LLM - Week 1

SÜLEYMAN EMİR TAŞAN

Agenda

- What is an LLM?
- Transformer architecture
- Pre-Training
- Fine-Tuning
- Tokenization
- Embedding
- Evaluation of a LLM
- Real-world Applications

What is LLM?

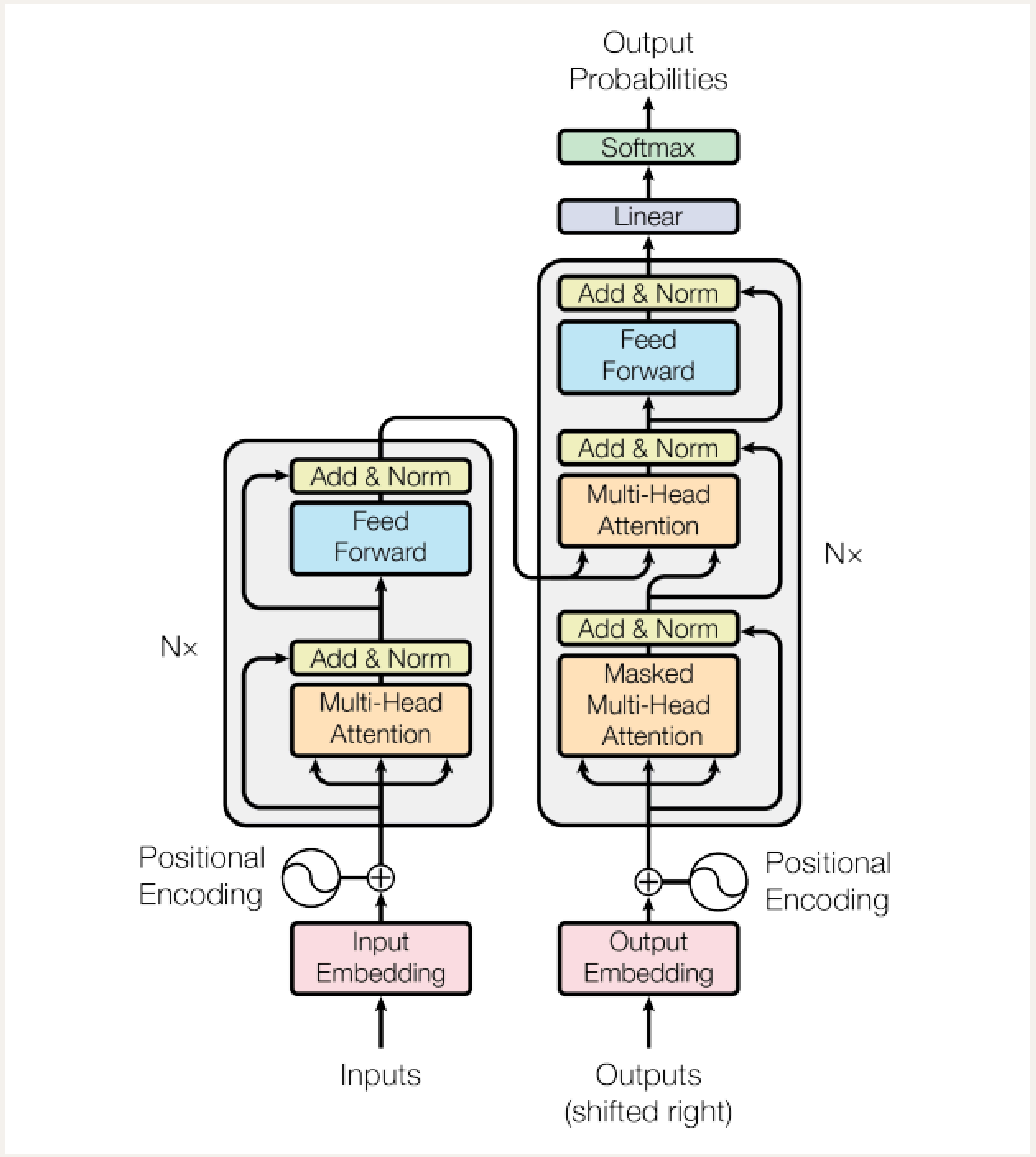
- LLM is an artificial intelligence model used in the NLP field, trained on very large amounts of text data.
- LLMs use a transformer structure and have billions (or even trillions) of parameters.
- In fact, we can say that LLMs are neural networks that produce output in human language.

Difference between traditional ML and NLP Models

- LLMs are transformers, while classical models are based on simple mathematical structures.
- LLMs have many parameters, but classical models have fewer.
- LLMs are generally general purpose, while classical models are task specific.
- LLMs can be adapted to different tasks, but classical models need to be retrained.
- LLMs can capture much more detailed contexts.

Transformer Architecture

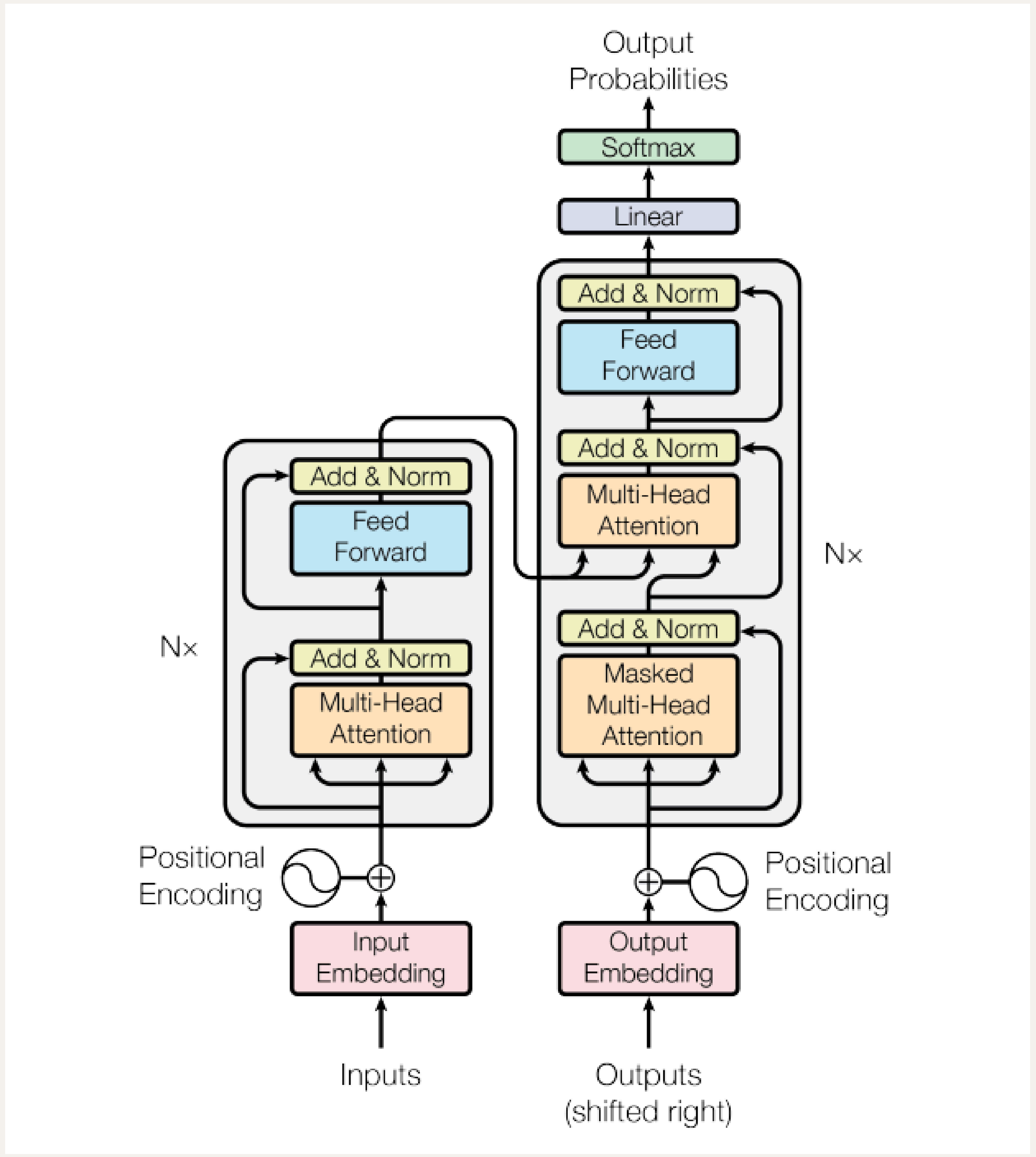
- Transformers basically consist of two structures called encoder and decoder. While encoders process the input text and convert it into meaningful vectors for the model, decoders produce results using these vectors. In this respect, the output of the encoder is an input for the decoder.
- Both consist of 6 identical layers. In these layers (as seen on the right), there are 2 sublayers in encoders and 3 sublayers in decoders.
- They form the building block of LLMs.



Transformer Architecture - Self Attention

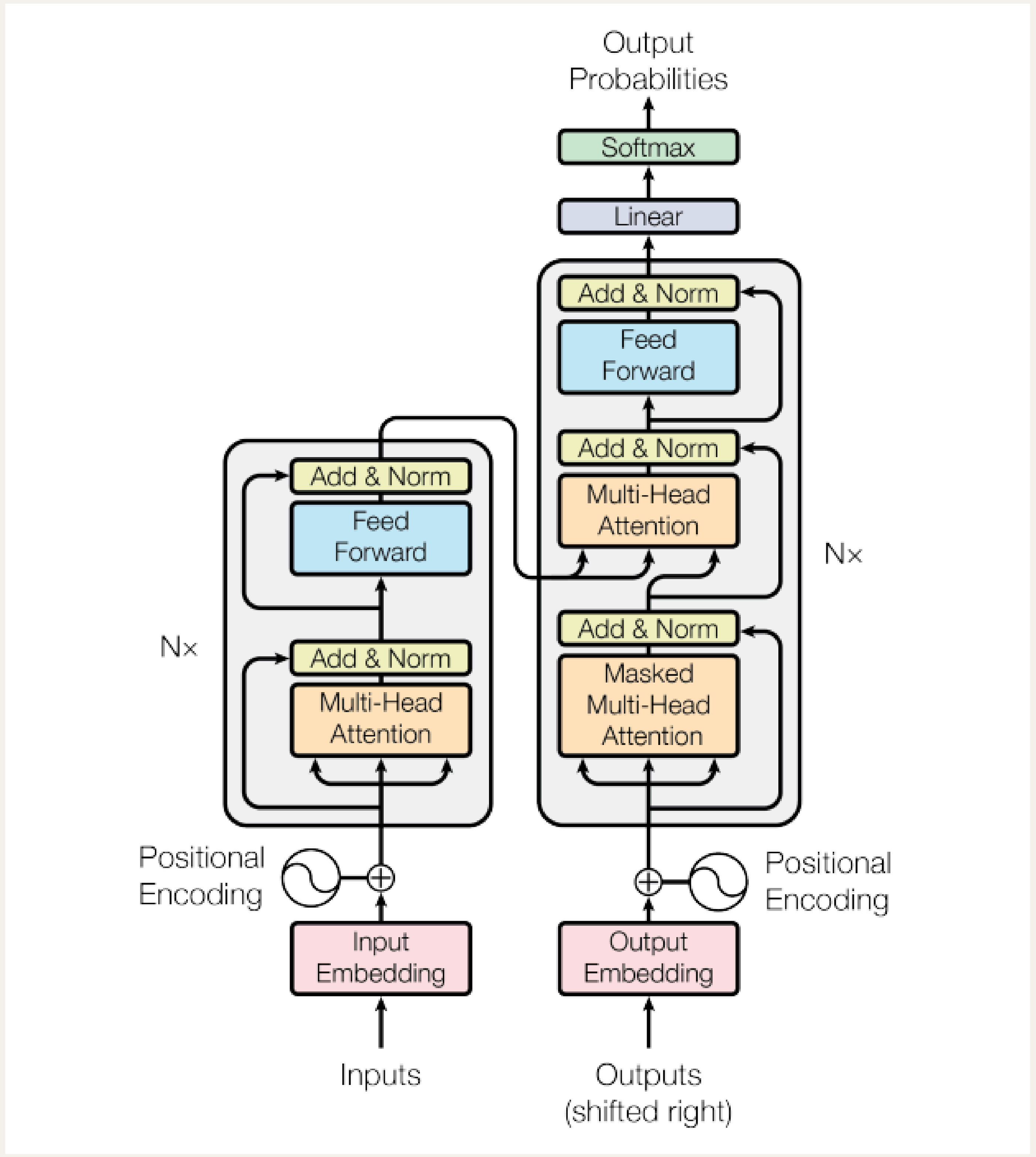
- It tries to understand the context by calculating the relationship of a word to other words in the sentence.
  "Kedi koltukta uzanıyor."

- "Kedi mavi renkli bir koltukta dört ayağını altına almış bir şekilde uzanıyor."
- The difference between these two sentences is that traditional RNN and LSTM models lose context as the distance between words increases, but in Self-Attention, since we look at all the words, this context is not lost.



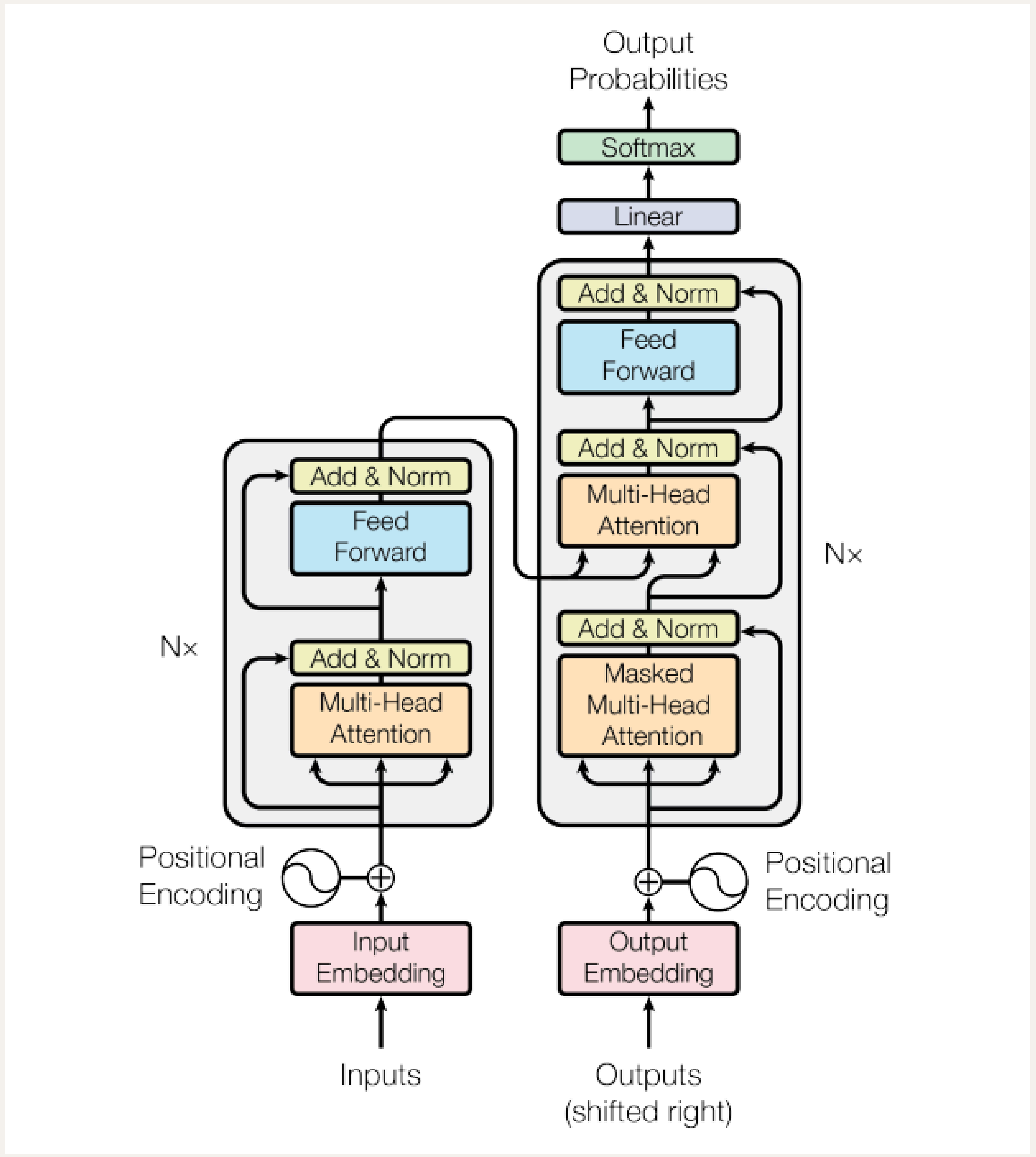
Transformer Architecture - Self Attention

- It is the most important mechanism for Transformers.
- 3 vectors are created for each word.
    - Query: What is the word looking for?
    - Key: The probability of the word matching with other words
    - Value: The actual meaning of the word
- Attention scores are calculated with various multiplications of these 3 vectors.
- Relationships between words are determined with these scores.
- It can work very fast because it can be parallelized.
- It understands contexts better.



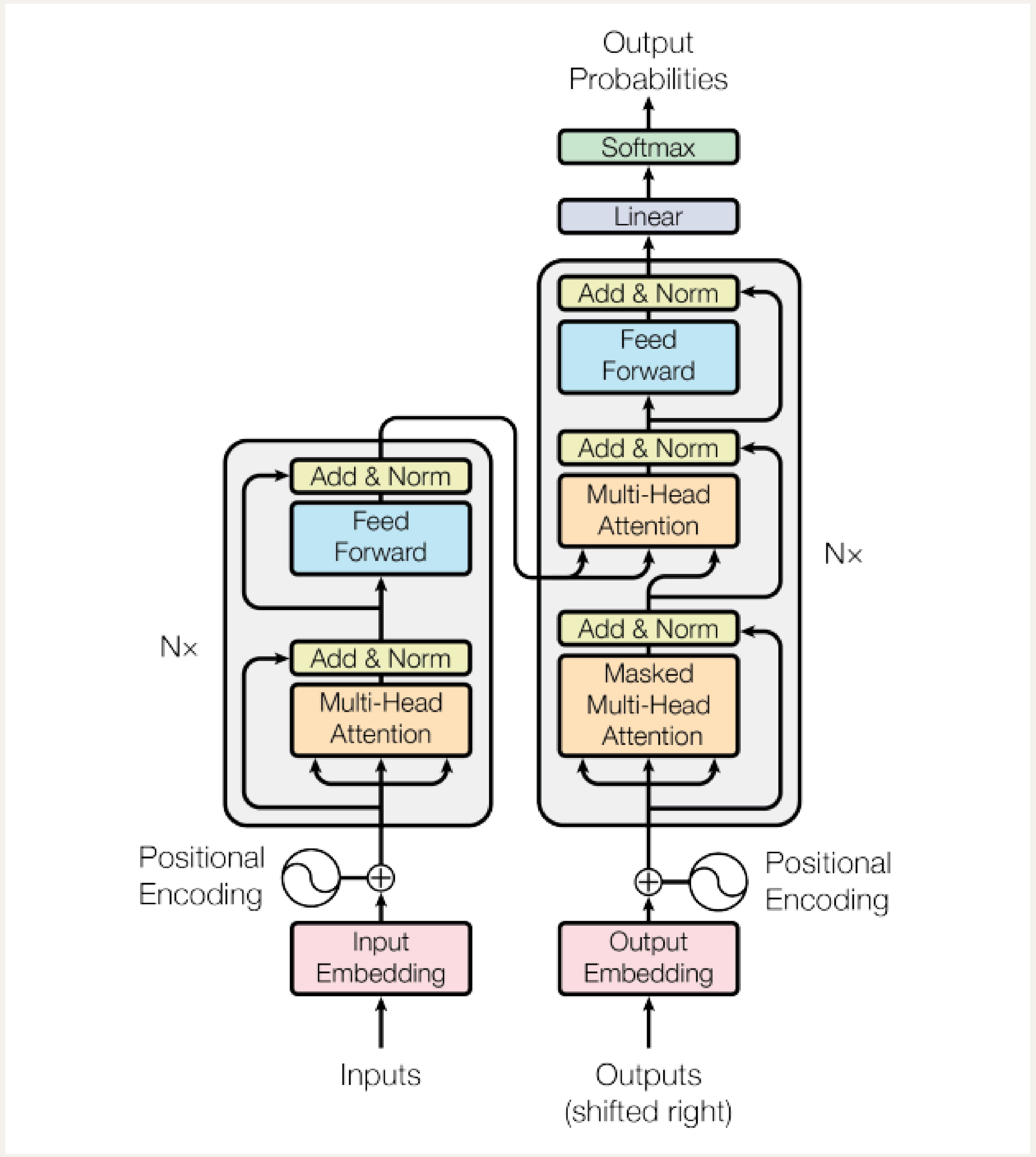
Transformer Architecture - Multihead-Attention

- More than one attention mechanism works at the same time.
- Each one looks at a different kind of relationship.
- In this way, different kinds of relationships are revealed at the same time.
    - One head looks at the time relationship
    - Another head looks at the subject-object relationship.



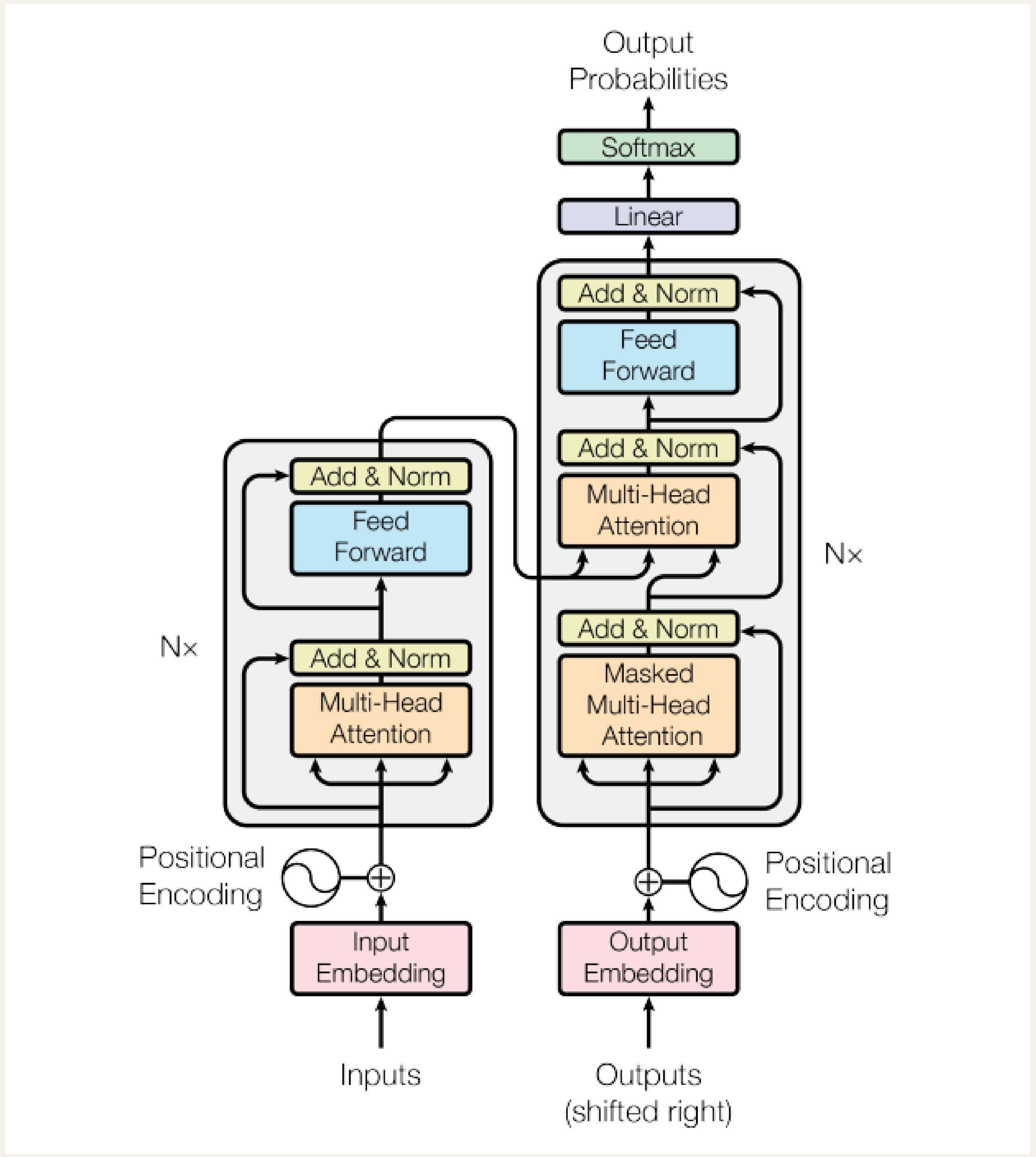
Transformer Architecture - Feed Forward

- After each attention layer, there is a fully connected feedforward network.
- It processes and transforms what comes from self attention. In this way, it captures the features that attention cannot capture.
- ReLU is used as activation.



Transformer Architecture - Positional Encoding

- Since Transformer performs parallel processing, it does not process the words in order, but the order of the words is also important in their relationship with each other. Therefore, positional encoding mechanism is used in order not to lose the order.

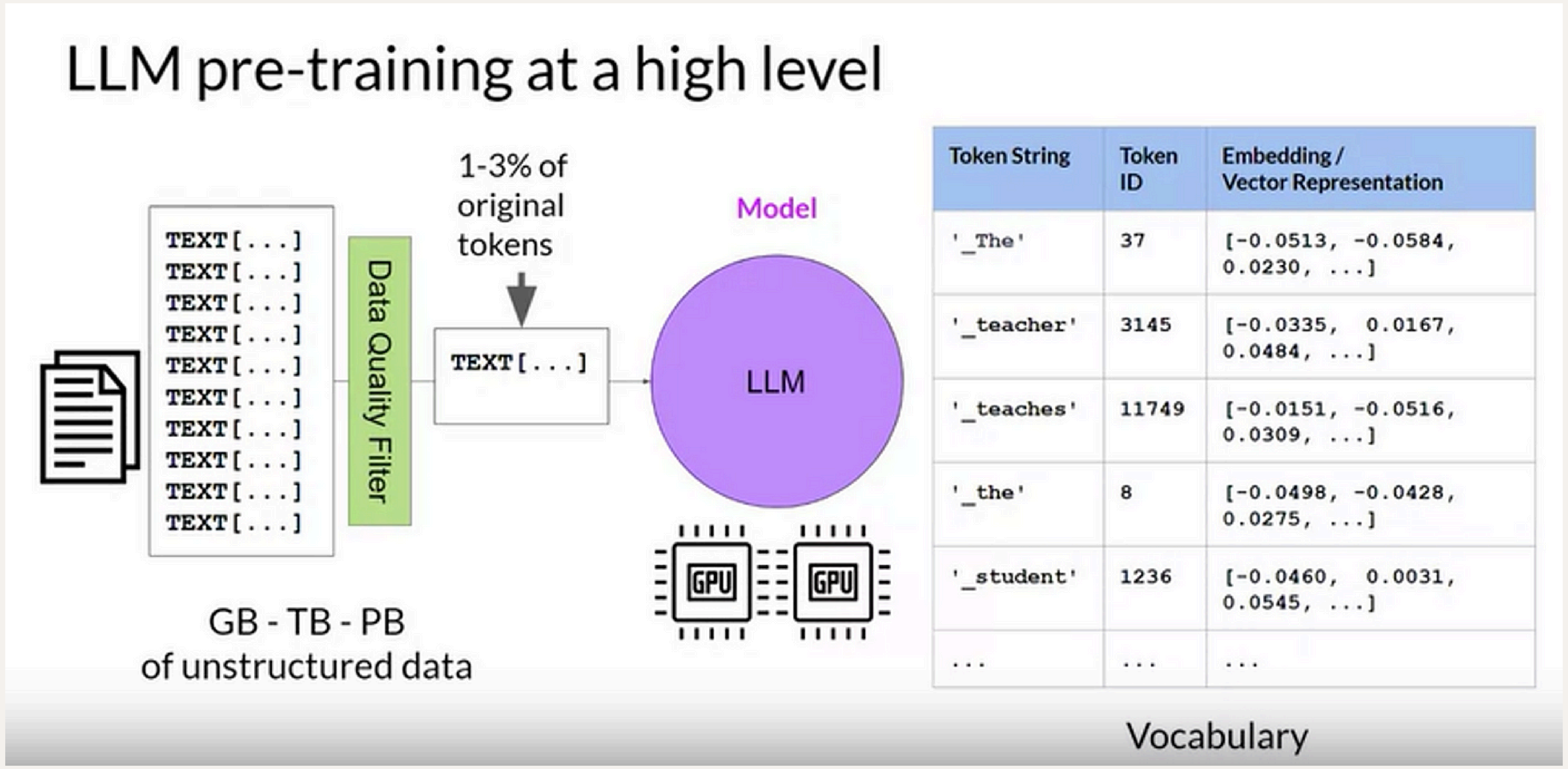


Transformer is Crucial for NLP

- Faster and better results (Parallel processing).
- Understanding of longer contextual sentences is ensured.
- The same model can perform multiple tasks. (foundation models)
- It has become an industry standard.
- The reason for the difference between LLM and classical models is the transformer. These differences are enough to explain the revolution.

Pre-Training

- It is the first stage for the model to gain general knowledge. In other words, it is the training stage before the model is developed for a specific task.
- Purpose:
    - It allows the learner to learn word meanings and contexts,
    - To understand sentence structure,
    - To internalize language rules, to gain general world knowledge.



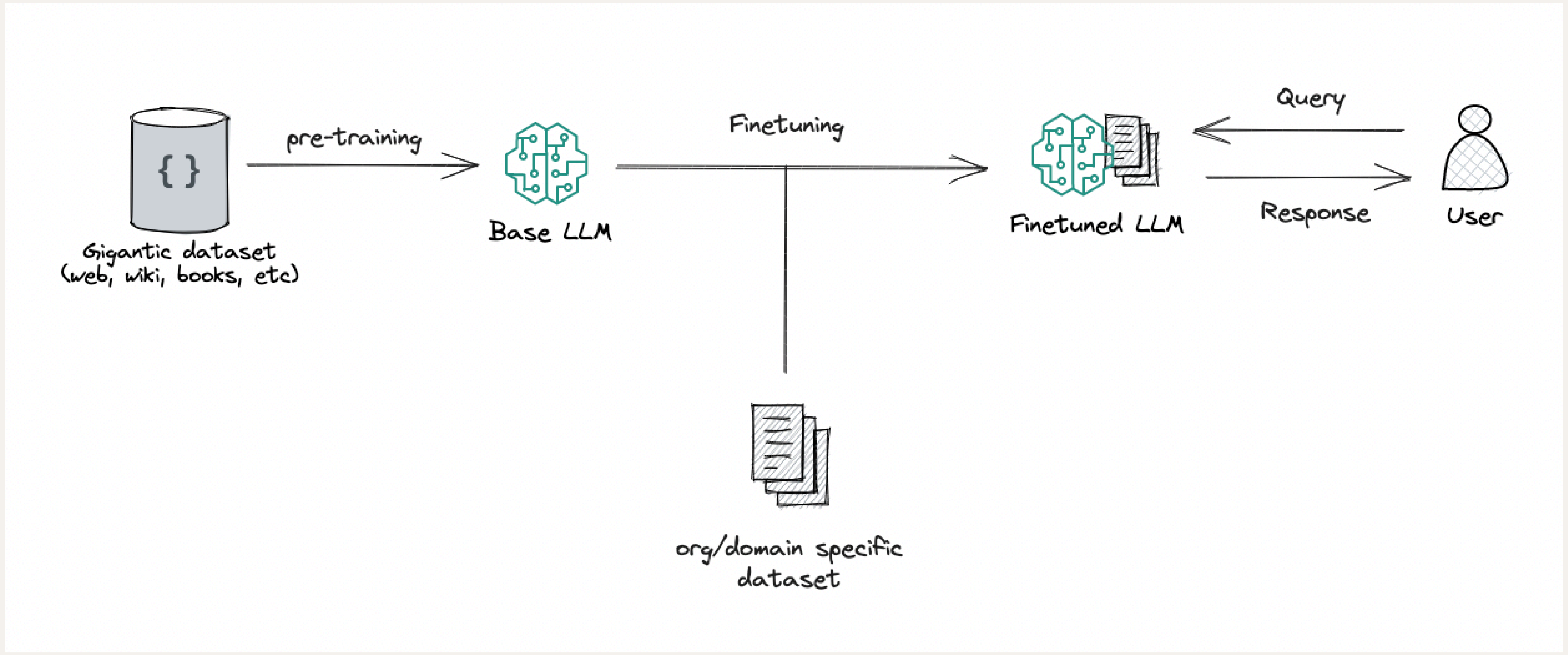
Token String	Token ID	Embedding / Vector Representation
'_The'	37	[-0.0513, -0.0584, 0.0230, ...]
'_teacher'	3145	[-0.0335, 0.0167, 0.0484, ...]
'_teaches'	11749	[-0.0151, -0.0516, 0.0309, ...]
'_the'	8	[-0.0498, -0.0428, 0.0275, ...]
'_student'	1236	[-0.0460, 0.0031, 0.0545, ...]
...	...	...

Pre-Training

- First, very large data sets are prepared.
- Then this data is cleaned (such as duplicate cases).
- Tokenization
- One of the pre-training methods such as MLM, CLM, NSP, Autoregressive modeling is used
    - Masked Language Modeling (MLM): The model tries to guess the missing word by hiding some words in the sentence (masking).
    - Causal Language Modeling (CLM): The model tries to guess the next word using only the previous words.
    - Next Sentence Prediction (NSP): Letting the model learn whether two sentences are meaningful to each other.
    - Autoregressive Models: Allows the model to generate new words using previous words.
- Control processes and re-training when necessary
- At the end of pre-training, the model generally masters the language knowledge, but fine-tuning is needed to perform the specific tasks.

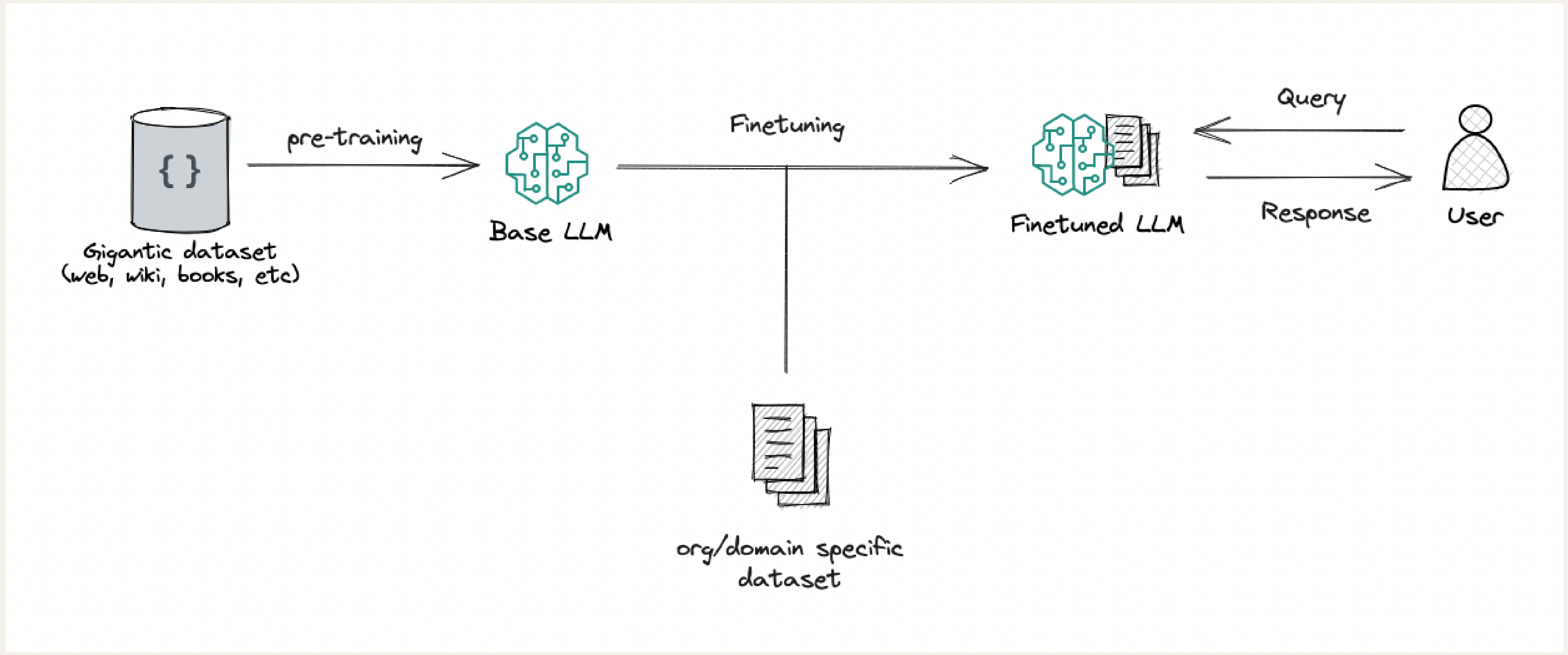
Fine-Tuning

- After the model has gained general information, it must be retrained to perform specific tasks. We can think of it as a master's and PhD after undergraduate.
- The name of this detailed training of the model is fine-tuning.



Fine-Tuning

- First, a dataset prepared for a specific task is created.
- A pre-trained model is loaded/selected.
- The model is then trained for a specific task (one of the fine-tuning methods is selected at this stage)
    - Full fine-tuning:
    - In-Context Learning
    - Adapter Layers


{ }
Gigantic dataset (web, wiki, books, etc)
pre-training
Base LLM
Finetuning
org/domain specific dataset
Finetuned LLM
Query
Response
User

Fine-Tuning

- Full fine-tuning:
  - All weights of the model are updated.
  - Requires large data sets and has high computational cost.
- In-Context Learning
  - Without training the model, guidance is done using only well-prepared examples (prompts).
  - The model learns to respond appropriately to the task.
  - No training is required, but the answers are not definite.
- Adapter Layers
  - Only certain layers of the model are trained, other layers are fixed
  - Faster and less costly
  - For example, with the LoRA (Low-Rank Adaptation) method, performance is increased by training a small portion of large models.

Fine-Tuning

- Challenges:
    - Overfitting: Limited training data may cause overfitting.
    - Data Quality: Effective fine-tuning relies on high-quality labeled data. Ensuring the data quality is hard in that sense.

Pre-Training vs Fine-Tuning

Feature	Pre-Training	Fine-Tuning
Goal	Learn general information	Learn specific information
Training Time	Very long	Short
Learning Procedure	Learns language structure and word relationships	Gains knowledge on a specific topic
Resource for Training	High	Low

Tokenization

- The process of breaking down text into smaller pieces. It is the process of converting it into a numerical format that LLM can understand.
- Types:
  - Word-based tokenization
    - Each word is a token
    - easy to understand because the direct meaning is preserved
    - problems in different languages, vocabulary must be very large
  - Character-based tokenization
    - each character is a token
    - very small vocabulary
    - works language independent
  - Subword-based tokenization
    - splits words into meaningful pieces
    - most common
    - BPE, WordPiece, SentencePiece
  - Byte-Level Tokenization

Tokenization

- Word-based: “Cats are wonderful.” -> [”Cats”, “are”, “wonderful”, “.”]
- Character-based: “Cats” -> [”C”, “a”, “t”, “s”]
- Subword: “undo” -> [”un”, “do”]

- Larger vocabulary requires more memory and processing, while smaller vocabulary requires less processing and memory.
- Tokenization is very important for a fast and efficient training process.

Embeddings

- It is the process of converting words/tokens into numerical vectors. It digitizes the words/tokens and allows them to be represented in virtual space.
- Types:
  - Word2Vec
  - GloVe
  - FastText
  - BERT Embeddings
  - Transformer Based (GPT)

Embeddings Process

- Tokenization:
    - Words or word fragments are created.
- Token Mapping:
    - A numeric ID is assigned to each word.
- Embedding Layer:
    - IDs are converted to fixed length vectors.
- Sentence Representation:
    - All word vectors are combined to produce a sentence vector.

Evaluation Criterias

- Perplexity (PPL):
    - Indicates how well the text is predicted. Lower is better.
- Accuracy:
    - Measures how often the model's predictions match the ground truth
- F1/Recall/Precision:
- Bilingual Evaluation Understudy (BLUE):
    - Measures how similar the generated text is to reference human-written text
- Recall-Oriented Understudy for Gisting Evaluation (ROUGE):
    - Measures the overlap between model-generated text and reference text.
- Metric for Evaluation of Translation with Explicit ORdering (METEOR):
    - Similar to BLEU but considers synonyms and word order.
- Human Rate:
    - Experts or crowd-workers evaluate model according to fluency, coherence, factual correctness, and relevance.

Real-World Applications

- Blog post production
- Article writing
- Translation
- Coding
- Customer Service
- Banking Sector: Fraud detection
- All kinds of report summarization
- Etc etc. In short, LLMs are now found in everything that comes to mind. Although ChatGPT is the “selpak” of this sector, LLMs are everywhere beyond ChatGPT.

THANKS

Süleyman Emir Taşan