Introduction to Agentic AI

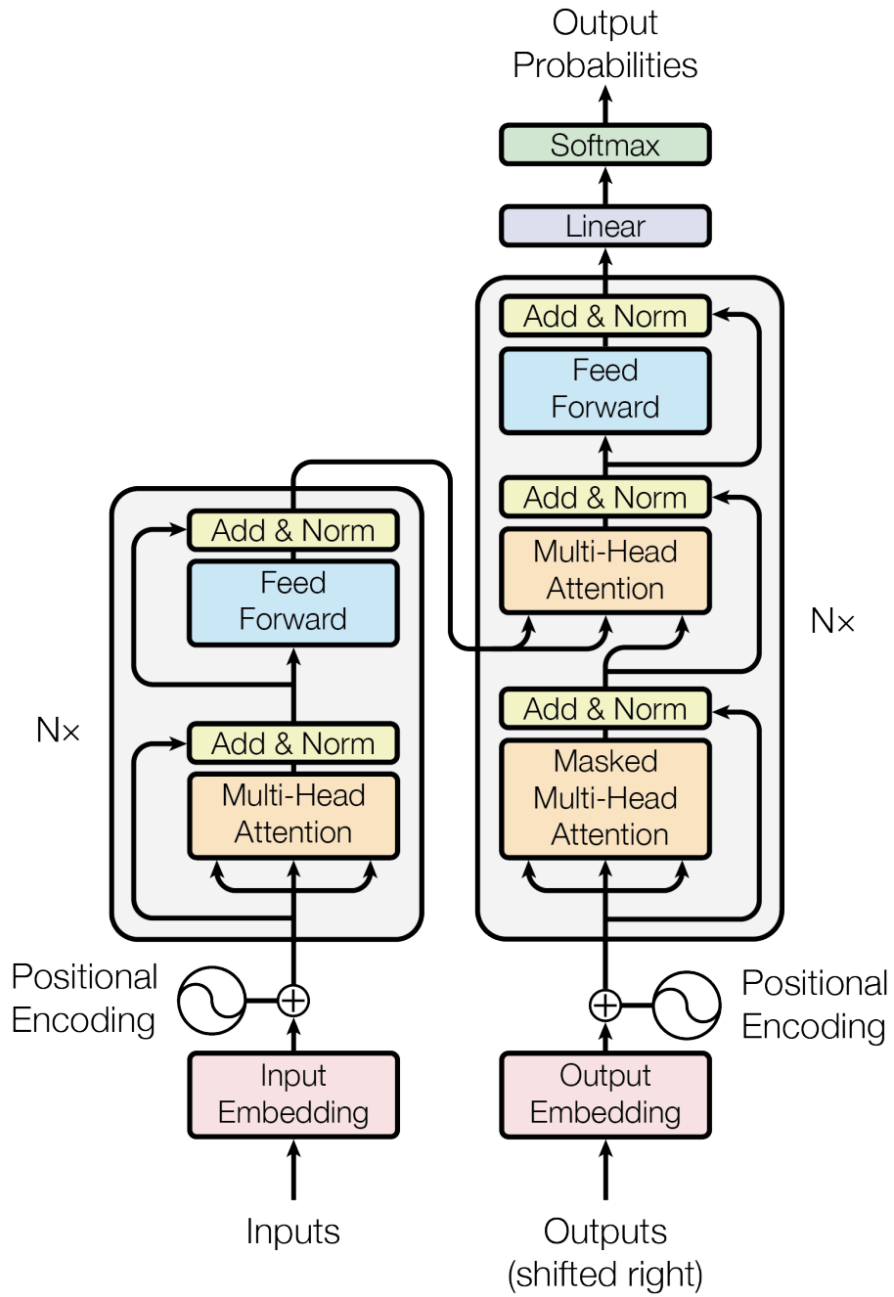
-- Language Modeling

Instructor: Guangjing Wang

guangjingwang@usf.edu

Last Lecture

- Cross-Attention
- Self-Attention
- Transformer



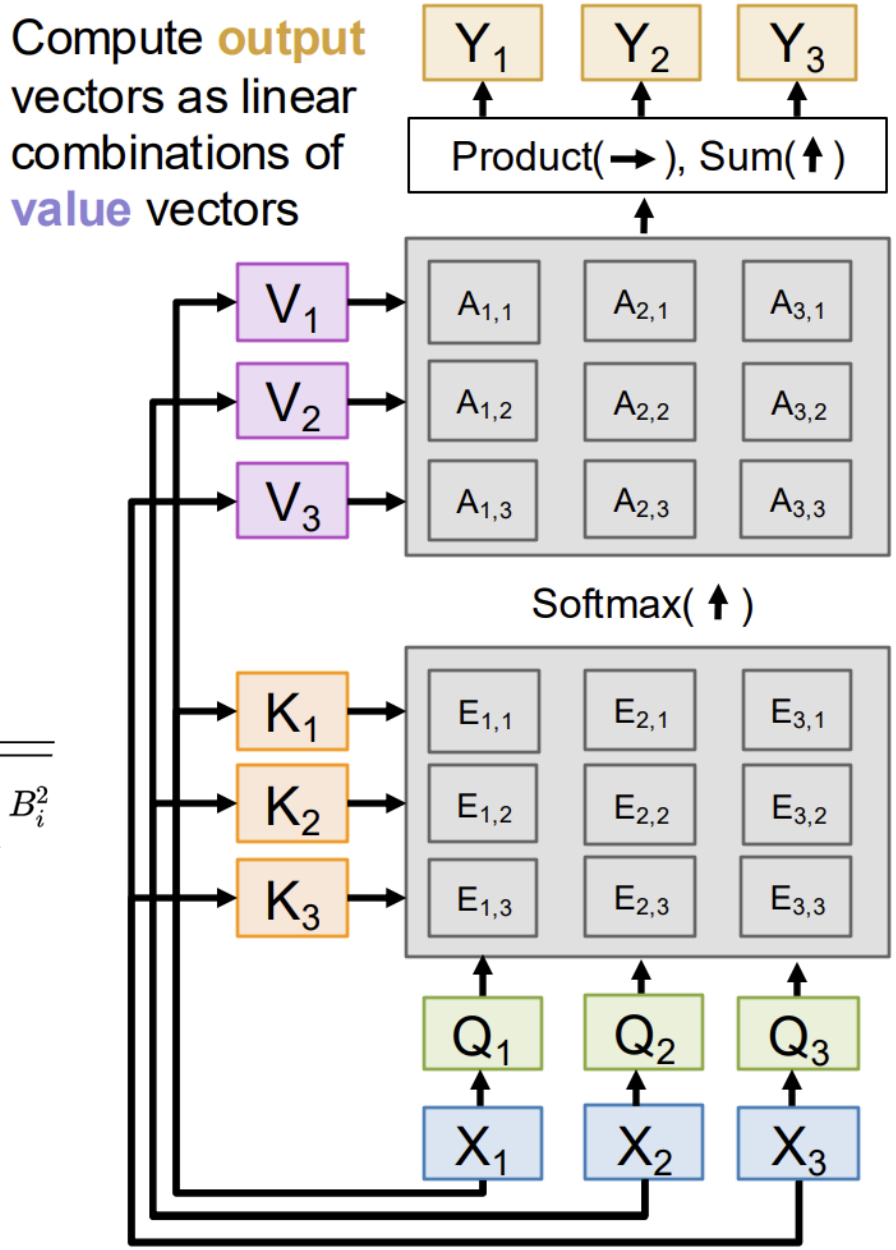
Self-Attention

Inputs:
Input vectors: X [N x D_{in}]
Key matrix: W_K [D_{in} x D_{out}]
Value matrix: W_V [D_{in} x D_{out}]
Query matrix: W_Q [D_{in} x D_{out}]

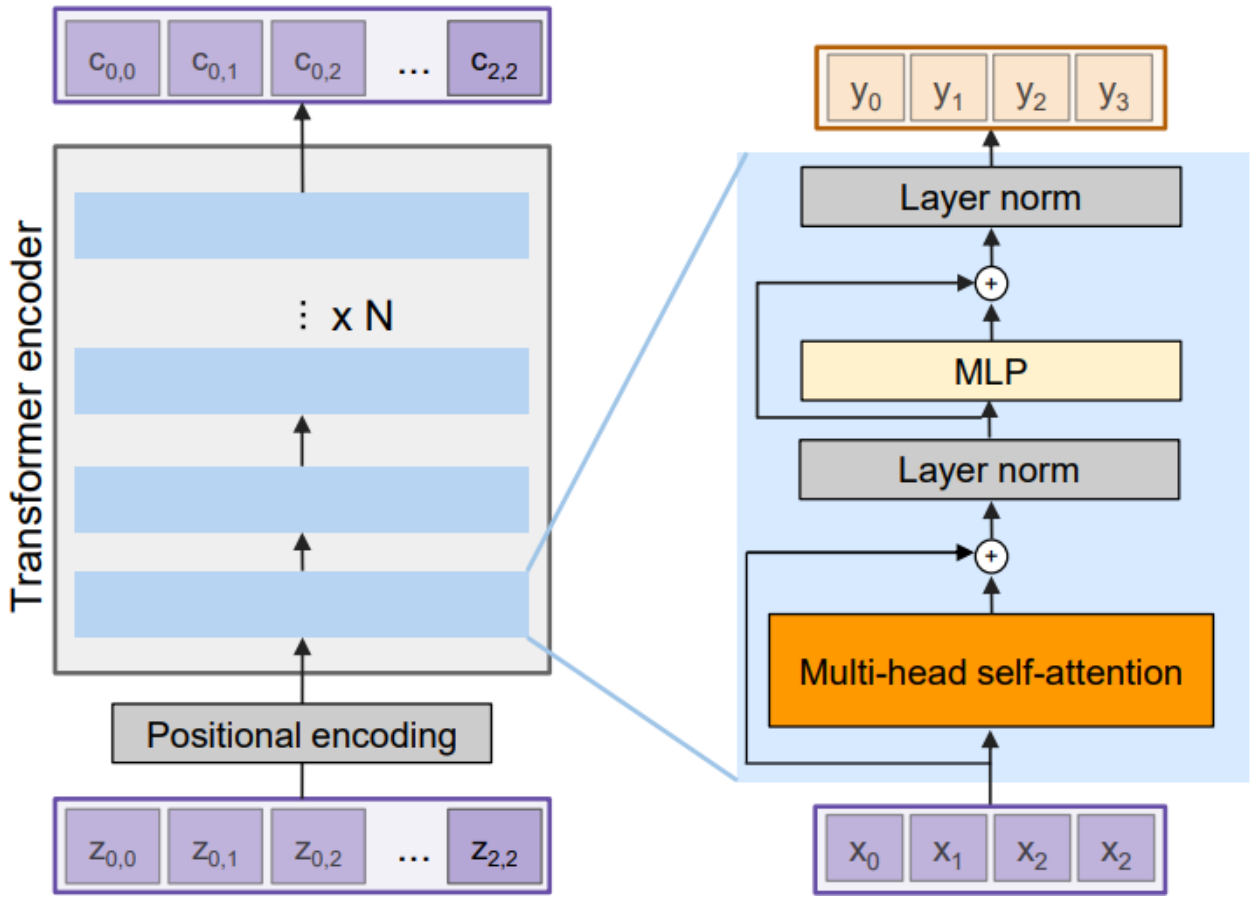
Each input produces one output, which is a mix of information from all inputs

Computation:
Queries: $Q = XW_Q$ [N x D_{out}]
Keys: $K = XW_K$ [N x D_{out}]
Values: $V = XW_V$ [N x D_{out}]
Similarities: $E = QK^T / \sqrt{D_Q}$ [N x N]
$E_{ij} = Q_i \cdot K_j / \sqrt{D_Q}$
Attention weights: A = softmax(E, dim=1) [N x N]
Output vector: $Y = AV$ [N x D_{out}]
$Y_i = \sum_j A_{ij} V_j$

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$



The Transformer encoder block



Transformer Encoder Block:

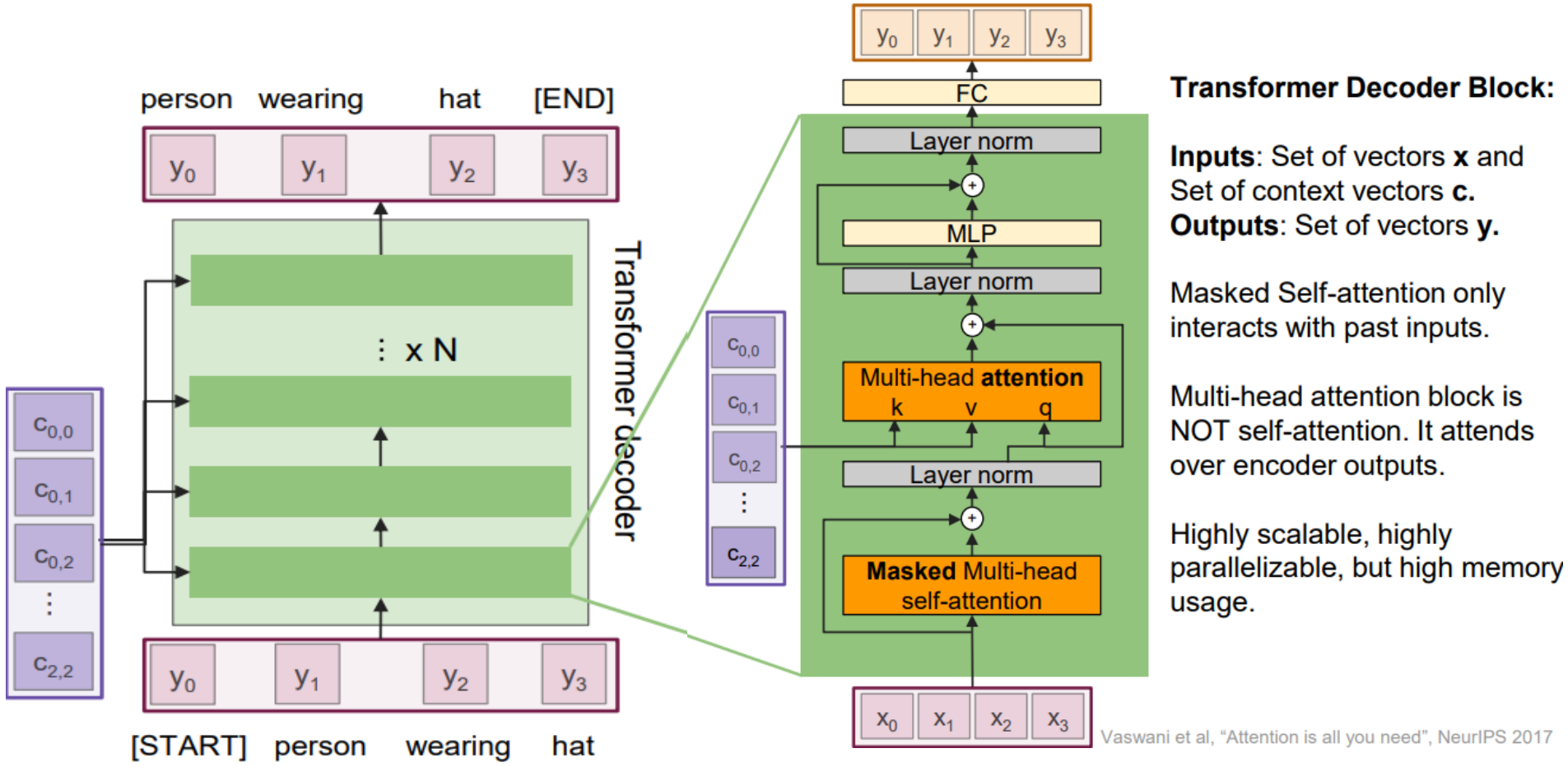
Inputs: Set of vectors **x**
Outputs: Set of vectors **y**

Self-attention is the only interaction between vectors.

Layer norm and MLP operate independently per vector.

Highly scalable, highly parallelizable, but high memory usage.

The Transformer decoder block



Transformer Decoder Block:

Inputs: Set of vectors **x** and Set of context vectors **c.**
Outputs: Set of vectors **y.**

Masked Self-attention only interacts with past inputs.

Multi-head attention block is NOT self-attention. It attends over encoder outputs.

Highly scalable, highly parallelizable, but high memory usage.

Vaswani et al, "Attention is all you need", NeurIPS 2017

Check: Rotary Positional Embedding (RoPE).
https://arxiv.org/abs/1904.09679

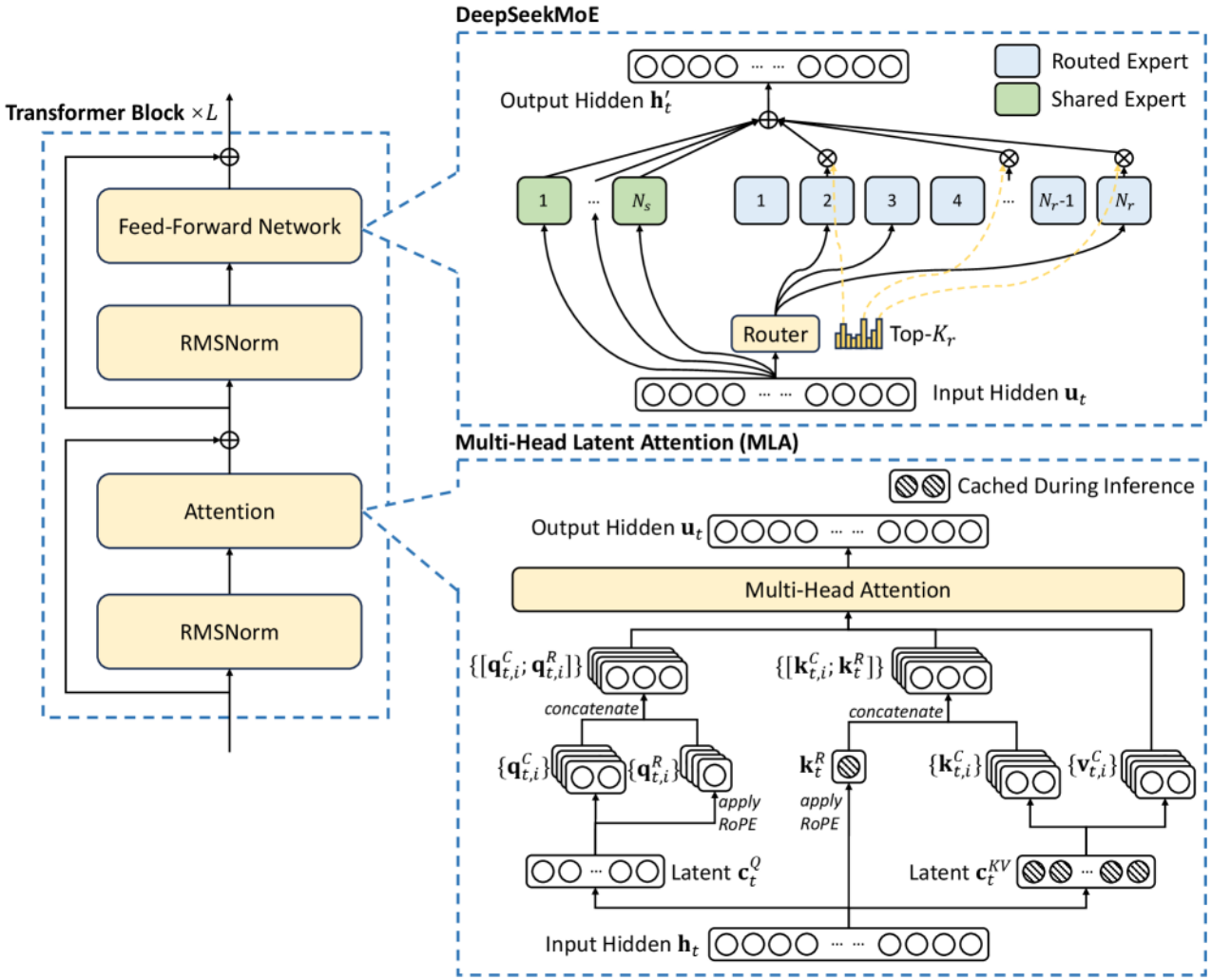
Transformer Variant: DeepSeek V3

RMSNorm: root mean square layer normalization

$$y_i = \frac{x_i}{\mathrm{RMS}(x)} * \gamma_i, \quad \text{where} \quad \mathrm{RMS}(x) = \sqrt{\epsilon + \frac{1}{n}\sum_{i=1}^{n} x_i^2}$$

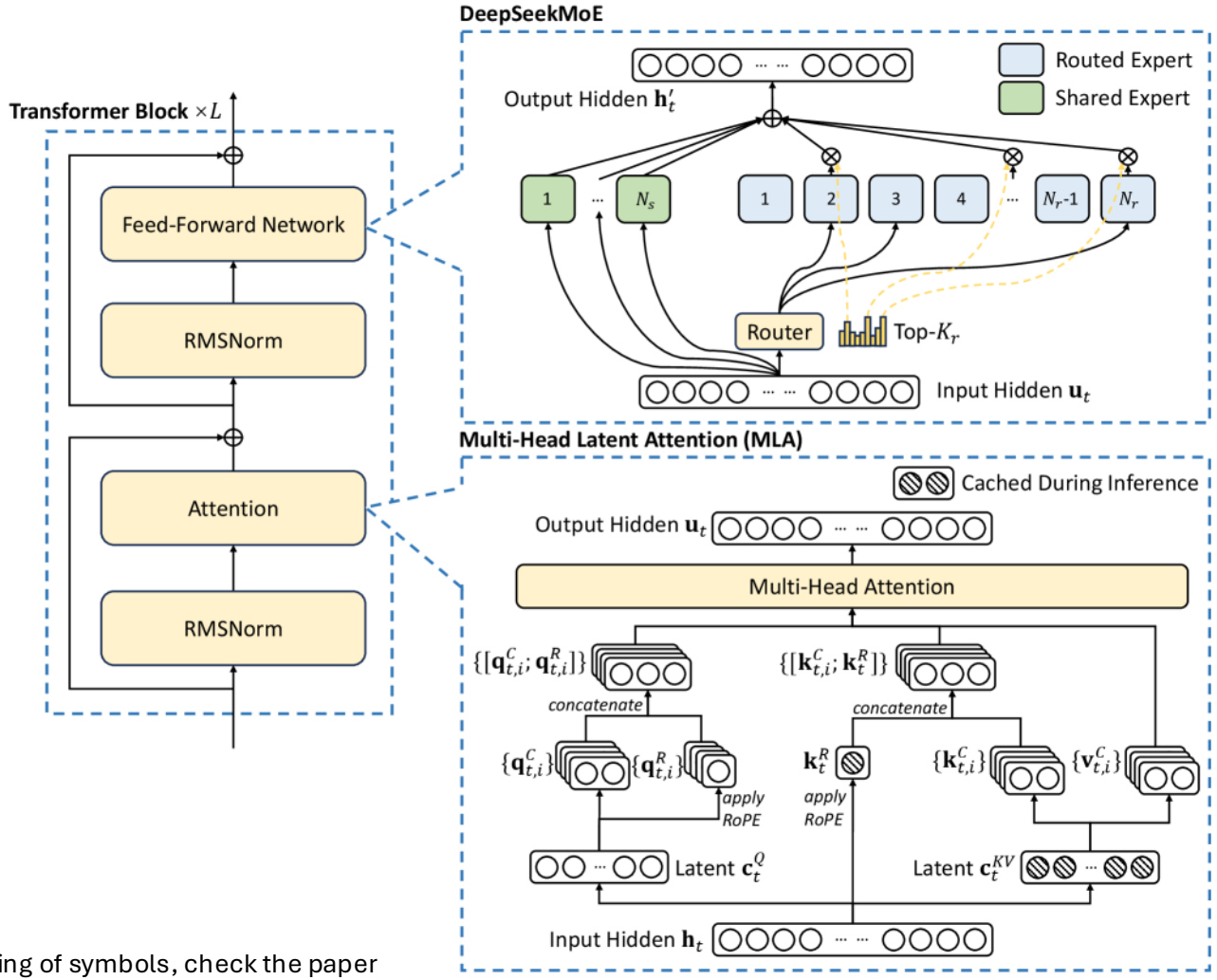
Query Vector

$$\begin{aligned}
\mathbf{c}_t^Q &= W^{DQ}\mathbf{h}_t, \\
[\mathbf{q}_{t,1}^C; \mathbf{q}_{t,2}^C; \ldots; \mathbf{q}_{t,n_h}^C] = \mathbf{q}_t^C &= W^{UQ}\mathbf{c}_t^Q, \\
[\mathbf{q}_{t,1}^R; \mathbf{q}_{t,2}^R; \ldots; \mathbf{q}_{t,n_h}^R] = \mathbf{q}_t^R &= \mathrm{RoPE}(W^{QR}\mathbf{c}_t^Q), \\
\mathbf{q}_{t,i} &= [\mathbf{q}_{t,i}^C; \mathbf{q}_{t,i}^R],
\end{aligned}$$



For the meaning of symbols, please check the paper
https://arxiv.org/html/2412.19437v1

Transformer Variant: DeepSeek V3



For the meaning of symbols, check the paper
https://arxiv.org/html/2412.19437v1

Value Vector

$$[\mathbf{v}^C_{t,1}; \mathbf{v}^C_{t,2}; \ldots; \mathbf{v}^C_{t,n_h}] = \mathbf{v}^C_t = W^{UV}\mathbf{c}^{KV}_t,$$

Key Vector

$$\mathbf{c}^{KV}_t = W^{DKV}\mathbf{h}_t,$$

$$[\mathbf{k}^C_{t,1}; \mathbf{k}^C_{t,2}; \ldots; \mathbf{k}^C_{t,n_h}] = \mathbf{k}^C_t = W^{UK}\mathbf{c}^{KV}_t,$$

$$\mathbf{k}^R_t = \mathrm{RoPE}(W^{KR}\mathbf{h}_t),$$

$$\mathbf{k}_{t,i} = [\mathbf{k}^C_{t,i}; \mathbf{k}^R_t],$$

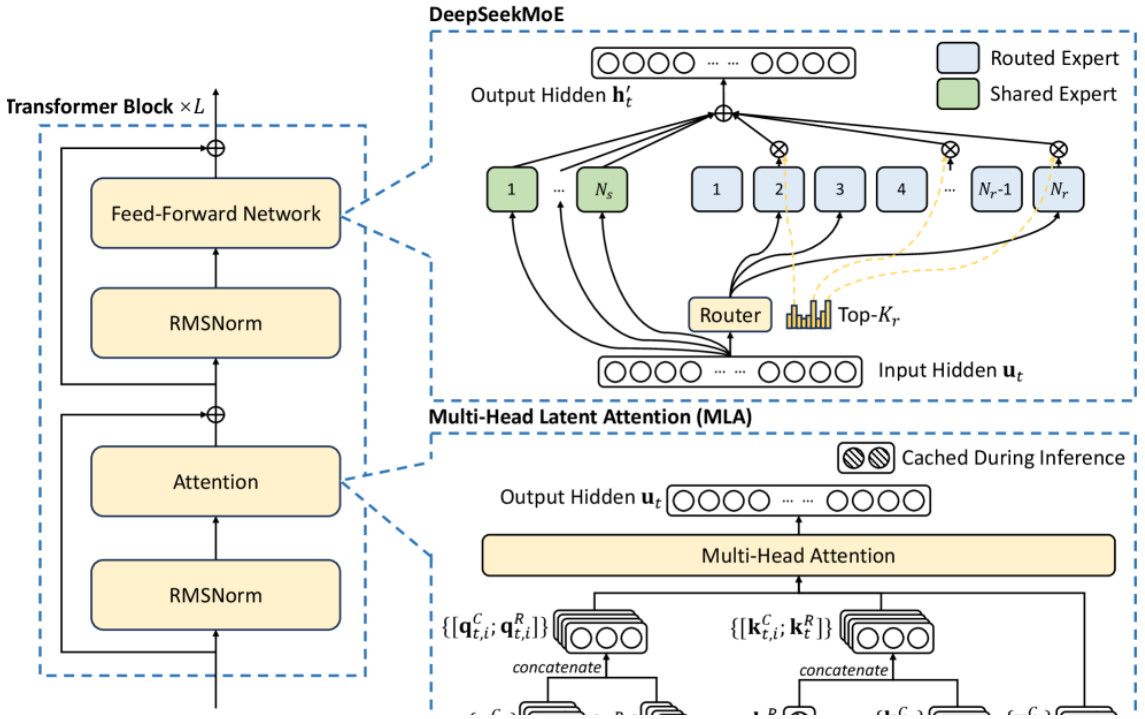
DeepSeekMoE: Mixture of Expert

$$\mathbf{h}_t^{'} = \mathbf{u}_t + \sum_{i=1}^{N_s} \mathrm{FFN}_i^{(s)}(\mathbf{u}_t) + \sum_{i=1}^{N_r} g_{i,t}\,\mathrm{FFN}_i^{(r)}(\mathbf{u}_t),$$

$$g_{i,t} = \frac{g_{i,t}^{'}}{\sum_{j=1}^{N_r} g_{j,t}^{'}},$$

$$g_{i,t}^{'} = \begin{cases} s_{i,t}, & s_{i,t} \in \mathrm{Topk}(\{s_{j,t} | 1 \leqslant j \leqslant N_r\}, K_r), \\ 0, & \text{otherwise}, \end{cases}$$

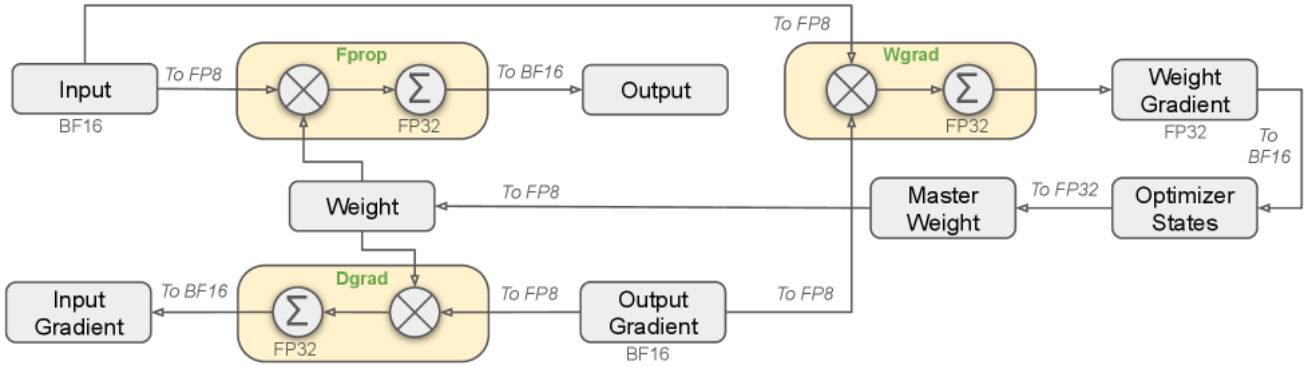
$$s_{i,t} = \mathrm{Sigmoid}\left(\mathbf{u}_t^{T} \mathbf{e}_i\right),$$



where N_s and N_r denote the numbers of shared experts and routed experts, respectively; $\mathrm{FFN}_i^{(s)}(\cdot)$ and $\mathrm{FFN}_i^{(r)}(\cdot)$ denote the i-th shared expert and the i-th routed expert, respectively; K_r denotes the number of activated routed experts; $g_{i,t}$ is the gating value for the i-th expert; $s_{i,t}$ is the token-to-expert affinity; $\mathbf{e}_i$ is the centroid vector of the i-th routed expert; and $\mathrm{Topk}(\cdot, K)$ denotes the set comprising K highest scores among the affinity scores calculated for the t-th token and all routed experts. Slightly different from DeepSeek-V2, DeepSeek-V3 uses the sigmoid function to compute the affinity

Pre-training of DeepSeek-V3 (671B)

- 2048 NVIDIA H800 GPUs for 3.7 days (180K H800 GPU hours) **per trillion token**
  - GPUs connected by NVLink and NVSwitch within nodes
  - InfiniBand (IB) interconnects are utilized to facilitate communications across nodes
- Low-precision training: FP8 (8-bit floating point).



- Pre-train DeepSeek-V3 on 14.8 trillion diverse and high-quality tokens.
- Total training cost $ 5.576 million, if H800 GPU is $2 per GPU hour.

This Lecture

- Masked language modeling
- Causal language modeling
- Sequence to sequence modeling
- LLM inference process
- Evaluation and Optimization Inference

Language Modeling

- LM: probability distribution over sequence of tokens

Step	Token (wi)	Context (History)	Conditional Probability	Estimated Value
1	The	[Start]	P("The")	0.1
2	cat	The	P("cat"|"The")	0.05
3	sat	The cat	P("sat"|"The cat")	0.2
4	.	The cat sat	P("."|"The cat sat")	0.8

$$P(\text{"The cat sat."}) = 0.10 \times 0.05 \times 0.20 \times 0.80$$

- **Autoregressive** language models:

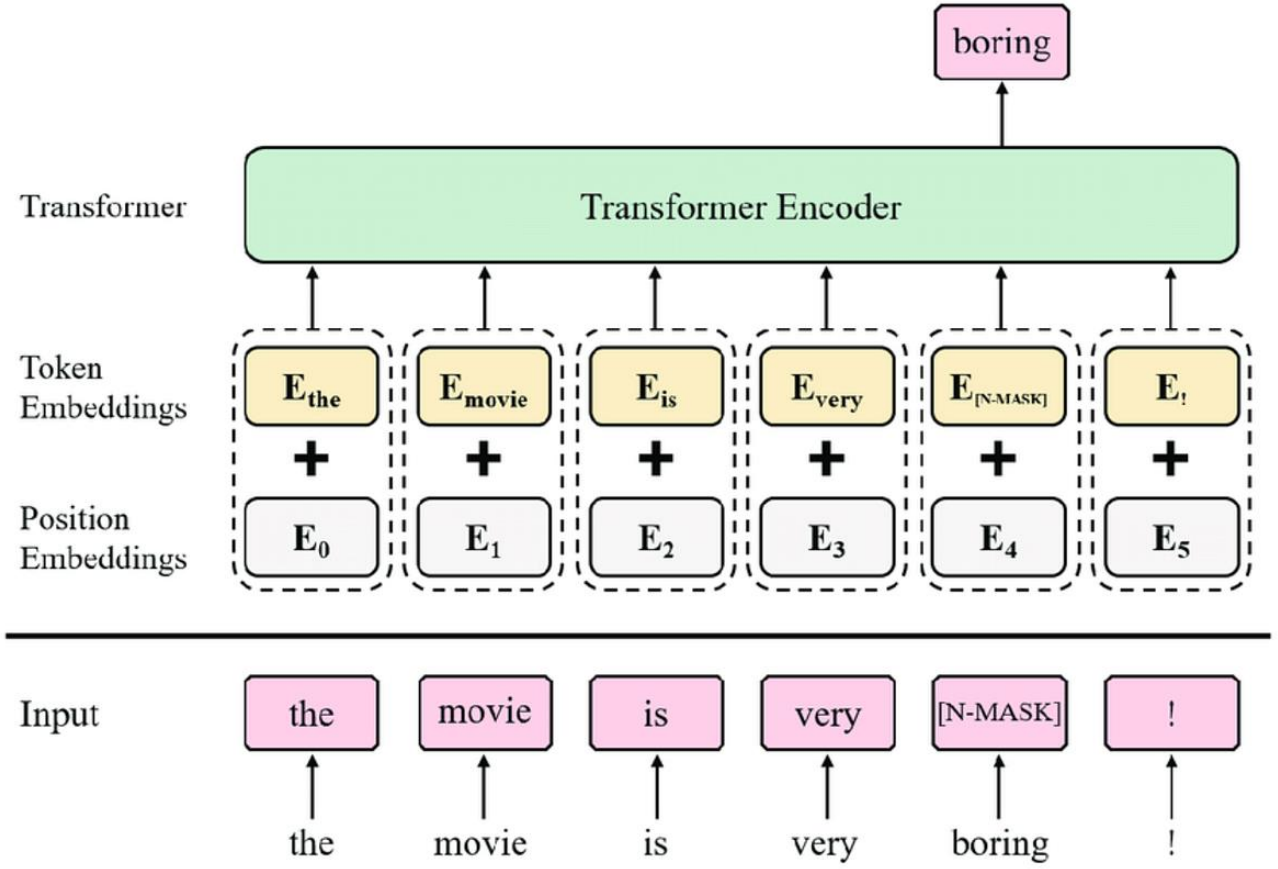
$$P(w_1, w_2, ..., w_n) = \prod_{i=1}^{n} P(w_i | w_1, ..., w_{i-1})$$

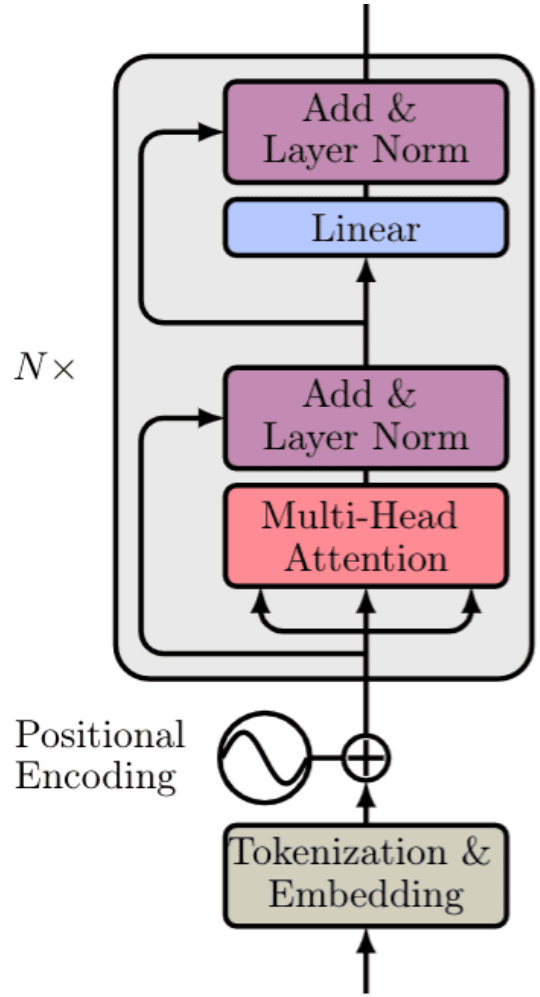
Masked Language Modeling

- Language models work by being trained to **predict the probability of a word** given **the context of surrounding words.**

- **Masked language modeling (MLM):** randomly masks some tokens in the input and trains the model to predict the original tokens based on the surrounding context.
  - Bidirectional context (looking at words both before and after the masked word)
  - Model example: BERT
  - Tasks such as classification, named entity recognition, and question answering

Masked Language Modeling



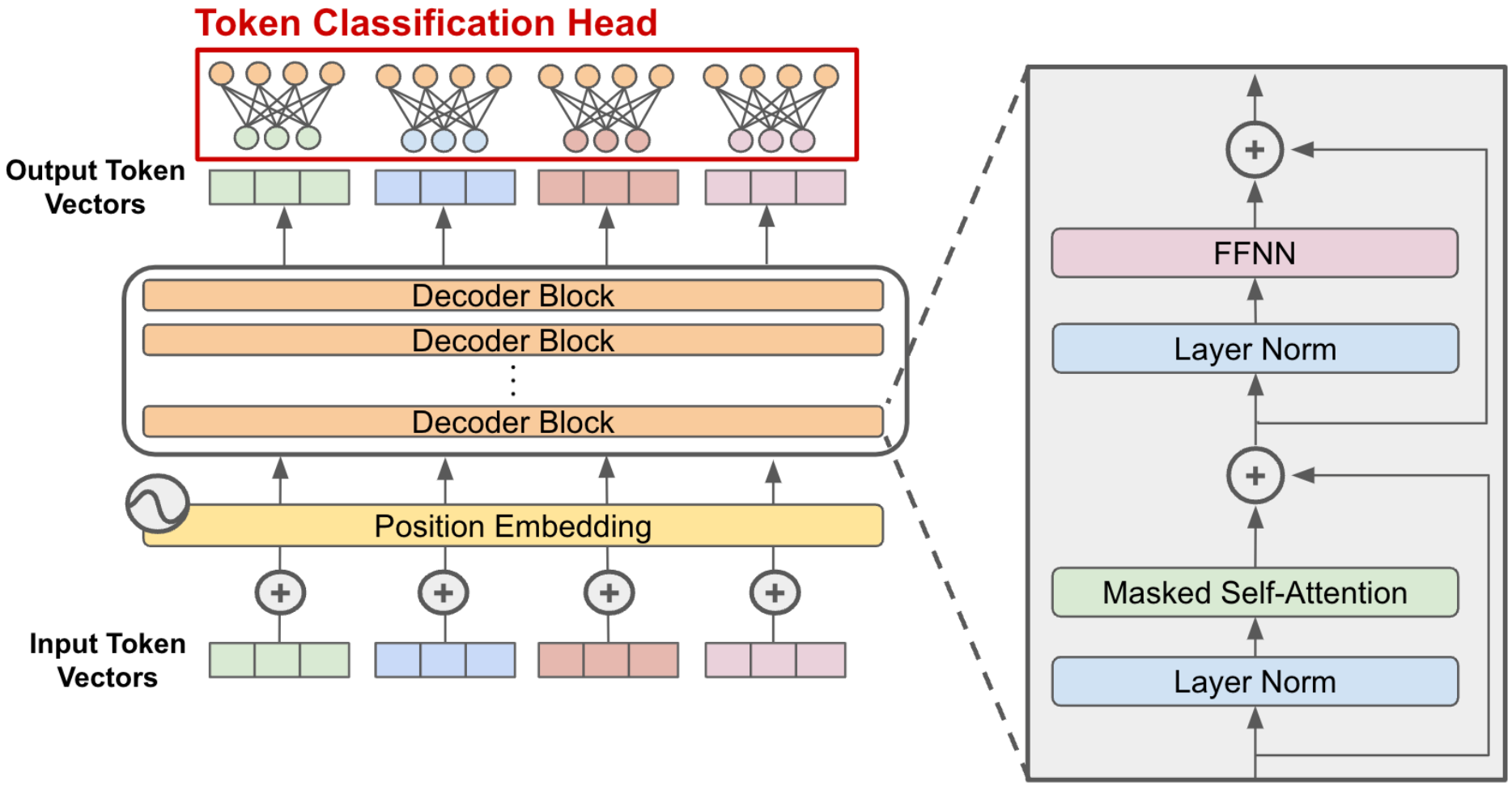


Causal Language Modeling

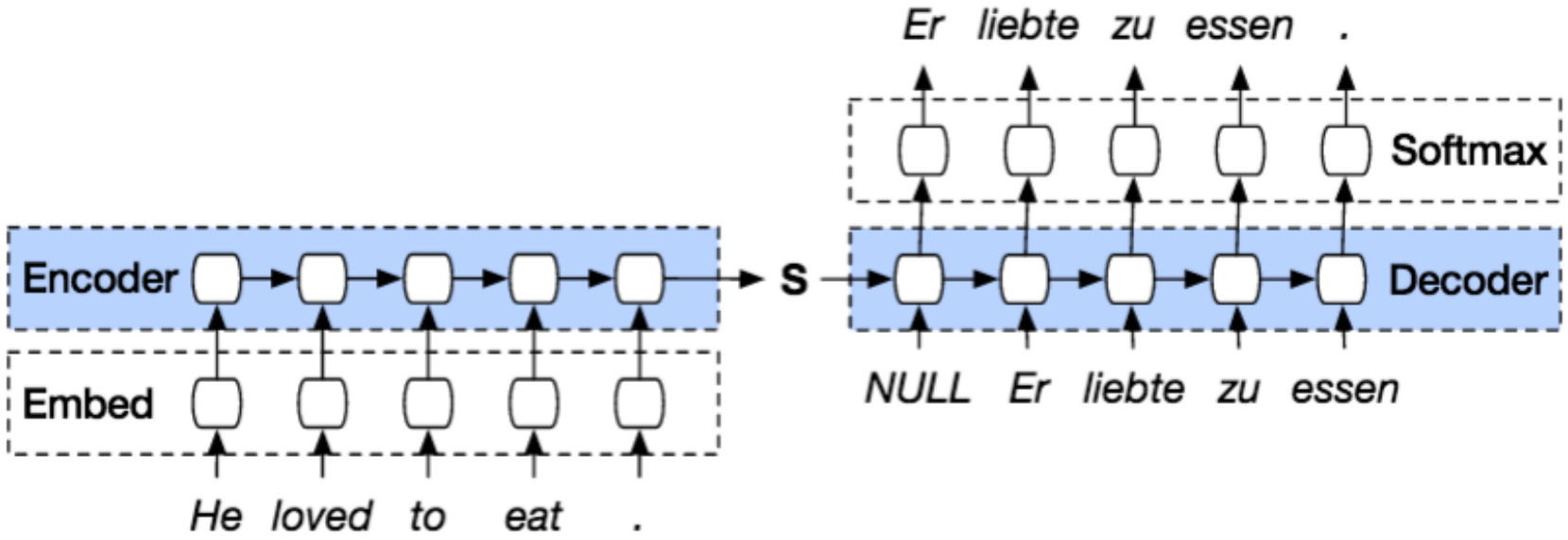
- **Causal language modeling (CLM):** predicts the next token based on all previous tokens in the sequence.

- The model can only use context from the left (previous tokens) to predict the next token.
  - Model example: GPT, Llama
  - They can complete sentences, write essays, or generate code based on a prompt.

Causal Language Modeling



Sequence to Sequence Modeling



Models such as T5, BART use an encoder to understand the input and a decoder to generate output. Tasks such as translation, summarization, and question answering

Prefill Phase in Inference Process:

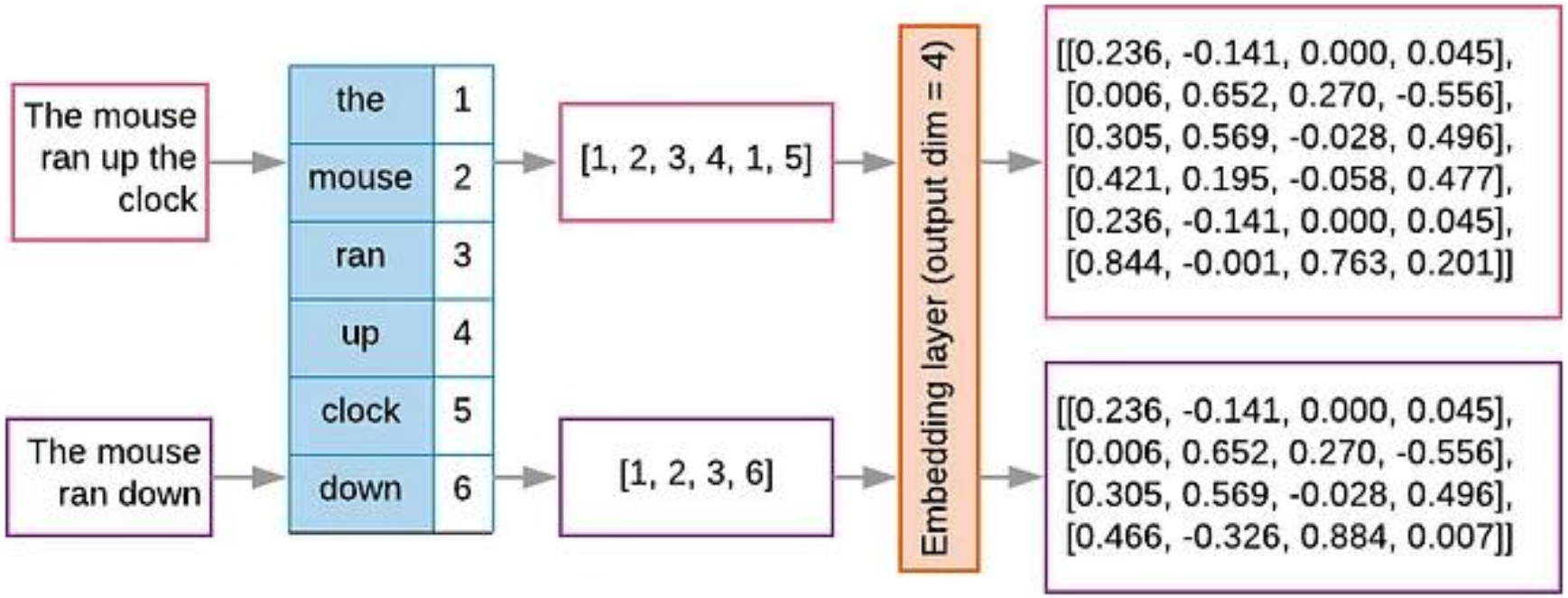
1. **Tokenization**: Converting the input text into tokens (think of these as the basic building blocks the model understands)

2. **Embedding Conversion**: Transforming these tokens into numerical representations that capture their meaning

3. **Initial Processing**: Running these embeddings through the model's neural networks to create a rich understanding of the context

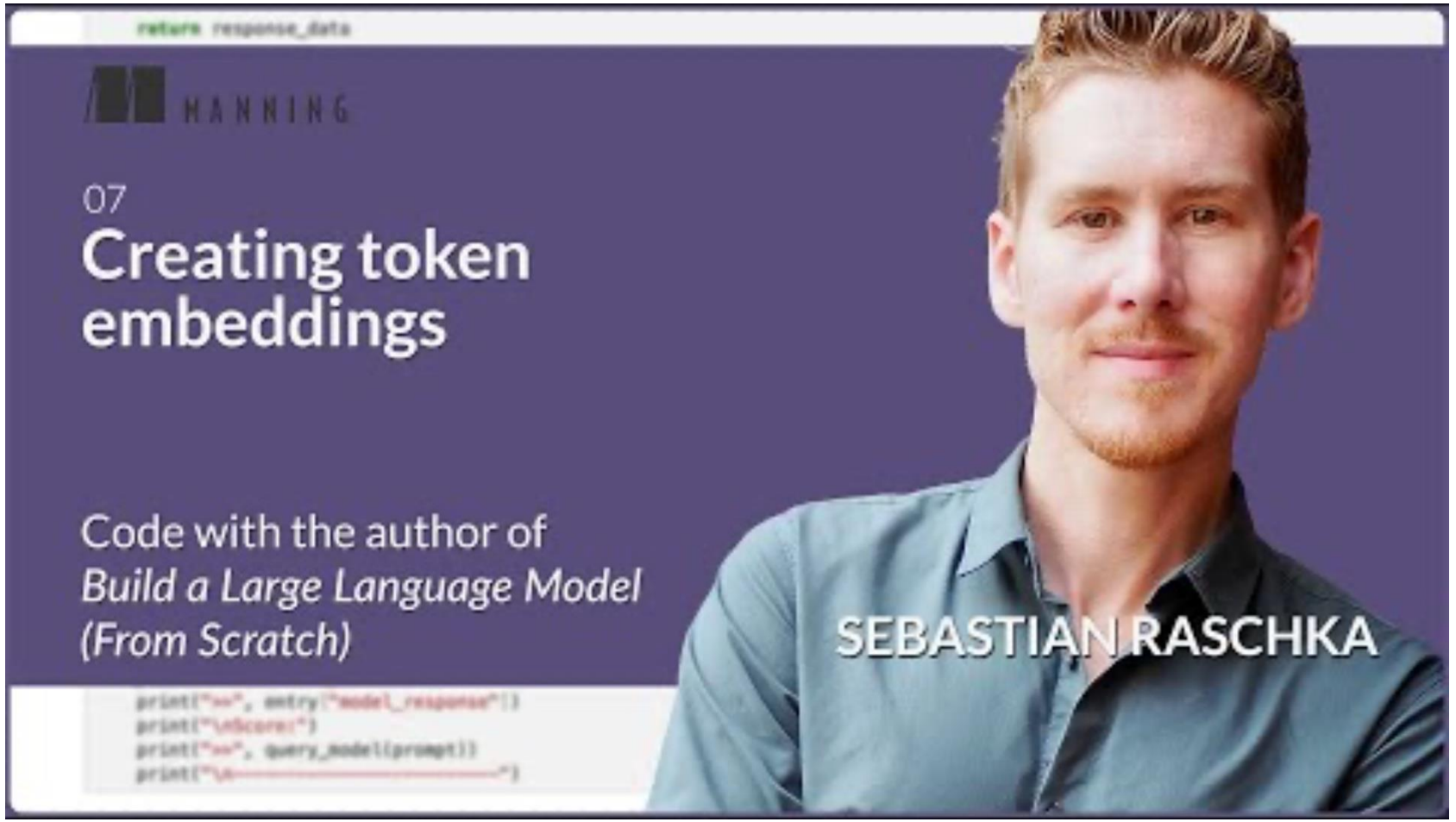
Tokenization

- Splitting the input into words, subwords, or symbols (like punctuation) that are called ***tokens***



https://airbyte.com/data-engineering-resources/tokenization-vs-embeddings

Creating Token Embedding During Model Training



https://www.youtube.com/watch?v=x2tYF_DHpgc
https://github.com/rasbt/LLMs-from-scratch

Decoding Phase in Inference Process:

1. **Attention Computation**: Looking back at all previous tokens to understand context

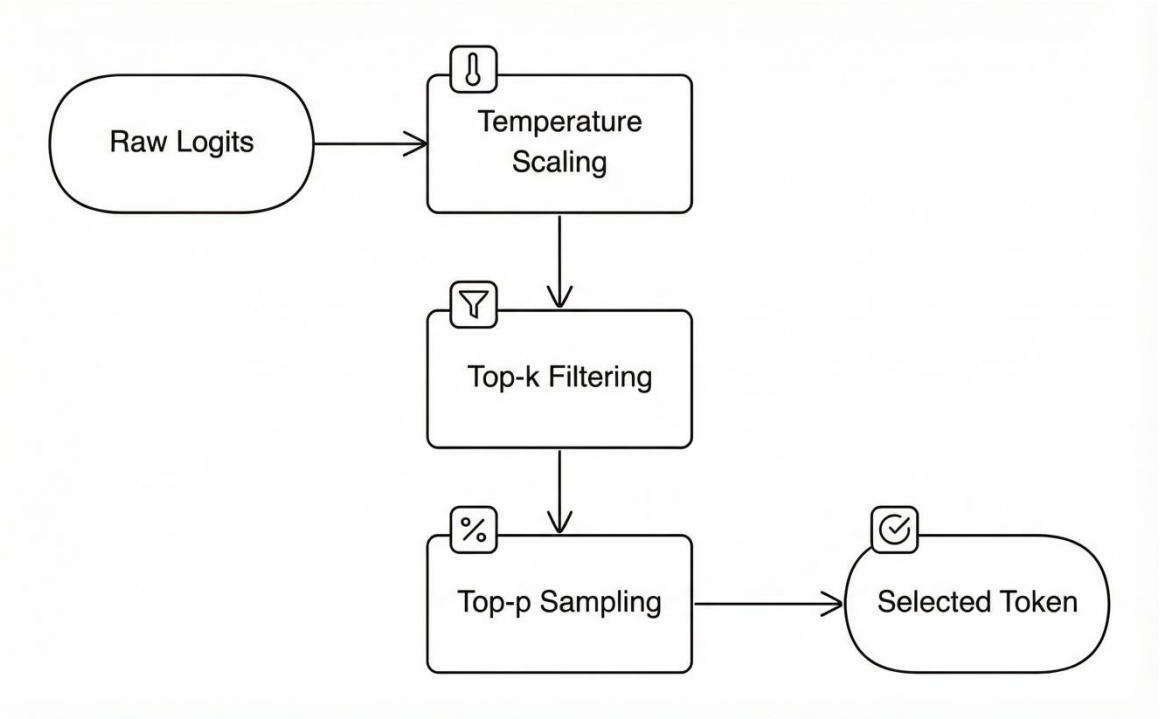
2. **Probability Calculation**: Determining the likelihood of each possible next token

3. **Token Selection**: Choosing the next token based on these probabilities

4. **Continuation Check**: Deciding whether to continue or stop generation

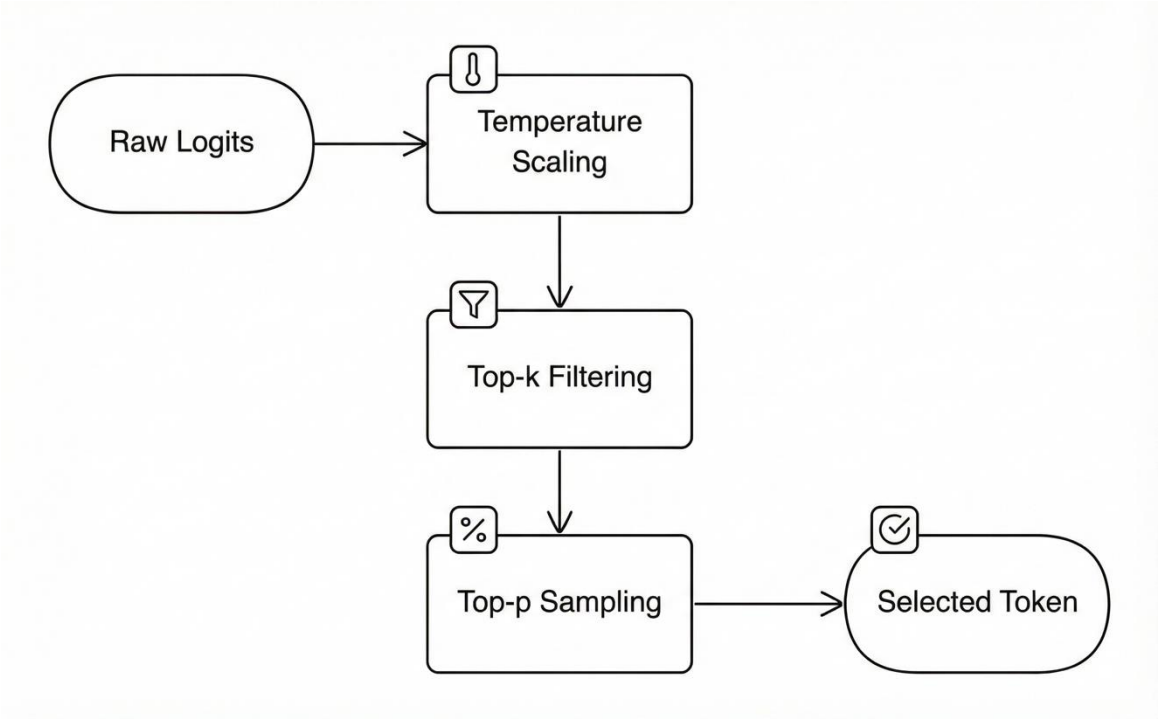
Token Selection: Probabilities to Token Choices

- **Raw Logits**: Think of these as the model's initial gut feelings about each possible next word.
- **Temperature Control**: Like a creativity dial - higher settings (>1.0) make choices more random and creative, lower settings (<1.0) make them more focused and deterministic.

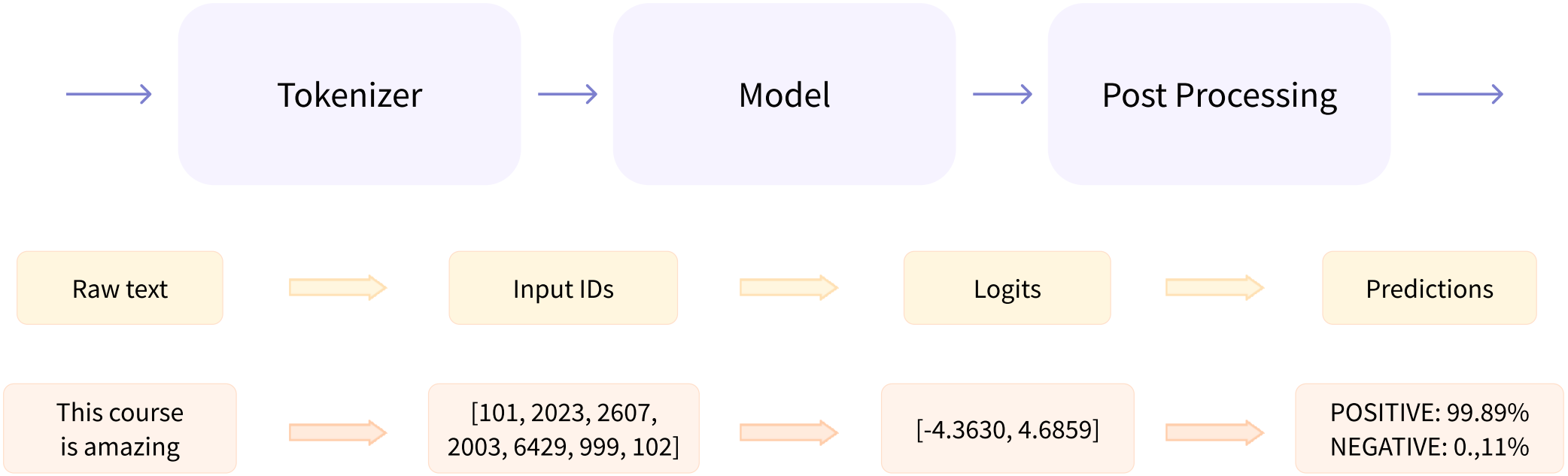


Token Selection: Probabilities to Token Choices

- **Top-p (Nucleus) Sampling**: Instead of considering all possible words, we only look at the most likely ones that add up to our chosen probability threshold (e.g., top 90%).
- **Top-k Filtering**: An alternative approach where we only consider the k most likely next words.



Full LLM Pipeline in Inference



https://huggingface.co/learn/llm-course/chapter2/2

Key Metrics

- **Time to First Token (TTFT)**: How quickly can you get the first response? This is crucial for user experience and is primarily affected by the prefill phase.
- **Time Per Output Token (TPOT)**: How fast can you generate subsequent tokens? This determines the overall generation speed.
- **Throughput**: How many requests can you handle simultaneously? This affects scaling and cost efficiency.
- **VRAM Usage**: How much GPU memory do you need? This often becomes the primary constraint in real-world applications.

Attention and Context Length

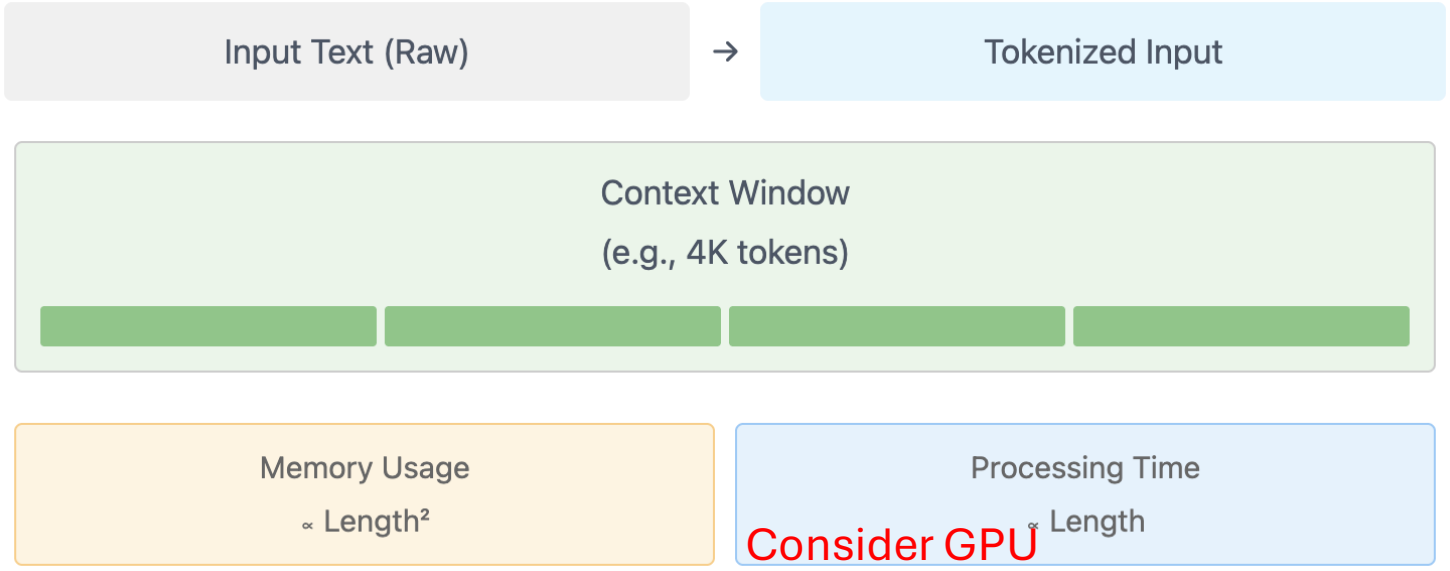
- The attention mechanism is what gives LLMs their ability to understand context and focus on relevant information.
  - **Remember the similarity score and attention weights in last lecture?**
  - When predicting the next word, not every word in a sentence carries equal weight.

- The context length refers to the maximum number of tokens (words or parts of words) that the LLM can process at once.
  - https://arxiv.org/html/2412.19437v1#S4 (Long Context Extension)

- The context is the LLM working memory, limited by **model architecture and size, computational resources, and the complexity of input and the desired output.**

Context Length Management in Inference

- **Memory Usage**: Grows quadratically with context length
- **Processing Speed**: Decreases linearly with longer contexts
- **Resource Allocation**: Requires careful balancing of VRAM usage



One solution: KV Cache; e.g., PagedAttention https://arxiv.org/abs/2309.06180

Building Large Language Models

https://www.youtube.com/watch?v=7xTGNNLPyMI

Another Suggested Video

https://www.youtube.com/watch?v=9vM4p9NN0Ts

References

- https://huggingface.co/learn/llm-course/chapter2/1
- https://arxiv.org/html/2412.19437v1
- https://huggingface.co/spaces/hesamation/primer-llm-embedding?section=what_are_embeddings?