# Native Sparse Attention: Hardware-Aligned and Natively Trainable Sparse Attention

Xing Ma

2025.3.20

# Outline

- Paper introduction
- Background
  - Fixed Sparse Pattern
  - Dynamic Token Pruning
  - Query-Aware Selection
- Implementation
- Evaluation
- Summary

# Paper introduction

## Native Sparse Attention: Hardware-Aligned and Natively Trainable Sparse Attention

Jingyang Yuan[*1,2], Huazuo Gao[1], Damai Dai[1], Junyu Luo[2], Liang Zhao[1], Zhengyan Zhang[1], Zhenda Xie[1], Y. X. Wei[1], Lean Wang[1], Zhiping Xiao[3], Yuqing Wang[1], Chong Ruan[1], Ming Zhang[2], Wenfeng Liang[1], Wangding Zeng[1]

[1]**DeepSeek-AI**
[2]**Key Laboratory for Multimedia Information Processing, Peking University, PKU-Anker LLM Lab**
[3]**University of Washington**
{yuanjy, mzhang_cs}@pku.edu.cn, {zengwangding, wenfeng.liang}@deepseek.com

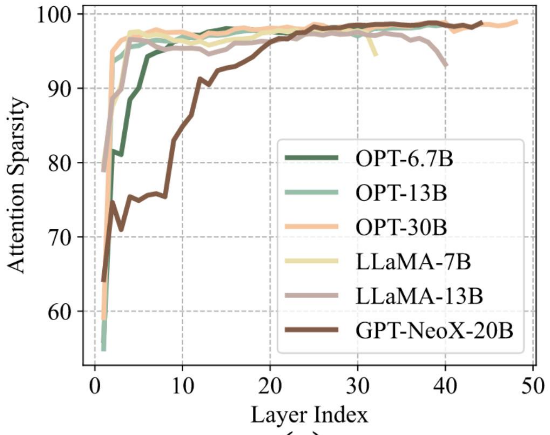
# Background

- Attention算子：$\mathrm{Attn}(\boldsymbol{q}, \boldsymbol{K}, \boldsymbol{V}) = \mathrm{Softmax}(\boldsymbol{q}\boldsymbol{K}^{\top})\boldsymbol{V},$

- Motivation：decode阶段计算量和现存随seq长度线性增加, 在64k上下文中attention算子占用70-80% latency

- Insight：LLM attention计算中存在的稀疏性
  - 右上图稀疏性定义: 阈值设置为每行最大值的 1%（H2O NIPS23）
  - 右下图是不同模型中score矩阵的热力图（SpargeAttn）

- 目标：识别出crucial tokens，attention计算中query只与这些crucial token的K, V vector发生计算，从而在保证计算结果的前提下减少访存量和计算量



(a)

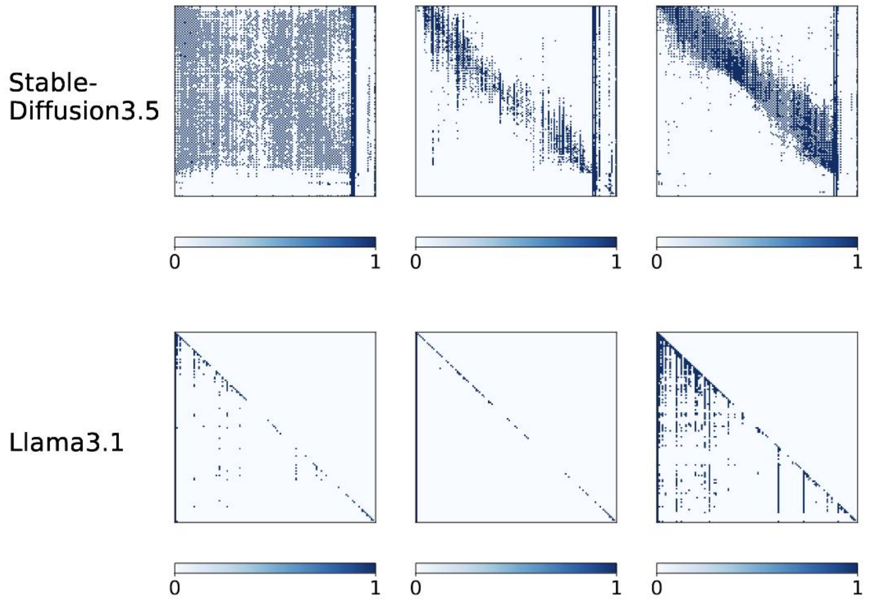


*Figure 2.* Some sampled patterns of attention map $P$ in video, image, and language generation models.

# Background

- 以往工作分类
    - Fixed Sparse Pattern，代表工作: streamingLLM (ICLR24)
    - Dynamic Token Pruning，代表工作: H2O (NIPS23)，SnapKV (NIPS24)
    - Query-Aware Selection，代表工作: Quest (ICML24)，InfLLM (NIPS24)

# Fixed Sparse Pattern

- Observation pattern

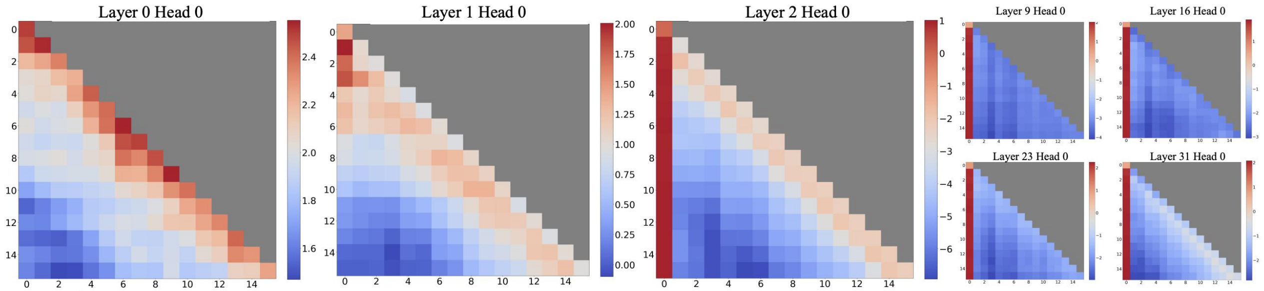
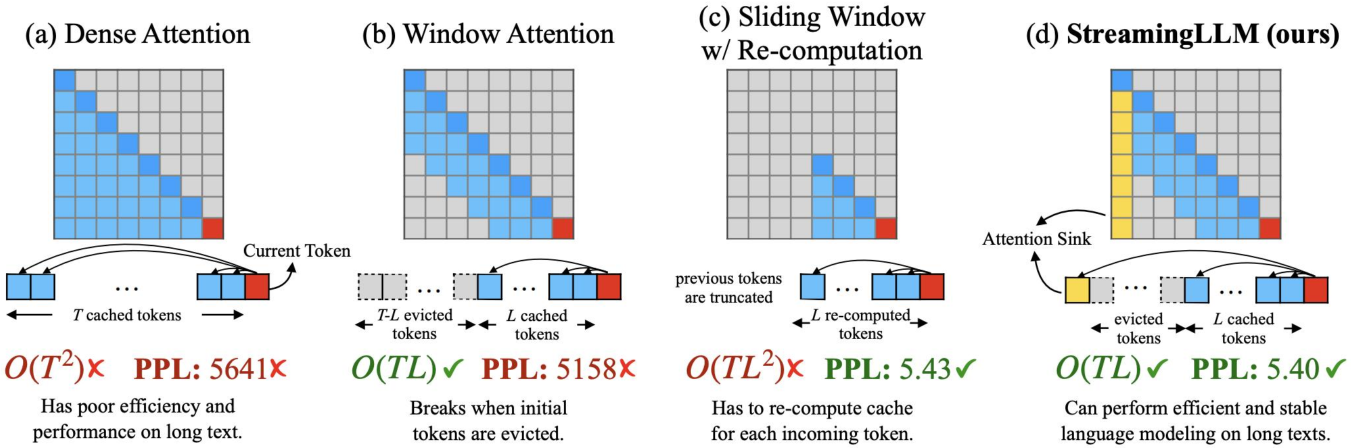


Figure 2: Visualization of the *average* attention logits in Llama-2-7B over 256 sentences, each with a length of 16. Observations include: (1) The attention maps in the first two layers (layers 0 and 1) exhibit the "local" pattern, with recent tokens receiving more attention. (2) Beyond the bottom two layers, the model heavily attends to the initial token across all layers and heads.

- Fixed mask: attention sink + local window



Has poor efficiency and performance on long text.

Breaks when initial tokens are evicted.

Has to re-compute cache for each incoming token.

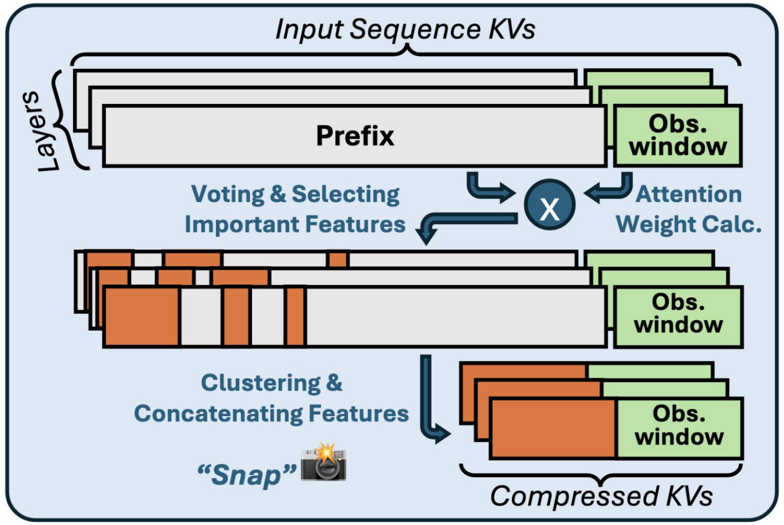
Can perform efficient and stable language modeling on long texts.

- Limitation: highly task-specific

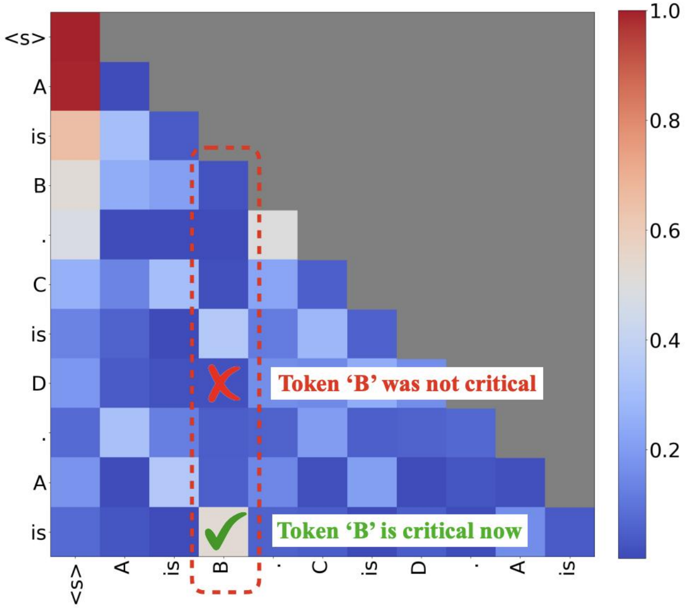
# Dynamic Token Pruning

- Token pruning，已经被剪枝的token的kv cache，不会再被后续decode阶段使用

- Observation：input sequence的最后一段称之为observation window，由这个observation window识别出来的crucial tokens在后续的generation（decode）阶段依然成立



snapkv



反例（Quest）

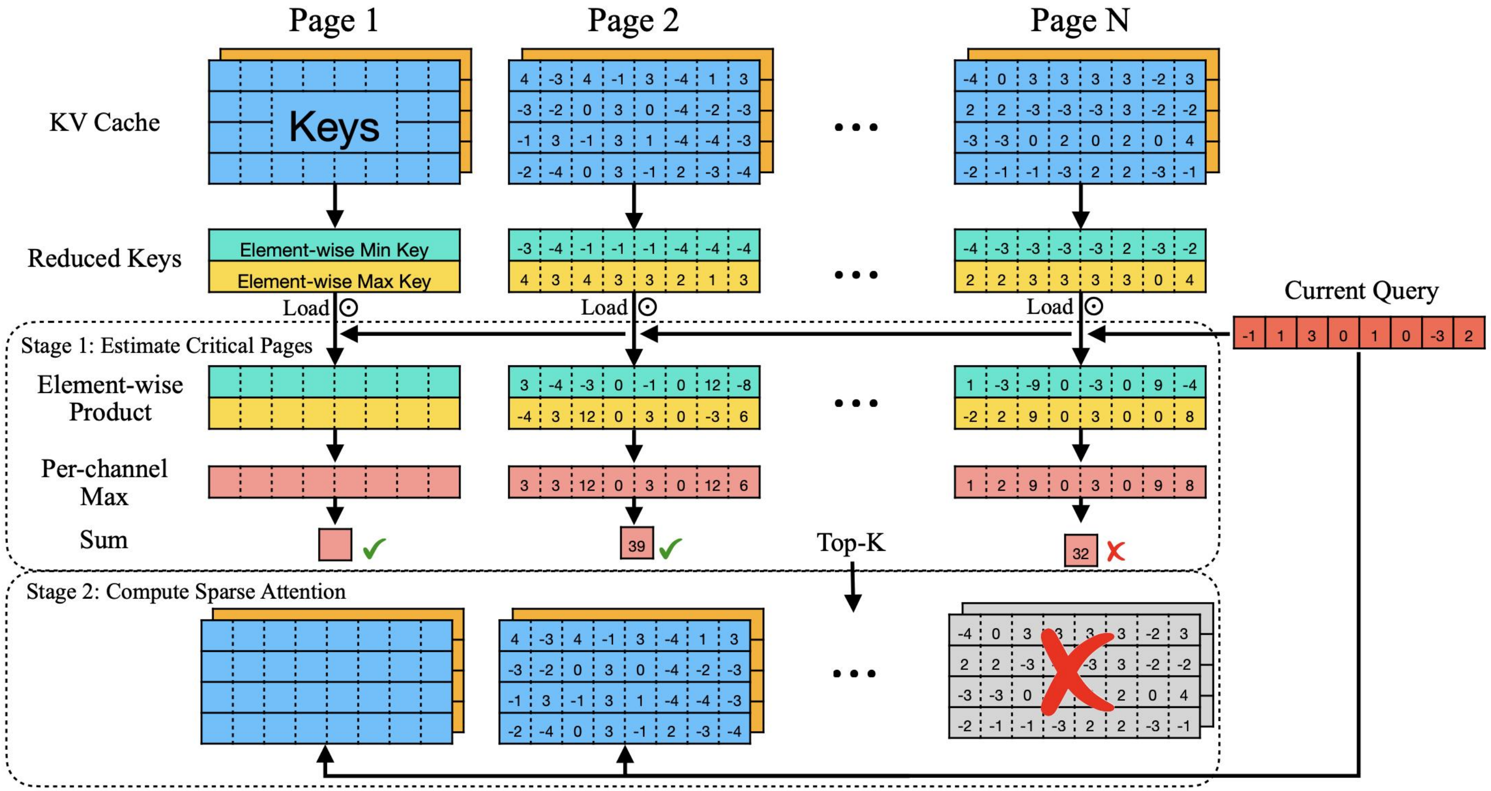
- Limitation: 只适用于长prefill, 短decode的场景，比如总结，检索任务，长decode场景下crucial tokens可能会改变

# Query-Aware Selection

- **保存所有kv cache**，在decode阶段**query-aware** 召回 **block-wise**的kv cache
- 在召回方法和kv cache管理这两个方面引申出不同方法
- 召回方法
  - Estimated upper bound（Quest）
  - Representative vector (InfLLM)
  - ANNS (Retrievalattention)

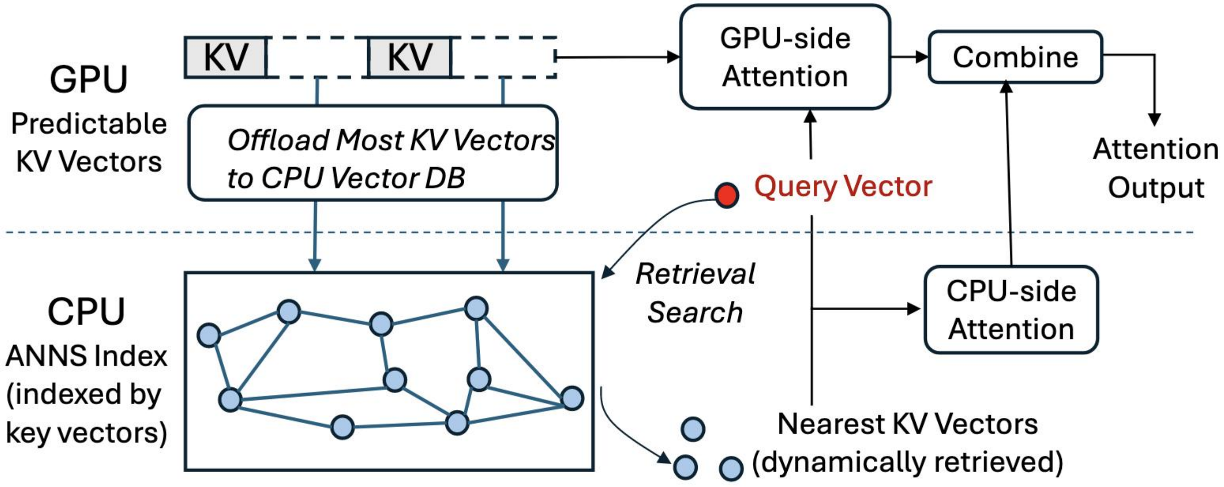
Estimated upper bound （Quest）

1. Per channel max and min
2. Element-wise product
3. Per-channel max



# Query-Aware Selection (cont.)

- kv cache管理
  - 由于对于一段query来说，不同block中的kv vector具有不同的重要性，所以可以构建hierarchical kv cache
  - hierarchical kv cache



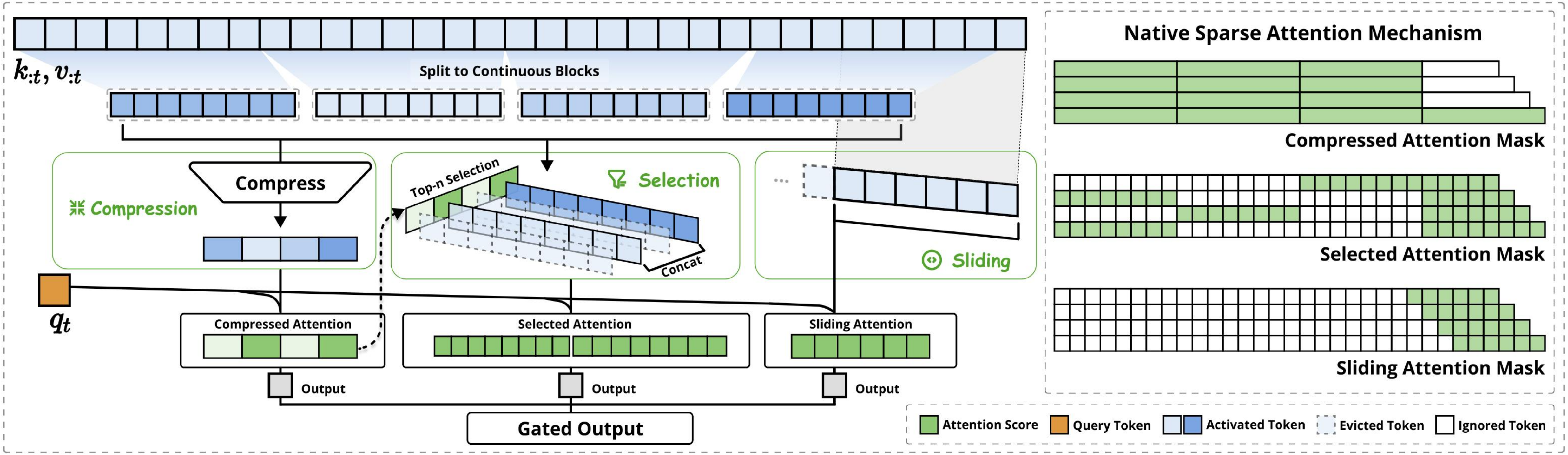
(a) Overall design of RetrievalAttention.

Retrievalattention将 cold k vector offload到cpu上，gpu端计算attention的时候，cpu端同时执行retrieval 和attention计算，最后将结果整合

# NSA Overview

- Two innovation：
  - (1)通过训练获取识别稀疏的能力
  - (2) 硬件友好的算法, 把稀疏转化成真实加速比

- Component: three attention branch and gate for output
  - (1) Compressed attention (use mlp)  (2) Selected attention  (3) Sliding attention

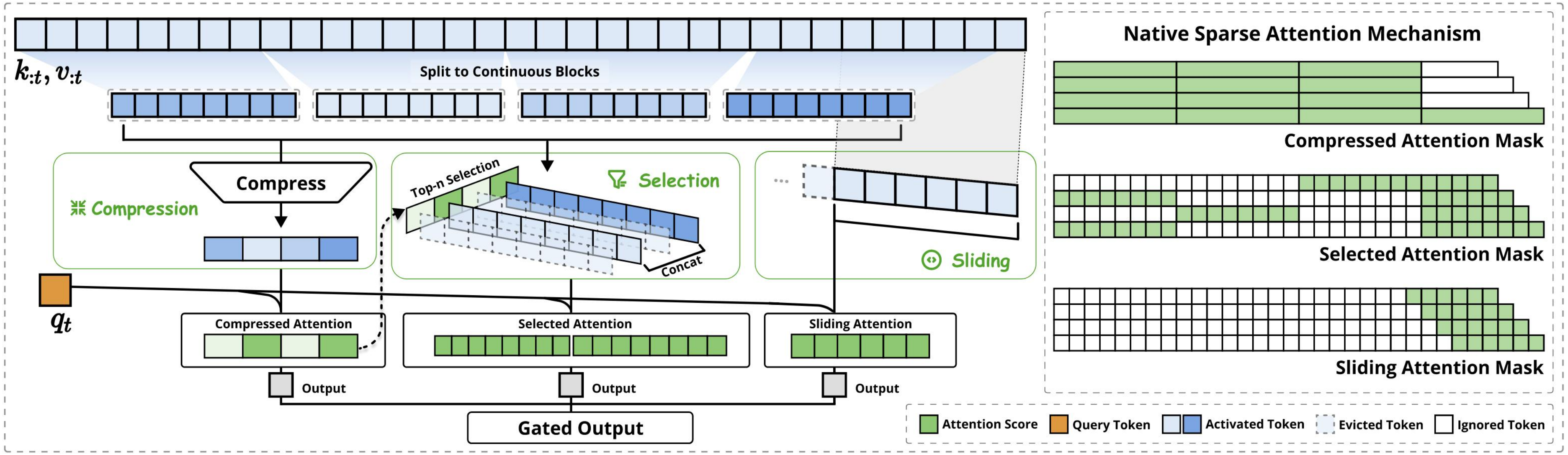


# Rethinking Sparse Attention Methods

- 以往sparse attention加速推理的局限：
  - (1) Phase-Restricted Sparsity: 只有prefill或者decode其中一个阶段能用上sparse
    - 比如snapkv这篇工作，prefill阶段和full attention一样，根据prefill完的attention score对kv cache进行剪枝，所以只有decode阶段可以用上稀疏加速
  - (2) Incompatibility with Advanced Attention Architecture
    - 比如gqa 机制，之前的方法为每个head选择各自的稀疏kv，所以每个group中的kv加载量是每个head的并集

- 这篇工作(Training-based methods)的优势
  - 有效加速training阶段
  - 消除training和inference之间的架构偏差

# Methodology



- Compression: $$\tilde{K}_t^{\text{cmp}} = f_K^{\text{cmp}}(\mathbf{k}_{:t}) = \left\{ \varphi(\mathbf{k}_{id+1:id+l}) \middle| 1 \leqslant i \leqslant \left\lfloor \frac{t-l}{d} \right\rfloor \right\}$$
  - φ is a learnable MLP with intra-block position encoding
- Blockwise Selection: $$\mathbf{p}_t^{\text{cmp}} = \text{Softmax}\left(\mathbf{q}_t^T \tilde{K}_t^{\text{cmp}}\right),$$

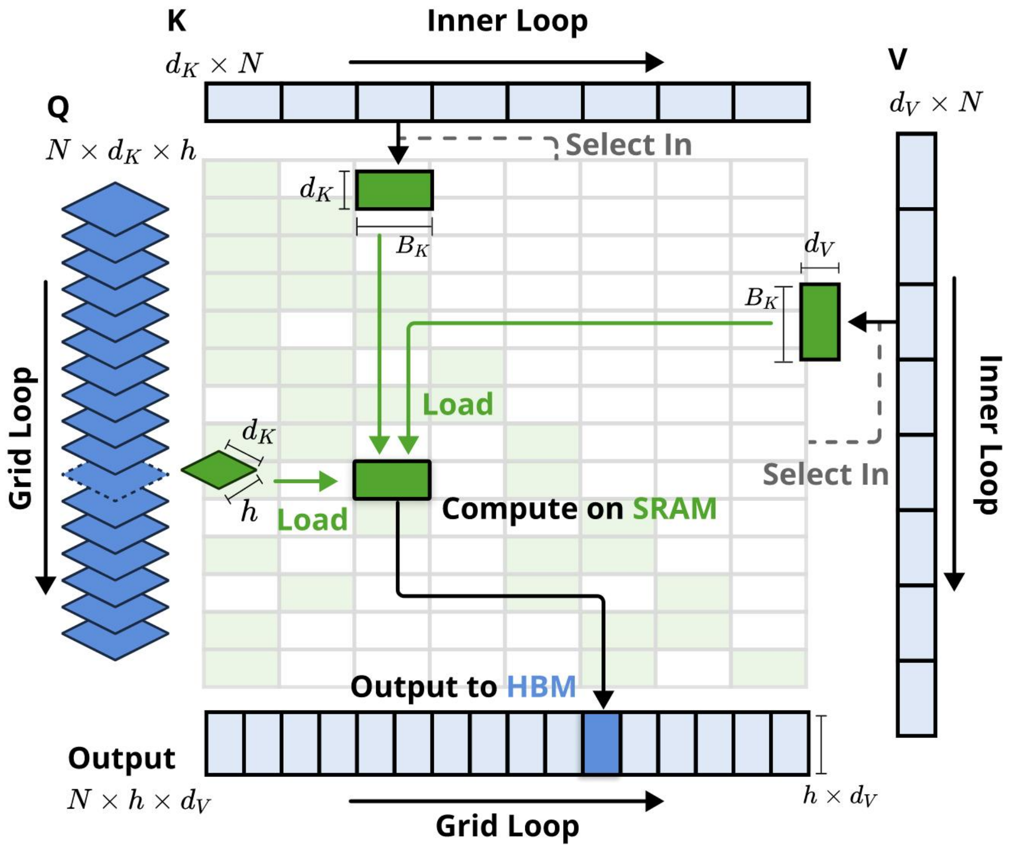
$$\mathbf{p}_t^{\text{slc}} = \mathbf{p}_t^{\text{cmp}}$$

$$\mathcal{I}_t = \{i \mid \text{rank}(\mathbf{p}_t^{\text{slc}\prime}[i]) \leqslant n\}$$

$$\tilde{K}_t^{\text{slc}} = \text{Cat}\left[\{\mathbf{k}_{il'+1:(i+1)l'} | i \in \mathcal{I}_t\}\right],$$

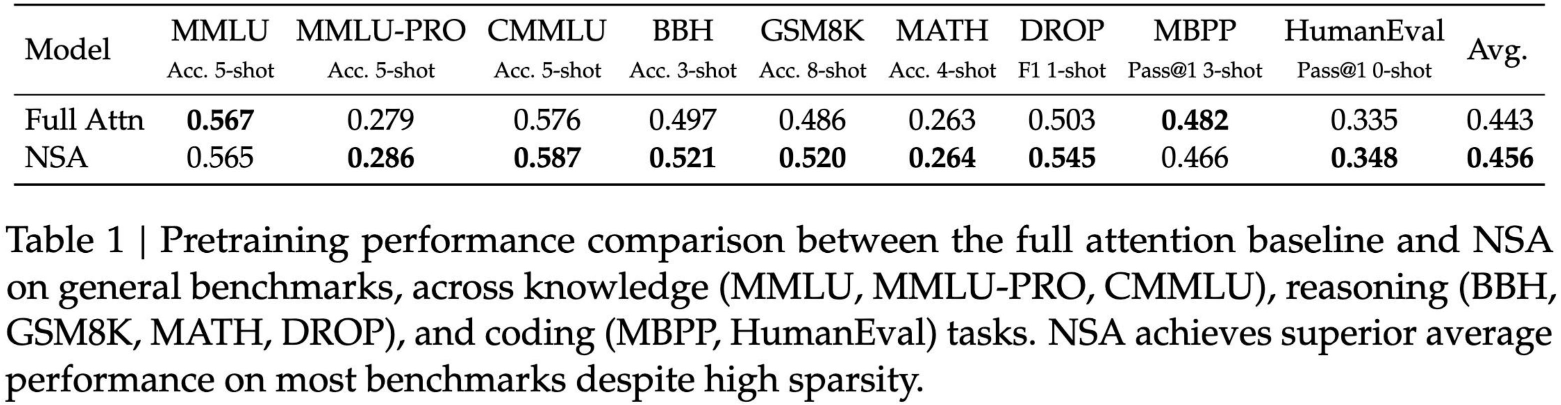
# Kernel design

- How to hardware-aligned
- 使用triton实现，一个thread block做一个token的一个query group计算
    - 消除了冗余的kv cache加载
    - 平衡了sm的负载



# Evaluation

- 在general benchmark上得分超过full attention
  - 原因：通过过滤噪声来提升性能

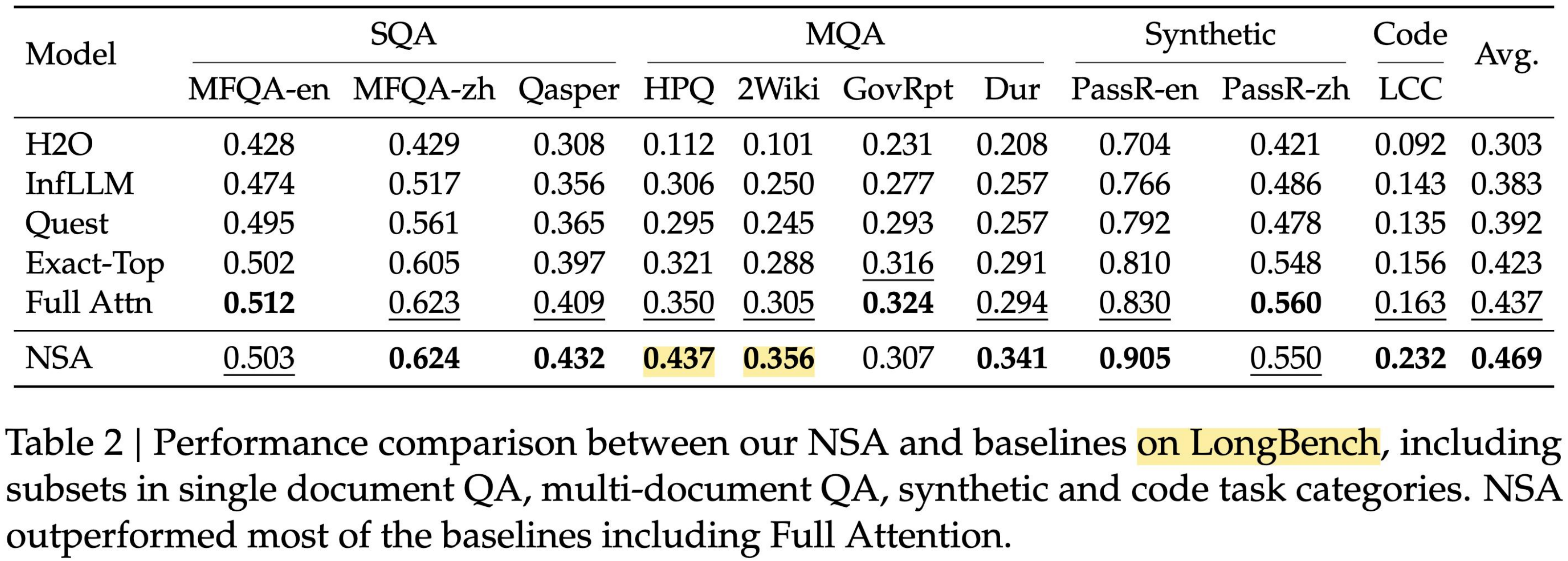


| Model | MMLU<br>Acc. 5-shot | MMLU-PRO<br>Acc. 5-shot | CMMLU<br>Acc. 5-shot | BBH<br>Acc. 3-shot | GSM8K<br>Acc. 8-shot | MATH<br>Acc. 4-shot | DROP<br>F1 1-shot | MBPP<br>Pass@1 3-shot | HumanEval<br>Pass@1 0-shot | Avg. |
|---|---|---|---|---|---|---|---|---|---|---|
| Full Attn | **0.567** | 0.279 | 0.576 | 0.497 | 0.486 | 0.263 | 0.503 | **0.482** | 0.335 | 0.443 |
| NSA | 0.565 | **0.286** | **0.587** | **0.521** | **0.520** | **0.264** | **0.545** | 0.466 | **0.348** | **0.456** |

Table 1 | Pretraining performance comparison between the full attention baseline and NSA on general benchmarks, across knowledge (MMLU, MMLU-PRO, CMMLU), reasoning (BBH, GSM8K, MATH, DROP), and coding (MBPP, HumanEval) tasks. NSA achieves superior average performance on most benchmarks despite high sparsity.

# Evaluation (cont.)

- 在LongBench(long context benchmark)上超过所有sparse attention和full attention



| Model | SQA | | | MQA | | | | Synthetic | | Code | Avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | MFQA-en | MFQA-zh | Qasper | HPQ | 2Wiki | GovRpt | Dur | PassR-en | PassR-zh | LCC | |
| H2O | 0.428 | 0.429 | 0.308 | 0.112 | 0.101 | 0.231 | 0.208 | 0.704 | 0.421 | 0.092 | 0.303 |
| InfLLM | 0.474 | 0.517 | 0.356 | 0.306 | 0.250 | 0.277 | 0.257 | 0.766 | 0.486 | 0.143 | 0.383 |
| Quest | 0.495 | 0.561 | 0.365 | 0.295 | 0.245 | 0.293 | 0.257 | 0.792 | 0.478 | 0.135 | 0.392 |
| Exact-Top | 0.502 | 0.605 | 0.397 | 0.321 | 0.288 | 0.316 | 0.291 | 0.810 | 0.548 | 0.156 | 0.423 |
| Full Attn | **0.512** | 0.623 | 0.409 | 0.350 | 0.305 | **0.324** | 0.294 | 0.830 | **0.560** | 0.163 | 0.437 |
| NSA | 0.503 | **0.624** | **0.432** | **0.437** | **0.356** | 0.307 | **0.341** | **0.905** | 0.550 | **0.232** | **0.469** |

Table 2 | Performance comparison between our NSA and baselines on LongBench, including subsets in single document QA, multi-document QA, synthetic and code task categories. NSA outperformed most of the baselines including Full Attention.

# Evaluation (cont.)

- stable training convergence and real speedup

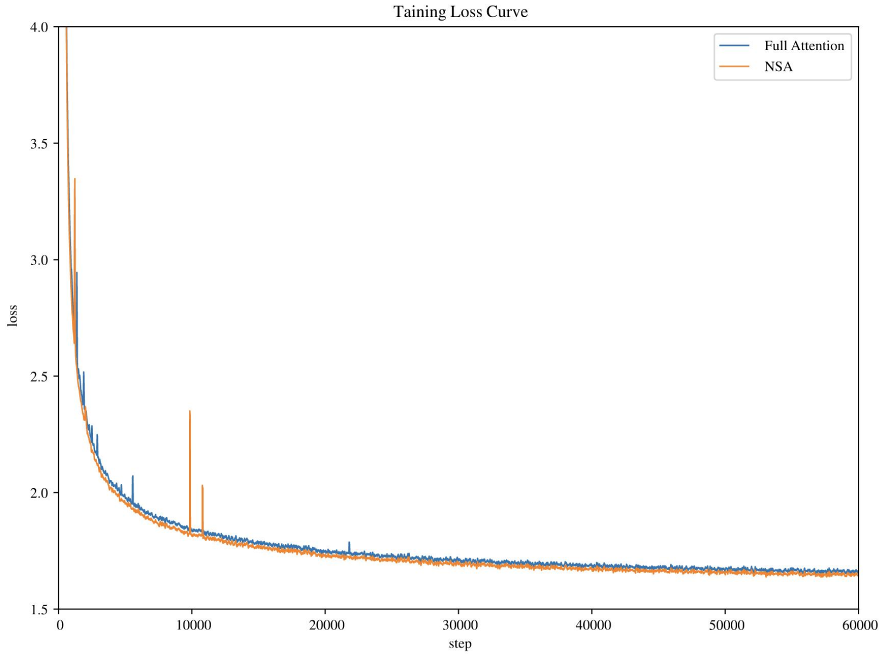
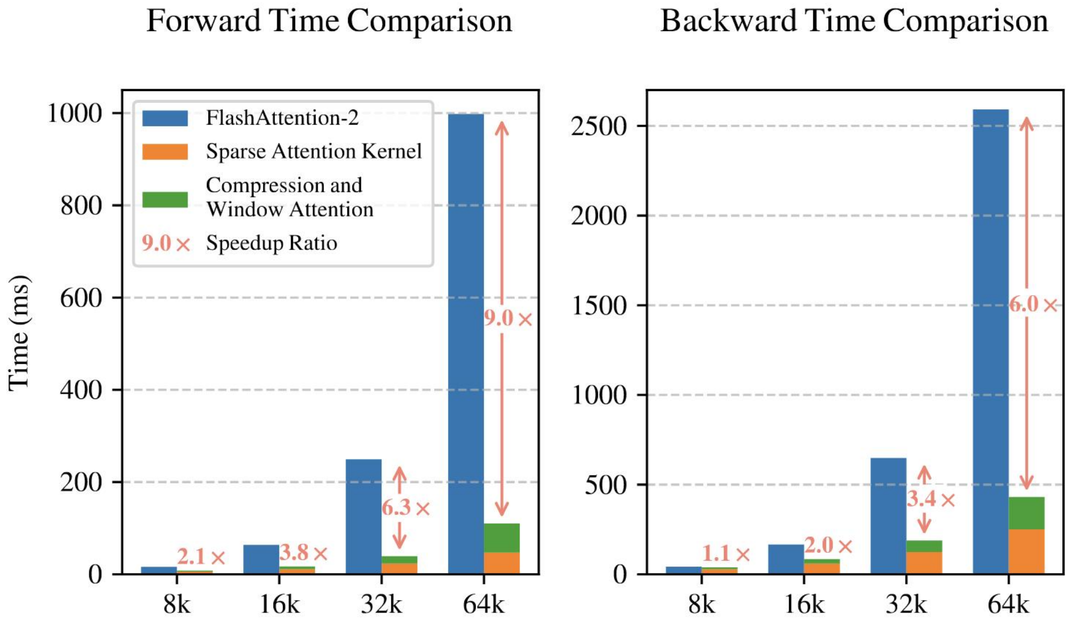


Figure 4 | Pretraining loss comparison between Full Attention and our NSA on 27B-parameter model. Both models exhibit stable convergence, with NSA achieving lower loss values.


Forward Time Comparison
Backward Time Comparison
FlashAttention-2
Sparse Attention Kernel
Compression and Window Attention
9.0 × Speedup Ratio
Time (ms)
2.1 ×
3.8 ×
6.3 ×
9.0 ×
1.1 ×
2.0 ×
3.4 ×
6.0 ×
8k
16k
32k
64k
8k
16k
32k
64k