REFRESH: LEARNING WHEN TO RETRIEVE FOR LONG-CONTEXT PREFILLING

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-context language models enable retrieval and reasoning over large documents, but quadratic self-attention makes prefilling a major bottleneck. Hybrid attention mitigates this cost by combining a few full-attention heads with cheaper streaming heads, yet its efficiency remains limited by the full-attention budget, whose reduction can hurt long-range retrieval. We observe that full attention serves two roles—computing attention outputs and identifying relevant context regions—and that the latter exhibits substantial redundancy across layers: the same key/value (KV) head in neighboring layers often attends to highly overlapping context regions. Existing cross-layer reuse methods exploit this redundancy, but typically reuse indices or attention according to fixed or heuristic schedules. Building on this observation, we introduce **ReFresh**, a hybrid attention framework that learns where global retrieval should be refreshed or reused across layers, yielding a binary Anchor/Reuse map through dense-to-sparse distillation with the language model frozen. Anchor slots perform full attention, derive block scores from the resulting attention probabilities, and cache the selected KV-block indices for each query block. Reuse slots attend sparsely using the latest indices from an earlier Anchor of the same KV head, preserving global access to distant blocks without recomputing selection—unlike streaming heads, whose access is restricted to attention sinks and a recent window. Across Qwen3-8B and Llama-3.1-8B on LongBench, LongBench v2, and RULER, ReFresh preserves long-context quality with 80% Reuse and achieves a $2.62\times$ prefill speedup for Qwen3-8B at 256K tokens over FlashAttention-2.

1 INTRODUCTION

Long-context language models enable retrieval and reasoning over increasingly large contexts, but their quadratic attention cost makes long-context prefilling expensive: prefilling Qwen3-8B at 256K tokens takes $68.4\,\mathrm{s}$ with FlashAttention-2 (Dao, 2024) on an NVIDIA H800 GPU. A promising direction is hybrid attention, which mixes a small set of full-attention heads with cheaper streaming heads (Xiao et al., 2025; Liu et al., 2025a; Chen et al., 2026b; Yang et al., 2026). Full-attention heads preserve unrestricted access to the entire context, while streaming heads reduce computation by restricting the attended tokens. This hybrid design provides a flexible quality–efficiency trade-off and has become an effective paradigm for long-context inference. Separately, dynamic sparse-prefill methods (Jiang et al., 2024; Xu et al., 2025; Lai et al., 2025; Wang et al., 2025a) reduce attention cost through query-dependent selection of a sparse subset of context positions.

These two directions improve prefilling efficiency in different ways, but both leave substantial redundancy unexploited. In hybrid attention, reducing the fraction of full-attention heads improves efficiency but can hurt long-range retrieval, while retaining more full-attention heads preserves quality at the cost of substantial quadratic computation. Sparse-prefill methods take a different route by replacing full attention with query-dependent sparse selection. Although this reduces the cost of each attention layer, the selection procedure is typically repeated independently at every layer, introducing additional computation during prefilling.

We revisit hybrid attention from the perspective of sparse retrieval and find that its full-attention heads provide more than global access: they also yield accurate retrieval indices that can be reused across nearby layers. Crucially, these retrieval patterns are often highly redundant across layers.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Previous work on cross-layer reuse has observed such redundancy (Deshmukh et al., 2025; Bai et al., 2026; Mu et al., 2024), and Figure 1 illustrates it with a LongBench v2 sample in a 32K context, where a key/value (KV) head attends to highly overlapping regions across neighboring layers. This redundancy suggests that global retrieval need not be refreshed at every layer: indices obtained from full attention can remain useful across subsequent layers. However, existing reuse methods determine where to refresh using fixed or heuristic schedules, rather than learning when a fresh retrieval is worth its cost.

Building on this, we introduce **ReFresh**, a hybrid attention framework that reuses global retrieval across layers. ReFresh combines the head-level routing of hybrid attention with the per-query block selection of sparse prefilling, and adds a learned cross-layer refresh schedule. Each layer–KV-head slot follows one of two routes: an **Anchor** performs full attention and extracts the most relevant blocks, while a **Reuse** slot attends sparsely to the latest blocks selected by the same KV head. Unlike streaming heads, whose access is limited to sinks and a recent window, Reuse preserves global access because the cached indices can point anywhere in the context. ReFresh therefore replaces the conventional full-versus-streaming trade-off as a choice between refreshing and reusing global retrieval, allowing one full-attention computation to serve multiple subsequent layers.

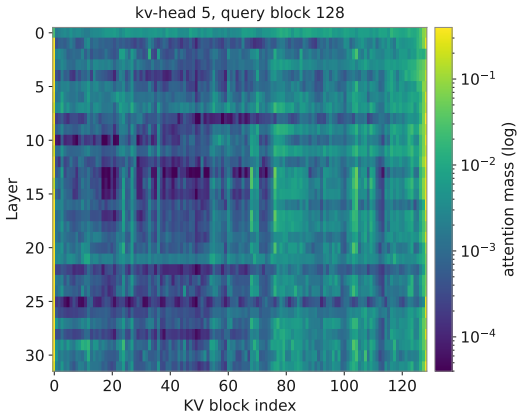

Figure 1: **Cross-layer redundancy in global retrieval.** Attention mass over KV blocks across layers for KV head 5 and query block 128 from a LongBench v2 sample at 32K context, shown on a logarithmic scale. Neighboring layers exhibit highly similar attention patterns, with high-mass regions repeatedly concentrated on the same KV blocks. This redundancy motivates reusing global retrieval results across layers.

A key question is where global retrieval should be refreshed. We learn the Anchor/Reuse routing map through dense-to-sparse distillation with the language model frozen, using learnable gates to balance hidden-state agreement with the dense teacher against Anchor usage. Crucially, training maintains the same persistent per–KV-head index cache used at inference: each slot makes a hard refresh-or-reuse decision, so Reuse slots receive the indices produced by the most recent Anchor rather than newly computed indices at each layer. This keeps index propagation consistent between training and inference. The learned logits are then exported as a single, prompt-independent binary route map at the target Reuse fraction. At inference, Anchor slots refresh the cache with block scoring fused into full attention, while Reuse slots consume the cached indices through block-sparse attention, leaving decoding unchanged.

We evaluate ReFresh on Qwen3-8B and Llama-3.1-8B across LongBench, LongBench v2, and RULER. ReFresh maintains long-context quality with 80% of layer–KV-head slots in Reuse mode, while substantially reducing prefill computation. On an NVIDIA H800 GPU, ReFresh achieves a $2.62\times$ speedup for Qwen3-8B prefill at 256K tokens relative to FlashAttention-2 (Dao, 2024).

Our contributions are summarized as follows:

- **Retrieval reuse for hybrid attention.** We observe that retrieval results can exhibit substantial redundancy across neighboring layers and introduce an Anchor/Reuse design that reuses retrieved block indices across layers while preserving global access.
- **Learned Anchor/Reuse routing.** We formulate the refresh schedule as a layer–KV-head routing problem and learn it via dense-to-sparse distillation with inference-consistent index propagation, yielding a binary Anchor/Reuse map at a target Reuse fraction.
- **Efficient ReFresh implementation.** We integrate fused block scoring into Anchor full attention, maintain a persistent per–KV-head index cache across layers, and execute Reuse slots with indexed block-sparse attention. ReFresh maintains long-context quality with 80% of layer–KV-head slots in Reuse mode and achieves up to a $2.62\times$ prefill speedup at 256K tokens on an NVIDIA H800 GPU.

2 Related Work

2.1 Sparse Attention for Long-Context Prefilling

Sparse attention reduces long-context cost by restricting the key positions that each query can access. Early methods impose fixed connectivity through structured patterns, local windows, or global links (Child et al., 2019; Beltagy et al., 2020; Zaheer et al., 2020). Routing Transformer and StreamingLLM introduce content-based routing and attention-sink preservation, respectively (Roy et al., 2021; Xiao et al., 2024). These methods establish the value of restricting attention, while long-context prefilling requires an additional mechanism to identify useful positions for each input.

Recent sparse-attention methods reduce the cost of constructing such selections in different ways. MInference (Jiang et al., 2024) assigns each attention head an offline pattern from A-shape, vertical-slash, and block-sparse families, then builds the corresponding indices at inference time. FlexPrefill (Lai et al., 2025) adapts sparse patterns and budgets to the input, while XAttention (Xu et al., 2025) estimates block importance with antidiagonal scores. PBS-Attn (Wang et al., 2025a) changes token order before block-sparse execution to concentrate important tokens into fewer blocks. SeerAttention (Gao et al., 2024) learns intrinsic sparse attention patterns, and Elastic Attention (Tang et al., 2026) adapts sparsity ratios at test time. FlashPrefill (Fan et al., 2026), VSPrefill (Chen et al., 2026a), and RouteSparse (Zhang et al., 2026) target selector or pattern-routing overhead, and ProxyAttn (Wang et al., 2025b) and Sketch-and-Walk (Le et al., 2026) reduce scoring work through representative heads or compact sketches. These methods improve pattern design, importance estimation, or kernel layout, but they still build a selection at every layer. ReFresh instead learns when to refresh the block indices during prefilling, so that a single retrieval result serves multiple layers.

2.2 Head Specialization and Cross-Layer Reuse

Hybrid attention mixes full-attention heads with cheaper streaming heads. Head-specialization methods are central to it, allocating different attention scopes to different heads. DuoAttention (Xiao et al., 2025) separates retrieval heads, which retain full context access, from streaming heads, which retain sinks and recent tokens. Retrieval heads have been identified as the mechanism behind long-context factuality (Wu et al., 2025), and RazorAttention (Tang et al., 2025) compresses the KV cache through retrieval heads. PruLong (Bhaskar et al., 2025) likewise learns which heads must retain the full KV cache, reducing memory while preserving long-context performance. HSA (Liu et al., 2025a) allocates full-versus-local attention at KV-head granularity, while ConSA (Chen et al., 2026b) does so at layer- and KV-head granularity, and Autonomy-of-Heads (Yang et al., 2026) infers retrieval and streaming behavior from frozen query-key geometry. ZigzagAttention (Liu et al., 2025b) further enforces homogeneous retrieval or streaming head types within each layer to reduce heterogeneous kernel execution. These methods characterize or assign the context scope and attention mode of a head, but they still retain full-attention heads whose quadratic cost limits efficiency, and they do not reuse retrieval across layers. ReFresh keeps global block access for both routes and learns whether each layer–KV-head slot should refresh its current block indices.

Cross-layer methods reuse attention or routing information across depth. Kascade (Deshmukh et al., 2025) and IndexCache (Bai et al., 2026) reuse sparse indices from selected layers, while Cross-layer Attention Sharing (Mu et al., 2024) and CLAA (McDanel et al., 2026) share or aggregate attention information across layers. LycheeDecode (Lin et al., 2026) learns retrieval and sparse head roles, with sparse heads reusing the tokens selected by retrieval heads, during autoregressive decoding. Evolving Sparsity (Han et al., 2026) also targets decoding, propagating token importance across steps and layers . TidalDecode (Yang et al., 2025b) similarly accelerates decoding with position-persistent sparse attention. ReFresh instead learns, via differentiable gates, which layer–KV-head slots refresh their indices rather than reuse them at fixed or heuristic layers. This enables efficient long-context prefilling by reusing selected indices across layers for each query block and KV head.

3 Method

ReFresh reduces long-context prefill cost by learning where to refresh global retrieval across layers. Each combination of a layer and a key/value (KV) head forms a slot with an Anchor or Reuse route. Anchors compute full attention and select blocks for later layers; Reuse slots compute sparse

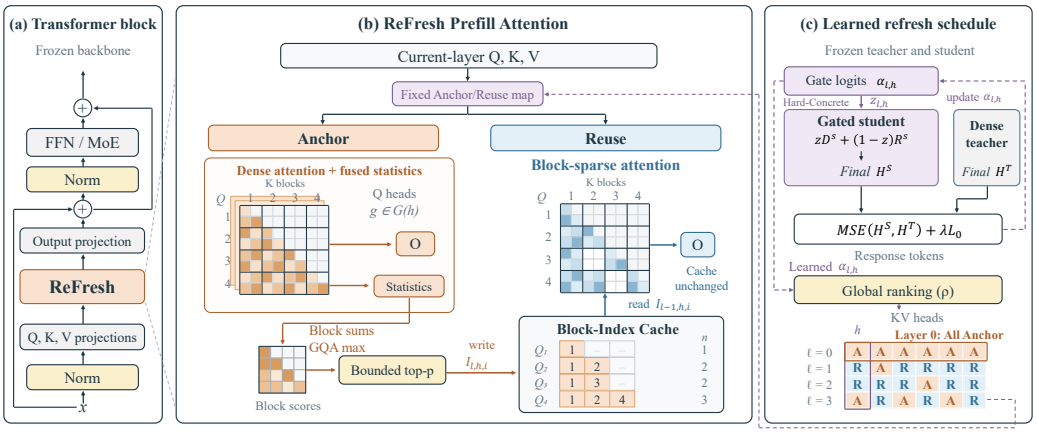

Figure 2: Overview of ReFresh. (a) ReFresh replaces prefill attention within a frozen Transformer block. (b) Anchor slots compute full attention, derive block indices, and update the per–KV-head cache; Reuse slots read cached indices and run block-sparse attention. (c) Dense-to-sparse distillation learns the layer and KV-head refresh schedule and exports a binary route map at target Reuse fraction ρ.

attention over the latest selection for the same KV head. Offline distillation learns the route map while the language-model weights remain frozen. ReFresh modifies only prefilling, while decoding retains the standard attention.

3.1 Anchor and Reuse Attention

Figure 2(b) illustrates how Anchor and Reuse cooperate during prefilling. An Anchor performs full attention and writes the retrieved block indices to a persistent cache, while subsequent Reuse slots consume these indices for block-sparse attention. The cached indices are propagated across layers until the next Anchor refreshes them, allowing each retrieval result to serve multiple layers.

Attention over KV blocks. ReFresh performs attention at the block level. We consider a pretrained model with L layers, indexed by $\ell = 0, \ldots, L-1$, and H_{KV} KV heads. We partition an input sequence of length N into $M = \lceil N/B \rceil$ consecutive blocks of at most B tokens. We use i and j to index query and KV blocks, respectively. Under causal attention, query block i can attend to $\mathcal{V}_i = \{1, \ldots, i\}$.

To unify Anchor and Reuse, let $\mathcal{I} \subseteq \mathcal{V}_i$ denote the KV blocks included in attention. At layer ℓ, let $\mathcal{G}(h)$ denote the set of query heads sharing KV head h. For each $g \in \mathcal{G}(h)$, let $Q_{\ell,g,i}$ denote the queries in block i, and let $K_{\ell,h,\mathcal{I}}$ and $V_{\ell,h,\mathcal{I}}$ denote the keys and values associated with the KV blocks in $\mathcal{I}$. We define

$$\mathrm{Attn}_{\ell,g,i}(\mathcal{I}) = \mathrm{softmax}\left(\frac{Q_{\ell,g,i}K_{\ell,h,\mathcal{I}}^{\top}}{\sqrt{d_k}} + A_{i,\mathcal{I}}\right)V_{\ell,h,\mathcal{I}}, \tag{1}$$

where d_k is the head dimension and $A_{i,\mathcal{I}}$ is the causal mask, which assigns 0 to valid causal pairs and $-\infty$ to future keys, including those within the diagonal block. Blocks determine the retrieval granularity, while attention is computed jointly over all tokens in $\mathcal{I}$. An Anchor uses $\mathcal{I} = \mathcal{V}_i$, whereas a Reuse slot attends to the block indices cached by the most recent Anchor.

Anchor: Full Attention and Index Refresh. For an Anchor slot, query block i attends to all causally valid KV blocks, i.e., $\mathcal{I} = \mathcal{V}_i$:

$$O_{\ell,g,i} = \mathrm{Attn}_{\ell,g,i}(\mathcal{V}_i), \qquad g \in \mathcal{G}(h). \tag{2}$$

In addition to producing the attention output, the Anchor reuses the resulting attention probabilities to identify the KV blocks retained by subsequent Reuse slots. The required block statistics are

therefore collected directly during full attention, without an additional scoring pass. Specifically, let $P_{\ell,g,i}$ denote the attention probability matrix of query block i for query head g, and let $P^{(j)}_{\ell,g,i}$ denote its entries corresponding to KV block j. We first aggregate the attention mass assigned to each KV block and then take the maximum across query heads sharing KV head h:

$$s_{\ell,h,i,j} = \max_{g \in \mathcal{G}(h)} \left\| P^{(j)}_{\ell,g,i} \right\|_1 . \tag{3}$$

where $\| \cdot \|_1$ denotes the entrywise ℓ_1 norm, i.e., the sum of all entries. Thus, a KV block receives a high score whenever it carries substantial attention mass for any query head in the GQA group. We then convert the block scores into a sparse index using a two-stage selection. First, we rank all causally valid KV blocks by their scores and keep at most $m_{\max}$ blocks as candidates. We then adaptively determine how many of these candidates to retain according to their cumulative attention mass: starting from the highest-scoring block, we keep adding blocks until their cumulative score accounts for at least a fraction p of the total score over all valid KV blocks.

Formally, let $m_i = \min\{m_{\max}, |\mathcal{V}_i|\}$ and let $j_1, \ldots, j_{m_i}$ denote the candidate blocks sorted by decreasing score, i.e., $s_{\ell,h,i,j_1} \geq \cdots \geq s_{\ell,h,i,j_{m_i}}$. We select the smallest r satisfying

$$r_p = \min \left\{ r \leq m_i : \sum_{q=1}^{r} s_{\ell,h,i,j_q} \geq p \sum_{j \in \mathcal{V}_i} s_{\ell,h,i,j} \right\}. \tag{4}$$

If no such r exists within the candidate set, we retain all m_i candidates. To avoid overly small selections, we additionally retain at least $\min\{m_{\min}, m_i\}$ candidates. Finally, we always include the first KV block and the diagonal block i, and remove duplicate indices. We denote the resulting sparse block index by $I^S_{\ell,h,i}$. We refer to this procedure as **bounded top-p selection**, which combines adaptive selection based on cumulative attention mass with lower and upper bounds controlled by $m_{\min}$ and $m_{\max}$. The selected indices are then written to the persistent cache:

$$I_{\ell,h,i} = I^S_{\ell,h,i}, \tag{5}$$

where $I_{\ell,h,i}$ denotes the cached block indices for KV head h and query block i after layer ℓ. The cache retains these indices until they are refreshed by a later Anchor. We set all KV heads in the first layer as Anchors to initialize the cache before reuse begins.

Reuse: Sparse Attention with Cached Indices. A Reuse slot skips block scoring and selection and directly uses the cached indices from the preceding layer. For query block i, it computes

$$O_{\ell,g,i} = \mathrm{Attn}_{\ell,g,i}(I_{\ell-1,h,i}), \qquad g \in \mathcal{G}(h), \tag{6}$$

using the current layer's queries, keys, and values. Since no new retrieval is performed, the cached indices are propagated unchanged:

$$I_{\ell,h,i} = I_{\ell-1,h,i}. \tag{7}$$

Consecutive Reuse slots therefore share the same block indices until the next Anchor refreshes the cache. Since the cached indices are derived from full attention, Reuse preserves access to globally relevant context while avoiding repeated global retrieval; unlike streaming heads, whose access is limited to attention sinks and a recent window, the cached indices can point to any causally valid block. For the final query block $i = M$, we apply full attention instead of the cached indices, without updating the cache.

3.2 Learning the Refresh Schedule

ReFresh learns where to refresh global retrieval through offline distillation from the dense model. We associate each layer–KV-head slot with a learnable gate and optimize the routing schedule to balance teacher agreement against Anchor usage. Only the gates in layers $1, \ldots, L-1$ are trained, while the language-model parameters remain frozen. Full training hyperparameters are in Appendix A.1.

Differentiable routing. Each trainable slot (ℓ, h) is associated with a Hard-Concrete gate $z_{\ell,h} \in [0, 1]$, parameterized by a learnable logit $\alpha_{\ell,h}$ (Louizos et al., 2018). This provides a differentiable relaxation of the binary Anchor/Reuse decision. During training, the student computes the Anchor

and Reuse outputs, $D^S_{\ell,h}$ and $R^S_{\ell,h}$, over all query blocks and query heads in $\mathcal{G}(h)$, and mixes them as

$$\widetilde{O}_{\ell,h} = z_{\ell,h} D^S_{\ell,h} + (1 - z_{\ell,h}) R^S_{\ell,h}. \tag{8}$$

Thus, $z_{\ell,h} = 1$ corresponds to Anchor and $z_{\ell,h} = 0$ to Reuse. To update the index cache during training, we use the hard decision $z_{\ell,h} > 0.5$: Anchor refreshes the cached indices, whereas Reuse preserves them. This mirrors inference-time index propagation, ensuring that each Reuse receives indices from the most recent preceding Anchor rather than indices recomputed at the current layer. We fix $z_{0,h} = 1$ for all KV heads to initialize the cache. Since the language model is frozen, the Anchor and Reuse outputs $D^S_{\ell,h}$ and $R^S_{\ell,h}$ are treated as constants: the mixture in Eq. equation 8 differentiates only through the gate, so gradients reach the logit $\alpha_{\ell,h}$ without backpropagating through either attention branch. Discrete block selections are likewise treated as stop-gradient.

Distillation objective. Following DuoAttention (Xiao et al., 2025), we train the routing gates on synthetic multi-passkey retrieval sequences. Let $\mathcal{M}$ denote the set of supervised response-token positions, and let $H^T, H^S \in \mathbb{R}^{N \times d_{\text{model}}}$ denote the final hidden states of the dense teacher and ReFresh student, respectively. We optimize the routing gates with two objectives: matching the teacher representations on $\mathcal{M}$ and reducing the number of Anchor slots. For a Hard-Concrete gate, $\Pr(z_{\ell,h} > 0)$ measures the probability that slot (ℓ, h) is active, and we use its sum as a differentiable penalty on Anchor usage. The resulting objective is

$$\mathcal{L} = \frac{1}{|\mathcal{M}| d_{\text{model}}} \sum_{t \in \mathcal{M}} \left\| H^T_t - H^S_t \right\|_2^2 + \lambda \sum_{\ell=1}^{L-1} \sum_{h} \Pr(z_{\ell,h} > 0)\,, \tag{9}$$

where λ controls the trade-off between teacher agreement and Anchor usage.

Exporting a binary route map. After training, we convert the learned gates into a binary route map for a target Reuse fraction ρ, defined over all LH_{KV} layer–KV-head slots. We globally rank the learned logits $\alpha_{\ell,h}$ across layers $1, \ldots, L-1$ and assign the $\text{round}(\rho L H_{\text{KV}})$ slots with the lowest logits to Reuse; all remaining slots use Anchor. Since the first layer is fixed to Anchor, the feasible range is $0 \le \rho \le (L-1)/L$.

4 Experiments

4.1 Experimental Setup

Models and Benchmarks. We evaluate ReFresh on **Llama-3.1-8B-Instruct** (Grattafiori et al., 2024) and **Qwen3-8B** (Yang et al., 2025a) using LongBench (Bai et al., 2024), LongBench v2 (Bai et al., 2025), and RULER (Hsieh et al., 2024), covering long-context understanding, reasoning, and retrieval.

LongBench contains 21 tasks across six categories: single-document question answering, multi-document question answering, summarization, few-shot learning, synthetic tasks, and code completion. We report the six category scores in their native metrics and compute Avg. over all 21 tasks, including the five Chinese tasks. Appendix B.2 provides the complete task-level breakdown.

LongBench v2 evaluates long-context understanding and reasoning through multiple-choice questions. We report overall accuracy together with breakdowns by difficulty and context length.

RULER evaluates synthetic retrieval, aggregation, and reasoning at 8K, 16K, 32K, 64K, and 128K tokens. We average the 13 task scores at each context length and then average across the five lengths.

In the main comparison tables, bold and underline denote the best and second-best sparse results in each column, respectively, with Full Attention included as a reference. In the RULER appendix B.1, we apply this highlighting only to the Avg. column.

Baselines. ReFresh is compared with Full Attention and three sparse attention methods: FlexPrefill (Lai et al., 2025), XAttention (Xu et al., 2025), and DuoAttention (Xiao et al., 2025). FlexPrefill adapts its sparse pattern and budget to each input and uses $\gamma = 0.95$, $\tau = 0.1$, and min_budget $= 1{,}024$. XAttention selects blocks with antidiagonal scores above a fixed threshold, using threshold 0.9 and stride 8. DuoAttention separates retrieval heads from streaming heads and is evaluated

Table 1: LongBench (Bai et al., 2024) category-level scores for Llama-3.1-8B-Instruct and Qwen3-8B. The six category columns cover all 21 tasks, including the five Chinese tasks within their corresponding categories. Avg. is the mean over all 21 tasks.

Method	Single-Doc QA	Multi-Doc QA	Summarization	Few-shot	Synthetic	Code	Avg.
Llama-3.1-8B-Instruct							
Full Attention	48.87	41.78	17.80	29.48	66.71	24.74	38.23
DuoAttention	47.19	38.91	17.77	26.83	66.51	25.08	37.05
FlexPrefill	47.79	40.15	17.76	**32.06**	56.56	25.51	36.64
XAttention	48.20	40.14	**17.81**	31.35	54.17	**26.22**	36.32
ReFresh	**49.40**	**41.93**	17.79	28.44	**68.01**	24.71	**38.38**
Qwen3-8B							
Full Attention	45.46	38.00	16.41	44.35	67.30	1.20	35.45
DuoAttention	44.38	34.36	16.36	34.79	62.00	**1.21**	32.18
FlexPrefill	44.87	37.95	**16.57**	39.71	50.17	1.05	31.72
XAttention	45.09	**38.01**	16.45	42.94	**63.55**	0.54	34.43
ReFresh	**45.65**	37.96	16.35	**43.28**	63.17	1.19	**34.60**

with sparsity 0.5, 128 sink tokens, and 256 recent tokens. Each baseline uses its official configuration, so the accuracy comparison reflects its recommended operating point while execution cost remains method-specific.

ReFresh Configuration. ReFresh uses block size $B = 128$, top-p threshold $p = 0.9$, minimum block count $m_{\min} = 8$, candidate cap $m_{\max} = 48$, and target Reuse-slot fraction $\rho = 0.8$. It shares block selections within each KV-head group, includes the first and diagonal KV blocks, and uses full attention for the final query block. The route map remains fixed across all benchmarks, while the block selections depend on each input. ReFresh changes prefill attention and keeps standard attention for decoding.

Evaluation Protocol. All experiments use BF16 weights and a batch size of one. Accuracy experiments run on eight NVIDIA H800 GPUs. End-to-end prefill latency is measured on Qwen3-8B with one NVIDIA H800 GPU. For latency evaluation, all methods use the same long-context inputs from LongBench v2, and prefill (time-to-first-token) latency is measured with CUDA events. We run three warm-up iterations and report the average over three measured inputs at each context length. Speedup is reported relative to FlashAttention-2 (Dao, 2024) under the same model, hardware, and timing setup.

4.2 Accuracy Evaluation

ReFresh retains dense-reference quality across all three benchmarks at a target Reuse fraction of 80%. The following results highlight where the learned refresh schedule is most effective and where long-context retrieval remains challenging.

LongBench. ReFresh achieves the best performance among sparse methods on both models (Table 1). On Llama-3.1-8B, it reaches 38.38, which is 0.15 points above Full Attention at 38.23 and 2.06 points above XAttention. On Qwen3-8B, ReFresh scores 34.60, 0.85 points below Full Attention and 0.17 points above XAttention. DuoAttention reaches 32.18 on Qwen3-8B, trailing ReFresh by 2.42 points. These results show that ReFresh preserves the aggregate quality of the dense model, while the learned refresh schedule changes the quality–efficiency trade-off across task categories.

LongBench v2. On Qwen3-8B, ReFresh achieves the highest overall score of 31.81, outperforming Full Attention by 0.80 points and XAttention by 1.39 points (Table 2). Its largest improvement over Full Attention appears in the Long subset, where ReFresh scores 29.63 compared with 24.07. On Llama-3.1-8B, ReFresh reaches 28.63, which is 0.59 points below Full Attention and 2.58 points below XAttention, making it the second-best sparse method on this model.

Table 2: LongBench v2 (Bai et al., 2025) accuracy by difficulty and context length, and RULER (Hsieh et al., 2024) scores across the five context lengths.

Method	LongBench v2						RULER					
	Easy	Hard	Short	Medium	Long	Overall	8K	16K	32K	64K	128K	Avg.
Llama-3.1-8B-Instruct												
Full Attention	30.73	28.30	37.78	26.51	20.37	29.22	93.71	91.59	85.07	83.67	74.20	85.65
DuoAttention	25.52	27.33	32.22	**25.58**	19.44	26.64	92.11	91.88	**88.29**	80.41	63.18	83.17
FlexPrefill	32.81	25.72	32.78	24.65	**28.70**	28.43	93.45	91.67	87.88	83.31	**74.23**	**86.11**
XAttention	**34.38**	**29.26**	**41.11**	24.65	27.78	**31.21**	93.45	91.36	86.89	81.60	69.98	84.66
ReFresh	31.77	26.69	37.22	24.19	23.15	28.63	**93.77**	**92.54**	85.36	**83.38**	73.60	85.73
Qwen3-8B												
Full Attention	36.46	27.65	37.22	29.30	24.07	31.01	89.50	86.19	83.78	74.80	67.82	80.42
DuoAttention	33.85	24.76	28.89	28.84	25.93	28.23	84.78	80.33	72.54	65.02	48.77	70.29
FlexPrefill	33.33	27.33	33.89	27.91	25.93	29.62	89.12	85.58	81.96	**75.42**	**67.07**	**79.83**
XAttention	**39.58**	24.76	32.78	**29.30**	28.70	30.42	**89.58**	**86.05**	**83.03**	73.48	65.81	79.59
ReFresh	38.02	**27.97**	**37.78**	27.91	**29.63**	**31.81**	89.00	85.79	82.64	74.81	65.74	79.60

RULER. ReFresh is the second-best sparse method on both RULER averages and remains close to Full Attention as context length increases (Table 2). Its average across the five context lengths differs from Full Attention by $+0.08$ points on Llama-3.1-8B and -0.82 points on Qwen3-8B. FlexPrefill outperforms ReFresh on RULER by 0.38 and 0.23 points on Llama-3.1-8B and Qwen3-8B, respectively, but trails ReFresh on LongBench by 1.74 and 2.88 points (Table 1).

At 128K, ReFresh reaches 73.60 on Llama and 65.74 on Qwen3-8B, versus 74.20 and 67.82 for Full Attention. The remaining Qwen3-8B loss concentrates in HotpotQA (-7.00) and variable tracking (-6.00); single-needle retrieval variant 1 stays at 100.00 for both methods. Appendix B, specifically Appendix B.1, reports the per-task breakdown, and Section 4.4 ablates the Qwen3-8B 128K setting. Overall, RULER helps localize the task-level sources of the remaining long-context performance gap.

Accuracy Takeaway. Across LongBench and RULER, every model-level average remains within 0.85 points of Full Attention at an 80% target Reuse fraction. On LongBench v2, ReFresh is 0.59 points below Full Attention on Llama-3.1-8B and 0.80 points above it on Qwen3-8B. Overall, these results show that ReFresh substantially reduces refresh computation while preserving full-attention quality across models and task families, ranking as the best or second-best sparse method on every benchmark average.

4.3 Efficiency Evaluation

Figure 3 reports quality against prefill speedup on Qwen3-8B. ReFresh moves the quality–speed point to the upper right: at 128K it reaches 31.81 on LongBench v2 and a $2.1\times$ prefill speedup, while FlashAttention-2 reaches 31.01 at $1.0\times$. FlexPrefill, XAttention, and DuoAttention fall below ReFresh on the same axes. The speedup also grows with context length, reaching $2.62\times$ at 256K, where FlashAttention-2 prefills Qwen3-8B in 68.4 s. As the sequence lengthens, the quadratic cost of refreshing a selection at every layer grows faster than the linear cost of consuming cached indices, so each skipped refresh saves more work. These results show that the quality retained in the accuracy experiments translates into lower prefill latency as context length grows.

4.4 Ablations

Ablations show that the learned anchor placement, rather than the anchor budget, is the main source of the 128K quality (Table 3). The default learned route map uses only 20.1% of slots as anchors with $m_{\max} = 48$ and scores 65.74. For the fixed-interval comparison we hold $m_{\max} = 64$: refreshing at every $N{=}2$ layers places anchors in 50% of the layer–KV-head slots and scores 61.18, while $N{=}4$ reduces the anchor fraction to 25% but scores 59.85; the learned map at the same budget reaches

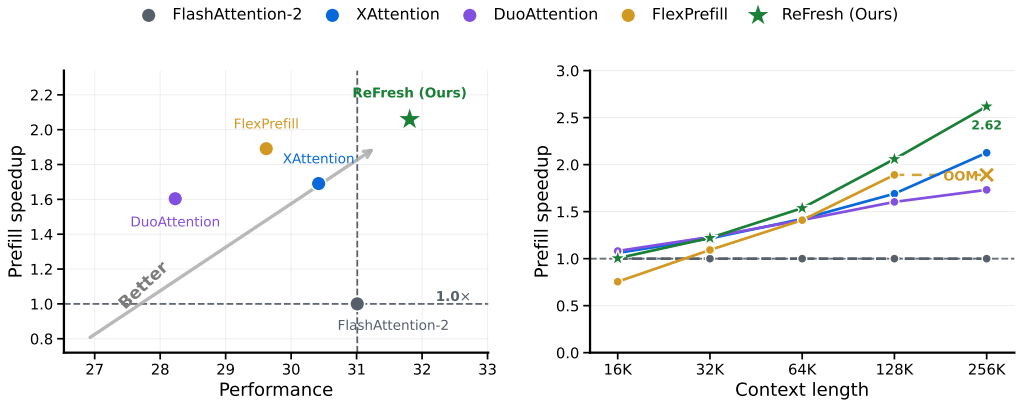

Figure 3: LongBench v2 quality and prefill speedup. The left panel pairs Qwen3-8B LongBench v2 overall accuracy with the 128K Qwen3-8B prefill speedup, normalized to FlashAttention-2. The right panel shows Qwen3-8B prefill speedup of all methods from 16K to 256K.

Table 3: Ablation of ReFresh on Qwen3-8B at RULER 128K. The highlighted row marks the default configuration (learned route map with $m_{\max} = 48$). S-NIAH, M-Key, M-Q/V, and QA are task-group averages, while VT, CWE, and FWE are individual tasks; Avg. averages all 13 tasks.

Schedule	Anchor slots (%)	$m_{\max}$	S-NIAH	M-Key	M-Q/V	VT	CWE	FWE	QA	Avg.
Learned	20.1	48	98.00	43.67	92.25	72.60	3.00	79.00	45.25	65.74
Learned	20.1	64	98.33	46.00	92.44	72.40	2.80	78.67	47.00	66.60
Learned	20.1	96	98.33	47.50	93.75	73.80	2.40	79.00	47.50	67.32
Learned	20.1	160	97.83	48.33	93.75	70.10	3.25	79.17	48.75	67.39
Learned	20.1	Uncapped	98.17	48.33	93.94	68.30	2.20	79.50	49.00	67.34
Fixed interval $N = 2$	50.0	64	95.67	39.17	82.19	59.30	2.15	76.50	44.25	61.18
Fixed interval $N = 4$	25.0	64	95.50	38.33	83.25	47.90	2.60	72.00	43.75	59.85

66.60. The group-level columns show that the fixed-interval schedule loses most on M-Key and VT: 39.17 and 59.30 for N=2, versus 38.33 and 47.90 for N=4.

The candidate cap is comparatively forgiving. Increasing $m_{\max}$ from 64 to 96 raises RULER by 0.72 points, but further increases to 160 and the uncapped setting change it by at most 0.07 points. A small, length-invariant budget therefore already captures nearly all retrieval mass at 128K, which keeps Reuse layers cheap as the sequence grows.

5 Conclusion

We introduced ReFresh, a hybrid attention framework that reduces long-context prefill cost by reusing global retrieval across layers. ReFresh builds on the observation that full-attention heads often retrieve highly overlapping context regions across neighboring layers. Rather than performing global retrieval independently at every layer, ReFresh alternates between Anchor slots, which perform full attention and refresh the retrieved block indices, and Reuse slots, which attend sparsely using the most recent cached indices. A learned layer–KV-head routing map determines where retrieval should be refreshed, with training designed to preserve the same cross-layer index propagation used at inference. Across LongBench, LongBench v2, and RULER, ReFresh preserves full-attention quality at an 80% target Reuse fraction while remaining among the strongest sparse methods. On an NVIDIA H800 GPU, it achieves up to a 2.62× prefill speedup for Qwen3-8B at 256K tokens over FlashAttention-2. These results show that global retrieval need not be recomputed at every layer, and that learned cross-layer reuse can substantially improve the quality–efficiency trade-off of long-context hybrid attention.

AI USE STATEMENT

The authors used generative AI tools only for language editing and stylistic refinement of the manuscript. These tools were not used to formulate the research problem, design the ReFresh method, conduct experiments, select baselines, retrieve literature, generate experimental data, or derive theoretical results. All AI-assisted revisions were manually reviewed by the authors, who verified the technical descriptions, experimental results, references, and conclusions and take full responsibility for the final manuscript.

REPRODUCIBILITY STATEMENT

The ReFresh architecture, Anchor/Reuse formulation, block scoring procedure, cross-layer index cache, and learned routing mechanism are described in the main paper. The experimental section specifies the evaluated models, benchmarks, Reuse fraction, block-selection configuration, numerical precision, hardware, and latency-measurement protocol. Additional implementation details, benchmark configurations, task-level results, and ablation settings are provided in the appendices. All comparative evaluations use the same underlying model checkpoints, benchmark inputs, batch-size setting, and evaluation protocol whenever applicable. Latency comparisons are performed on the same hardware using identical long-context inputs and timing procedures, with FlashAttention-2 used as the dense reference.

REFERENCES

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, et al. LongBench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3119–3137, 2024.

Yushi Bai, Shangqing Tu, Jiajie Zhang, Hao Peng, Xiaozhi Wang, Xin Lv, Shulin Cao, Jiazheng Xu, Lei Hou, Yuxiao Dong, et al. LongBench v2: Towards deeper understanding and reasoning on realistic long-context multitasks. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3639–3664, 2025.

Yushi Bai, Qian Dong, Ting Jiang, Xin Lv, Zhengxiao Du, Aohan Zeng, Jie Tang, and Juanzi Li. IndexCache: Accelerating sparse attention via cross-layer index reuse. *arXiv preprint arXiv:2603.12201*, 2026.

Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

Adithya Bhaskar, Alexander Wettig, Tianyu Gao, Yihe Dong, and Danqi Chen. Cache me if you can: How many KVs do you need for effective long-context LMs? *arXiv preprint arXiv:2506.17121*, 2025.

Guanzhong Chen, Tianhao Gao, and Kai Yang. VSPrefill: Vertical-slash sparse attention with lightweight indexing for long-context prefilling. In *COLM 2026 Workshop on Efficient Reasoning*, 2026a.

Yao Chen, Yinqi Yang, Junyuan Shang, Xiangzhao Hao, Simeng Zhang, Yilong Chen, Tingwen Liu, Shuohuan Wang, and Dianhai Yu. ConSA: Controllable sparsity in hybrid attention via learnable allocation. *arXiv preprint arXiv:2606.18056*, 2026b.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations*, volume 2024, pp. 35549–35562, 2024.

Dhruv Deshmukh, Saurabh Goyal, Nipun Kwatra, and Ramachandran Ramjee. Kascade: A practical sparse attention method for long-context LLM inference. *arXiv preprint arXiv:2512.16391*, 2025.

Qihang Fan, Huaibo Huang, Zhiying Wu, Juqiu Wang, Bingning Wang, and Ran He. Flashprefill: Instantaneous pattern discovery and thresholding for ultra-fast long-context prefilling. *arXiv preprint arXiv:2603.06199*, 2026.

Yizhao Gao, Zhichen Zeng, Dayou Du, Shijie Cao, Peiyuan Zhou, Jiaxing Qi, Junjie Lai, Hayden Kwok-Hay So, Ting Cao, Fan Yang, and Mao Yang. SeerAttention: Learning intrinsic sparse attention in your LLMs. *arXiv preprint arXiv:2410.13276*, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Ruizi Han, Miao Zhang, Ziyue Qiao, and Liqiang Nie. Evolving sparsity: Leveraging token importance dynamics for efficient LLM decoding with sparse attention. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, 2026.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. RULER: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*, 2024.

Huiqiang Jiang, Yucheng Li, Chengruidong Zhang, Qianhui Wu, Xufang Luo, Surin Ahn, Zhenhua Han, Amir H. Abdi, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. MInference 1.0: Accelerating pre-filling for long-context LLMs via dynamic sparse attention. In *Advances in Neural Information Processing Systems*, 2024.

Xunhao Lai, Jianqiao Lu, Yao Luo, Yiyuan Ma, and Xun Zhou. FlexPrefill: A context-aware sparse attention mechanism for efficient long-sequence inference. In *International Conference on Learning Representations*, 2025.

Hoang Anh Duy Le, Sahil Joshi, Zeyu Yang, Zhaozhuo Xu, and Anshumali Shrivastava. Scout before you attend: Sketch-and-walk sparse attention for efficient LLM inference. *arXiv preprint arXiv:2602.07397*, 2026.

Gang Lin, Dongfang Li, Zhuoen Chen, Yukun Shi, Xuhui Chen, Baotian Hu, and Min Zhang. Lycheedecode: Accelerating long-context LLM inference via hybrid-head sparse decoding. *arXiv preprint arXiv:2602.04541*, 2026.

Jing Liu, Jianqiao Lu, Yao Luo, Yuan Yang, Chen Zheng, Deyi Liu, Mengzhao Chen, Chaoyi Zhang, Yunshui Li, Jin Ma, Yutao Zeng, Ya Wang, Hongmin Chen, Ziqiang Pei, Fan Xia, Tianqi Zhang, Xiaoying Jia, Yiyuan Ma, and Siyuan Qiao. HSA: Head-wise sparse attention for efficient and accurate long-context inference, 2025a. URL https://openreview.net/forum?id=WytruI9uM3.

Zhuorui Liu, Chen Zhang, and Dawei Song. Zigzagattention: Efficient long-context inference with exclusive retrieval and streaming heads. *arXiv preprint arXiv:2508.12407*, 2025b.

Christos Louizos, Max Welling, and Diederik P. Kingma. Learning sparse neural networks through L_0 regularization. In *International Conference on Learning Representations*, 2018.

Bradley McDanel, Steven Li, and Harshit Khaitan. CLAA: Cross-layer attention aggregation for accelerating LLM prefill. *arXiv preprint arXiv:2602.16054*, 2026.

Yongyu Mu, Yuzhang Wu, Yuchun Fan, Chenglong Wang, Hengyu Li, Jiali Zeng, Qiaozhi He, Murun Yang, Fandong Meng, Jie Zhou, Tong Xiao, and Jingbo Zhu. Cross-layer attention sharing for pre-trained large language models. *arXiv preprint arXiv:2408.01890*, 2024.

Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. Efficient content-based sparse attention with routing transformers. *Transactions of the Association for Computational Linguistics*, 9:53–68, 2021.

Hanlin Tang, Yang Lin, Jing Lin, Qingsen Han, Danning Ke, Shikuan Hong, Yiwu Yao, and Gongyi Wang. RazorAttention: Efficient KV cache compression through retrieval heads. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=tkiZQlL04w.

Zecheng Tang, Quantong Qiu, Yi Yang, Zhiyi Hong, Haiya Xiang, Kebin Liu, Qingqing Dang, Juntao Li, and Min Zhang. Elastic attention: Test-time adaptive sparsity ratios for efficient transformers. *arXiv preprint arXiv:2601.17367*, 2026.

Xinghao Wang, Pengyu Wang, Dong Zhang, Chenkun Tan, Shaojun Zhou, Zhaoxiang Liu, Shiguo Lian, Fangxu Liu, Kai Song, and Xipeng Qiu. Sparser block-sparse attention via token permutation. *arXiv preprint arXiv:2510.21270*, 2025a.

Yixuan Wang, Huang He, Siqi Bao, Hua Wu, Haifeng Wang, Qingfu Zhu, and Wanxiang Che. ProxyAttn: Guided sparse attention via representative heads. *arXiv preprint arXiv:2509.24745*, 2025b.

Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. Retrieval head mechanistically explains long-context factuality. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=EytBpUGB1Z.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations*, volume 2024, pp. 21875–21895, 2024.

Guangxuan Xiao, Jiaming Tang, Jingwei Zuo, Junxian Guo, Shang Yang, Haotian Tang, Yao Fu, and Song Han. Duoattention: Efficient long-context llm inference with retrieval and streaming heads. In *International Conference on Learning Representations*, volume 2025, pp. 37228–37253, 2025.

Ruyi Xu, Guangxuan Xiao, Haofeng Huang, Junxian Guo, and Song Han. XAttention: Block sparse attention with antidiagonal scoring. *arXiv preprint arXiv:2503.16428*, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025a.

Lijie Yang, Zhihao Zhang, Zhuofu Chen, Zikun Li, and Zhihao Jia. TidalDecode: Fast and accurate LLM decoding with position persistent sparse attention. In *International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=EkfLaCJ7bk.

Yehan Yang, Junyuan Shang, Yang Li, Guanqun Zhao, Shuohuan Wang, and Dianhai Yu. Autonomy-of-heads: Data-free sparse attention from frozen query-key geometry. *arXiv preprint arXiv:2608.06849*, 2026.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems*, 2020.

Chao Zhang, Yifan Ji, Ziyan Zhang, Kai Song, and Fei Lin. Routesparse: Input-conditional pattern routing for budgeted long-context prefilling. *arXiv preprint arXiv:2608.29058*, 2026.

Table 4: RULER (Hsieh et al., 2024) task-level accuracy at 8K context length for Llama-3.1-8B-Instruct and Qwen3-8B. Avg. is the mean over the 13 tasks.

Method	N-MK1	N-MK2	N-MK3	N-MQ	N-MV	N-S1	N-S2	N-S3	CWE	FWE	QA-1	QA-2	VT	Avg.
Llama-3.1-8B-Instruct														
Full Attention	99.50	100.00	99.00	99.25	99.88	100.00	100.00	100.00	98.50	83.50	80.00	59.50	99.10	93.71
DuoAttention	94.00	99.00	99.00	99.25	99.25	100.00	100.00	99.50	98.40	83.67	77.00	52.50	95.90	92.11
FlexPrefill	97.50	99.50	99.50	97.12	99.62	100.00	100.00	100.00	98.10	85.17	79.50	59.00	99.90	<u>93.45</u>
XAttention	99.50	99.50	98.00	98.62	99.25	100.00	100.00	100.00	98.20	84.83	78.00	60.00	98.90	<u>93.45</u>
ReFresh	100.00	100.00	99.00	99.25	99.38	100.00	100.00	100.00	98.10	84.33	79.00	60.00	100.00	**93.77**
Qwen3-8B														
Full Attention	98.00	82.00	77.00	100.00	98.50	100.00	100.00	100.00	91.40	87.83	77.50	53.50	97.80	89.50
DuoAttention	88.00	77.50	63.50	95.50	83.75	100.00	100.00	100.00	90.65	87.00	71.00	48.50	96.80	84.78
FlexPrefill	97.00	80.50	79.00	99.50	96.12	100.00	100.00	100.00	91.80	87.50	76.50	52.00	98.70	<u>89.12</u>
XAttention	99.00	82.50	80.00	99.75	98.50	100.00	100.00	100.00	91.75	86.50	74.00	53.50	99.00	**89.58**
ReFresh	96.50	81.50	76.50	99.25	96.38	100.00	100.00	99.50	93.95	86.17	75.50	53.50	98.30	89.00

A Implementation Details

A.1 Route-Map Training

We train a separate route map for Llama-3.1-8B-Instruct and Qwen3-8B. We freeze the language-model parameters and optimize only the Hard-Concrete gates (Louizos et al., 2018) in layers $1, \ldots, L-1$; every KV head in the first layer remains an Anchor. The gates use temperature $\beta = 2/3$ and stretch bounds $\gamma = -0.1$ and $\zeta = 1.1$. Each trainable logit $\alpha_{\ell,h}$ starts at zero. We optimize the gates for $2{,}000$ steps with AdamW, learning rate 0.01, zero weight decay, batch size one, and no gradient accumulation. We set the L_0 penalty weight $\lambda = 0.003$.

The training data contain synthetic multi-passkey retrieval sequences with context lengths from $8{,}000$ to $128{,}000$ tokens and needle depths from 5% to 95%. The block selection uses $p = 0.9$, $m_{\min} = 8$, and $m_{\max} = 64$ during training, with $m_{\max}$ reduced to 48 at inference. After optimization, we globally rank the learned logits and export a binary map with target Reuse fraction $\rho = 0.8$.

B Additional Experiment Results

B.1 Detailed RULER Results

The main paper reports the average RULER score at each context length. Tables 4, 5, 6, 7, and 8 show the corresponding task-level scores. Each table includes the 13 RULER tasks and the arithmetic mean for Llama-3.1-8B-Instruct and Qwen3-8B. Here, N-MK, N-MQ, N-MV, and N-S denote multi-key, multi-query, multi-value, and single-needle retrieval tasks; CWE and FWE denote Common Words Extraction and Frequent Words Extraction; QA-1 and QA-2 denote SQuAD and HotpotQA, and VT denotes Variable Tracking.

B.2 Detailed LongBench Results

Table 9 reports the complete task-level breakdown for all 21 LongBench tasks. The five Chinese tasks are included alongside the English and code tasks, and the final column uses the official all-task LongBench average reported by each evaluation run. Here, NQA denotes NarrativeQA, MFQA denotes Multi-field QA, TREC denotes the TREC classification task, and PassageRet denotes passage retrieval; the suffixes `-en` and `-zh` identify the English and Chinese variants, respectively.

Table 5: RULER (Hsieh et al., 2024) task-level accuracy at 16K context length for Llama-3.1-8B-Instruct and Qwen3-8B. Avg. is the mean over the 13 tasks.

Method	N-MK1	N-MK2	N-MK3	N-MQ	N-MV	N-S1	N-S2	N-S3	CWE	FWE	QA-1	QA-2	VT	Avg.
Llama-3.1-8B-Instruct														
Full Attention	99.50	100.00	99.00	96.62	99.00	100.00	100.00	100.00	91.90	87.67	79.50	56.00	81.50	91.59
DuoAttention	95.50	100.00	97.00	98.00	98.38	100.00	100.00	100.00	92.35	87.67	74.00	53.00	98.50	91.88
FlexPrefill	99.50	99.50	99.50	93.38	97.12	100.00	100.00	100.00	93.10	87.00	74.50	54.50	93.60	91.67
XAttention	99.50	98.50	98.50	94.75	97.75	100.00	100.00	100.00	92.60	88.83	78.00	55.50	83.80	91.36
ReFresh	99.50	100.00	99.50	97.88	99.25	100.00	100.00	100.00	88.05	87.17	79.50	58.00	94.20	**92.54**
Qwen3-8B														
Full Attention	93.50	67.00	70.00	99.88	96.25	100.00	100.00	100.00	79.60	94.67	71.00	50.00	98.60	86.19
DuoAttention	76.50	63.00	56.50	95.88	82.12	100.00	100.00	100.00	74.70	90.00	64.00	47.50	94.10	80.33
FlexPrefill	96.00	66.00	64.50	99.75	94.25	100.00	100.00	100.00	81.15	93.00	72.50	46.50	98.90	85.58
XAttention	97.50	66.50	69.00	100.00	96.25	100.00	100.00	100.00	80.70	94.50	67.00	48.50	98.70	**86.05**
ReFresh	94.50	69.50	65.50	99.38	94.88	100.00	100.00	100.00	84.85	91.00	70.50	47.00	98.10	85.79

Table 6: RULER (Hsieh et al., 2024) task-level accuracy at 32K context length for Llama-3.1-8B-Instruct and Qwen3-8B. Avg. is the mean over the 13 tasks.

Method	N-MK1	N-MK2	N-MK3	N-MQ	N-MV	N-S1	N-S2	N-S3	CWE	FWE	QA-1	QA-2	VT	Avg.
Llama-3.1-8B-Instruct														
Full Attention	100.00	98.50	100.00	91.62	97.25	100.00	100.00	100.00	13.85	88.50	79.50	50.00	86.70	85.07
DuoAttention	95.00	99.00	96.00	95.25	95.50	100.00	98.50	100.00	57.95	91.17	71.00	50.50	97.90	**88.29**
FlexPrefill	99.00	99.00	99.00	88.00	92.75	100.00	98.00	100.00	59.70	87.83	69.50	50.50	99.20	87.88
XAttention	100.00	92.50	98.00	89.88	95.88	100.00	100.00	100.00	42.30	89.67	75.00	51.00	95.40	86.89
ReFresh	100.00	98.00	97.50	93.62	97.50	100.00	100.00	100.00	5.10	88.17	78.50	52.50	98.80	85.36
Qwen3-8B														
Full Attention	92.50	75.50	62.00	96.88	96.75	100.00	100.00	100.00	52.25	93.50	73.50	46.50	99.70	83.78
DuoAttention	70.00	52.50	24.00	91.38	89.75	100.00	100.00	100.00	31.80	92.83	60.00	37.50	93.20	72.54
FlexPrefill	89.00	73.50	53.00	96.12	96.12	100.00	100.00	98.50	52.80	93.00	68.50	45.00	99.90	81.96
XAttention	91.00	72.00	61.00	96.62	96.38	100.00	100.00	100.00	47.80	93.67	73.00	48.00	99.90	**83.03**
ReFresh	92.00	76.00	53.50	95.88	96.25	100.00	100.00	99.50	53.10	90.67	71.00	46.50	99.90	82.64

Table 7: RULER (Hsieh et al., 2024) task-level accuracy at 64K context length for Llama-3.1-8B-Instruct and Qwen3-8B. Avg. is the mean over the 13 tasks.

Method	N-MK1	N-MK2	N-MK3	N-MQ	N-MV	N-S1	N-S2	N-S3	CWE	FWE	QA-1	QA-2	VT	Avg.
Llama-3.1-8B-Instruct														
Full Attention	98.50	97.50	98.50	94.75	97.00	100.00	100.00	100.00	0.40	81.50	77.50	48.50	93.60	83.67
DuoAttention	95.00	97.00	86.00	91.12	92.62	100.00	91.50	100.00	0.50	83.33	65.00	45.00	98.20	80.41
FlexPrefill	99.00	98.50	94.00	93.25	96.75	100.00	99.00	100.00	9.80	79.17	71.50	44.00	98.00	83.31
XAttention	98.00	86.50	92.00	89.88	95.50	99.50	99.00	100.00	2.00	81.67	72.50	46.50	97.70	81.60
ReFresh	99.50	98.50	93.50	92.38	97.00	100.00	98.50	100.00	0.05	78.00	79.50	49.50	97.50	**83.38**
Qwen3-8B														
Full Attention	87.50	57.00	21.00	95.12	97.38	100.00	100.00	100.00	27.25	78.17	67.50	46.50	95.00	74.80
DuoAttention	71.00	34.00	8.00	84.38	84.50	100.00	100.00	99.50	6.35	78.17	51.00	38.50	89.90	65.02
FlexPrefill	90.50	52.00	25.50	96.00	95.25	99.50	100.00	99.50	28.80	78.50	69.50	45.50	99.90	**75.42**
XAttention	89.50	47.00	18.50	96.25	98.00	100.00	100.00	100.00	23.75	76.17	63.00	46.00	97.10	73.48
ReFresh	88.50	58.50	27.00	96.88	94.25	100.00	100.00	100.00	19.85	79.67	66.50	47.00	94.40	74.81

Table 8: RULER (Hsieh et al., 2024) task-level accuracy at 128K context length for Llama-3.1-8B-Instruct and Qwen3-8B. Avg. is the mean over the 13 tasks.

Method	N-MK1	N-MK2	N-MK3	N-MQ	N-MV	N-S1	N-S2	N-S3	CWE	FWE	QA-1	QA-2	VT	Avg.
Llama-3.1-8B-Instruct														
Full Attention	97.50	85.00	58.00	98.12	97.62	100.00	99.50	100.00	0.10	59.83	73.00	38.50	57.40	74.20
DuoAttention	82.00	63.50	9.50	89.50	83.88	96.88	88.50	92.50	0.15	65.17	40.00	30.00	79.70	63.18
FlexPrefill	96.50	85.00	54.50	98.00	97.00	98.44	100.00	99.00	0.10	50.00	71.50	36.50	78.40	**74.23**
XAttention	91.50	67.50	39.50	96.50	93.62	98.44	98.50	100.00	0.10	67.00	56.50	37.00	63.60	69.98
ReFresh	95.00	83.00	48.50	97.25	94.00	98.44	100.00	100.00	0.10	55.00	74.00	37.50	74.00	73.60
Qwen3-8B														
Full Attention	76.00	55.00	8.50	90.38	95.12	100.00	93.50	95.00	4.10	84.00	55.00	46.50	78.60	67.82
DuoAttention	47.50	17.50	0.00	32.25	46.25	100.00	50.50	91.50	1.30	86.83	41.50	31.00	87.90	48.77
FlexPrefill	78.50	50.00	10.00	92.50	94.88	100.00	92.00	94.00	4.90	81.83	57.00	42.00	74.30	**67.07**
XAttention	78.00	46.50	7.00	90.00	93.50	98.44	91.50	90.50	4.25	85.50	51.50	44.00	74.80	65.81
ReFresh	77.00	48.00	6.00	90.25	94.25	100.00	95.50	98.50	3.00	79.00	51.00	39.50	72.60	65.74

Table 9: LongBench (Bai et al., 2024) task-level scores for Llama-3.1-8B-Instruct and Qwen3-8B. The table reports all 21 tasks; the abbreviations denote NarrativeQA (NQA), Qasper (QSP), Multi-field QA in English (MFQ-A) and Chinese (MFQ-Z), 2WikiMQA (2W), HotpotQA (HP), MuSiQue (MQ), GovReport (GR), QMSum (QMS), MultiNews (MN), TREC (TREC), TriviaQA (TQA), SAMSum (SAM), passage count (PC), passage retrieval in English (PR-en) and Chinese (PR-zh), DuReader (DR), VCSum (VCS), LSHT (LSHT), LCC (LCC), and RepoBench-P (RP). Avg. is the official LongBench average over all tasks, computed by the evaluation script.

Method	NQA	QSP	MFQ-A	2W	HP	MQ	GR	QMS	MN	TREC	TQA
Llama-3.1-8B-Instruct											
Full Attention	29.30	47.22	55.91	48.45	57.61	32.77	19.98	19.35	15.36	15.00	91.42
DuoAttention	26.53	45.89	55.80	44.26	56.58	27.70	19.68	19.20	15.45	9.00	89.25
FlexPrefill	29.37	46.10	55.65	42.52	55.08	31.60	19.89	19.31	15.20	19.00	90.45
XAttention	28.54	46.54	56.33	45.35	54.80	30.70	20.17	19.28	15.36	18.00	89.53
ReFresh	31.41	47.39	55.46	49.87	57.83	31.39	20.30	19.08	15.42	11.00	92.10
Qwen3-8B											
Full Attention	24.85	43.18	51.12	44.11	57.98	29.65	18.16	17.92	14.60	21.50	86.25
DuoAttention	23.58	42.56	49.17	38.56	53.41	25.37	18.31	17.25	14.65	12.50	86.87
FlexPrefill	23.55	42.15	50.49	41.82	61.52	27.59	18.19	18.35	14.63	17.00	86.72
XAttention	24.93	43.49	50.65	43.71	58.53	29.45	18.19	18.02	14.63	21.00	86.26
ReFresh	24.50	43.54	51.10	42.48	57.11	31.71	18.15	17.96	14.56	18.50	86.83

Method	SAM	PC	PR-en	MFQ-Z	DR	VCS	LSHT	PR-zh	LCC	RP	Avg.
Llama-3.1-8B-Instruct											
Full Attention	11.52	10.50	99.50	63.06	28.30	16.51	0.00	90.12	24.58	24.90	38.23
DuoAttention	8.09	4.00	99.50	60.52	27.09	16.76	1.00	96.04	24.15	26.01	37.05
FlexPrefill	15.80	3.00	81.00	60.05	31.39	16.62	3.00	85.67	24.28	26.75	36.64
XAttention	17.87	4.50	81.00	61.38	29.73	16.43	0.00	77.02	25.13	27.31	36.32
ReFresh	10.65	9.75	100.00	63.32	28.64	16.38	0.00	94.27	24.28	25.13	38.38
Qwen3-8B											
Full Attention	31.64	5.39	97.00	62.70	20.25	14.95	38.00	99.50	0.45	1.95	35.45
DuoAttention	30.55	0.00	90.00	62.20	20.09	15.22	9.25	96.00	0.79	1.64	32.18
FlexPrefill	31.11	1.00	60.50	63.27	20.86	15.11	24.00	89.00	0.63	1.47	31.72
XAttention	31.23	5.14	91.50	61.29	20.34	14.98	33.25	94.00	0.42	0.65	34.43
ReFresh	31.54	3.50	95.00	63.48	20.56	14.74	36.25	91.00	0.39	1.99	34.60