Token Sparse Attention: Efficient Long-Context Inference with Interleaved Token Selection

Dongwon Jo, Beomseok Kang, Jiwon Song, Jae-Joon Kim

The Forty-Third International Conference on Machine Learning (ICML), 2026.

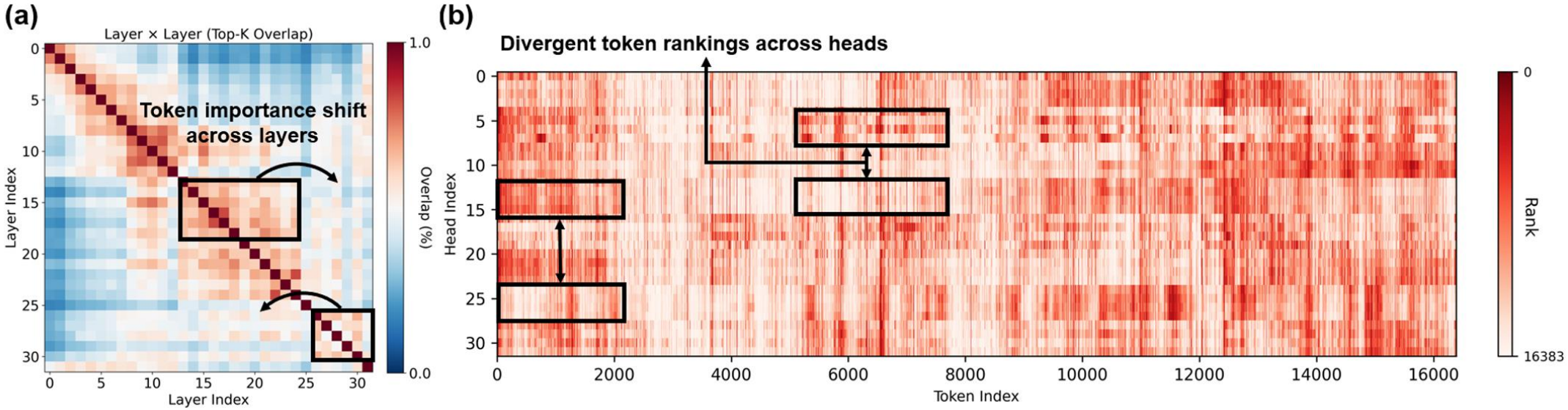
VLSI LAB

Paper: https://arxiv.org/abs/2602.03216 **Code**: https://github.com/dongwonjo/Token-Sparse-Attention

Demo: https://dongwonjo.github.io/portfolio/token_sparse_attention/

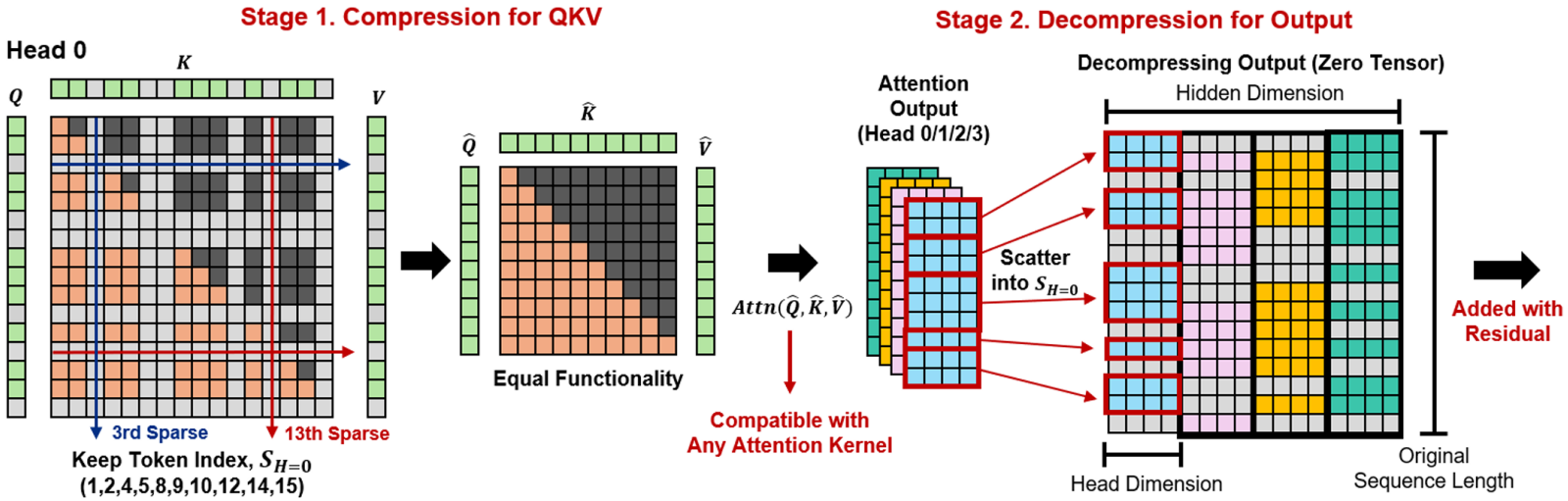
Motivation

Why Another Attention Acceleration Method?



- Long-context prefill cost grows quadratically with sequence length
- Existing sparse attention methods operate at block granularity → Irrelevant tokens may survive inside selected blocks
- Existing token eviction methods permanently remove tokens → Assume token importance remains stable across layers
- In reality:
  → Token importance shifts across layers **(Figure a.)**
  → Different attention heads focus on different tokens **(Figure b.)**

Token Sparse Attention



Algorithm 1 Dynamic Token Coverage and Index Search

Require: Q, K, Token Coverage τ, Sequence Length L
Ensure: Head-wise selected token indices $\{S_h\}_{h=1}^{H}$

// Compute attention scores using recent queries
$\hat{A} \leftarrow \text{softmax}\left(Q_{[-last\_q:]}K^{\top}/\sqrt{d}\right)$
// Compute token-level scores per head
$s_h \leftarrow \text{pooling}(\text{sum}_{\text{vertical}}(\hat{A}))$
// Aggregate and normalize token scores across heads
$s_l \leftarrow (\sum_{h=1}^{H} s_h)/(\sum_{t=1}^{L}\sum_{h=1}^{H} s_h[t])$
// Sort tokens by ascending importance
$I \leftarrow \text{argsort}(s_l)$
// Determine the number of sparse token
$k_{\text{sparse}} \leftarrow \arg\min_{k\in\{0,\ldots,L\}} \left\{\sum_{j=1}^{k} s_l[I[j]] \geq \tau\right\}$
$k_{\text{keep}} \leftarrow L - k_{\text{sparse}}$
// Head-wise top-k_{keep} token selection
$\{S_h\}_{h=1}^{H} \leftarrow \{\text{TopK}(s_h, k_{\text{keep}})\}_{h=1}^{H}$
return $\{S_h\}_{h=1}^{H}$

Compress and Decompress

- Each head independently selects important tokens and compresses Q, K, and V.
- Attention is performed on the resulting compact dense tensors.
- The attention output is scattered back to the original sequence layout.
- Unselected tokens are preserved through the residual connection.

Key Benefit: Every layer can re-evaluate token importance from the full context.

Dynamic Token Coverage

- Token importance is estimated using recent-query attention scores.
- We identify the least-important token tail using a coverage threshold τ.
- Each head then selects its own top-k tokens under the allocated budget.
- Sparsity adapts automatically to the attention score distribution.

Performance Highlights (1/2)

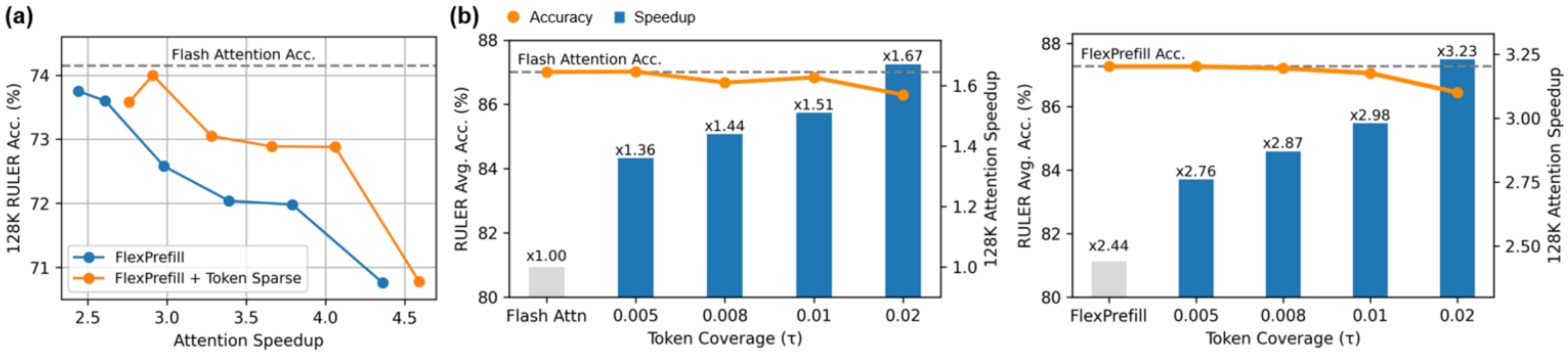
Complementary to Existing Sparse Attention

- Token Sparse Attention is designed to work alongside existing attention kernels.
- Accuracy remains nearly unchanged across FlashAttention, Minference, and FlexPrefill.
- Consistent attention speedup improvements are observed across all baselines.
- ×2.44 → ×2.76 speedup at the same 87.3% RULER accuracy.

Method	4K	8K	16K	32K	64K	128K	Avg.	Speedup
LLaMA-3.1-8B-Instruct								
Flash-Attention	95.82	92.77	91.02	84.87	83.43	74.15	87.01	×1.00
w/ Token Sparse	96.06	92.90	91.82	84.81	82.83	73.68	87.02	×1.36
Minference	93.46	92.29	91.01	85.34	83.19	73.63	86.49	×1.12
w/ Token Sparse	93.05	92.00	91.03	85.10	82.92	72.18	86.05	×1.38
FlexPrefill	95.48	92.71	91.40	87.20	83.05	73.75	87.27	×2.44
w/ Token Sparse	95.33	92.47	91.49	87.68	83.07	73.58	87.27	×2.76
Mistral-Nemo-12B-Instruct								
Flash-Attention	95.19	92.29	85.60	64.86	47.98	19.68	67.60	×1.00
w/ Token Sparse	95.07	92.10	85.97	63.67	47.97	19.44	67.37	×1.22
Minference	92.52	91.02	84.29	65.18	46.52	19.00	66.42	×1.13
w/ Token Sparse	93.01	91.36	84.31	64.70	46.67	18.68	66.46	×1.28
FlexPrefill	94.79	93.13	86.62	64.58	49.30	20.54	68.16	×1.22
w/ Token Sparse	94.84	93.31	86.14	64.89	48.70	19.58	67.91	×1.33

Method	4K	8K	16K	32K	64K	128K	Avg.	Speedup
LLaMA-3.1-8B-Instruct								
Flash-Attention	95.82	92.77	91.02	84.87	83.43	74.15	87.01	×1.00
w/ Token Sparse	96.06	92.90	91.82	84.81	82.83	73.68	87.02	×1.36
SeerAttention	95.82	92.75	90.97	84.81	83.71	73.85	86.99	×2.19
w/ Token Sparse	95.82	92.84	91.47	85.17	83.19	73.60	87.02	×2.47
X-Attention	96.14	92.46	89.99	84.05	76.05	61.65	83.39	×2.72
w/ Token Sparse	96.14	92.16	90.01	84.26	76.03	62.43	83.51	×3.49

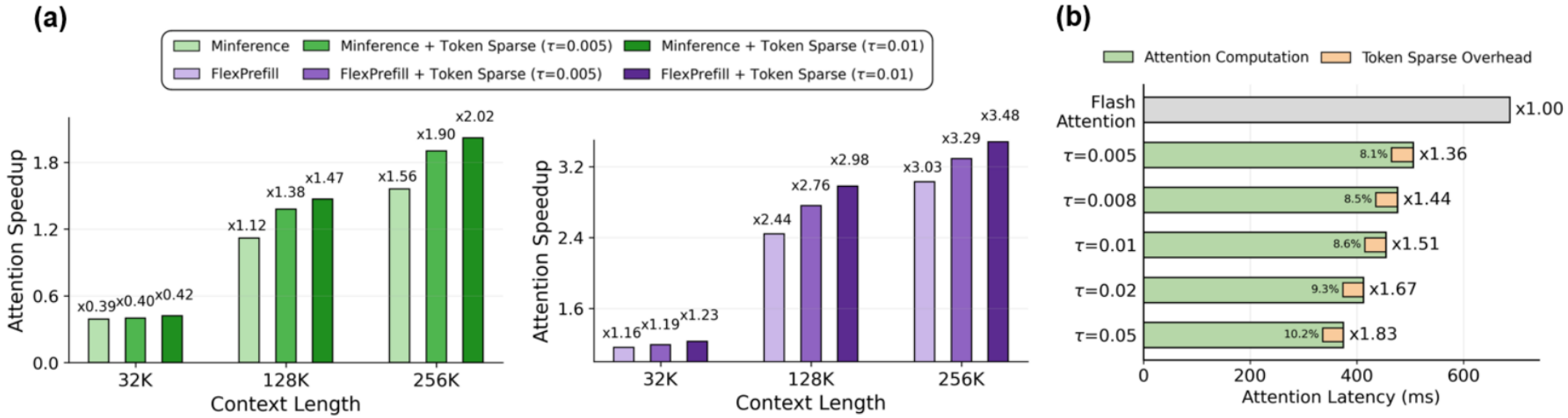
Performance Highlights (2/2)



Better Accuracy-Speedup Trade-off

- FlexPrefill + Token Sparse Attention consistently pushes the Pareto frontier outward
- Same accuracy with higher speedup
- Dynamic token sparsity complements block sparsity

Efficiency Highlights



More Effective at Longer Contexts

- Attention sparsity naturally increases as context length grows.
- Selected-layer sparsity rises from 17% at 4K to 54% at 128K.
- Additional overhead remains below 11% of total attention latency.
- End-to-end latency improvements closely track attention-level speedups.

	8K		128K	
Method	Latency (sec)	Speedup	Latency (sec)	Speedup
Flash-Attention	0.68 ± 0.0061	$\times 1.00$	31.04 ± 0.0569	$\times 1.00$
w/ Token Sparse	0.70 ± 0.0064	$\times 0.97$	24.35 ± 0.0195	$\times 1.27$
Minference	1.52 ± 0.0079	$\times 0.45$	26.84 ± 0.0492	$\times 1.16$
w/ Token Sparse	1.53 ± 0.0103	$\times 0.44$	23.32 ± 0.0124	$\times 1.33$
FlexPrefill	0.90 ± 0.0084	$\times 0.76$	18.45 ± 0.0439	$\times 1.68$
w/ Token Sparse	0.92 ± 0.0098	$\times 0.74$	16.81 ± 0.0272	$\times 1.85$

Summary

Takeaways

- Token Sparse Attention introduces reversible token-level sparsification
- Enables dynamic layer-wise and head-wise token selection
- Fully compatible with existing attention kernels
- Provides complementary gains on top of sparse attention methods

Main Result

- Up to ×3.23 attention speedup
- Less than 1% accuracy degradation
- Consistently improves accuracy–latency trade-off