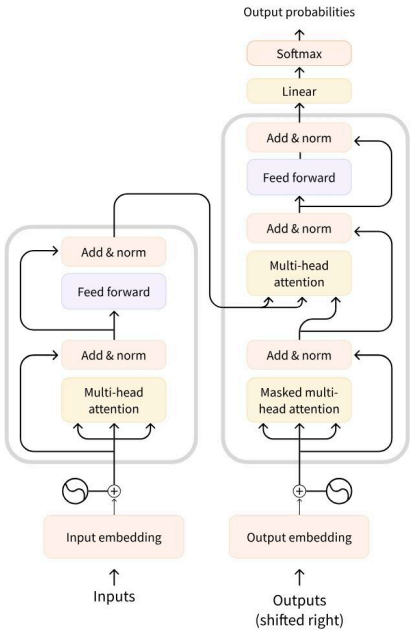
What is a Large Language Model?

its objective is to predict the next token, given a sequence of previous tokens

An LLM is a type of AI model that excels at understanding and generating human language. They are trained on vast amounts of text data, allowing them to learn patterns, structure, and even nuance in language. These models typically consist of many millions of parameters.

Most LLMs nowadays are built on the Transformer architecture—a deep learning architecture based on the “Attention” algorithm, that has gained significant interest since the release of BERT from Google in 2018.



31

There are 3 types of transformers:

1. Encoders
   An encoder-based Transformer takes text (or other data) as input and outputs a dense representation (or embedding) of that text.
   - Example: BERT from Google

    - Use Cases: Text classification, semantic search, Named Entity Recognition
    - Typical Size: Millions of parameters
2. Decoders
   A decoder-based Transformer focuses on generating new tokens to complete a sequence, one token at a time.
    - Example: Llama from Meta
    - Use Cases: Text generation, chatbots, code generation
    - Typical Size: Billions (in the US sense, i.e., 10^9) of parameters
3. Seq2Seq (Encoder–Decoder)
   A sequence-to-sequence Transformer *combines* an encoder and a decoder. The encoder first processes the input sequence into a context representation, then the decoder generates an output sequence.
    - Example: T5, BART
    - Use Cases: Translation, Summarization, Paraphrasing
    - Typical Size: Millions of parameters

Although Large Language Models come in various forms, LLMs are typically decoder-based models with billions of parameters. Here are some of the most well-known LLMs:
Like : Deepseek R1, OpenAi, SmolLM2

The underlying principle of an LLM is simple yet highly effective: its objective is to predict the next token, given a sequence of previous tokens. A “token” is the unit of information an LLM works with. You can think of a “token” as if it was a “word”, but for efficiency reasons LLMs don’t use whole words.

For example, while English has an estimated 600,000 words, an LLM might have a vocabulary of around 32,000 tokens (as is the case with Llama 2). Tokenization often works on sub-word units that can be combined.

Each LLM has some special tokens specific to the model. The LLM uses these tokens to open and close the structured components of its generation. For example, to indicate the start or end of a sequence, message, or response. Moreover, the input prompts that we pass to the model are also structured with special tokens. The most important of those is the End of sequence token (EOS).

The forms of special tokens are highly diverse across model providers.

Example :

GPT 4 EOS token `<|endoftext|>`

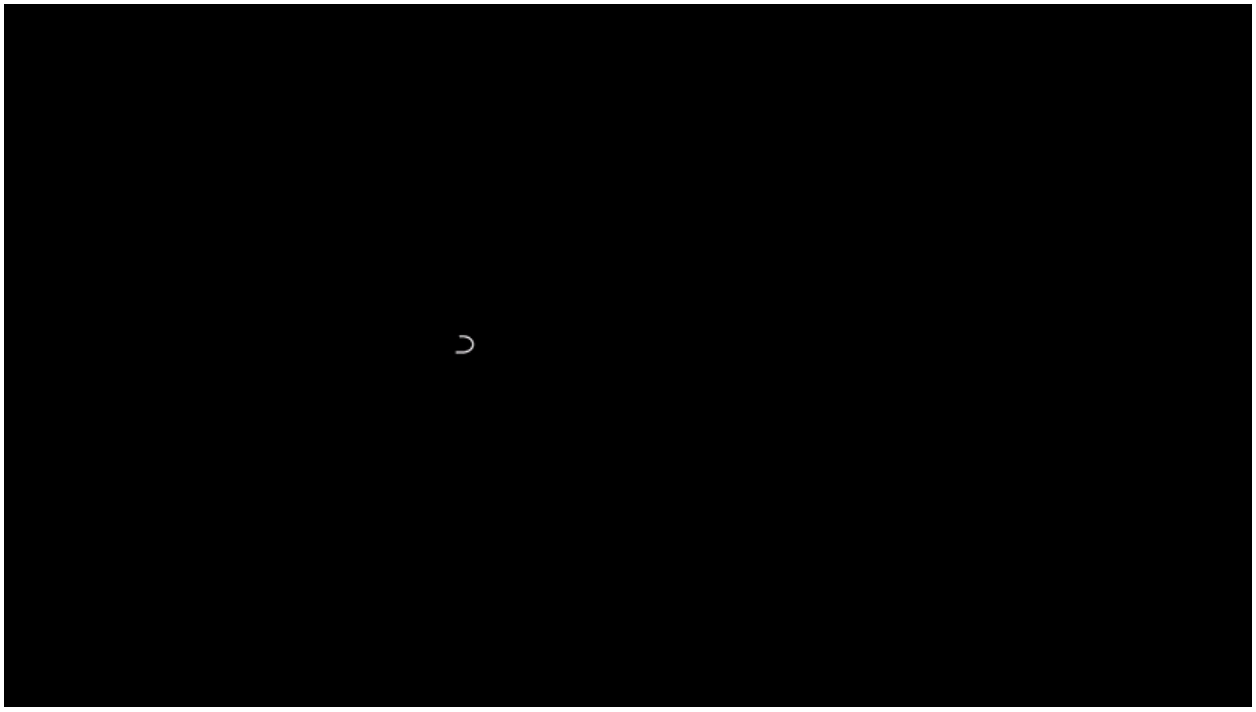
Understanding next token prediction.

LLMs are said to be autoregressive, meaning that the output from one pass becomes the input for the next one. This loop continues until the model predicts the next token to be the EOS token, at which point the model can stop.

In other words, an LLM will decode text until it reaches the EOS. But what happens during a single decoding loop?

While the full process can be quite technical for the purpose of learning agents, here's a brief overview:

- Once the input text is tokenized, the model computes a representation of the sequence that captures information about the meaning and the position of each token in the input sequence.
- This representation goes into the model, which outputs scores that rank the likelihood of each token in its vocabulary as being the next one in the sequence.



Based on these scores, we have multiple strategies to select the tokens to complete the sentence.

- The easiest decoding strategy would be to always take the token with the maximum score.

beam search explores multiple candidate sequences to find the one with the maximum total score–even if some individual tokens have lower scores.

Attention is all you need

A key aspect of the Transformer architecture is Attention. When predicting the next word, not every word in a sentence is equally important; words like “France” and “capital” in the sentence *“The capital of France is …”* carry the most meaning.

This process of identifying the most relevant words to predict the next token has proven to be incredibly effective.

Although the basic principle of LLMs—predicting the next token—has remained consistent since GPT-2, there have been significant advancements in scaling neural networks and making the attention mechanism work for longer and longer sequences.

If you’ve interacted with LLMs, you’re probably familiar with the term *context length*, which refers to the maximum number of tokens the LLM can process, and the maximum *attention span* it has.

Prompting the LLM is important

Considering that the only job of an LLM is to predict the next token by looking at every input token, and to choose which tokens are “important”, the wording of your input sequence is very important.

The input sequence you provide an LLM is called *a prompt*. Careful design of the prompt makes it easier to guide the generation of the LLM toward the desired output.

How are LLMs trained?

LLMs are trained on large datasets of text, where they learn to predict the next word in a sequence through a self-supervised or masked language modeling objective.

From this unsupervised learning, the model learns the structure of the language and underlying patterns in text, allowing the model to generalize to unseen data.

After this initial *pre-training*, LLMs can be fine-tuned on a supervised learning objective to perform specific tasks. For example, some models are trained for conversational structures or tool usage, while others focus on classification or code generation.

How can I use LLMs?

You have two main options:

1. Run Locally (if you have sufficient hardware).
2. Use a Cloud/API (e.g., via the Hugging Face Serverless Inference API).

Throughout this course, we will primarily use models via APIs on the Hugging Face Hub. Later on, we will explore how to run these models locally on your hardware.

How are LLMs used in AI Agents?

LLMs are a key component of AI Agents, providing the foundation for understanding and generating human language.

They can interpret user instructions, maintain context in conversations, define a plan and decide which tools to use.