Microsoft UNIVERSITY OF SURREY

NEURAL INFORMATION PROCESSING SYSTEMS

[NeurIPS'24 Spotlight]
Fri 13 Dec 4:30-7:30 pm PST

MInference 1.0: Accelerating Long-Context LLMs Inference via **Dynamic Sparse Attention**

Huiqiang Jiang[†], Yucheng Li[◊†], Chengruidong Zhang[†], Qianhui Wu, Xufang Luo, Surin Ahn, Zhenhua Han, Amir H. Abdi, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, Lili Qiu

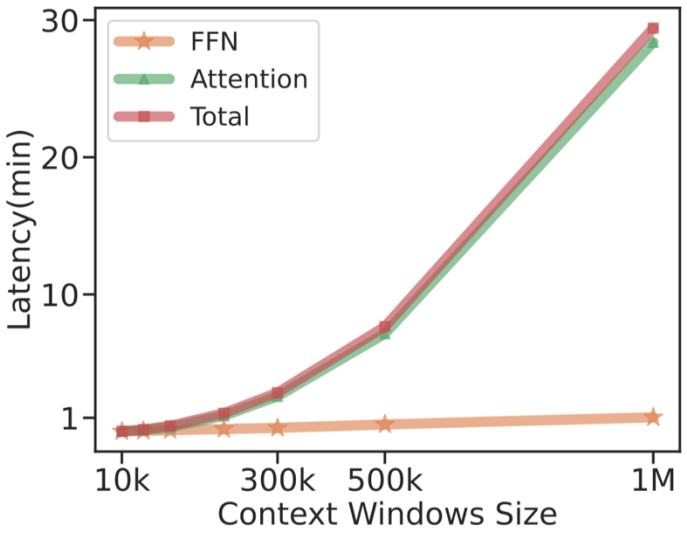
Microsoft Corporation, [◊]University of Surrey

https://aka.ms/MInference

Microsoft Research

Long-context LLMs Inference Bottleneck

- **Long Prefilling Latency,** 30 minutes to process 1M tokens on an A100 for an 8B LLM.
- **KV Cache Storage Issue,** Storing 512K tokens requires 62GB of GPU memory in fp16.



(a) Attention incurs heavy cost.

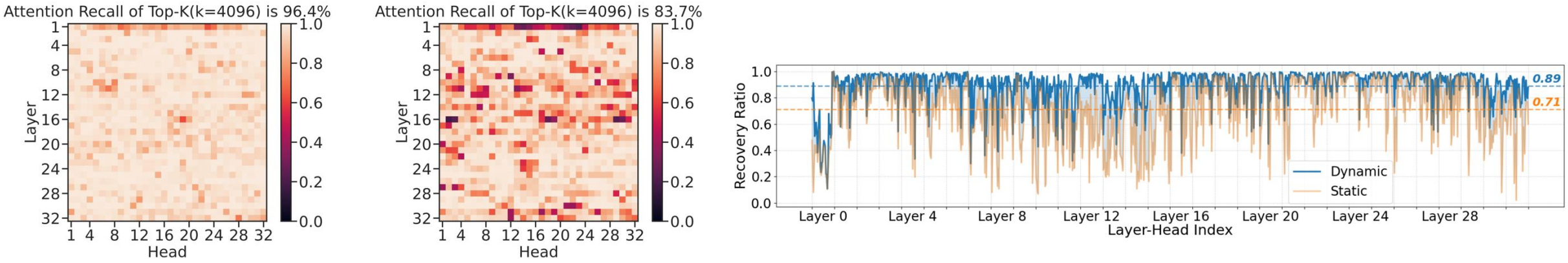
Prompt Length	128K	256K	512K	1M
Total Latency (s)	32.8	111	465	1,765
FFN (s)	7.6	15	31	70
Attention (s)	25.2	96	434	1,695
GPU Memory KV Cache (GB)	15.6	31.2	62.5	125

Table 1: Decoding latency and memory required for KV cache of Llama-3-8B across different context lengths on one A100 GPU.

$$\text{OTTFT} = \frac{\text{FLOPs required for 1M prompt}}{\text{A100's fp16 TFLOPs}} = \frac{1M \times (2 \times 8B + 2 \times 32 \text{ layers} \times 1M \times 4096 \text{ dim} + \text{softmax portion})}{312 \text{ TFLOPs}} = 14.52 \text{ mins}$$

*1.5 = 21.78 mins = **60+** A100 * 20s TTFT

Observation 1: Attention is Dynamically Sparse



(a) Attention is sparse. (b) Sparsity of attention is dynamic.

(c) Dynamically sparsity in decoding.

in prefilling

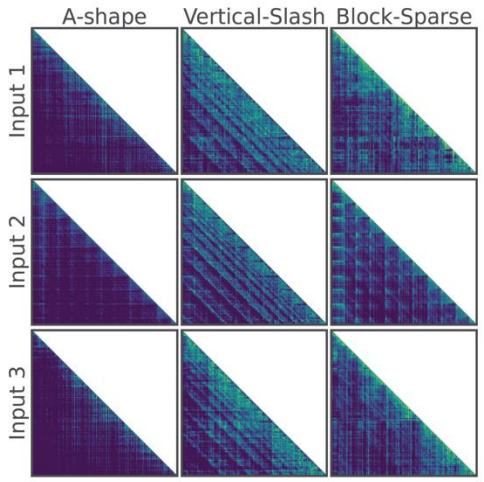
Figure. The dynamic sparsity in attention. (a) How much attention scores can top-k (k=4096) columns cover in a 128k context. (b) Less attention scores are retrieved when reusing the top-k indices from another examples, indicating its dynamic nature. Visualizations are based on LLaMa-3-8B with a single A100.

in decoding

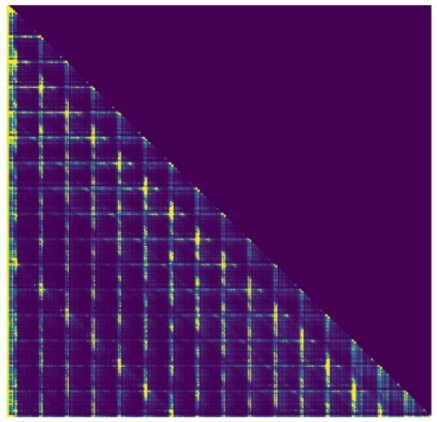
Figure. (c) The dynamic sparsity of each layer and head in Llama-3-8B model in the KV retrieval test of 100k tokens. The blue curve shows that dynamically selecting top-1000 critical tokens achieves an 89% average recovery ratio, indicating high attention sparsity. In contrast, the orange curve shows that statically using the initial top-1000 critical tokens drops the ratio to 71%.

Observation 2: Attention Sparsity Exhibits Patterns

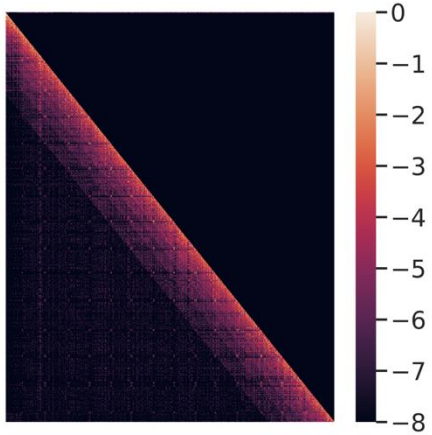
- After pretraining, attention **exhibits various sparse patterns**, including A-shape, Vertical-Slash, and block-sparse patterns.
- These sparse **patterns are fixed** for each head across different inputs.
- The specific **sparse elements** (e.g., column index, slash index) **dynamically change** depending on the context.



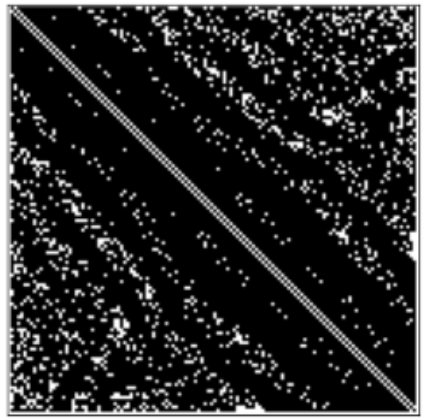
Text LLMs



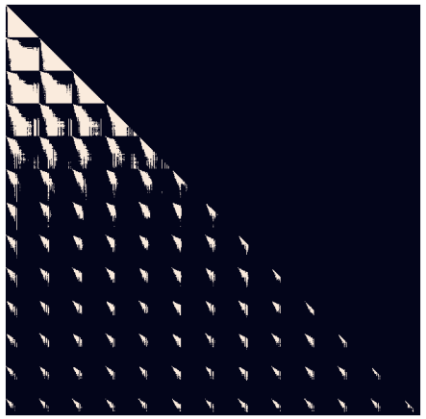
Multi-modal LLMs
(image, video)



Encoder-Decoder



Decoder-only



SSM or Hybrid LLMs

MInference 1.0

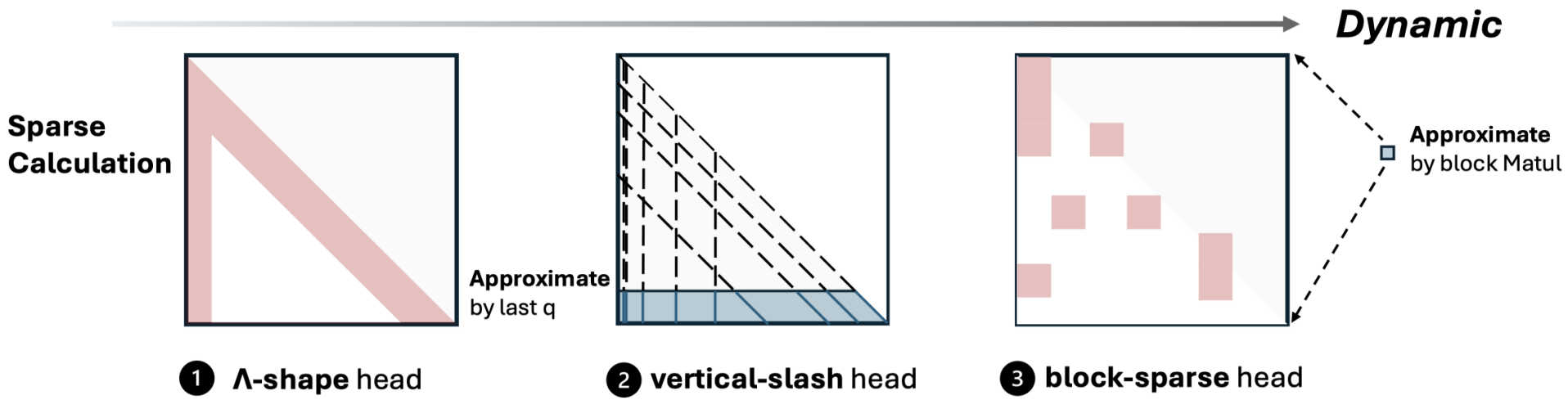


Figure 4: The three sparse methods in MInference.

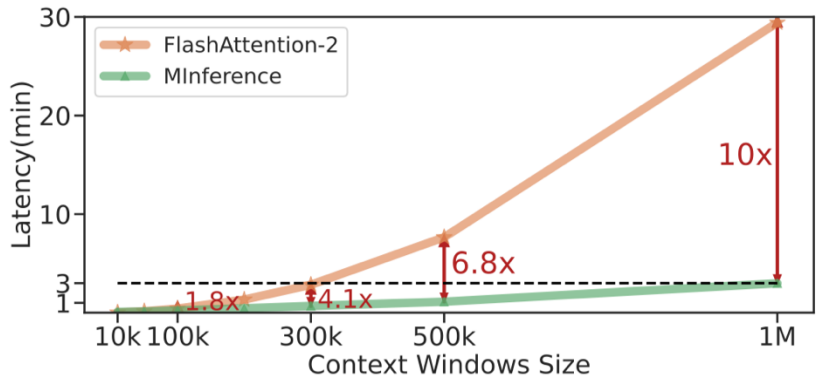
Table 1: Comparison of different sparse patterns.

Patterns	A-shape	Vertical-Slash	Block-Sparse	Top-K
Spatial Distribution	Static structured	Dynamic structured	Dynamic structured	Dynamic fine-grained
Latency on GPU	Low	Medium	Low	High
Time to build the index	Zero	Small	Small	High

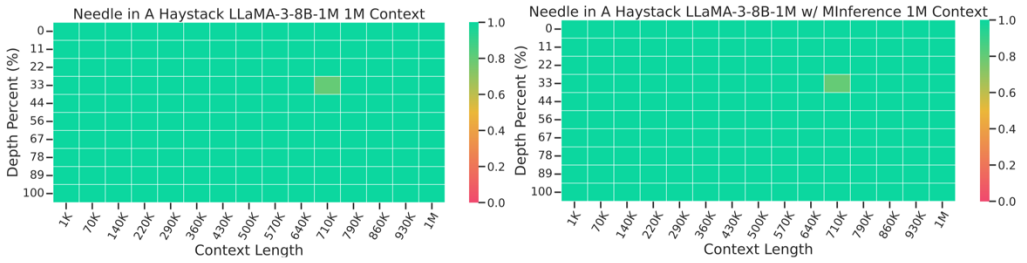
How effective and efficient are MInference?

Table 3: Performance (%) of different models and different methods on RULER [HSK+24] evaluated at lengths from 4k to 128k.

Methods	Claimed	Effective	4K	8K	16K	32K	64K	128K	Avg.
LLaMA-3-8B-262K	262K	16K	97.2	91.8	87.3	80.8	77.4	72.2	84.4
StreamingLLM	-	4K	97.2	38.1	37.5	17.2	14.2	9.4	35.0
StreamingLLM w/ dilated	-	<4K	23.4	0.7	1.4	18.8	16.5	15.6	12.7
StreamingLLM w/ strided	-	<4K	2.0	0.7	0.6	0.6	0.7	1.3	1.0
InfLLM	-	4K	89.4	79.8	70.1	55.6	43.0	39.5	62.9
Ours	-	**32K**	**97.7**	**91.2**	**88.5**	**85.0**	**82.3**	**77.6**	**87.0**
Yi-9B-200K	200K	8K	91.9	90.2	78.8	76.3	68.1	62.9	78.1
StreamingLLM	-	4K	91.9	37.8	33.9	18.6	13.0	12.8	34.3
StreamingLLM w/ dilated	-	<4K	44.8	42.8	38.5	29.8	26.8	23.9	34.4
StreamingLLM w/ strided	-	<4K	2.6	0.7	0.6	0.6	1.2	0.5	1.1
InfLLM	-	<4K	80.3	83.9	60.7	45.2	38.6	30.2	56.5
Ours	-	8K	**92.3**	**89.7**	**79.0**	**73.8**	**64.7**	**56.9**	**74.7**
GLM-4-9B-1M	1M	64K	93.8	91.6	89.3	87.4	85.2	80.8	88.0
StreamingLLM	-	4K	93.8	66.9	58.5	51.4	45.9	39.1	59.3
InfLLM	-	8K	**94.7**	89.5	76.4	66.5	56.8	53.5	72.9
Ours	-	64K	94.6	**93.1**	**91.0**	**89.6**	**85.5**	**84.0**	**89.6**

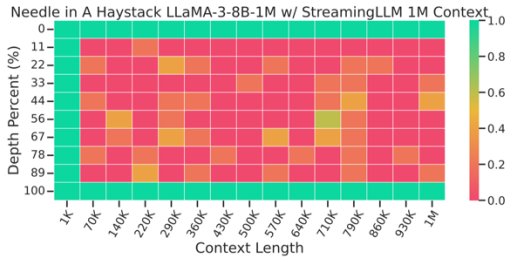


(b) Latency Speedup

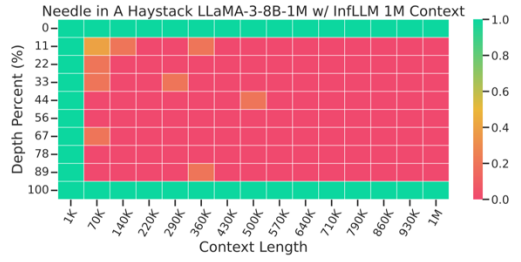


(a) LLaMA-3-8B

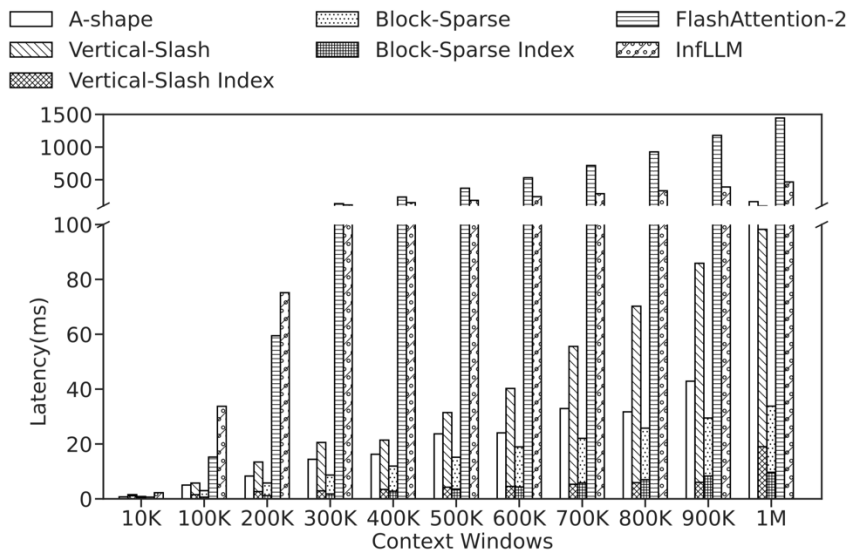
(b) LLaMA-3-8B w/ MInference



(c) LLaMA-3-8B w/ StreamingLLM



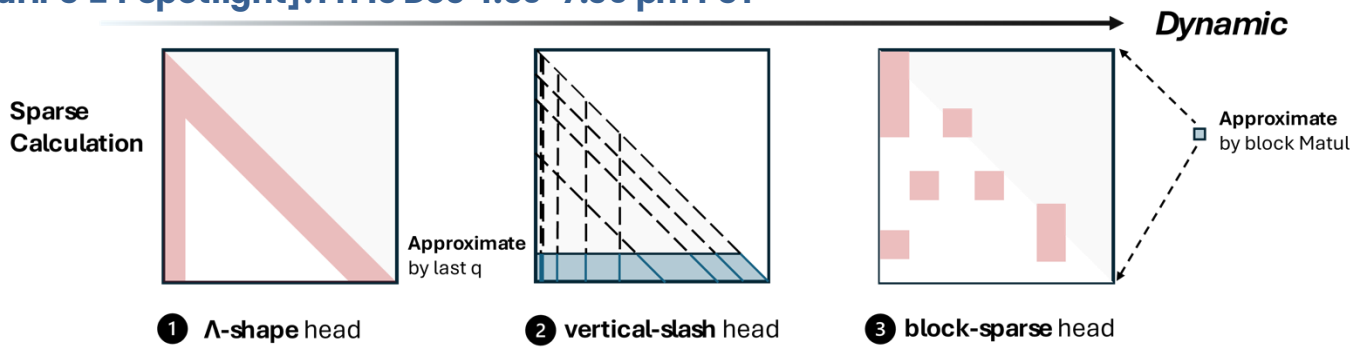
(d) LLaMA-3-8B w/ InfLLM

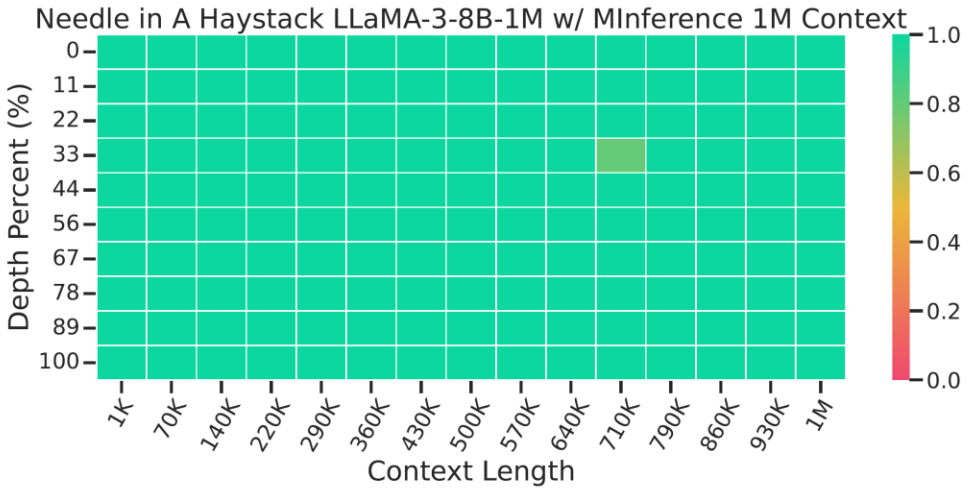


Minference 1.0: Accelerating Pre-filling for Long-Context LLMs via Dynamic Sparse Attention

[NeurIPS'24 Spotlight]: Fri 13 Dec 4:30-7:30 pm PST

Project Page **Code**



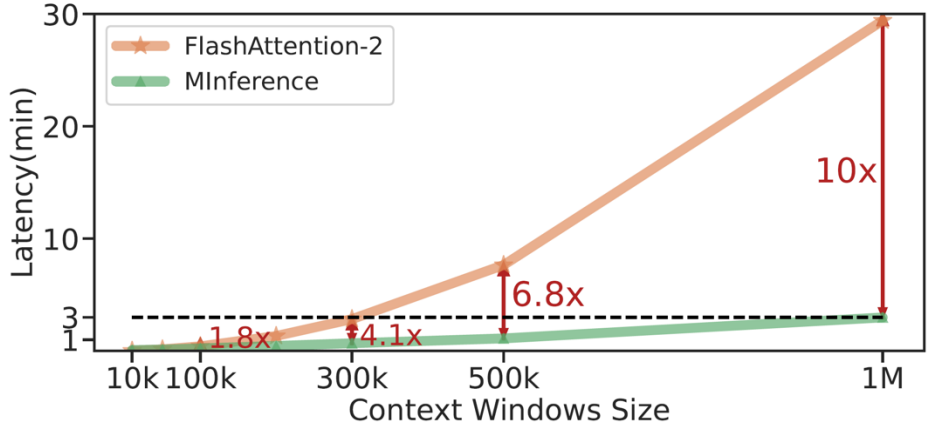


(a) Needle In A Haystack

The main ideas involve **dynamically calculating sparse attention** in long-context LLMs, as shown in Fig(b), which significantly speeds up inference by up to ***10x*** on an A100 while maintaining accuracy across different context windows in Fig(a). The motivation is as follows:

- *Attention, especially in long-context scenarios, is sparse and dynamic.*
- *This dynamic sparsity presents spatial aggregation patterns that are fixed for all inputs.*

Based on this, we propose ***MInference 1.0***, which uses **minimal overhead** to **approximate** the dynamic sparse index and **calculate** the dynamic sparse attention using the optimal GPU kernels, thereby accelerating long-context LLM inference. You can find more details in the paper.



(b) Latency Speedup

https://aka.ms/MInference

🔍 **MInference: Approximate and Calculate the Dynamic Sparse Attention of Long-context LLMs.**