Brookhaven National Laboratory

U.S. DEPARTMENT OF ENERGY

Large Language Models (LLM) Tutorial

Xi Yu

Research Associate Computational Science

xyu1@bnl.gov

8/2/2024

@BrookhavenLab

About Me

Position:

- Postdoctoral Research Associate, Foundation Model Group, CSI department, 2022-present.

Education:

- Ph.D. in Electrical Engineering, University of Florida. 2019-2022.

Research Interests:

- Multi-model Large Language Model
- Adversarial Robustness
- Domain Generalization

Contents

Contents

- **Introduction to LLM**
  - Definition and significance
  - Key components and architecture
  - Examples of popular LLMs (e.g., GPT, BERT)
- **Applications of LLMs**
  - Conversation AI
  - LLM Agents
  - Multi-Model LLM
- **Training LLMs**
  - Data collection and preprocessing
  - Fine-tuning techniques
  - In-Context Learning

Definition and significance

❑ What is the definition of LLM?

- *A Large Language Model (LLM) is a type of artificial intelligence (AI) model that uses deep learning techniques to understand and generate human language.*

- *LLMs are trained on vast amounts of text data to learn patterns, structures, and meanings within language, enabling them to perform various natural language processing (NLP) tasks such as text generation, translation, summarization, and answering questions.*

Definition and significance

LLM can

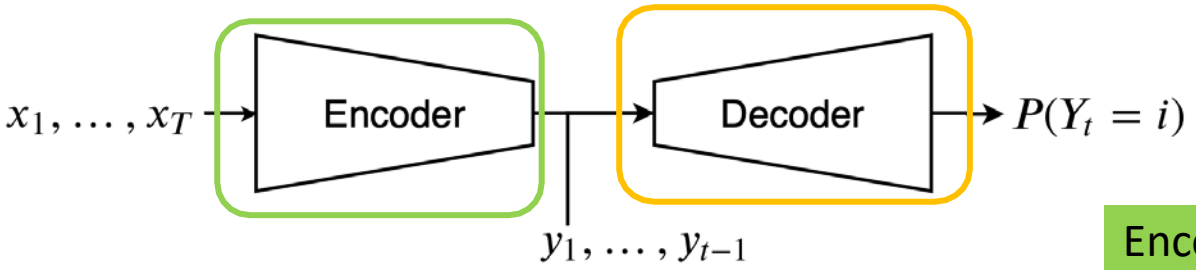
- Translate
- Summarize
- Proof-read and correct
- Explain words
- Create article/email
- Make restaurant suggestions
- Chat with users
- Do Math Calculations
- Answer questions on many subjects
- Suggest names
- Write code

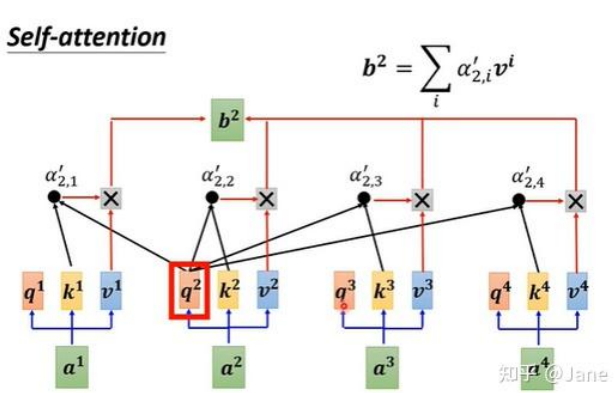
Contents

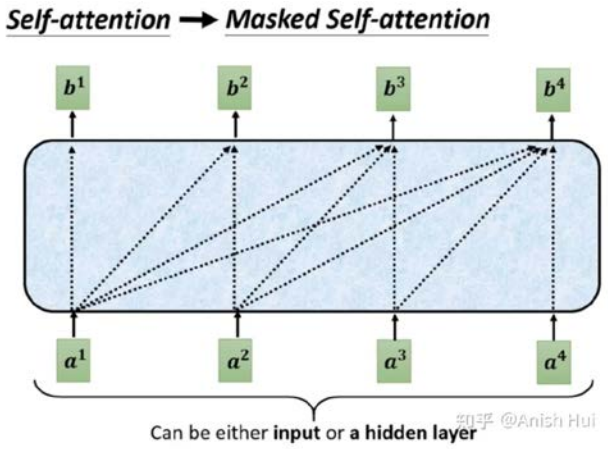
- **Introduction to LLM**
  - Definition and significance
  - Key components and architecture
  - Examples of popular LLMs (e.g., GPT, BERT)
- **Applications of LLMs**
  - Conversation AI
  - LLM Agents
  - Multi-Model LLM
- **Training LLMs**
  - Data collection and preprocessing
  - Fine-tuning techniques
  - In-Context Learning

Key components and architecture

- **Transformers**
  - Encoder-Decoder

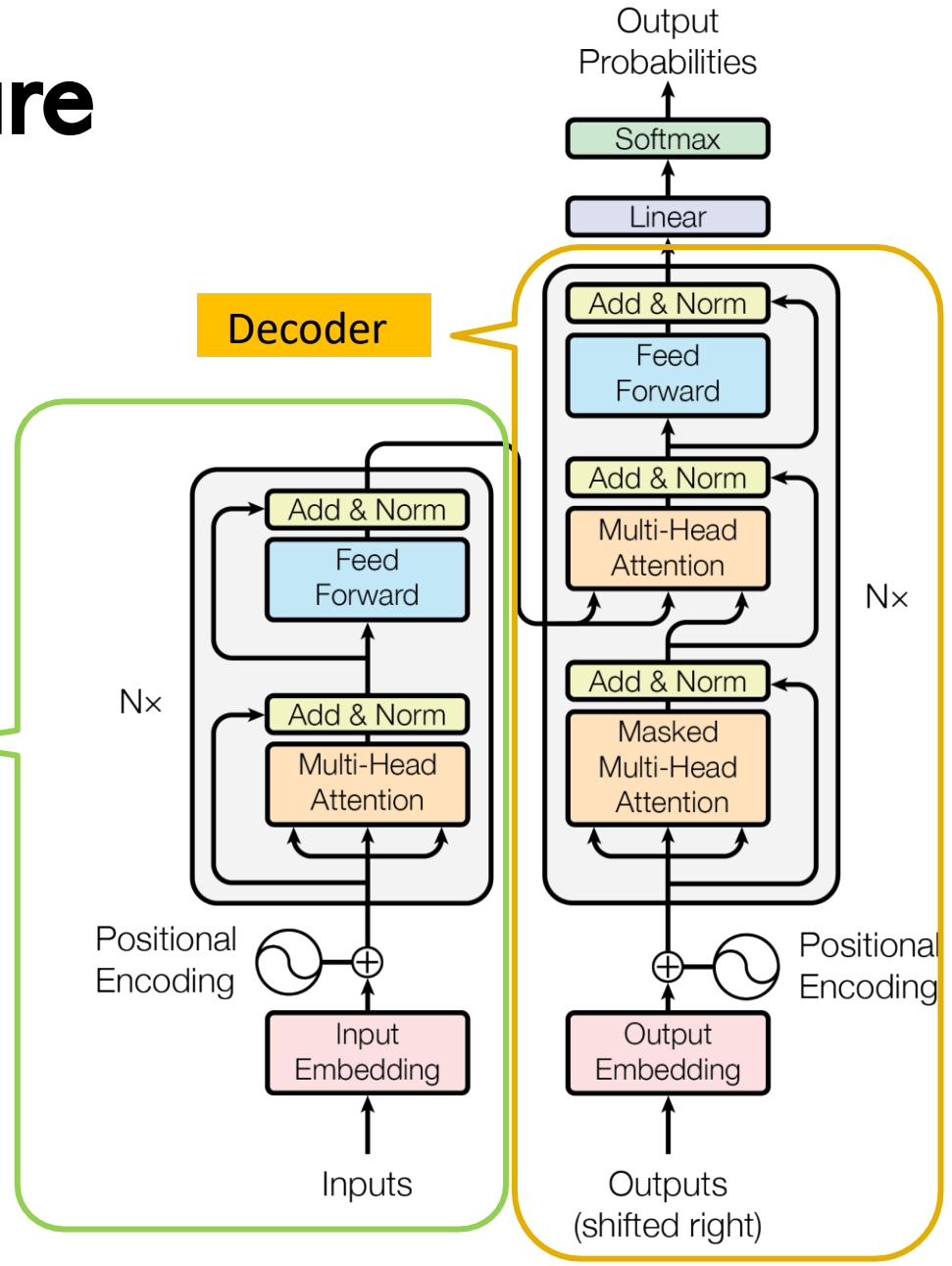






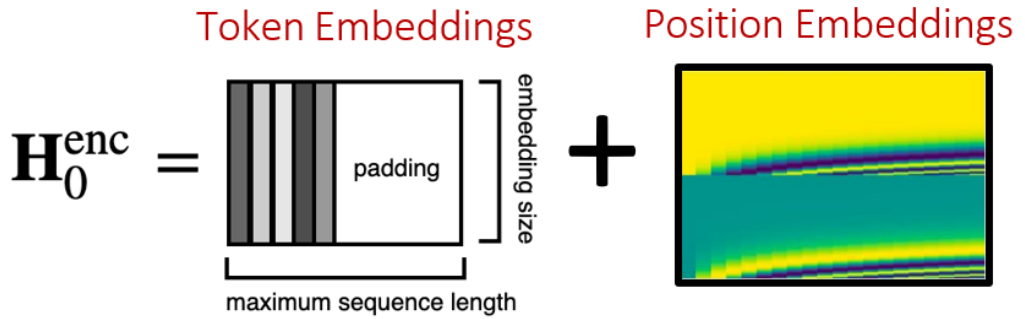
Encoder

Decoder

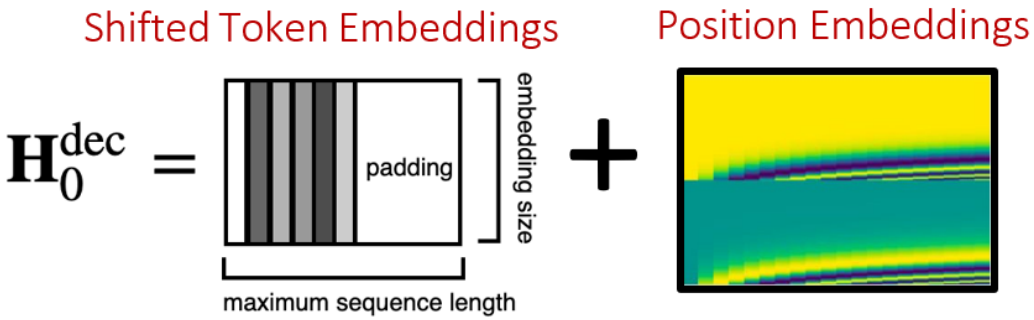


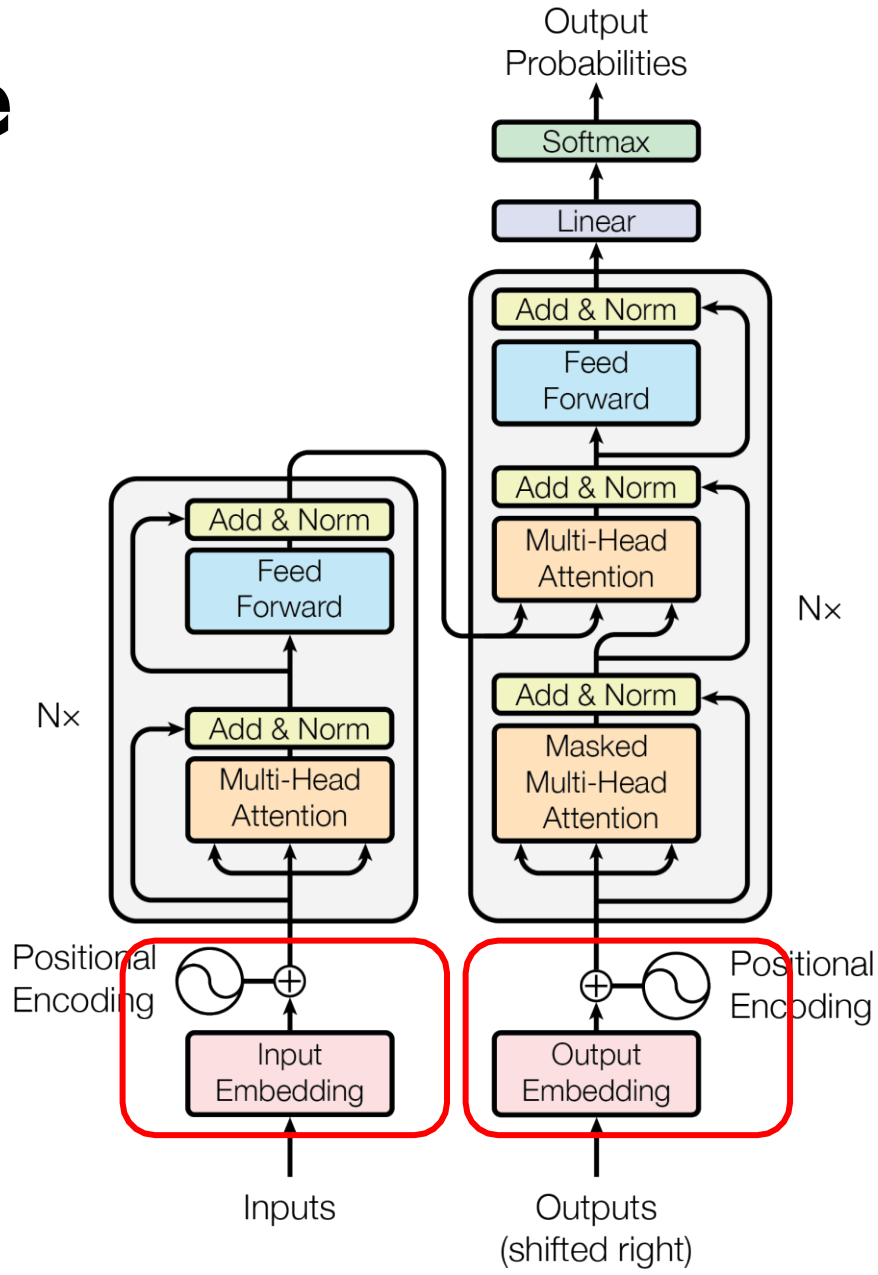
Key components and architecture

- **Transformers**
  - Inputs of encoder and decoder
    - The input into the encoder looks like:



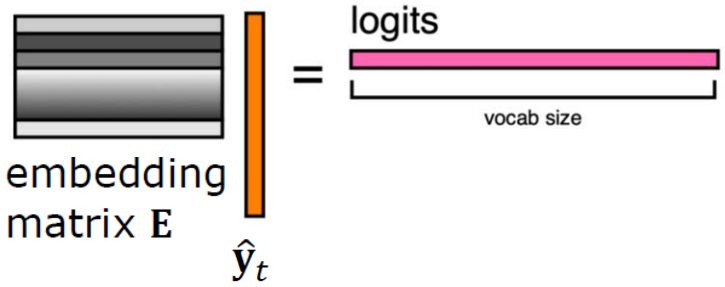
    - The input into the encoder looks like:



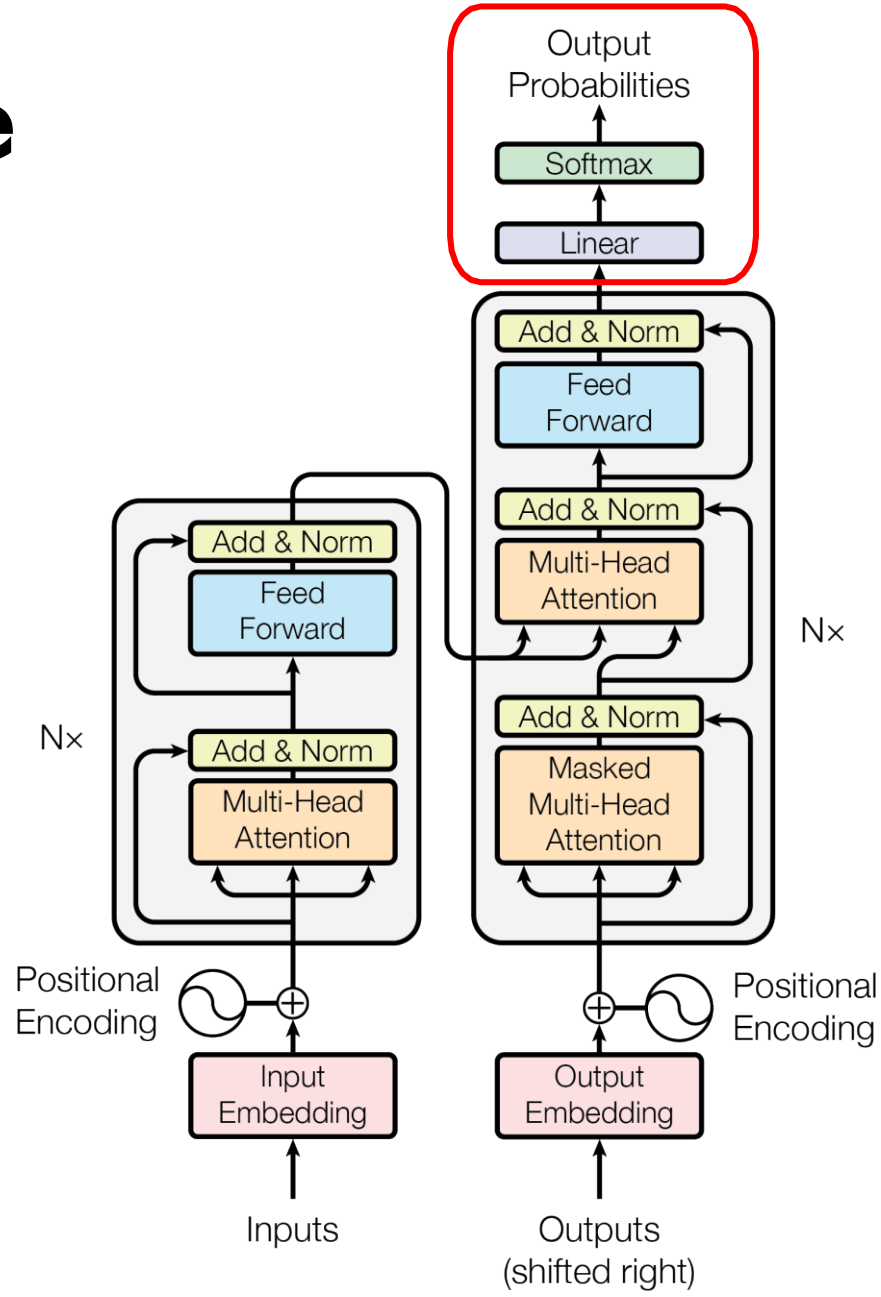


Key components and architecture

- **Transformers**
  - Outputs



$$P(Y_t = i|\mathbf{x}_{1:T}, \mathbf{y}_{1:t-1}) = \frac{\exp(\mathbf{E}\hat{\mathbf{y}}_t[i])}{\sum_j \exp(\mathbf{E}\hat{\mathbf{y}}_t[j])}$$



Key components and architecture

- **Transformers**
  - Self-Attention

$$\mathbf{q}_i = \mathbf{x}_i\mathbf{W}^{\mathbf{Q}}; \mathbf{k}_i = \mathbf{x}_i\mathbf{W}^{\mathbf{K}}; \mathbf{v}_i = \mathbf{x}_i\mathbf{W}^{\mathbf{V}}$$

Final verson:

$$\text{score}(\mathbf{x}_i, \mathbf{x}_j) = \frac{\mathbf{q}_i \cdot \mathbf{k}_j}{\sqrt{d_k}}$$

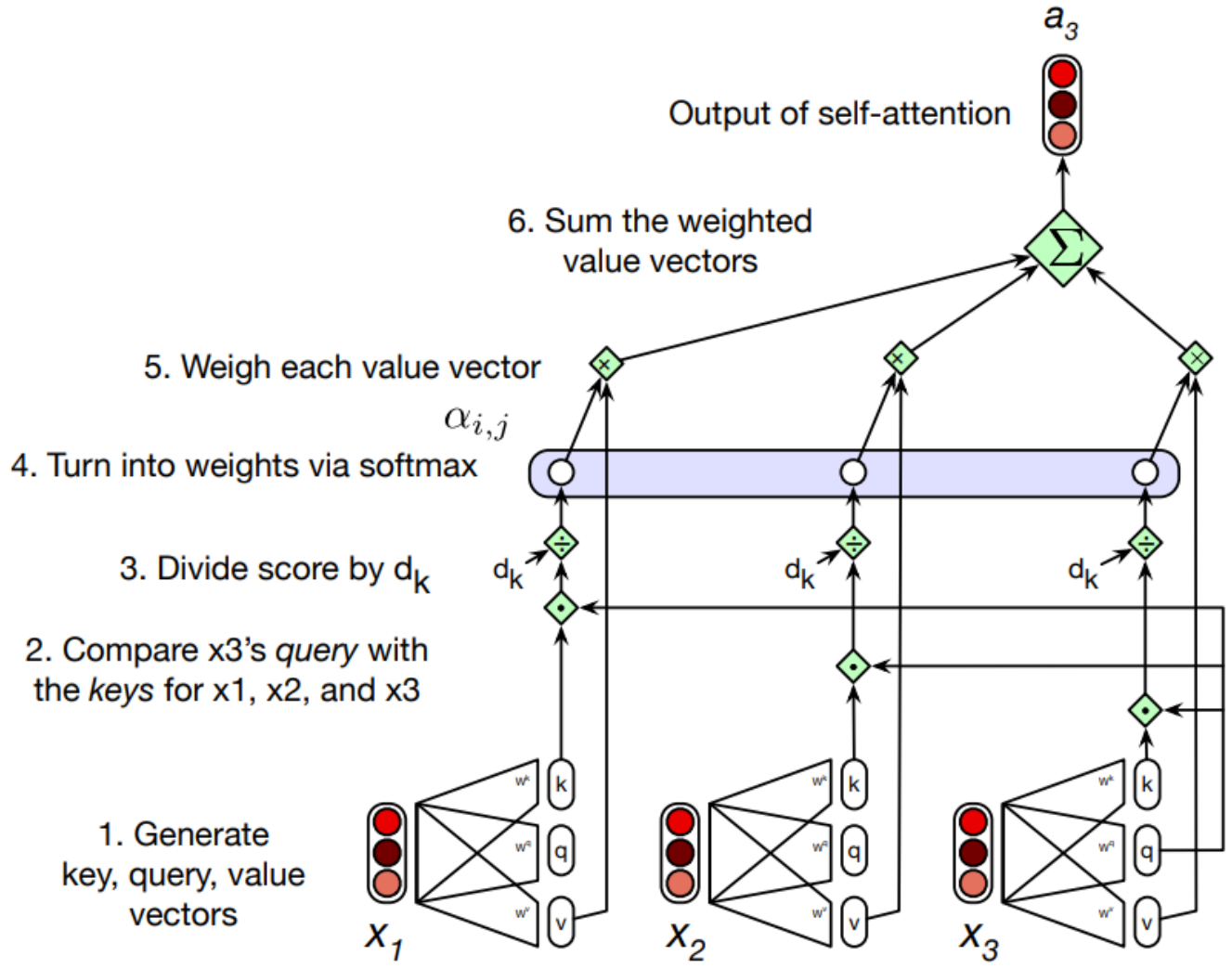
$$\alpha_{ij} = \text{softmax}(\text{score}(\mathbf{x}_i, \mathbf{x}_j)) \ \forall j \leq i$$

$$\mathbf{a}_i = \sum_{j \leq i} \alpha_{ij}\mathbf{v}_j$$

Parallelizing self-attention using a single matrix X

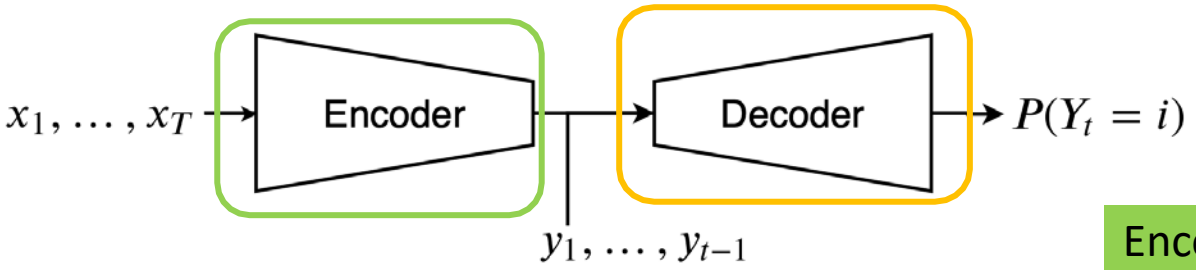
$$\mathbf{Q} = \mathbf{X}\mathbf{W}^{\mathbf{Q}}; \ \mathbf{K} = \mathbf{X}\mathbf{W}^{\mathbf{K}}; \ \mathbf{V} = \mathbf{X}\mathbf{W}^{\mathbf{V}}$$

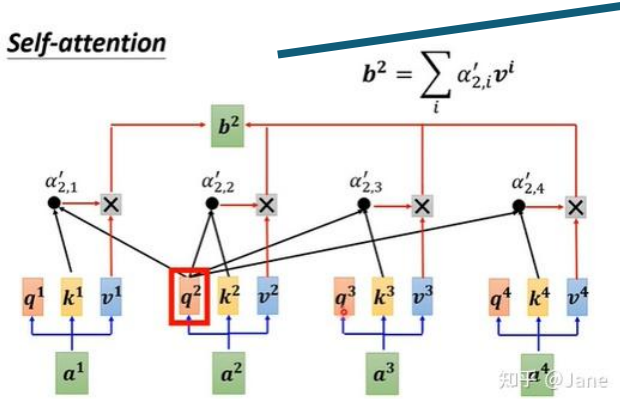
$$\mathbf{A} = \text{SelfAttention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^{\mathsf{T}}}{\sqrt{d_k}}\right)\mathbf{V}$$

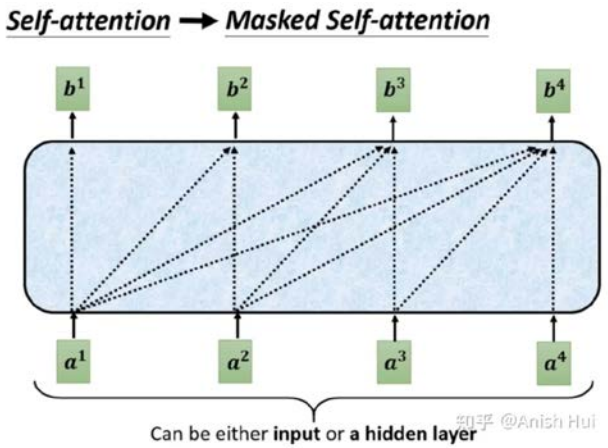


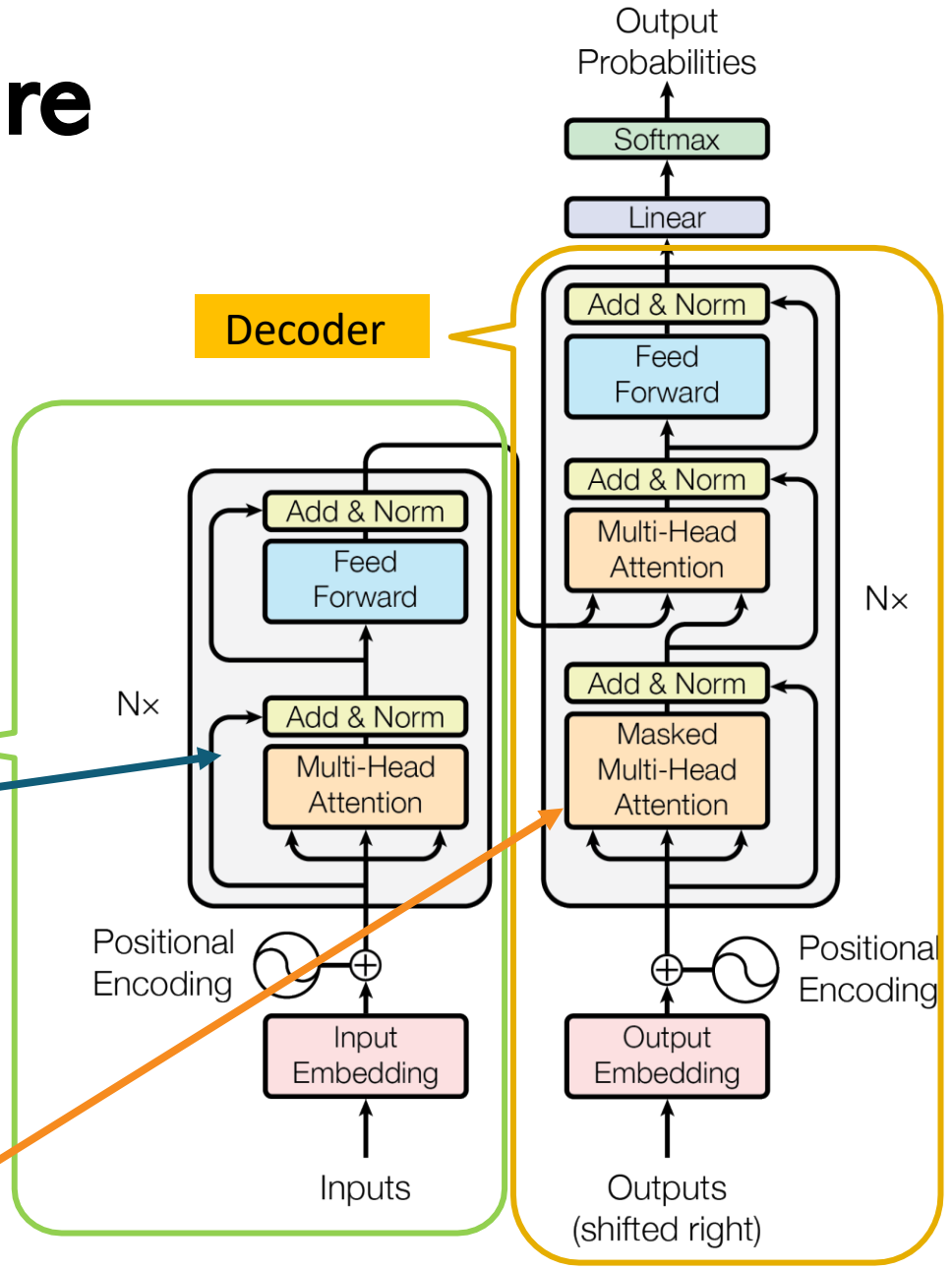
Key components and architecture

- **Transformers**
  - Encoder-Decoder









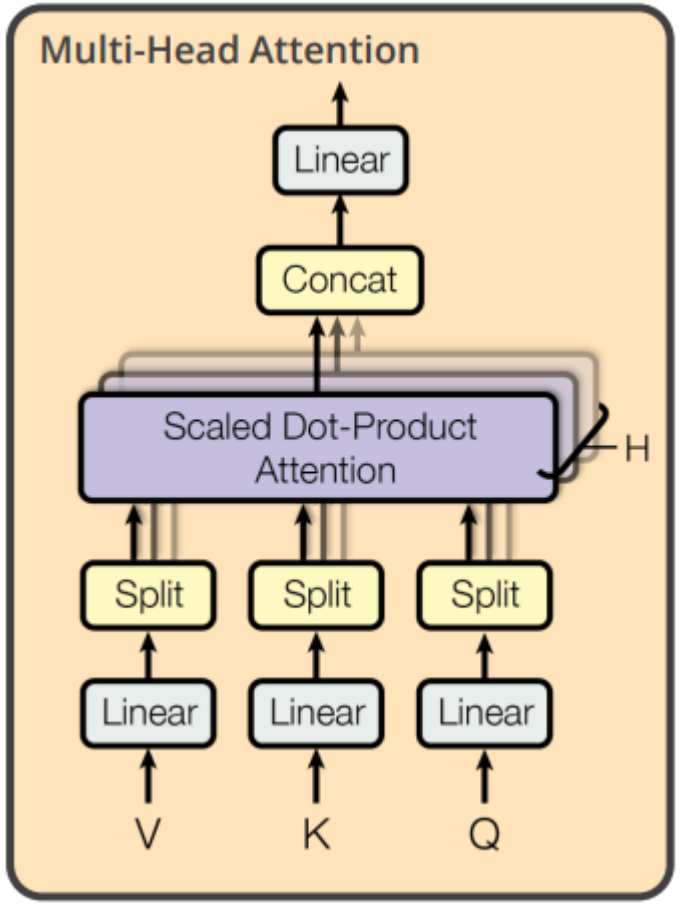
Key components and architecture

- **Transformers**
  - Multi-head Attention

$$\mathbf{Q} = \mathbf{X}\mathbf{W}_i^Q ; \mathbf{K} = \mathbf{X}\mathbf{W}_i^K ; \mathbf{V} = \mathbf{X}\mathbf{W}_i^V$$

$$\mathbf{head}_i = \mathrm{SelfAttention}(\mathbf{Q}, \mathbf{K}, \mathbf{V})$$

$$\mathbf{A} = \mathrm{MultiHeadAttention}(\mathbf{X}) = (\mathbf{head}_1 \oplus \mathbf{head}_2 \ldots \oplus \mathbf{head}_h)\mathbf{W}^O$$



Key components and architecture

❑ Transformers

➢ Encoder step by step

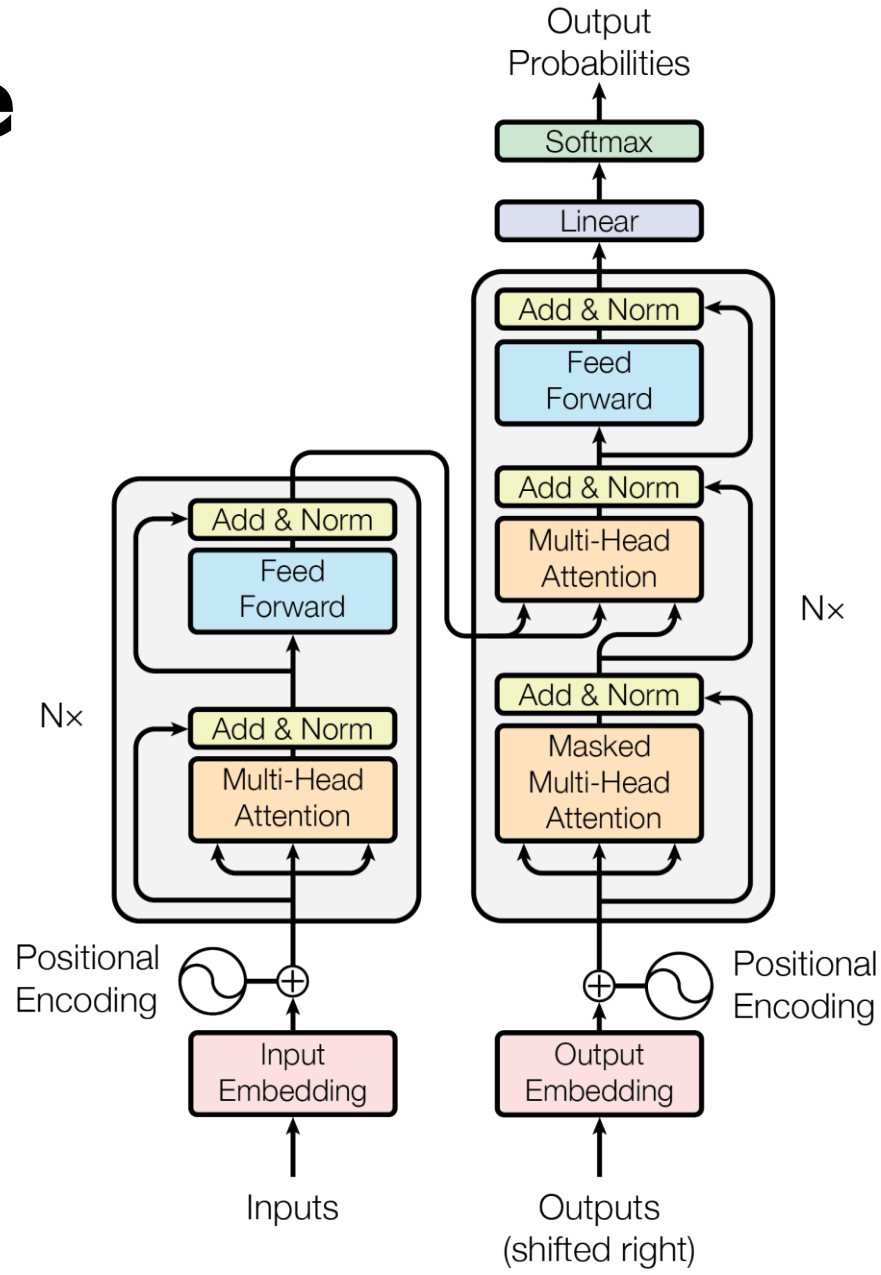
Multi-Head Attention $= \text{MultiHeadAtt}(\mathbf{H}_i^{enc}, \mathbf{H}_i^{enc}, \mathbf{H}_i^{enc})$

Add & Norm $= \text{LayerNorm}($ Multi-Head Attention $+\mathbf{H}_i^{enc})$

Feed Forward $= \max(0,$ Add & Norm $\mathbf{W}_1 + b_1)\mathbf{W}_2 + b_2)$

Add & Norm (2) $= \text{LayerNorm}($ Feed Forward $+$ Add & Norm $)$

$\mathbf{H}_{i+1}^{enc} =$ Add & Norm(2)



Key components and architecture

❑ Transformers

➢ Decoder step by step

Masked Multi-Head Attention $= \text{MultiHeadAtt}(\mathbf{H}_i^{dec}, \mathbf{H}_i^{dec}, \mathbf{H}_i^{dec})$

Add & Norm $= \text{LayerNorm}($ Masked Multi-Head Attention $+\mathbf{H}_i^{dec})$

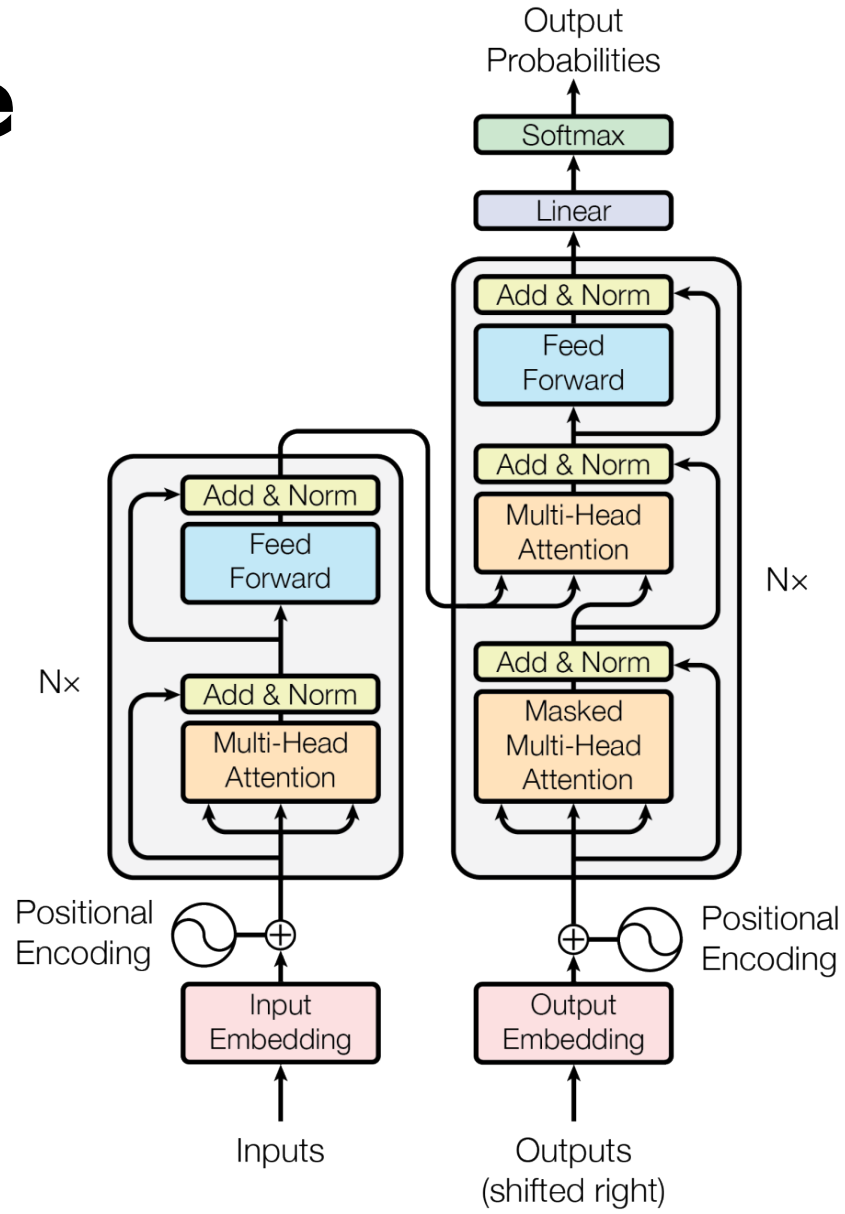
Enc-Dec Multi-Head Attention $= \text{MultiHeadAtt}(\mathbf{H}_i^{enc}, \mathbf{H}_i^{enc},$ Add & Norm $)$

Add & Norm (2) $= \text{LayerNorm}($ Enc-Dec Multi-Head Attention $+$ Add & Norm $)$

Feed Forward $= \max(0,$ Add & Norm (2) $\mathbf{W}_1 + b_1)\mathbf{W}_2 + b_2)$

Add & Norm (3) $= \text{LayerNorm}($ Feed Forward $+$ Add & Norm (2) $)$

$\mathbf{H}_{i+1}^{enc} =$ Add & Norm(3)

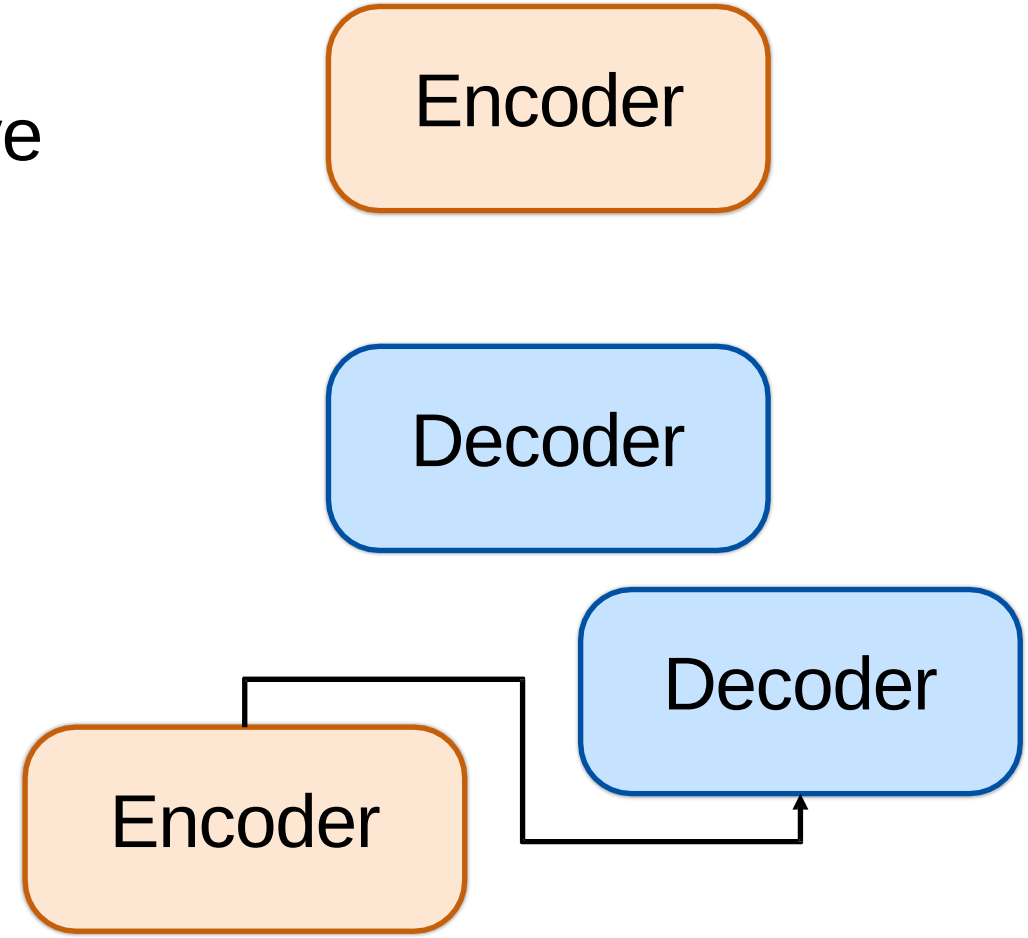


Contents

- **Introduction to LLM**
  - Definition and significance
  - Key components and architecture
  - Examples of popular LLMs (e.g., GPT, BERT)
- **Applications of LLMs**
  - Conversation AI
  - LLM Agents
  - Multi-Model LLM
- **Training LLMs**
  - Data collection and preprocessing
  - Fine-tuning techniques
  - In-Context Learning

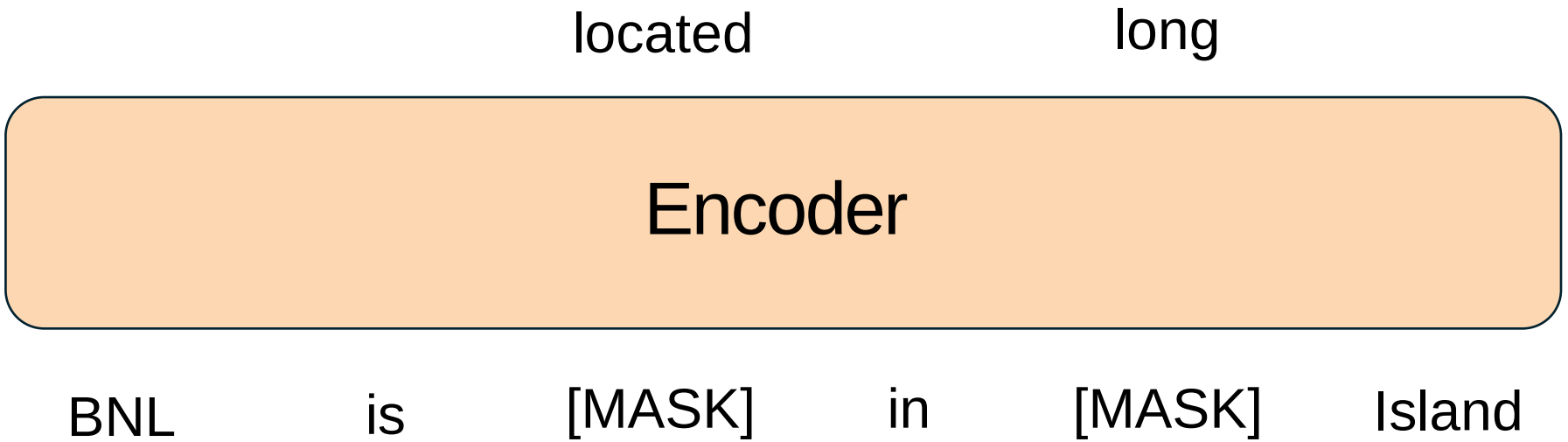
Examples of popular LLMs

- Examples of popular LLMs
  - Encoder-Only Language Model
    - Masked LLM Non-autoregressive
  - Decoder Only Language Model
    - Autoregressive
  - Encoder-Decoder Language Model



Examples of popular LLMs

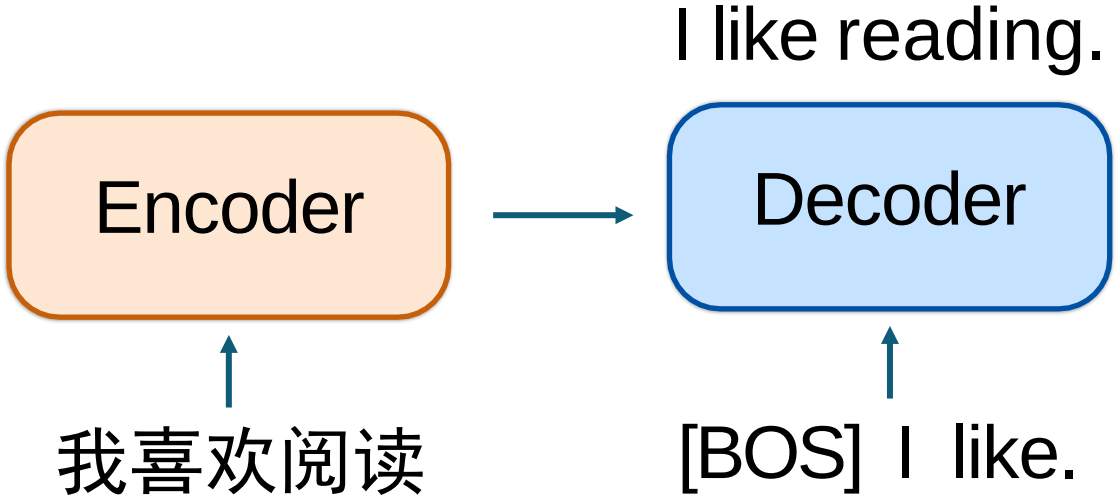
- Examples of popular LLMs
  - Encoder-Only Language Model
    - Masked LLM Non-autoregressive
    - Mask Prediction: **BERT** $P_{\theta_{enc}}(\boldsymbol{X}_{mask}|\boldsymbol{X})$
    - Can also be used to generate language



Examples of popular LLMs

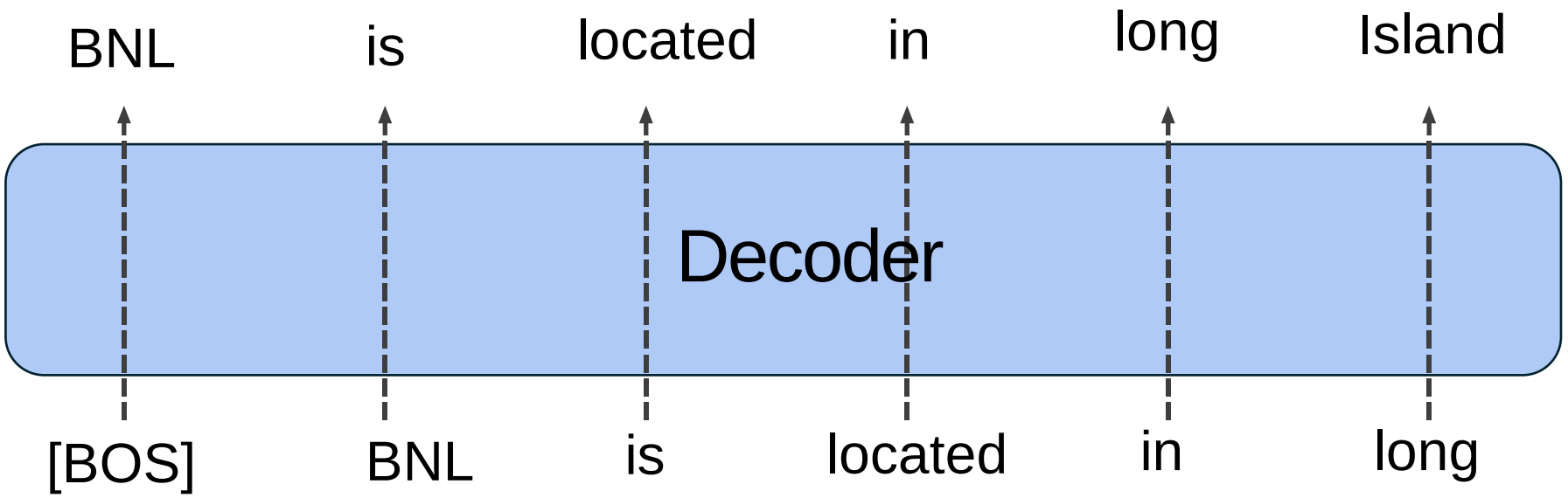
- Examples of popular LLMs
  - Encoder-Decoder Language Model
    - Text-to-Text LM: **Google T5**
    - Model the probability

$$P(Y_{1:m}|X_{1:n}) = \prod_{i=1}^{m} P_{\{\theta_{enc},\theta_{dec}\}}(y_i|Y_{1:i-1}, X_{1:n})$$



Examples of popular LLMs

- Examples of popular LLMs
  - Decoder-Only Language Model
    - **ChatGPT**, Causal Language Model.
    - Model the probability $P(x_i|X_{1:i-1})$



Examples of popular LLMs

- Examples of popular LLMs
  - Why ChatGPT, LLaMA is Decoder-Only Language Model?
    - More efficient for training and inference than Encoder-Decoder model.
    - More appropriate than Encoder-Only model for predict the next token of input.

Contents

- Introduction to LLM
  - Definition and significance
  - Key components and architecture
  - Examples of popular LLMs (e.g., GPT, BERT)
- Applications of LLMs
  - Conversation AI
  - LLM Agents
  - Multi-Model LLM
- Training LLMs
  - Data collection and preprocessing
  - Fine-tuning techniques
  - In-Context Learning

Applications of LLMs

❑ Conversation AI

Math Reasoning

You

Roger has 4 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

ChatGPT

Roger initially has 4 tennis balls. When he buys 2 more cans of tennis balls, and each can has 3 tennis balls, he adds 2 cans * 3 tennis balls per can = 6 tennis balls to his collection.
So, Roger now has a total of 4 initial tennis balls + 6 newly purchased tennis balls = 10 tennis balls.

Rewrite Text with Style

You

Please write the following sentence in Yoda's style. Here are a few examples.
You have much to learn. -> Much to learn, you still have.
I would say you have found someone, hmmm? -> Found someone, you have, I would say, hmmm?

You should not drive that fast. ->

ChatGPT

Drive that fast, you should not.

You

You are given a 0-indexed array mountain. Your task is to find all the peaks in the mountain array. please write a python program.

ChatGPT

```
def find_peaks(mountain):
    peaks = []
    for i in range(1, len(mountain) - 1):
        if mountain[i] > mountain[i - 1] and
mountain[i] > mountain[i + 1]:
            peaks.append(i)
    return peaks
```

Commonsense Reasoning

You

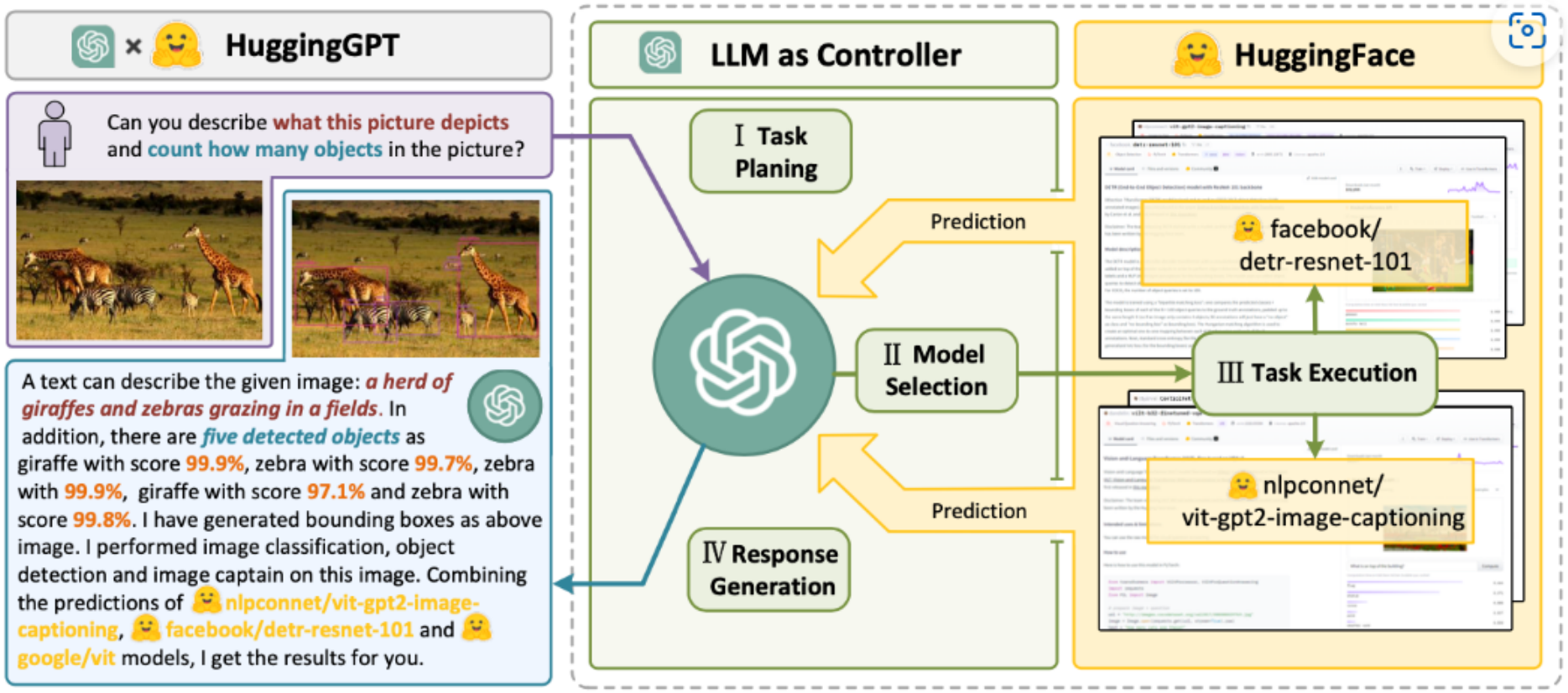
Fill in the blank: Steven was disgusted that Robert chose to cook pork for dinner, because _ was a vegetarian.
"option1": "Steven", "option2": "Robert"

ChatGPT

The correct answer is "Steven"

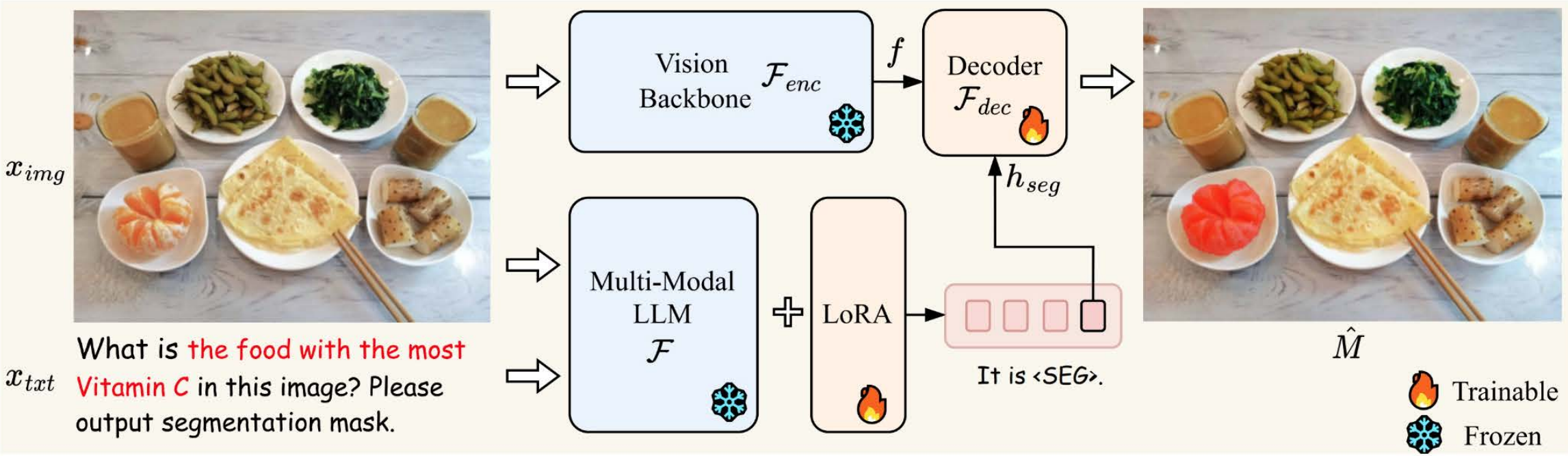
Examples of popular LLMs

❑ LLM Agents



Applications of LLMs

- Multi-Model LLM



Contents

- **Introduction to LLM**
  - Definition and significance
  - Key components and architecture
  - Examples of popular LLMs (e.g., GPT, BERT)

- **Applications of LLMs**
  - **Conversation AI**
  - **LLM Agents**
  - **Multi-Model LLM**

- **Training LLMs**
  - Data collection and preprocessing
  - Fine-tuning techniques
  - In-Context Learning

Training LLMs

- Data collection and preprocessing
  - What's the datasets for training LLM looks like?
    - JSON file.
    - Contain instruction, input, and output. (May different for different datasets, but it has similar format)
  - Example: Alpaca-GPT4 dataset


```
    {
      "instruction": "Compile a list of 5 US states located in the Mid West.",
      "input": "",
      "output": "1. Ohio\n2. Indiana\n3. Michigan\n4. Illinois\n5. Wisconsin"
    },
```

```
{
  "instruction": "Suggest a good restaurant.",
  "input": "Los Angeles, CA",
  "output": "If you're looking for a great restaurant experience in Los Angeles, CA, I would recommend you try Bestia.
},
```

Training LLMs

- Data collection and preprocessing
  - Data preprocessing
    - Adding prompt in front of “instruction”, different pre-trained LLM may have different default prompt.
    - For example, for Alpaca-GPT4 dataset, some instruction has input, and some may not.

Below is an instruction that describes a task. "
"Write a response that appropriately completes the request.

Below is an instruction that describes a task, paired with an input that provides further context. "
"Write a response that appropriately completes the request

Training LLMs

- Data collection and preprocessing
  - Data preprocessing
    - Adding prompt in front of “instruction”, different pre-trained LLM may have different default prompt.
    - For example, for Alpaca-GPT4 dataset, some instruction has input, and some may not.
    - Adding end of string token (EOS)
      - This token is essential because it tells the model when to stop producing text, for LLama models, EOS_TOKEN = "</s>"

Training LLMs

- Data collection and preprocessing
  - Data preprocessing

Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request.

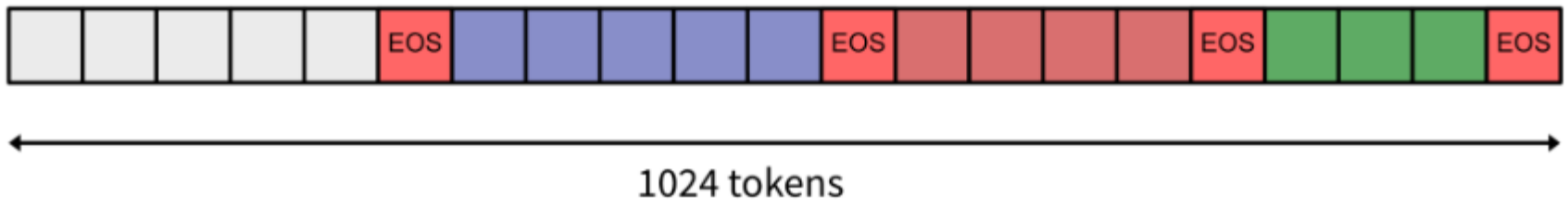
Instruction: Given a piece of text, suggest an appropriate title.

Input: "She was a kind hearted woman who always had a smile on her face."

Response: Title: "The Kind Hearted Woman"</s>

Training LLMs

- Data collection and preprocessing
  - Data preprocessing
  - Converting text to numbers: Tokenizer
    - It tokenizes the text (Byte Pair Encoding)
    - Converts the outputs to PyTorch tensors
    - Pads the inputs to match the length



Contents

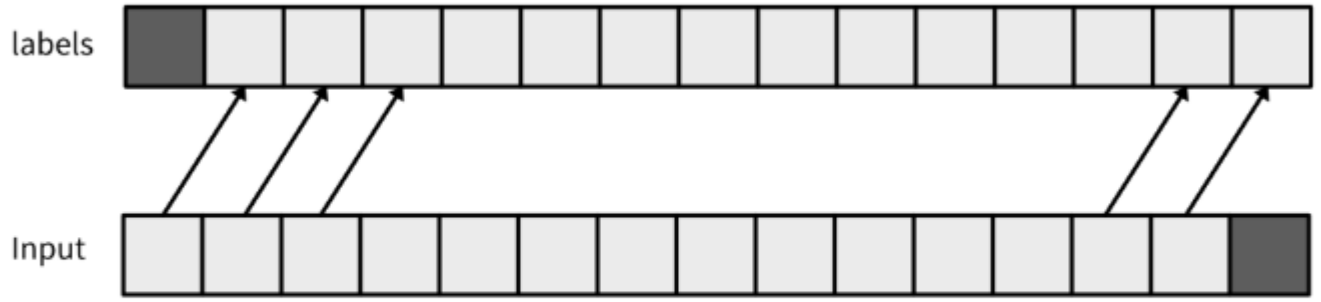
- **Introduction to LLM**
  - Definition and significance
  - Key components and architecture
  - Examples of popular LLMs (e.g., GPT, BERT)

- **Applications of LLMs**
  - **Conversation AI**
  - **LLM Agents**
  - **Multi-Model LLM**

- **Training LLMs**
  - Data collection and preprocessing
  - Training Process
  - In-Context Learning

Training LLMs

- Training Process
  - Architecture
    - Decoder-Only Transformer



as input and target are the same but shifted, we lose one token at each end

Training LLMs

- Training Process
  - Architecture
    - Decoder-Only Transformer
  - Training Loss: Cross-Entropy loss between the input and Labels
  - Testing: Using “generate” function of HuggingFace Transformer package

Let’s check more details for the Jupyter Notebook

Contents

- **Introduction to LLM**
  - Definition and significance
  - Key components and architecture
  - Examples of popular LLMs (e.g., GPT, BERT)

- **Applications of LLMs**
  - **Conversation AI**
  - **LLM Agents**
  - **Multi-Model LLM**

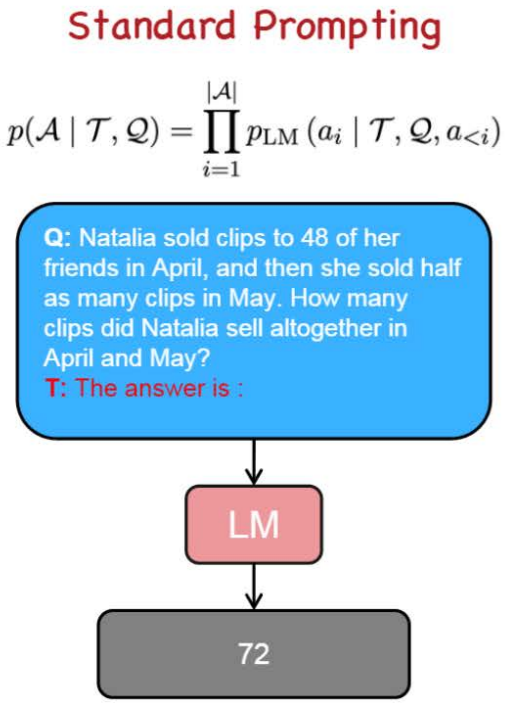
- **Training LLMs**
  - Data collection and preprocessing
  - Training Process
  - In-Context Learning

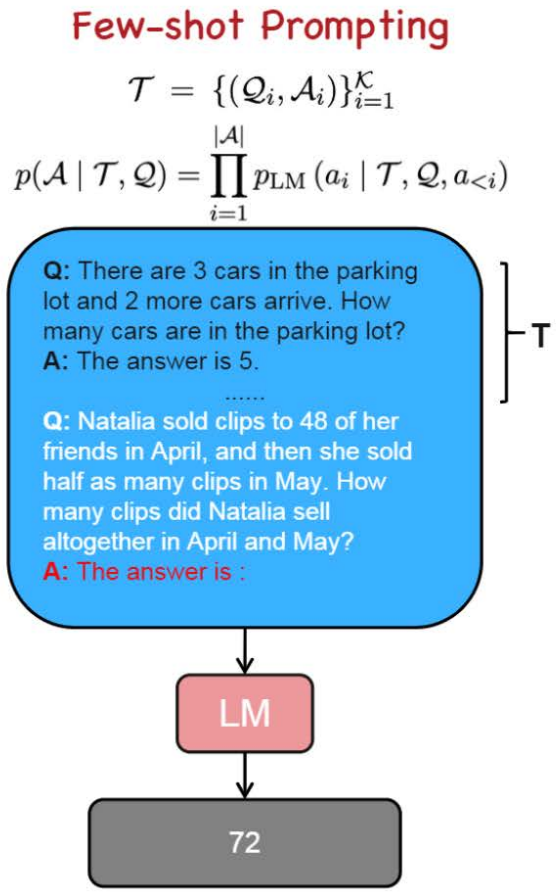
Training LLMs

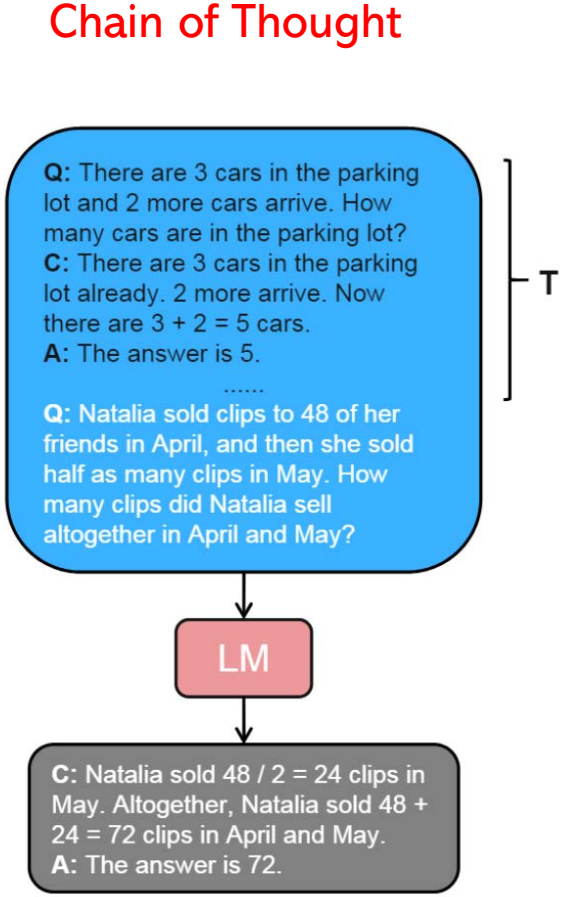
- In-Context Learning
  - Some LLMs is not open sourced (like ChatGPT), and We don't have enough computational resource or enough data to train the LLM.
  - How to improve the response of LLM and guide them to generate the correct answers.
  - In-Context Learning could help, and you don't update any weights of LLM, just like prompt learning to add some prompt questions and Answer pairs in front of new questions.

Training LLMs

- In-Context Learning
  - For example: Chain of Thought







Any Questions?

Section Break

Subtitle