WHAT IS LLM

SR LLM - GROUP 1

What is an LLM?

- LLM stands for Large Language Model. LLM's are a specific type of generative AI models that are able to process, understand and generate human-like texts.
- LLM's are generally trained on books, articles, wikipedia pages etc. and that means petabytes of data.
- Unlike the old language models, LLM's are capable of understanding the context by associating words with each other. Therefore, they are able to process correctly long paragraphs and even books.
- LLM applications are used in different fields such as consumer services, content creation and linguistics.

LLM’s VS Other Types Of ML Models

LLM’s have unique features that distinguish them from other types of machine learning models:

1- Unlike the recurrent neural networks, they are based on transformer architecture. Transformers are capable of understanding the context by associating the related words with each other in a sentence or a paragraph. By using transformer architecture, these models can process larger texts more accurately than traditional models.

2- LLM’s are developed in 2 phases: Pre-training and fine-tuning. In pre-training phase, LLM’s are trained on very large data sets in an unsupervised manner to learn general language rules. After pre-training, they are trained on more specific data to be able to do their special tasks such as summarizing or storytelling etc.

LLM’s VS Other Types Of ML Models

3- Unlike other traditional models, LLM’s are able to process more complex language patterns and they are more creative because of their millions of parameters.

4- Because LLM’s are trained based on vast amount of unlabeled data in an unsupervised manner, sometimes they may create misleading information and answer biased, which leads to ethical concerns. However, traditional models generally trained on labeled data, which reduces the risk of misleading outcomes.

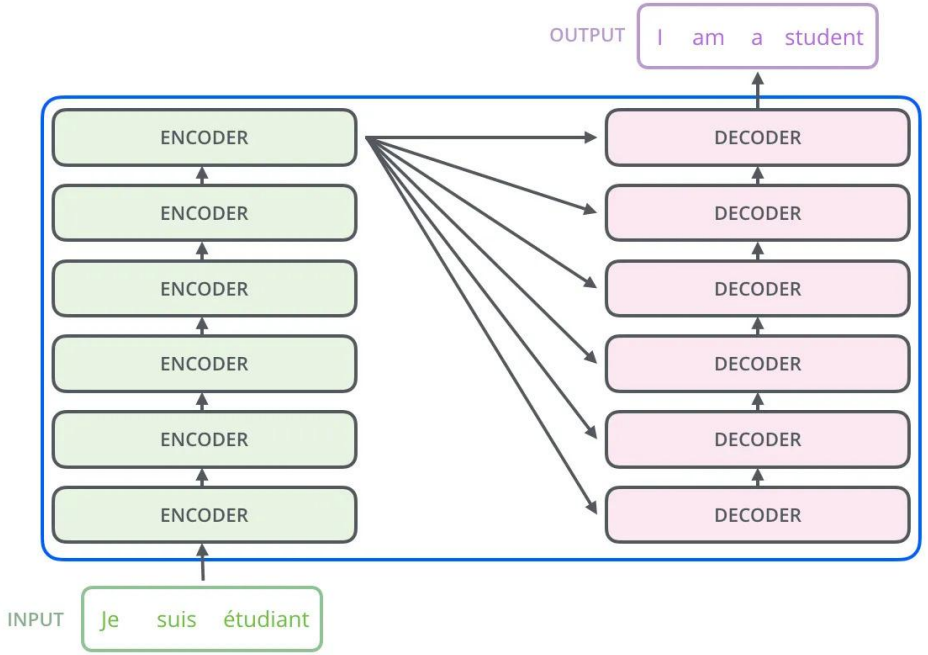
TRANSFORMERS

- Traditional models mostly process data sequentially, this becomes quite ineffective when elements are distant from each other.
- Also if the chain is longer, more information and data get lost in the process.
- Attention mechanisms were introduced with transformers and helped these issues.

- Transformers use attention mechanisms with encoders and decoders.

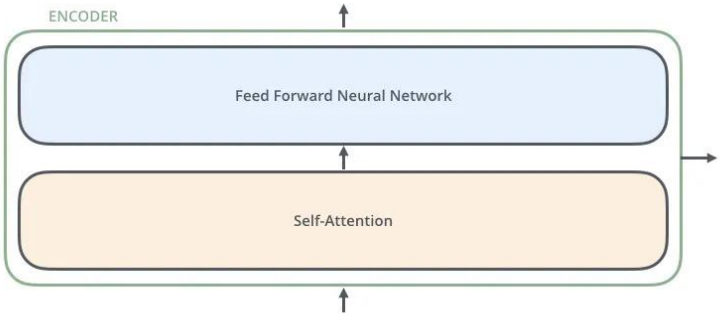
- Since transformers don't inherently understand sequence order, they use positional encodings added to the input embeddings to provide some notion of order.

- For each word in the input, self-attention calculates a weight for every other word in the sequence, effectively letting the model decide which words are more relevant to the current word being processed.



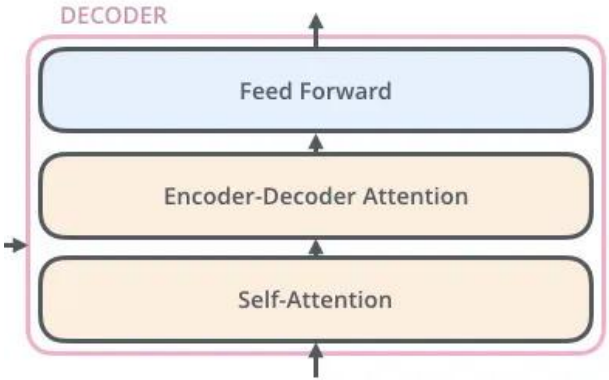
Encoders

- Takes input and turns them into embeddings



Decoders

- Takes the encoded information and generate the output



TOKENIZATION

- Tokenization is the process of converting a text into a sequence of tokens, which can be words, subwords, or characters.
- These tokens are the smallest units of meaning in a text that can be processed by a language model.
- For example, the sentence “Hello, world!” can be tokenized into [“Hello”, “,”, “world”, “!”].

- After tokenization, every token is assigned a unique value.
- For every token, there is a corresponding row in a lookup table that provides its vector representation.
- Since machine learning models operate on numbers, it is necessary to represent tokens as integers.
- Tokenization can be done at various levels, such as character-level and word-level. However, these methods often lead to very large lookup tables and lengthy sequences, making them less efficient. So BPE

BYTE PAIR ENCODING (BPE)

- Reduces the size of vocabulary
- Handles rare words better by splitting them into more frequent subwords

-> How does it work?

1. Treat each character as an individual token.
2. Find the most frequent pair of tokens and merge them into a single token.
3. Continue merging the most frequent pairs until a predefined vocabulary size is reached or the process stabilizes.

- **Example**:
  - Initial Tokens: `` `["l", "o", "w", "e", "r", "l", "o", "w"]` ``
  - Most Frequent Pair: `` `"l" + "o"` `` → `` `"lo"` ``
  - Merged Tokens: `` `["lo", "w", "e", "r", "lo", "w"]` ``
  - Repeat Merging: `` `"lo" + "w"` `` → `` `"low"` ``
  - Final Tokens: `` `["low", "e", "r", "low"]` ``

EMBEDDINGS

- Embeddings are vectors stored in an index within a vector database.
- Instead of treating words as unique symbols, embeddings capture the relationships and similarities between words based on their meaning and usage.
- Each word is mapped to a point in a multi-dimensional space. Words with similar meanings are close together in this space.
- Embeddings are learned during training so that the model can capture the relationships between words

Why Use Embeddings?

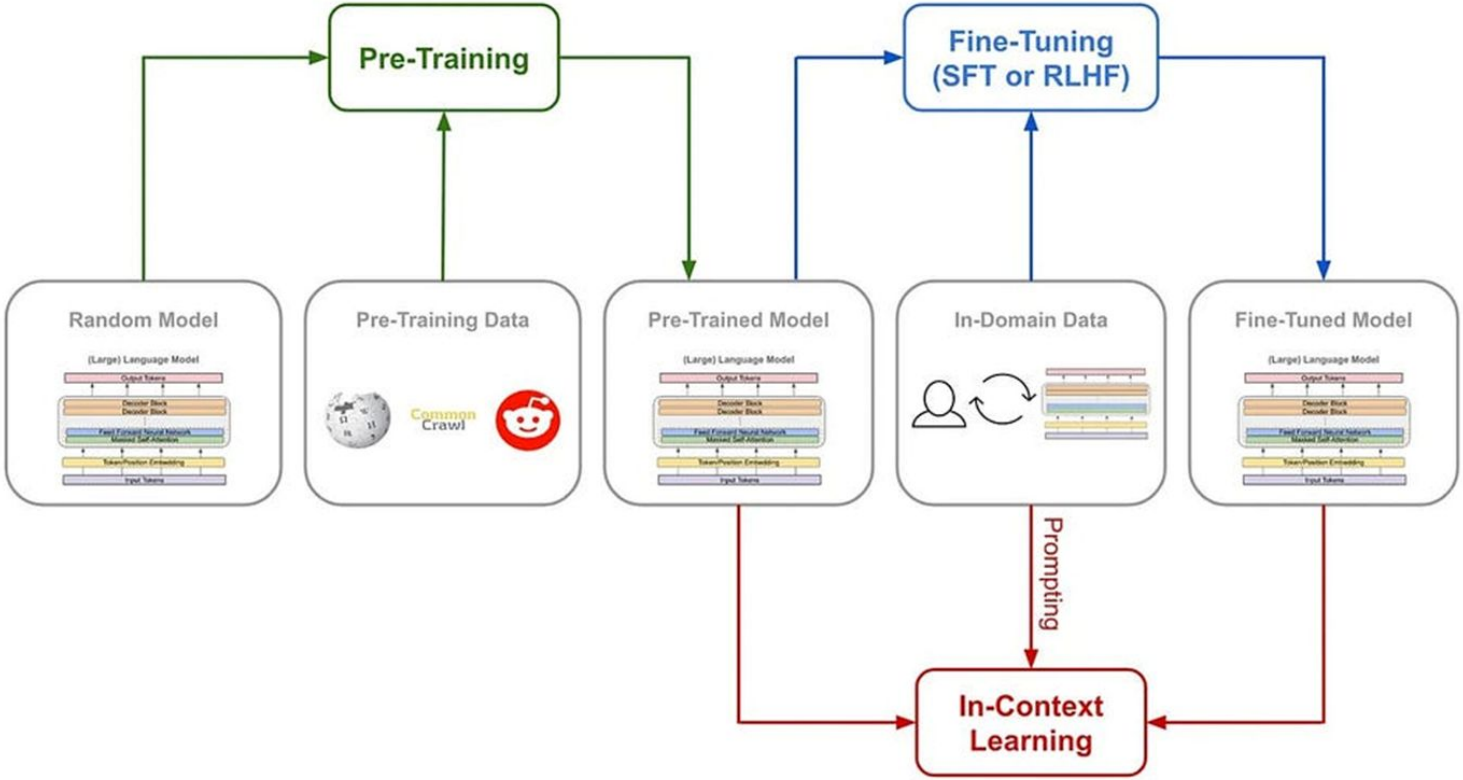
1. **Efficient:** They allow words with similar meanings to have similar vector representations, making it easier for models to generalize and understand the context.
2. **Compact:** Instead of huge one-hot vectors (where each word is a large, sparse vector), embeddings represent words in a much smaller, dense vector space.

Pre-Training

- Involves training the model on a large, diverse datasets.
- Focuses on acquiring general language knowledge and understanding.
- Enables the model to learn grammar, context, and factual information in an unsupervised manner.

Fine-Tuning

- Refines the model for specific tasks using smaller and labeled datasets.
- Adapts the general knowledge to perform well on task-specific objectives.
- Enhances the model’s ability to specialize in various areas, such as translation, summarization, etc.


Pre-Training
Fine-Tuning
(SFT or RLHF)
Random Model
(Large) Language Model
Pre-Training Data
Common
Crawl
Pre-Trained Model
(Large) Language Model
In-Domain Data
Fine-Tuned Model
(Large) Language Model
Prompting
In-Context
Learning

Evaluating LLMs

Evaluating large language models is crucial for understanding their performance. Some of the key metrics used to measure the performance of LLMs are:

- Perplexity
- Accuracy
- Human evaluation
- BLEU (Bilingual Evaluation Understudy Score)
- ROUGE (Recall-Oriented Understudy for Gisting Evaluation)

Real-World Applications of LLMs

- Chatbots (ChatGPT, Gemini)
- Virtual Assistants (Alexa, Siri)
- Content Generation (Jasper AI)
- Translation (DeepL)
- Text Summarization (SummarizeBot)
- Code Generation (GitHub Copilot)
- Education (Duolingo, Grammarly)

THANK YOU FOR LISTENING!