On Long-context Efficiency

Chen Zhang

Overview

- Long-context efficiency: concerned measures correlated with the length of contexts.
- Two bottlenecks: 1) memory, and 2) latency
- Memory perspective: KV cache
- Latency perspective: 1) prefiling latency, and 2) decoding latency

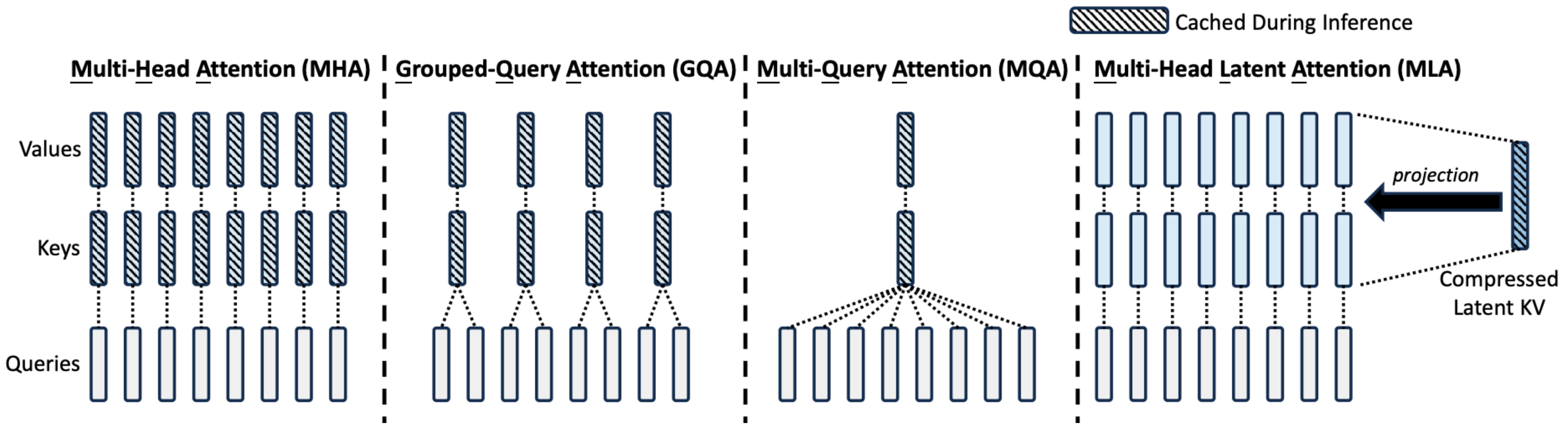
Memory Perspective

- KV cache
  - Architecture: 1) MHA => GQA, MQA, MLA, and 2) full layer => cross-layer sharing, etc.
  - Quantization: KVQuant
  - Merging: DMC, CAM
  - Decomposition: SVD-a
  - Eviction: Attention Sink, H2O

Memory Perspective

KV Cache - Architecture

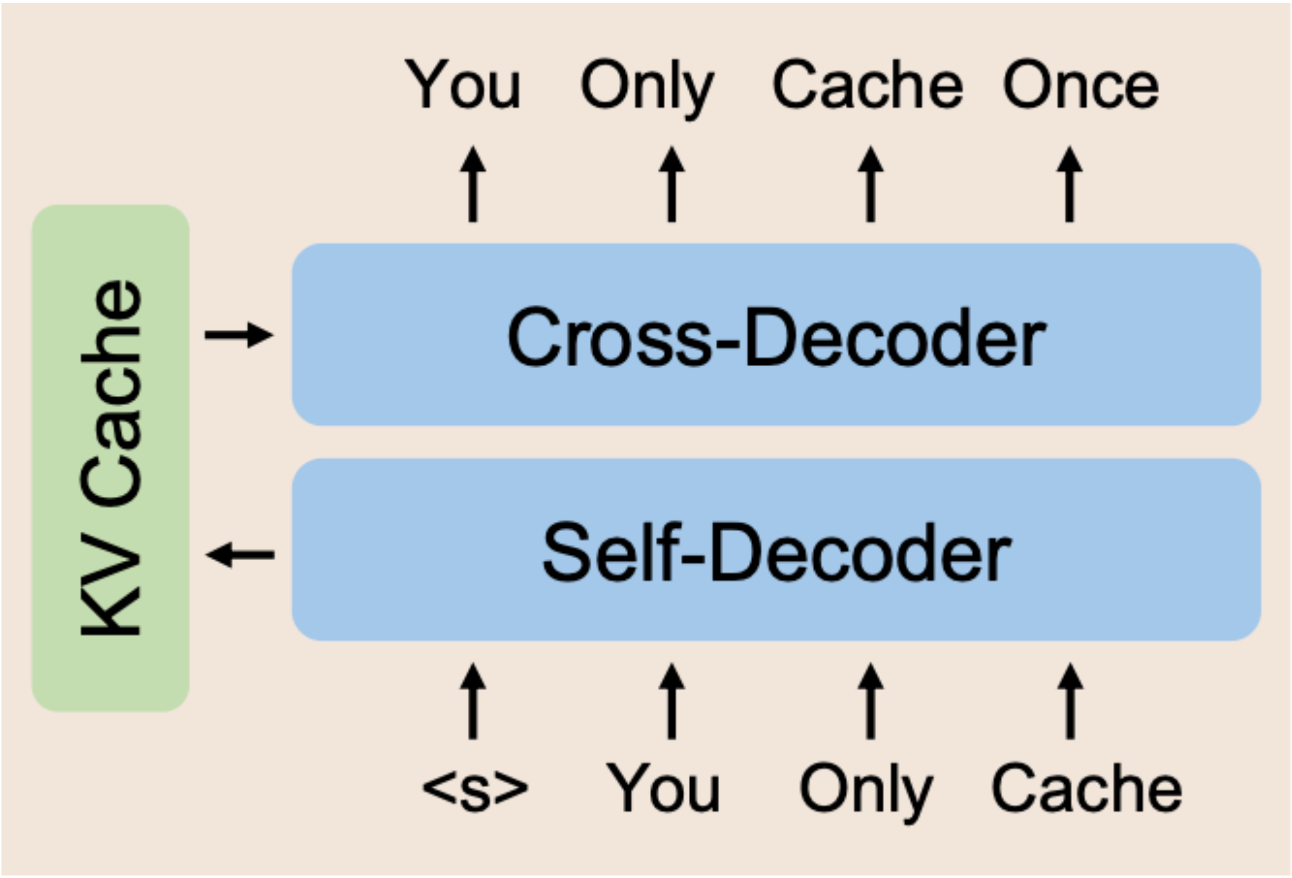
- MHA => MQA => GQA (https://arxiv.org/pdf/2305.13245)
- => MLA (https://arxiv.org/pdf/2405.04434)

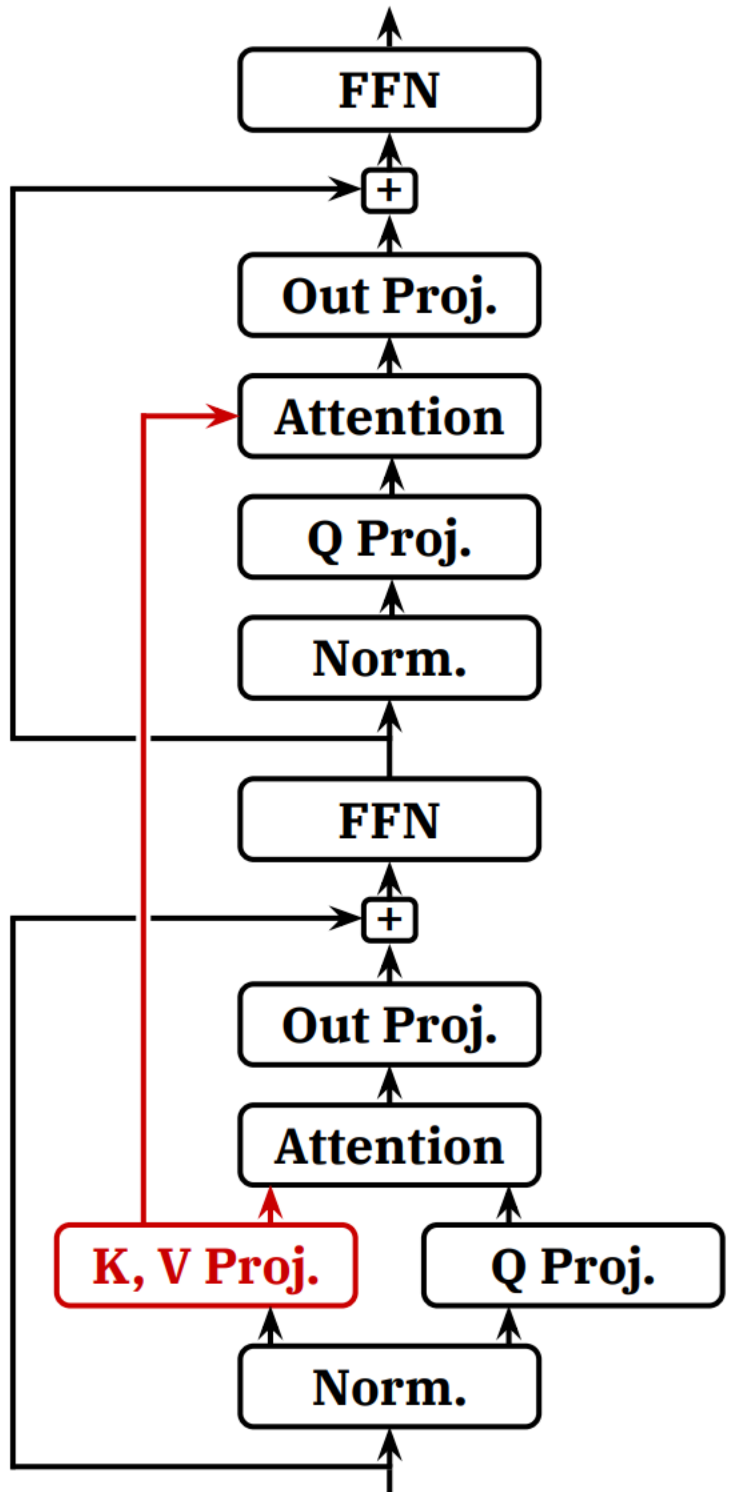


Memory Perspective

KV Cache - Architecture

- YOCO: bottom-top cache sharing (https://arxiv.org/pdf/2405.05254)
- Cross-layer Attention: interleaved cache sharing (https://arxiv.org/pdf/2405.12981)

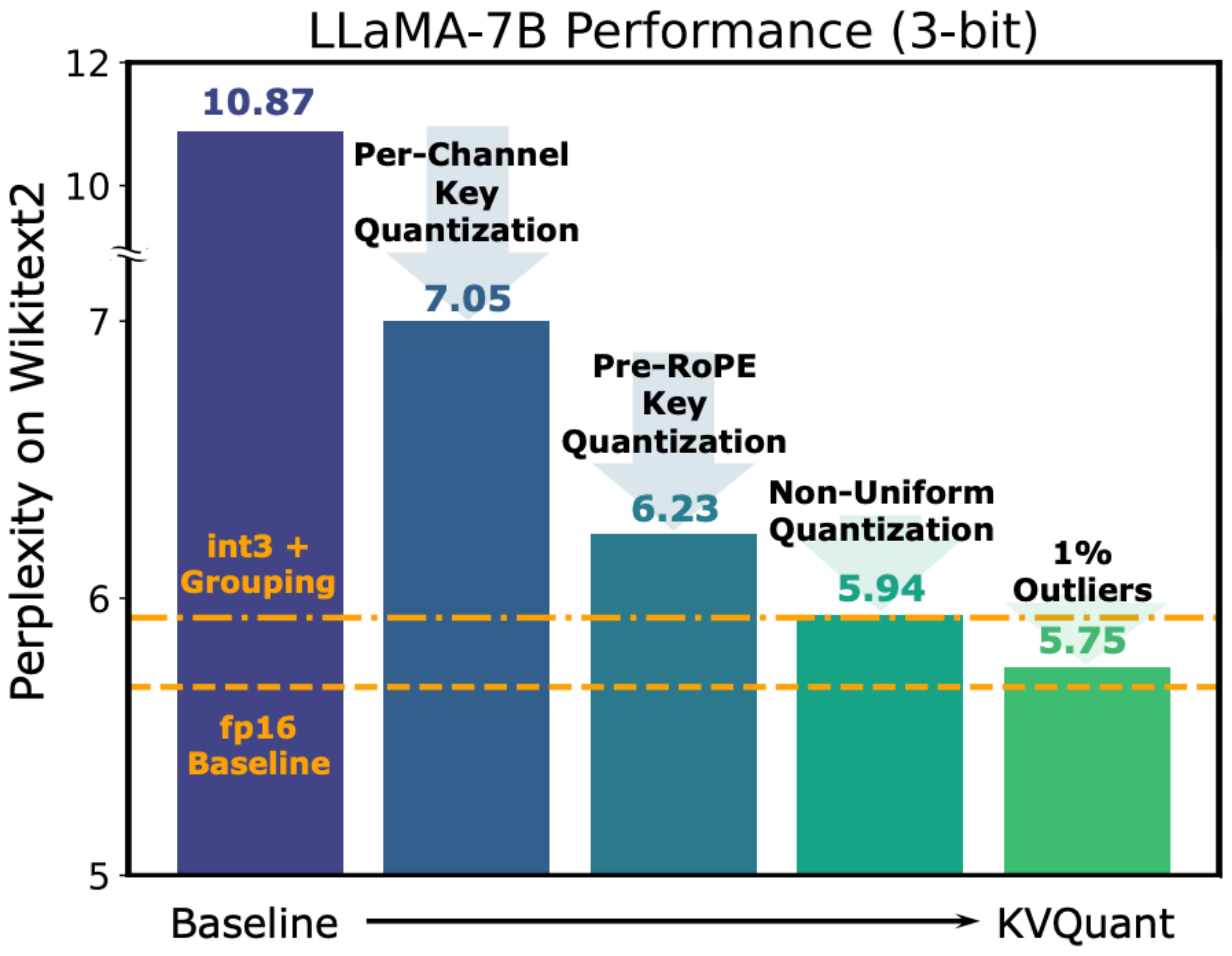




Memory Perspective

KV Cache - Quantization

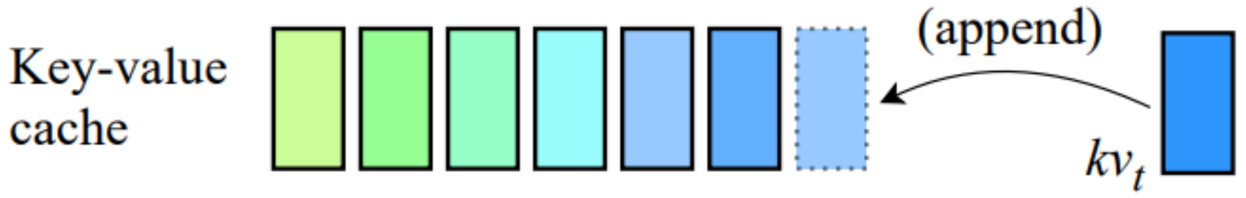
- KVQuant (https://arxiv.org/pdf/2401.18079)



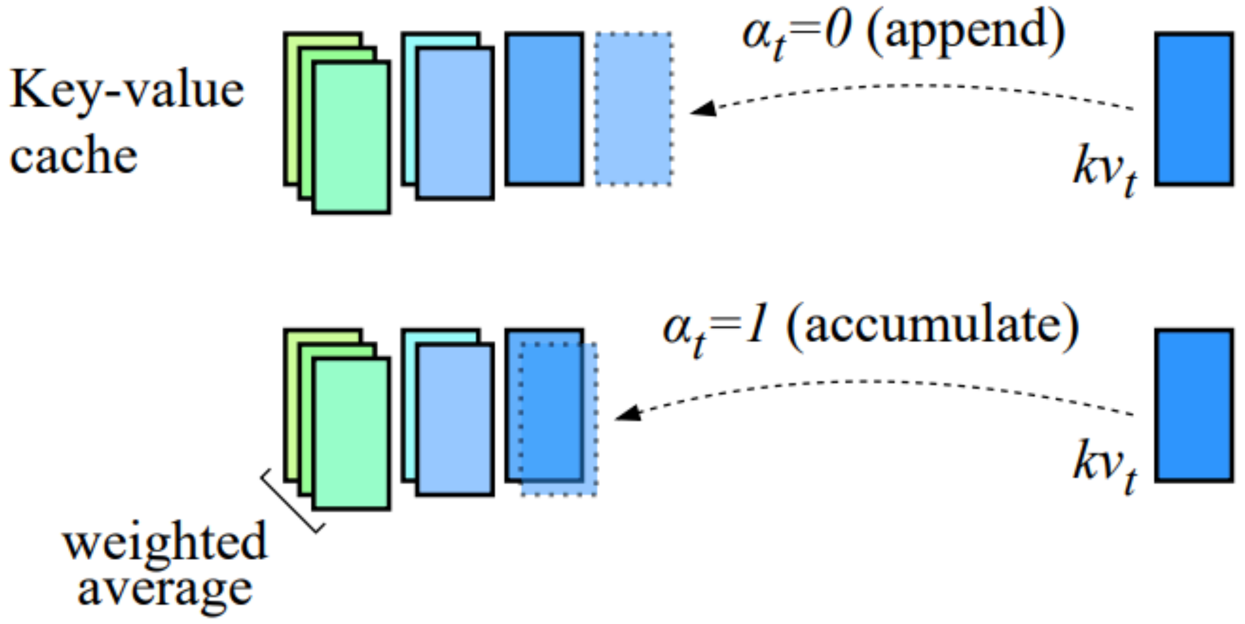
Memory Perspective

KV Cache - Merging

- DMC (https://arxiv.org/pdf/2403.09636v1)



(a) Regular key–value cache with items kv_i depicted as boxes. New items are always appended.



(b) Dynamic Memory Compression (DMC) chooses whether to accumulate or append current items, resulting in a smaller key–value cache.

Algorithm 1 Single-head KV cache update with Dynamic Memory Compression (DMC)

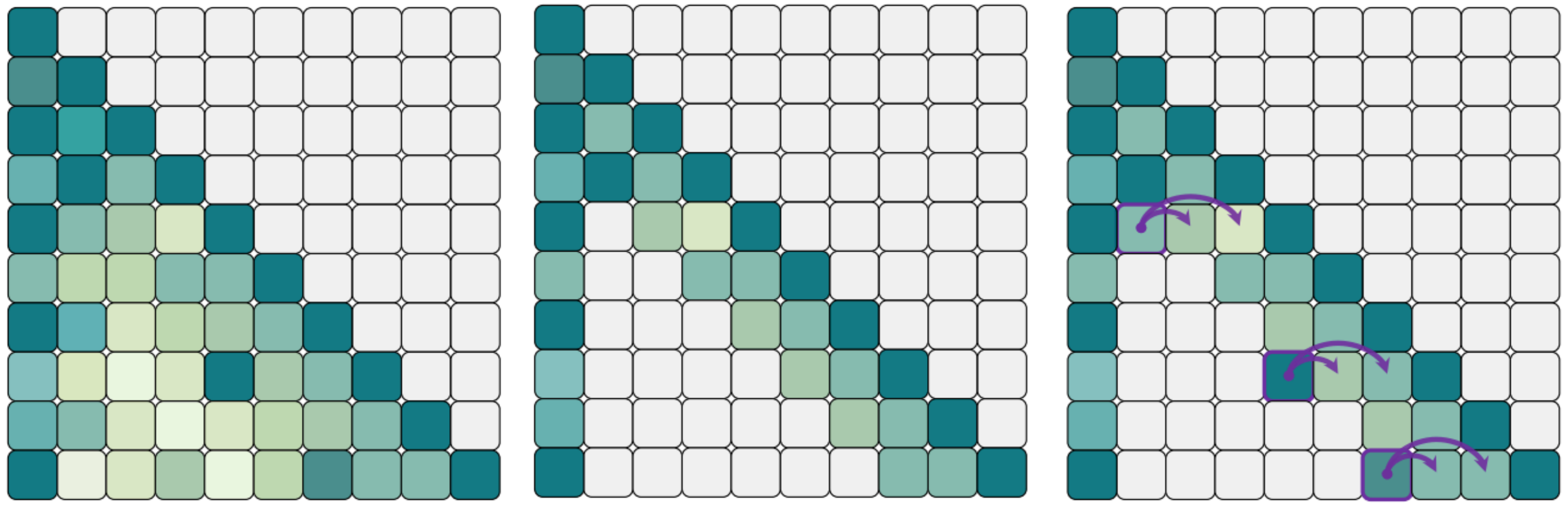
---

procedure KV-UPDATE($K, V, \mathbf{q}_t, \mathbf{k}_t, \mathbf{v}_t$)
    $\alpha_t \leftarrow \text{round}(\text{sigmoid}(\mathbf{k}_t[0]))$
    $\omega_t \leftarrow \text{sigmoid}(\mathbf{q}_t[0])$
    if $\alpha_t = 1$ **then** ▷ ACCUMULATE
        $z_t \leftarrow z_{t-1} + \omega_t$
        $K \leftarrow [K_{1:l-1}, (\mathbf{k}_l z_{t-1} + \mathbf{k}_t \omega_t)\,/\,z_t]$
        $V \leftarrow [V_{1:l-1}, (\mathbf{v}_l z_{t-1} + \mathbf{v}_t \omega_t)\,/\,z_t]$
    else ▷ APPEND
        $z_t \leftarrow \omega_t$
        $K \leftarrow [K_{1:l}\ ,\ \mathbf{k}_t]$
        $V \leftarrow [V_{1:l}\ ,\ \mathbf{v}_t]$
    $\mathbf{k}_t[0] \leftarrow 0$
    $\mathbf{q}_t[0] \leftarrow 0$
    return $K, V, \mathbf{q}_t, \mathbf{k}_t$

Memory Perspective

KV Cache - Merging

- CAM (https://openreview.net/pdf?id=LCTmppB165)



(a) Dense Attention (b) StreamingLLM (c) Cache Merging (Ours)

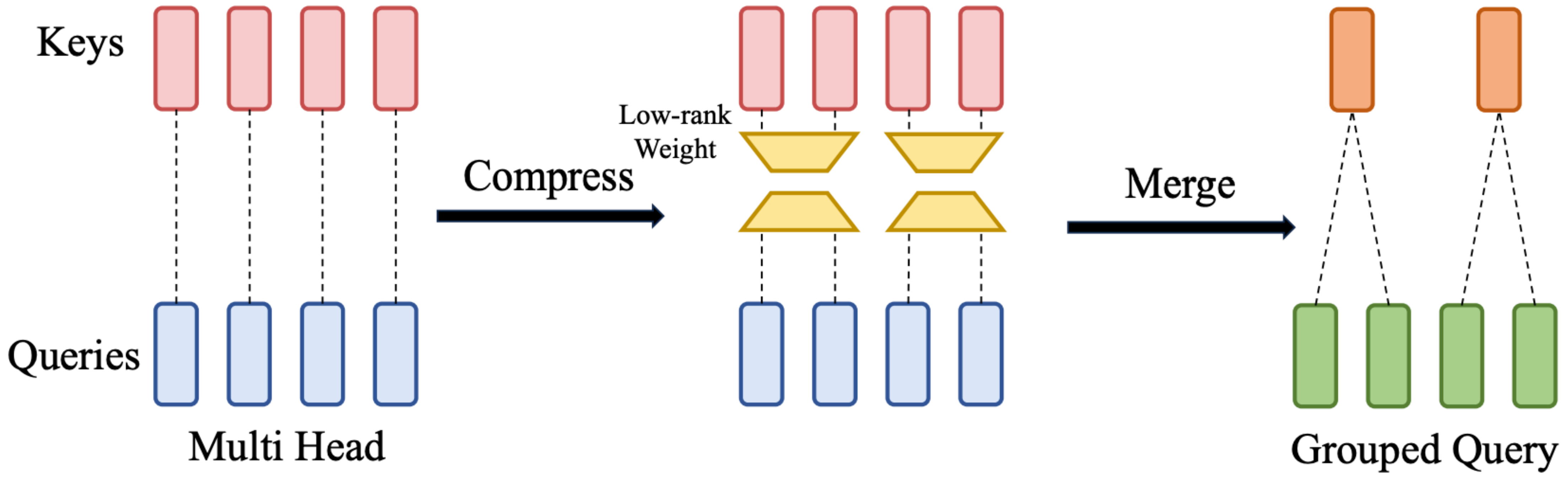
Algorithm 1 Cache Merging at t-th Generation Step

Input: Attention weights A, V-Cache V, $i, j, m \in \mathbb{N}$
Let i denote the index of to-be-evicted cache
Let j denote the first index of local tokens
Let m denote the number of to-be-merged tokens
$\bar{A} = \sum_{k=1}^{t} A^k$
$M = \text{Bernoulli}(\text{clamp}(\frac{\bar{A}_i}{\text{avg}(\bar{A}_{j:j+m})}, 0, 1))$ ◁ Eq. (14)
for $k = j$ to $j + m$ **do**
  $\bar{V}_k = V_k + M\frac{V_i}{m}$ ◁ Eq. (10)
  $V_k = \bar{V}_k$
end for
Output: Updated V-Cache V

Memory Perspective

KV Cache - Decomposition

- SVD-a (https://arxiv.org/pdf/2406.07056)



where $\tilde{K}_i, \tilde{V}_i \in \mathbb{R}^{l \times td_h}$ and $i \in \{0, 1, \cdots, g-1\}$. Then, we perform SVD on those caches, i.e., $\tilde{K}_i = \Phi^i \Sigma^i \Psi^i$ and $\tilde{V}_i = \Theta^i \Lambda^i \Omega^i$, where $\Phi^i, \Theta^i \in \mathbb{R}^{l \times td_h}$ and $\Psi^i, \Omega^i \in \mathbb{R}^{td_h \times td_h}$ are orthonormal matrices. $\Sigma^i, \Lambda^i \in \mathbb{R}^{td_h \times td_h}$ are diagonal rectangular matrices containing singular values in the decreasing order. Therefore, we can get the low-rank approximation of $\tilde{K}_i, \tilde{V}_i$ as

$$\tilde{K}_i \approx \tilde{K}_i (\Psi^i_{d_h})^\top \Psi^i_{d_h}, \tag{9}$$

$$\tilde{V}_i \approx \tilde{V}_i (\Omega^i_{d_h})^\top \Omega^i_{d_h}, \tag{10}$$

where $\Psi^i_{d_h}, \Omega^i_{d_h} \in \mathbb{R}^{d_h \times td_h}$ are the top-d_h rows of Ψ^i and Ω^i, respectively. Because KV caches usually have abundant parameters, it is not practical to collect all caches and calculate SVD directly. Instead, we update $\tilde{K}_i^\top \tilde{K}_i$ and $\tilde{V}_i^\top \tilde{V}_i$ in a streamline fashion and then compute their eigen-decompositions. Note that there is no non-linear layer between those key and value matrices and $(\Psi^i_{d_h})^\top$ & $(\Omega^i_{d_h})^\top$. Therefore, we can merge them directly and get the new i-th key and value matrices in GQA as

$$\tilde{W}_{K_i} = [W_{K_{i \times t}}, W_{K_{i \times t+1}}, \ldots, W_{K_{i \times t+t-1}}](\Psi^i_{d_h})^\top, \tag{11}$$

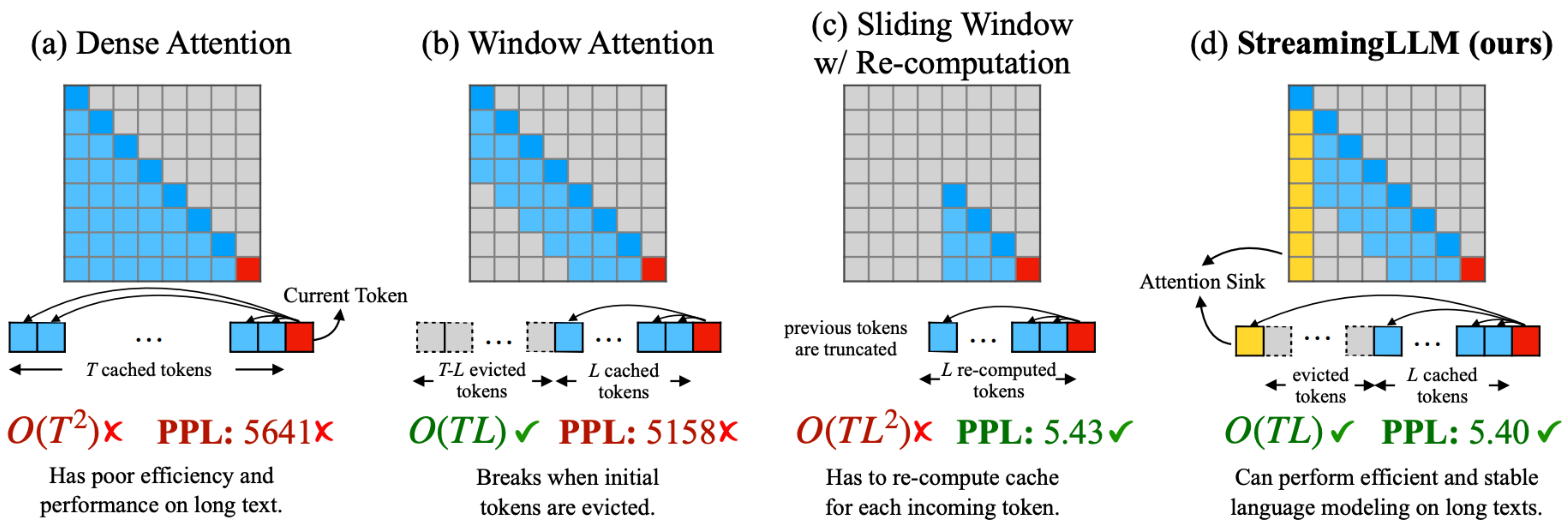
$$\tilde{W}_{V_i} = [W_{V_{i \times t}}, W_{V_{i \times t+1}}, \ldots, W_{V_{i \times t+t-1}}](\Omega^i_{d_h})^\top, \tag{12}$$

where $\tilde{W}_{K_i}, \tilde{W}_{V_i} \in \mathbb{R}^{d \times d_h}$. With the above transformation, the number of KV heads can be reduced

Memory Perspective

KV Cache - Eviction

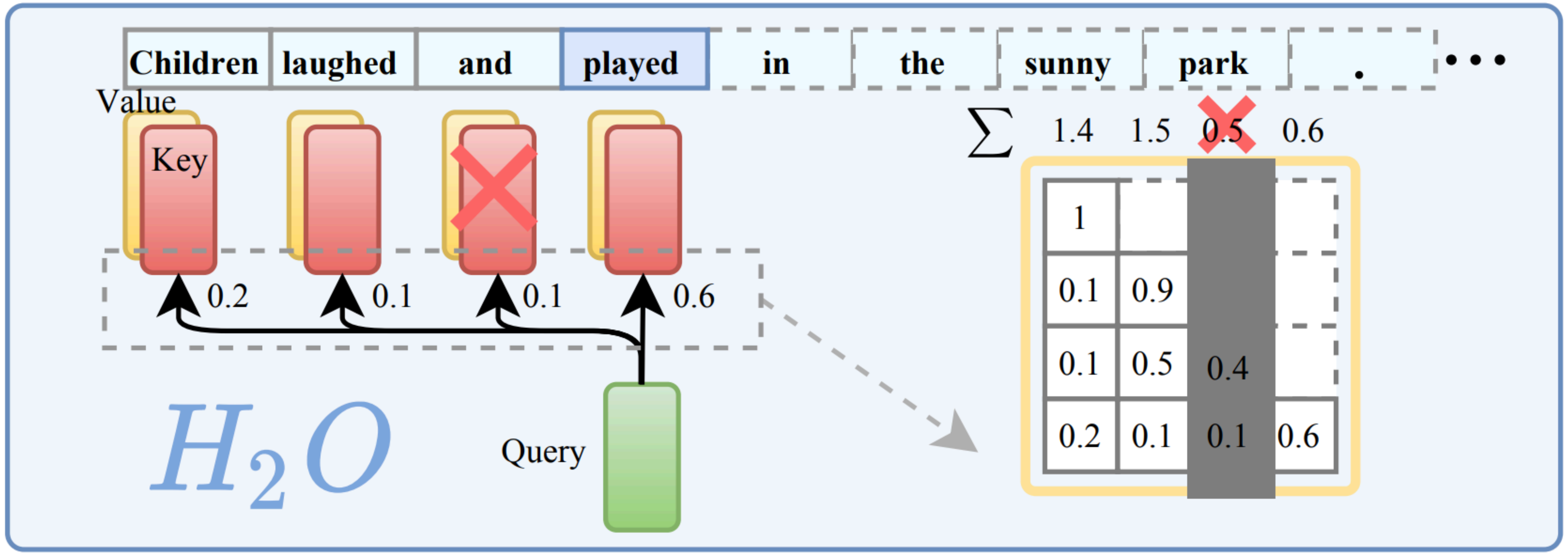
- Attention Sink (https://arxiv.org/pdf/2406.07056)



Memory Perspective

KV Cache - Eviction

- H2O (https://arxiv.org/pdf/2306.14048)



Memory Perspective

KV Cache - Eviction

- Is a uniform distribution of KV cache budget across all layer reasonable?
- An asymmetric distribution is way better!

Latency Perspective

- Prefilling latency: time to first token
  - Token dropping: SelectiveContext, LLMLingua
  - Token offloading: CEPE, LLoCO
  - Token skipping: Sparse Attention, EarlyExiting, Mixture-of-Depths

Latency Perspective

Prefilling - Token Dropping

- SelectiveContext (https://arxiv.org/pdf/2310.06201)

Original: INTRODUCTION Continual Learning (~~CL)~~ ~~,~~ ~~also known as Lifelong Learning , is~~ a promising learning paradigm to design models ~~that~~ have to ~~learn~~ how ~~to perform multiple tasks~~ across ~~different environments~~ over ~~their lifetime [~~To uniform the language and enhance ~~the readability of the paper we~~ adopt the unique term continual learning ~~(CL) .] .~~ Ideal CL models in ~~the real world~~ should ~~be~~ deal ~~with~~ domain shifts ~~,~~ researchers ~~have~~ recently started ~~to~~ sample tasks ~~from~~ two different datasets ~~.~~ For instance ~~,~~ proposed to train and evaluate ~~a model~~ on Imagenet first ~~and then~~ challenge ~~its performance on the~~ Places365 ~~dataset .~~ considers more scenarios ~~,~~ starting ~~with~~ Imagenet or Places365 ~~, and then moving on to~~ the VOC/CUB/Scenes datasets ~~.~~ Few works propose more advanced scenarios built ~~on~~ top ~~of~~ more than two datasets ~~.~~

Filtered: INTRODUCTION Continual Learning (a promising learning paradigm to design models have to how across overTo uniform the language and enhance adopt the unique term continual learning Ideal CL models in should deal domain shifts researchers recently started sample tasks two different datasets For instance proposed to train and evaluate on Imagenet first challenge Places365considers more scenarios starting Imagenet or Places365 the VOC/CUB/Scenes datasetsFew works propose more advanced scenarios built top more than two datasets

$$I(x_i) = -\log_2 P(x_i|x_0, x_1, ..., x_{i-1}) \quad (5)$$

Latency Perspective

Prefilling - Token Dropping

- LLMLingua (https://arxiv.org/pdf/2310.05736)

Algorithm 2 Pseudo code of Iterative Token-level Prompt Compression (ITPC).

Input: A small language model $\mathcal{M}_s$; the prompt from budget controller $\boldsymbol{x}' = (\boldsymbol{x}^{\text{ins}}, \boldsymbol{x}^{\mathcal{D}}, \boldsymbol{x}^{\text{que}})$; target compression rateτ, adjusted compression rate $\triangle\tau_{\text{ins,que}}$.

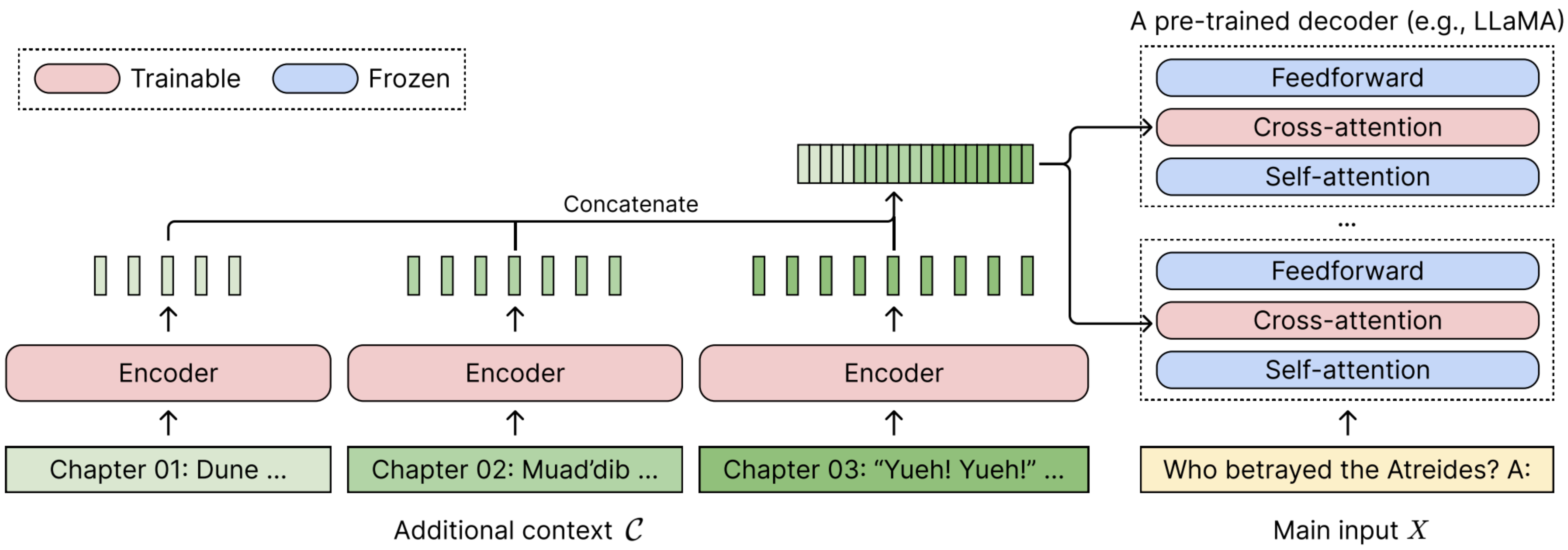
Set the selected token set $\mathcal{T} = \phi$
Get segment set $\mathcal{S}$.
for $i = 1, 2, \ldots, m$ **do**
    Get the conditional probabilities $p(\boldsymbol{s}_i)$ via Eq.(5)
    Get the compression threshold γ_i with Eq. (6).
    Append the compressed token to $\mathcal{T}$ via Eq.(7).
end for
Concatenate all tokens in $\mathcal{T}$ as $\widetilde{\boldsymbol{x}}$.

Output: The compressed prompt $\widetilde{\boldsymbol{x}}$.

Latency Perspective

Prefilling - Token Offloading

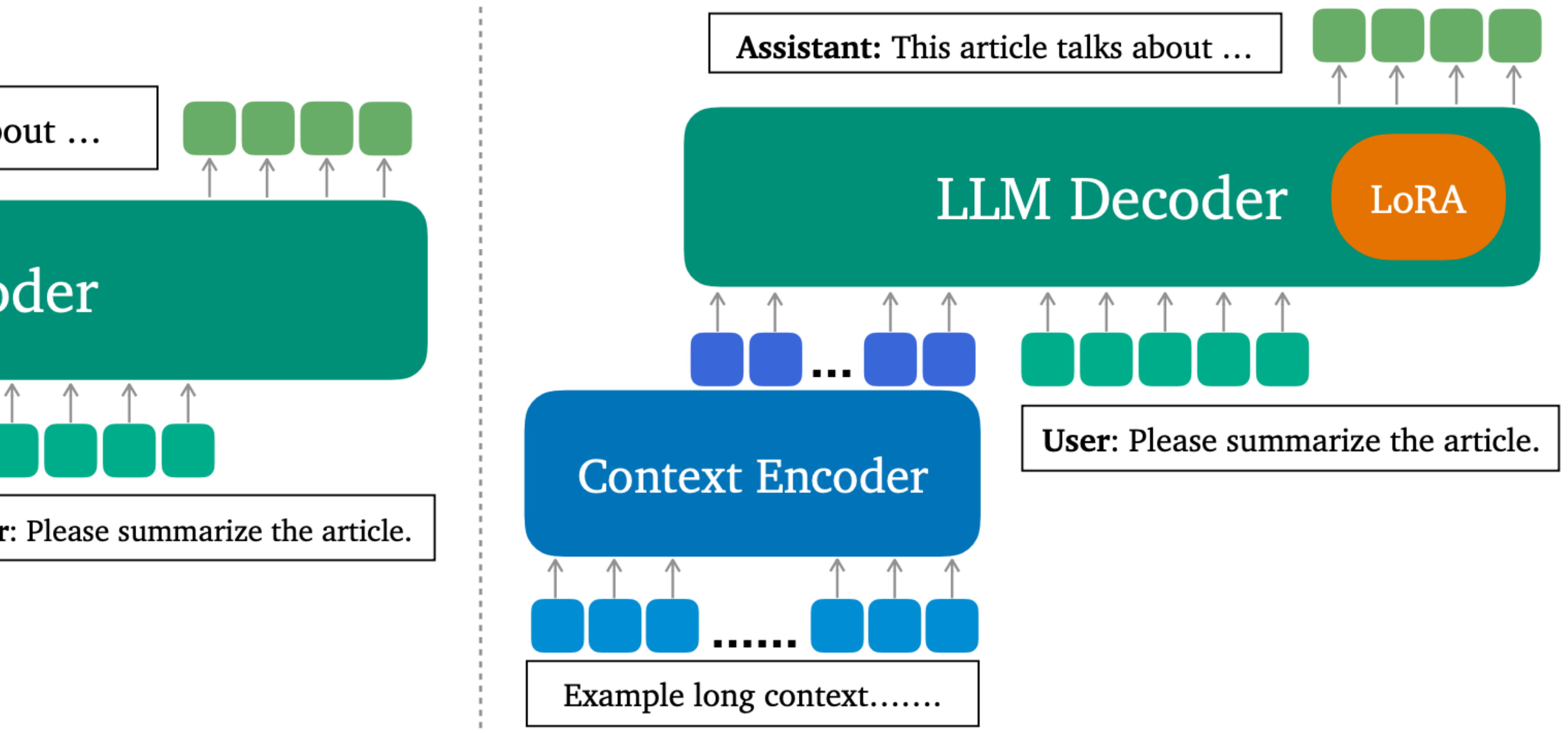
- CEPE (https://arxiv.org/pdf/2402.16617)

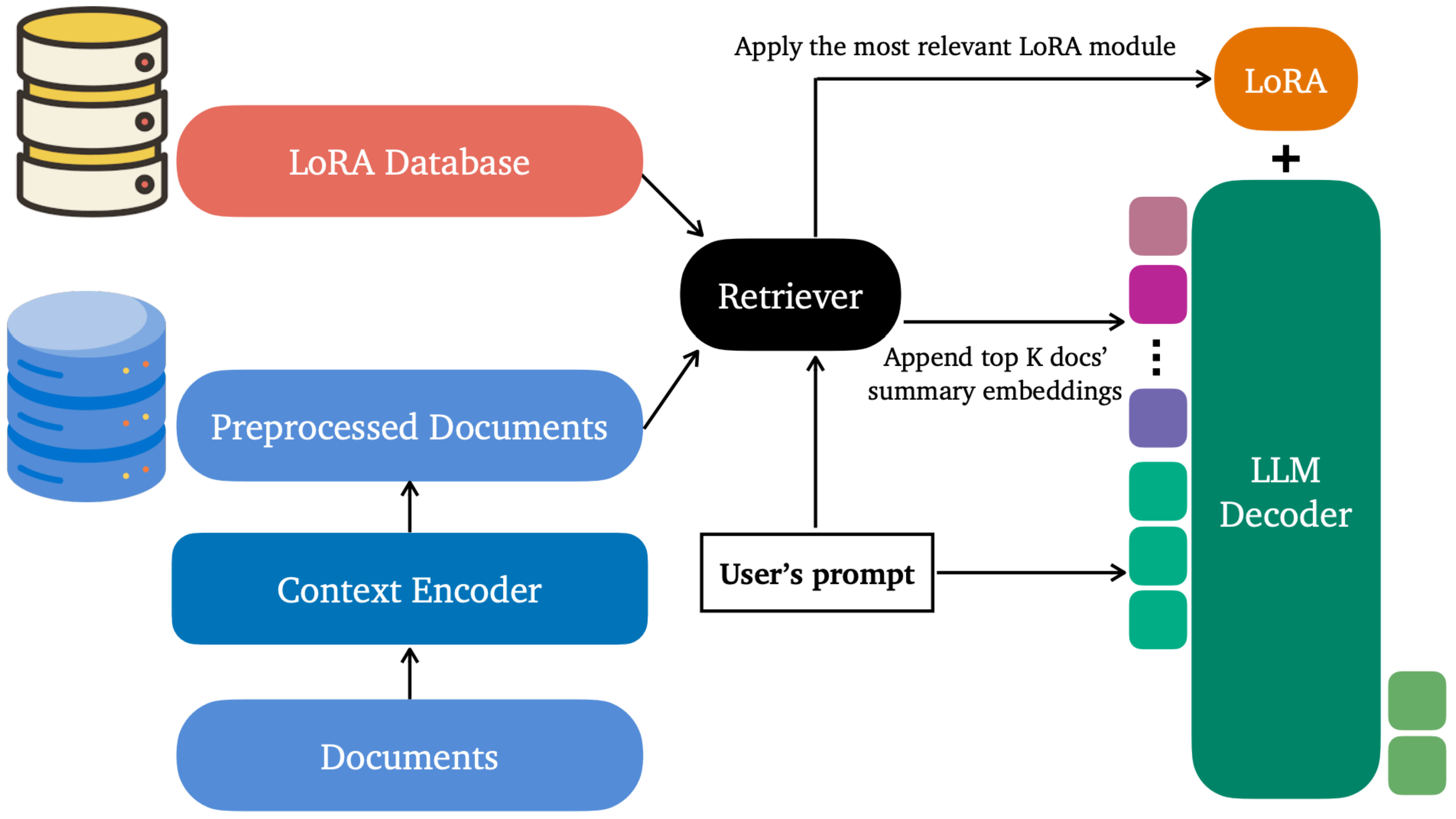


Latency Perspective

Prefilling - Token Offloading

- LLoCO (https://arxiv.org/pdf/2404.07979)

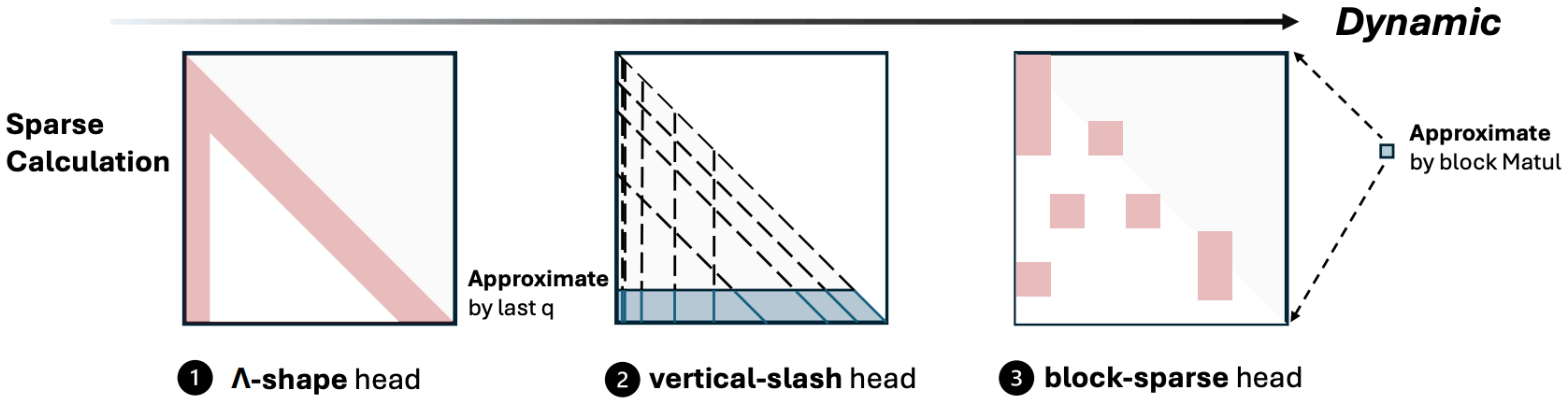




Latency Perspective

Prefilling - Token Skipping

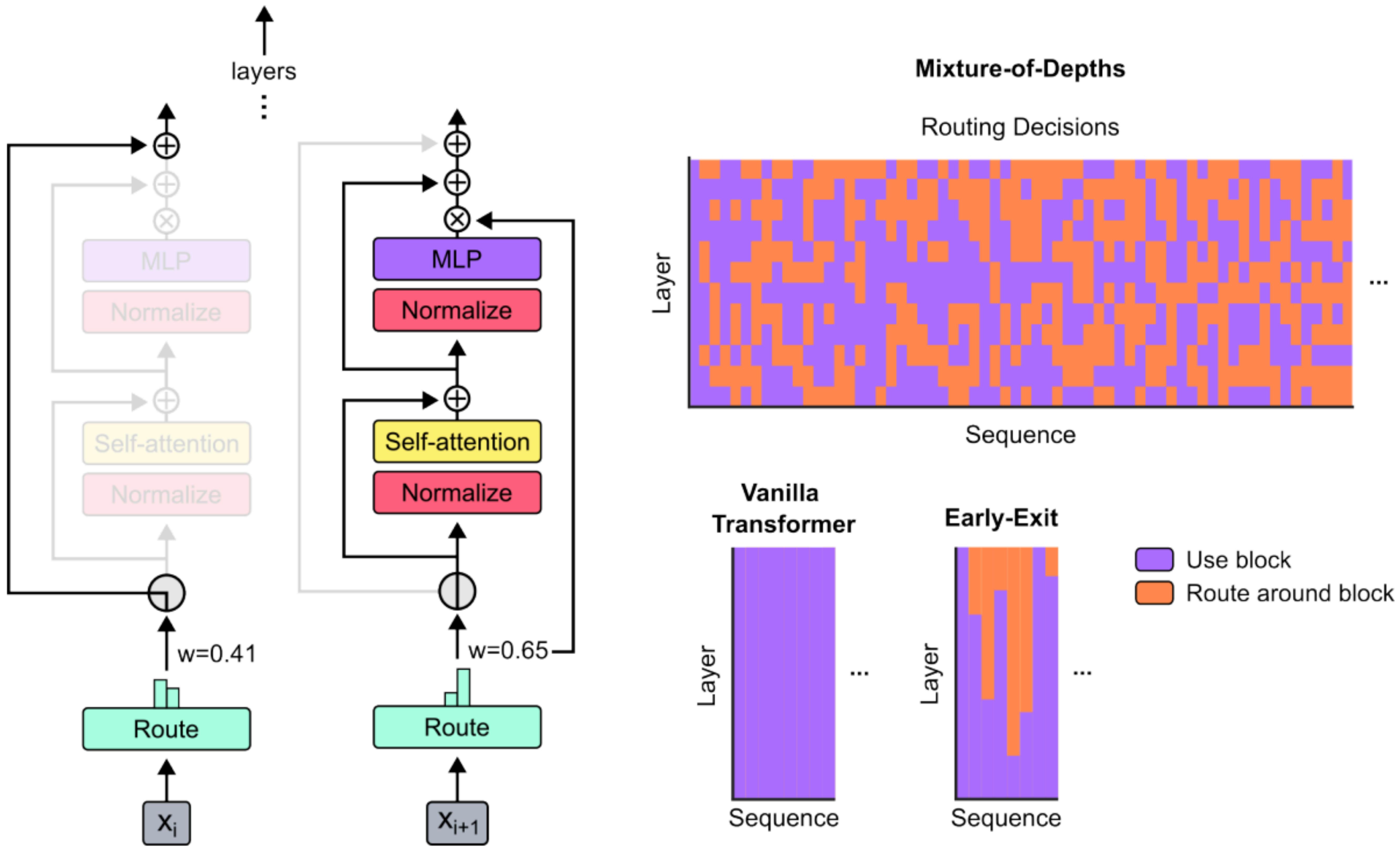
- MInference (https://arxiv.org/pdf/2407.02490)



Latency Perspective

Prefilling - Token Skipping

- Mixture-of-Depths (https://arxiv.org/pdf/2404.02258)



Latency Perspective

- Decoding latency: time per output token
  - Speculative decoding: Speculative sampling, Assisted Generation
  - Blockwise parallel decoding: Medusa
  - Token skipping: Sparse Attention, EarlyExiting, Mixture-of-Depths

Latency Perspective

Decoding - Speculative Decoding

- Speculative Sampling (https://arxiv.org/pdf/2302.01318)

Algorithm 2 Speculative Sampling (SpS) with Auto-Regressive Target and Draft Models

Given lookahead K and minimum target sequence length T.
Given auto-regressive target model $q(.|.)$, and auto-regressive draft model $p(.|.)$, initial prompt sequence $x_0, \ldots, x_t$.
Initialise $n \leftarrow t$.
while $n < T$ **do**
  for $t = 1 : K$ **do**
    Sample draft auto-regressively $\tilde{x}_t \sim p(x|, x_1, \ldots, x_n, \tilde{x}_1, \ldots, \tilde{x}_{t-1})$
  end for
  In parallel, compute $K + 1$ sets of logits from drafts $\tilde{x}_1, \ldots, \tilde{x}_K$:

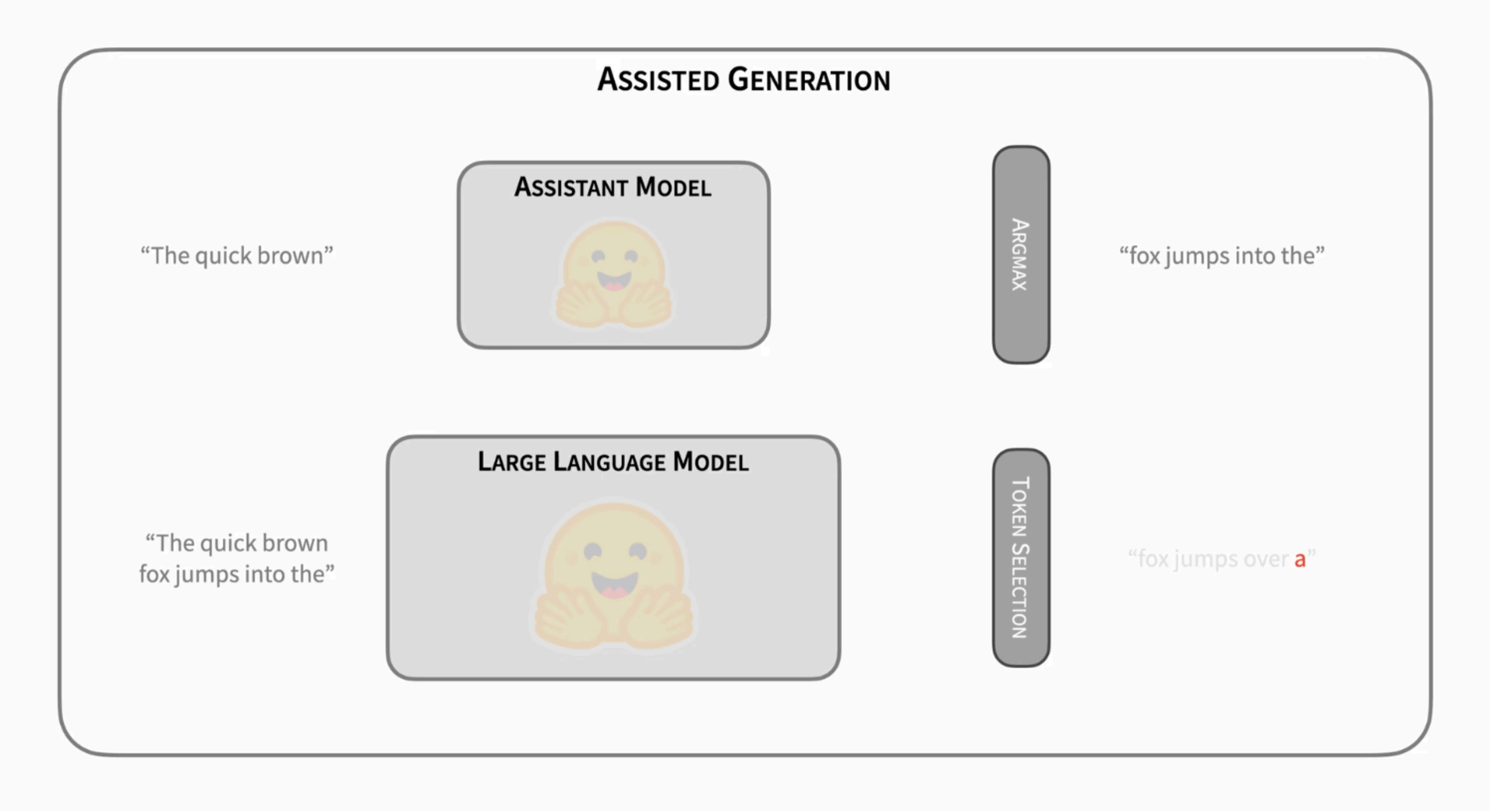
$$q(x|, x_1, \ldots, x_n),\ q(x|, x_1, \ldots, x_n, \tilde{x}_1),\ \ldots,\ q(x|, x_1, \ldots, x_n, \tilde{x}_1, \ldots, \tilde{x}_K)$$

  for $t = 1 : K$ **do**
    Sample $r \sim U[0, 1]$ from a uniform distribution.
    if $r < \min\left(1, \frac{q(x|x_1,\ldots,x_{n+t-1})}{p(x|x_1,\ldots,x_{n+t-1})}\right)$, **then**
      Set $x_{n+t} \leftarrow \tilde{x}_t$ and $n \leftarrow n + 1$.
    else
      sample $x_{n+t} \sim (q(x|x_1, \ldots, x_{n+t-1}) - p(x|x_1, \ldots, x_{n+t-1}))_+$ and exit for loop.
    end if
  end for
  If all tokens $x_{n+1}, \ldots, x_{n+K}$ are accepted, sample extra token $x_{n+K+1} \sim q(x|, x_1, \ldots, x_n, x_{n+K})$ and set $n \leftarrow n + 1$.
end while

Latency Perspective

Decoding - Speculative Decoding

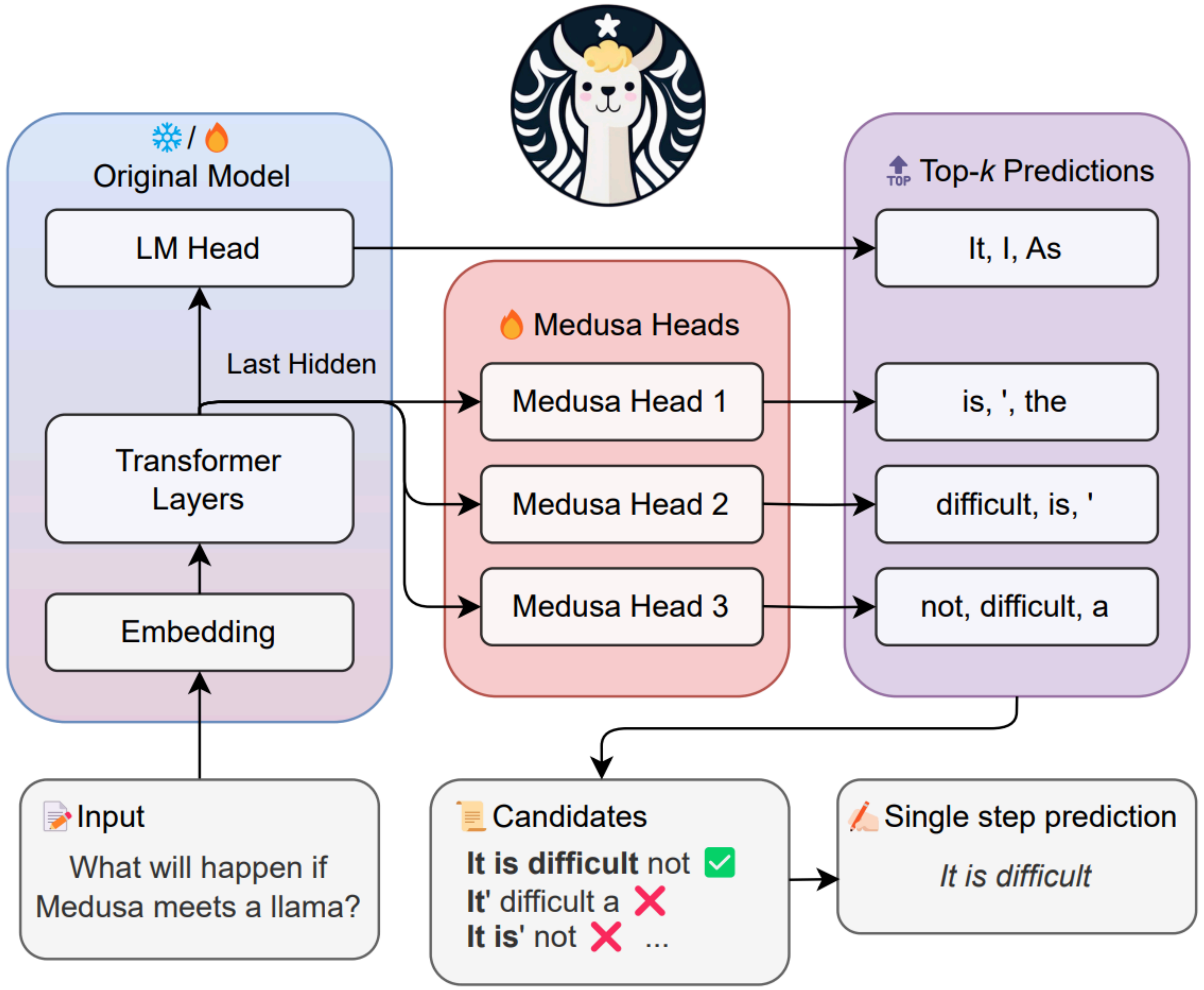
- Assisted Generation (https://huggingface.co/blog/assisted-generation)

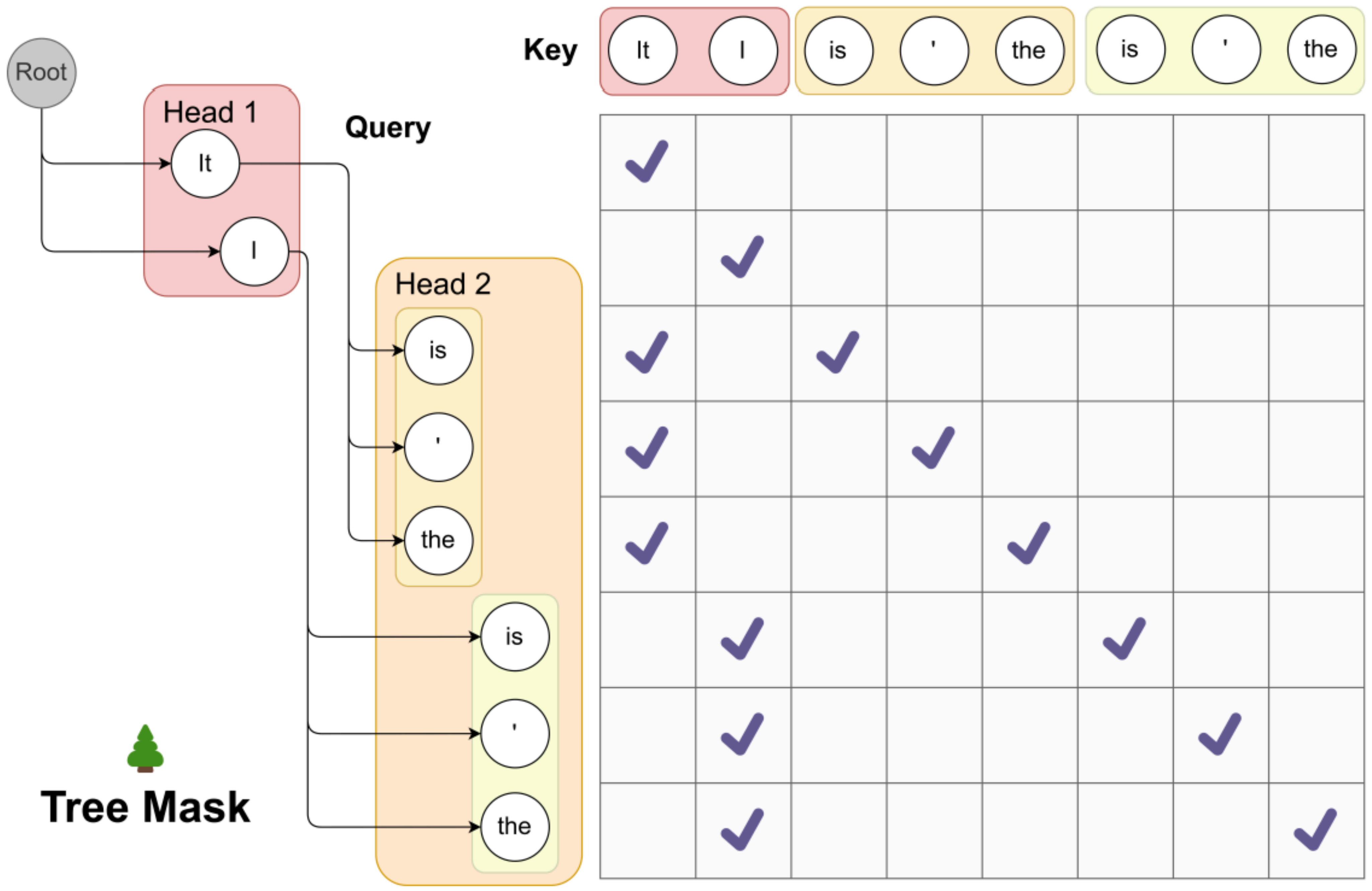


Latency Perspective

Decoding - Block Parallel Decoding

- Medusa (https://arxiv.org/pdf/2401.10774)





Latency Perspective

Decoding - Speculative Decoding

- What is the most suitable number of draft tokens?
- Roughly 3-5 tokens are most appropriate!