PICKPACK: SCALING CONTEXT EXTENSION TO MILLIONS OF TOKENS WITH SPARSE ATTENTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Due to a growing range of applications, long-context processing and long-form generation with AI are becoming increasingly common, particularly with the transition to agentic workloads. Agents are long-lived, continuously produce and consume streams of tokens, and base their subsequent responses on their entire interaction history; multi-agent systems can further amplify this token volume. Models capable of processing long contexts are therefore increasingly necessary, but training with larger context windows is expensive, so most models are trained with windows of 32K to 256K tokens. Application demands can extend to millions of tokens, which creates a need for training-free context extension. Training-free context extension presents two key challenges: (1) *out-of-distribution context embeddings*, where unseen context lengths can shift attention distributions away from those seen during training, and (2) *computational complexity*, since attention's quadratic cost makes processing millions of tokens expensive. We propose PICKPACK, which addresses both challenges with two composable operators in a single FlashAttention pass: PACK maps every query–key pair to a relative position inside the trained window, and PICK uses the attention scores to dynamically decide which key blocks to skip. At 50% sparsity, PICKPACK improves accuracy over dense attention under the same position map on seven of eight Qwen3 model sizes and context lengths, while being up to 1.2× faster. Against the recipes that released Qwen2.5, Qwen3.5, and Qwen3.8 models ship with, it is 3.4–11 points more accurate and up to 1.2× faster from 128K to one million tokens, and 11.6 points more accurate at four million tokens, 16× the trained window of Qwen3.5-9B. Code: https://anonymous.4open.science/r/Pick-Pack/

1 INTRODUCTION

Large language models are increasingly deployed in context-rich settings, particularly with the transition to agentic execution. Agentic and multi-agent systems ingest and produce large numbers of tokens, and each response is conditioned on the entire interaction history. For instance, agents interleave reasoning, planning, and tool use over many turns (Yao et al., 2023), while coding agents operate over entire software repositories. Long-context benchmarks reflect this shift: coding agents are evaluated on contexts of up to millions of tokens (Rando et al., 2025), and retrieval benchmarks place entire corpora of millions of tokens in the prompt (Lee et al., 2025b). Serving such workloads therefore requires models whose context windows accommodate millions of tokens.

Recent frontier open models, such as Kimi K3, DeepSeek-V4, and GLM-5.2, support context windows of one million tokens (Kimi Team, 2026; DeepSeek-AI, 2026; Z.ai, 2026). Most open models, however, are trained with windows of 32K to 256K tokens. Qwen2.5 and Qwen3 are pretrained with 32K-token windows (Qwen Team, 2024; 2025b), and even Qwen2.5-7B-Instruct-1M, Qwen3.5, and Qwen3.8 are trained on sequences of up to 256K tokens and reach one million tokens by extending the window at inference time (Qwen Team, 2025a; 2026a;b). Extending training to substantially longer contexts is challenging for three reasons. First, the cost of attention in each forward and backward pass grows quadratically with context length, which makes long-context training expensive in both compute and memory. Second, genuine long-context training data are scarce (Tang et al., 2026). Third, extending the context during fine-tuning can degrade performance on shorter contexts (Ding et al., 2024; Tang et al., 2026). These challenges make it attractive to extend the

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.

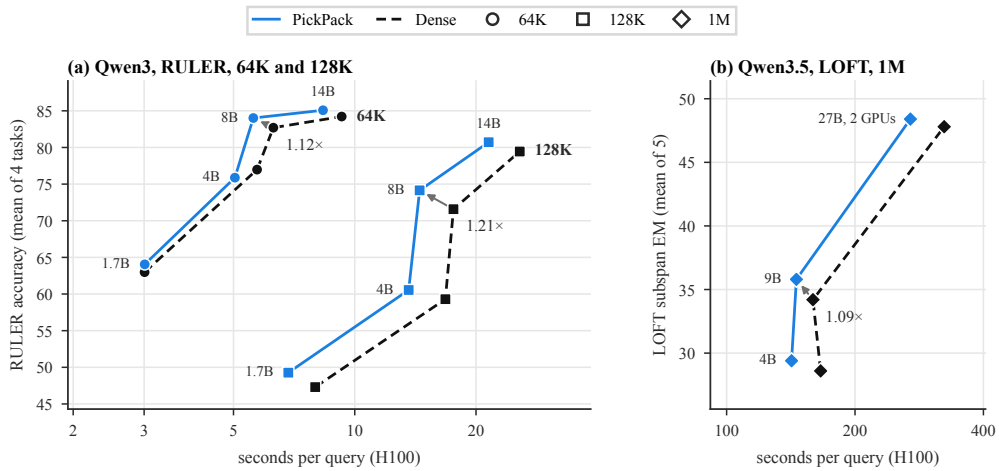


Figure 1: **PICKPACK makes context extension faster and more accurate.** We compare PICKPACK at 50% sparsity with dense attention under the same position map (Tang et al., 2026), on a four-task RULER subset (Table 2) for Qwen3 base models at 64K and 128K **(a)** and on LOFT for Qwen3.5 at one million tokens, with 27B on two GPUs **(b)**. PICKPACK is up to 1.2× faster, and more accurate at three of the four sizes at 64K and at every size at 128K and 1M.

context window of a pretrained model at inference time, without additional training (Peng et al., 2024; Jin et al., 2024; An et al., 2024; Tang et al., 2026).

Training-free context extension needs to address two challenges. (1) *Out-of-distribution context embeddings.* The embeddings that attention receives become out of distribution for two reasons. First, beyond the trained window, rotary position embeddings (Su et al., 2021) reach rotation angles never seen during training (Chen et al., 2023; Peng et al., 2024), so training-free methods map every relative position, or its rotation angle, back into the trained range (Chen et al., 2023; Peng et al., 2024; Jin et al., 2024; An et al., 2024; Tang et al., 2026). Second, even after this mapping, attention still processes far more tokens than it saw in training, all compressed into a limited rotational space, and prior work shows that this raises the entropy of attention (Han et al., 2024; Nakanishi, 2025). (2) *Computational complexity.* Training-free extension does not reduce the cost of prefill, which still grows quadratically with the prompt length, and at million-token lengths attention becomes the most time-consuming step. In our measurements, dense prefill of Qwen2.5-7B takes 168.5 seconds at 512K tokens and 639 seconds at 1M tokens, 3.8× longer for twice the tokens.

We address both challenges with sparsity aligned with FlashAttention. The effect on prefill time is direct: skipping blocks of the attention matrix, that is, ignoring a set of keys for a set of queries, reduces FlashAttention's work. Sparsity can also reduce the out-of-distribution problem: by skipping the tiles whose scores fall far below the running maximum, PICK, our selection step, keeps each query's attention on fewer keys, closer to what the model saw in training. Adding sparsity improves downstream accuracy across position maps and model sizes (Section 5.2). We also measure the effect directly: in an in-window simulation of context extension (Section 3), sparsity reduces the error of the attention output (Section 5.6). This is consistent with prior work: LM-Infinite (Han et al., 2024) bounds the attention context so that attention entropy stays within the range seen during training, and in an idealized model, length generalization provably holds when each prediction depends on only a few tokens within a bounded span (Golowich et al., 2025). Our method, PICKPACK, therefore combines repositioning with hardware-aware sparsity. It runs two composable operators within the attention forward pass: PACK places each key block at positions inside the trained window, and PICK computes only the blocks that can contribute significantly to the attention output.

We show that PICKPACK improves accuracy and reduces prefill cost beyond the trained window over state-of-the-art context extension methods (Figure 1 and Section 5). On Qwen3-8B at 128K, it is 2.05 points more accurate on RULER-13 than Jet-Long, the strongest training-free baseline, while running 1.21× faster (Table 1). On Qwen3.5-9B at one million tokens, it is 7.6 points more accurate on LOFT than the released YaRN recipe while running 1.09× faster (Table 3). We also extend models far beyond the lengths of prior work: on Qwen3.5-9B at 4M tokens, 16× its trained window, PICKPACK reaches 34.0 QAMPARI coverage against 22.4 for YaRN (Table 4).

2 Related Work

Prior work falls into four lines: training-free context extension, sparse attention for prefill, token selection with remapped positions, and training for length generalization.

Training-free context extension. Position Interpolation compresses position indices into the trained range (Chen et al., 2023), and NTK-aware and dynamic NTK scaling rescale the rotary base instead (bloc97, 2023; emozilla, 2023); YaRN rescales the low-frequency rotary components and adds an attention temperature (Peng et al., 2024), and later variants refine the rescaling or interpolate logits (Tian et al., 2026; Li et al., 2025). Grouping maps reuse trained positions: ReRoPE caps distances (Su, 2023), Self-Extend groups distant tokens by floor division (Jin et al., 2024), and DCA splits attention into intra-chunk, successive-chunk, and inter-chunk parts (An et al., 2024). Length-adaptive maps set the grouping from the input length (Xu et al., 2025b; Dang et al., 2025; Zhang et al., 2026), and Jet-Long sets the group size to $G = \lceil L/W \rceil$ (Tang et al., 2026). All of these compute dense attention under the new positions. PickPack uses such a map as its Pack operator (Jet-Long's, DCA's, and a three-region map, Appendix B) and adds Pick.

Sparse attention for prefill. A second line of work reduces the quadratic cost of prefill for inputs inside the trained window. MInference assigns each head a sparse pattern, such as vertical and slash lines, and estimates its indices from the last queries (Jiang et al., 2024); FlexPrefill and XAttention keep blocks until their estimated attention mass reaches a threshold, from query-aware patterns and antidiagonal scores respectively (Lai et al., 2025; Xu et al., 2025a); and SampleAttention and SpargeAttn filter blocks from sampled queries or compressed query and key representatives (Zhu et al., 2024; Zhang et al., 2025). SpargeAttn additionally skips a block inside the FlashAttention loop (Dao et al., 2022) when its scores fall far below the running maximum (Zhang et al., 2025), and BLASST uses this test on its own (Yuan et al., 2026). PickPack uses BLASST's test as its Pick operator, but beyond the trained window and on the scores under the position map, where we study its effect on accuracy as well as on cost. Decode-time sparsity and KV-cache eviction (Xiao et al., 2024b; Zhang et al., 2023; Li et al., 2024; Tang et al., 2024; Xiao et al., 2025; Joshi et al., 2026) act on the cache that prefill produces, and complement PickPack, which sparsifies prefill only.

Selection with remapped positions. Several training-free methods already combine token selection with remapping. LM-Infinite attends to the initial and recent tokens with a capped distance and can also keep the middle tokens with the largest scores (Han et al., 2024); InfLLM, LongHeads, ReAttention, and InfiniteHiP select context blocks or tokens and place them at shared or consecutive positions inside the trained window (Xiao et al., 2024a; Lu et al., 2024; Liu et al., 2025; Lee et al., 2025a); and Qwen2.5-1M serves DCA together with MInference, estimating MInference's patterns under continuous positions because, as its authors report, DCA's discontinuous positions disrupt them (Qwen Team, 2025a). These methods either assign positions to what they select or select under positions that attention does not use. PickPack instead fixes the positions with a training-free map and selects inside the attention pass, from the scores under that same map.

Training for length generalization. Other approaches change the model. Continued pretraining and fine-tuning extend the window with long data (Xiong et al., 2024; Fu et al., 2024; Chen et al., 2024; Ding et al., 2024), as Qwen2.5-1M does up to 256K (Qwen Team, 2025a); sparse architectures are trained with fixed or learned attention patterns (Child et al., 2019; Beltagy et al., 2020; Zaheer et al., 2020; Yuan et al., 2025; Lu et al., 2025), and sparse attention trained with an adaptive entmax generalizes to far longer contexts (Vasylenko et al., 2026); and encodings and architectures without RoPE avoid rotation angles altogether (Press et al., 2022; Kazemnejad et al., 2023; Katharopoulos et al., 2020; Gu & Dao, 2024). We focus on the training-free setting, in which the checkpoint is fixed and only inference changes. A checkpoint trained for length can still be extended past its own window with PickPack, as we do with Qwen2.5-7B-Instruct-1M.

3 Motivation: Sparsity Recovers In-Distribution Attention

Context-extension methods remap relative positions into the range observed during training. This keeps positional features in distribution, but many more keys still compete for each query: compressing many tokens into the same trained range can increase attention entropy and dilute the mass

assigned to relevant evidence (Han et al., 2024; Nakanishi, 2025). We hypothesize that sparsity partly reverses this effect by removing competing keys.

Testing this hypothesis requires a reference computation. For sparse attention inside the trained window, the natural reference is dense attention at the same length, but extension beyond the window has no gold-standard output. We therefore use an *in-window simulation*. For a trained window W, we choose an apparent window $\tilde{W} = W/G$ and a pair of inputs of lengths $\tilde{W}$ and W that contain the same task-relevant evidence and require the same next-token target; the additional tokens are distractors. Applying a PACK recipe built for window $\tilde{W}$ to the length-W input simulates a $G\times$ extension, while dense attention on the same input stays inside the trained window. With $\hat{o}_G$ the attention output of the simulated extension and o_W that of dense attention at native positions, we measure the relative error $\mathcal{E}_{\mathrm{sim}}(G) = \|\hat{o}_G - o_W\|_2 / \|o_W\|_2$. A sparse extension supports our hypothesis when it reduces this error relative to its dense counterpart. The error is only a proxy: additional context can change the model's computation even when the evidence and target are fixed. We use it as a micro-metric for context-extension mechanisms and report it for PACK_{JL} in Section 5.6.

4 PICKPACK

To implement PICKPACK, we make two changes to the FlashAttention kernel. First, the relative position of every query–key pair must stay within the trained window W. Second, the kernel must sparsify the attention computation without significant additional overhead. PACK handles the first change and PICK the second, and PICKPACK is their decoupled composition: we write $\text{PICK} \circ \text{PACK}_X$ for PICK applied under the PACK recipe X. By decoupled, we mean that the choice of PICK places no restriction on the choice of PACK, and vice versa.

4.1 PACK ALGORITHM

Consider prefill over a context of $L > W$ tokens, where W is the trained context window of the model. With rotary position embeddings (Su et al., 2021), the raw attention score of the query at position i for the key at position j is

$$z_{ij} = q_i^\top R_{\delta_{ij}} k_j / \sqrt{d}, \qquad \delta_{ij} = i - j, \tag{1}$$

where R_δ is the RoPE rotation by δ and d is the head dimension. The output of query i is

$$o_i = \sum_{j \le i} \frac{\exp(z_{ij})}{\sum_{t \le i} \exp(z_{it})} v_j, \tag{2}$$

where v_j is the value at position j. When $L > W$, $i - j$ can exceed $W - 1$, beyond the range seen in training. A PACK recipe assigns new positions to each query–key pair to prevent this.

Definition 1 (PACK recipe)**.** *A* PACK *recipe for a trained context window W is a pair of position maps ϕ_q, ϕ_k defined for every causal query–key pair $0 \le j \le i < L$. The recipe is valid if*

$$0 \le \phi_q(i,j) - \phi_k(i,j) \le W - 1 \qquad \textit{for all } 0 \le j \le i < L. \tag{3}$$

Thus, a valid recipe maps every relative query–key position into the range seen during training.

Grouping-based approaches to context extension can be expressed as PACK recipes. For example, let w, with $0 < w < W$, be the local window, and let $G = \lceil L/W \rceil$. The Jet-Long recipe PACK_{JL} is

$$\big(\phi_q^{\text{JL}}(i,j), \phi_k^{\text{JL}}(i,j)\big) = \begin{cases} (i,\, j), & i > j \text{ and } i - j < w, \\ \big(\lfloor i/G \rfloor,\, \lfloor j/G \rfloor\big), & \text{otherwise.} \end{cases} \tag{4}$$

Proposition 1. *The Jet-Long recipe in equation 4 is a valid* PACK *recipe.*

Proof. For a local pair, the mapped displacement is $0 < i - j < w < W$. Otherwise, both mapped positions lie in $\{0, \ldots, W-1\}$ because $G = \lceil L/W \rceil$, and $\lfloor i/G \rfloor \ge \lfloor j/G \rfloor$ because $i \ge j$, so their difference lies in $[0, W-1]$. Therefore equation 3 holds for every causal pair. □

Similarly, we can show that other methods such as DCA also provide mappings that are valid PACK_{DCA} recipes, as does the three-region recipe PACK_{TR} of Appendix B.

Algorithm 1 Prefill attention with PICKPACK. The local window and the remote key blocks keep separate softmax statistics and are merged exactly by equation 16.

Require: query blocks $\{\mathbf{Q}_{c_i}\}$, key/value blocks $\{\mathbf{K}_{c_j}, \mathbf{V}_{c_j}\}$; PACK recipe (ϕ_q, ϕ_k) with local window w; threshold λ
Ensure: output blocks $\{\mathbf{O}_{c_i}\}$
for each query block c_i **do**
  $(\mathbf{O}^{\mathrm{loc}}_{c_i}, \mathbf{z}^{\mathrm{loc}}_{c_i}) \leftarrow$ dense attention over the local window, at native positions (local window)
  $\mathbf{u} \leftarrow -\infty$, $\boldsymbol{\ell} \leftarrow 0$, $\mathbf{O}^{\mathrm{rem}}_{c_i} \leftarrow 0$
  for each causal key block c_j outside the local window **do**
    $\mathbf{Z} \leftarrow \mathbf{Z}^{\phi_q,\phi_k}_{c_i,c_j}$ (mapped scores, equation 7)
    $\widetilde{\mathbf{m}} \leftarrow \mathrm{rowmax}(\mathbf{Z})$, $\mathbf{u}' \leftarrow \max(\mathbf{u}, \widetilde{\mathbf{m}})$ (tile and running maxima)
    if $\pi_{\lambda;\phi_q,\phi_k}(\mathbf{Z}, \mathbf{u}) = 0$ **then continue** (skip tile, equation 8)
    $\widetilde{\mathbf{P}} \leftarrow \exp(\mathbf{Z} - \mathbf{u}')$
    $\boldsymbol{\ell} \leftarrow e^{\mathbf{u}-\mathbf{u}'}\boldsymbol{\ell} + \mathrm{rowsum}(\widetilde{\mathbf{P}})$, $\mathbf{O}^{\mathrm{rem}}_{c_i} \leftarrow \mathrm{diag}(e^{\mathbf{u}-\mathbf{u}'})\,\mathbf{O}^{\mathrm{rem}}_{c_i} + \widetilde{\mathbf{P}}\mathbf{V}_{c_j}$, $\mathbf{u} \leftarrow \mathbf{u}'$
  end for
  $\mathbf{O}^{\mathrm{rem}}_{c_i} \leftarrow \mathrm{diag}(\boldsymbol{\ell})^{-1}\mathbf{O}^{\mathrm{rem}}_{c_i}$, $\mathbf{z}^{\mathrm{rem}}_{c_i} \leftarrow \mathbf{u} + \log \boldsymbol{\ell}$ (remote key blocks)
  $\mathbf{O}_{c_i} \leftarrow$ merge of $(\mathbf{O}^{\mathrm{loc}}_{c_i}, \mathbf{z}^{\mathrm{loc}}_{c_i})$ and $(\mathbf{O}^{\mathrm{rem}}_{c_i}, \mathbf{z}^{\mathrm{rem}}_{c_i})$ by equation 16
end for
return $\{\mathbf{O}_{c_i}\}$

4.2 PICK ALGORITHM

FlashAttention partitions the queries and keys into blocks $\{\mathbf{Q}_{c_i}\}$ and $\{\mathbf{K}_{c_j}\}$. For each query block c_i, it visits the causal key blocks c_j in order. Fix a PACK recipe (ϕ_q, ϕ_k), and let $\mathbf{X}^{\phi_q,\phi_k}_{c_i,c_j}$ denote the information available while processing the current tile after applying those position maps, such as the mapped attention scores. Let $\mathbf{s}^{\phi_q,\phi_k}_{c_i,c_j}$ summarize the interaction of query block c_i with the previously visited key blocks under the same maps.

Definition 2 (PICK recipe)**.** *A* PICK *recipe parameterized by a* PACK *recipe* (ϕ_q, ϕ_k) *is an online decision rule* π_{ϕ_q,ϕ_k} *together with a state-update rule* $\mathcal{U}_{\phi_q,\phi_k}$*. At tile* (c_i, c_j)*, the recipe produces*

$$d_{c_i,c_j} = \pi_{\phi_q,\phi_k}\left(\mathbf{X}^{\phi_q,\phi_k}_{c_i,c_j}, \mathbf{s}^{\phi_q,\phi_k}_{c_i,c_j}\right) \in \{0,1\}, \tag{5}$$

where $d_{c_i,c_j} = 1$ *keeps the tile and* $d_{c_i,c_j} = 0$ *skips it. It then updates its summary state according to*

$$\mathbf{s}^{\phi_q,\phi_k}_{c_i,c_j+1} = \mathcal{U}_{\phi_q,\phi_k}\left(\mathbf{s}^{\phi_q,\phi_k}_{c_i,c_j}, \mathbf{X}^{\phi_q,\phi_k}_{c_i,c_j}, d_{c_i,c_j}\right). \tag{6}$$

A PICK *recipe is* FlashAttention-compatible *if both rules are evaluated within the existing tile loop, require no additional pass over the queries or keys, and maintain only fixed-size state per query block, independent of the number of previously visited key blocks.*

Thus, a FlashAttention-compatible PICK recipe can decide whether to keep a tile using only information computed under its chosen (ϕ_q, ϕ_k) maps for the current tile and a compact summary of the tiles encountered so far.

We can define many simple PICK recipes and parameterize them with various PACK recipes. In our experiments, we use the PICK recipe based on BLASST (Yuan et al., 2026). We explain it in detail below. Any valid PICK and PACK recipes can be composed together.

4.3 PICKPACK WITH BLASST-STYLE SPARSITY

Fix a PACK recipe (ϕ_q, ϕ_k) and a threshold $\lambda \in (0,1)$. For a query token r in block c_i and a key token t in block c_j, let

$$\mathbf{Z}^{\phi_q,\phi_k}_{c_i,c_j}[r,t] = \frac{\mathbf{q}_r^\top R_{\phi_q(r,t)-\phi_k(r,t)}\mathbf{k}_t}{\sqrt{d}} \tag{7}$$

be its mapped attention score. BLASST instantiates the current-tile information and summary state from Definition 2 as $\mathbf{X}^{\phi_q,\phi_k}_{c_i,c_j} = \mathbf{Z}^{\phi_q,\phi_k}_{c_i,c_j}$ and $\mathbf{s}^{\phi_q,\phi_k}_{c_i,c_j} = \mathbf{u}^{\phi_q,\phi_k}_{c_i,c_j}$, where $u_r^{\phi_q,\phi_k}$ is the largest mapped score among the previously kept key blocks for row r, initialized to $-\infty$. From the current-tile

information, define $\widetilde{m}_r^{\phi_q,\phi_k} = \max_{t\in c_j} \mathbf{X}_{c_i,c_j}^{\phi_q,\phi_k}[r,t]$ as the mapped tile maximum of row r. The decision rule is

$$\pi_{\lambda;\phi_q,\phi_k}\left(\mathbf{X}_{c_i,c_j}^{\phi_q,\phi_k}, \mathbf{s}_{c_i,c_j}^{\phi_q,\phi_k}\right) = \mathbb{I}\left[\max_{r\in c_i}\left(\widetilde{m}_r^{\phi_q,\phi_k} - s_r^{\phi_q,\phi_k}\right) \geq \log\lambda\right] = d_{c_i,c_j}. \tag{8}$$

Its state-update rule is

$$\mathcal{U}_{\lambda;\phi_q,\phi_k}\left(\mathbf{s}_{c_i,c_j}^{\phi_q,\phi_k}, \mathbf{X}_{c_i,c_j}^{\phi_q,\phi_k}, d_{c_i,c_j}\right)_r = \begin{cases} \max\left(s_r^{\phi_q,\phi_k}, \widetilde{m}_r^{\phi_q,\phi_k}\right), & d_{c_i,c_j}=1, \\ s_r^{\phi_q,\phi_k}, & d_{c_i,c_j}=0, \end{cases} = s_{c_i,c_j+1,r}^{\phi_q,\phi_k}. \tag{9}$$

We keep a tile if any row has a tile maximum within $\log(1/\lambda)$ of its running maximum; otherwise, we skip its exponentiation and probability–value product. Algorithm 1 gives the prefill procedure.

Proposition 2. *For any valid* PACK *recipe* (ϕ_q, ϕ_k)*, the pair* $(\pi_{\lambda;\phi_q,\phi_k}, \mathcal{U}_{\lambda;\phi_q,\phi_k})$ *defined in equations 8 and 9 is a FlashAttention-compatible* PICK *recipe parameterized by those maps.*

Proof. The decision uses only the current score tile computed under (ϕ_q, ϕ_k) and the running row maximum maintained under the same maps by the online softmax. Its state consists of one scalar per query row and is therefore independent of the number of previously visited key blocks. Both the decision and state update occur inside the existing tile loop and require no additional pass over the queries or keys. Hence the recipe satisfies the definition of FlashAttention compatibility. □

5 EXPERIMENTS

5.1 SETUP

Models. Our controlled experiments use the Qwen3 base models (Qwen Team, 2025b) with 1.7B, 4B, 8B, and 14B parameters, each trained with a window of $W = 32$K tokens, which we extend to 64K and 128K at inference time without fine-tuning. We follow the setting of Tang et al. (2026), which lets us compare with the baselines they report. We also use the post-trained Qwen3-8B, to study sparsity after instruction tuning. We compare with the recipes that released long-context models ship with on Qwen2.5-7B-Instruct ($W = 32$K, YaRN $\times 4$; Qwen Team, 2024), Qwen2.5-7B-Instruct-1M ($W = 256$K, DCA; Qwen Team, 2025a), and Qwen3.5-4B, 9B, and 27B and Qwen3.8-27B ($W = 256$K, YaRN $\times 4$; Qwen Team, 2026a;b). We also extend Qwen3.5-9B by $16\times$, to four million tokens.

PACK recipes and baselines. We use three PACK recipes: PACK_{JL} (equation 4), PACK_{DCA} (An et al., 2024), and PACK_{TR}, a three-region recipe (Appendix B). For PACK_{JL} and PACK_{DCA} we use the hyperparameters of Tang et al. (2026). On the Qwen3 base models, the baselines are no extension and three training-free methods, Dynamic NTK (emozilla, 2023), YaRN (Peng et al., 2024), and Self-Extend (Jin et al., 2024), whose scores we take from Tang et al. (2026).

Benchmarks. We evaluate on three suites. (i) RULER (Hsieh et al., 2024) tests retrieval and aggregation over synthetic long inputs. We report three means over its tasks: over all thirteen (RULER-13), as in Tang et al. (2026); over four hard tasks, the second multi-key retrieval task (MK2), variable tracking (VT), frequent-word extraction (FWE), and the first question-answering task (QA1) (RULER-4), to compare model sizes; and over passkey, multi-query (MQ), and multi-value (MV) retrieval (Needles), for released models. (ii) LOFT (Lee et al., 2025b) tests retrieval-augmented question answering over real corpora and serves as our application benchmark. We report sub-span exact match, averaged over five of its datasets (NQ, HotpotQA, MuSiQue, QAMPARI, and QUEST), and, for QAMPARI from one to four million tokens, answer coverage over 50 queries. (iii) PG19 (Rae et al., 2019) tests language modeling on books, for which we report perplexity. All generations use greedy decoding. Accuracies are in percent and differences in points.

Implementation and timing. We implement PICKPACK on BLASST's kernel for Hopper GPUs (Yuan et al., 2026), a FlashAttention kernel whose tile loop applies the PICK rule, and we apply the PACK position maps to its scores. Speedup is the time of dense attention divided by that of PICKPACK, measured on H100 GPUs unless noted, with batch size 1, per query or for prefill alone. We calibrate the PICK threshold λ per task and length to 50% sparsity (Appendix A).

Table 1: **PICKPACK keeps or improves the quality of every PACK recipe, and with PACK_{JL} it runs 1.12–1.21× faster.** RULER-13 accuracy, PG19 perplexity, and speedup (per query) for Qwen3-8B base (W=32K), with each recipe run dense and with PICK at 50–59% sparsity; parentheses give the change from dense. †Reported by Tang et al. (2026) in our setting; dashes: not reported. ∼Estimated from the PACK_{JL} timing, since the recipe does not change the kernel's cost.

	64K			128K		
Method	RULER ↑	PPL ↓	Speedup	RULER ↑	PPL ↓	Speedup
No extension	66.06	9.18		0.06	79.43	
Dynamic NTK† (emozilla, 2023)	75.35	9.17		–	10.50	
YaRN† (Peng et al., 2024)	67.53	10.75		–	10.16	
Self-Extend† (Jin et al., 2024)	76.64	8.77		–	8.55	
PACK_{DCA} (An et al., 2024)	69.42	8.70		64.47	8.65	
PICK ∘ PACK_{DCA}	69.29 (−0.13)	8.71	∼1.1×	64.43 (−0.04)	8.66	∼1.2×
PACK_{TR} (App. B)	77.82	8.75		65.71	8.87	
PICK ∘ PACK_{TR}	78.03 (+0.21)	8.74	∼1.1×	67.17 (+1.46)	8.88	∼1.2×
PACK_{JL} (Tang et al., 2026)	84.17	**8.63**		70.00	8.54	
PICK ∘ PACK_{JL}	**84.71** (+0.54)	**8.63**	1.12×	**72.05** (+2.05)	**8.53**	1.21×

5.2 SPARSITY IMPROVES CONTEXT EXTENSION

We first test whether PICK helps regardless of the PACK recipe. We extend Qwen3-8B base from 32K to 64K and 128K with PACK_{JL}, PACK_{DCA}, and PACK_{TR}, each run dense and with PICK (Table 1). We observe that PICK keeps or improves every recipe, by up to 2.05 points, and PICK ∘ PACK_{JL} is the most accurate method at both lengths, 8.07 points above Self-Extend at 64K and 1.12–1.21× faster than dense PACK_{JL}. PICK thus composes with each PACK recipe at no cost in accuracy.

We then test whether this gain holds across model scale. We fix PACK_{JL} and compare dense attention with PICKPACK on Qwen3 base models from 1.7B to 14B at 64K and 128K, using RULER-4, the mean of four hard RULER tasks (Table 2). PICKPACK is more accurate in seven of the eight combinations of size and length, and it is up to 1.1× faster at 64K and about 1.2× faster at 128K. The benefit of sparsity therefore holds across model scale.

Table 2: **PICKPACK is up to 1.2× faster and more accurate at seven of eight.** RULER-4 accuracy (mean of the hard tasks MK2, VT, FWE, and QA1) and speedup (per query) for Qwen3 base models (W=32K) under PACK_{JL}, dense and with PICKPACK at 47–50% sparsity.

	64K				128K			
RULER-4	**1.7B**	**4B**	**8B**	**14B**	**1.7B**	**4B**	**8B**	**14B**
PACK_{JL}	62.98	**76.98**	82.70	84.22	47.30	59.30	71.58	79.45
PICK ∘ PACK_{JL}	**64.05**	75.88	**84.02**	**85.08**	**49.27**	**60.55**	**74.15**	**80.72**
(speedup)	1.00×	1.14×	1.12×	1.11×	1.17×	1.23×	1.21×	1.19×

5.3 PICKPACK OUTPERFORMS RELEASED LONG-CONTEXT RECIPES

To test whether PICKPACK helps models that already ship a long-context recipe, we replace that recipe, YaRN at factor 4 or DCA with dense attention, with PICK ∘ PACK_{JL} on six Qwen2.5, Qwen3.5, and Qwen3.8 models, at 128K to one million tokens (Table 3). PICKPACK is more accurate on every model: by 3.4–7.0 points on Qwen2.5, 5.0–7.6 on Qwen3.5, and 11.0 on Qwen3.8-27B.

PICKPACK is also up to 1.2× faster than the released recipes, which run dense attention. The gain holds on models trained with 256K windows and on Qwen3.5, which mixes linear-attention and softmax-attention layers, so PICKPACK can replace the released recipe at inference without training.

5.4 PICKPACK SCALES TO FOUR MILLION TOKENS

To test how performance scales beyond one million tokens, we extend Qwen3.5-9B, trained with a 256K window, to 1M, 2M, and 4M tokens (4×, 8×, and 16× its window), and compare PICKPACK with YaRN, the recipe it ships with, using a YaRN factor matched to each length (Table 4). YaRN

Table 3: **Against the recipe each released model ships with, PICKPACK is 3.4–11 points more accurate and up to 1.2× faster.** The released recipes are YaRN ×4 or DCA, both with dense attention. Accuracy is on RULER needles (mean of passkey, multi-query, and multi-value) and on LOFT and QAMPARI (subspan exact match; LOFT is the mean of five datasets); speedups are per query, over dense attention. Qwen2.5-7B-Instruct beyond 256K is in Appendix C.

Model (W)	Recipe	Benchmark	L	Released	PICK ∘ PACK$_{\text{JL}}$	Speedup
Qwen2.5-7B-Instruct (32K)	YaRN ×4	Needles	128K	81.7	**86.7**	1.03×
			256K	61.3	**64.7**	1.12×
Qwen2.5-7B-Instruct-1M (256K)	DCA	Needles	512K	84.3	**89.3**	1.17×
			1M	68.7	**75.7**	1.17×
Qwen3.5-4B (256K)	YaRN ×4	LOFT	1M	21.8	**29.4**	1.17×
Qwen3.5-9B (256K)	YaRN ×4	LOFT	1M	28.2	**35.8**	1.09×
Qwen3.5-27B (256K)	YaRN ×4	LOFT	1M	43.4	**48.4**	1.20×
Qwen3.8-27B (256K)	YaRN ×4	QAMPARI	937K	22.0	**33.0**	1.19×

loses accuracy faster than PICKPACK as the context grows: PICKPACK's lead on needle retrieval grows from 2.1 points at 1M to 16.7 at 2M, and on QAMPARI from 3.8 points at 1M to 11.6 at 4M.

The gap is largest on multi-query and multi-value retrieval, where YaRN falls from 56.3 to 37.5 between 1M and 2M while PICKPACK rises from 59.4 to 62.5. PICKPACK is 1.09–1.22× faster at every length. The benefit of PICKPACK grows with context length in both accuracy and speed.

Table 4: **PICKPACK scales better with context length than the released recipe.** Qwen3.5-9B (256K window) at 4×, 8×, and 16× its window: needle retrieval, as the mean of passkey, multi-query, and multi-value (Needles) and of multi-query and multi-value alone (MQ+MV), and LOFT QAMPARI coverage over 50 queries. Speedups are over dense attention at the query level (H200).

	1M				2M				4M	
Method	Needles	MQ+MV	QAMPARI	Speedup	Needles	MQ+MV	QAMPARI	Speedup	QAMPARI	Speedup
YaRN	70.8	56.3	45.8		58.3	37.5	36.0		22.4	
PICK ∘ PACK$_{\text{JL}}$	**72.9**	**59.4**	**49.6**	1.09×	**75.0**	**62.5**	**48.0**	1.20×	**34.0**	1.22×

5.5 HIGHER SPARSITY IS FASTER AND LOWERS PERPLEXITY

To see how speed and quality change with sparsity, we vary the PICK threshold under PACK$_{\text{JL}}$ (Table 5). On Qwen3.5-4B, the speedup over dense attention grows with sparsity: at 512K it rises from 1.00× at low sparsity (8–13%) to 1.15× at high sparsity (50–69%), and at high sparsity it rises from 1.15× at 512K to 1.20× at 2M. Perplexity on Qwen3-8B at 128K falls as sparsity rises, from 8.542 dense to 8.475 at 82%. Higher sparsity thus makes prefill faster without hurting perplexity.

Table 6 repeats the sweep on the post-trained Qwen3-8B, with the thirteen RULER tasks at 64K and 128K. PICKPACK stays competitive with dense attention up to the default target of 50% sparsity (observed 60–61%), where it runs 1.18× faster at 64K and 1.24× faster at 128K. At 25% sparsity it is 0.47 points more accurate than dense attention at 128K, while already 1.11× faster. The default target of 50% sparsity therefore gives most of the speedup at competitive accuracy.

Table 5: **PICKPACK is faster and has lower perplexity at higher sparsity. (a)** Prefill speedup of PICKPACK on Qwen3.5-4B over dense attention (the same kernel at near-zero sparsity) under PACK$_{\text{JL}}$ at three thresholds; the observed sparsity rises with length, so each row gives a range. **(b)** PG19 perplexity of Qwen3-8B base at 128K under PACK$_{\text{JL}}$. Default settings are marked in gray.

sparsity	512K	1M	2M
dense	51.0 s	177.8 s	636.6 s
low (8–13%)	1.00×	1.01×	0.99×
mid (26–36%)	1.04×	1.07×	1.06×
high (50–69%)	**1.15**×	**1.19**×	**1.20**×

(a) **Speedup**. Speedup grows with sparsity, reaching 1.20× at 2M tokens.

sparsity	dense	25%	47%	58%	67%	82%
PG19 ppl ↓	8.542	8.539	8.525	8.529	8.518	**8.475**

(b) **Perplexity**. Perplexity falls as sparsity rises, from 8.542 dense to 8.475 at 82%.

Table 6: **On the post-trained Qwen3-8B, PICKPACK runs 1.18–1.24× faster while remaining competitive with dense on RULER-13.** Per-task RULER-13 under PACK_{JL} at 64K and 128K; parentheses give the change in average from dense; speedups are end-to-end prefill. S = single needle, MK = multi-key, MV/MQ = multi-value/multi-query, VT = variable tracking, CWE/FWE = common/frequent word extraction, QA = question answering. Default settings are marked in gray.

L	Sparsity	S1	S2	S3	MK1	MK2	MK3	MV	MQ	VT	CWE	FWE	QA1	QA2	Avg	Speedup
64K	0% (dense)	100	100	100	98	96	**96**	**99.0**	99.0	**100.0**	**65.2**	**85.3**	52	**52**	**87.89**	
	26%	100	100	100	98	96	94	**99.0**	99.0	**100.0**	64.0	84.0	**54**	**52**	87.69 (−0.19)	1.11×
	61%	100	100	100	98	**98**	86	98.0	99.0	**100.0**	61.0	83.3	52	50	86.56 (−1.32)	1.18×
	77%	100	100	100	98	86	70	**99.0**	**99.5**	99.6	62.8	83.3	48	46	84.02 (−3.87)	**1.24**×
128K	0% (dense)	100	100	100	98	**82**	56	**97.5**	97.5	**100.0**	32.8	79.3	48	**38**	79.16	
	25%	100	100	100	98	**82**	**60**	**97.5**	97.5	99.6	**34.0**	**80.7**	48	**38**	**79.64** (+0.47)	1.11×
	60%	100	100	100	98	76	50	97.0	97.5	99.6	28.2	78.0	48	36	77.56 (−1.60)	1.24×
	78%	100	100	100	98	74	38	97.0	**98.0**	98.4	26.4	78.0	48	**38**	76.45 (−2.72)	**1.33**×

5.6 SPARSITY REDUCES THE EXTENSION ERROR

We test the hypothesis of Section 3 with the in-window simulation. On Qwen3-8B base ($W = 32$K), we apply PACK_{JL} built for an apparent window of 16K or 8K ($G = 2$ or 4) to 32K-token inputs, with dense attention and with PICK at 50% sparsity (Table 7). PICK lowers the relative error of the attention output from 0.156 to 0.122 at $G = 2$ and from 0.241 to 0.216 at $G = 4$, by 21% and 11%. Sparsity brings the extended attention closer to what the model computes inside its trained window.

Table 7: **PICK lowers the simulated extension error of PACK_{JL}.** Relative error of the attention output (lower is better) for PACK_{JL} simulating a $G\times$ extension, dense and with PICK at 50% sparsity.

$\check{W}$	G	PACK_{JL}	PICK ∘ PACK_{JL}
16K	2	0.156	**0.122**
8K	4	0.241	**0.216**

6 CONCLUSION

We introduce PICKPACK, which performs repositioning and sparse attention in a single forward pass: PACK places every key inside the trained window with a training-free position map, and PICK computes only the remote blocks whose scores under that map come close to the running maximum. On seven of eight Qwen3 model sizes and context lengths, PICKPACK is more accurate than dense attention under the same map while being up to 1.2× faster. Against the recipes that released Qwen2.5, Qwen3.5, and Qwen3.8 models ship with, it is 3.4–11 points more accurate and up to 1.2× faster from 128K to one million tokens, and 11.6 points more accurate at four million tokens, 16× the trained window of Qwen3.5-9B. Sparse attention thus serves length generalization and prefill cost. Extending PICK to decoding, and choosing positions for the kept blocks, are next steps.

Limitations. PICKPACK sparsifies prefill only, so decoding still attends to the full cache. We fit the threshold λ for each task and length on the evaluation prompts; the fit uses no labels, but the fitted threshold varies across tasks (Table 8), so a single threshold per model would not hold the same sparsity everywhere. In hybrid models, PICKPACK sparsifies only the softmax attention layers. Most of our results use one sparsity level, about 50%, and §5.5 varies it from 8% to 82%.

AI USE STATEMENT

We used large language model assistants to help draft and edit the text of this paper and to check it for clarity and consistency. The authors designed the method and the experiments, and reviewed and take responsibility for all text, technical claims, and citations.

ETHICS STATEMENT

PICKPACK changes only how existing models run inference: it lowers the cost of long-context prefill and does not change any model's weights or training data. Our experiments use publicly available models and benchmarks, and we do not identify ethical considerations specific to this work beyond those of the underlying models.

Reproducibility statement

§5.1 gives the models, benchmarks, position-map settings, and timing protocol, and Appendix A the threshold calibration. Our implementation builds on Jet-Long's position map (Tang et al., 2026) and the released BLASST kernel (Yuan et al., 2026). We will release our code and calibration scripts.

References

Chenxin An, Fei Huang, Jun Zhang, Shansan Gong, Xipeng Qiu, Chang Zhou, and Lingpeng Kong. Training-free long-context scaling of large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 1493–1510. PMLR, 2024. URL https://proceedings.mlr.press/v235/an24b.html.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020. URL https://arxiv.org/abs/2004.05150.

bloc97. NTK-Aware scaled RoPE allows LLaMA models to have extended (8k+) context size without any fine-tuning and minimal perplexity degradation. Reddit r/LocalLLaMA, 2023. URL https://www.reddit.com/r/LocalLLaMA/comments/14lz7j5/ntkaware_scaled_rope_allows_llama_models_to_have/.

Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. Extending context window of large language models via positional interpolation. *arXiv preprint arXiv:2306.15595*, 2023. URL https://arxiv.org/abs/2306.15595.

Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. LongLoRA: Efficient fine-tuning of long-context large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2309.12307.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019. URL https://arxiv.org/abs/1904.10509.

Phat Thanh Dang, Saahil Thoppay, Wang Yang, Qifan Wang, Vipin Chaudhary, and Xiaotian Han. SELF: Self-extend the context length with logistic growth function. *arXiv preprint arXiv:2505.17296*, 2025. URL https://arxiv.org/abs/2505.17296.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems*, volume 35, pp. 16344–16359, 2022. URL https://arxiv.org/abs/2205.14135.

DeepSeek-AI. DeepSeek-V4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026. URL https://arxiv.org/abs/2606.19348.

Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang Xu, Fan Yang, and Mao Yang. LongRoPE: Extending LLM context window beyond 2 million tokens. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 11091–11104. PMLR, 2024. URL https://proceedings.mlr.press/v235/ding24i.html.

emozilla. Dynamically scaled RoPE further increases performance of long context LLaMA with zero fine-tuning. Reddit r/LocalLLaMA, 2023. URL https://www.reddit.com/r/LocalLLaMA/comments/14mrgpr/dynamically_scaled_rope_further_increases/.

Yao Fu, Rameswar Panda, Xinyao Niu, Xiang Yue, Hannaneh Hajishirzi, Yoon Kim, and Hao Peng. Data engineering for scaling language models to 128K context. In *Proceedings of the 41st International Conference on Machine Learning*, 2024. URL https://arxiv.org/abs/2402.10171.

Noah Golowich, Samy Jelassi, David Brandfonbrener, Sham M. Kakade, and Eran Malach. The role of sparsity for length generalization in transformers. *arXiv preprint arXiv:2502.16792*, 2025. URL https://arxiv.org/abs/2502.16792.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling*, 2024. URL https://arxiv.org/abs/2312.00752.

Chi Han, Qifan Wang, Hao Peng, Wenhan Xiong, Yu Chen, Heng Ji, and Sinong Wang. LM-Infinite: Zero-shot extreme length generalization for large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics*, 2024. URL https://arxiv.org/abs/2308.16137.

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? In *Conference on Language Modeling*, 2024. URL https://arxiv.org/abs/2404.06654.

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. URL https://openreview.net/forum?id=fPBACAbqSN.

Hongye Jin, Xiaotian Han, Jingfeng Yang, Zhimeng Jiang, Zirui Liu, Chia-Yuan Chang, Huiyuan Chen, and Xia Hu. LLM maybe LongLM: SelfExtend LLM context window without tuning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 22099–22114. PMLR, 2024. URL https://proceedings.mlr.press/v235/jin24b.html.

Sahil Joshi, Prithvi Dixit, Agniva Chowdhury, Anshumali Shrivastava, Joseph E. Gonzalez, Ion Stoica, Kumar Krishna Agrawal, and Aditya Desai. Inference time context sparsity: Illusion or opportunity? *arXiv preprint arXiv:2605.24168*, 2026. URL https://arxiv.org/abs/2605.24168.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *Proceedings of the 37th International Conference on Machine Learning*, 2020. URL https://arxiv.org/abs/2006.16236.

Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. The impact of positional encoding on length generalization in transformers. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2305.19466.

Kimi Team. Kimi K3: Open frontier intelligence, July 2026. URL https://github.com/MoonshotAI/Kimi-K3/blob/main/k3_tech_report.pdf.

Xunhao Lai, Jianqiao Lu, Yao Luo, Yiyuan Ma, and Xun Zhou. FlexPrefill: A context-aware sparse attention mechanism for efficient long-sequence inference. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=OfjIlbelrT.

Heejun Lee, Geon Park, Jaduk Suh, and Sung Ju Hwang. InfiniteHiP: Extending language model context up to 3 million tokens on a single GPU. *arXiv preprint arXiv:2502.08910*, 2025a. URL https://arxiv.org/abs/2502.08910.

Jinhyuk Lee et al. Can long-context language models subsume retrieval, RAG, SQL, and more? In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://arxiv.org/abs/2406.13121.

Yan Li, Tianyi Zhang, Zechuan Li, and Caren Han. A training-free length extrapolation approach for LLMs: Greedy attention logit interpolation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 8773–8793, 2025. URL https://arxiv.org/abs/2502.02659.

Yuhong Li, Yingbing Huang, Bowen Yang, et al. SnapKV: LLM knows what you are looking for before generation. In *Advances in Neural Information Processing Systems*, 2024. URL https://arxiv.org/abs/2404.14469.

Xiaoran Liu, Ruixiao Li, Qipeng Guo, Zhigeng Liu, Yuerong Song, Kai Lv, Hang Yan, Linlin Li, Qun Liu, and Xipeng Qiu. ReAttention: Training-free infinite context with finite attention scope. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2407.15176.

Enzhe Lu, Zhejun Jiang, Jingyuan Liu, et al. MoBA: Mixture of block attention for long-context LLMs. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2502.13189.

Yi Lu, Xin Zhou, Wei He, Jun Zhao, Tao Ji, Tao Gui, Qi Zhang, and Xuanjing Huang. LongHeads: Multi-head attention is secretly a long context processor. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 7136–7148, 2024. URL https://arxiv.org/abs/2402.10685.

Ken M. Nakanishi. Scalable-softmax is superior for attention. *arXiv preprint arXiv:2501.19399*, 2025. URL https://arxiv.org/abs/2501.19399.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. YaRN: Efficient context window extension of large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=wHBfxhZu1u.

Ofir Press, Noah A. Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2108.12409.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. URL https://arxiv.org/abs/2412.15115.

Qwen Team. Qwen2.5-1M technical report. *arXiv preprint arXiv:2501.15383*, 2025a. URL https://arxiv.org/abs/2501.15383.

Qwen Team. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025b. URL https://arxiv.org/abs/2505.09388.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026a. URL https://qwen.ai/blog?id=qwen3.5.

Qwen Team. Qwen3.8-Max: A new bar for coding and cowork, August 2026b. URL https://qwen.ai/blog?id=qwen3.8.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, and Timothy P. Lillicrap. Compressive transformers for long-range sequence modelling. *arXiv preprint arXiv:1911.05507*, 2019. URL https://arxiv.org/abs/1911.05507.

Stefano Rando, Luca Romani, Alessio Sampieri, Luca Franco, John Yang, Yuta Kyuragi, Fabio Galasso, and Tatsunori Hashimoto. LongCodeBench: Evaluating coding LLMs at 1M context windows. In *Second Conference on Language Modeling*, 2025. URL https://arxiv.org/abs/2505.07897.

Jianlin Su. Rectified rotary position embeddings. https://github.com/bojone/rerope, 2023. Blog post: https://kexue.fm/archives/9708.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021. URL https://arxiv.org/abs/2104.09864.

Haozhan Tang, Zerui Wang, Yuxian Gu, Song Han, and Han Cai. Jet-Long: Efficient long-context extension with dynamic bifocal RoPE. *arXiv preprint arXiv:2607.07740*, 2026. URL https://arxiv.org/abs/2607.07740.

Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, and Song Han. Quest: Query-aware sparsity for efficient long-context LLM inference. In *Proceedings of the 41st International Conference on Machine Learning*, 2024. URL https://arxiv.org/abs/2406.10774.

Qingyuan Tian, Wenhong Zhu, Xiaoran Liu, Xiaofeng Wang, and Rui Wang. MrRoPE: Mixed-radix rotary position embedding. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2601.22181.

Pavlo Vasylenko, Hugo Pitorro, André F. T. Martins, and Marcos Treviso. Long-context generalization with sparse attention. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2506.16640.

Chaojun Xiao, Pengle Zhang, Xu Han, Guangxuan Xiao, Yankai Lin, Zhengyan Zhang, Zhiyuan Liu, and Maosong Sun. InfLLM: Training-free long-context extrapolation for LLMs with an efficient context memory. In *Advances in Neural Information Processing Systems*, volume 37, 2024a. URL https://arxiv.org/abs/2402.04617.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations*, 2024b. URL https://openreview.net/forum?id=NG7sS51zVF.

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 *The Thirteenth International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.10819.

Wenhan Xiong, Jingyu Liu, Igor Molybog, et al. Effective long-context scaling of foundation models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics*, 2024. URL https://arxiv.org/abs/2309.16039.

Ruyi Xu, Guangxuan Xiao, Haofeng Huang, Junxian Guo, and Song Han. XAttention: Block sparse attention with antidiagonal scoring. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 69819–69831. PMLR, 2025a. URL https://proceedings.mlr.press/v267/xu25ag.html.

Xinhao Xu, Jiaxin Li, Hui Chen, Zijia Lin, Jungong Han, and Guiguang Ding. Extending LLM context window with adaptive grouped positional encoding: A training-free method. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 573–587, 2025b. URL https://aclanthology.org/2025.acl-long.28/.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.03629.

Jiayi Yuan, Cameron Shinn, Kai Xu, Jingze Cui, George Klimiashvili, Guangxuan Xiao, Perkz Zheng, Bo Li, Yuxin Zhou, Zhouhai Ye, Weijie You, Tian Zheng, Dominic Brown, Pengbo Wang, Markus Hoehnerbach, Richard Cai, Julien Demouth, John D. Owens, Xia Hu, Song Han, Timmy Liu, and Huizi Mao. BLASST: Dynamic BLocked attention sparsity via softmax thresholding. In *Proceedings of Machine Learning and Systems*, volume 8, 2026. URL https://arxiv.org/abs/2512.12087.

Jingyang Yuan, Huazuo Gao, Damai Dai, et al. Native sparse attention: Hardware-aligned and natively trainable sparse attention. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23078–23097, 2025. URL https://arxiv.org/abs/2502.11089.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, et al. Big bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://arxiv.org/abs/2007.14062.

Z.ai. GLM-5.2: Built for long-horizon tasks, June 2026. URL https://huggingface.co/blog/zai-org/glm-52-blog.

Jintao Zhang, Chendong Xiang, Haofeng Huang, Jia Wei, Haocheng Xi, Jun Zhu, and Jianfei Chen. SpargeAttention: Accurate and training-free sparse attention accelerating any model inference. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025. URL https://arxiv.org/abs/2502.18137.

Sikui Zhang, Guangze Gao, Ziyun Gan, Chunfeng Yuan, Zefeng Lin, Houwen Peng, Bing Li, and Weiming Hu. LaMPE: Length-aware multi-grained positional encoding for adaptive long-context scaling without training. In *Findings of the Association for Computational Linguistics: ACL 2026*, 2026. URL https://arxiv.org/abs/2508.02308.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, et al. H_2O: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2306.14048.

Qianchao Zhu, Jiangfei Duan, Chang Chen, Siran Liu, Guanyu Feng, Xin Lv, Xiao Chuanfu, Dahua Lin, and Chao Yang. SampleAttention: Near-lossless acceleration of long context LLM inference with adaptive structured sparse attention. *arXiv preprint arXiv:2406.15486*, 2024. URL https://arxiv.org/abs/2406.15486.

A Threshold Calibration

BLASST sets the threshold as $\lambda = a/L$, with a factor $a = \alpha e^{\beta S}$ for a target fraction S of skipped tiles, and fits α and β once per model on RULER sequences from 4K to 64K (Yuan et al., 2026). We instead calibrate a for each task and length on the evaluation prompts, which needs no labels because the target is a sparsity. Table 8 lists the factors that held 50% sparsity on Qwen2.5-7B. They vary across tasks, from 60 to 381 at 1M, so a factor fitted once would not hold the same sparsity everywhere.

Table 8: **BLASST's threshold factor, calibrated to 50% sparsity on Qwen2.5-7B.** Each cell is the factor that reached a tile sparsity of 0.495–0.507 for that task, length, and checkpoint within one or two attempts; multi-value reused the multi-query factor.

Checkpoint	Task	128K	256K	512K	1M
Qwen2.5-7B-Instruct	Passkey	270.3	154.7	88.9	59.6
	Multi-query, multi-value	325.1	373.3	318.1	380.7
Qwen2.5-7B-Instruct-1M	Passkey	—	—	86.0	71.0
	Multi-query, multi-value	—	—	167.4	146.2

B The Three-Region Recipe Pack_{TR}

This appendix defines TightRoPE, the three-region Pack recipe we denote Pack_{TR} (Table 1), and the merge of its regions.

B.1 Regions and Scale

With zero-based positions, trained window W, protected sink count n_s, protected local width ℓ, and middle placement-chunk size C, the causal keys of a query $i \geq W$ partition as

$$\mathcal{K}_i^{\text{sink}} = \{0, \ldots, n_s - 1\}, \qquad \mathcal{K}_i^{\text{mid}} = \{n_s, \ldots, i - \ell\}, \qquad \mathcal{K}_i^{\text{loc}} = \{i - \ell + 1, \ldots, i\}, \tag{10}$$

and the three-region recipe assigns query and key positions

$$\big(\phi_q(i,j), \phi_k(i,j)\big) = \begin{cases} (i, j), & i < W, \\ (W-1,\, j), & i \geq W,\ j \in \mathcal{K}_i^{\text{sink}}, \\ (W-1,\, p_L(j)), & i \geq W,\ j \in \mathcal{K}_i^{\text{mid}}, \\ (i, j), & i \geq W,\ j \in \mathcal{K}_i^{\text{loc}}. \end{cases} \tag{11}$$

The displacement $\phi_q - \phi_k$ then lies in $[W - n_s,\, W - 1]$ for sinks, $[\ell,\, W - n_s - 1]$ for middle keys, and $[0,\, \ell - 1]$ for local keys, which together tile exactly the trained range of equation 3. The recipe also multiplies queries and keys by a magnitude factor

$$m_i = \begin{cases} 1 + \alpha \log(L_{\text{target}}/W), & i \geq W \text{ and } L_{\text{target}} > W, \\ 1, & \text{otherwise}, \end{cases} \tag{12}$$

with target length L_{target} and constant α, so the logits of row i scale by m_i^2; rows within the trained window keep $m_i = 1$ and are unchanged.

B.2 Low-Discrepancy Placement

The placement map p_L of equation 11 uses the number of chunk-start slots and the number of middle chunks,

$$W_C = W - n_s - \ell - C + 1, \qquad N_L = \left\lceil \frac{L - n_s - \ell}{C} \right\rceil, \tag{13}$$

the Weyl sequence $\xi_c = \{\frac{1}{2} + c\gamma\}$ for $c \in \{0, \ldots, N_L - 1\}$ with $\gamma = (\sqrt{5} - 1)/2$ (braces denote the fractional part), and its order statistics $\xi_{(c)}$. A middle key j lies in chunk $c = \lfloor (j - n_s)/C \rfloor$ and is placed at

$$p_L(j) = n_s + \lfloor W_C\, \xi_{(c)} \rfloor + (j - n_s - cC), \tag{14}$$

which keeps it inside $[n_s, W - \ell - 1]$; chunk ranges overlap once $N_L > W_C$.

B.3 Region Merge

Each region $r \in \{\text{sink}, \text{mid}, \text{loc}\}$ contributes the log-normalizer

$$z_i^{(r)} = \log \sum_{j \in \mathcal{A}_i^{(r)}} \exp\bigl(a_{ij}^{(r)}\bigr) \tag{15}$$

over its kept keys $\mathcal{A}_i^{(r)}$ (the full sink or local group, or the middle keys in kept tiles), and with region outputs $o_i^{(r)}$ the merge is

$$o_i = \frac{\sum_r \exp(z_i^{(r)})\, o_i^{(r)}}{\sum_r \exp(z_i^{(r)})}, \tag{16}$$

the softmax over the union of the kept keys.

C Needle Retrieval on Qwen2.5-7B

Table 9 gives the per-task needle results behind the Qwen2.5-7B rows of Table 3, from 128K to one million tokens.

Table 9: **RULER needle retrieval from 128K to 1M, Qwen2.5-7B.** String-match accuracy (0–100, ↑) over 20, 20, 15, and 10 samples per task at 128K, 256K, 512K, and 1M. PICK ∘ PACK$_{\text{JL}}$ runs at 50% sparsity and differs from the dense PACK$_{\text{JL}}$ row above it only in the tiles it skips. The 1M checkpoint is trained to 256K, and its PACK$_{\text{JL}}$ rows are not run inside that window (—). At 256K, dense PACK$_{\text{JL}}$ fails multi-query and multi-value retrieval, and PICK restores them.

	Passkey ↑				Multi-query ↑				Multi-value ↑			
Method	128K	256K	512K	1M	128K	256K	512K	1M	128K	256K	512K	1M
Qwen2.5-7B-Instruct (W=32K)												
YaRN ×4	100	100	100	0	79	45	5	0	66	39	2	0
DCA	100	100	80	70	88	62	2	0	66	35	0	0
PACK$_{\text{JL}}$	100	90	80	60	85	0	0	0	68	0	0	0
PICK ∘ PACK$_{\text{JL}}$	100	90	67	40	88	60	0	0	72	44	0	0
Qwen2.5-7B-Instruct-1M (trained to 256K)												
No extension	100	100	100	0	100	97	45	0	85	76	30	0
DCA	100	100	100	100	100	97	88	68	89	78	65	38
PACK$_{\text{JL}}$	—	—	93	100	—	—	95	82	—	—	68	47
PICK ∘ PACK$_{\text{JL}}$	—	—	100	100	—	—	95	82	—	—	73	45