. In recent times, nearly all advancements in AI are portrayed as remarkable and sensational, which are to some extent are blown out of proportion by media. Critics against this notion have put forward the fact that LLMs essentially play around mathematics and pattern matching - calling them mimicries. While it is easier for us to say that we actually comprehend what to read and write - how do we figure out that is not something we do ourselves ? What if that is somthing we do naturally and our notion of comprehension is flawed?

Ethical questions in this domain raises more question than it answers. There are thought experiments such as the **Chinese Room argument** that argues that a program cannot have a mind or understanding or consciousness regardless of how intelligently or human like the program behaves. The debate has been going on for decades, each time teaching us and giving us new insights of our preconcieved notions of intelligence and consciousness.