D2O: Dynamic Discriminative Operations for Efficient Long-context Inference in Large Language Models

Zhongwei Wan, Xinjian Wu, Yu Zhang, Yi Xin, Chaofan Tao, Zhihong Zhu, Xin Wang, Siqi Luo, Jing Xiong, Longyue Wang, Mi Zhang

ICLR 2025

THE OHIO STATE UNIVERSITY

Motivation

- Long-context generative inference in LLMs faces **memory bottlenecks** due to extensive KV cache demands.
- Existing eviction-based methods lead to **context loss and hallucinations**.
- Different layers exhibit varying attention densities; uniform KV cache allocation is **suboptimal**.

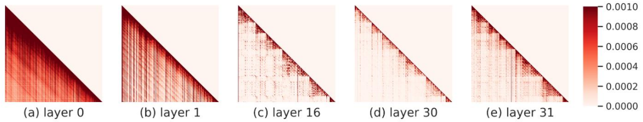


Figure 1: **Attention map density** comparisons of shallow layers (layer 0, 1) and deep layers (layer 16, 30, 31) of LLaMA-2-7B on the GSM8K dataset. We use the mean value of heads for each layer.

Our Approach: Dynamic Discriminative Operations (D2O)

Layer-Level Operation:

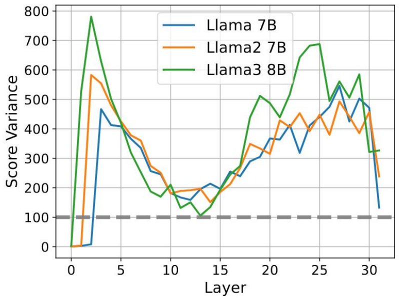


Figure 2: **Variances of attention score** across different layers for various models.

$$S_l = \alpha_l \cdot S,$$

$$\text{where } \alpha_l = \frac{\exp(-F_v^l)}{\sum_{l=1}^{L} \exp(-F_v^l)} \cdot L \cdot \rho,$$

- Dynamically allocate KV cache budget ratio using **inverse variance softmax**, prioritizing layers with denser attention maps.
- Figure 2 shows **variances of attention score** across different layers for various models.

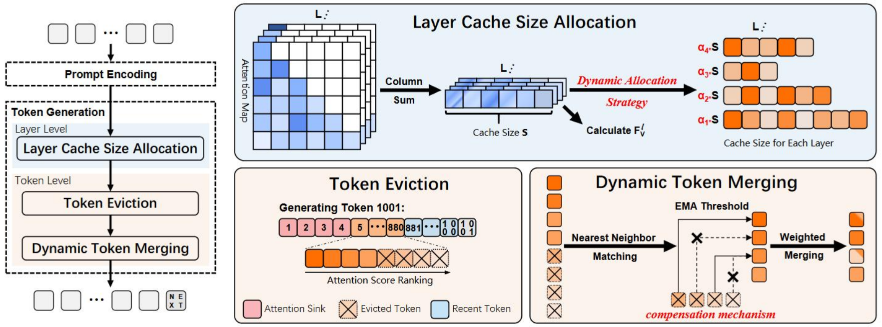
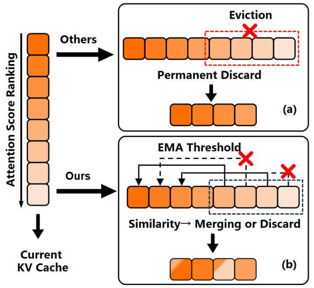


Figure 3: Overview of D2O

Token-Level Operation:



- **A.** Evict tokens based on cumulative attention with attention sink preservation.
- **B.** Dynamically merge evicted tokens using an **EMA-based similarity threshold** to maintain context relevance.

Figure 4: Illustration of dynamic token merging mechanism

A.

$$\text{AttnScore} = \begin{cases} \sum_{i=0}^{L_{\text{prompt}}} \mathbf{A}_p[i,:], & \text{if token } i <= L_{\text{prompt}}, \\ \text{Softmax}\left(\mathbf{q}_i \mathbf{K}^\top / \sqrt{D}\right) + \sum_{i=1}^{L_{\text{prompt}}} \mathbf{A}_p[i,:], & \text{otherwise, token generation} \end{cases}$$

$$\mathbf{K}_c = [\mathbf{K}[:T,:], \mathbf{K}[I,:], \mathbf{K}[-M:,:]], \quad \mathbf{V}_c = [\mathbf{V}[:T,:], \mathbf{V}[I,:], \mathbf{V}[-M:,:]],$$

$$\text{and } I = \text{Top}_N\left(\text{AttnScore}[T:-M], N\right),$$

B.

$$\mathbf{k}_*^{\text{nearest}} = \underset{j \in I^c}{\text{Argmax}} \left(u_{i,j}\right), \text{ where } u_{i,j} = \frac{\mathbf{k}_i^\top \mathbf{k}_j}{\|\mathbf{k}_i\| \|\mathbf{k}_j\|}$$

$$\tau_t = \begin{cases} \frac{1}{L^e} \sum_{i=0}^{L^e} \text{Max}(\mathbf{U}_t[i,:]), & \text{if } t = 0 \text{ for prompt encoding} <= L_{\text{prompt}}, \mathbf{U}_t \in \mathbb{R}^{L^e \times L^c} \\ \beta \text{Max}(\mathbf{U}_t[:]) + (1-\beta)\,\tau_{t-1} & \text{otherwise, } t > 0 \text{ for token generation}, \mathbf{U}_t \in \mathbb{R}^{L^c} \end{cases}$$

$$\mathbf{k}_{cj} = \mathbf{w}_{cj}\mathbf{k}_{cj} + \sum_{\mathbf{k}_{ei} \in \mathbf{K}_e} \mathbf{w}_{ei}\mathbf{k}_e, \quad \mathbf{v}_{cj} = \mathbf{w}_{cj}\mathbf{v}_{cj} + \sum_{\mathbf{v}_{ei} \in \mathbf{V}_e} \mathbf{w}_{ei}\mathbf{v}_e,$$

$$\mathbf{w}_{cj} = \frac{e}{\sum_{\mathbf{k}_{ei} \in \mathbf{K}_e} \exp(\mathbf{u}_{ij})\mathbf{m}_{ij} + e}, \quad \mathbf{w}_{ei} = \frac{\sum_{\mathbf{k}_{ei} \in \mathbf{K}_e} \exp(\mathbf{u}_{ij})\mathbf{m}_{ij}}{\sum_{\mathbf{k}_{ei} \in \mathbf{K}_e} \exp(\mathbf{u}_{ij})\mathbf{m}_{ij} + e},$$

Results & Evaluation

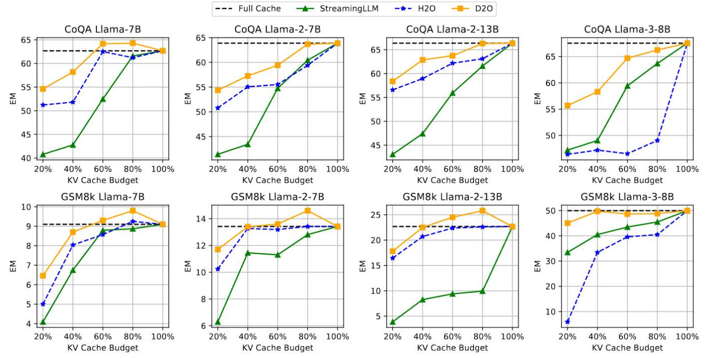


Figure 4: Reasoning dataset

	Methods	NrtvQA	Qasper	MF-en	HotpotQA	2WikiMQA	Musique	GovReport	QMSum	MultiNews	TREC	TriviaQA	SAMSum	PCount	PRe	Lcc	RB-P
		Single-Document QA			Multi-Document QA			Summarization			Summarization			Synthetic		Code	
Llama-3-8B	Full Model	14.25	12.89	22.45	11.03	12.17	6.98	30.80	23.25	4.02	71.00	90.10	42.08	6.33	12.51	72.94	61.26
	Local Window	1.78	4.64	4.10	6.11	6.91	2.81	0.56	10.33	0.02	33.5	28.67	10.56	5.69	2.00	32.80	23.68
	StreamingLLM	10.47	9.96	13.82	9.64	11.05	5.53	19.99	20.53	3.13	62.67	90.05	41.30	5.44	14.05	70.44	57.93
	H_2O	13.27	11.05	17.72	10.38	11.23	6.38	21.29	21.33	3.38	66.63	89.19	41.12	5.52	11.11	71.86	58.29
	RoCo	10.77	10.55	16.54	9.98	8.95	9.52	20.78	20.15	2.59	63.98	86.26	38.59	5.55	10.05	68.78	56.66
	CaM	11.15	11.02	16.84	10.47	8.83	9.45	21.23	20.73	2.57	64.10	87.21	38.69	5.86	10.40	69.72	57.51
	D_2O	**14.43**	**12.66**	**19.93**	**11.92**	**12.79**	**9.88**	**24.36**	**23.42**	**3.95**	**69.72**	**90.99**	**42.36**	**6.61**	**14.67**	**72.43**	**60.00**

Table 1: LongBench benchmarks

Prompt+Gen	256+1024	512+2048	1024+4096	2048+8192
	Max Batch Size			
Full Model	8	4	2	1
H_2O	32 (256)	16 (512)	8 (1024)	4 (2048)
D_2O	32 (256)	16 (512)	8 (1024)	4 (2048)
	Throughput: tokens /s			
Full Model	374.79	198.94	96.95	43.44
H_2O	919.77 (2.45×)	511.75 (2.57×)	281.36 (2.90×)	134.66 (3.10×)
D_2O	878.62 (2.34×)	495.50 (2.49×)	272.28 (2.80×)	132.45 (3.04×)

Table 2: Throughput Comparison

Methods	L=50k	L=100k	L=50k	L=100k
Full Model	97.88	94.46	97.88	94.46
	4096		8192	
StreamingLLM	58.64	47.93	62.84	51.34
H_2O	79.84	69.81	82.32	72.34
SnapKV	83.55	76.22	86.63	80.42
CaM	82.66	78.22	87.59	78.88
D_2O	**91.27**	**87.74**	**94.48**	**91.88**

Table 3: Needle-in-a-haystack

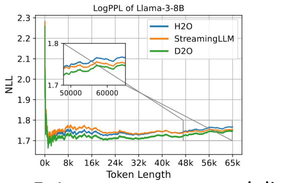


Figure 5: Long sequence modeling

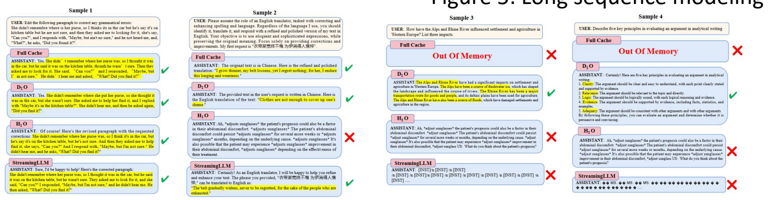


Figure 6: MT bench