Qualcomm

QuoKA

Query oriented KV Selection for Efficient LLM Prefill

Dalton Jones, M. Harper Langston, Junyoung Park, Matthew Morse, Mingu Lee, Chris Lott

ICLR

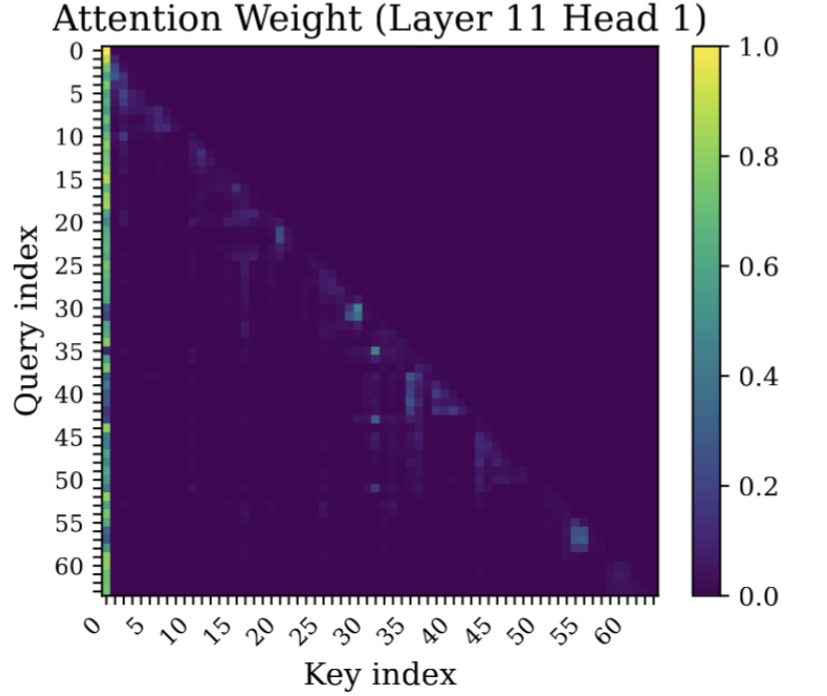
Qualcomm AI research

Qualcomm AI research

ICLR

Motivation

- **Context:** Large Language Models (LLMs) have heavy compute and memory costs during the *prefill* stage of inference.
- **Problem:** Attention computation over all key-value (KV) pairs becomes costly when inputs are long (e.g., long-context tasks).
- **Challenge:** Many sparse or KV-reduction techniques require retraining, special hardware, or compromise accuracy.



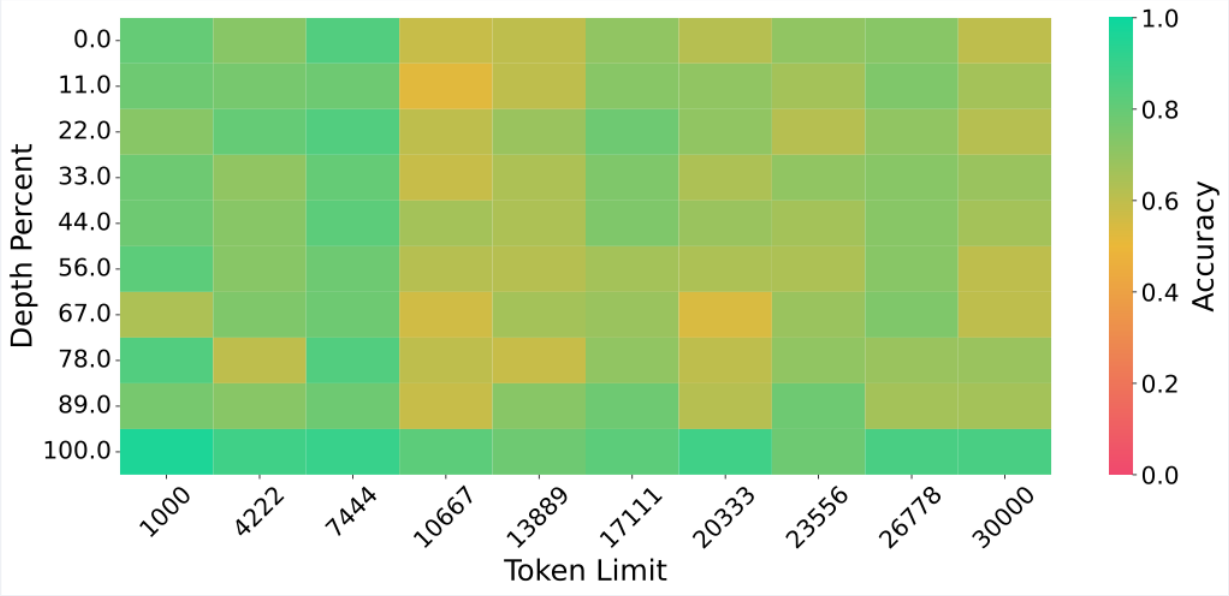
Qualcomm AI research

ICLR

Motivation

Many important LLM applications require long context processing:

- Long contexts allow LLMs to reason over research papers, books, articles, codebases etc.
- Long contexts preserve conversation history, user preferences, prior instructions.
- Essential for realistic chat, agents, and memory-heavy applications.
- Long contexts support multi-step reasoning, long planning chains, deep multi-document synthesis



Motivation

Problems with Long Context inference

Slow Time-to-First-Token (TTFT)

- LLM prefill requires computing attention over **large sequences**, which becomes slow as context length increases. In most cases this is compute bound.

Slow Per-Token Decode Latency

- During generation, **every new token must attend to the full history**, causing latency to grow with context length. Decode over long contexts transfers **hundreds of MBs of KV cache per token processed**, quickly dominating runtime. Memory **bandwidth**, not compute, becomes the limiting factor—especially on GPUs, NPUs, and mobile devices.

Long-Context Inference on Limited Hardware

- Edge devices, laptops, and consumer GPUs often lack sufficient VRAM, adequate memory bandwidth and high performance tensor cores for long-context inference without heavy sparsification.

An algorithm that can decrease the compute and memory requirements for attention will significantly decrease latency for TTFT and decode respectively

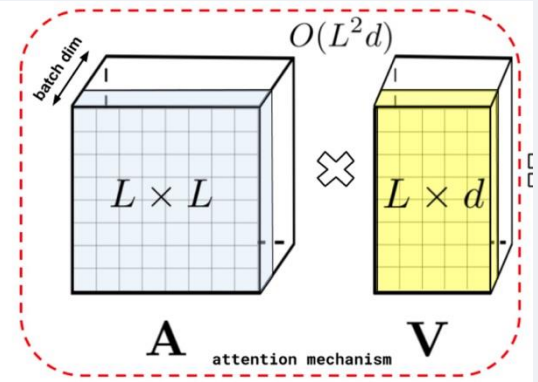
Qualcomm AI research

ICLR

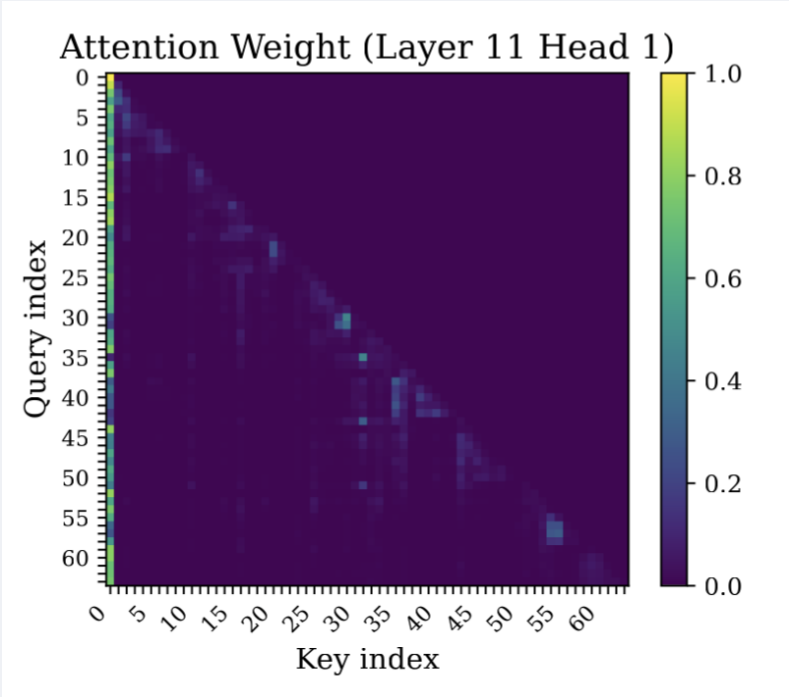
Motivation

Attention is very sparse

- Many keys will have very small inner product with all incoming queries, resulting in small attention weights
- These keys (and their corresponding value tokens) negligibly contribute to attention output
- If these can be efficiently identified, these KV pairs can be skipped in the attention computation, reducing both compute and memory requirements
- This can lead to significantly reduced latency and increased throughput, particularly on edge devices



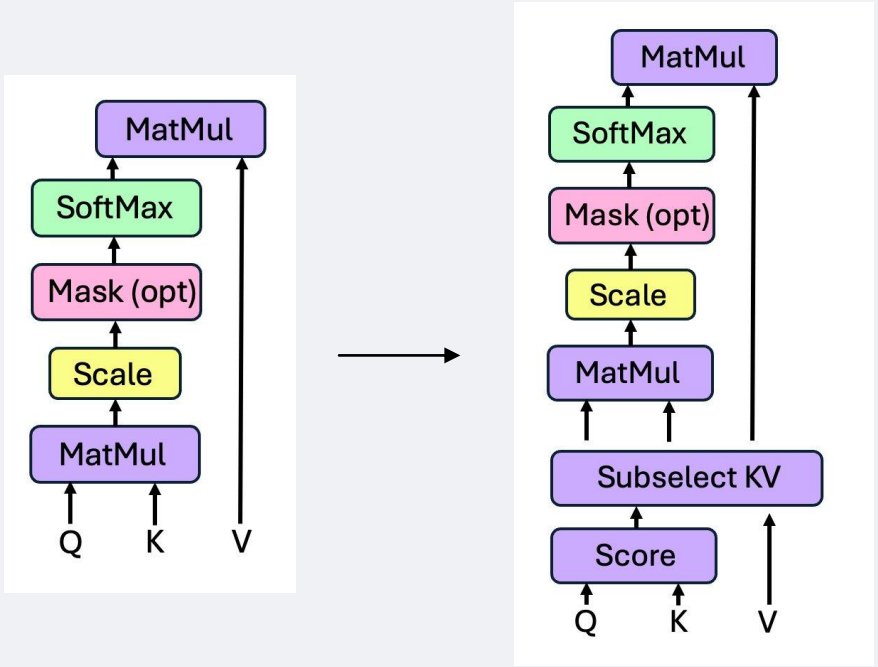
$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$



QuoKA: Query-oriented KV selection for efficient LLM Prefil:
D. Jones, M.H. Langston, J. Park, M. Morse, M. Lee, and C. Lott

What is QuoKA

- QuoKA is a training-free, hardware-agnostic sparse attention method tailored for chunked prefill, built entirely on standard linear algebra operations.
- Identify a key empirical insight: queries with low cosine similarity to the mean query interact with the most keys, enabling an efficient strategy for query selection.
- Propose a two-stage KV selection strategy—query subselection and cosine-similarity–based KV scoring—plus group-aware aggregation to maintain compatibility.
- Achieve significant latency improvements, across long-context benchmarks.
- Demonstrates broad generalization and robustness, showing stable performance across sparsity levels and diverse LLM families (Llama3, Qwen3, SmolLM3, GPT-OSS) and architectural variants (RoPE/NoPE, MoE).
- Results include ***3× lower TTFT, 5× GPU attention speedup, and up to 7× CPU speedup, while using 88% fewer KV pairs and maintaining near-baseline accuracy***



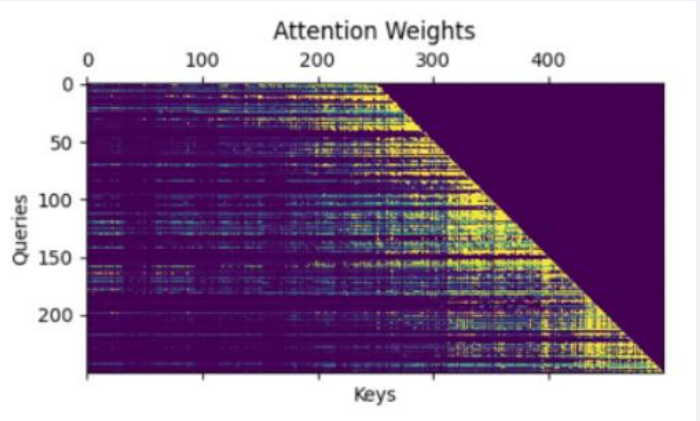
Qualcomm AI research

ICLR

Sparse Attention and Importance Estimation

Attention is Expensive

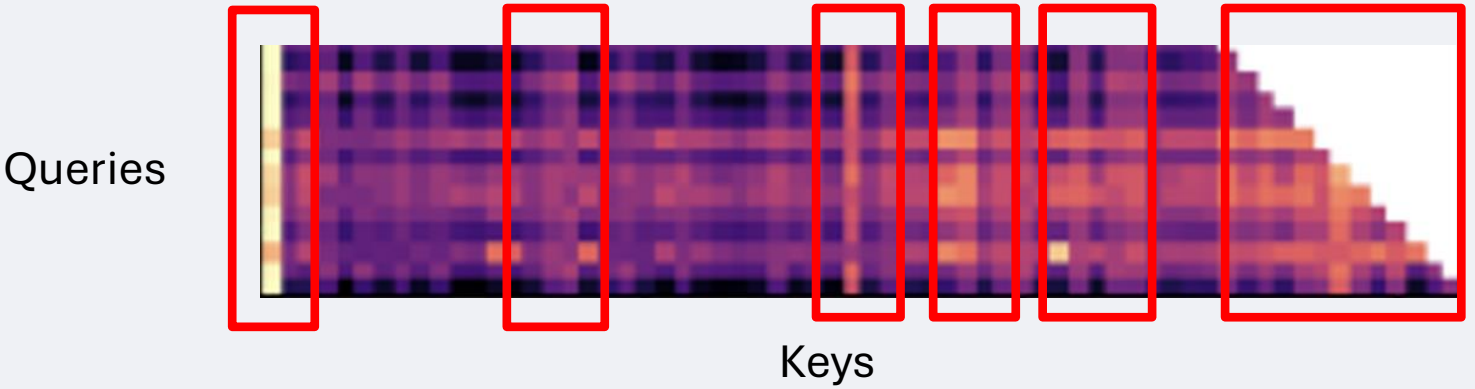
- Attention scales quadratically with sequence length
- Most of the quadratic computation is wasted on near-zeros in $softmax(QK^T)\,V$



Attention is Sparse

- Attention matrices $softmax(QK^T)$ are very sparse with most values close to zero.
- This enhances selectivity of transformer heads and avoids instability.
- Keys with small attention values do not significantly contribute to attention output

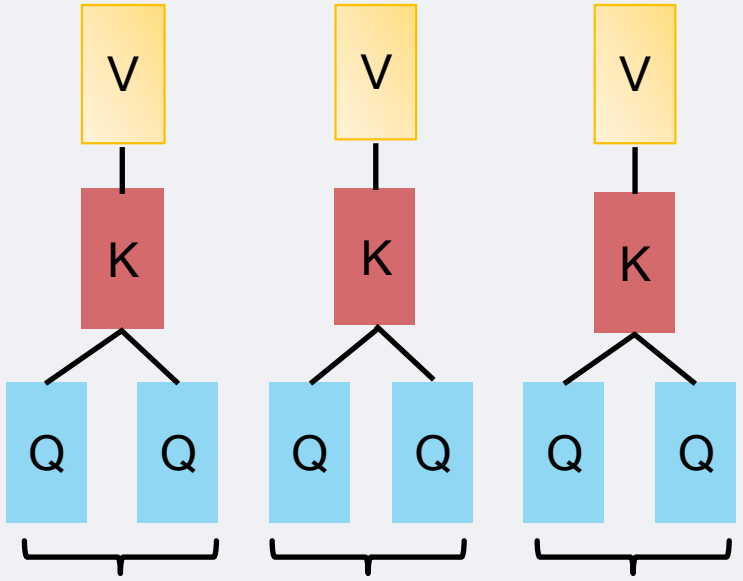
$softmax(QK^T)$



QuoKA: Query-oriented KV selection for efficient LLM Prefil:
D. Jones, M.H. Langston, J. Park, M. Morse, M. Lee, and C. Lott

Sparse Attention and Importance Estimation

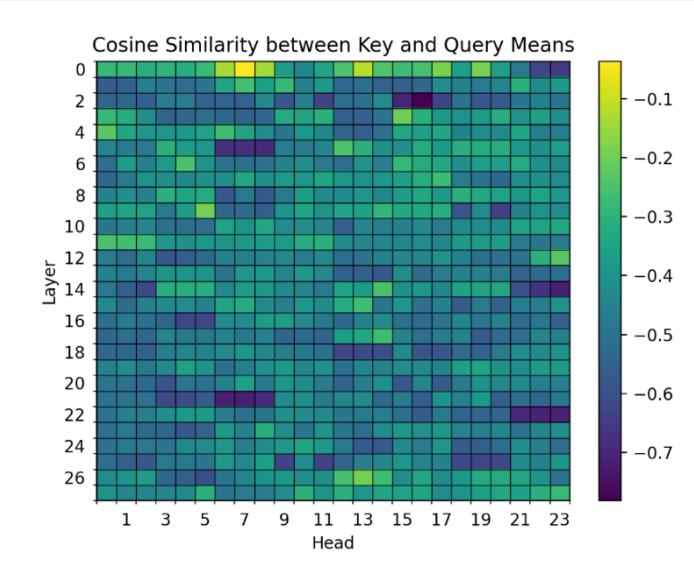
- Prior work has demonstrated that the dot product between normalized queries and keys is an accurate proxy for key importance when aggregated across queries, heads.
- Individual keys influence queries across groups in GQA, we can average these query groups prior to dot product to compute average influence Q heads
- This reduces scoring compute requirements by a a factor of $\frac{number\ of\ query\ heads}{number\ of\ key\ heads}$

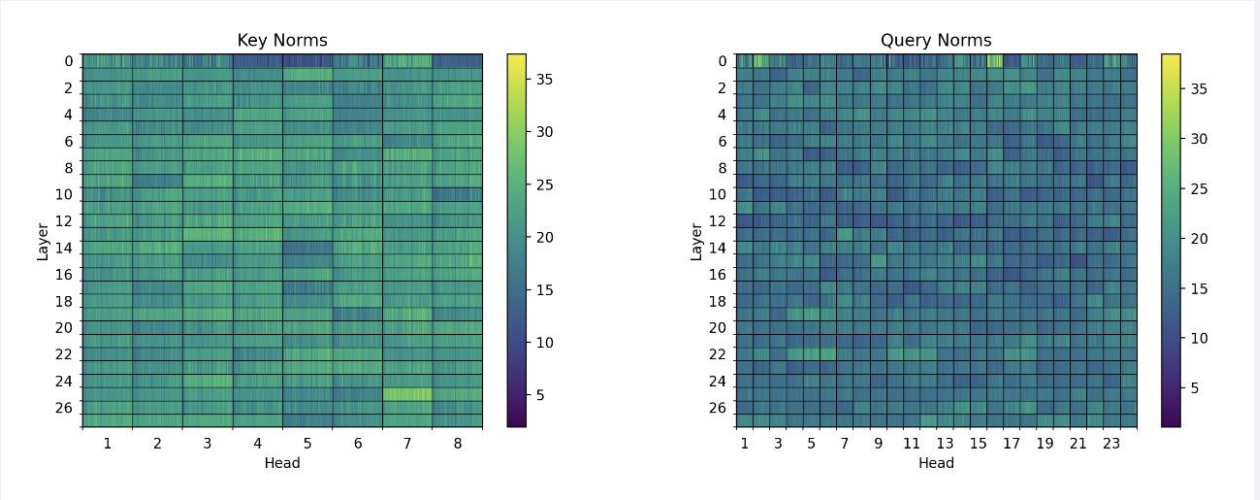


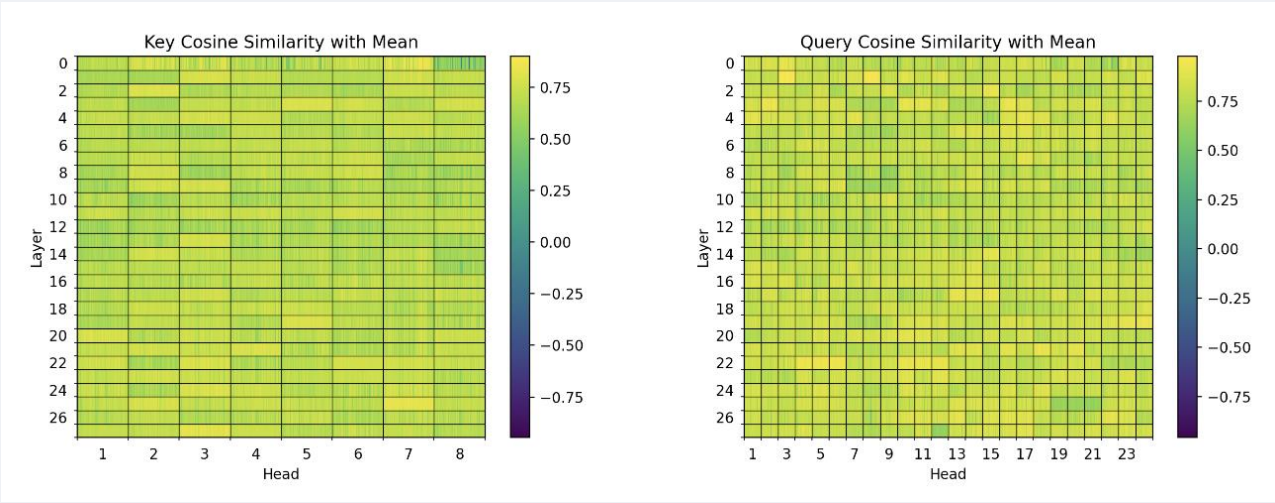
Average Q over KV heads prior to scoring

Key and Query Geometry

- Key and query L2 norms are bounded within all layers of LLM (due to LayerNorms)
- Key tend to **cluster** angularly around their average
- Queries also tend to cluster angularly around their average
- Majority of geometric deviation can be attributed to RoPE
- Some important tokens have more unique geometry which is **easily detectable**
- In essence, most keys point in the same direction and most queries point in the same direction and have bounded norm. High attention weights come from keys and queries pointing in **more similar directions**.







Key and Query Geometry

Theorem B.1. *Suppose that for a fixed query token q, there is a set of key tokens $\{k_i\}_{i=1}^n$ such that $||k_i||_2^2 < M,\ \forall\ i$. Without loss of generality suppose $||q|| = 1$ and assume k^* is a key not in $\{k_i\}_{i=1}^n$ with $||k^*||_2^2 < M$ that has attention weight $w > 0$:*

$$w = \frac{\exp\left(k^{*\top} q\right)}{\exp\left(k^{*\top} q\right) + \sum_{i=1}^{n} \exp\left(k_i^{\top} q\right)}.$$

Then

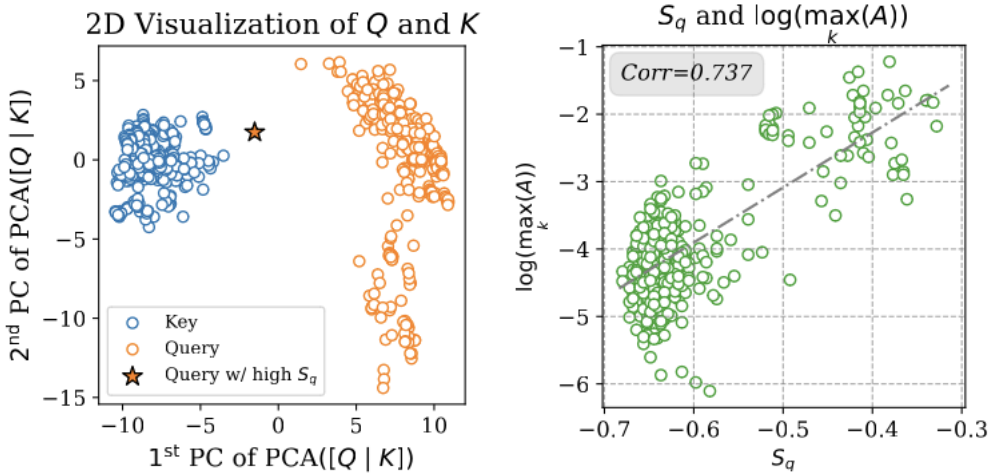
$$\frac{\log\left(\frac{n}{n+1}\right) - \log(1-w)}{2M} - 1 \leq CosSim(k^*, q)$$

- Cosine Similarity correlates with attention scores and is more uniform across layers, leading to more stable scoring after aggregation

Query Subselection and Information Redundancy

Geometric properties of keys and queries within attention heads can guide query subselection

- Attention plot shows some queries are consistently higher scoring across keys, these are more important to use for attention estimation
- We see using PCA that keys and queries tend to cluster amongst themselves and that higher scoring queries tend to be near the bulk of the keys
- The score we use to determine query subselection, S_q, also defined as the *negative cosine similarity* of queries with their mean, correlates with query tokens with high maximum attention score. These tokens are more important at the current step to retain the mixing properties of the attention module.



(b) PCA viz. of Q and K (c) Scatter plot S_q v.s. $\max_k(A)$

Qualcomm AI research

ICLR

Quoka Algorithm

- After query subselection and mean pooling over head dimension, compute cossim with keys
- Max pooling over query dimension yields quoka score
- The top k scores are collected and KV pairs corresponding to these scores are passed to attention. This is the QuoKA algorithm

Algorithm 1 KV cache sub-selection using QUOKA

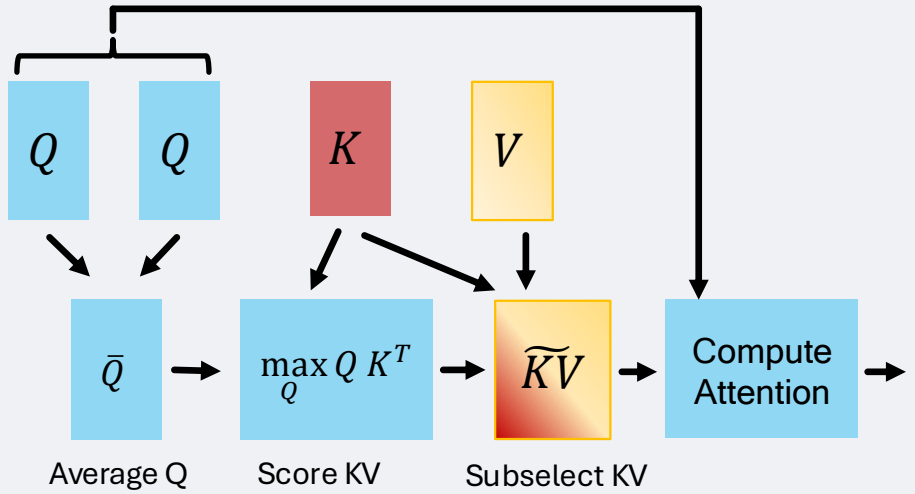
Require: queries Q, keys K, values V, prefill chunk size B_{CP}, selective attention budget B_{SA}, max queries N_Q, number of KV heads n_{KV}, number of attention heads n_{Q}

▷ Query Sub-selection

if $B_{\text{CP}} > N_Q$ **then**
  $M_Q \leftarrow \texttt{mean}(Q, \texttt{dim=2})$
  $S_Q \leftarrow \text{CosSim}(Q, M_Q)$
  $Q \leftarrow \texttt{gather}(\texttt{topk}(-S_Q, N_Q), Q)$
end if

▷ Efficient Cosine Similarity via Pre-aggregation

$Q \leftarrow Q/\texttt{norm}(Q, \texttt{dim=-1})$ ▷ (b, n_q, N_Q, d)
$K \leftarrow K/\texttt{norm}(K, \texttt{dim=-1})$ ▷ (b, n_{KV}, T, d)
$\bar{Q} \leftarrow \texttt{mean}\left(Q.\texttt{reshape}(b, n_{\text{KV}}, \frac{n_{\text{Q}}}{n_{\text{KV}}}, N_Q, d), \texttt{dim=2}\right)$ ▷ (b, n_{KV}, N_Q, d)
$S \leftarrow \bar{Q}K^{\top}$ ▷ $(b, n_{\text{KV}}, N_Q, T)$
$\hat{S} \leftarrow \max(S, \texttt{dim=2})$ ▷ (b, n_{KV}, T)
$I \leftarrow \texttt{topk}(\hat{S}, B_{\text{SA}})$
$K^{\star}, V^{\star} \leftarrow \texttt{gather}(K, I), \texttt{gather}(V, I)$



QuoKA: Query-oriented KV selection for efficient LLM Prefil:
D. Jones, M.H. Langston, J. Park, M. Morse, M. Lee, and C. Lott

Qualcomm AI research

ICLR

Quoka Algorithm

- **Group-aware query aggregation** (GQA-compatibility): For architectures with GQA, representative queries regrouped and averaged per KV-head group, ensuring key scoring aligns with how queries are shared across heads.
- **Two-stage KV selection pipeline**: ***Stage 1***: compute and subselect *representative queries*; average these among key value groups. ***Stage 2***: compute *cosine similarity scorings* with tokens from Stage 1 and normalized keys; use resulting score and *topk* to subselect highest scoring KV pairs to form reduced KV cache.
- **Training-free sparse attention using chunked prefill**: QuoKA performs KV selection independently for each prefill chunk, relying on standard linear algebra ops and no model modifications, enabling efficient and portable sparsity.
- **Unlike learned sparse attention or redesigns, QuoKA seamlessly drops into existing attention modules without training.**

Algorithm 1 KV cache sub-selection using QUOKA

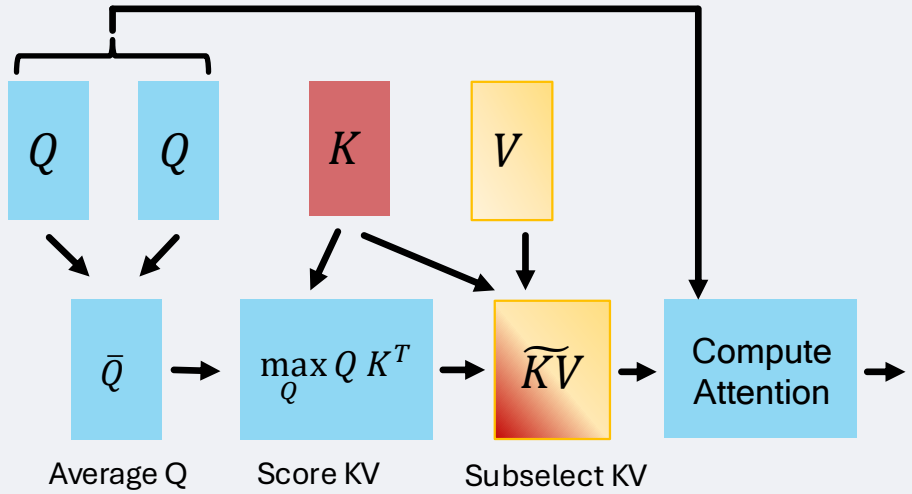
Require: queries Q, keys K, values V, prefill chunk size B_{CP}, selective attention budget B_{SA}, max queries N_Q, number of KV heads n_{KV}, number of attention heads n_{Q}

▷ Query Sub-selection

if $B_{\text{CP}} > N_Q$ **then**
  $M_Q \leftarrow \texttt{mean}(Q, \texttt{dim=2})$
  $S_Q \leftarrow \text{CosSim}(Q, M_Q)$
  $Q \leftarrow \texttt{gather}(\texttt{topk}(-S_Q, N_Q), Q)$
end if

▷ Efficient Cosine Similarity via Pre-aggregation

$Q \leftarrow Q/\texttt{norm}(Q, \texttt{dim=-1})$ ▷ (b, n_q, N_Q, d)
$K \leftarrow K/\texttt{norm}(K, \texttt{dim=-1})$ ▷ (b, n_{KV}, T, d)
$\bar{Q} \leftarrow \texttt{mean}\left(Q.\texttt{reshape}(b, n_{\text{KV}}, \frac{n_{\text{Q}}}{n_{\text{KV}}}, N_Q, d), \texttt{dim=2}\right)$ ▷ (b, n_{KV}, N_Q, d)
$S \leftarrow \bar{Q}K^{\top}$ ▷ $(b, n_{\text{KV}}, N_Q, T)$
$\hat{S} \leftarrow \max(S, \texttt{dim=2})$ ▷ (b, n_{KV}, T)
$I \leftarrow \texttt{topk}(\hat{S}, B_{\text{SA}})$
$K^{\star}, V^{\star} \leftarrow \texttt{gather}(K, I), \texttt{gather}(V, I)$



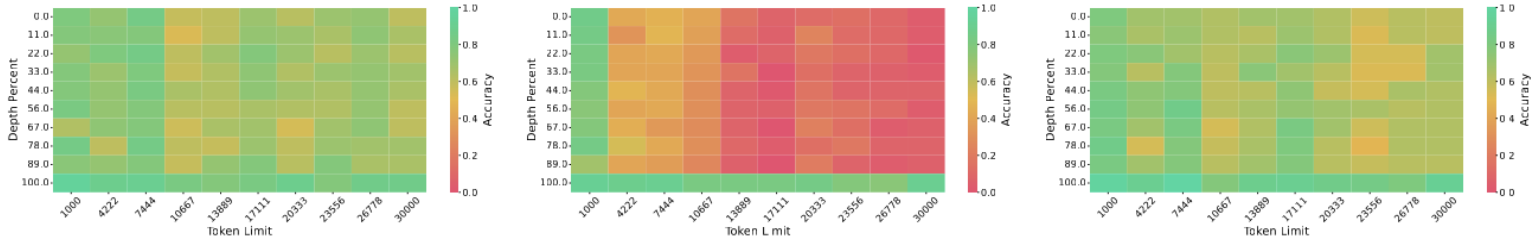
QuoKA: Query-oriented KV selection for efficient LLM Prefil:
D. Jones, M.H. Langston, J. Park, M. Morse, M. Lee, and C. Lott

Qualcomm AI research

ICLR

QuoKA Results: Long Context Benchmarks

- QuoKA achieves baseline performance on NIAH tasks up to 32k with selective budget 2048
- Longbench results demonstrate near baseline performance for QuoKA, much better than other sparse attention methods applied to chunked prefill.



(a) QUOKA (b) SampleAttention (c) Full

Figure 4: Accuracy across document length and needle depth for NIAH with $B_{SA} = 2048$ and $B_{CP} = 128$. Results for additional sparse attention methods are shown in Figure 7.

Table 3: **LongBench results (Higher is better)**. For each model, the three columns correspond to selective budgets $B_{SA} \in \{512, 1024, 2048\}$.

	Llama3.2-3B-Instruct			*Qwen2.5-3B-Instruct*			*Qwen3-4B*			*Smollm3*		
Budget	512	1024	2048	512	1024	2048	512	1024	2048	512	1024	2048
LessIsMore	0.703	0.788	0.850	0.461	0.556	0.659	0.665	0.773	0.868	0.765	0.842	0.918
SparQ	0.721	0.802	0.842	0.636	0.726	0.808	0.686	0.782	0.872	0.725	0.805	0.922
Loki	0.686	0.757	0.842	0.589	0.671	0.787	0.622	0.782	0.872	0.384	0.801	0.622
SampleAttn	0.738	0.800	0.901	0.660	0.756	0.828	0.755	0.875	0.947	0.856	0.929	0.966
QUOKA	**0.945**	**0.972**	**0.986**	**0.869**	**0.945**	**0.977**	**0.966**	**0.992**	**0.995**	**0.998**	**1.03**	**1.028**

Qualcomm AI research

ICLR

QuoKA Results: Long Context Benchmarks

- QuoKA performs significantly better on the benchmark when using both a fixed selective budget and with a selective budget as the percentage of current cache size
- Results persist across model architecture

Table 2: **RULER eval of QUOKA** with B_{SA} set to 25% of the KV cache length

Model	Budget	Prompt Length			
		4096	8192	16384	32768
Llama3.2-3B	Full	87.50	81.33	78.98	76.31
	25%	86.94	79.72	76.02	74.14
Qwen2.5-3B	Full	89.56	81.99	76.09	71.69
	25%	86.07	79.78	74.25	68.84
Qwen3-4B	Full	93.32	91.68	91.18	88.54
	25%	92.50	91.35	90.63	87.87
Qwen3-30B A3B-Instruct	Full	94.08	93.75	92.02	91.87
	25%	93.25	92.77	91.90	91.08
Smollm3	Full	91.12	83.46	80.11	75.18
	25%	89.60	81.45	78.72	73.55
GPT-OSS-20B	Full	79.35	79.32	79.78	75.47
	25%	77.40	80.66	77.17	73.45

Table 1: **RULER evaluation results across increasing lengths with** $B_{SA} = 1,024$. Highest per column in **bold** (Higher is better). Attention sparsification applied on full attention layers.

	Llama-3.2-3B-Instruct				*Qwen-2.5-3B-Instruct*				*Qwen-3-4B*				*Smollm3*				*GPT-OSS-20B*			
Length	4k	8k	16k	32k	4k	8k	16k	32k	4k	8k	16k	32k	4k	8k	16k	32k	4k	8k	16k	32k
SnapKV	29.15	13.70	12.44	6.21	19.25	12.40	9.88	8.13	27.29	17.15	10.94	10.07	43.72	35.98	24.21	12.23	58.18	31.45	24.82	16.63
KeyDiff	53.34	31.10	24.29	14.87	34.09	20.68	15.79	10.32	37.29	27.15	20.94	12.07	62.85	51.60	41.64	30.37	69.58	28.76	15.86	9.65
LessIsMore	75.15	49.23	30.44	19.16	36.66	20.21	12.84	10.12	65.55	42.75	24.39	14.87	79.67	50.17	35.12	24.21	67.35	54.49	38.27	20.11
Loki	74.84	56.50	25.76	8.05	74.40	60.09	48.96	34.12	82.83	65.19	52.29	39.31	84.52	64.20	50.10	22.66	75.48	65.36	54.67	39.92
SparQ	79.36	60.80	48.59	31.14	78.07	59.87	54.71	36.74	87.93	68.97	56.02	35.20	82.45	57.62	32.18	18.69	70.07	54.00	30.75	15.20
SampleAttn	78.25	61.14	48.31	31.73	77.17	60.88	56.64	36.17	87.84	72.46	59.57	40.72	85.72	66.44	59.10	45.98	76.20	70.35	53.91	30.42
QUOKA	**86.71**	**80.19**	**70.90**	**57.01**	**87.85**	**74.27**	**66.82**	**59.37**	**93.73**	**91.07**	**88.57**	**74.83**	**89.97**	**79.94**	**72.69**	**61.37**	**78.92**	**79.19**	**73.40**	**57.79**

QuoKA: Query-oriented KV selection for efficient LLM Prefil:
D. Jones, M.H. Langston, J. Park, M. Morse, M. Lee, and C. Lott

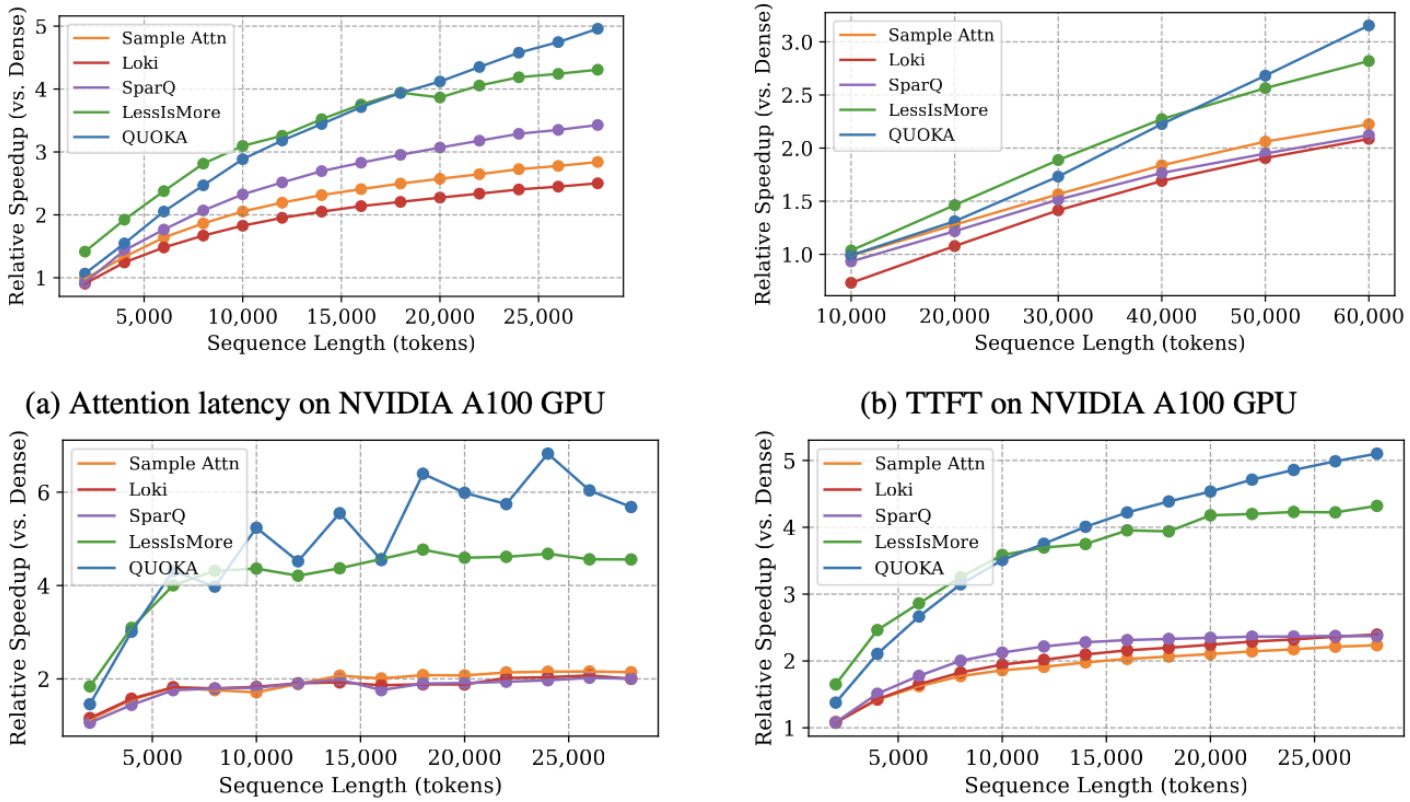
Qualcomm AI research

ICLR

Quoka Complexity and Timing

- Quoka has superior theoretical memory/compute complexity than attention and other sparse attention methods
- TTFT and attention module latency significantly reduced by up to 3-5x on different hardware.

	Runtime Complexity	Memory Complexity
QUOKA	$\mathcal{O}(B_{\text{CP}} + (N_Q(1 + dn_{\text{KV}}))T)$	$\mathcal{O}(n_{\text{KV}} N_Q T)$
SampleAttention	$\mathcal{O}((dn_{\text{Q}} + n_{\text{Q}}/n_{\text{KV}} + n_{\text{KV}})N_Q T)$	$\mathcal{O}(n_{\text{Q}} N_Q T)$
SparQ	$\mathcal{O}(B_{\text{CP}} T d_l n_{\text{Q}})$	$\mathcal{O}(n_{\text{Q}} B_{\text{CP}} T)$
Loki	$\mathcal{O}(d_l n_{\text{Q}}(B_{\text{CP}} T + d(B_{\text{CP}} + T)))$	$\mathcal{O}(n_{\text{Q}} B_{\text{CP}} T)$
LessIsMore	$\mathcal{O}(dn_{\text{Q}} B_{\text{CP}} T/L)$	$\mathcal{O}(n_{\text{Q}} B_{\text{CP}} T/L)$



(a) Attention latency on NVIDIA A100 GPU

(b) TTFT on NVIDIA A100 GPU

(c) Attention latency on Intel Xeon W-2125 CPU

(d) Attention latency on NVIDIA RTX 2080 GPU

Figure 5: Relative speedup of attention and TTFT compared to dense attention baseline using $B_{\text{CP}} = 128$ on different hardware.

QuoKA: Query-oriented KV selection for efficient LLM Prefil:
D. Jones, M.H. Langston, J. Park, M. Morse, M. Lee, and C. Lott

Qualcomm AI research

ICLR

Conclusion and Next Steps

- QuoKA shows that **query-oriented KV selection** enables efficient, training-free sparse attention for long-context prefill.
- Achieves large latency reductions while preserving accuracy across architectures and model families(Cosine similarity ensures bounded, scale-invariance).
- Results include ***3× lower TTFT, 5×GPU attention speedup, and up to7× CPU speedup, while using 88% fewer KV pairs and maintaining near-baseline accuracy***
- Future work includes adaptive query selection, per-layer and head adaptive budgeting, extensions beyond prefill, and integration with complementary efficiency techniques.

Please visit us during our poster session …!

Qualcomm AI research

QuoKA: Query-oriented KV selection for efficient LLM Prefill

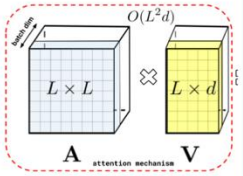
Dalton Jones†, M. Harper Langston, Junyoung Park, Matthew Morse, Mingu Lee, Chris Lott
Qualcomm AI Research*

†daltjone@qti.qualcomm.com

ICLR

Motivation

- **Transformers** have become the backbone of modern models in **vision, language, and beyond**. But **self-attention mechanism** suffers from:
  - Quadratic compute and memory complexity
  - Poor scalability for on-device or embedded applications
- These limitations hinder deployment in real-time, mobile, and resource-constrained environments
- While many efficient alternatives exist, they often compromise generality or require redesigning the architecture

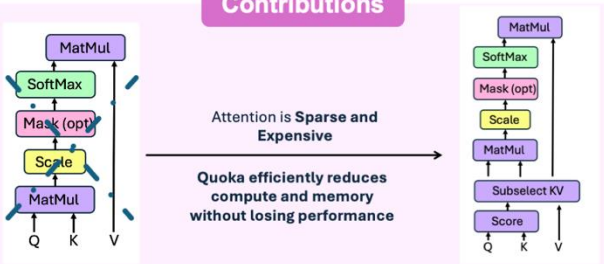


$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

We present QuoKA (Query-oriented KV selection for efficient LLM Prefill), a training-free generalized method to reduce the computation and memory complexity of attention without sacrificing accuracy with the following benefits:

Full Attention	Prior Sparse Methods	Quoka
Quadratic compute & memory	Often training-dependent	**Subquadratic**
Accurate but slow	Architecture changes	**Training-free drop-in**
Hardware-unfriendly ops (e.g., softmax)	Complex kernels	**Pure linear algebra**

Contributions



- Introduce QuoKA, a training-free, hardware-agnostic sparse attention method tailored for chunked prefill, built entirely on standard linear algebra operations.
- Identify a key empirical insight: queries with low cosine similarity to the mean query interact with the most keys, enabling an efficient strategy for query selection.
- Propose a two-stage KV selection strategy—query subselection and cosine-similarity–based KV scoring—plus group-aware aggregation to maintain compatibility.
- Achieve significant latency improvements, across long-context benchmarks.
- Demonstrates broad generalization and robustness, showing stable performance across sparsity levels and diverse LLM families (Llama3, Qwen3, SmolLM, GPT-OSS) and architectural variants (RoPE/NoPE, MoE).
- Results include ***3× lower TTFT, 5× GPU attention speedup, and up to 7× CPU speedup, while using 88% fewer KV pairs and maintaining near-baseline accuracy***

Method

QuoKA: training-free, hardware-friendly sparse attention that reduces prefill latency by efficiently identifying high-impact KV subsets.

- Query subselection via **cosine dissimilarity**: Quoka identifies representative queries, computing cosine similarity between each query and mean query; only queries with lowest similarity (i.e., most informative, highest diversity) are retained for KV selection.
- **Cosine-normalized scoring for stable KV relevance** estimation: Both queries and keys are L2-normalized, enabling efficient computation of bounded query–key relevance score using cosine similarity, proving stable ranking across layers and heads.

Algorithm 1 KV cache sub-selection using QUOKA

Require: queries Q, keys K, values V, prefill chunk size B_{CP}, selective attention budget B_{SA}, max queries N_Q, number of KV heads n_{KV}, number of attention heads n_Q

▷ Query Sub-selection

1: **if** $B_{\text{CP}} > N_Q$ **then**
2: $M_Q \leftarrow \text{mean}(Q, \text{dim=2})$
3: $S_Q \leftarrow \text{CosSim}(Q, M_Q)$
4: $Q \leftarrow \text{gather}(\text{topk}(-S_Q, N_Q), Q)$
5: **end if**

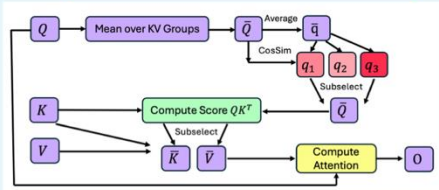
▷ Efficient Cosine Similarity via Pre-aggregation

6: $Q \leftarrow Q/\text{norm}(Q, \text{dim=-1})$ ▷ (b, n_q, N_Q, d)
7: $K \leftarrow K/\text{norm}(K, \text{dim=-1})$ ▷ (b, n_{KV}, T, d)
8: $\bar{Q} \leftarrow \text{mean}\left(Q.\text{reshape}(b, n_{\text{KV}}, \frac{n_Q}{n_{\text{KV}}}, N_Q, d), \text{dim=2}\right)$ ▷ $(b, n_{\text{KV}}, N_Q, d)$
9: $S \leftarrow \bar{Q}K^\top$ ▷ $(b, n_{\text{KV}}, N_Q, T)$
10: $\hat{S} \leftarrow \max(S, \text{dim=2})$ ▷ (b, n_{KV}, T)
11: $I \leftarrow \text{topk}(\hat{S}, B_{\text{SA}})$
12: $K^*, V^* \leftarrow \text{gather}(K, I), \text{gather}(V, I)$

Key Intuition: Queries that differ most from the average query capture rare or informative token interactions and therefore touch the largest set of relevant keys.

- **Group-aware query aggregation** (GQA-compatibility): For architectures with GQA, representative queries regrouped and averaged per KV-head group, ensuring key scoring aligns with how queries are shared across heads.
- **Two-stage KV selection pipeline**: ***Stage 1***: compute and subselect *representative queries*; average these among key value groups. ***Stage 2***: compute *cosine similarity scorings* with tokens from Stage 1 and normalized keys; use resulting score and *topk* to subselect highest scoring KV pairs to form reduced KV cache.
- **Training-free sparse attention using chunked prefill**: QuoKA performs KV selection independently for each prefill chunk, relying on standard linear algebra ops and no model modifications, enabling efficient and portable sparsity.
- **Unlike learned sparse attention or redesigns, QuoKA seamlessly drops into existing attention modules.**

Implementation



- No softmax / complex kernels
- Pure matmul + norm + reduction / standard linear algebra operations
- Hardware agnostic / Training-free drop-in

Complexity

	Runtime Complexity
QUOKA	$\mathcal{O}(B_{\text{CP}} + (N_Q(1 + dn_{\text{KV}}))T)$
SampleAttention	$\mathcal{O}((dn_Q + n_Q/n_{\text{KV}} + n_{\text{KV}})N_Q T)$
SparQ	$\mathcal{O}(B_{\text{CP}} T d_i n_Q)$
Loki	$\mathcal{O}(d_i n_Q(B_{\text{CP}}T + d(B_{\text{CP}} + T)))$
LessIsMore	$\mathcal{O}(dn_Q B_{\text{CP}} T/L)$

	Memory Complexity
QUOKA	$\mathcal{O}(n_{\text{KV}} N_Q T)$
SampleAttention	$\mathcal{O}(n_Q N_Q T)$
SparQ	$\mathcal{O}(n_Q B_{\text{CP}} T)$
Loki	$\mathcal{O}(n_Q B_{\text{CP}} T)$
LessIsMore	$\mathcal{O}(n_Q B_{\text{CP}} T/L)$

Memory and computational complexity significantly lower than full attention and improves on SoTA sparse methods

Results

- We implement **QuoKA** in the attention module of models with a variety of architectures, compare this with competing sparse attention methods and test using several long context benchmarks. **

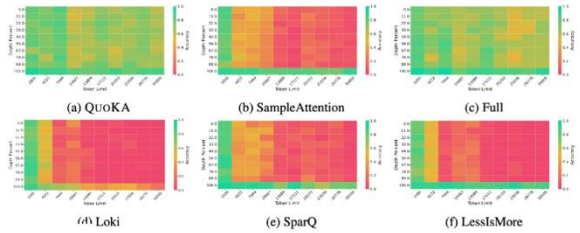


Figure 7: Accuracy across document length and needle depth for NIAH using additional sparse attention methods with $B_{\text{SA}} = 2048$ and $B_{\text{CP}} = 128$.

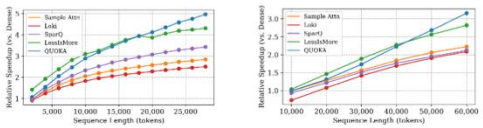
Table 1: **RULER evaluation results across increasing lengths with $B_{\text{SA}} = 1,024$.** Highest per column in **bold** (Higher is better). Attention sparsification applied on full attention layers.

	Llama-3.2-3B-Instruct				*Qwen-2.5-3B-Instruct*				*Qwen-3-4B*				*Smollm3*				*GPT-OSS-20B*			
Length	4k	8k	16k	32k	4k	8k	16k	32k	4k	8k	16k	32k	4k	8k	16k	32k	4k	8k	16k	32k
SnapKV	29.15	13.70	12.44	6.21	19.25	12.40	9.88	8.13	27.29	17.15	10.94	10.07	43.72	35.98	24.21	12.23	58.18	31.45	24.82	16.63
KeyDiff	53.34	31.10	24.29	14.87	34.09	20.68	15.79	10.32	37.29	27.15	20.94	12.07	62.85	51.60	41.64	30.37	69.58	28.76	15.86	9.65
LessIsMore	75.15	49.23	30.44	19.16	36.66	20.21	12.84	10.12	65.55	42.75	24.39	14.87	79.67	50.17	35.12	24.21	67.35	54.49	38.27	20.11
Loki	74.84	56.50	25.76	8.05	74.40	60.09	48.96	34.12	82.83	65.19	52.29	39.31	84.52	64.20	50.10	22.66	75.48	65.36	54.67	39.92
SparQ	79.36	60.80	48.59	31.14	78.07	59.87	54.71	36.74	87.93	68.97	56.02	35.20	82.45	57.62	32.18	18.69	70.07	54.00	30.75	15.20
SampleAttn	78.25	61.14	48.31	31.73	77.17	60.88	56.64	36.17	87.84	72.46	59.57	40.72	85.72	66.44	59.10	45.98	76.20	70.35	53.91	30.42
QUOKA	**86.71**	**80.19**	**70.90**	**57.01**	**87.85**	**74.27**	**66.82**	**59.37**	**93.73**	**91.07**	**88.57**	**74.83**	**89.97**	**79.94**	**72.69**	**61.37**	**78.92**	**79.19**	**73.40**	**57.79**

Table 3: **LongBench results (Higher is better)**. For each model, the three columns correspond to selective budgets $B_{\text{SA}} \in \{512, 1024, 2048\}$.

	Llama3.2-3B-Instruct			*Qwen2.5-3B-Instruct*			*Qwen3-4B*			*Smollm3*		
Budget	512	1024	2048	512	1024	2048	512	1024	2048	512	1024	2048
LessIsMore	0.703	0.788	0.850	0.461	0.556	0.659	0.665	0.773	0.868	0.765	0.842	0.918
SparQ	0.721	0.802	0.842	0.636	0.726	0.808	0.686	0.782	0.872	0.725	0.805	0.922
Loki	0.686	0.757	0.842	0.589	0.671	0.787	0.622	0.782	0.872	0.384	0.801	0.622
SampleAttn	0.738	0.800	0.901	0.660	0.756	0.828	0.755	0.875	0.947	0.856	0.929	0.966
QUOKA	**0.945**	**0.972**	**0.986**	**0.869**	**0.945**	**0.977**	**0.966**	**0.992**	**0.995**	**0.998**	**1.03**	**1.028**

- **Latency Experiments: Up to 3× end to end speedup**



(a) Attention latency on NVIDIA A100 GPU (b) TTFT on NVIDIA A100 GPU

Conclusion & Next Steps

- QuoKA shows that **query-oriented KV selection** enables efficient, training-free sparse attention for long-context prefill.
- Achieves large latency reductions while preserving accuracy across architectures and model families (Cosine similarity ensures bounded, scale-invariance).
- Results include ***3× lower TTFT, 5× GPU attention speedup, and up to 7× CPU speedup, while using 88% fewer KV pairs and maintaining near-baseline accuracy***
- Future work includes adaptive query selection, extensions beyond prefill, and integration with complementary efficiency techniques.

*Qualcomm AI Research is a division of Qualcomm Technologies, Inc

** All latency numbers include end-to-end prefill and exclude kernel warm-up
Batch size, sequence length, and sparsity held constant across methods..

Thank you

Follow us on:

For more information, visit us at qualcomm.com & qualcomm.com/blog