FlexiCache

Leveraging Temporal Stability of Attention Heads for Efficient KV Cache Management

Nazmul Takbir

Hamidreza Alikhani Nikil Dutt Sangeetha Abdu Jyothi

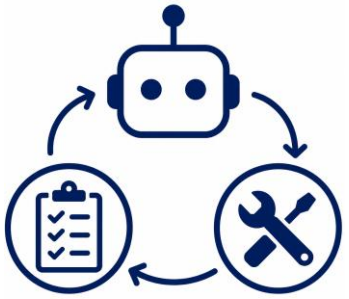
UCI University of California, Irvine

MLSys 2026 · Bellevue, WA, USA

Modern LLM Workloads Demand Long Context

Coding

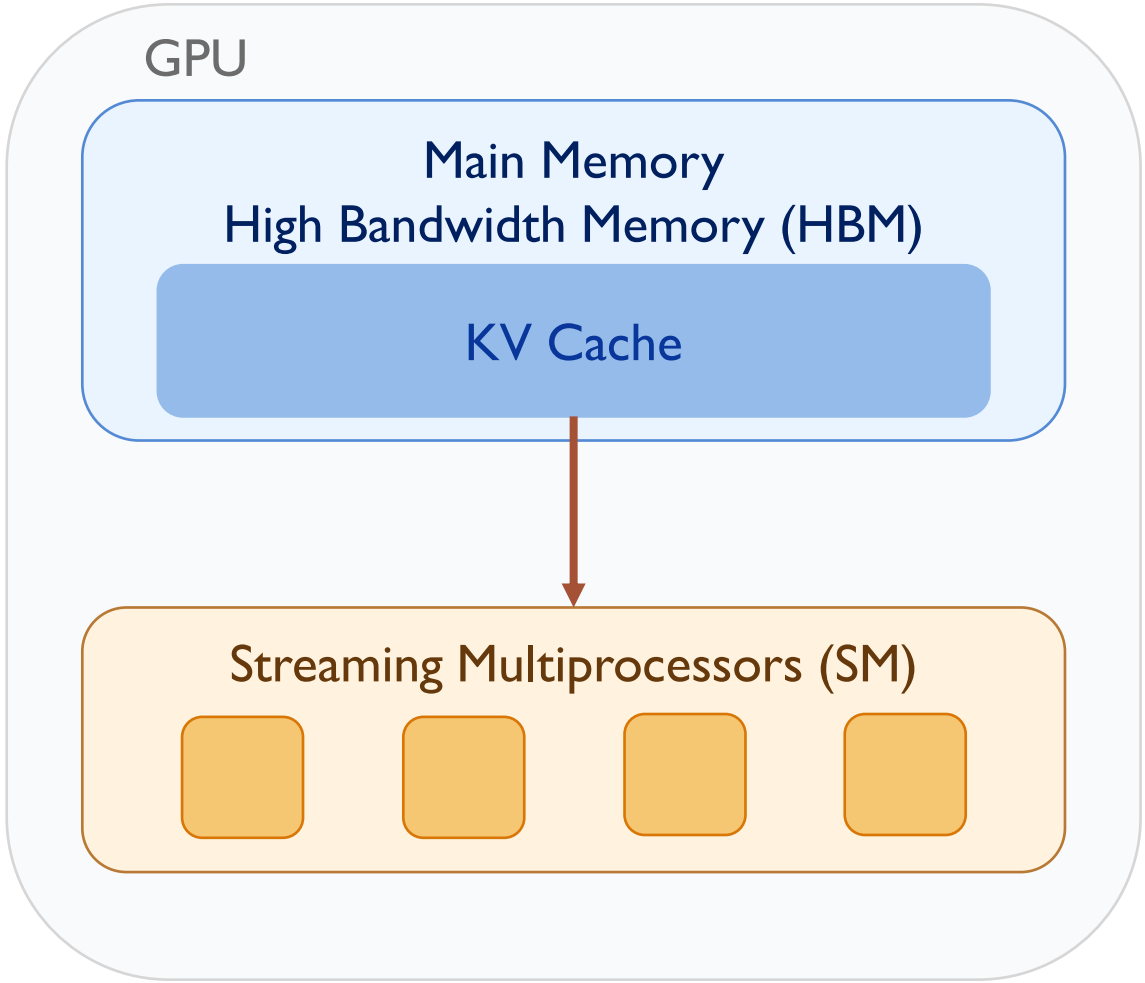
Long Document Q/A



Complex Agentic Workflows

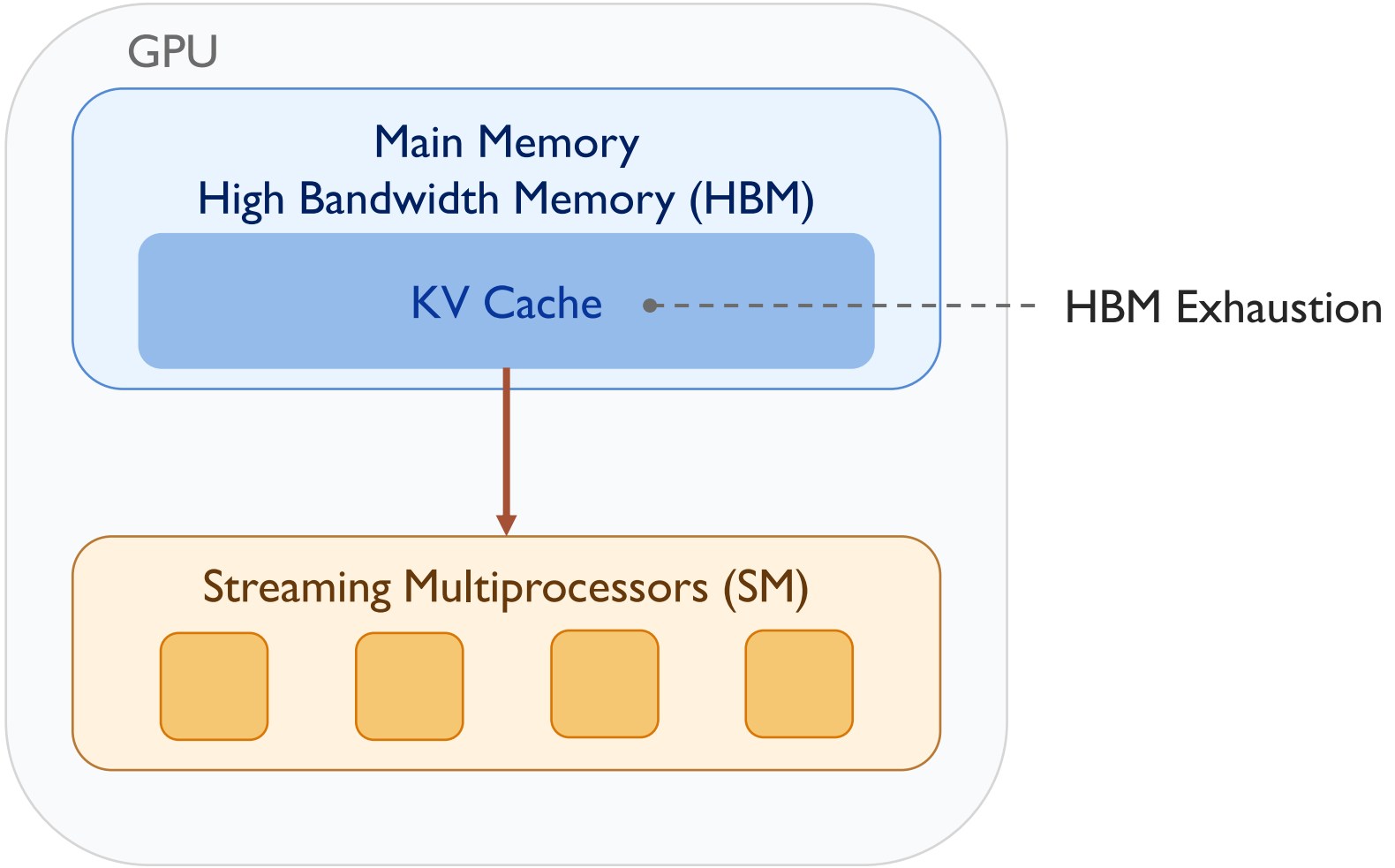
Long Context → Large KV Cache

Issues of large KV Cache



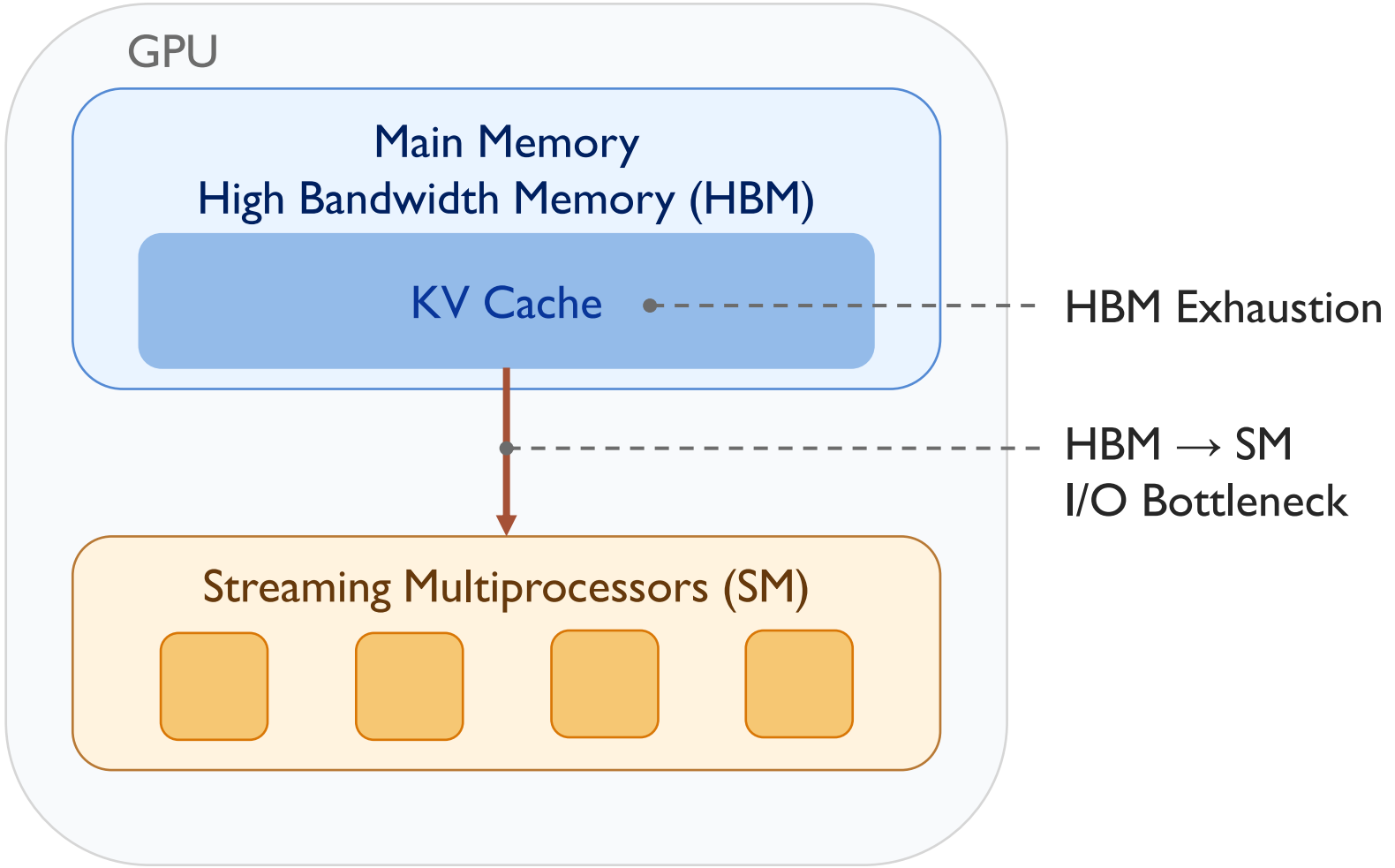
Long Context → Large KV Cache

Issues of large KV Cache



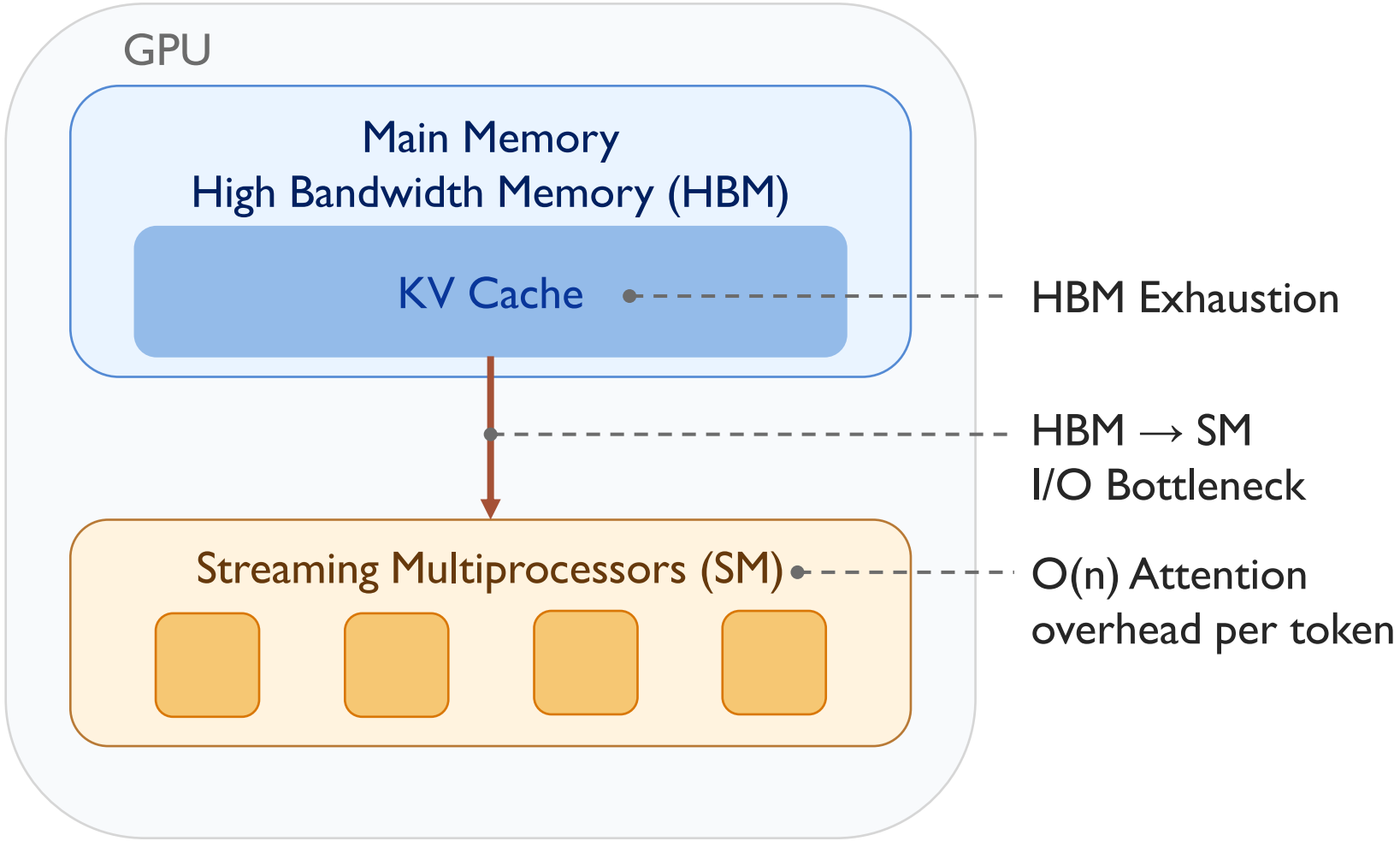
Long Context → Large KV Cache

Issues of large KV Cache



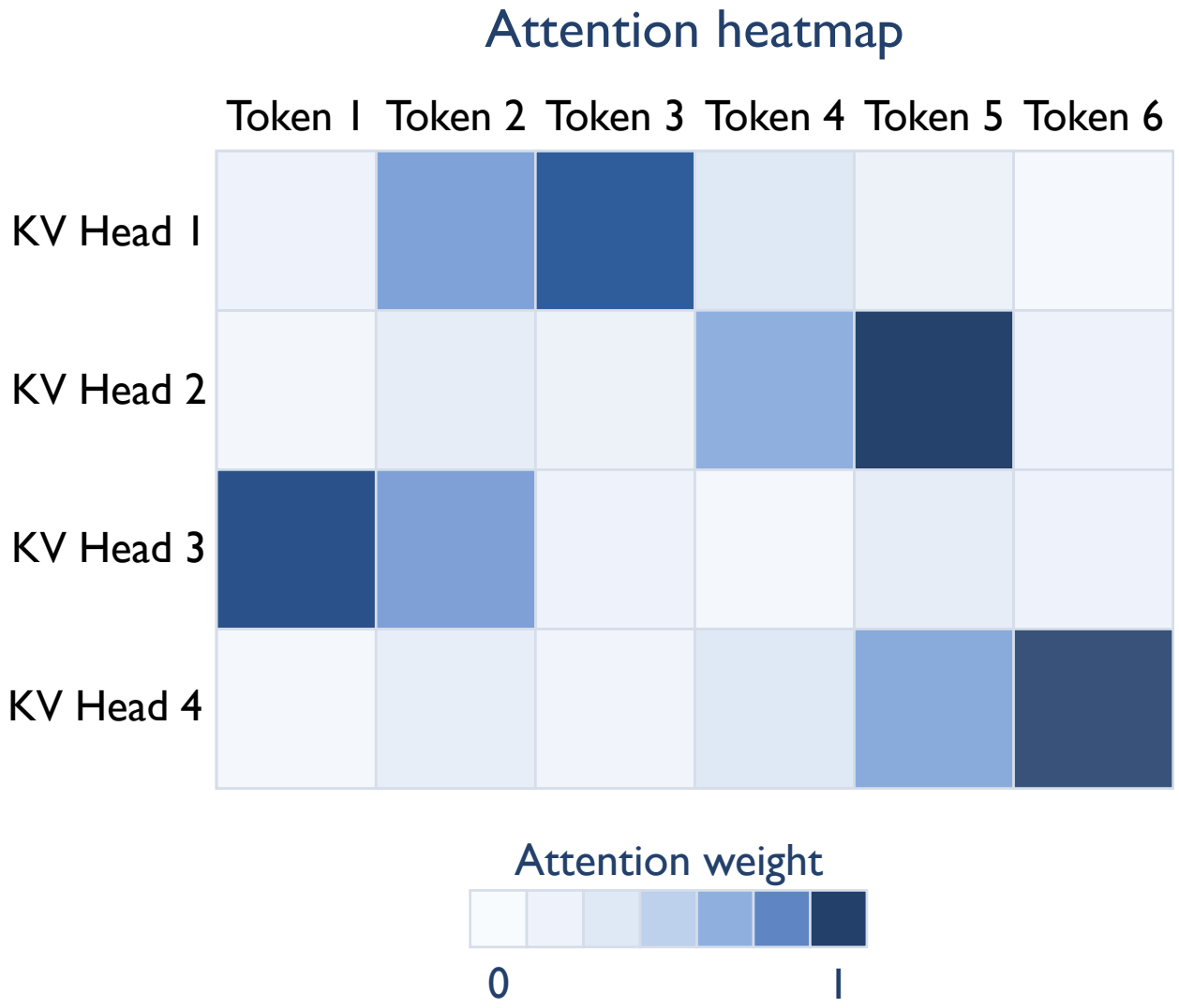
Long Context → Large KV Cache

Issues of large KV Cache

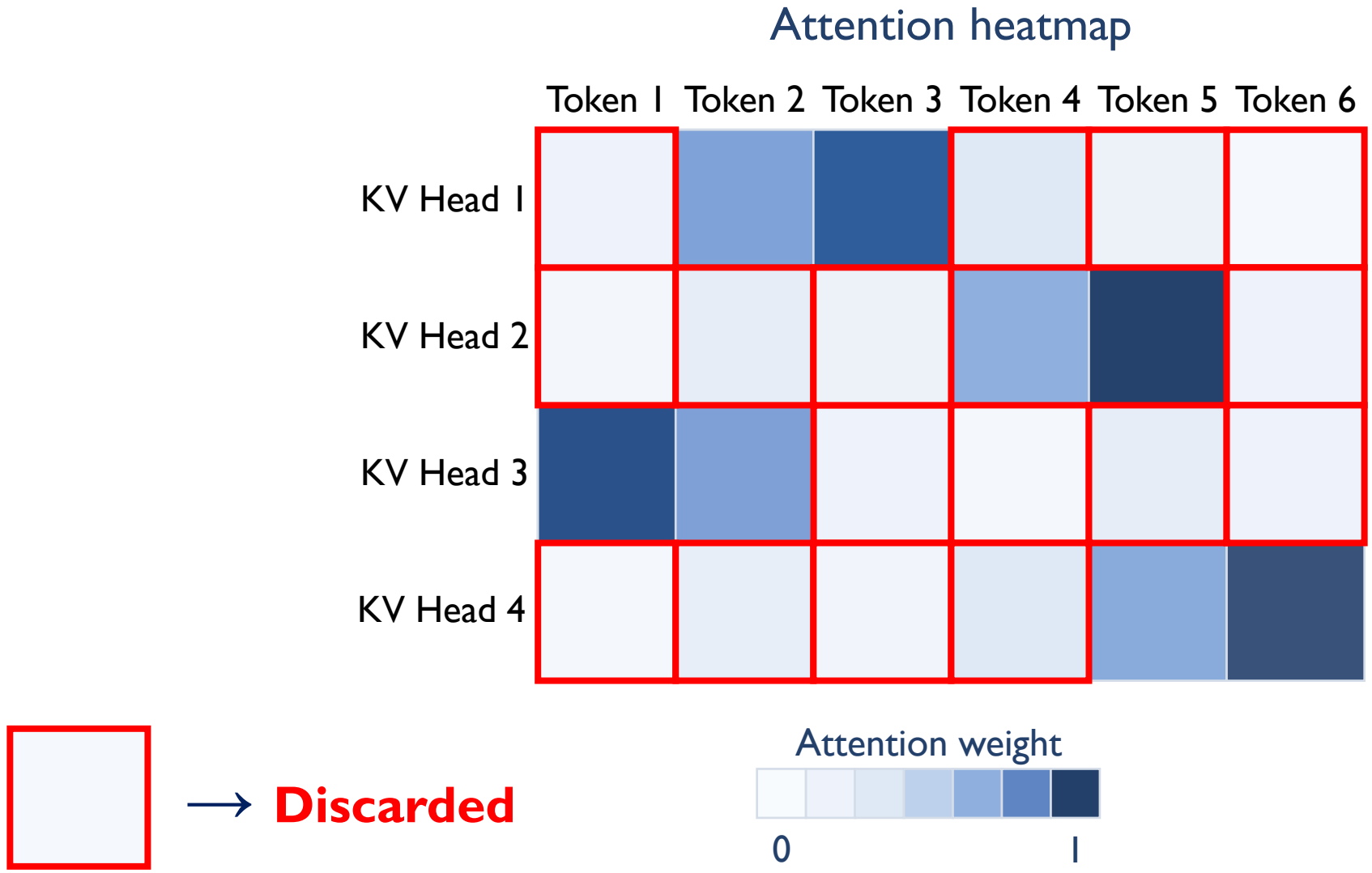


Attention is Sparse

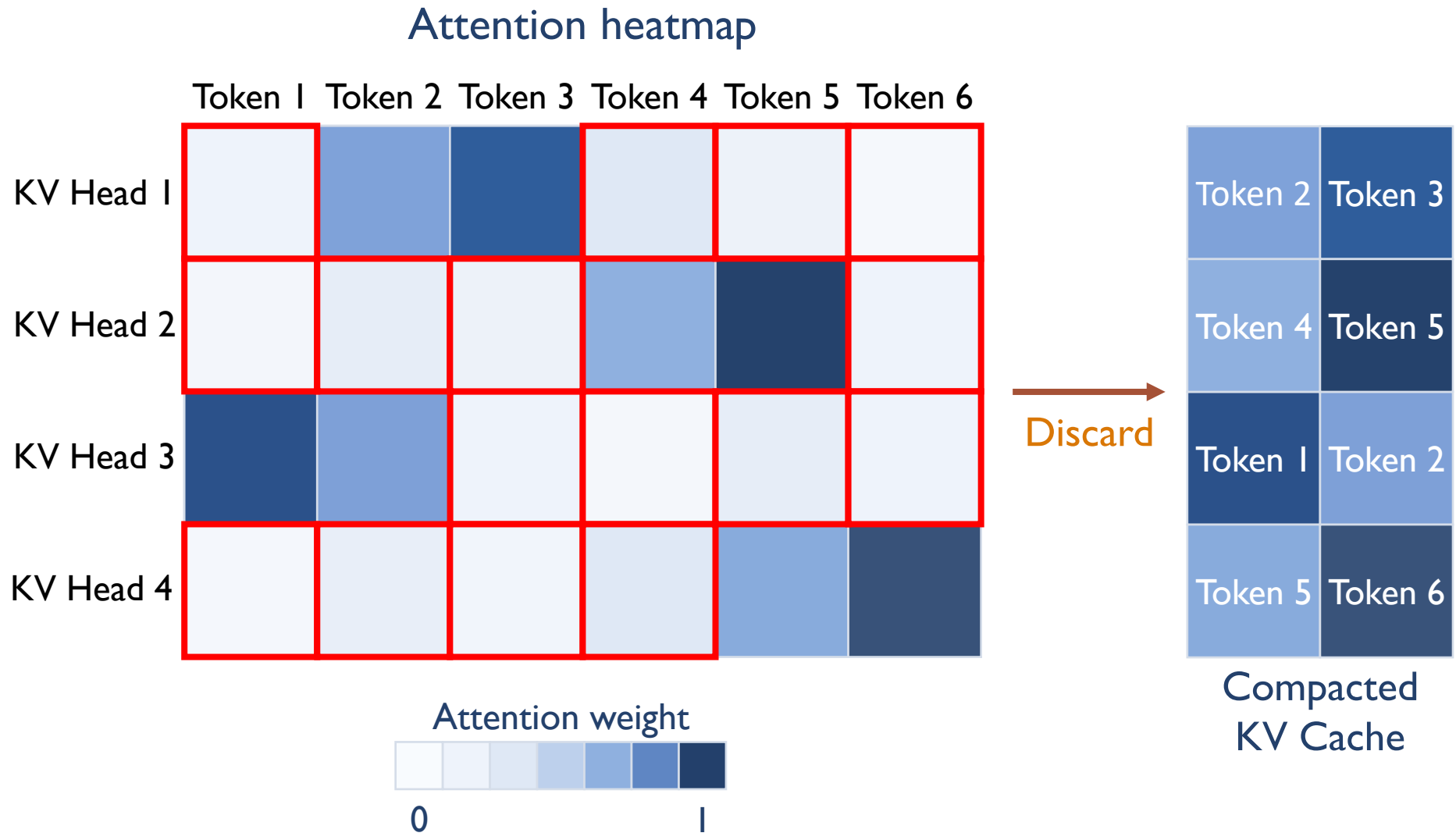
Each KV head focuses on a small subset of tokens



Leveraging Sparsity: Discard KV of Unimportant Tokens

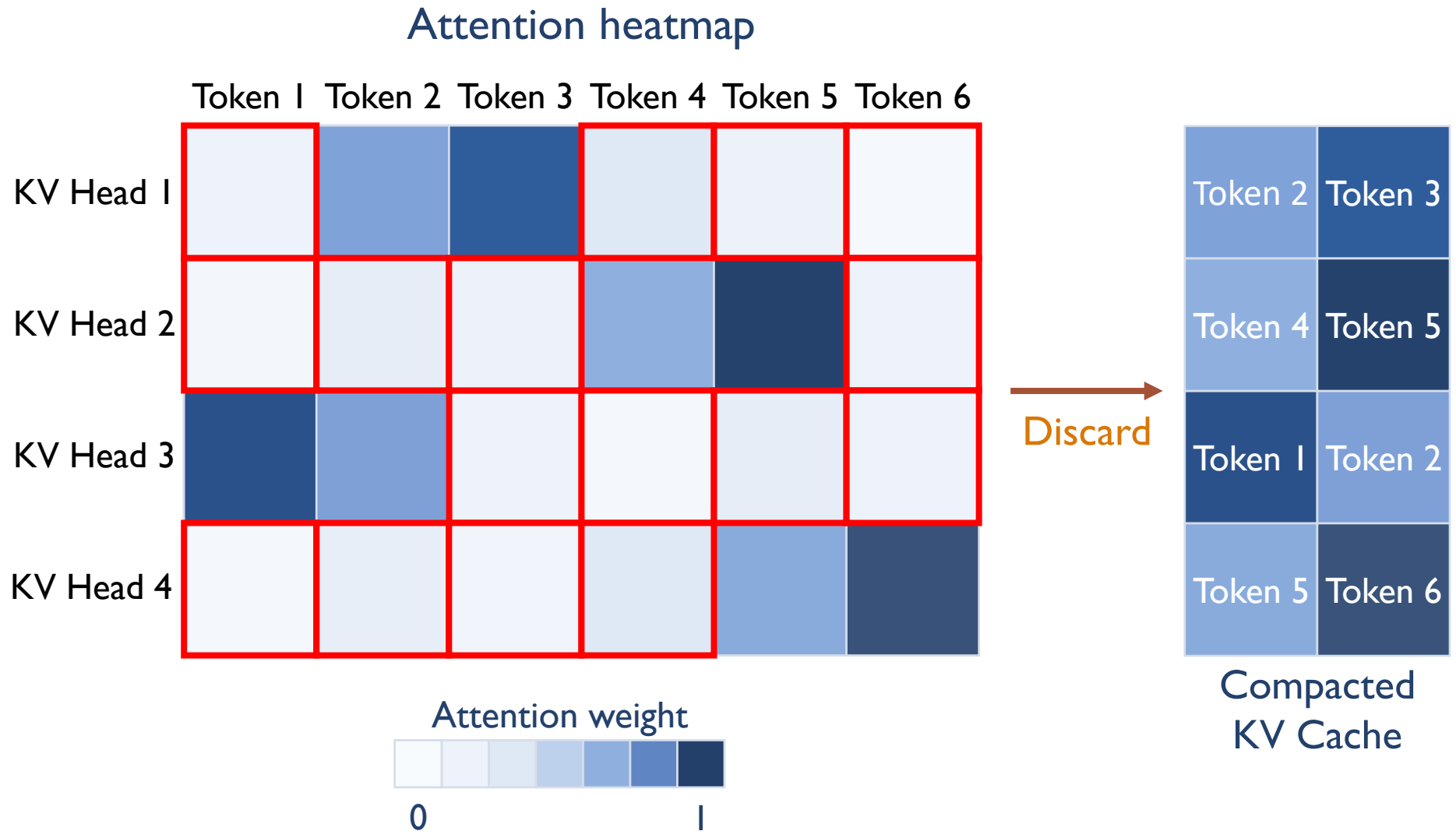


Leveraging Sparsity: Discard KV of Unimportant Tokens



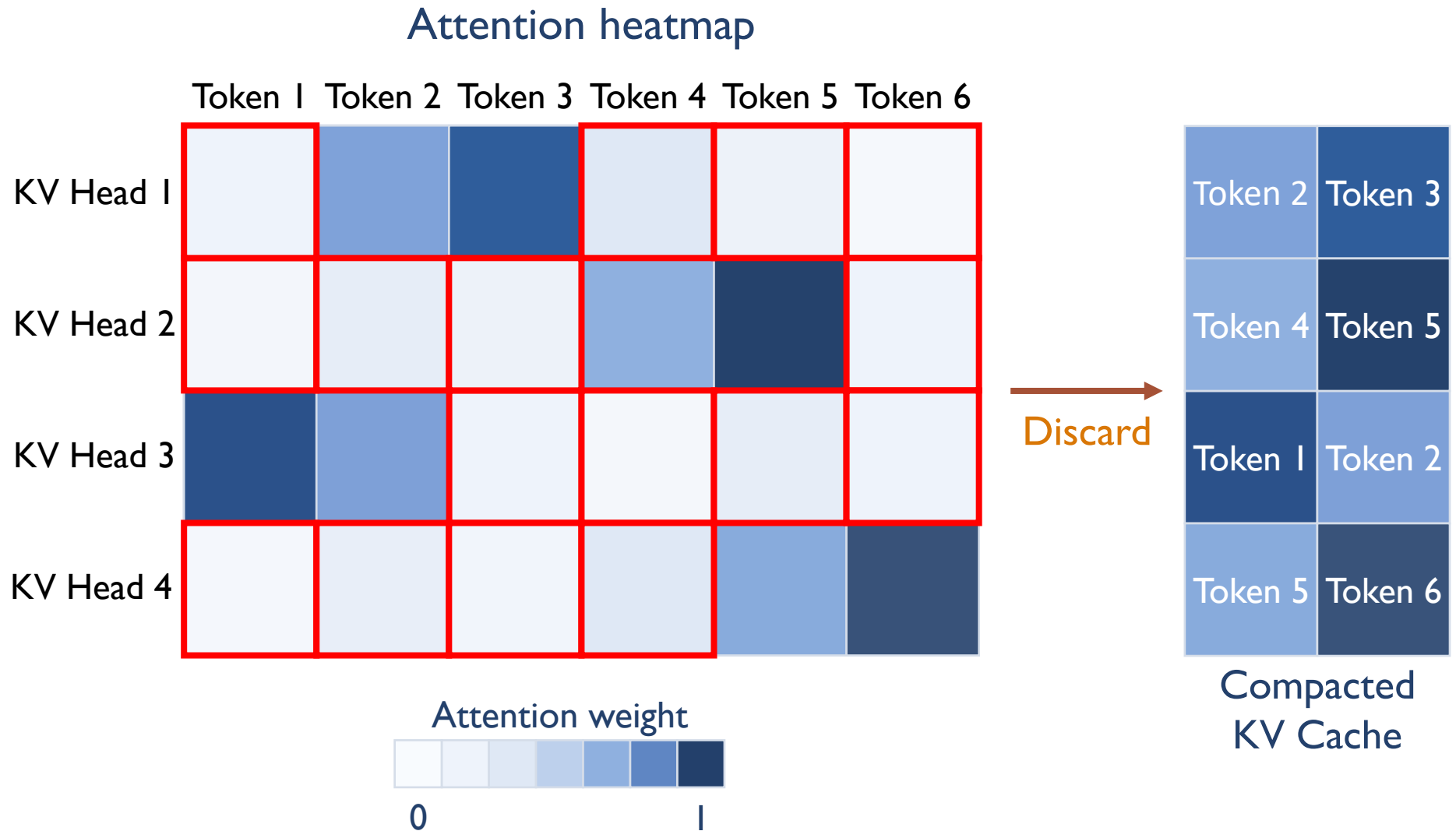
Leveraging Sparsity: Discard KV of Unimportant Tokens

- HBM usage ↓



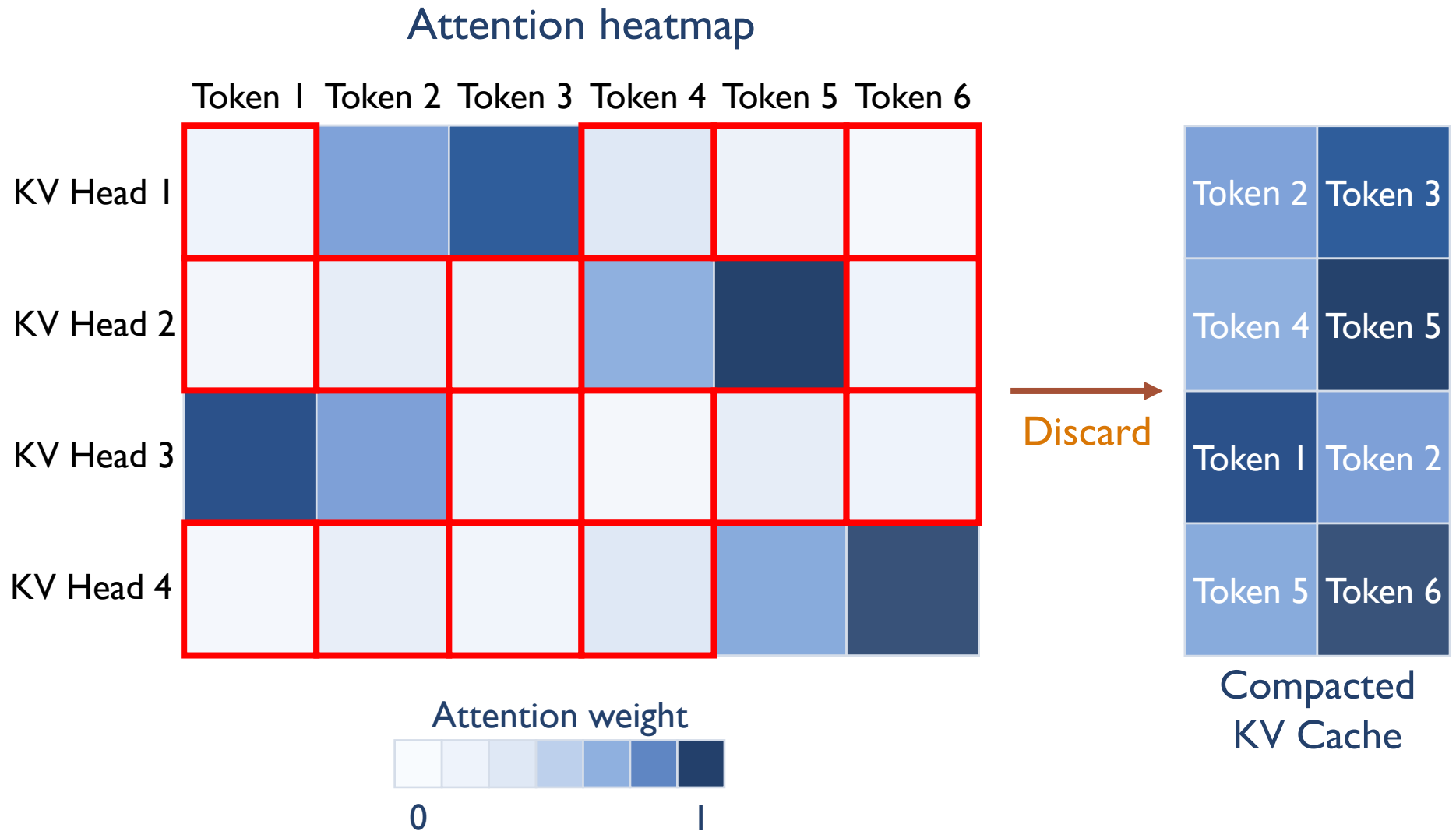
Leveraging Sparsity: Discard KV of Unimportant Tokens

- HBM usage ↓
- HBM → SM I/O ↓

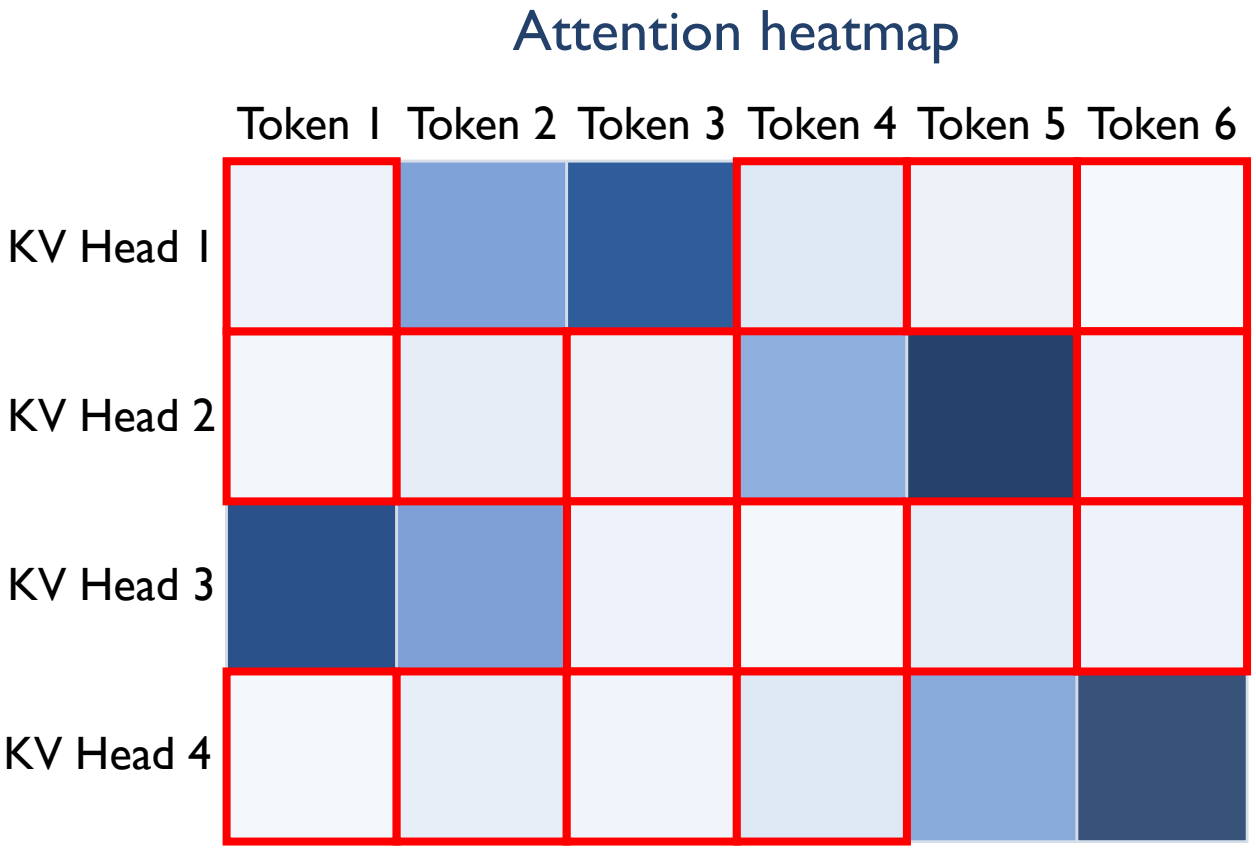


Leveraging Sparsity: Discard KV of Unimportant Tokens

- HBM usage ↓
- HBM → SM I/O ↓
- Attention Compute ↓

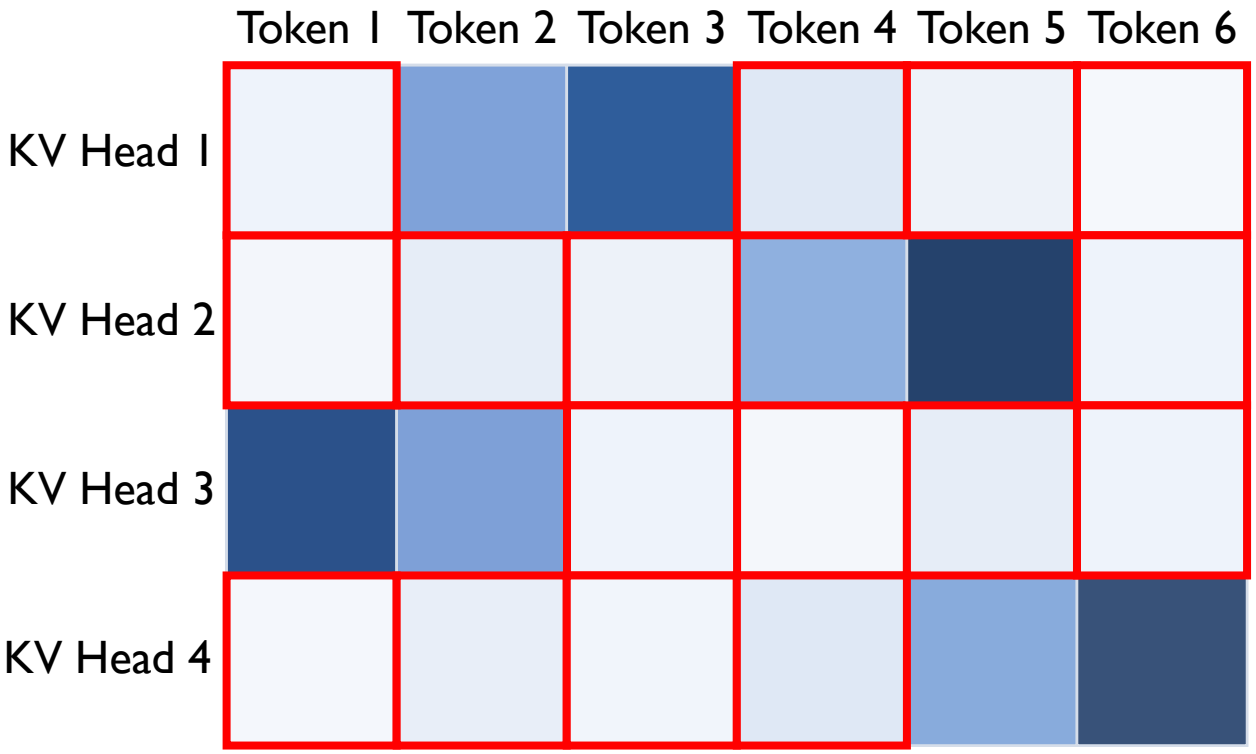


Problem: Discarding is irreversible

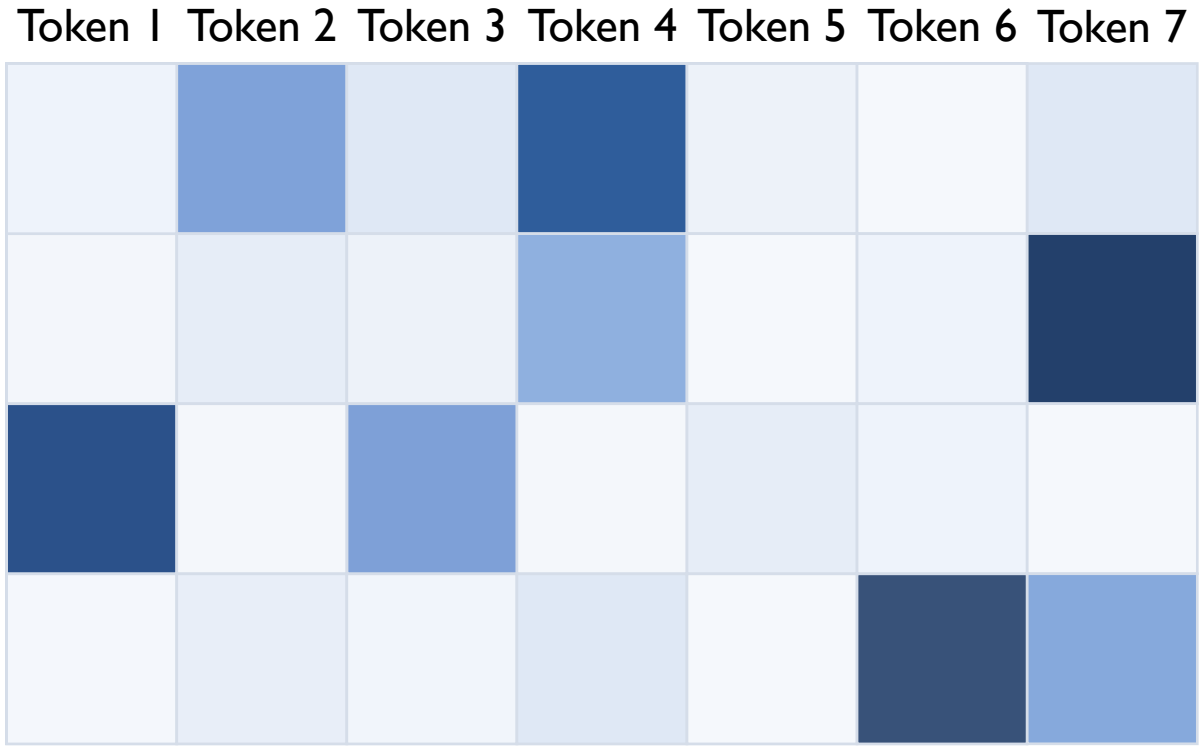


Problem: Discarding is irreversible

Attention heatmap

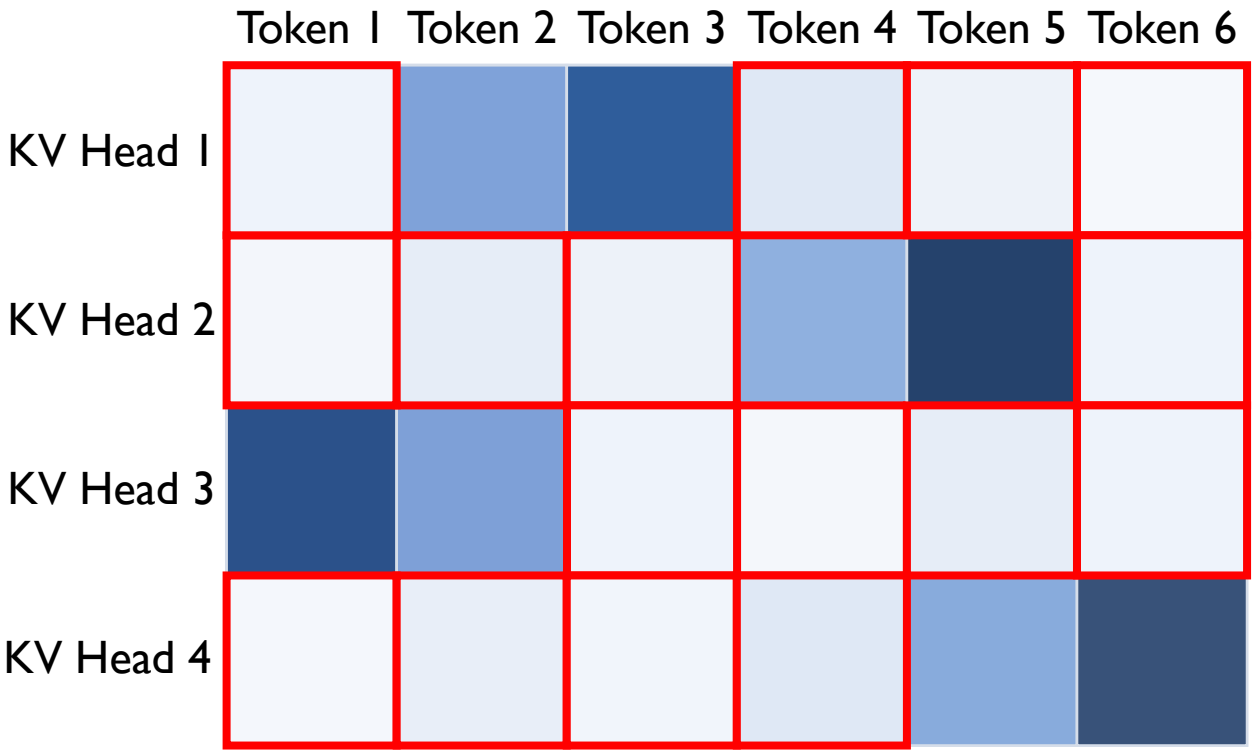


Attention heatmap for next decode step

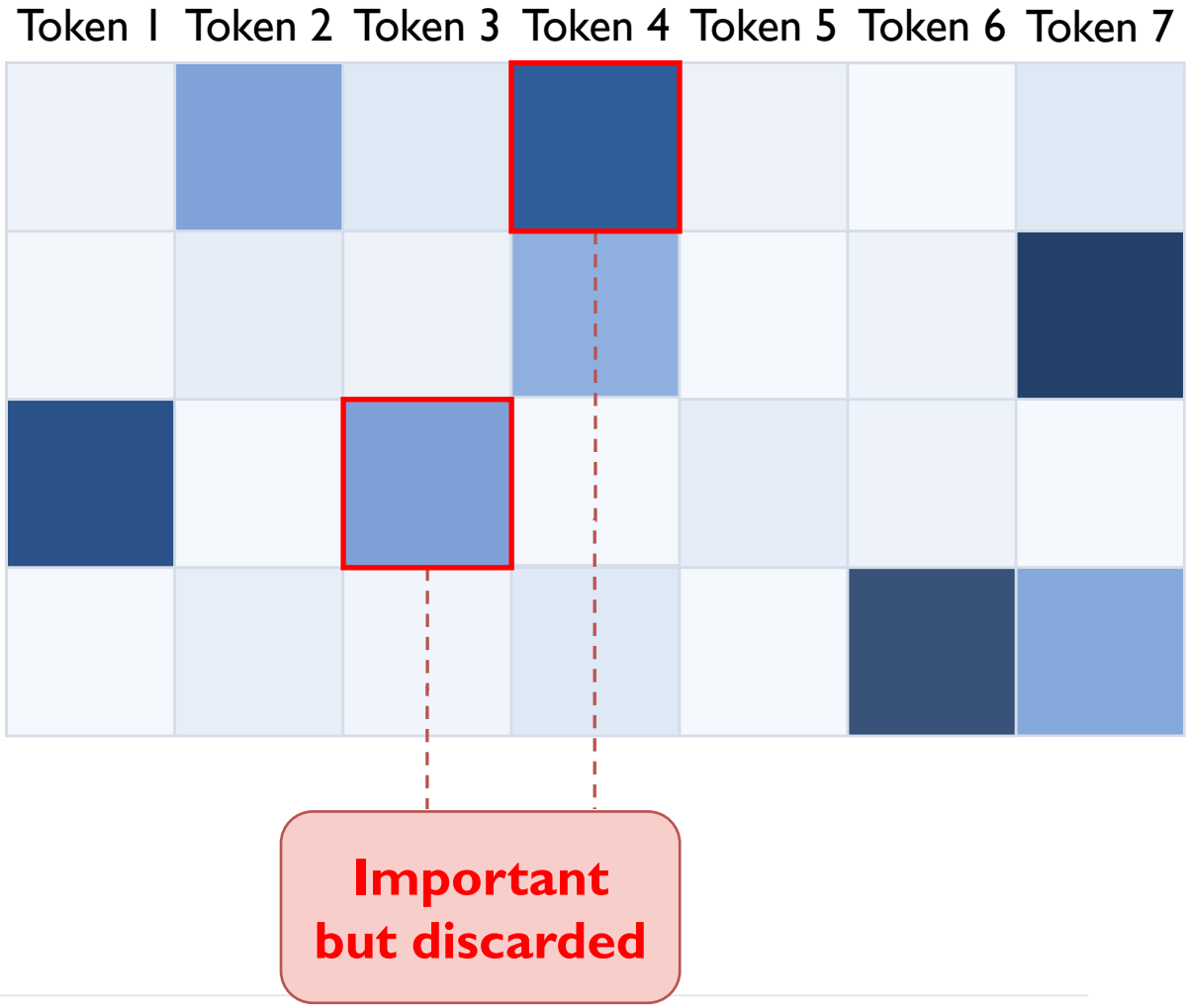


Problem: Discarding is irreversible

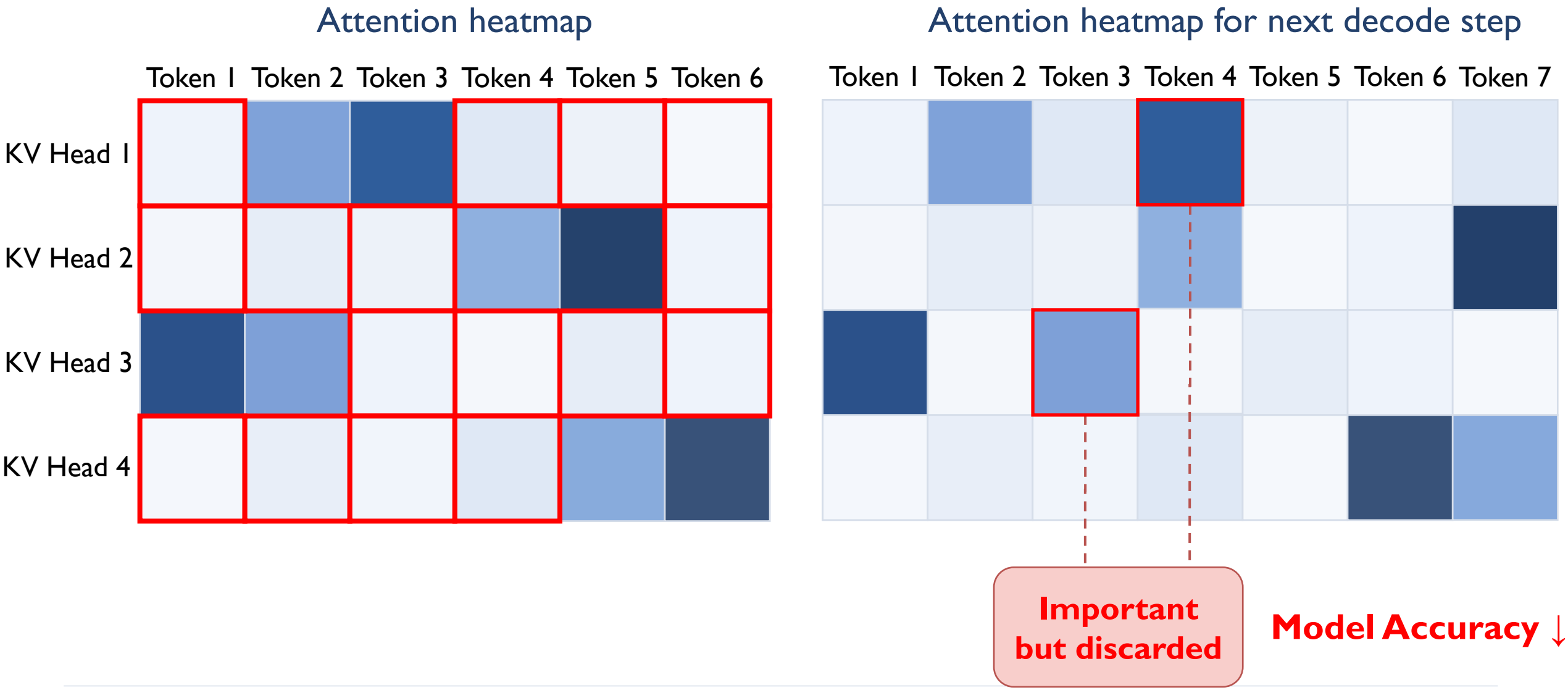
Attention heatmap



Attention heatmap for next decode step

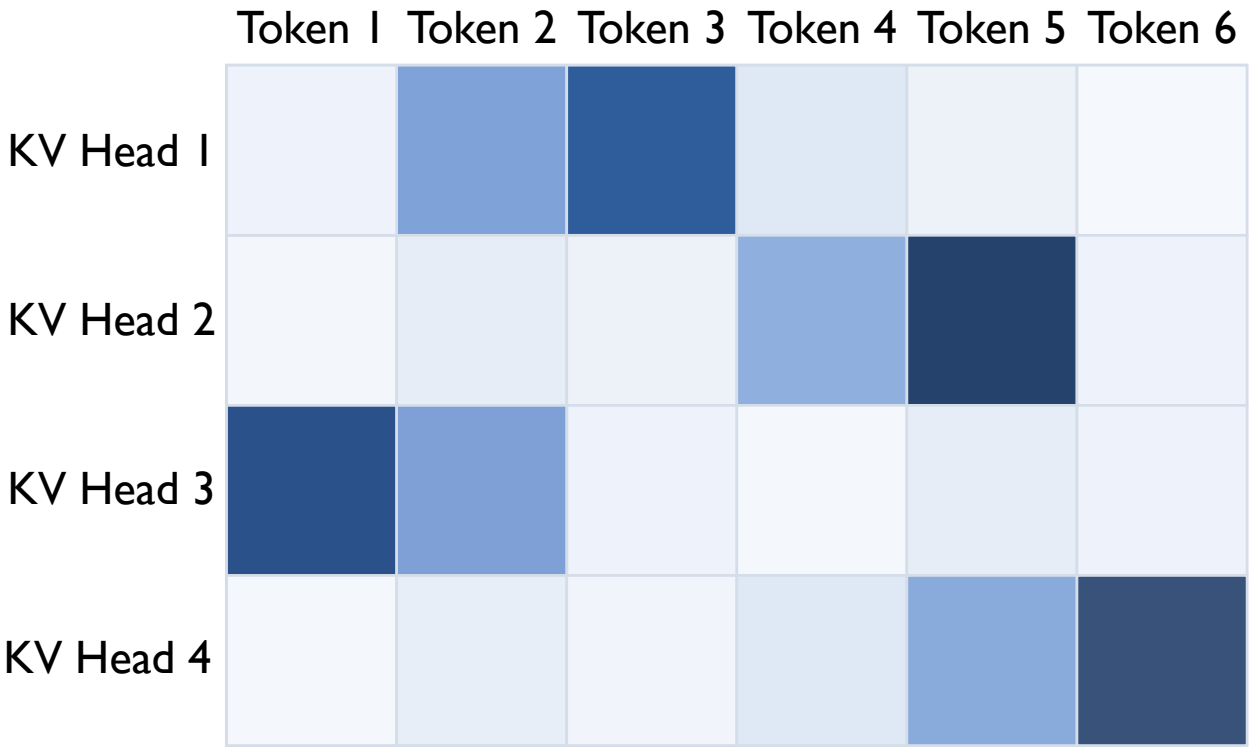


Problem: Discarding is irreversible



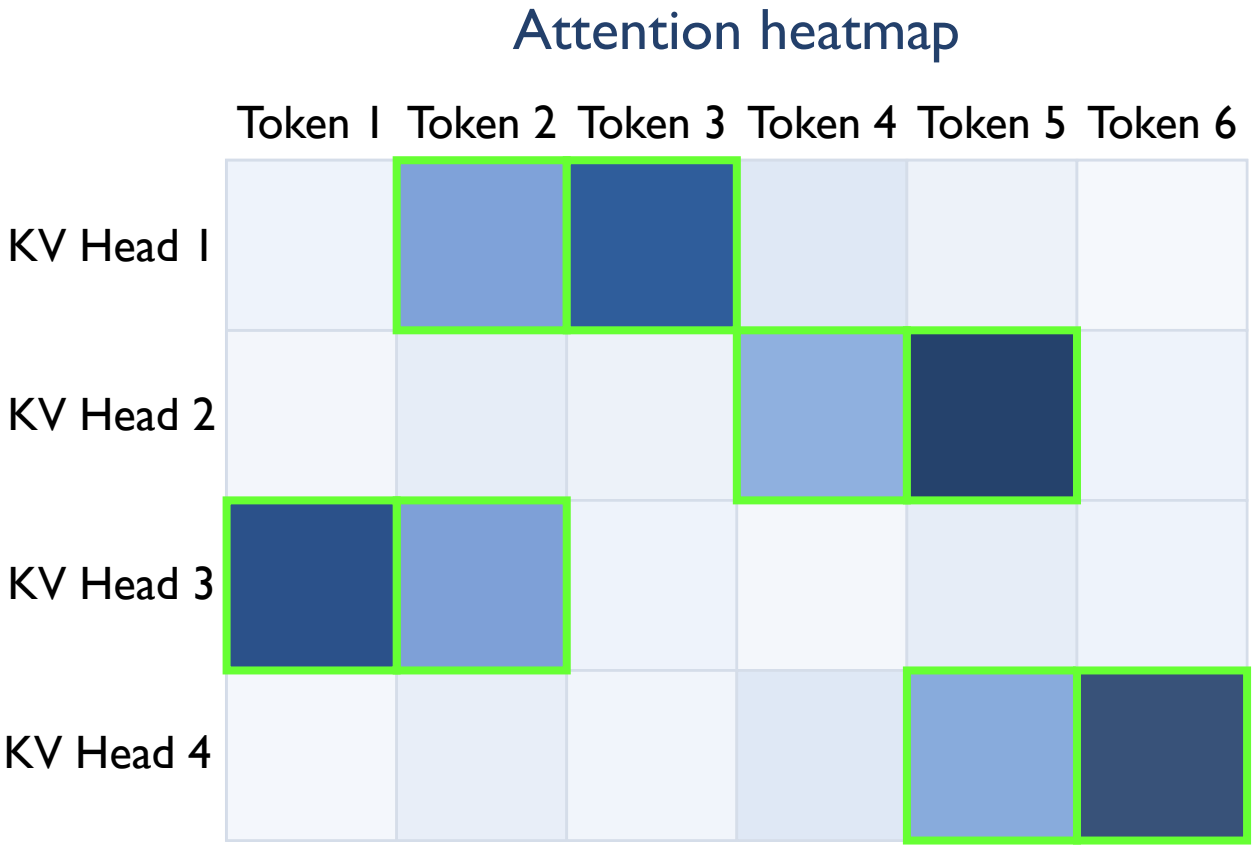
Leveraging Sparsity: Query Aware KV Selection

Attention heatmap



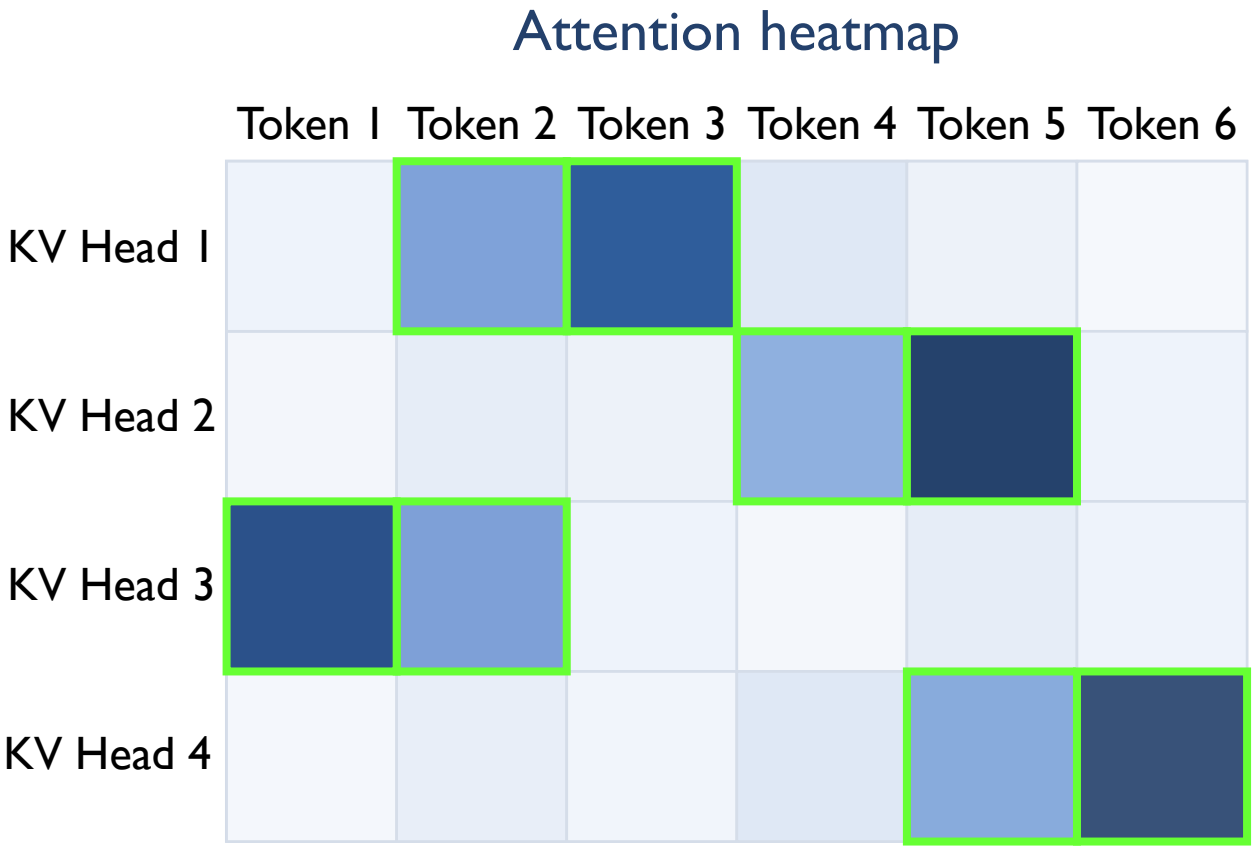
At each decode step, the current query vector is used to select the most important KV

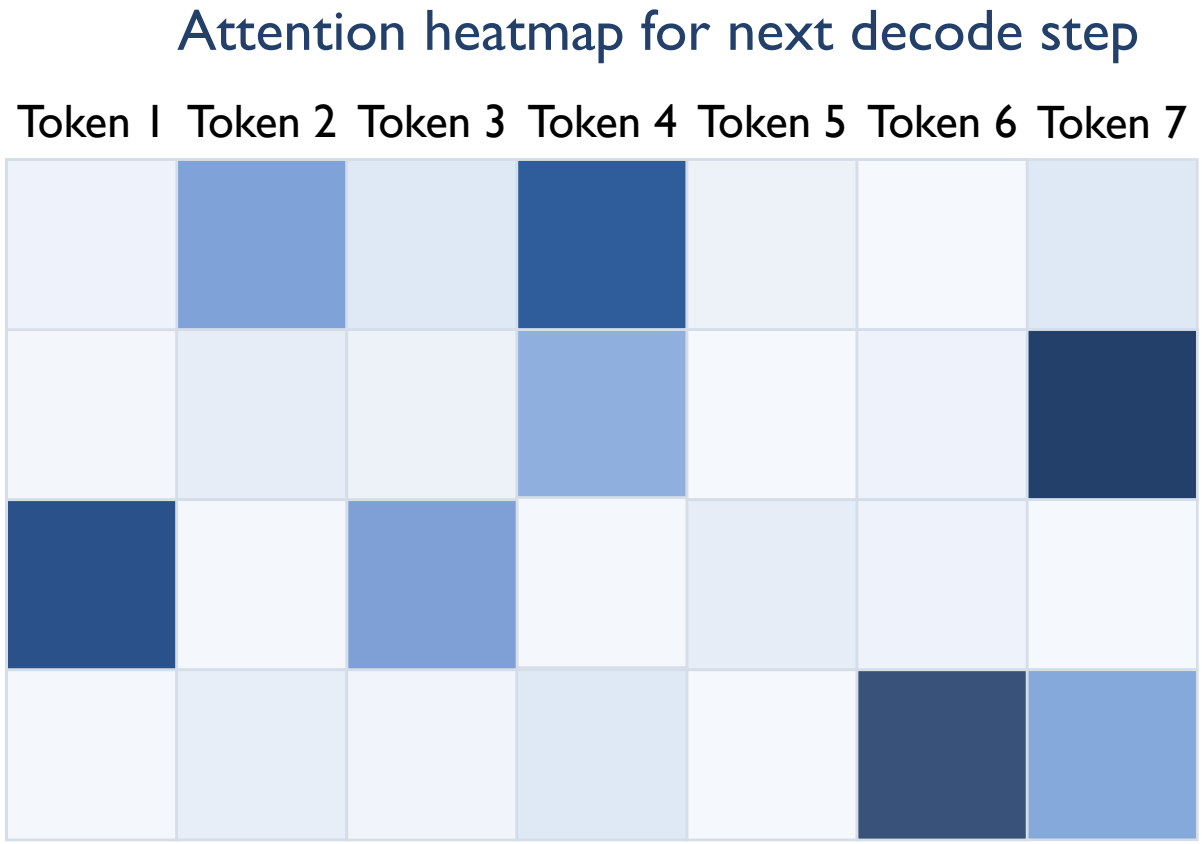
Leveraging Sparsity: Query Aware KV Selection



At each decode step, the current query vector is used to select the most important KV

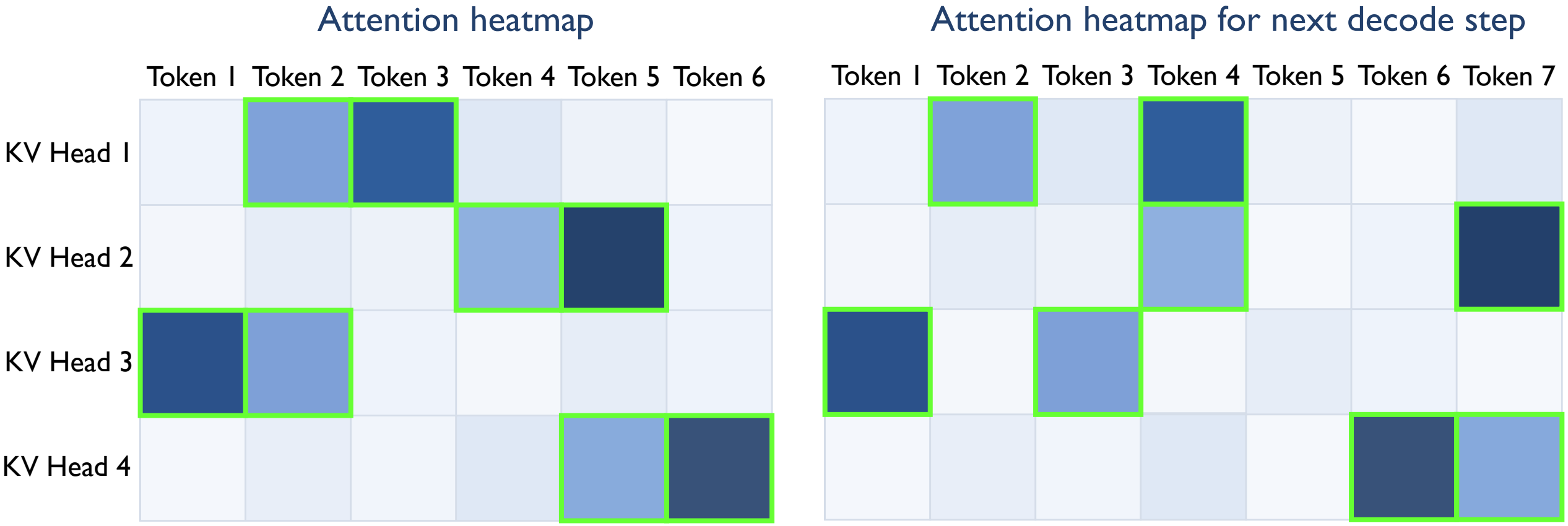
Leveraging Sparsity: Query Aware KV Selection





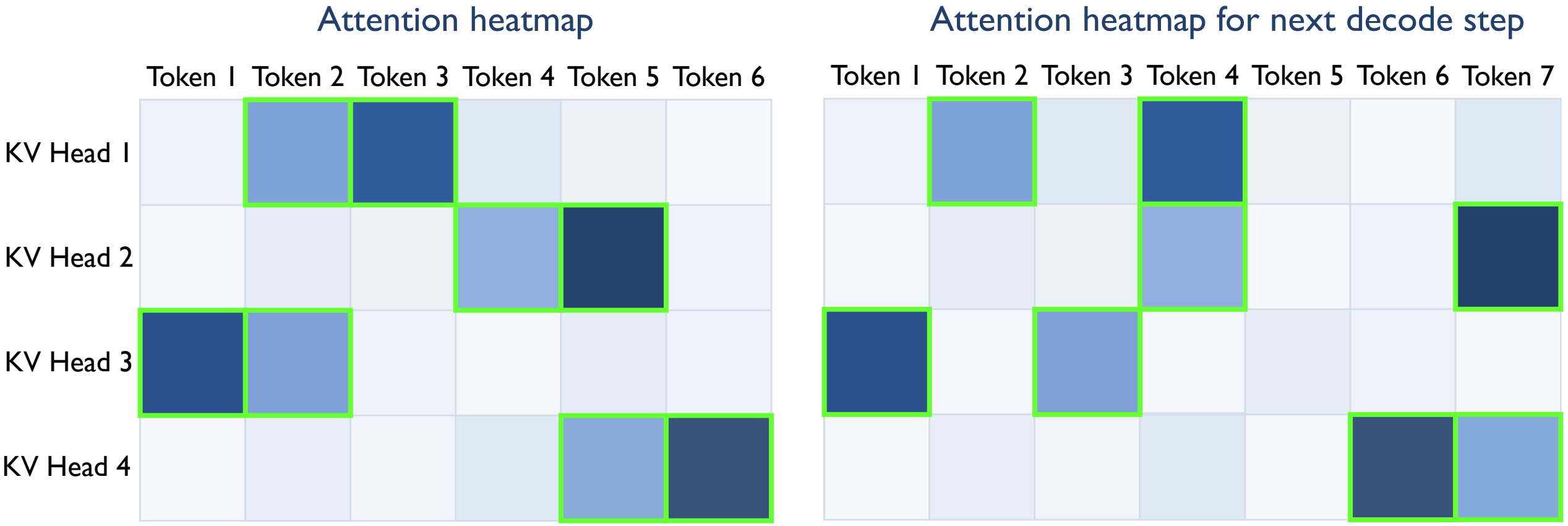
At each decode step, the current query vector is used to select the most important KV

Leveraging Sparsity: Query Aware KV Selection



At each decode step, the current query vector is used to select the most important KV

Leveraging Sparsity: Query Aware KV Selection



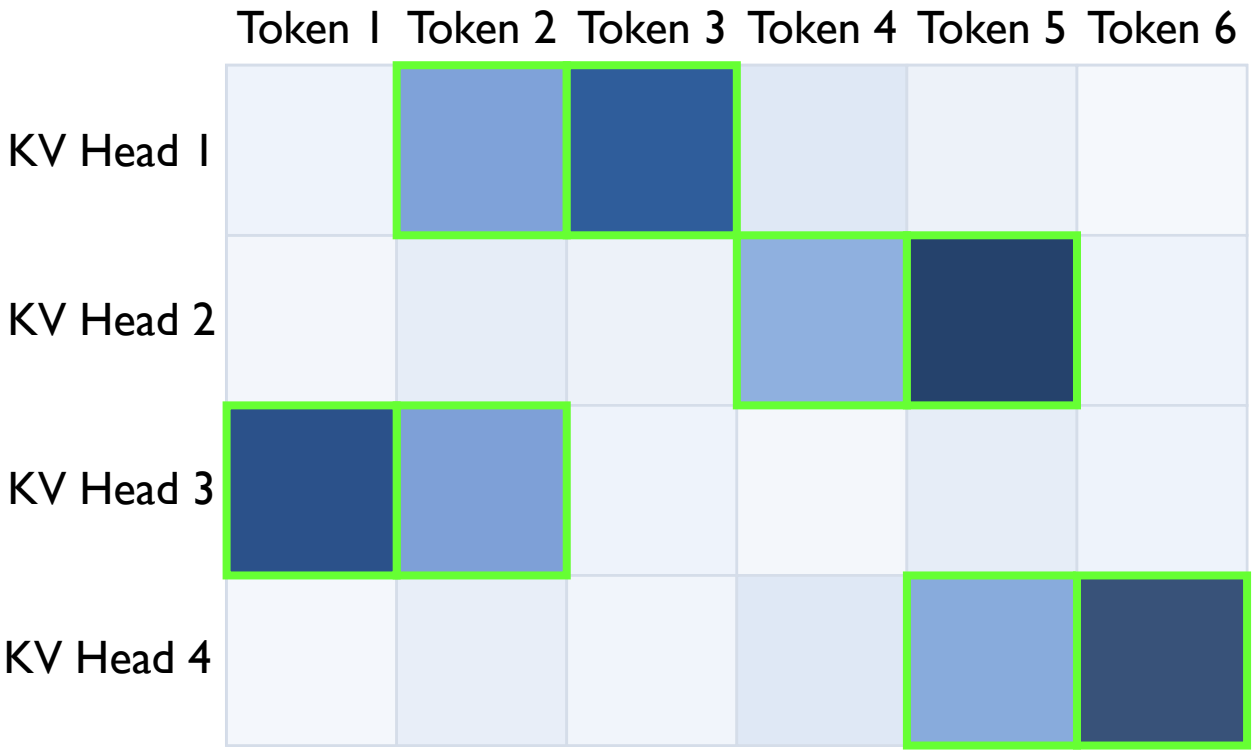
Model Accuracy Preserved

HBM → SM I/O ↓

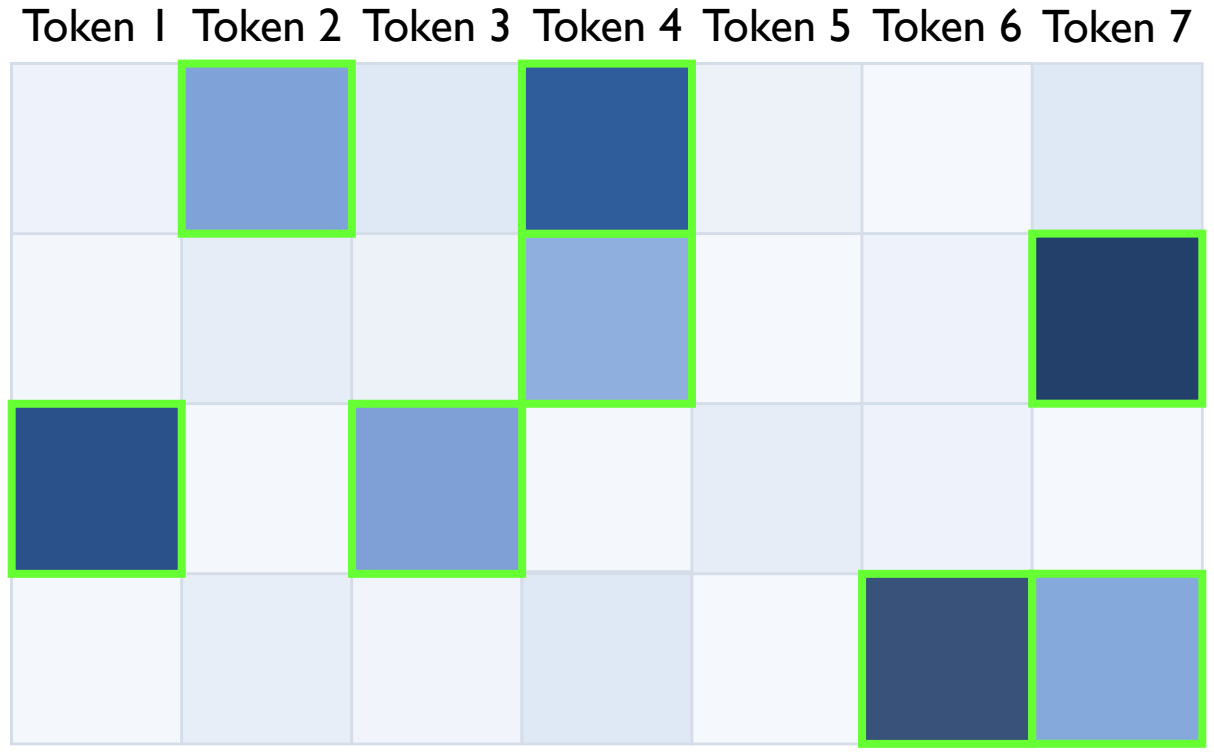
Attention Compute ↓

Leveraging Sparsity: Query Aware KV Selection

Attention heatmap

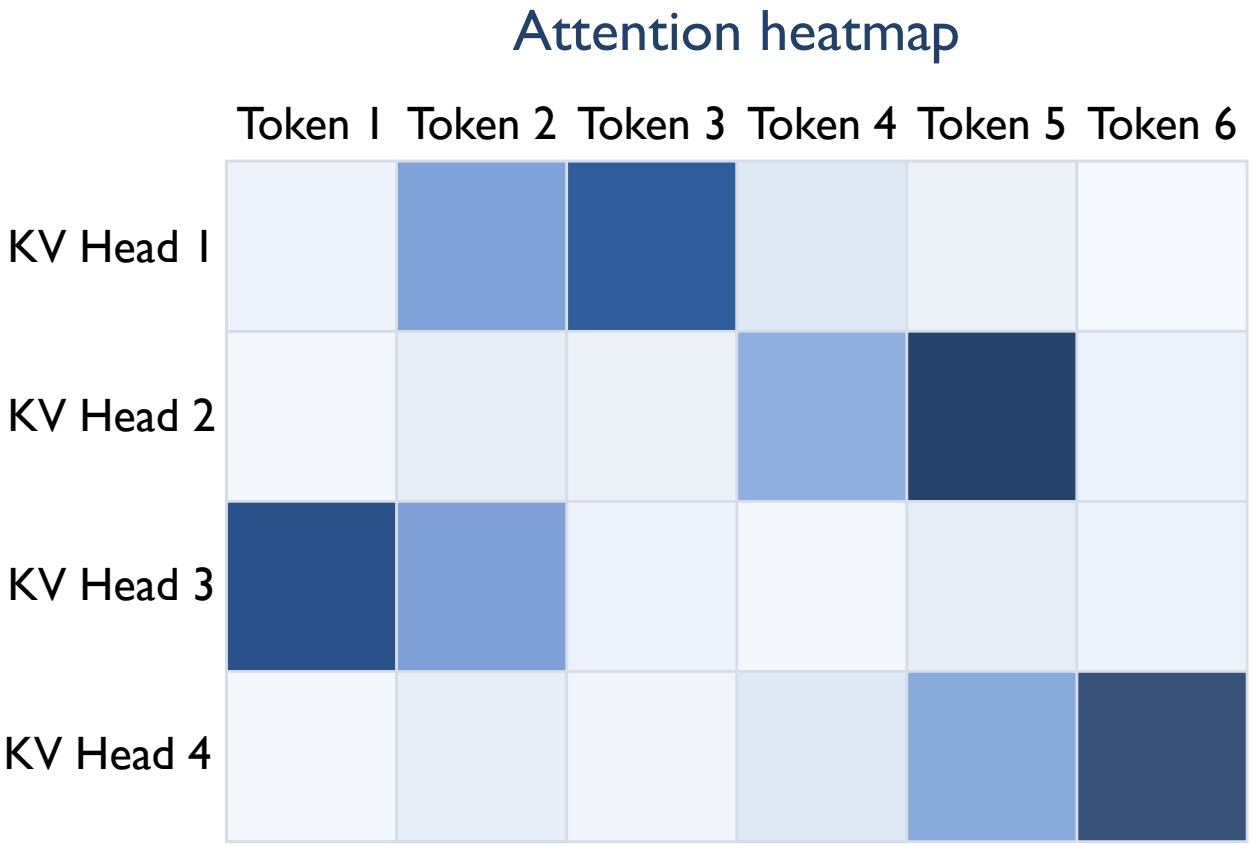


Attention heatmap for next decode step

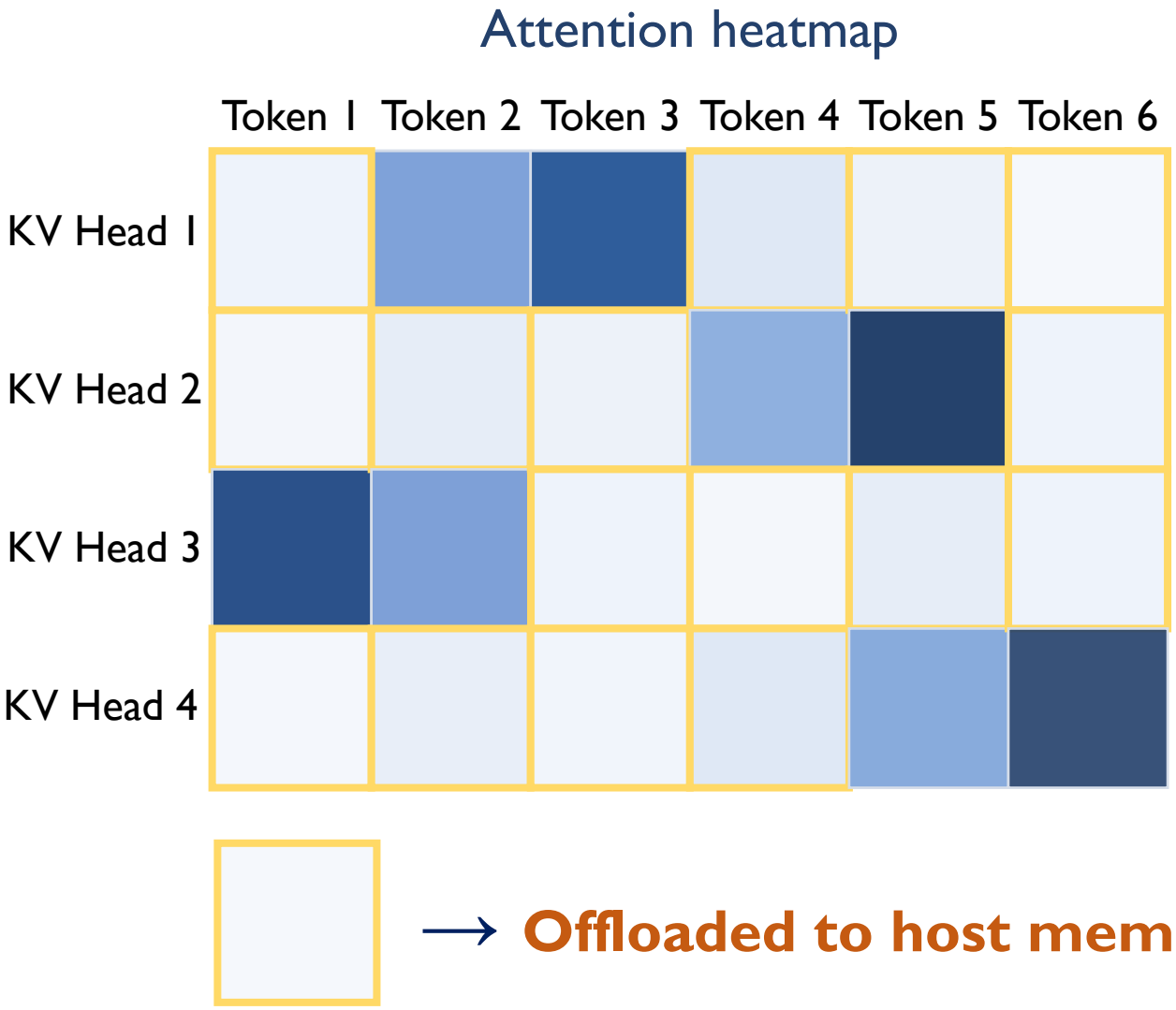


HBM usage unchanged

Leveraging Sparsity: Offloading + Query Aware Selection

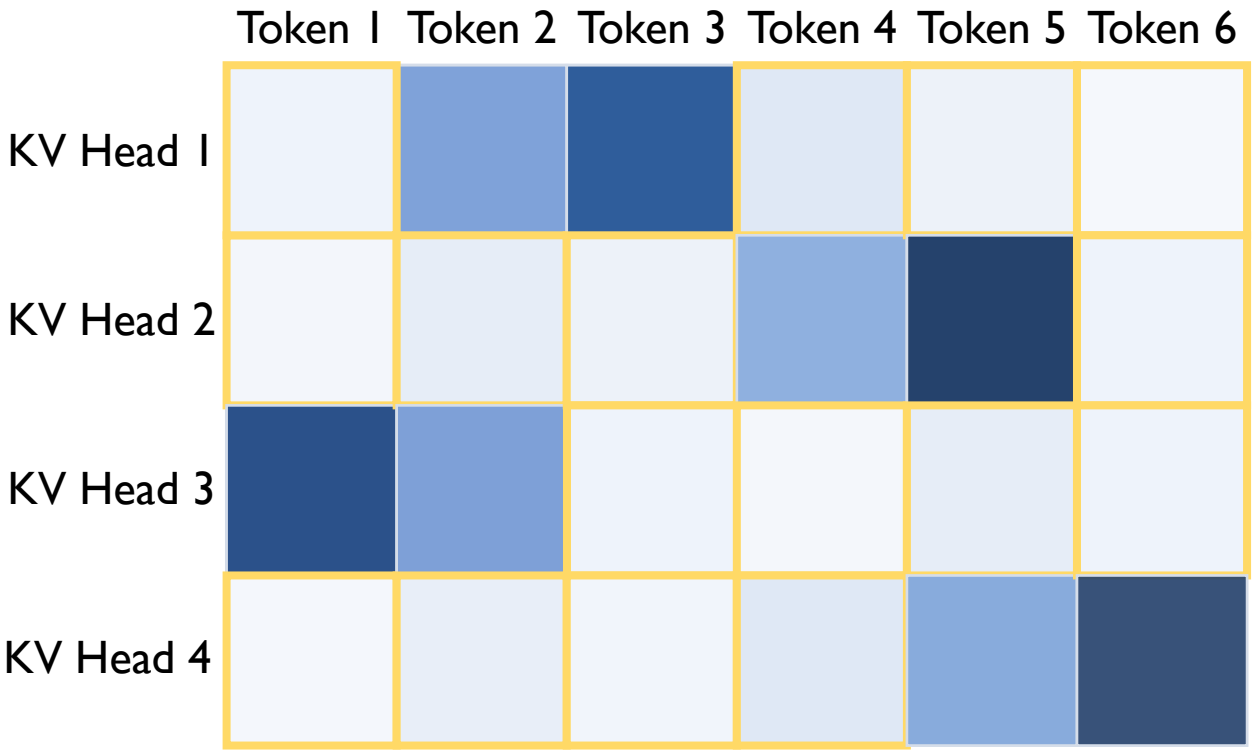


Leveraging Sparsity: Offloading + Query Aware Selection

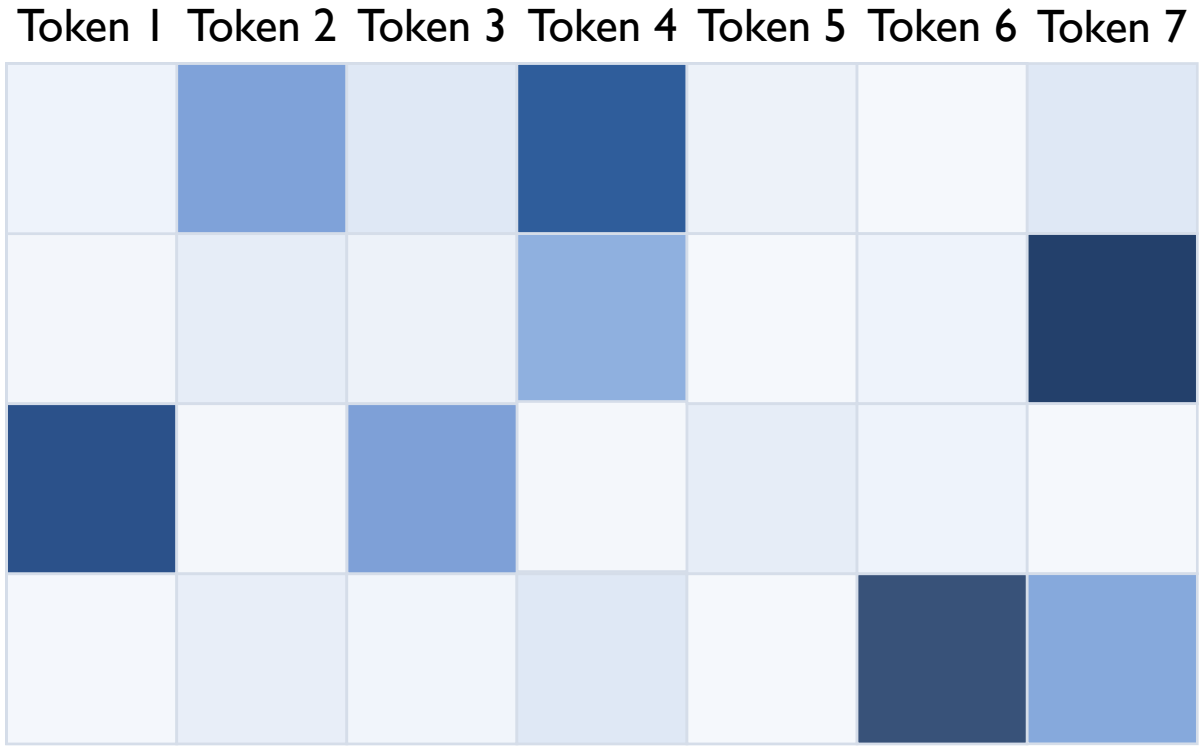


Leveraging Sparsity: Offloading + Query Aware Selection

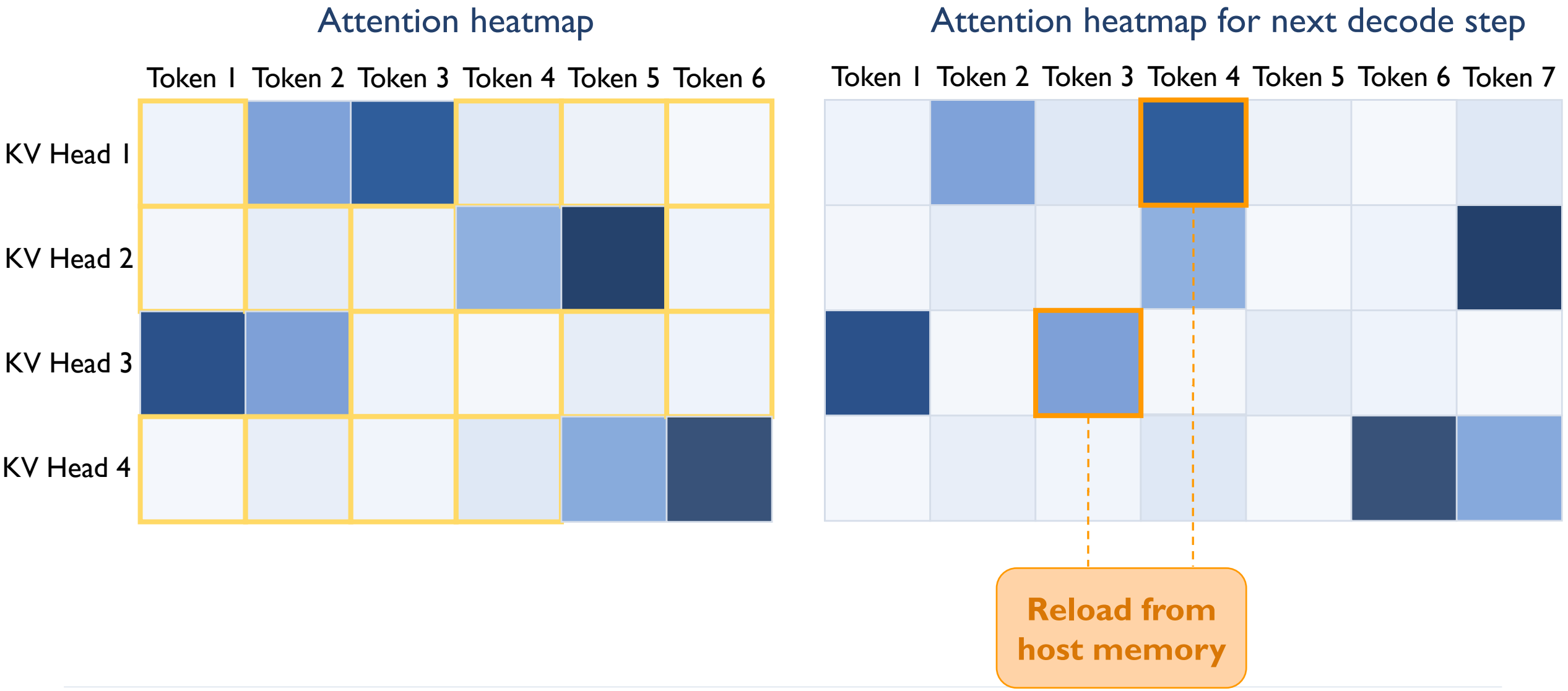
Attention heatmap



Attention heatmap for next decode step

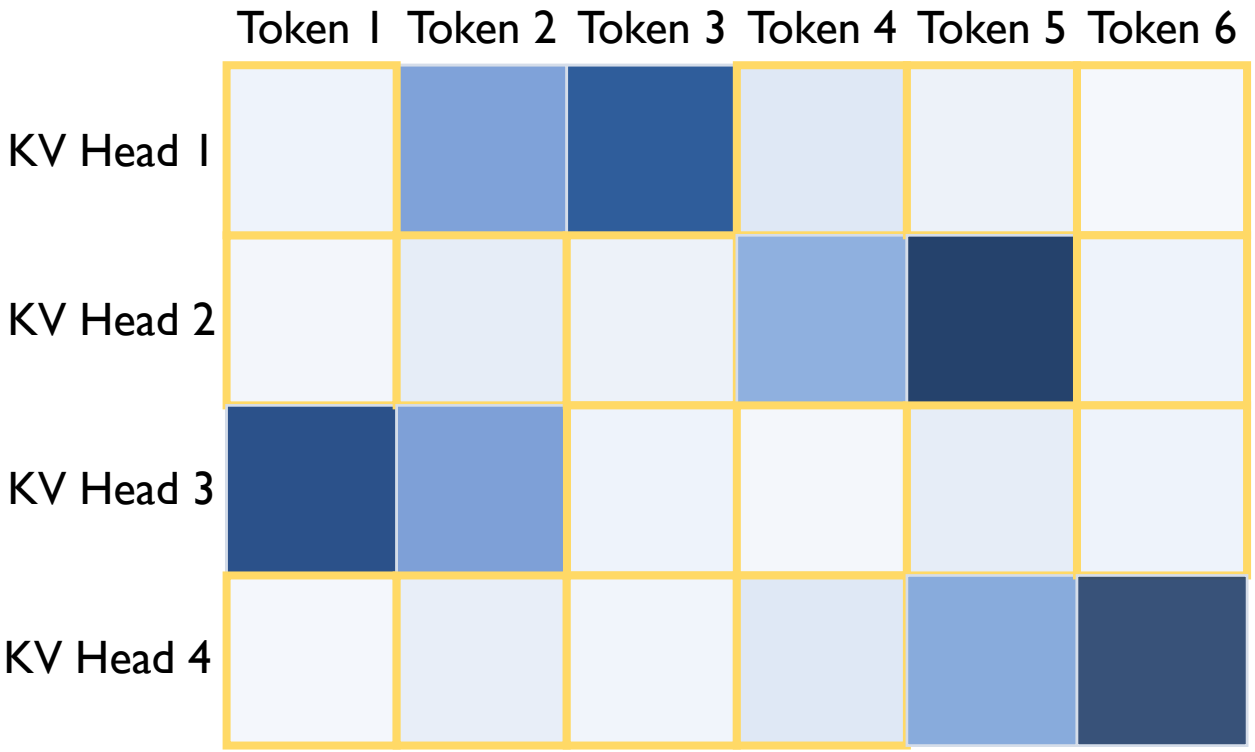


Leveraging Sparsity: Offloading + Query Aware Selection

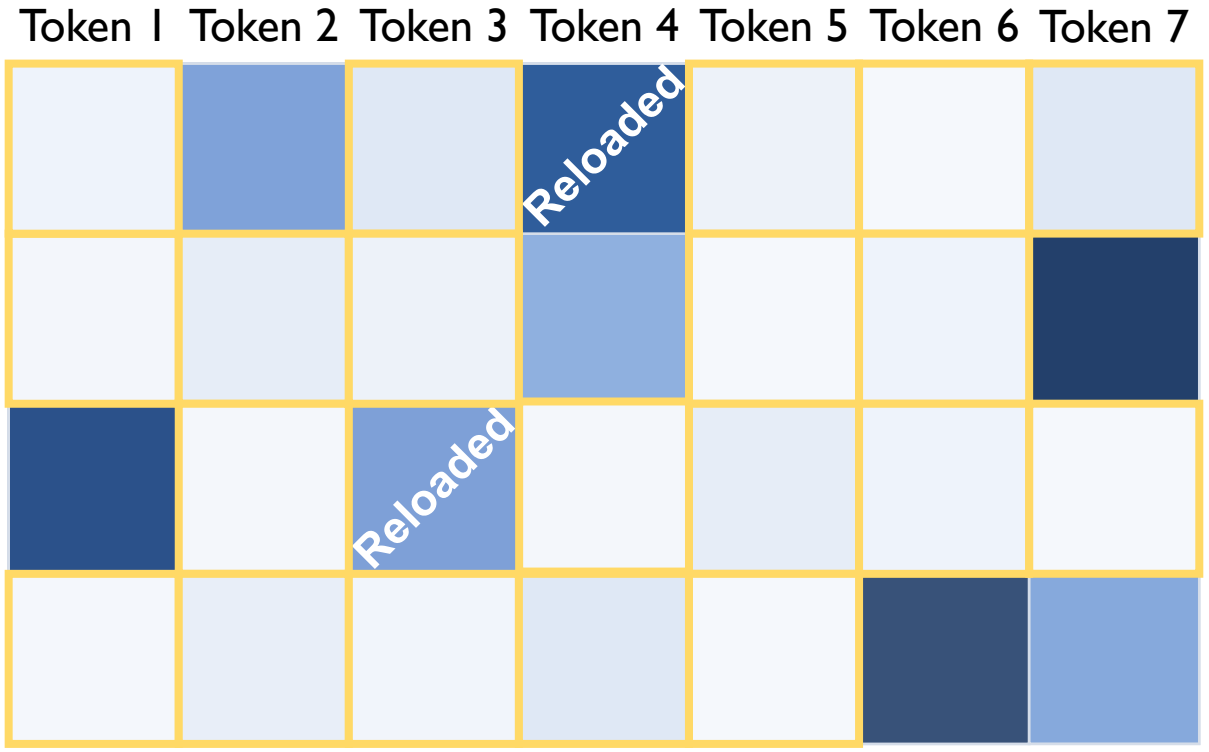


Leveraging Sparsity: Offloading + Query Aware Selection

Attention heatmap

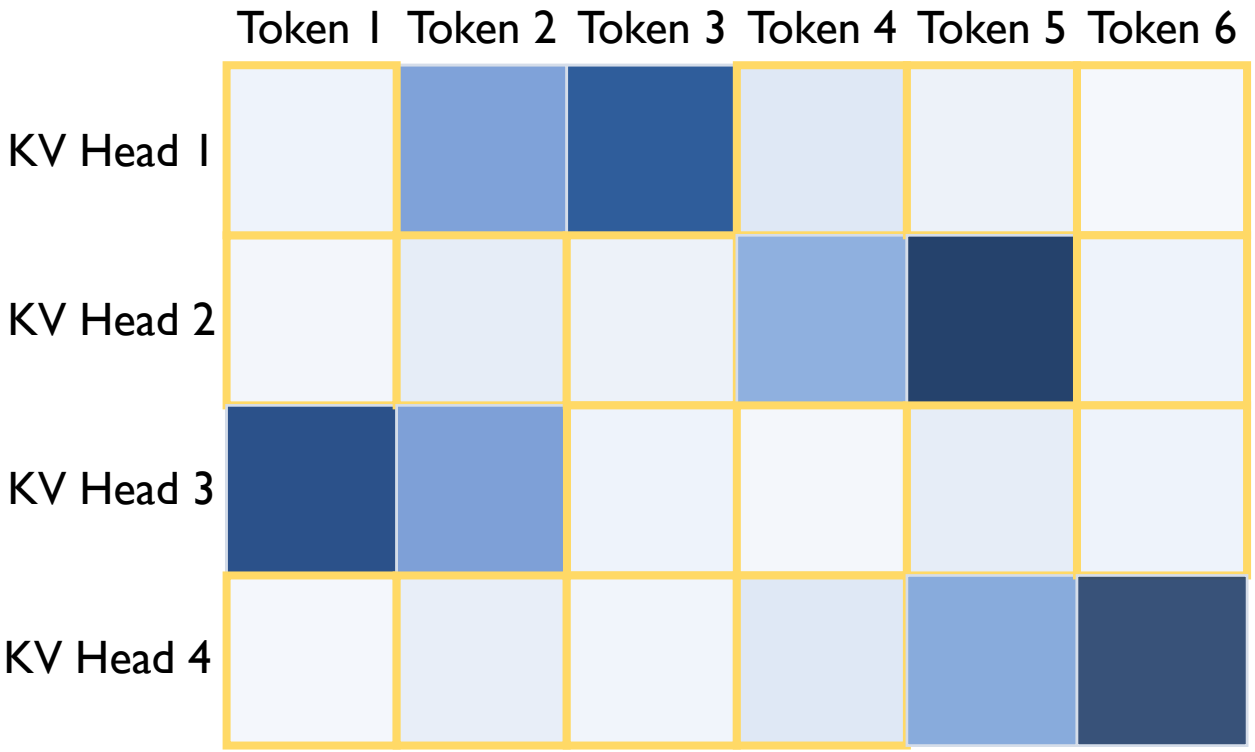


Attention heatmap for next decode step

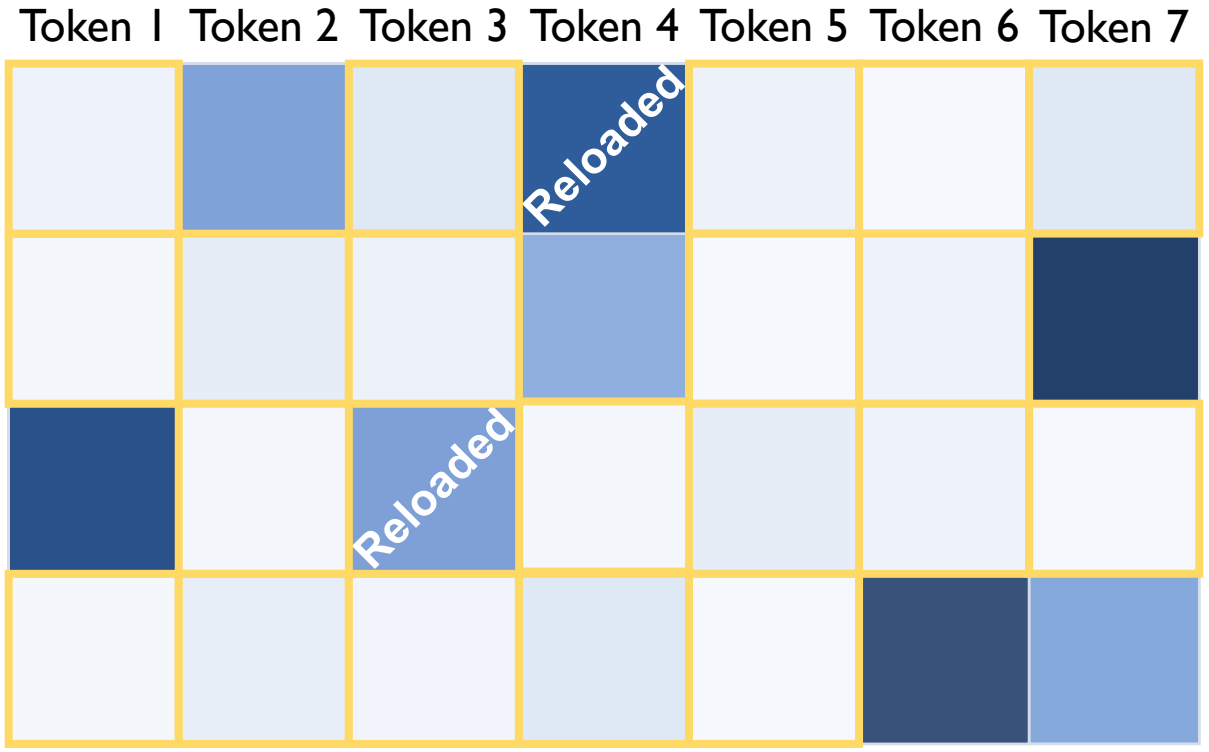


Leveraging Sparsity: Offloading + Query Aware Selection

Attention heatmap



Attention heatmap for next decode step

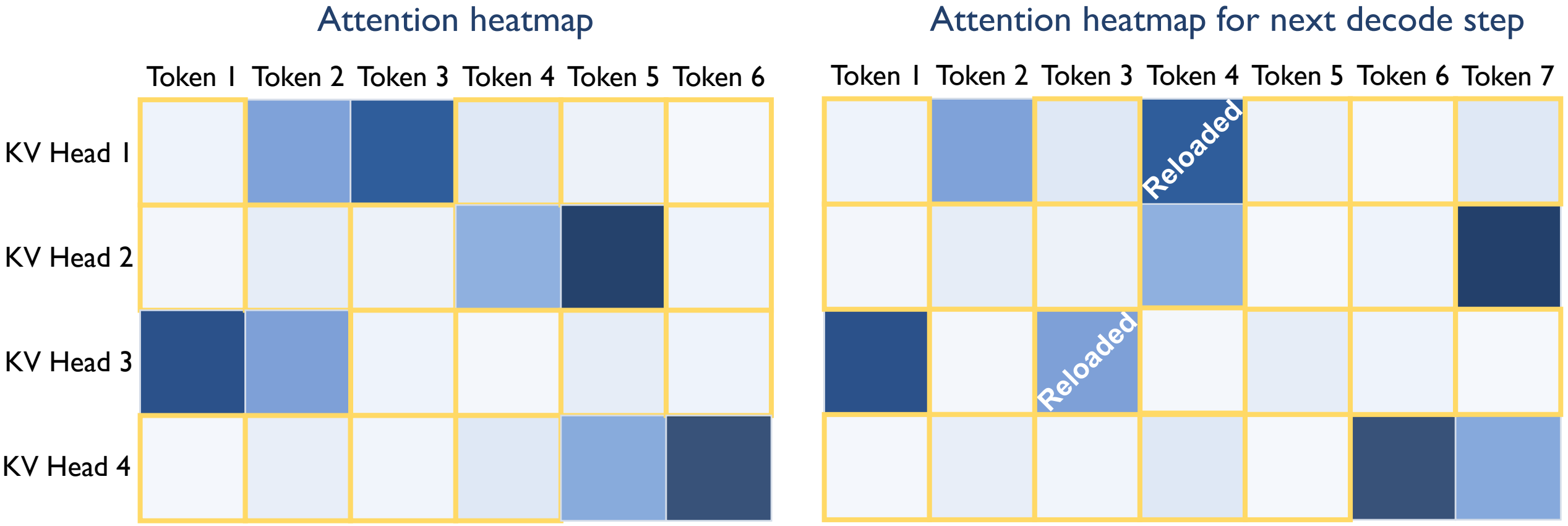


Model Accuracy Preserved

HBM usage ↓

Attention Compute ↓

Leveraging Sparsity: Offloading + Query Aware Selection



Host → GPU
I/O Bottleneck

Leveraging Sparsity: Summary

Method	HBM ↓	I/O ↓	Compute ↓	Accuracy Preserved

Leveraging Sparsity: Summary

Method	HBM ↓	I/O ↓	Compute ↓	Accuracy Preserved
Discarding KV	✅	✅	✅	❌

Leveraging Sparsity: Summary

Method	HBM ↓	I/O ↓	Compute ↓	Accuracy Preserved
Discarding KV	✅	✅	✅	❌
Query Aware KV Selection	❌	✅	✅	✅

Leveraging Sparsity: Summary

Method	HBM ↓	I/O ↓	Compute ↓	Accuracy Preserved
Discarding KV	✅	✅	✅	❌
Query Aware KV Selection	❌	✅	✅	✅
Offloading + Query Aware KV Selection	✅	❌	✅	✅

Leveraging Sparsity: Summary

Method	HBM ↓	I/O ↓	Compute ↓	Accuracy Preserved
Discarding KV	✅	✅	✅	❌
Query Aware KV Selection	❌	✅	✅	✅
Offloading + Query Aware KV Selection	✅	❌	✅	✅
FlexiCache	✅	✅	✅	✅

FlexiCache achieves all four

Key Insight: Temporal stability

- **Temporal Stability:** How consistently an attention head keeps selecting the same Top-K KV across decode steps

Key Insight: Temporal stability

- **Temporal Stability:** How consistently an attention head keeps selecting the same Top-K KV across decode steps

- **Quantifying Temporal Stability:**
  - Random Corrected Overlap (RCO)

Key Insight: Temporal stability

- **Temporal Stability:** How consistently an attention head keeps selecting the same Top-K KV across decode steps

- **Quantifying Temporal Stability:**
  - Random Corrected Overlap (RCO)

$$\mathrm{RCO}(s,t) = \frac{\frac{\left|S^{(s)} \cap S^{(t)}\right|}{K} - \frac{K}{N_t}}{1 - \frac{K}{N_t}}$$

where,

- $S^{(s)}$: Top−K KV set at decode step s
- $S^{(t)}$: Top−K KV set at decode step t
- $\frac{\left|S^{(s)} \cap S^{(t)}\right|}{K}$: **Fraction of Top-K KV set that remains the same across steps**

Key Insight: Temporal stability

- **Temporal Stability:** How consistently an attention head keeps selecting the same Top-K KV across decode steps

- **Quantifying Temporal Stability:**
  - Random Corrected Overlap (RCO)

$$\mathrm{RCO}(s,t) = \frac{\frac{\left|S^{(s)} \cap S^{(t)}\right|}{K} - \frac{K}{N_t}}{1 - \frac{K}{N_t}}$$

where,

- $\frac{\left|S^{(s)} \cap S^{(t)}\right|}{K}$: Fraction of Top−K KV that remains the same across steps
- **Subtract $\frac{K}{N_t}$: remove random chance overlap**

Key Insight: Temporal stability

- **Temporal Stability:** How consistently an attention head keeps selecting the same Top-K KV across decode steps

- **Quantifying Temporal Stability:**
  - Random Corrected Overlap (RCO)

$$\mathrm{RCO}(s,t) = \frac{\frac{\left|S^{(s)} \cap S^{(t)}\right|}{K} - \frac{K}{N_t}}{1 - \frac{K}{N_t}}$$

where,

- $\frac{\left|S^{(s)} \cap S^{(t)}\right|}{K}$: Fraction of Top−K KV that remains the same across steps
- Subtract $\frac{K}{N_t}$: remove random chance overlap
- **Divide by $1 - \frac{K}{N_t}$: scale RCO between 0 and 1**

Key Insight: Temporal stability

- **Temporal Stability:** How consistently an attention head keeps selecting the same Top-K KV across decode steps

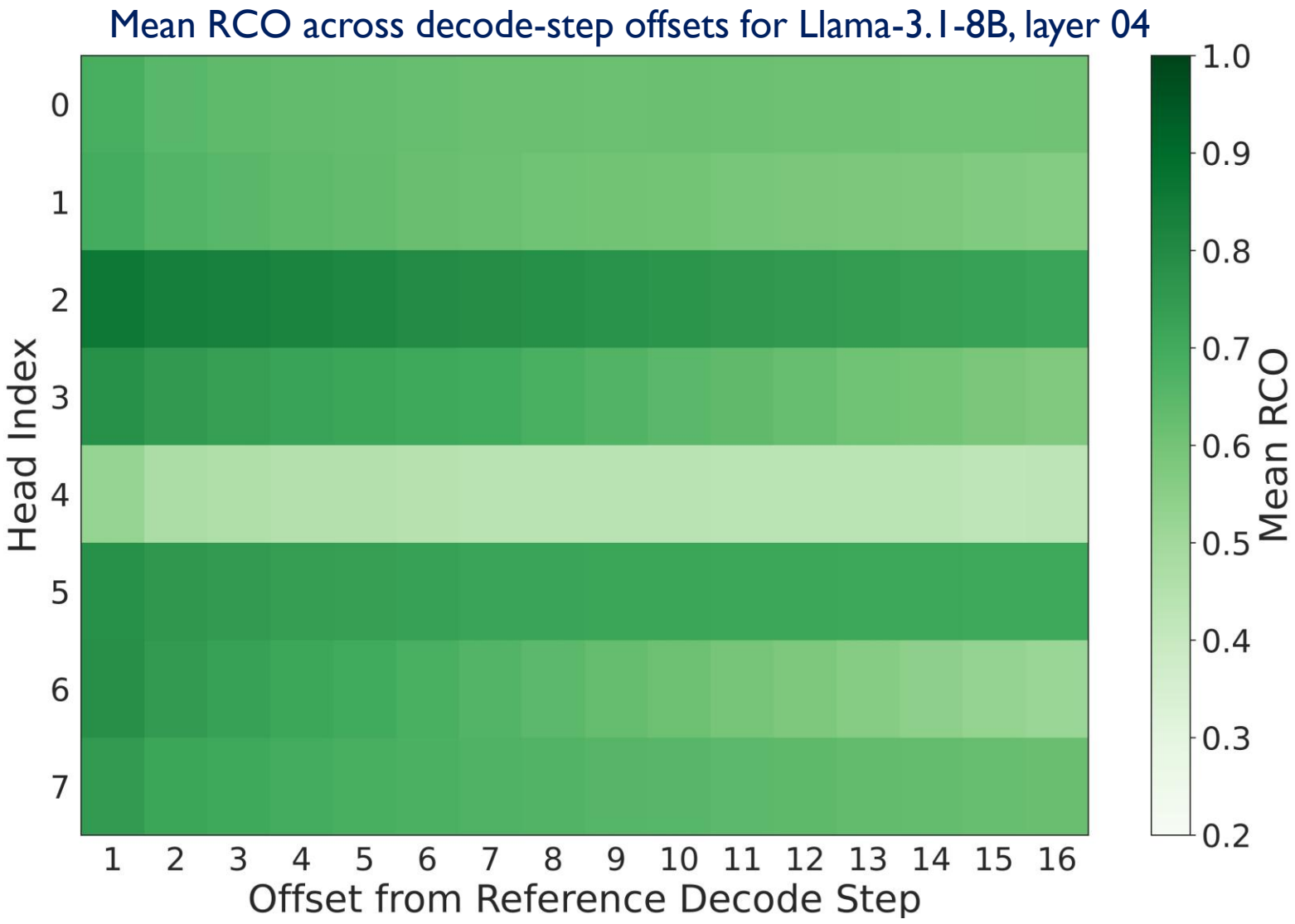
- **Quantifying Temporal Stability:**
  - Random Corrected Overlap (RCO)

$$\mathrm{RCO}(s,t) = \frac{\frac{\left|S^{(s)} \cap S^{(t)}\right|}{K} - \frac{K}{N_t}}{1 - \frac{K}{N_t}}$$

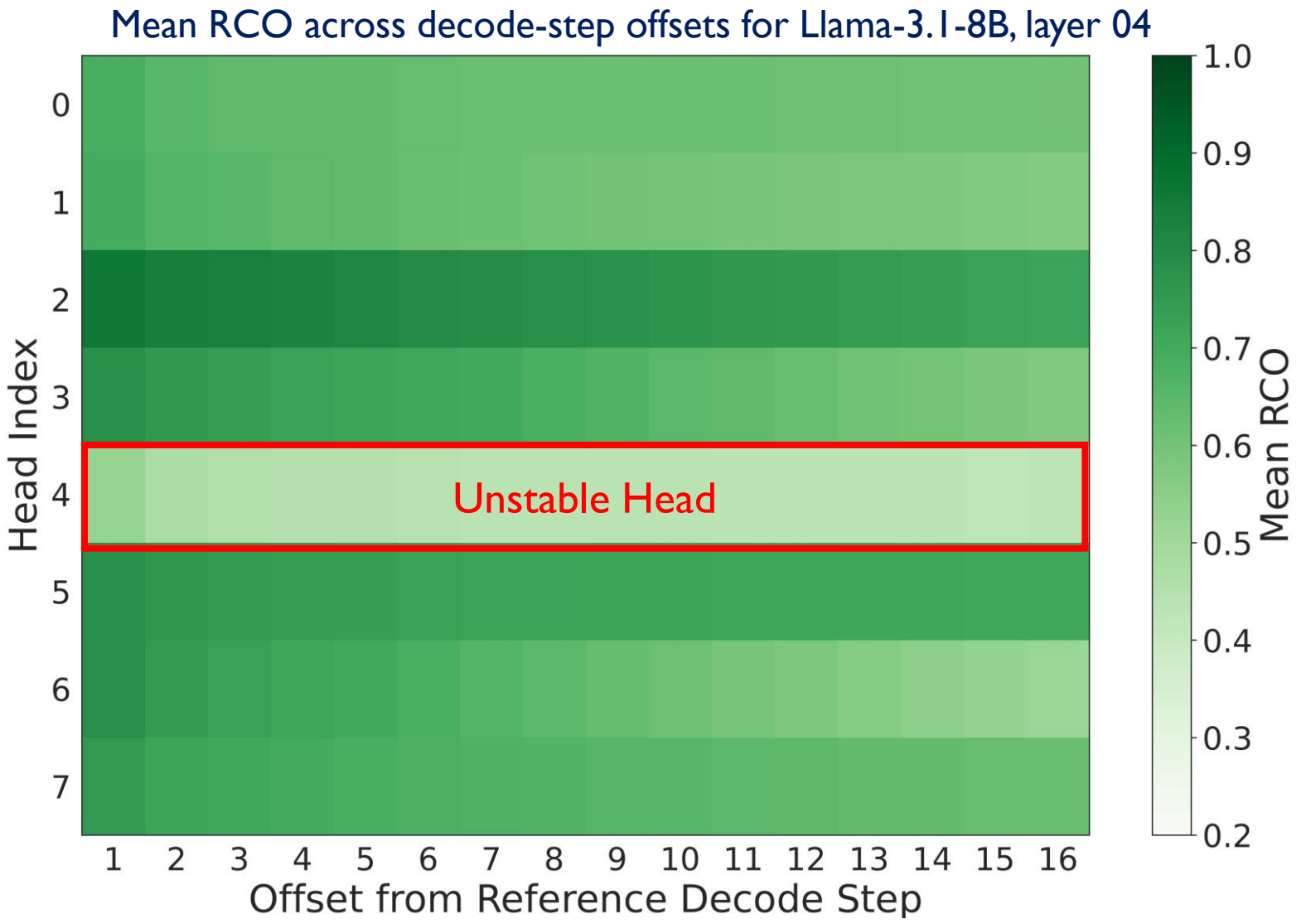
where,

- RCO of 0 → random-selection level overlap
- RCO of 1 → perfect overlap

Key Insight: Non-uniform temporal stability

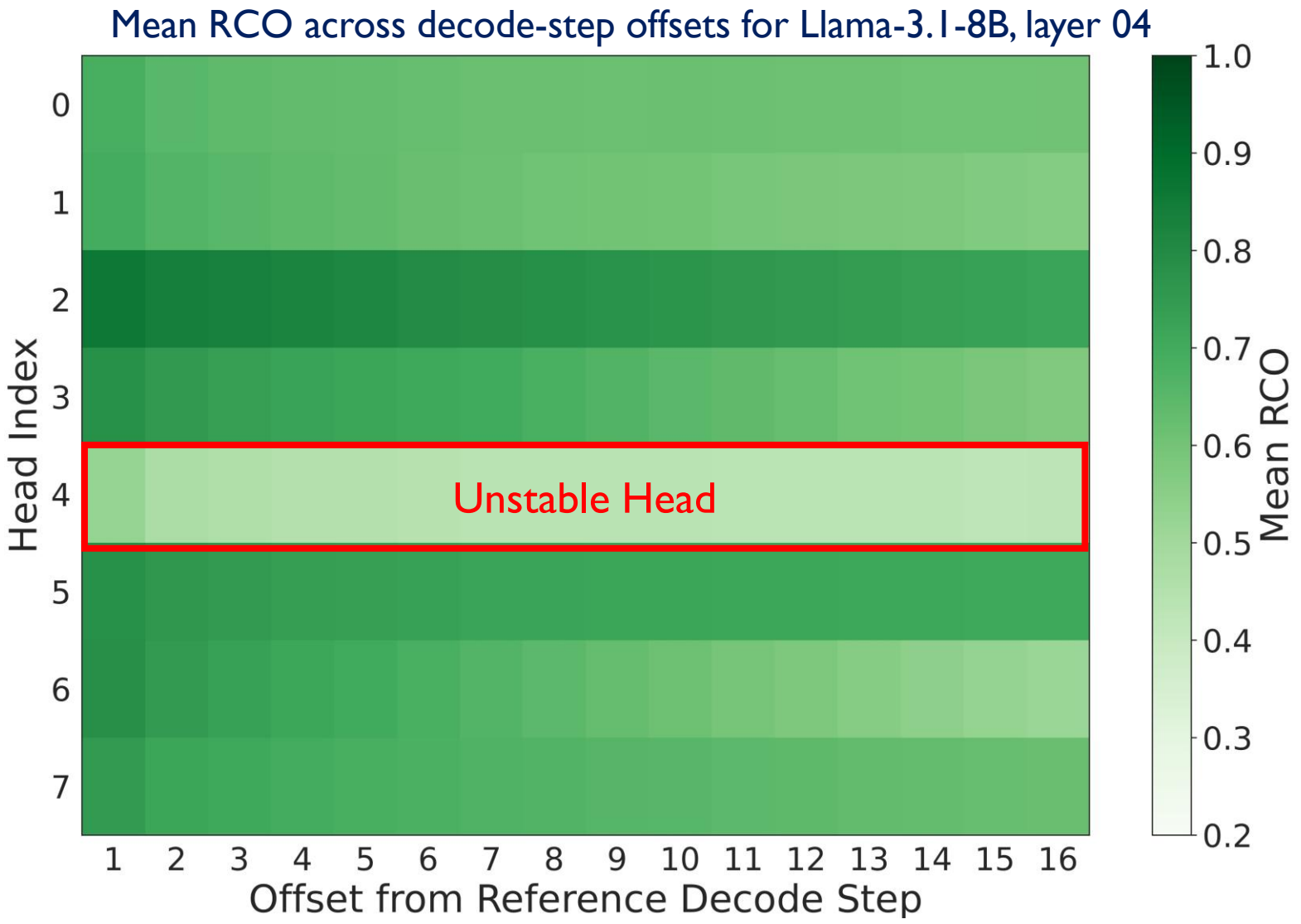


Key Insight: Unstable Heads



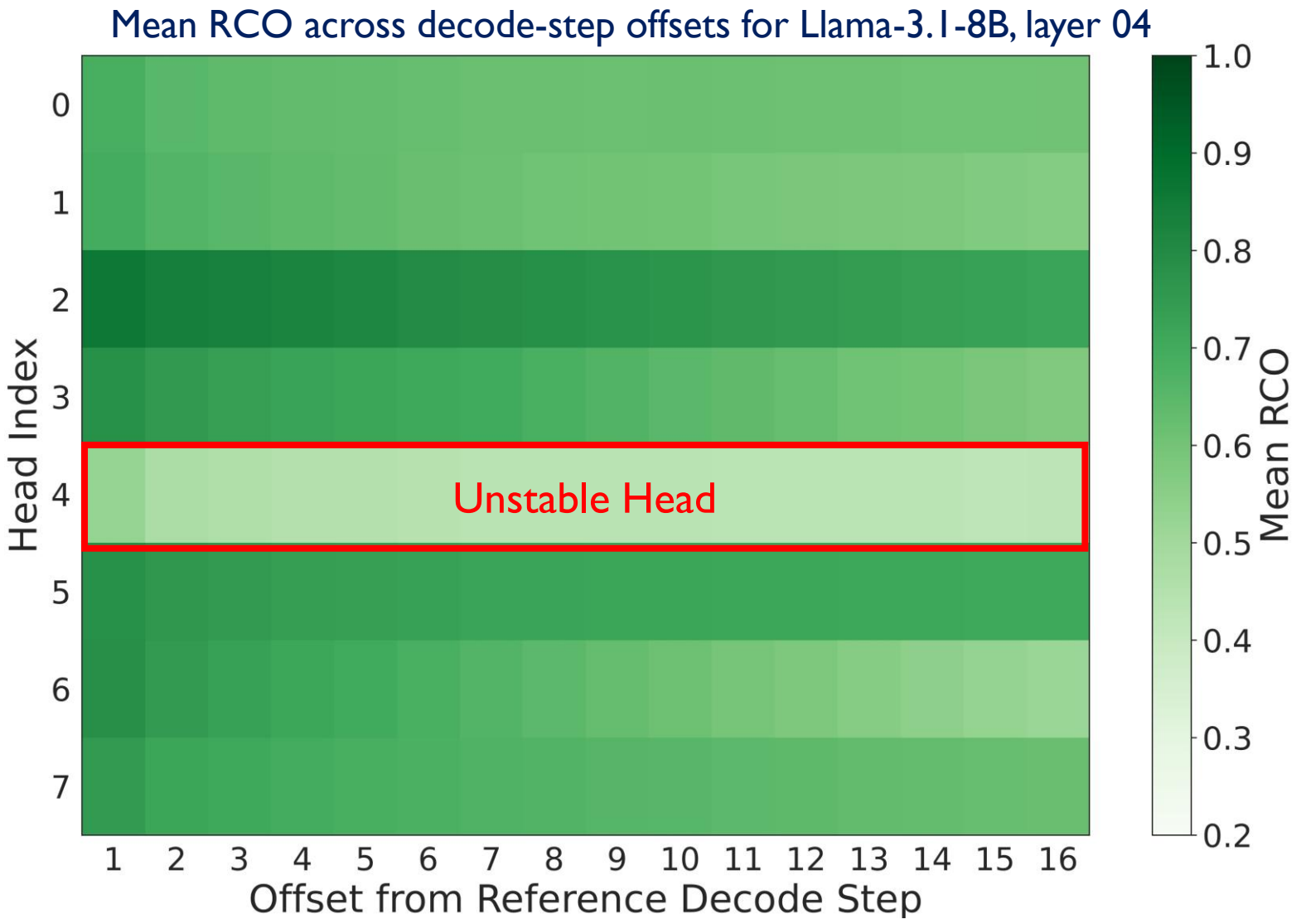
Key Insight: Unstable Heads

- For each KV head: Compute mean RCO across decode steps



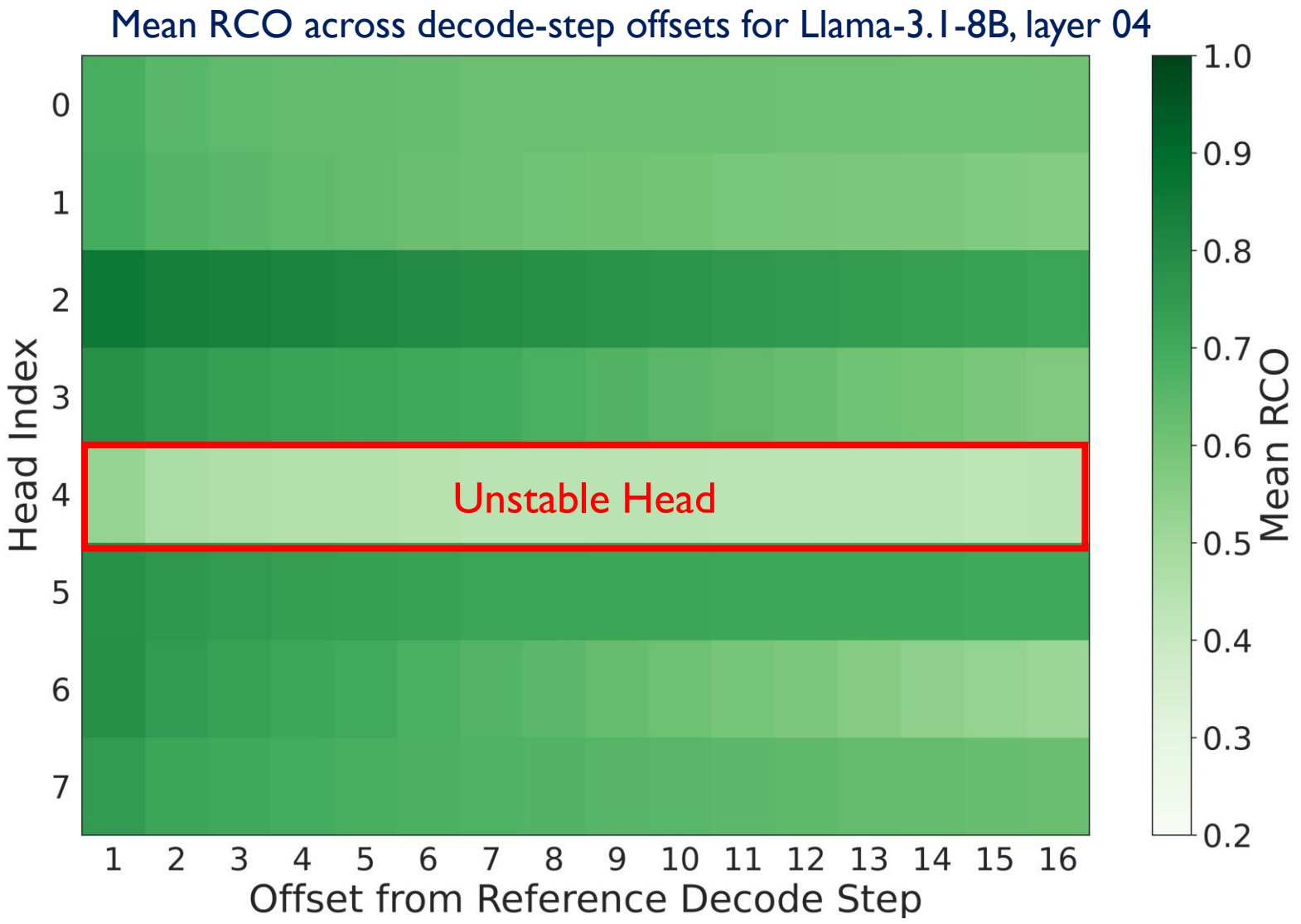
Key Insight: Unstable Heads

- For each KV head: Compute mean RCO across decode steps
- Rank KV heads by mean RCO



Key Insight: Unstable Heads

- For each KV head: Compute mean RCO across decode steps
- Rank KV heads by mean RCO
- Classify least stable 25% of KV heads as unstable



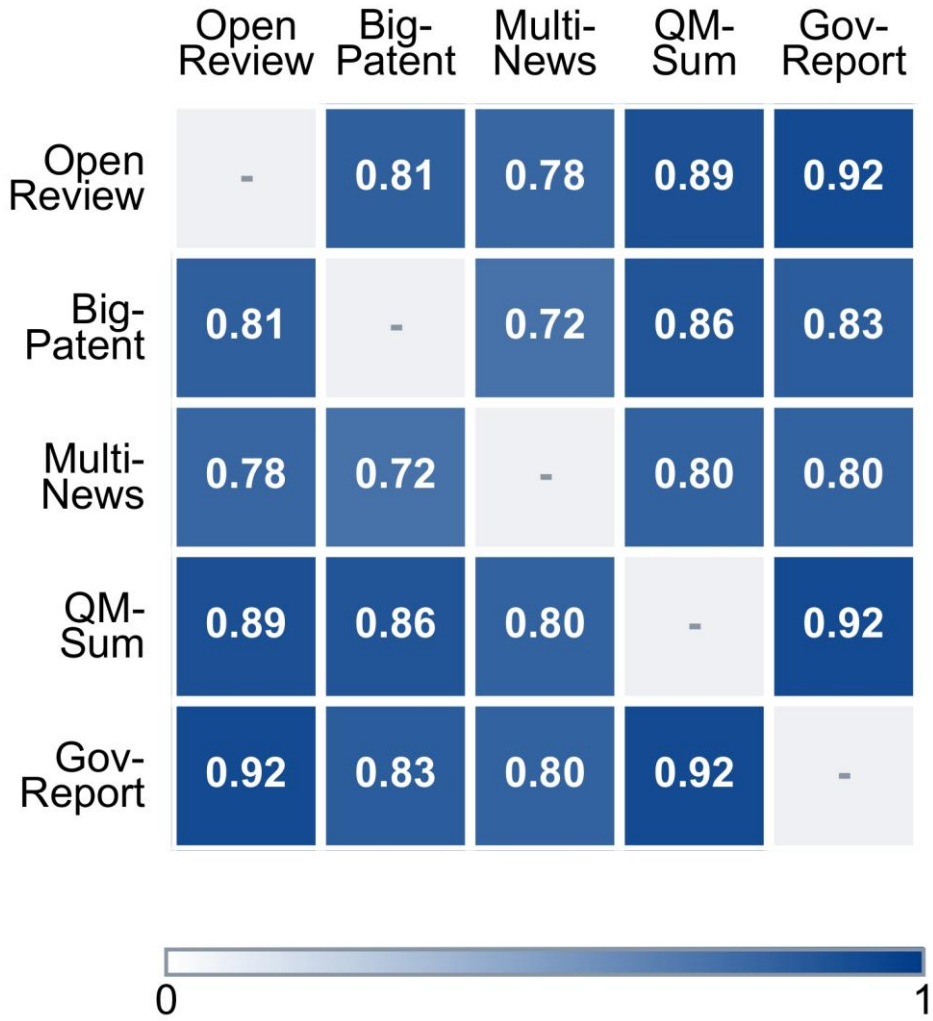
Key Insight: Model Intrinsic Unstable Heads

- Identify unstable heads independently for several datasets
- High overlap → instability is model intrinsic

Key Insight: Model Intrinsic Unstable Heads

- Identify unstable heads independently for several datasets
- High overlap → instability is model intrinsic

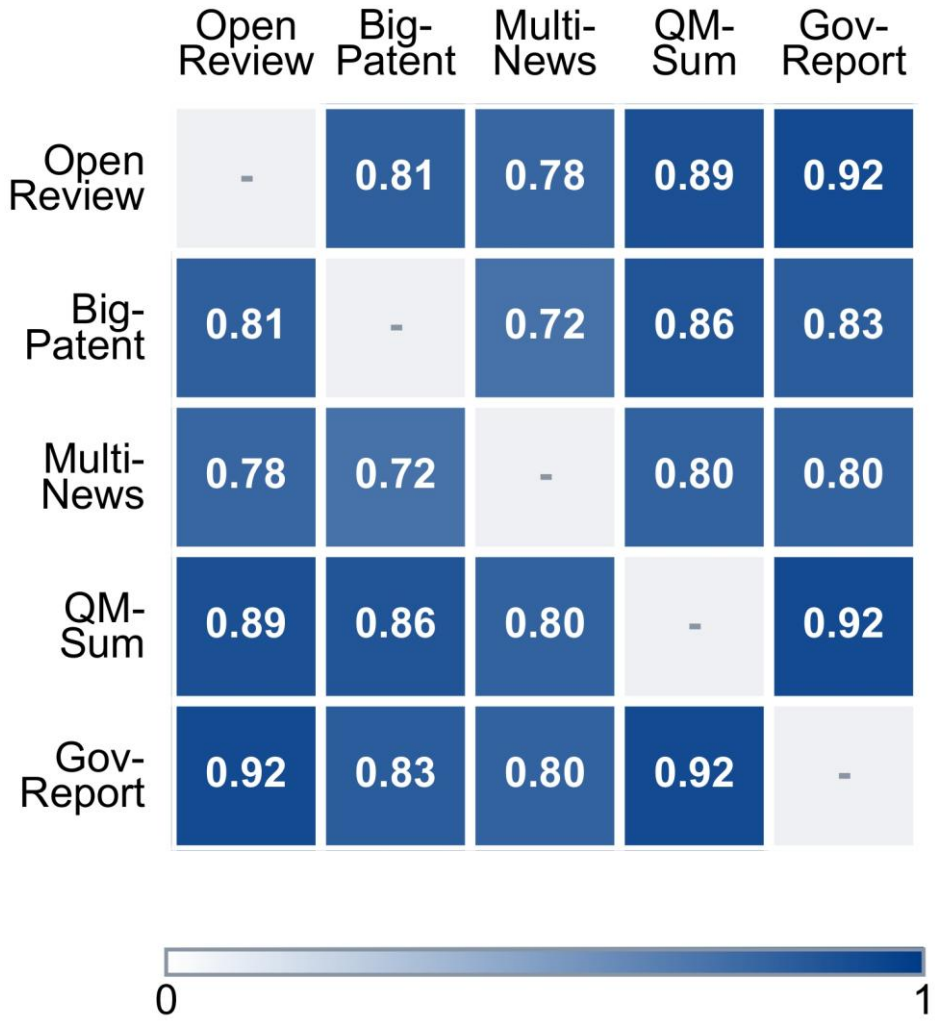
Cross-dataset overlap of unstable heads for Llama-3.1-8B-Instruct



Key Insight: Model Intrinsic Unstable Heads

- Identify unstable heads independently for several datasets
- High overlap → instability is model intrinsic
- Paper reports similar trends for more models & datasets

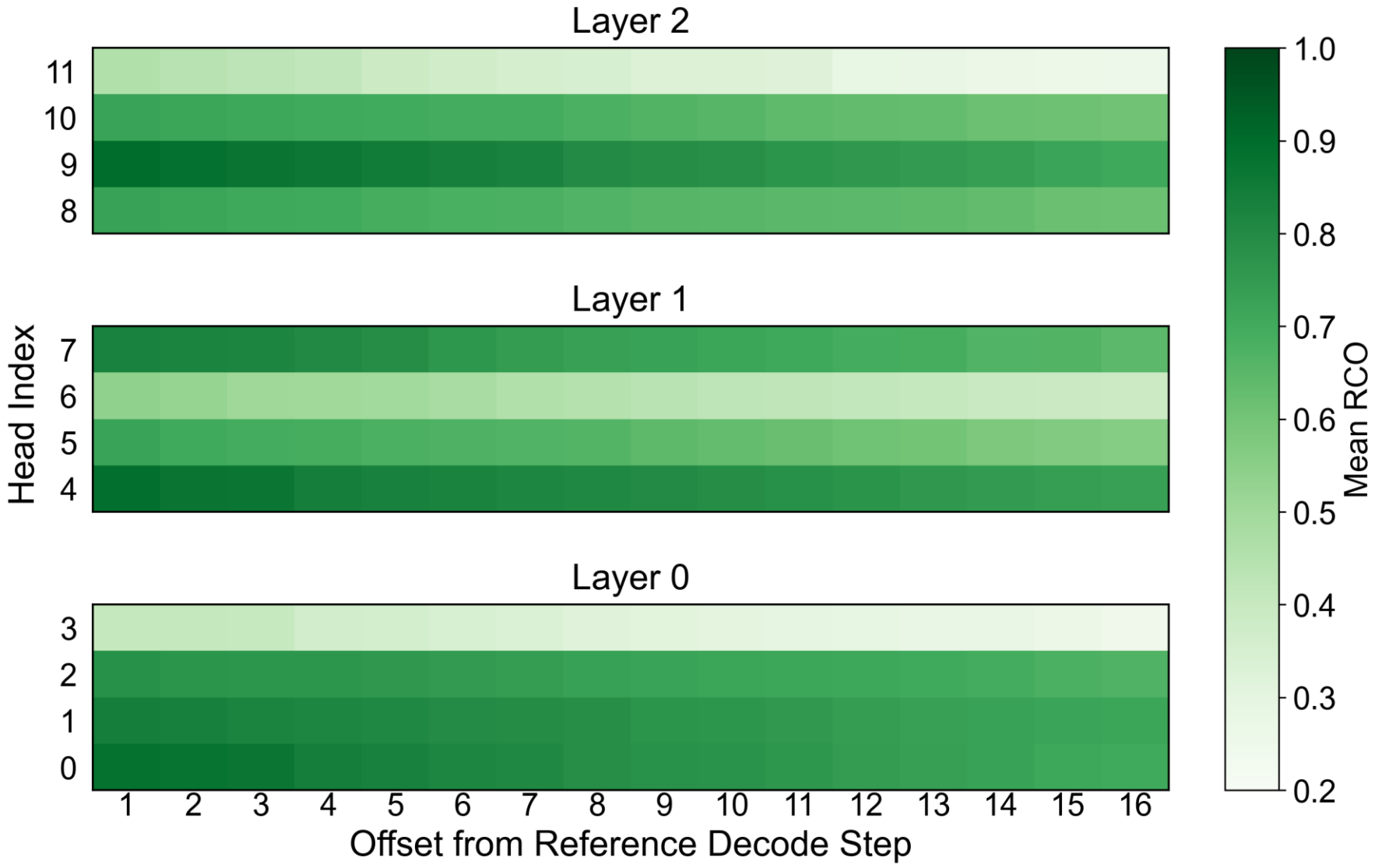
Cross-dataset overlap of unstable heads for Llama-3.1-8B-Instruct



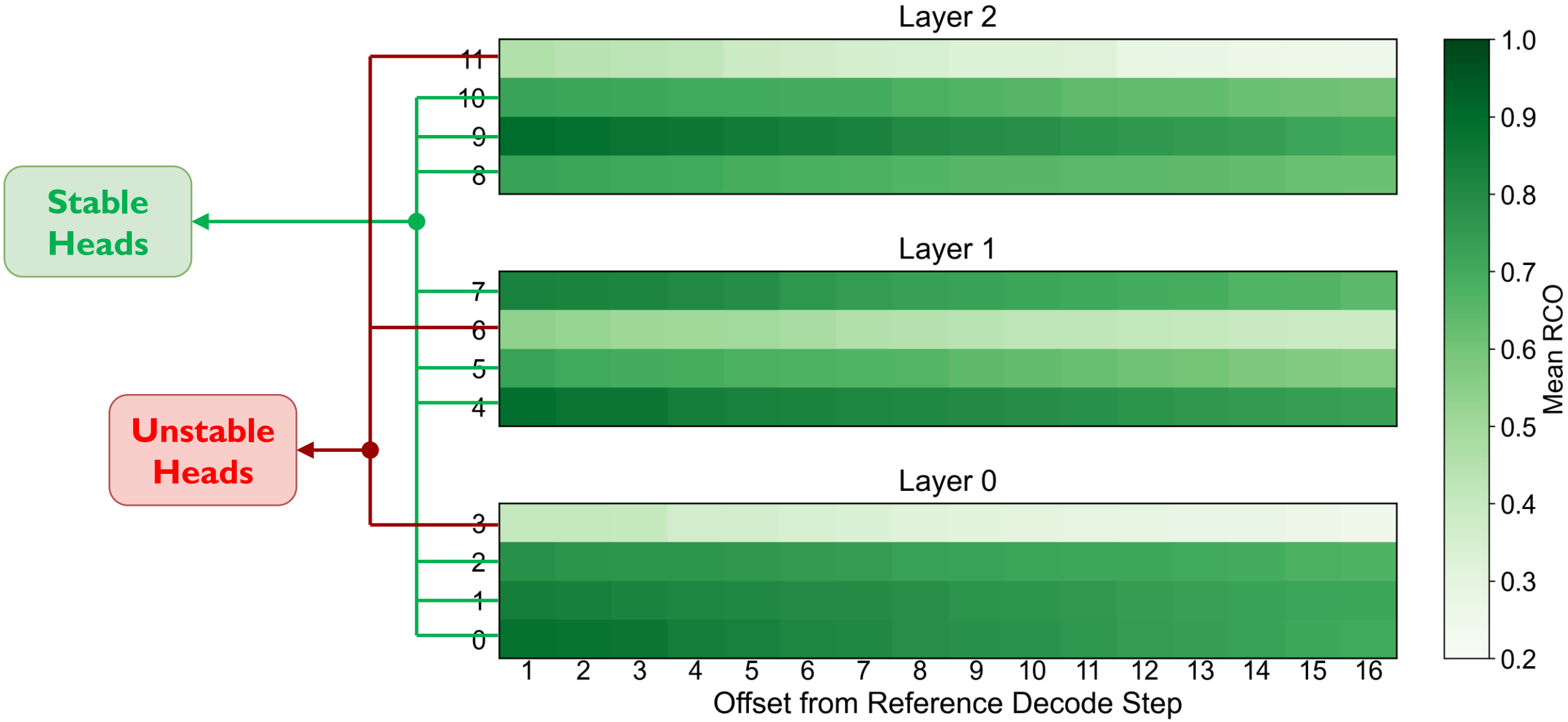
FlexiCache

Temporal Stability Aware Hierarchical KV Cache Management

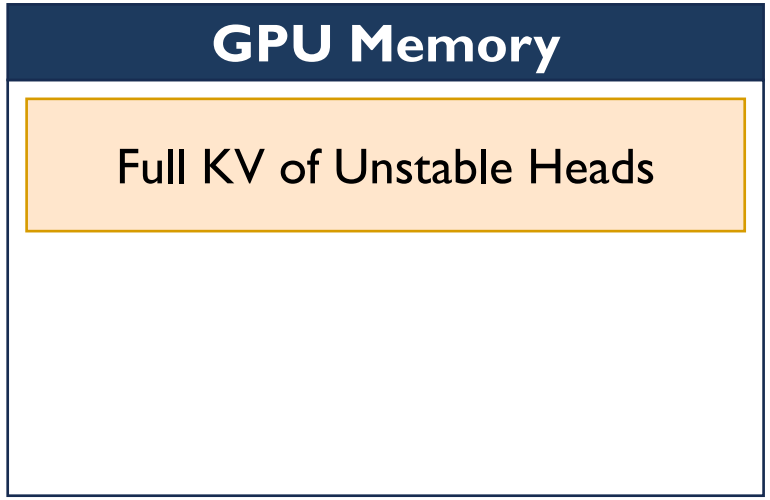
FlexiCache: Offline KV Head Classification



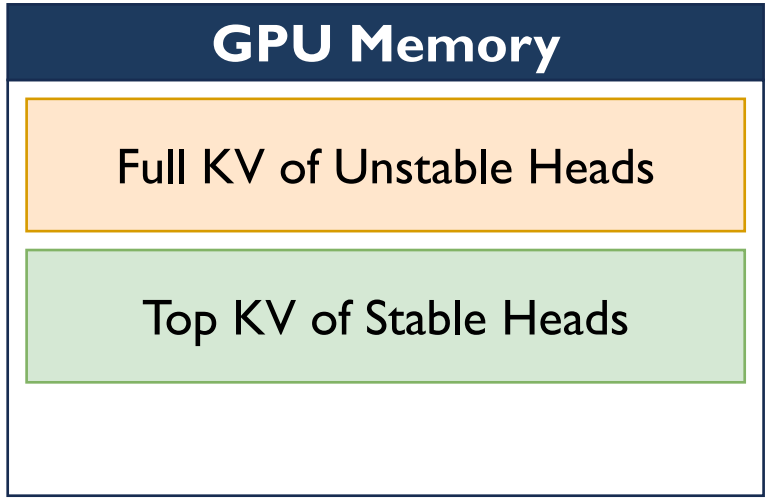
FlexiCache: Offline KV Head Classification



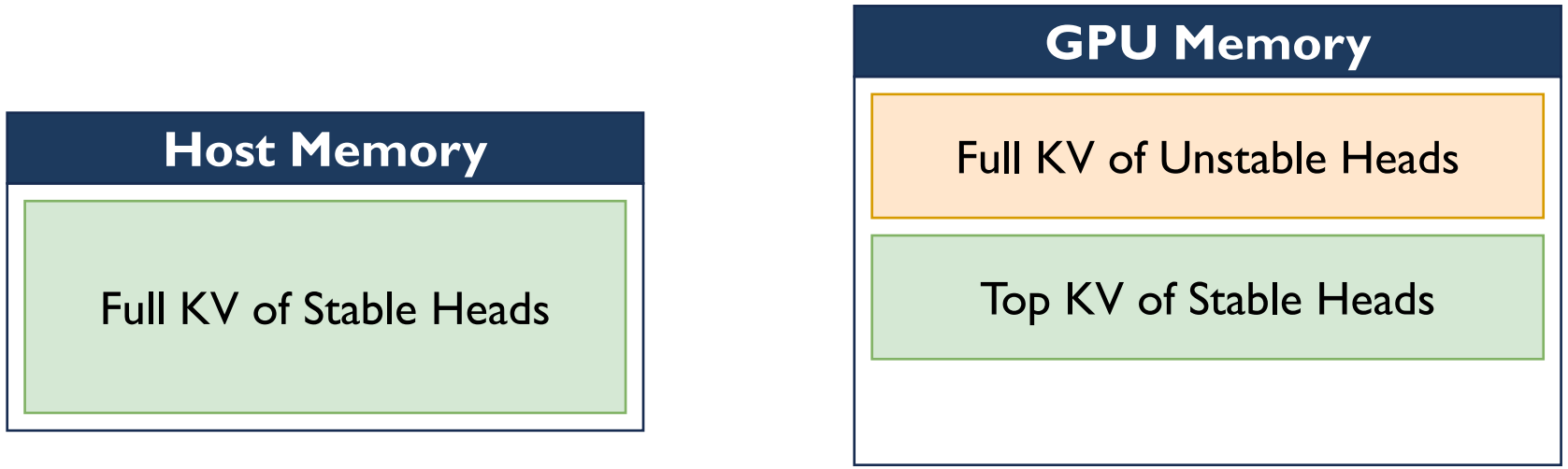
FlexiCache: Stability Aware Hierarchical KV Placement



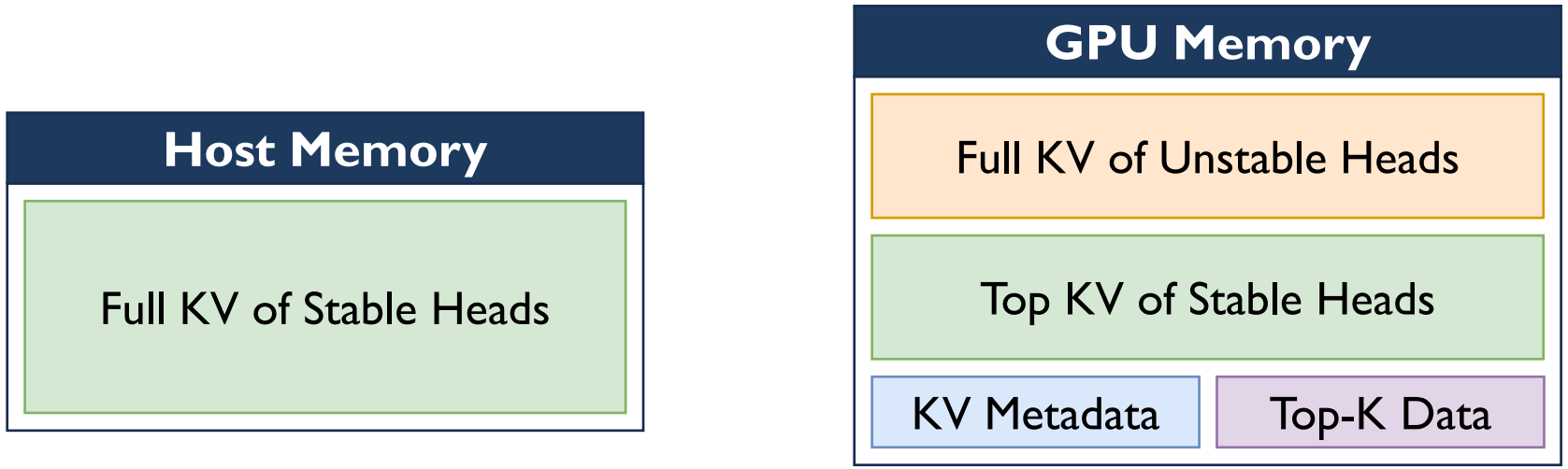
FlexiCache: Stability Aware Hierarchical KV Placement



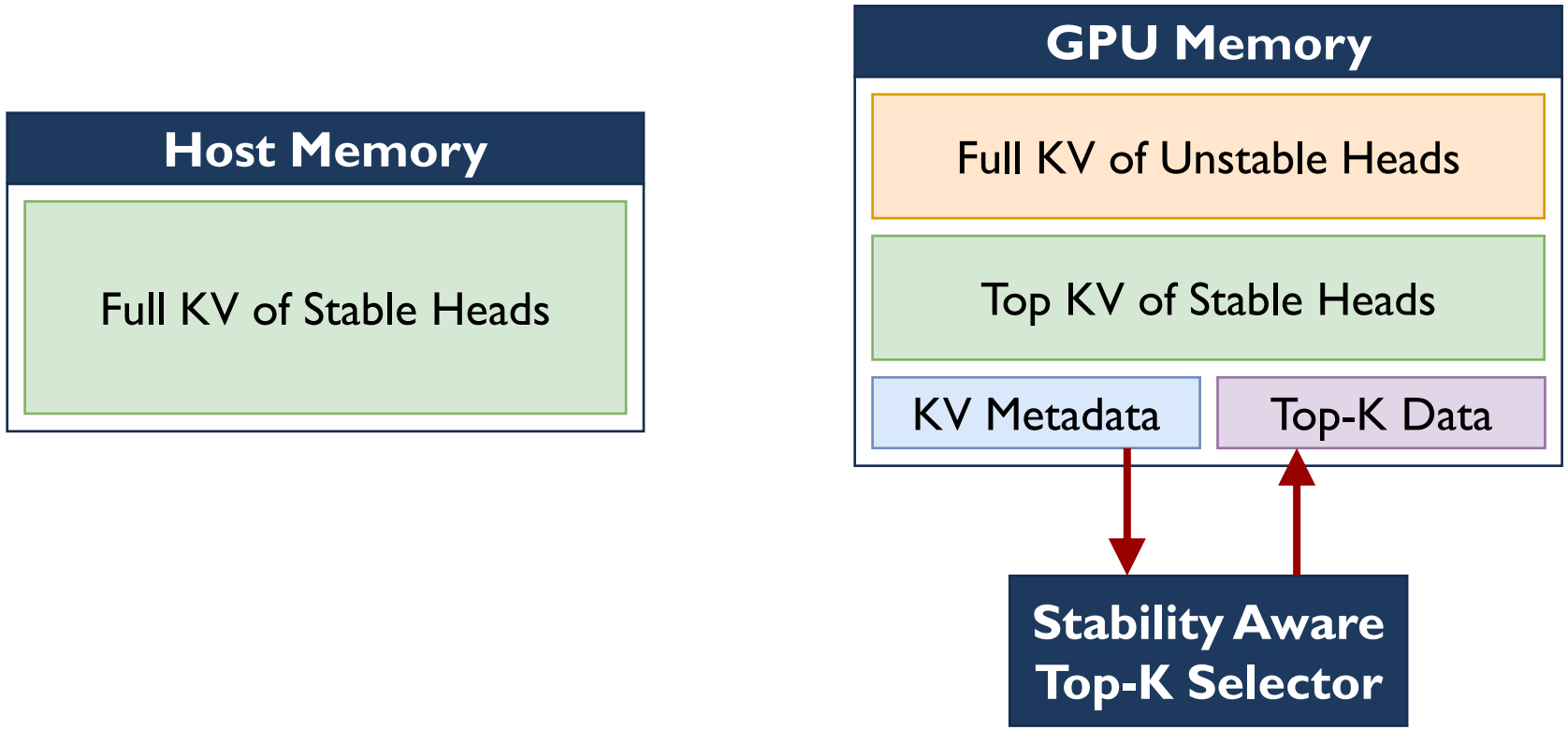
FlexiCache: Stability Aware Hierarchical KV Placement



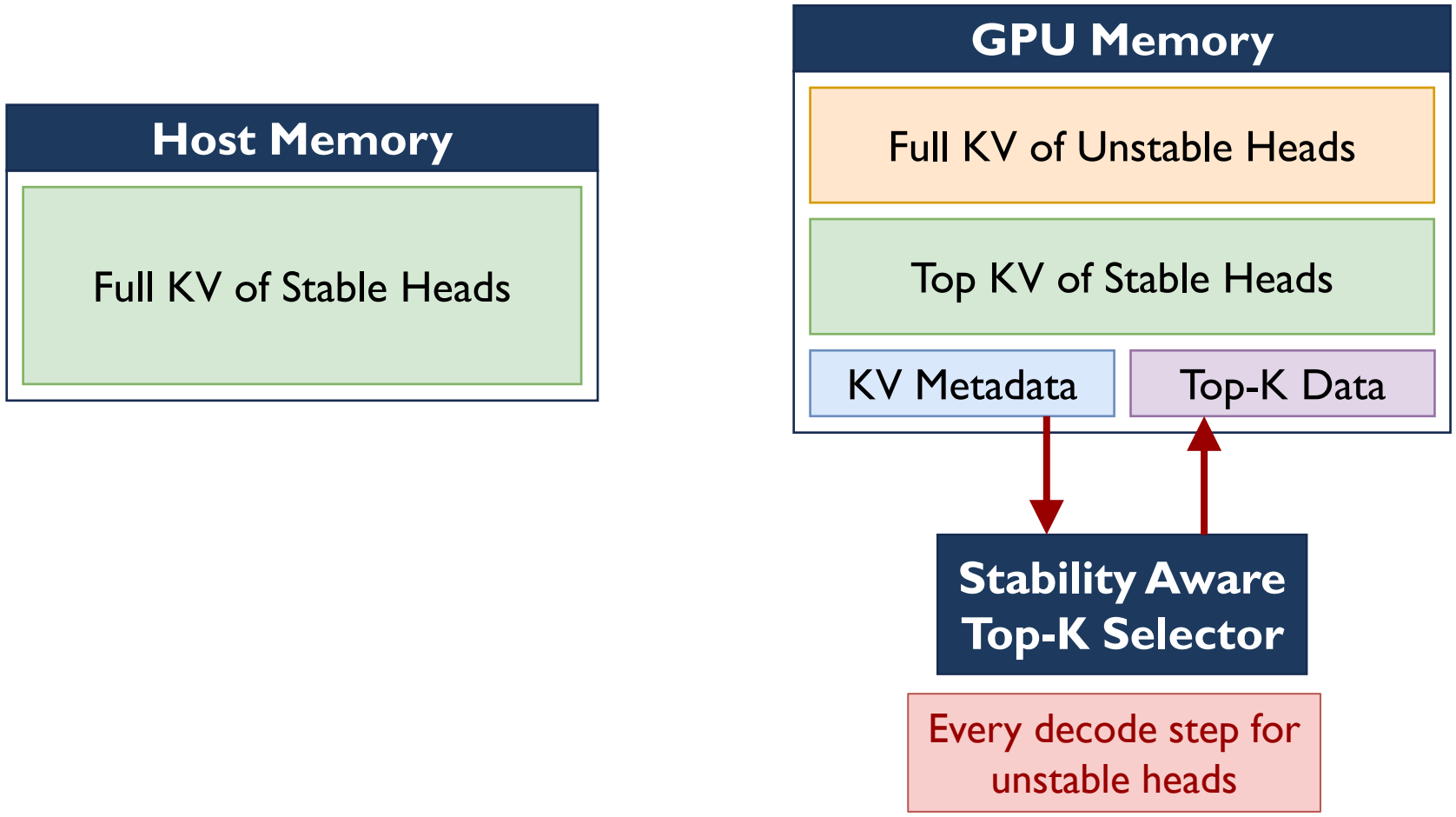
FlexiCache: Stability Aware Hierarchical KV Placement



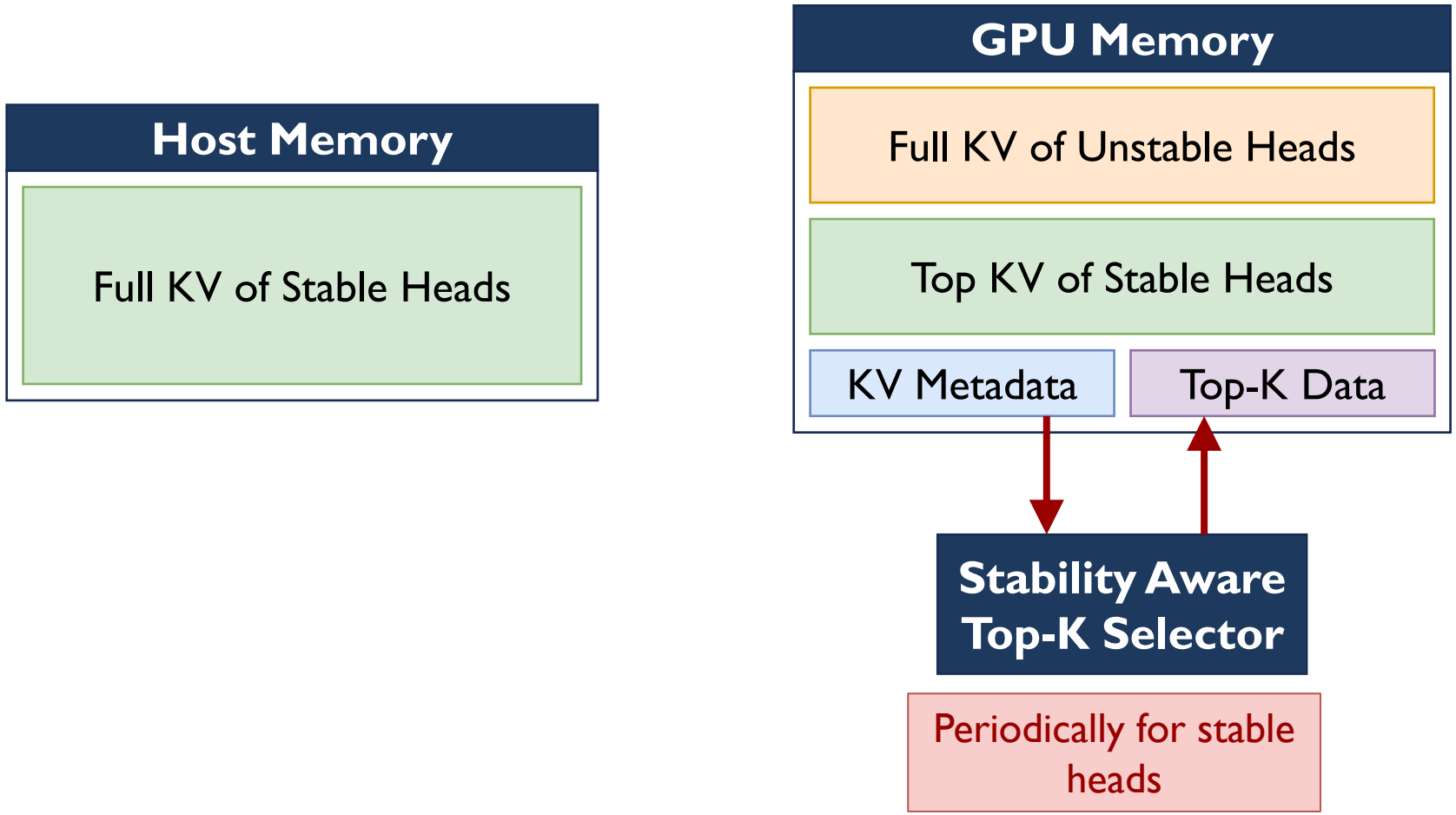
FlexiCache: Stability Aware Top-K KV Selector



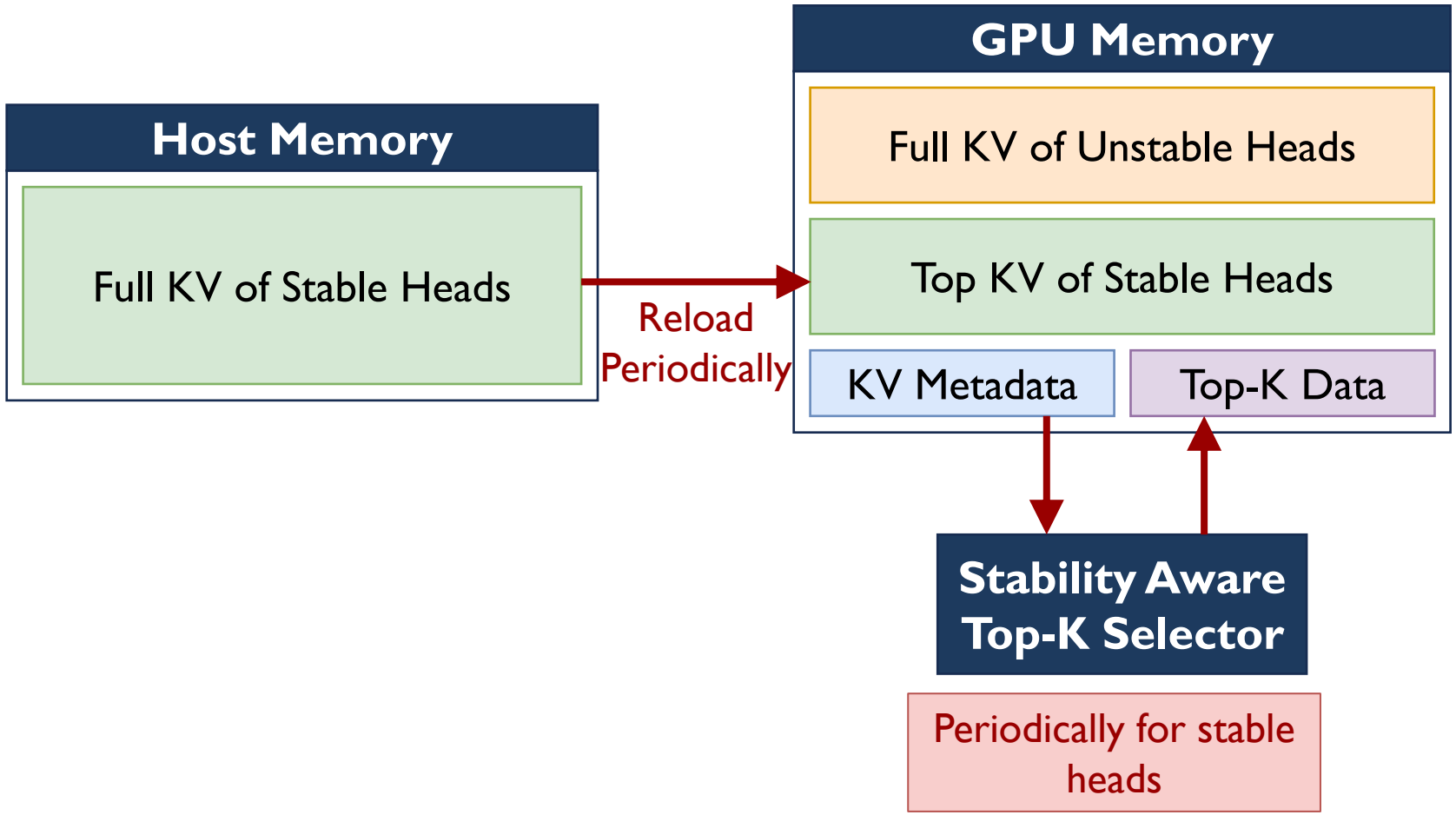
FlexiCache: Stability Aware Top-K KV Selector



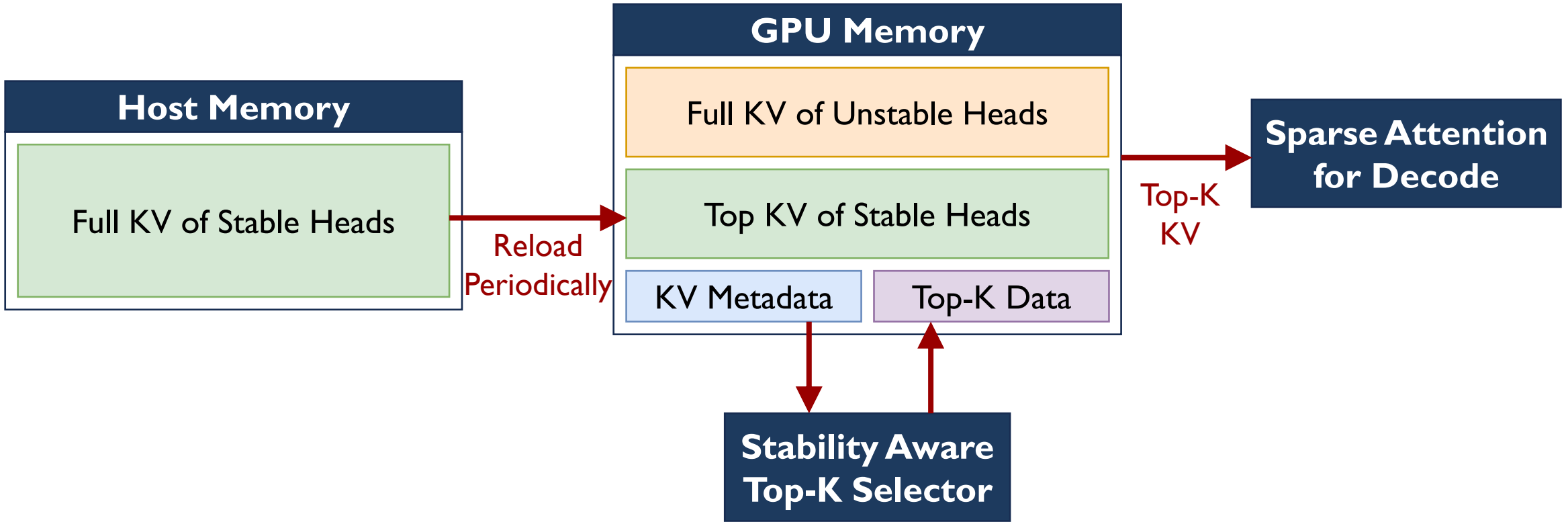
FlexiCache: Stability Aware Top-K KV Selector



FlexiCache: Stability Aware Top-K KV Selector



FlexiCache: Sparse Decode on Top-K KV



FlexiCache: Selecting Top-K KV

Query-aware page scoring

- Select Top-K KV cache at **page-level granularity**
  - Compatible with paged attention

FlexiCache: Selecting Top-K KV

Query-aware page scoring

- Select Top-K KV cache at **page-level granularity**
  - Compatible with paged attention

- Score pages using MinMax key summaries
  - Adapted from Quest [ICML 2024]

FlexiCache: Selecting Top-K KV

Query-aware page scoring

- Select Top-K KV cache at **page-level granularity**
  - Compatible with paged attention

- Score pages using MinMax key summaries
  - Adapted from Quest [ICML 2024]

- **Stability-aware reranking**
  - Unstable heads: rerank every decode step
  - Stable heads: rerank periodically and reuse Top-K pages between reranks

FlexiCache: Other Optimizations

Efficient Host-GPU KV Movement

Efficient Block Table Management

More details in the paper …

Implementation

- Built on top of vLLM
- Modified Triton Flash-Decoding kernel
  - Decode attention only over selected Top-K KV pages
- Optimized KV movement with custom CUDA kernels
  - Overlap KV offload/reload with compute

Evaluation

- **Models**
  - Llama-3.1-8B-Instruct, Mistral-7B-Instruct-v0.2, Mistral-Small-24B, Qwen2.5-32B

- **Testbed:**
  - NVIDIA H100 GPU (94 GB) and 180 GB allocated on host memory for offloaded KV cache

Evaluation

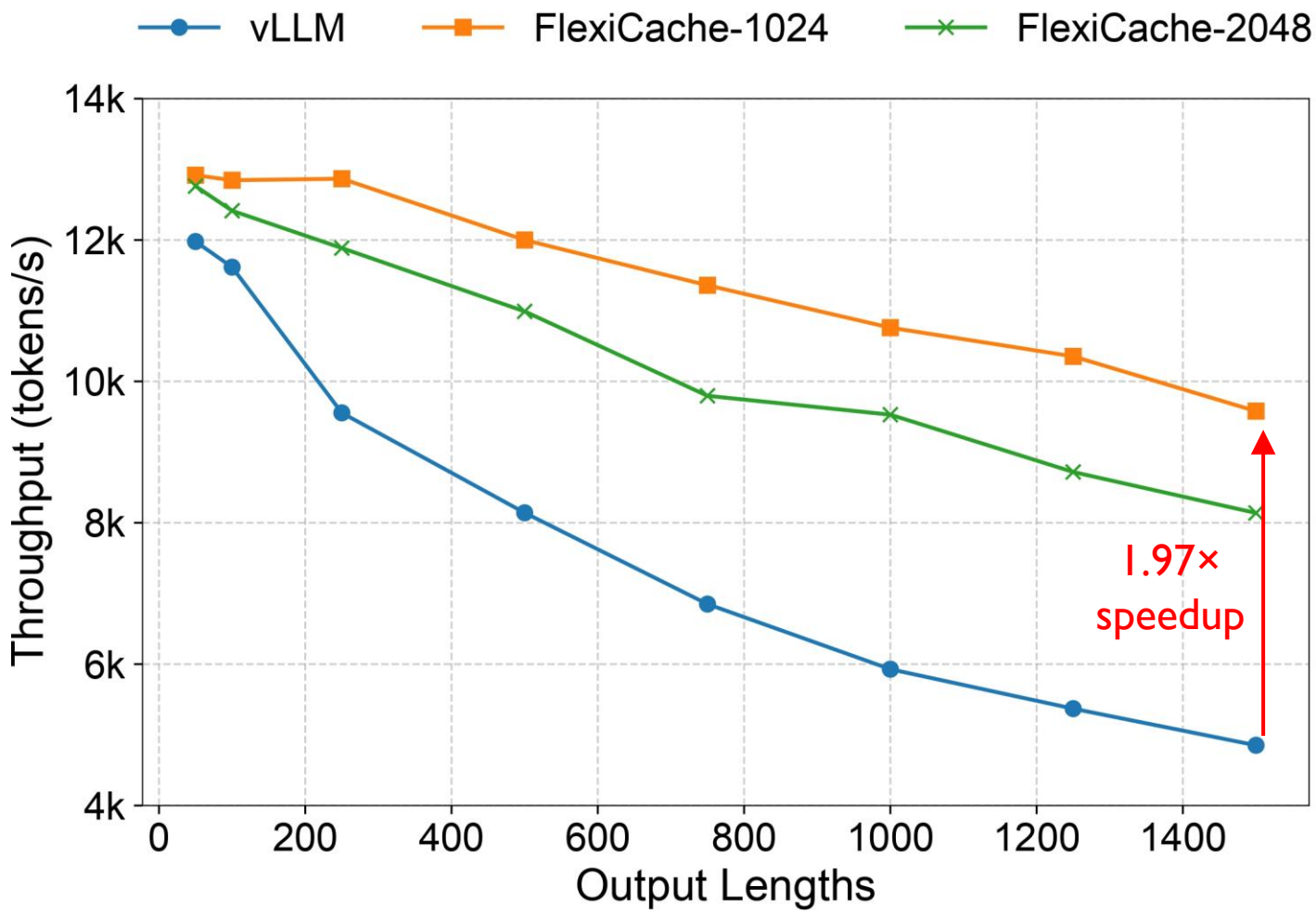
- **Models**
  - **Llama-3.1-8B-Instruct**, Mistral-7B-Instruct-v0.2, Mistral-Small-24B, Qwen2.5-32B

- **Testbed:**
  - NVIDIA H100 GPU (94 GB) and 180 GB allocated on host memory for offloaded KV cache

Offline Serving: Up to 1.97× Higher Throughput

- FlexiCache consistently outperforms vLLM across output lengths

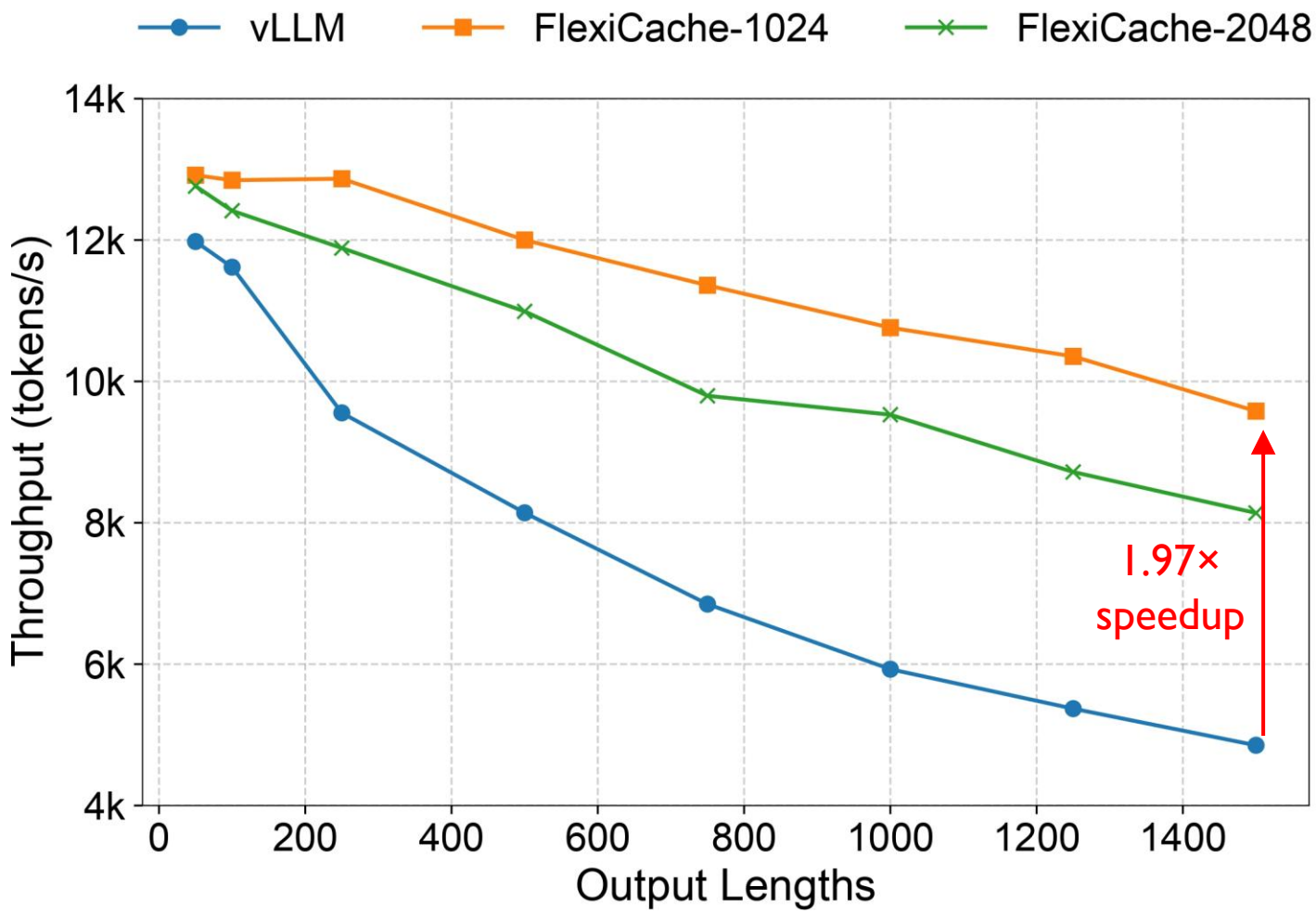
End-to-end token throughput for Llama-3.1-8B-Instruct on L-Eval



Offline Serving: Up to 1.97× Higher Throughput

- FlexiCache consistently outperforms vLLM across output lengths
- Longer output → decoding dominates runtime → larger throughput gains

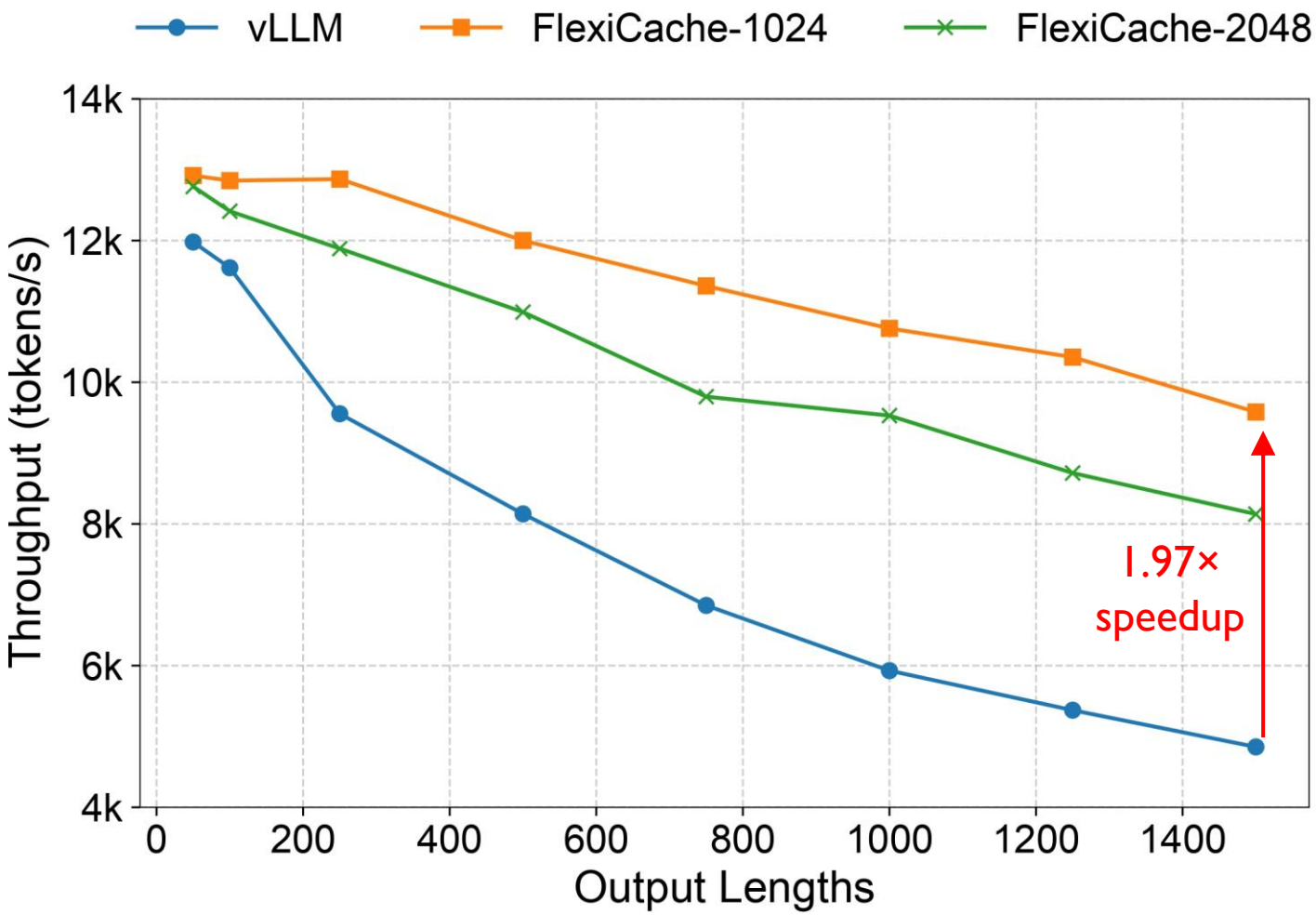
End-to-end token throughput for Llama-3.1-8B-Instruct on L-Eval



Offline Serving: Up to 1.97× Higher Throughput

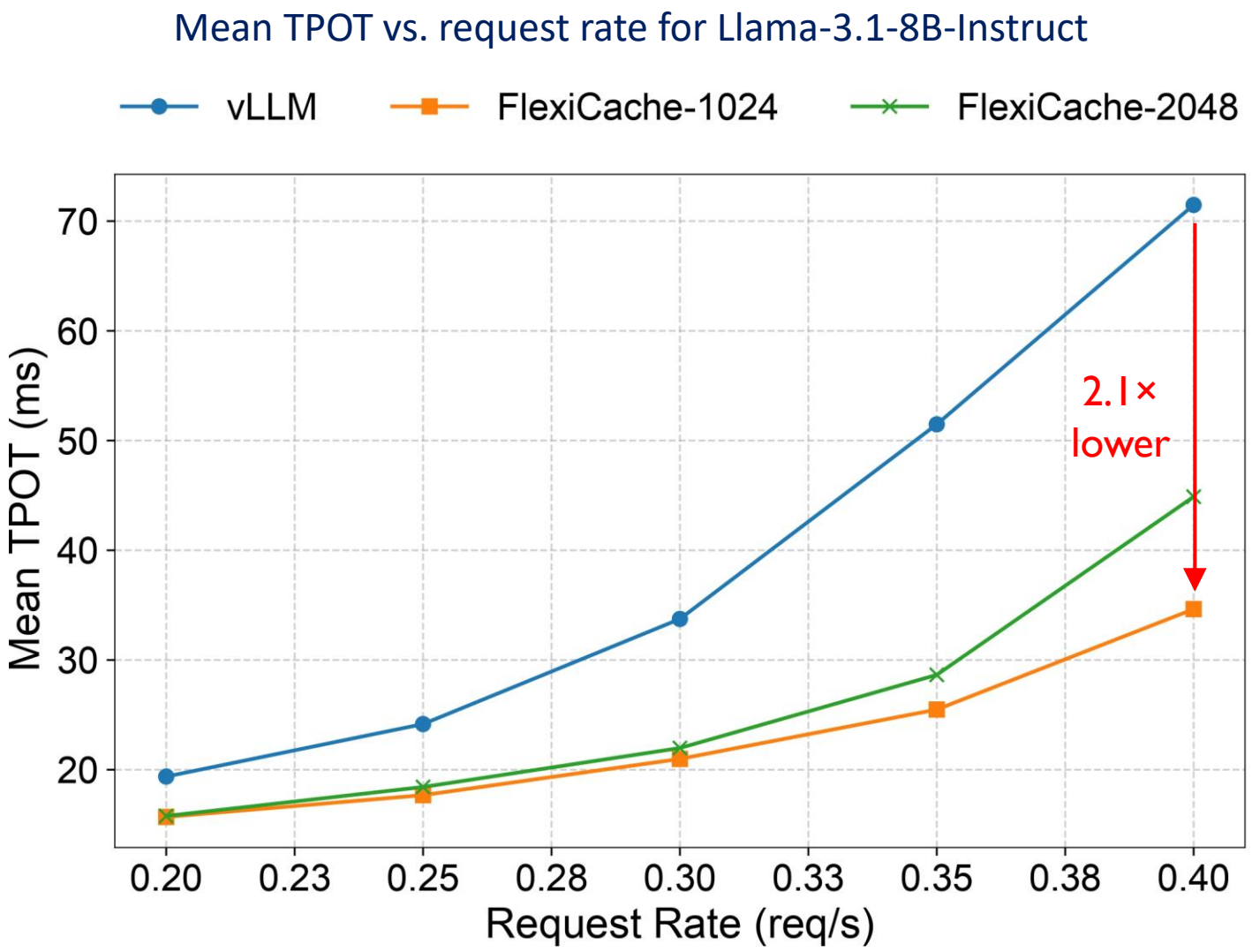
- FlexiCache consistently outperforms vLLM across output lengths
- Longer output → decoding dominates runtime → larger throughput gains
- Smaller Top-K budget → higher throughput
  FlexiCache-1024 > FlexiCache-2048

End-to-end token throughput for Llama-3.1-8B-Instruct on L-Eval



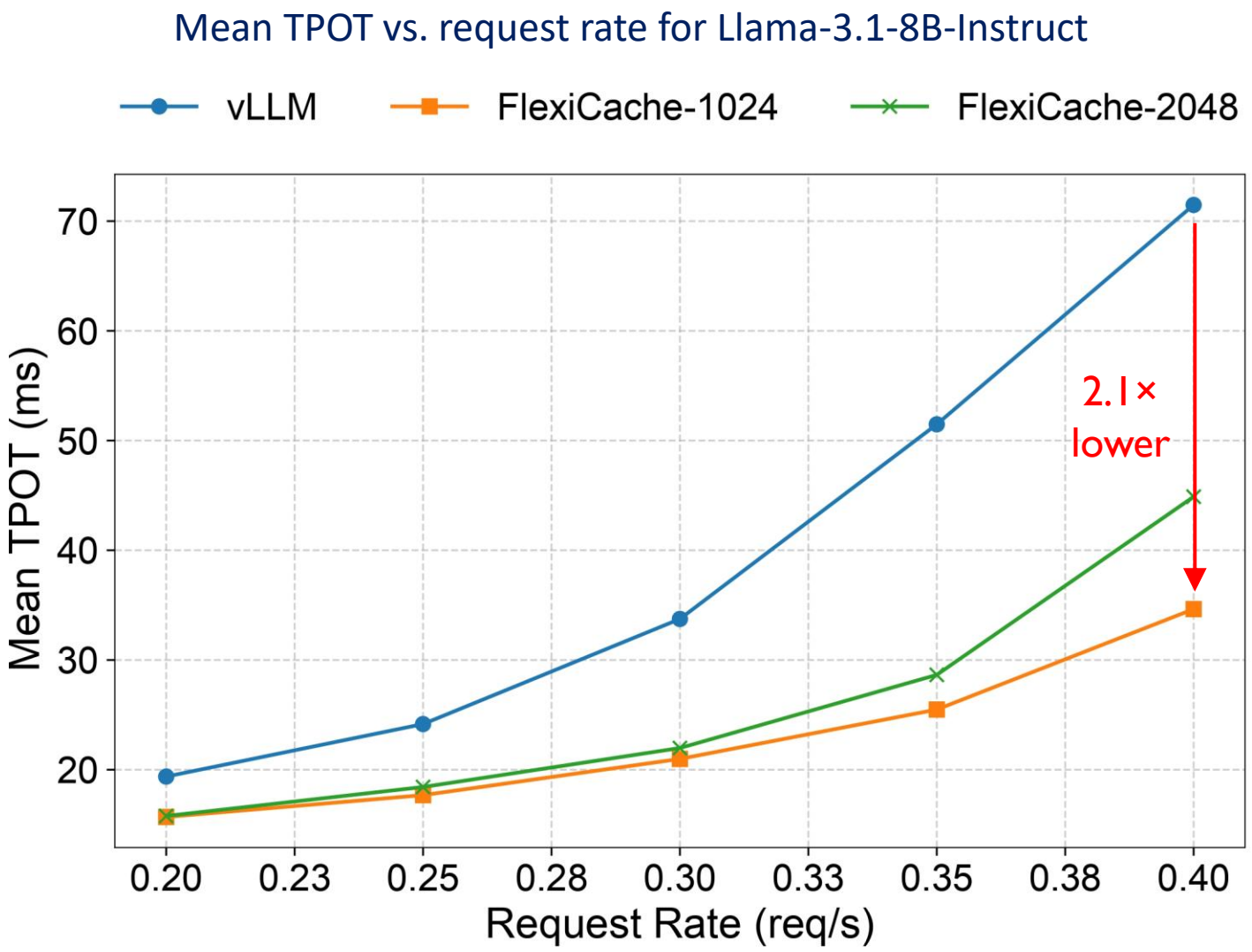
Online Serving: Up to 2.1× Lower TPOT

- FlexiCache reduces mean Time Per Output Token (TPOT) across all request rates



Online Serving: Up to 2.1× Lower TPOT

- FlexiCache reduces mean Time Per Output Token (TPOT) across all request rates
- Higher request rate → higher HBM pressure → larger FlexiCache gains



Accuracy Retention: ≈ Full-KV Accuracy

Accuracy retention on selected L-Eval tasks

Model	Method	Long-FQA	Gov-Report	CUAD	Multi-News	Retention Ratio Avg.
Llama 3.1 8B Instruct	Full KV	49.6	27.2	37.2	17.5	1.01
	FlexiCache	50.8	26.2	38.1	18.0	
Mistral 7B Instruct	Full KV	48.5	27.5	35.3	15.4	1.01
	FlexiCache	46.6	28.2	34.4	16.8	

- FlexiCache preserves accuracy close to Full KV across long-context long-generation tasks

Accuracy Retention: ≈ Full-KV Accuracy

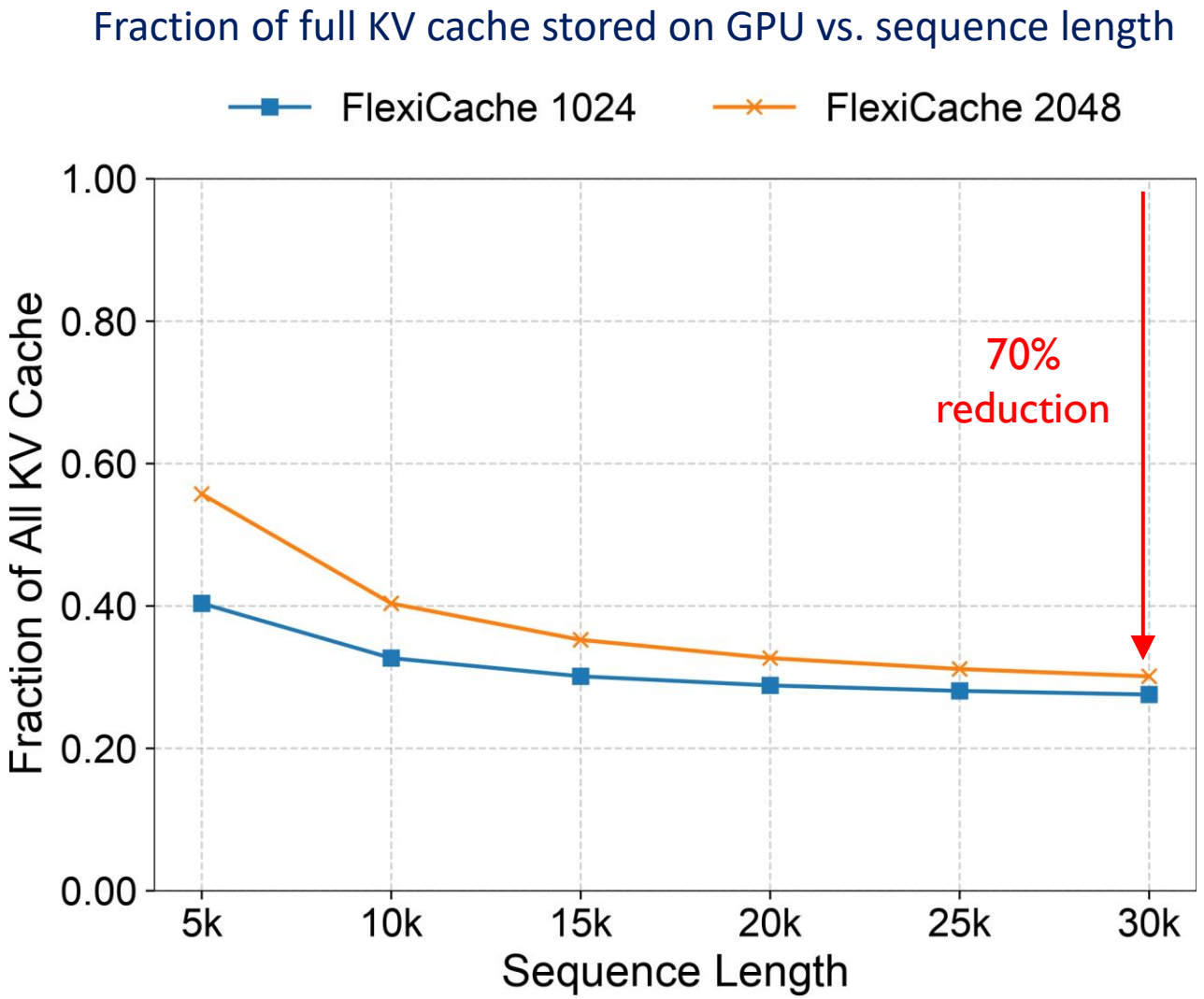
Accuracy retention on selected L-Eval tasks

Model	Method	Long-FQA	Gov-Report	CUAD	Multi-News	Retention Ratio Avg.
Llama 3.1 8B Instruct	Full KV	49.6	27.2	37.2	17.5	1.01
	FlexiCache	50.8	26.2	38.1	18.0	
Mistral 7B Instruct	Full KV	48.5	27.5	35.3	15.4	1.01
	FlexiCache	46.6	28.2	34.4	16.8	

- FlexiCache preserves accuracy close to Full KV across long-context long-generation tasks
- Paper reports similar accuracy retention across more models and datasets

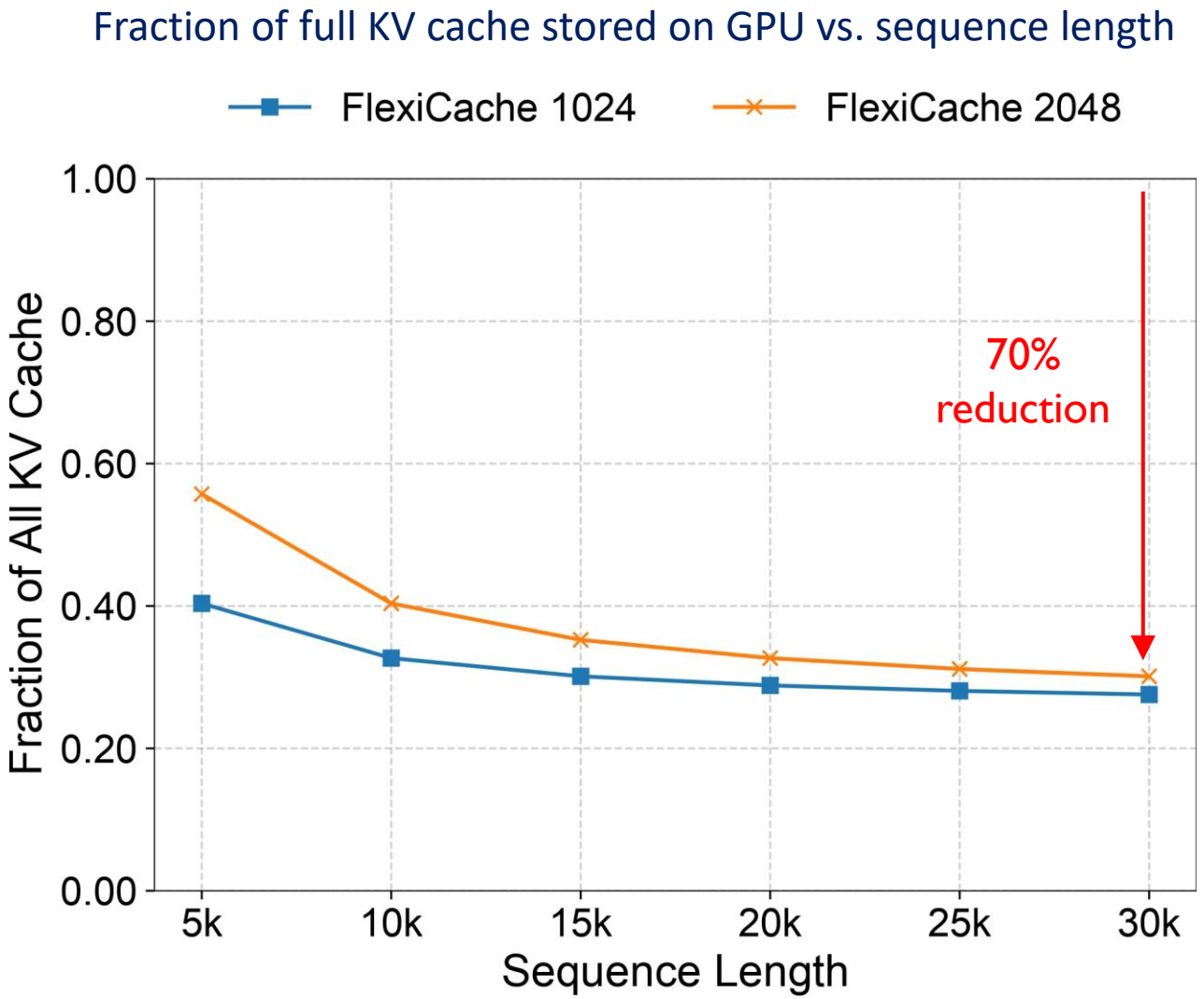
GPU Memory Savings: Over 70%

- GPU KV-cache fraction drops as sequence length increases
  - Why: stable heads keep a fixed Top-K budget on GPU



GPU Memory Savings: Over 70%

- GPU KV-cache fraction drops as sequence length increases
  - Why: stable heads keep a fixed Top-K budget on GPU
- With Top-K = 1024, FlexiCache achieves over 70% GPU memory savings beyond 20k tokens



Summary: FlexiCache

Summary: FlexiCache

- **Key observation:** Non-uniform temporal stability across attention heads

Summary: FlexiCache

- **Key observation:** Non-uniform temporal stability across attention heads
- **Design:**
  - Head-stability-aware hierarchical KV placement across GPU and host memory
  - Sparse decode attention over selected KV pages

Summary: FlexiCache

- **Key observation:** Non-uniform temporal stability across attention heads
- **Design:**
  - Head-stability-aware hierarchical KV placement across GPU and host memory
  - Sparse decode attention over selected KV pages
- **Benefits:**
  - **HBM capacity ↓ :** fewer KV pages on GPU
  - **HBM → SM I/O ↓ :** sparse decode
  - **Host → GPU I/O ↓ :** periodically reload newly promoted pages
  - **Compute ↓ :** sparse decode
  - **Accuracy preserved:** KV cache never permanently discarded

Summary: FlexiCache

- **Key observation:** Non-uniform temporal stability across attention heads
- **Design:**
  - Head-stability-aware hierarchical KV placement across GPU and host memory
  - Sparse decode attention over selected KV pages
- **Benefits:**
  - **HBM capacity ↓ :** fewer KV pages on GPU
  - **HBM → SM I/O ↓ :** sparse decode
  - **Host → GPU I/O ↓ :** periodically reload newly promoted pages
  - **Compute ↓ :** sparse decode
  - **Accuracy preserved:** KV cache never permanently discarded

https://github.com/NazmulTakbir/FlexiCache