* Self-Attention and Transformers
* Pretraining
* Foundation Model
* Natural Language Generation

### Motivation

Language modeling is usually framed as unsupervised distribution estimation from a set of examples (x1, x2, ..., xn) each composed of variable length sequences of symbols

(s1, s2, ..., sn). Learning to perform a single task can be expressed in a probabilistic framework as estimating a conditional distribution p(output|input). Since a general system should be able to perform many different tasks, even for the same input, it should condition not only on the input but also on the task to be performed. That is, it should model p(output|input, task).

### Training Data

Web can be generally unintelligible, hence how can that we used for training. Reddit hyperlink is used for this purpose. Any hyperlink with 3 outward + sign is used as proxy for good link.

Input Representation

1. UTF-8 byte level representation doesn’t come to picture as they are not competitive with word level
2. include pre-processing steps such as lowercasing, tokenization, and out-of-vocabulary tokens which restrict the space
3. Byte pair encoding : frequent words as a single tokens, while less frequent words are represented by multiple tokens, each of them representing a word part.[7] For example, the word "transformers" would be represented by two tokens, one encoding a frequent word "transform" as its first subword and the other encoding the "ers" as another frequent subword. BPE brings the perfect balance between character- and word-level hybrid representations which makes it capable of managing large corpora. This behavior also enables the encoding of any rare words in the vocabulary with appropriate subword tokens without introducing any “unknown” tokens. This especially applies to foreign languages like German where the presence of many compound words can make it hard to learn a rich vocabulary otherwise. With this tokenization algorithm, every word can now overcome their fear of being forgotten (athazagoraphobia)
4. Unigram tokenization:

### Architecture

Transformer long sentensteces

In brief, the five common LLM challenges include:

1. Understanding model limitations
   1. Model Bias
   2. Build
2. Choosing your model’s endpoint
3. Finetuning the model to your task
4. Choosing the right set of parameters
5. Designing prompt for the model
6. Training and deployment

<https://txt.cohere.com/best-practices-for-deploying-language-models/>

<https://txt.cohere.com/5-challenges-of-working-with-large-language-models-and-how-to-cope-with-them/>

<https://txt.cohere.com/how-to-train-your-pet-llm-prompt-engineering/>

## RLHF

Writing a loss function to capture these attributes seems intractable and most language models are still trained with a simple next token prediction loss (e.g. cross entropy). Reinforcement Learning from Human Feedback (RLHF) can help/