THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

RELAXED SYSTEM LAB

Generative Inference

COMP4901Y

Binhang Yuan

Recall Language Modeling

What Is a Language Model?

- The classic definition of a ***language model (LM)*** is a probability distribution over sequences of tokens.
- Suppose we have a vocabulary $\mathcal{V}$ of a set of tokens.
- A language model P assigns each sequence of tokens $x_1, x_2, \ldots, x_L \in \mathcal{V}$ to a probability (a number between 0 and 1): $p(x_1, x_2, \ldots, x_L) \in [0,1]$.
- The probability intuitively tells us how "good" a sequence of tokens is.
  - For example, if the vocabulary is $\mathcal{V} = \{\text{ate}, \text{ball}, \text{cheese}, \text{mouse}, \text{the}\}$, the language model might assign:

$$p(\text{the}, \text{mouse}, \text{ate}, \text{the}, \text{cheese}) = 0.02$$
$$p(\text{the}, \text{cheese}, \text{ate}, \text{the}, \text{mouse}) = 0.01$$
$$p(\text{mouse}, \text{the}, \text{the}, \text{chesse}, \text{ate}\ ) = 0.0001$$

Decoder-only Models

- Decoder-only models are our standard autoregressive language models.
- Given a prompt $x_{1:i}$ produces both contextual embeddings and a distribution over next tokens x_{i+1}, and recursively, over the entire completion $x_{i+1:L}$:

$$x_{1:i} \Rightarrow \emptyset(x_{1:i}), p(x_{i+1}|x_{1:i})$$

- Example: text autocomplete
  - [[CLS],the,movie,was] ⇒ great
- The probabily $p(x_{i+1}|x_{1:i})$ is usually determined by:

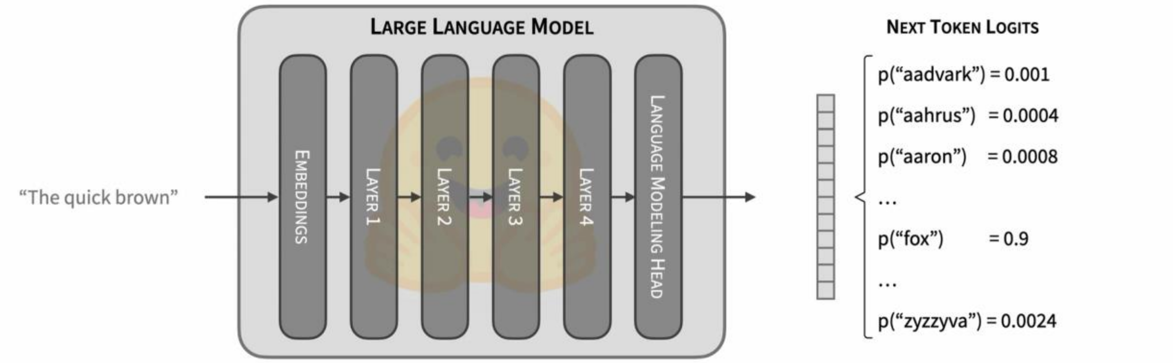
$$p(x_{i+1}|x_{1:i}) = \mathrm{softmax}(x_i W_{lm}), x_i \in \mathbb{R}^D, W_{lm} \in \mathbb{R}^{D\times|\mathcal{V}|}$$

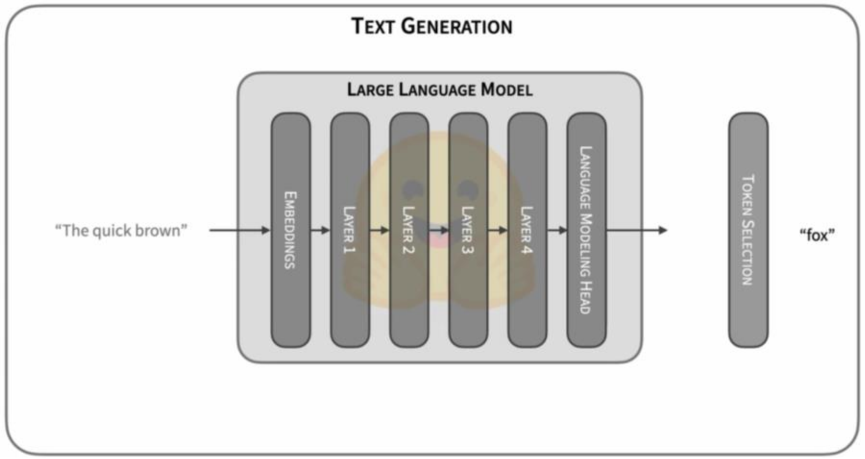
Autoregressive Generation

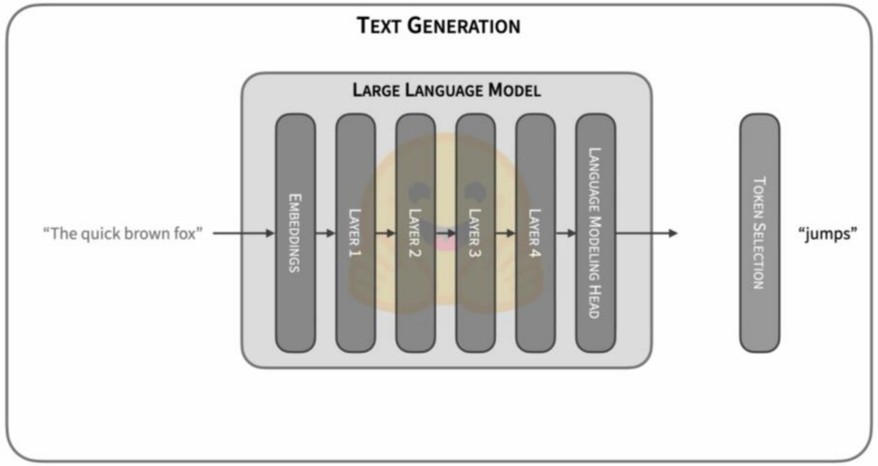
Autoregressive Generation

- LLM takes a sequence of text tokens as input and returns the probability distribution for the next token.
- A critical aspect of autoregressive generation with LLMs is *how to select the next token from this probability distribution*.
- There are many ways in this step as long as you end up with a token for the next iteration.
- The simplest way is to **select the most likely token from the probability distribution.**
- More complex solutions, e.g., applying a dozen transformations before sampling from the resulting distribution.

Autoregressive Generation







The quick brown => fox

Step 1

The quick brown fox => jumps

Step 2

Naïve Implementation

RELAXED
SYSTEM LAB

TransformerBlocks$(x \in R^{L\times D}) \rightarrow x' \in \mathbb{R}^{L\times D}$

RELAXED SYSTEM LAB

For each inference request:

- ~~$B = 1$;~~
- L is the input sequence length;
- D is the model dimension;
- Multi-head attention: $D = n_H \times H$
- H is the head dimension;
- n_h is the number of heads.

Generate the first token.

$$p(x_{L+1}|x_{1:L}) = \text{softmax}(x_L W_{lm})$$

Computation	Input	Output
$Q = xW^Q$	$x \in \mathbb{R}^{L\times D}$, $W^Q \in \mathbb{R}^{D\times D}$	$Q \in \mathbb{R}^{L\times D}$
$K = xW^K$	$x \in \mathbb{R}^{L\times D}$, $W^K \in \mathbb{R}^{D\times D}$	$K \in \mathbb{R}^{L\times D}$
$V = xW^V$	$x \in \mathbb{R}^{L\times D}$, $W^V \in \mathbb{R}^{D\times D}$	$V \in \mathbb{R}^{L\times D}$
$[Q_1, Q_2 \dots, Q_{n_h}] = \text{Partition}_{-1}(Q)$	$Q \in \mathbb{R}^{L\times D}$	$Q_i \in \mathbb{R}^{L\times H}, i = 1, \dots n_h$
$[K_1, K_2 \dots, K_{n_h}] = \text{Partition}_{-1}(K)$	$K \in \mathbb{R}^{L\times D}$	$K_i \in \mathbb{R}^{L\times H}, i = 1, \dots n_h$
$[V_1, V_2 \dots, V_{n_h}] = \text{Partition}_{-1}(V)$	$V \in \mathbb{R}^{L\times D}$	$V_i \in \mathbb{R}^{L\times H}, i = 1, \dots n_h$
$\text{Score}_\text{i} = \text{softmax}(\frac{Q_i K_i^T}{\sqrt{D}}), i = 1, \dots n_h$	$Q_i, K_i \in \mathbb{R}^{L\times H}$	$\text{score}_\text{i} \in \mathbb{R}^{L\times L}$
$Z_i = \text{score}_i V_i, i = 1, \dots n_h$	$\text{score}_\text{i} \in \mathbb{R}^{L\times L}$, $V_i \in \mathbb{R}^{L\times H}$	$Z_i \in \mathbb{R}^{L\times H}$
$Z = \text{Merge}_{-1}([Z_1, Z_2 \dots, Z_{n_h}])$	$Z_i \in \mathbb{R}^{L\times H}, i = 1, \dots n_h$	$Z \in \mathbb{R}^{L\times D}$
$\text{Out} = ZW^O$	$Z \in \mathbb{R}^{L\times D}$, $W^O \in \mathbb{R}^{D\times D}$	$\text{Out} \in \mathbb{R}^{L\times D}$
$A = \text{Out}\, W^1$	$\text{Out} \in \mathbb{R}^{L\times D}$, $W^1 \in \mathbb{R}^{D\times 4D}$	$A \in \mathbb{R}^{L\times 4D}$
$A' = \text{relu}(A)$	$A \in \mathbb{R}^{L\times 4D}$	$A' \in \mathbb{R}^{L\times 4D}$
$x' = A'W^2$	$A' \in \mathbb{R}^{L\times 4D}$, $W^2 \in \mathbb{R}^{4D\times D}$	$x' \in \mathbb{R}^{L\times D}$

TransformerBlocks$(x \in R^{(L+1)\times D}) \rightarrow x' \in \mathbb{R}^{(L+1)\times D}$

RELAXED SYSTEM LAB

For each inference request:
- ~~$B = 1$;~~
- $L+1$ is the current input sequence length;
- D is the model dimension;
- Multi-head attention: $D = n_H \times H$
- H is the head dimension;
- n_h is the number of heads.

Computation	Input	Output
$Q = xW^Q$	$x \in \mathbb{R}^{(L+1)\times D}$, $W^Q \in \mathbb{R}^{D\times D}$	$Q \in \mathbb{R}^{(L+1)\times D}$
$K = xW^K$	$x \in \mathbb{R}^{(L+1)\times D}$, $W^K \in \mathbb{R}^{D\times D}$	$K \in \mathbb{R}^{(L+1)\times D}$
$V = xW^V$	$x \in \mathbb{R}^{(L+1)\times D}$, $W^V \in \mathbb{R}^{D\times D}$	$V \in \mathbb{R}^{(L+1)\times D}$
$[Q_1, Q_2 \dots, Q_{n_h}] = \text{Partition}_{-1}(Q)$	$Q \in \mathbb{R}^{(L+1)\times D}$	$Q_i \in \mathbb{R}^{(L+1)\times H}, i = 1, \dots n_h$
$[K_1, K_2 \dots, K_{n_h}] = \text{Partition}_{-1}(K)$	$K \in \mathbb{R}^{(L+1)\times D}$	$K_i \in \mathbb{R}^{(L+1)\times H}, i = 1, \dots n_h$
$[V_1, V_2 \dots, V_{n_h}] = \text{Partition}_{-1}(V)$	$V \in \mathbb{R}^{(L+1)\times D}$	$V_i \in \mathbb{R}^{(L+1)\times H}, i = 1, \dots n_h$
$\text{Score}_\text{i} = \text{softmax}(\frac{Q_i K_i^T}{\sqrt{D}}), i = 1, \dots n_h$	$Q_i, K_i \in \mathbb{R}^{(L+1)\times H}$	$\text{score}_\text{i} \in \mathbb{R}^{(L+1)\times(L+1)}$
$Z_i = \text{score}_i V_i, i = 1, \dots n_h$	$\text{score}_\text{i} \in \mathbb{R}^{(L+1)\times(L+1)}$, $V_i \in \mathbb{R}^{L+1\times H}$	$Z_i \in \mathbb{R}^{(L+1)\times H}$
$Z = \text{Merge}_{-1}([Z_1, Z_2 \dots, Z_{n_h}])$	$Z_i \in \mathbb{R}^{(L+1)\times H}, i = 1, \dots n_h$	$Z \in \mathbb{R}^{(L+1)\times D}$
$\text{Out} = ZW^O$	$Z \in \mathbb{R}^{(L+1)\times D}$, $W^O \in \mathbb{R}^{D\times D}$	$\text{Out} \in \mathbb{R}^{(L+1)\times D}$
$A = \text{Out}\, W^1$	$\text{Out} \in \mathbb{R}^{(L+1)\times D}$, $W^1 \in \mathbb{R}^{D\times 4D}$	$A \in \mathbb{R}^{(L+1)\times 4D}$
$A' = \text{relu}(A)$	$A \in \mathbb{R}^{(L+1)\times 4D}$	$A' \in \mathbb{R}^{(L+1)\times 4D}$
$x' = A'W^2$	$A' \in \mathbb{R}^{(L+1)\times 4D}$, $W^2 \in \mathbb{R}^{4D\times D}$	$x' \in \mathbb{R}^{(L+1)\times D}$

Generate the second token.

$$p(x_{L+2}|x_{1:L+1}) = \text{softmax}(x_{L+1}W_{lm})$$

Reuse KV Cache

Some Observation

- Only the last contextual embedding is needed to compute the probabilistic distribution of the next token.
- Contextual embedding for x_i can only depend **unidirectionally** on the left context ($x_{1:i-1}$). In the previous naïve implementation, most of the computation is redundant.
- State-of-the-art implementation splits the computation to two phrases:
  - Prefill phrase: the model takes a prompt sequence as input and engages in the generation of a key-value cache (KV cache) for each Transformer layer.
  - Decode phrase: for each decode step, the model updates the KV cache and reuses the KV to compute the output.

Prefill: TransformerBlocks$(x \in R^{L\times D}) \rightarrow x' \in \mathbb{R}^{L\times D}$

For each inference request:

- ~~$B = 1$;~~
- L is the input sequence length;
- D is the model dimension;
- Multi-head attention:
  $D = n_H \times H$
- H is the head dimension;
- n_h is the number of heads.

Computation	Input	Output
$Q = xW^Q$	$x \in \mathbb{R}^{L\times D}$, $W^Q \in \mathbb{R}^{D\times D}$	$Q \in \mathbb{R}^{L\times D}$
$K = xW^K$	$x \in \mathbb{R}^{L\times D}$, $W^K \in \mathbb{R}^{D\times D}$	$K \in \mathbb{R}^{L\times D}$
$V = xW^V$	$x \in \mathbb{R}^{L\times D}$, $W^V \in \mathbb{R}^{D\times D}$	$V \in \mathbb{R}^{L\times D}$
$[Q_1, Q_2 \ldots, Q_{n_h}] = \text{Partition}_{-1}(Q)$	$Q \in \mathbb{R}^{L\times D}$	$Q_i \in \mathbb{R}^{L\times H}, i = 1, \ldots n_h$
$[K_1, K_2 \ldots, K_{n_h}] = \text{Partition}_{-1}(K)$	$K \in \mathbb{R}^{L\times D}$	$K_i \in \mathbb{R}^{L\times H}, i = 1, \ldots n_h$
$[V_1, V_2 \ldots, V_{n_h}] = \text{Partition}_{-1}(V)$	$V \in \mathbb{R}^{L\times D}$	$V_i \in \mathbb{R}^{L\times H}, i = 1, \ldots n_h$
$\text{Score}_\text{i} = \text{softmax}(\frac{Q_i K_i^T}{\sqrt{D}}), i = 1, \ldots n_h$	$Q_i, K_i \in \mathbb{R}^{L\times H}$	$\text{score}_\text{i} \in \mathbb{R}^{L\times L}$
$Z_i = \text{score}_i V_i, i = 1, \ldots n_h$	$\text{score}_\text{i} \in \mathbb{R}^{L\times L}$, $V_i \in \mathbb{R}^{L\times H}$	$Z_i \in \mathbb{R}^{L\times H}$
$Z = \text{Merge}_{-1}([Z_1, Z_2 \ldots, Z_{n_h}])$	$Z_i \in \mathbb{R}^{L\times H}, i = 1, \ldots n_h$	$Z \in \mathbb{R}^{L\times D}$
$\text{Out} = ZW^O$	$Z \in \mathbb{R}^{L\times D}$, $W^O \in \mathbb{R}^{D\times D}$	$\text{Out} \in \mathbb{R}^{L\times D}$
$A = \text{Out}\, W^1$	$\text{Out} \in \mathbb{R}^{L\times D}$, $W^1 \in \mathbb{R}^{D\times 4D}$	$A \in \mathbb{R}^{L\times 4D}$
$A' = \text{relu}(A)$	$A \in \mathbb{R}^{L\times 4D}$	$A' \in \mathbb{R}^{L\times 4D}$
$x' = A'W^2$	$A' \in \mathbb{R}^{L\times 4D}$, $W^2 \in \mathbb{R}^{4D\times D}$	$x' \in \mathbb{R}^{L\times D}$

Generate the first token.

$$p(x_{L+1}|x_{1:L}) = \text{softmax}(x_L W_{lm})$$

Decode: TransformerBlocks$(t \in R^{1\times D}) \rightarrow t' \in \mathbb{R}^{1\times D}$

RELAXED SYSTEM LAB

For each inference request:
- ~~$B = 1$;~~
- L is the current cached sequence length; it increases by 1 after each step.
- D is the model dimension;
- Multi-head attention: $D = n_H \times H$
- H is the head dimension;
- n_h is the number of heads.

Update the KV cache:

$$K = \text{concat}(K_{\text{cache}}, K_d)$$
$$V = \text{concat}(V_{\text{cache}}, V_d)$$

Generate the second token:

$$p(x_{L+2}|x_{1:L+1}) = \text{softmax}(x_{L+1} W_{lm})$$

Output of last transformer block's t'.

Computationt	Input	Output
$Q = Q_d = tW^Q$	$t \in \mathbb{R}^{1\times D}$, $W^Q \in \mathbb{R}^{D\times D}$	$Q, Q_d \in \mathbb{R}^{1\times D}$
$K_d = tW^K$	$t \in \mathbb{R}^{1\times D}$, $W^K \in \mathbb{R}^{D\times D}$	$K_d \in \mathbb{R}^{1\times D}$
$K = \text{concat}(K_{\text{cache}}, K_d)$	$K_{\text{cache}} \in \mathbb{R}^{L\times D}, K_d \in \mathbb{R}^{1\times D}$	$K \in \mathbb{R}^{(L+1)\times D}$
$V_d = tW^V$	$t \in \mathbb{R}^{1\times D}$, $W^V \in \mathbb{R}^{D\times D}$	$V_d \in \mathbb{R}^{1\times D}$
$V = \text{concat}(V_{\text{cache}}, V_d)$	$V_{\text{cache}} \in \mathbb{R}^{L\times D}, V_d \in \mathbb{R}^{1\times D}$	$V \in \mathbb{R}^{(L+1)\times D}$
$[Q_1, Q_2 \dots, Q_{n_h}] = \text{Partition}_{-1}(Q)$	$Q \in \mathbb{R}^{1\times D}$	$Q_i \in \mathbb{R}^{1\times H}, i = 1, \dots n_h$
$[K_1, K_2 \dots, K_{n_h}] = \text{Partition}_{-1}(K)$	$K \in \mathbb{R}^{(L+1)\times D}$	$K_i \in \mathbb{R}^{(L+1)\times H}, i = 1, \dots n_h$
$[V_1, V_2 \dots, V_{n_h}] = \text{Partition}_{-1}(V)$	$V \in \mathbb{R}^{(L+1)\times D}$	$V_i \in \mathbb{R}^{(L+1)\times H}, i = 1, \dots n_h$
$\text{Score}_\text{i} = \text{softmax}(\frac{Q_i K_i^T}{\sqrt{D}}), i = 1, \dots n_h$	$Q_i \in \mathbb{R}^{1\times H}, K_i \in \mathbb{R}^{(L+1)\times H}$	$\text{score}_\text{i} \in \mathbb{R}^{1\times(L+1)}$
$Z_i = \text{score}_i V_i, i = 1, \dots n_h$	$\text{score}_\text{i} \in \mathbb{R}^{1\times(L+1)}$, $V_i \in \mathbb{R}^{(L+1)\times H}$	$Z_i \in \mathbb{R}^{1\times H}$
$Z = \text{Merge}_{-1}([Z_1, Z_2 \dots, Z_{n_h}])$	$Z_i \in \mathbb{R}^{1\times H}, i = 1, \dots n_h$	$Z \in \mathbb{R}^{1\times D}$
$\text{Out} = ZW^O$	$Z \in \mathbb{R}^{1\times D}$, $W^O \in \mathbb{R}^{D\times D}$	$\text{Out} \in \mathbb{R}^{1\times D}$
$A = \text{Out}\, W^1$	$\text{Out} \in \mathbb{R}^{1\times D}$, $W^1 \in \mathbb{R}^{D\times 4D}$	$A \in \mathbb{R}^{1\times 4D}$
$A' = \text{relu}(A)$	$A \in \mathbb{R}^{1\times 4D}$	$A' \in \mathbb{R}^{1\times 4D}$
$t' = A'W^2$	$A' \in \mathbb{R}^{1\times 4D}$, $W^2 \in \mathbb{R}^{4D\times D}$	$t' \in \mathbb{R}^{1\times D}$

Reuse the KV Cache

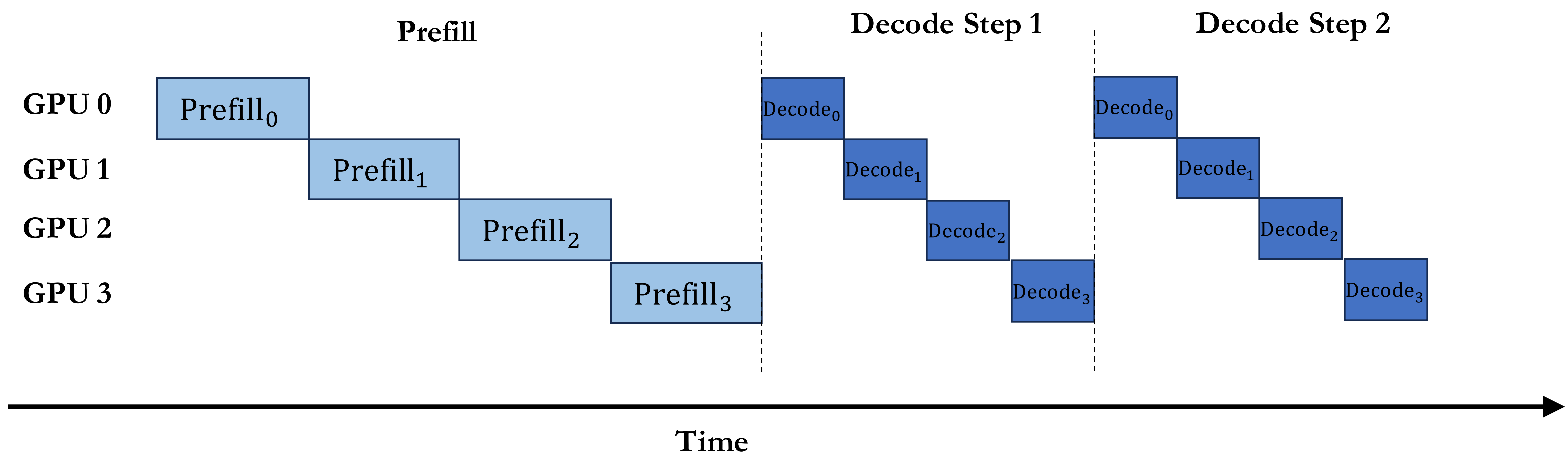
- Performance analysis of this computation paradigm:
  - Prefill phrase: computation bounded.
  - Decode phrase: IO bounded.
  - What is the arithmetic intensity of each step? (Homework 3)
- The memory footprint of the generative inference computation:
  - Model parameters;
  - KV-cache. This will become more significant since the latest models are targeting long context comprehension.

Parallel Generative Inference

Pipeline Parallelism

- Similar to training, pipeline parallelism partitions the model into multiple stages and serves the inference computation as a pipeline, where each GPU or (group of GPUs) handles a stage.
- During the inference computation, the GPU(s) serving stage-(i) needs to **send** the activations to the GPU(s) serving stage- $(i+1)$.
- For inference computation, pipeline parallelism **cannot** reduce the completion time for a single request since only one stage can be active.

Pipeline Parallelism for Inference.



The number on each block represents the stage index.

Pipeline Parallelism

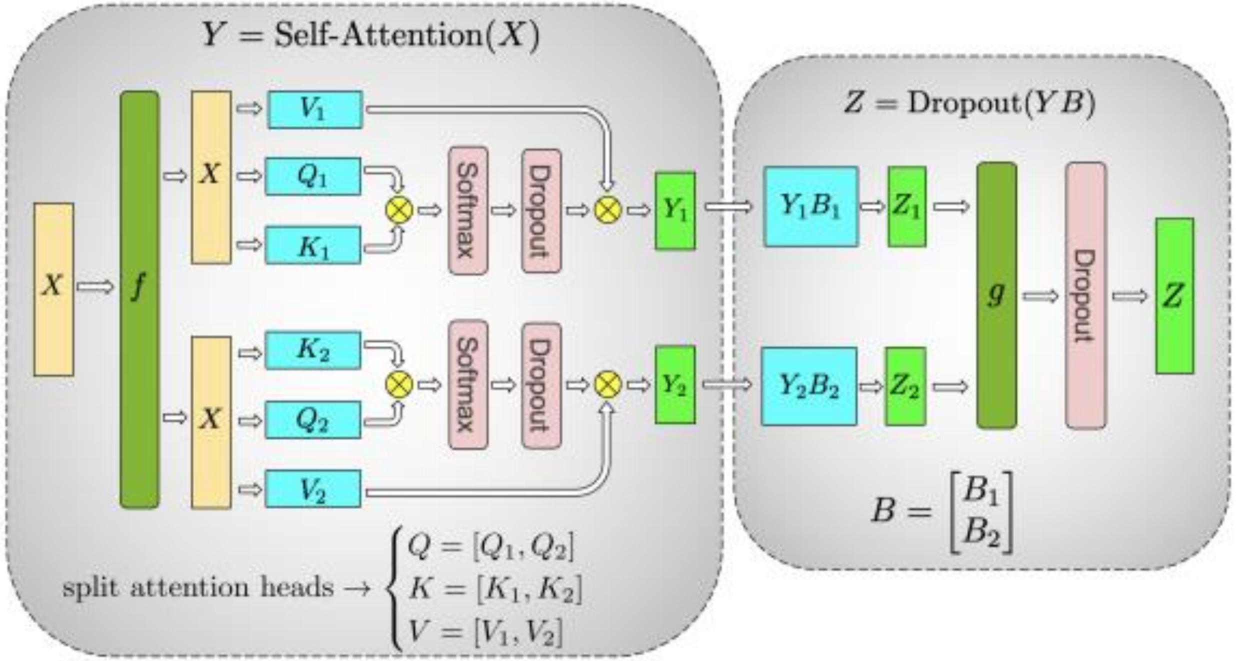
- P2P communication volume:
  - Assume the computation and the communication are all in FP16.
  - L is the input sequence length;
  - D is the model dimension.
  - Prefill stage: $2LD$ bytes.
  - Decode stage: $2D$ bytes for each generated token.

Tensor Model Parallelism

- Tensor model parallelism partitions the inference computation at the level of transformer layers over multiple GPUs, where the weight matrices are distributed both row-wisely and column-wisely.
- Two **AllReduce** operations are required to aggregate each layer's output activations:
  - One **AllReduce** for the Multihead-Attention.
  - One **AllReduce** for the MLP.
- Tensor model parallelism splits both the data scan and computation among a tensor model parallel group, which can effectively scale out the inference computation if the connection is fast among the group.

Multi-Head Attention in Tensor Model Parallelism

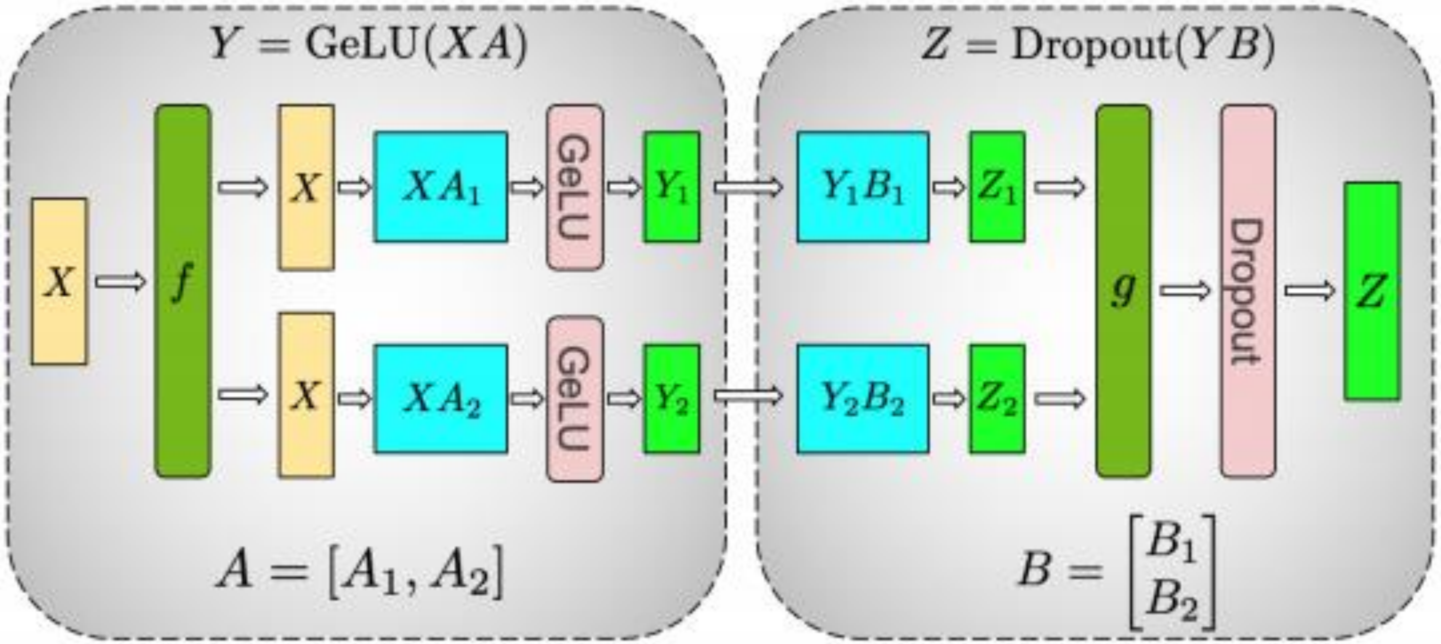
RELAXED SYSTEM LAB



(b) Self-Attention

- f is the identity operator in the forward pass ~~and the AllReduce operator in the backward pass.~~
- g is the AllReduce operator in the forward pass ~~and the identity operator in the backward pass.~~

MLP in Tensor Model Parallelism



(a) MLP

- f is the identity operator in the forward pass ~~and the AllReduce operator in the backward pass.~~
- g is the AllReduce operator in the forward pass ~~and the identity operator in the backward pass.~~

Tensor Model Parallelism

- Collective communication volume:
  - Assume the computation and the communication are all in FP16.
  - L is the input sequence length;
  - D is the model dimension.
  - Prefill stage:
    - For each layer, two **AllReduce**s, where each aggregates $2LD$ bytes.
  - Decode stage:
    - For each generated token, each layer, two **AllReduce**s, where each aggregates $2D$ bytes.

References

- https://arxiv.org/abs/2402.16363
- https://huggingface.co/docs/transformers/en/llm_tutorial