Introduction to Large Language Models (LLMs) & Basic Prompting Techniques

TURING

Outline

- What are LLMs ?
- How are LLMs Trained?
- Generative Pretrained Transformer(GPT) Architecture Explained

What Are LLMs?

LLMs, or Large Language Models, are advanced artificial intelligence systems designed to understand and generate human-like text based on vast amounts of data. They are built using deep learning techniques, particularly transformer architectures, and are trained on extensive corpora of text from diverse sources such as books, articles, websites, and other written materials.

How are LLMs Trained?

Analogy: LLM as a College Student

Foundation Building:

- LLM: Trained on vast amounts of diverse text data, like a human brain exposed to a variety of experiences from birth.
- Student: Early education (K-12) provides a broad knowledge base across many subjects, preparing for specialized learning in college.

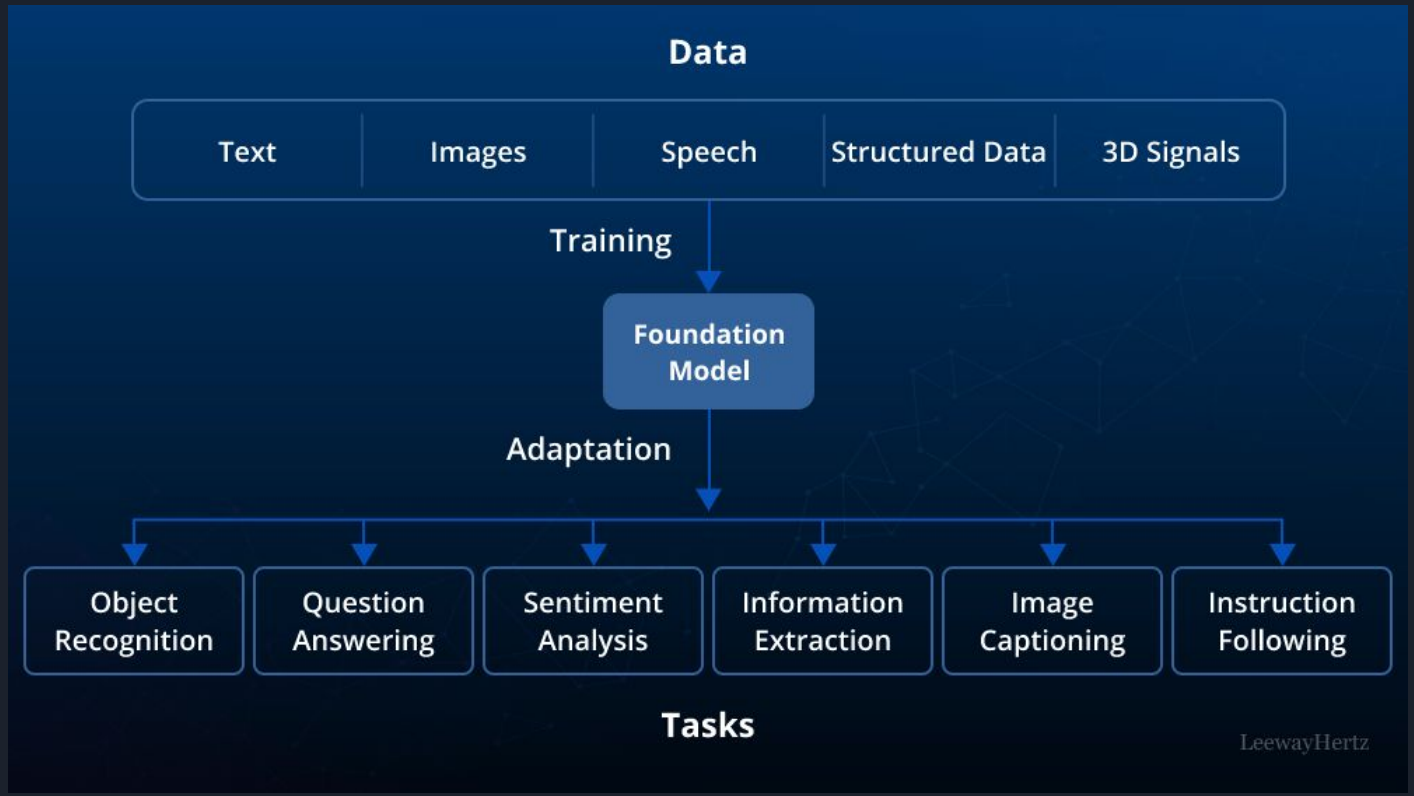
How are LLMs Trained?

Specialized Learning:

LLM: Fine-tuned on specific datasets, similar to choosing a major in college.

Student: Focuses on their major, taking specialized courses in their field.

How are LLMs Trained?

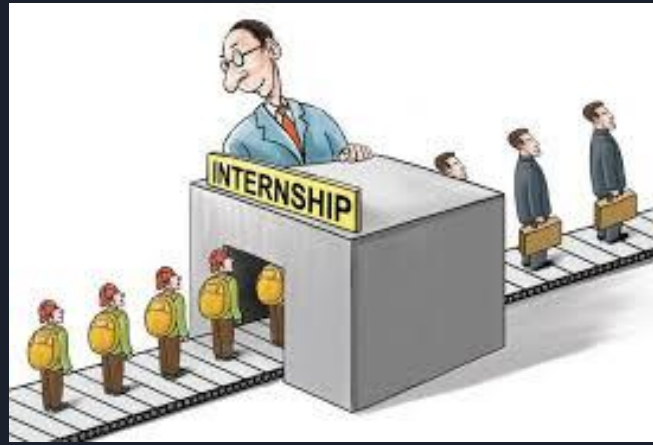


How are LLMs Trained?

Continuous Learning:

LLM: Updated with new data to improve accuracy and relevance.

Student: Engages in lifelong learning, internships, research projects, and continuing education.

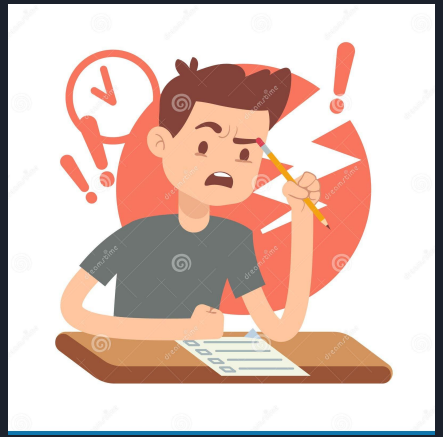


How are LLMs Trained?

Performance Assessment:

LLM: Evaluated through benchmarks and real-world tasks, fine-tuned based on feedback.

Student: Assessed through exams, assignments, and projects, using feedback to improve.



How are LLMs Trained?

Practical Application:

LLM: Generates text, answers questions, and assists with various tasks.

Student: Applies knowledge in the workforce, solving real-world problems.

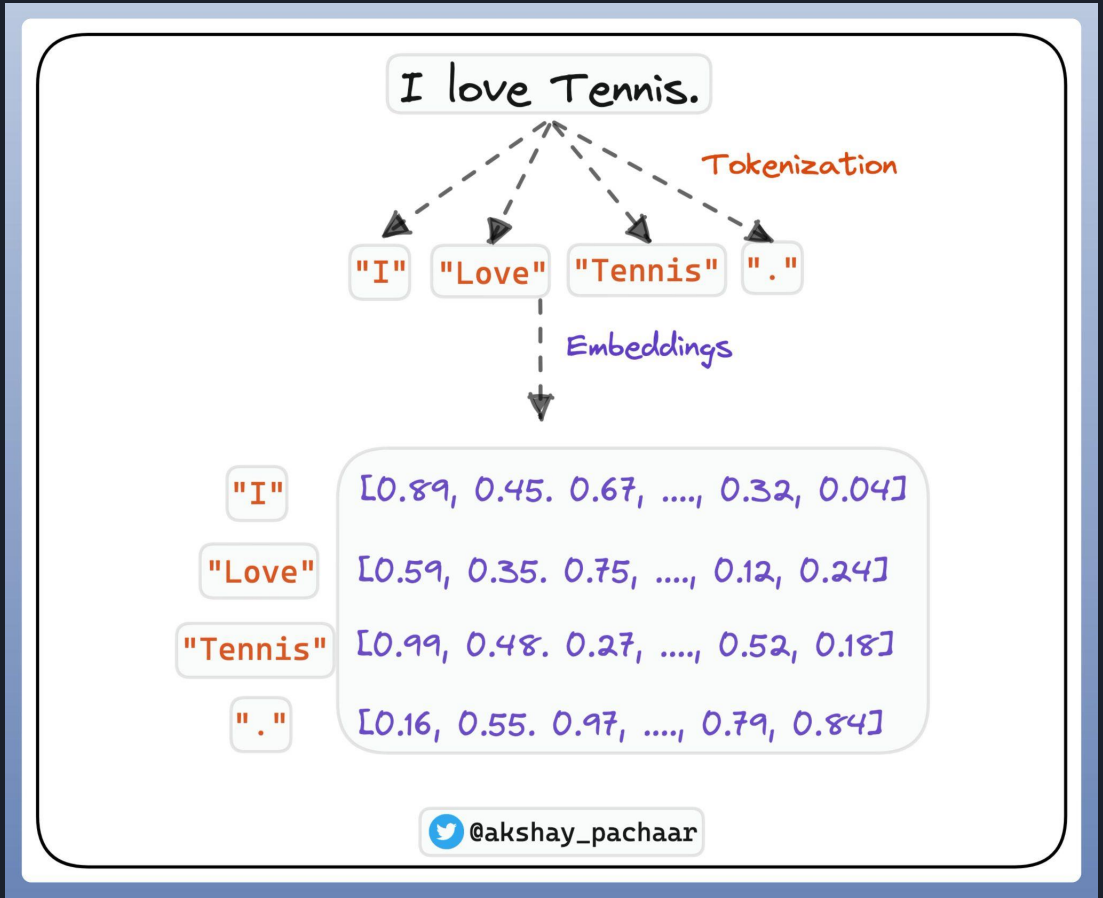
Generative Pretrained Transformer(GPT) Architecture Explained

Input Representation:

Tokenization: Break the input text into smaller pieces called tokens (words or parts of words) and convert them into numbers that the model can process.

Embedding: Turn each token into a high-dimensional vector (like a list of numbers) that captures its meaning. Add positional embeddings to provide information about the order of each token in the sequence.

Generative Pretrained Transformer(GPT) Architecture Explained



Generative Pretrained Transformer(GPT) Architecture Explained

Transformer Blocks:

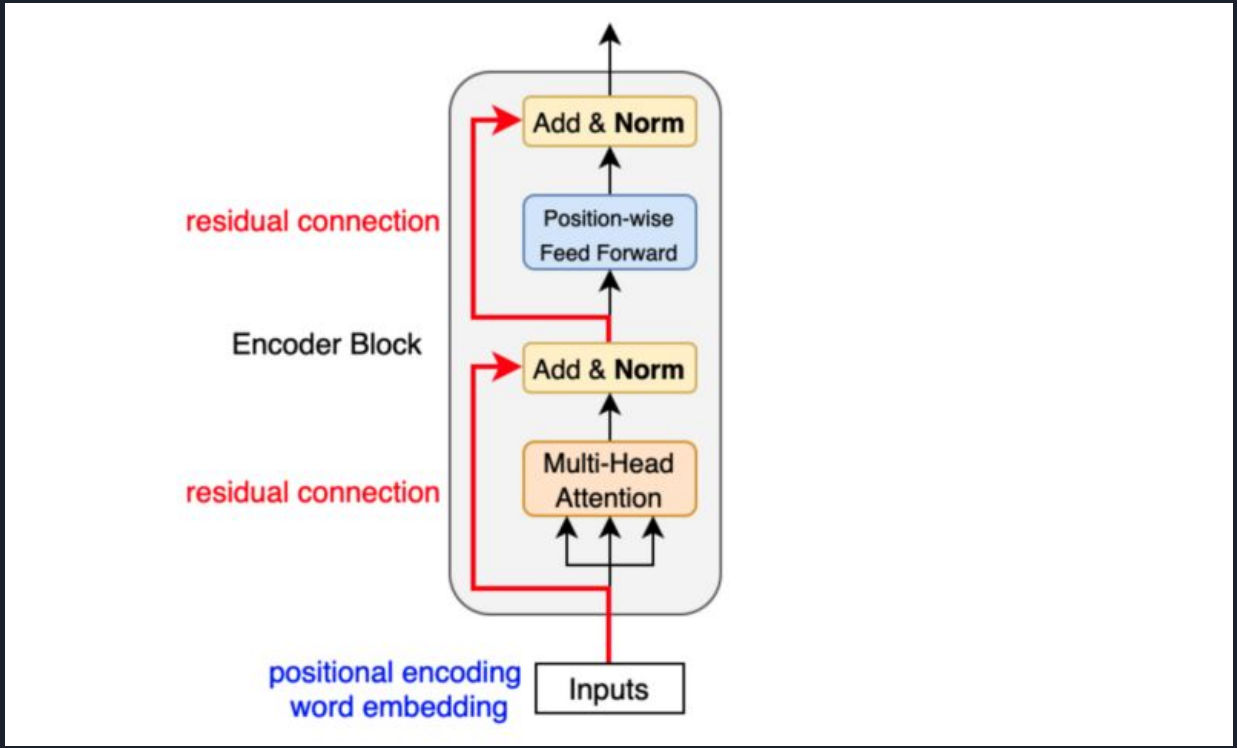
Encoder (Understanding the Input):

Self-Attention: The model looks at all the words in the input and figures out which ones are important for understanding each word. This helps the model focus on relevant parts of the input when generating or understanding text.

Feed-Forward Network: Each word’s representation is processed through a small neural network to learn more complex patterns.

Normalization and Residuals: Normalizes the output and adds shortcuts to help the model learn better.

Generative Pretrained Transformer(GPT) Architecture Explained



Generative Pretrained Transformer(GPT) Architecture Explained

Decoder (Generating the Output):

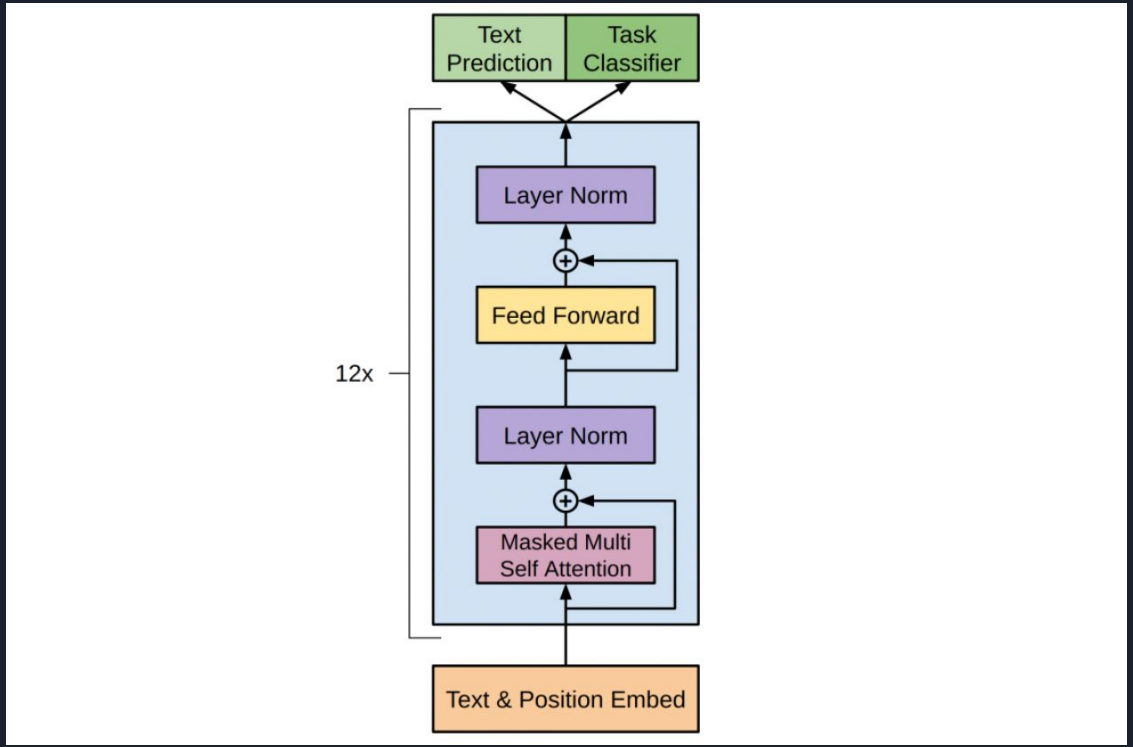
Self-Attention: The decoder also uses self-attention to focus on important words in the output sequence generated so far.

Encoder-Decoder Attention: This allows the decoder to pay attention to the encoded input, understanding how different parts of the input relate to the output generation.

Feed-Forward Network: Further processes the representations to generate the next word in the sequence.

Normalization and Residuals: Stabilizes learning and improves the flow of information.

Generative Pretrained Transformer(GPT) Architecture Explained



Generative Pretrained Transformer(GPT) Architecture Explained

Output Layer:

Predicting Next Word: After processing through all the layers, the model uses a final layer to predict the next word in the sequence. It converts the processed information back into probabilities for each possible next word.

Generating Text: The model generates text one word at a time. It starts with an initial input, predicts the next word, appends that word to the input, and repeats the process until it produces a complete sentence or paragraph.

Generative Pretrained Transformer(GPT) Architecture Explained

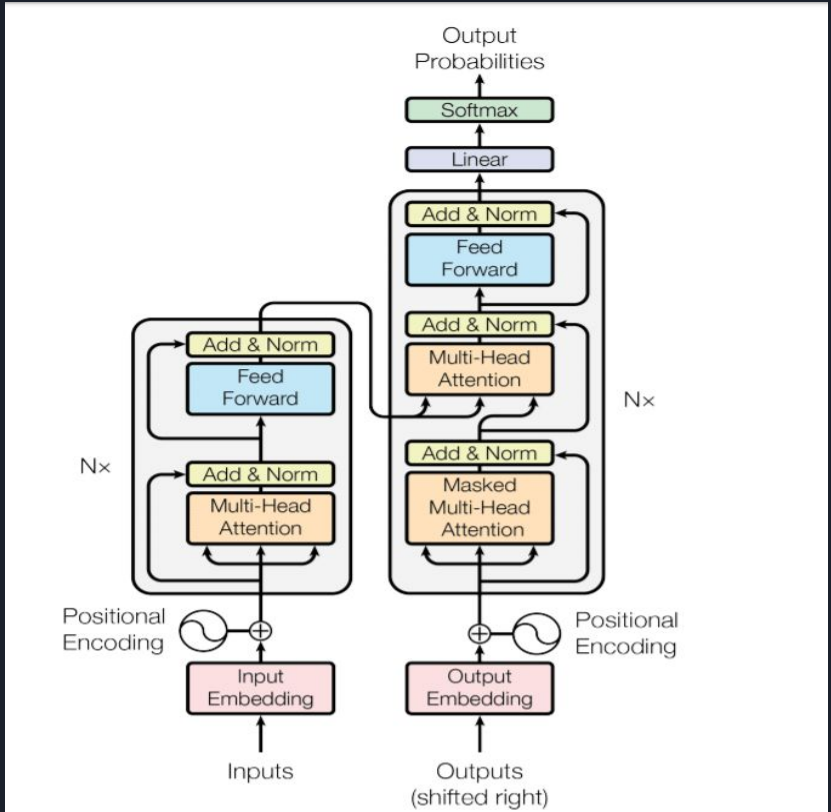


Figure 1: The Transformer - model architecture.

Some resources:

Attention is all you need:

https://arxiv.org/abs/1706.03762

How large Language medals work:
https://medium.com/data-science-at-microsoft/how-large-language-models-work-91c362f5b78f

Basic Prompting Techniques

Turing AI Academy

Basic prompting techniques

What is a prompt ?

Prompting refers to the method of interacting with large language models (LLMs) by providing input, or "prompts," to generate desired outputs. A prompt is essentially a question, instruction, or piece of text that you give to an LLM to elicit a response.

The quality and clarity of the prompt directly influence the effectiveness and relevance of the model's response.

Prompt Patterns

Role-Playing

Description: Asking the model to respond from the perspective of a particular character or entity.

Example: "As a customer service representative, how would you respond to a complaint about late delivery?"

Use Case: For scenarios where perspective-taking or empathy is required.

Prompt Patterns

Chain-of-Thought

Description: Encouraging the model to reason through its answer step-by-step, similar to how humans think through problems.

Example: "To solve this math problem let's reason through it step by step , first identify the known variables, then apply the appropriate formula, and finally, solve for the unknown variable."

Use Case: For tasks requiring logical reasoning and problem-solving.

Prompt Patterns

Examples and Patterns

Description: Providing examples or patterns to illustrate the desired response format.

Example: "Generate a JSON object with the following structure: { 'name': 'string', 'age': 'integer', 'email': 'string' }"

Use Case: For tasks requiring specific formats or when the structure of the response is important.

Prompt Patterns

Step-by-Step Instructions

Description: Breaking down a complex task into a series of simpler, ordered steps.

Example: "Explain how to change a flat tire. First, list the tools needed. Then, describe the steps to replace the tire."

Use Case: For tasks that require detailed procedures or multi-step processes.

Prompt Patterns

Contextual Prompts

Description: Providing context or background information to guide the model's response.

Example: "In a world where humans and robots coexist peacefully, describe a typical day for a robot."

Use Case: For generating creative content or when the context is crucial for understanding the prompt.

Best Practices

Be Clear and Specific:

Why: Clear and specific prompts help the model understand exactly what you are asking for, reducing ambiguity and leading to more accurate responses.

How: Instead of vague questions, provide detailed instructions or ask for specific information.

Example: Rather than asking "What is Python?" ask "What are the main features of the Python programming language?"

Best Practices

Provide Context:

Why: Context helps the model generate responses that are more relevant and informed by the surrounding information.

How: Include any relevant background information or details that set the stage for the question.

Example: Instead of "How does it work?" provide context like "In the context of machine learning, how does supervised learning work?"

Best Practices

Use Simple and Direct Language:

Why: Simple and direct language minimizes misunderstandings and ensures the model can process the prompt correctly.

How: Avoid overly complex sentences or jargon unless necessary for the topic.

Example: Instead of "Elucidate the multifaceted ramifications of global warming," say "Explain the effects of global warming."

Best Practices

Specify the Desired Format:

Why: Specifying the format can guide the model to structure the response in a way that is most useful for you.

How: Indicate whether you want a list, a summary, a step-by-step guide, etc.

Example: Instead of "Tell me about the benefits of exercise," say "List three benefits of regular exercise."

Best Practices

Iterate and Refine Prompts:

Why: Iterating and refining prompts can help achieve more precise and accurate responses, especially if the initial output isn't satisfactory.

How: Start with a basic prompt, evaluate the response, and then refine your prompt for clarity or additional detail.

Example: If "Describe a healthy diet" gives a broad answer, refine it to "Describe a healthy diet for someone with diabetes."

THANK YOU!

Lets connect