ALLEVIATING FORGETFULNESS OF LINEAR ATTENTION BY HYBRID SPARSE ATTENTION AND CONTEXTUALIZED LEARNABLE TOKEN EVICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

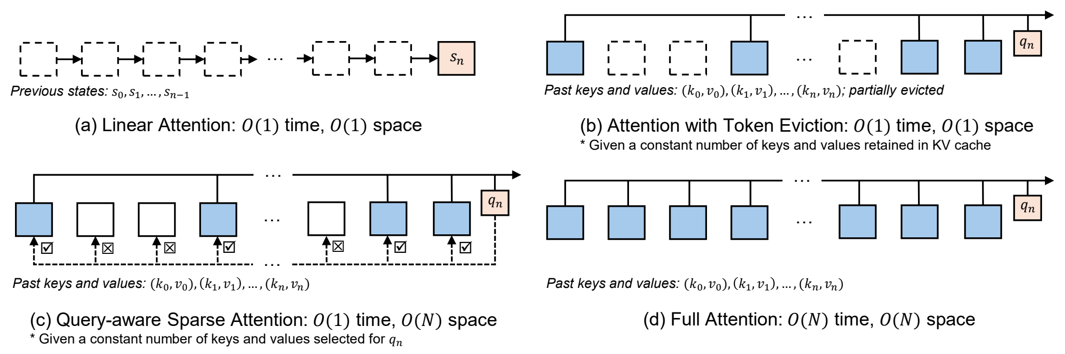
Linear-attention models that compress the entire input sequence into a fixed-size recurrent state offer an efficient alternative to Transformers, but their finite memory induces forgetfulness that harms retrieval-intensive tasks. To mitigate the issue, we explore a series of hybrid models that restore direct access to past tokens. We interleave token mixers with intermediate time and space complexity between linear and full attention, including sparse attention with token eviction, and the query-aware native sparse attention. Particularly, we propose a novel learnable token eviction approach. Combined with sliding-window attention, an end-to-end trainable lightweight CNN aggregates information from both past and future adjacent tokens to adaptively retain a limited set of critical KV-pairs per head, maintaining linear attention's constant time and space complexity. Efficient Triton kernels for the sparse attention mechanisms are provided. Empirical evaluations on retrieval-intensive benchmarks support the effectiveness of our approaches.

1 INTRODUCTION

The recent emergence of large language models (LLMs) highlights the strong capability and efficiency of the transformer architecture (Vaswani et al., 2017). By explicitly scoring pairwise token relevance, transformers excel in capturing long-term dependence and retrieving information from the distant past. Furthermore, transformers can be efficiently trained in parallel on an unprecedented scale of model and data capacity on modern GPUs. All these advantages enable transformers to supplant recurrent neural networks (RNNs) in the field of natural language processing. However, the blessing of pairwise attention is also a curse: compute time scales linearly with past context length per token (i.e. quadratically for the whole sequence), and efficient decoding requires a key-value (KV) cache that grows linearly in GPU memory; both of which drive computational costs and hardware demands. In response, novel hardware-efficient recurrent or linear attention token mixers such as Mamba (Gu & Dao, 2024) and DeltaNet (Yang et al., 2024d) have gained traction. Like early RNNs, these models compress the entire history into a fixed-size memory recurrently updated for each new token (illustrated in Figure 1 (a)), yielding $O(1)$ time and space per step, while remaining competitive with transformers on many language tasks. However, this also comes with a cost: a variable-length history can never be losslessly compressed into a fixed-size state. As time elapses, information from distant tokens inevitably decays, resulting in diminishing performance on retrieval-intensive tasks compared to standard transformers, even at modest context lengths (Wen et al., 2025; Jelassi et al., 2024; Arora et al., 2024a;b).

Various methods have been proposed to mitigate this forgetfulness. A prominent direction is to improve the update rule itself to increase memory expressiveness and update selectivity (Gu & Dao, 2024; Yang et al., 2024b; Du et al., 2025), without altering the recurrent nature. Others adopt hybrid models by interleaving full and linear attention layers (Lenz et al., 2025; Waleffe et al., 2024; MiniMax et al., 2025). While this combination or compromise of two worlds allows more effective direct retrieval of past tokens, the complexity issue resurfaces. This motivates a natural question: Can we directly access past tokens for better retrieval without much sacrifice in computational complexity?

Other than a complete fall back from linear to standard attention, there exists a series of token mixers with time and space complexity between the two extremes, as exemplified in Figure 1. Among them



Previous states: $s_0, s_1, \ldots, s_{n-1}$
s_n
(a) Linear Attention: $O(1)$ time, $O(1)$ space
Past keys and values: $(k_0, v_0), (k_1, v_1), \ldots, (k_n, v_n)$; partially evicted
q_n
(b) Attention with Token Eviction: $O(1)$ time, $O(1)$ space
* Given a constant number of keys and values retained in KV cache
Past keys and values: $(k_0, v_0), (k_1, v_1), \ldots, (k_n, v_n)$
(c) Query-aware Sparse Attention: $O(1)$ time, $O(N)$ space
* Given a constant number of keys and values selected for q_n
Past keys and values: $(k_0, v_0), (k_1, v_1), \ldots, (k_n, v_n)$
(d) Full Attention: $O(N)$ time, $O(N)$ space

Figure 1: Hierarchy of token mixers across different time and space complexity per step. More complete and direct access to past tokens entails higher time and space costs.

recent query-aware sparse attention mechanisms such as MoBA (Lu et al., 2025) and Native Sparse Attention (NSA) (Yuan et al., 2025) are promising options. These models introduce a lightweight "probing" step that first compares the current query with blocks of previous KVs. Only a small set of possible most relevant past tokens are selected for the actual attention computation, so as to approximate full attention without $O(N)$-time computation. This motivates our proposal of a hybrid **laNSA** model that interleaves **l**inear **a**ttention and **NSA** layers, aiming at directly accessing past information while largely preserving the favorable time complexity.

However, these methods still require accessing all past tokens and thus an $O(N)$-size KV cache, undermining linear attention's constant-space appeal; block-wise probing also limits granularity and adds overheads. Ideally, one could determine or *forecast* if the current token will be relevant for future queries using certain relevant criteria, and proactively discard less relevant tokens in advance, retaining only $O(1)$ tokens in GPU memory (Figure 1 (b)). This has inspired a thread of research from the earlier sparse and local attention (Beltagy et al., 2020) to recent token eviction rules driven by accumulated attention scores (Zhang et al., 2023; Chen et al., 2024b). This approach yet leads back to forgetfulness: despite the direct access to the distant past, once a token is dropped, it is no longer available to any future query. An accurate eviction rule is henceforth of utmost importance. Most prior work relies on fixed, handcrafted heuristics that are not input-adaptive. By contrast, we introduce learnable token eviction (LTE), which leverages a novel lightweight convolution neural network (CNN) module that combines sliding-window attention (SWA) to predict the individual KV importance per head from both the past and future adjacent context. Trained end-to-end under sparsity regularization, LTE allows each sparse attention layer to retain the most relevant KVs under a learned per-head cache budget. We then build our **laLTE** model with interleaved linear and LTE-sparse attention layers. We further approach actual speedup with a dedicated KV cache layout, fused Triton kernels for SWA+KV-sparse attention computation and caching, and a lazy scoring scheme. In this way, laLTE retains both the constant time and space complexity of linear attention, while aimed at best approaching the performance of laNSA and standard transformer models.

We conduct a series of empirical studies to evaluate our approaches at the scale of 0.4B and 1.4B models on several short-context and long-context/retrieval-intensive benchmarks. Both the laLTE and laNSA models outperform baselines including hybrids with SWA or heuristic eviction, and approach the performance of full transformers. To summarize, our contributions are threefold: 1) We propose a novel contextualized learnable token eviction mechanism; 2) We build hybrid laNSA and laLTE architectures with efficient implementation; 3) We examine the performance of hybrid linear attention models across the token mixer hierarchy. Our source code and models are available at [ANONYMIZED].

2 Background and related work

2.1 Sparse attention

Sparsity patterns A substantial body of work is aimed at reducing the $O(N^2)$ time complexity of transformer stemmed from the attention computation $A(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d}})\boldsymbol{V}$ or $A(\boldsymbol{q}_i, \boldsymbol{K}, \boldsymbol{V})$ for each query vector q_i. The $N \times d$ matrices $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$ represent the sequence of N d-dimensional *query*, *key*, and *value* vectors. A common approach is sparse attention, i.e., limiting the attention computation of each query to a small set of KV pairs $\boldsymbol{K}_{\mathcal{I}}$, $\boldsymbol{V}_{\mathcal{I}}$ at indices $\mathcal{I} \subseteq \{1, 2, ..., n\}$ (or $\mathcal{I}_i$ per q_i) of tokens believed to be important, reducing the overall complexity to $O(N|\mathcal{I}|)$. $\mathcal{I}_i$ can be simply determined by fixed patterns, e.g., by selecting special tokens (Child et al., 2019; Ge et al., 2023) or specific positions. A prominent example is sliding-window attention (SWA), that selects the most recent w tokens and discards KVs that fall outside the window during decoding. SWA is often combined with "global tokens", e.g., the first s tokens of the input (Beltagy et al., 2020; Ainslie et al., 2020; Zaheer et al., 2020), a.k.a. attention sink (Xiao et al., 2023). The resulting A-shaped pattern $\mathcal{I}_i = \{j : j \leq s \vee i - w < j \leq i\}$ helps a pretrained standard transformer to generalize to longer contexts (Han et al., 2024). Despite simplicity and efficiency, fixed patterns are nevertheless agnostic to the encoded inputs and show suboptimal performance.

Token eviction Many works focus on more flexible strategies to identify tokens believed to be important for future attention, discarding less important k_j, v_j from the KV cache or evict them from any future $\mathcal{I}_i, i > j$ to maintain a token budget $|\mathcal{I}_j| \leq S$. Because high historical attention score (i.e. more contribution to past outputs) often correlates with future importance, a common strategy retains tokens with top-K accumulated attention scores in the whole history (Wang et al., 2021; Zhang et al., 2023; Jo & Shin, 2024), in a window (Liu et al., 2023; Zhao et al., 2024; Oren et al., 2024; Li et al., 2024), or to specific query tokens (Chen et al., 2024b; Kim et al., 2024). Notably, attention patterns vary by layer and head, which motivates per-head cache budget, possibly with global budget allocation (Feng et al., 2025; Zhou et al., 2025), more budget in lower layers (Yang et al., 2024a; Cai et al., 2024), or to heads with more dispersed attention (Qin et al., 2024b). All these methods are primarily based on heuristics on attention scores and training-free. However, many tokens are *typically* important (e.g., infobox in a Wikipedia article) but without high local importance and attention score. Also, attention scores are not directly available from standard Flash Attention (Dao et al., 2022); extracting them adds implementation complexity and overheads.

A more flexible and potent alternative is to predict *typically* important tokens, possibly for each head, with a learnable module, a direction explored less extensively. Zhang et al. (2024) train a separate RNN to predict token attention scores, without considering any interaction with the base model. Other works co-train LTE together with the whole model. An early attempt is ColT5, which selects tokens with a lightweight MLP router (Ainslie et al., 2023b), taking each token as input and hence not contexualized. Anagnostidis et al. (2023) also train a layerwise bias on attention scores for token importance. Zeng et al. (2024) train a separate attention layer for LTE. Although being fully contextualized, this involves heavy quadratic computation and is limited to prefilling with global context. More aggressive methods skip entire attention layers for tokens selected by per-token routers (Raposo et al., 2024; Jiang et al., 2024b; Sharma et al., 2025), precluding any interaction with skipped tokens. Very recently, several concurrent works explore LTE with per-token single-layer projection: Łańcucki et al. (2025) fine-tune LLMs with a global compression ratio constraint, which allows many layers to remain uncompressed and contradicts our $O(1)$ complexity goal. Piękos et al. (2025) train small scale LMs under predetermined cache budget, without considering layer and head variations. Shi et al. (2025) predict importance scores that are then multiplied with attention scores, which deviates from standard attention computation and leads to extra implementation complexity. As far as we are aware, this paper is the first to introduce a contextualized lightweight LTE variant.

Query-aware selection The token eviction mechanisms above attempt to identify tokens relevant to future queries without actually knowing them; this kind of forecast is inherently fallible. If we relax the space constraint and retain all past KVs, we can select the $\mathcal{I}_i = \text{select}(\boldsymbol{q}_i, \boldsymbol{K}, \boldsymbol{V})$ in a $\boldsymbol{q}_i$-aware manner, via an $O(N)$ but lightweight probing step before the actual attention computation. Early selection methods utilize online hashing and clustering (Kitaev et al., 2020; Roy et al., 2021), while many recent approaches highlight GPU-efficient blockwise approximation, exemplified by Mixture of Block Attention (MoBA) (Lu et al., 2025): input sequences are split into M-sized blocks, and keys in each block B_m are compressed into a block-level key $\boldsymbol{k}_m^B$ by

$$\boldsymbol{k}_m^B = \phi(\boldsymbol{k}_j), j \in B_m \tag{1}$$

where ϕ denotes mean-pooling. Then $\text{score}(\boldsymbol{q}, B_m) = \langle \boldsymbol{q}, \boldsymbol{k}_m^B \rangle$ represents an approximated attention score at block level, and KVs in $\mathcal{I}_i^{(slc)} = \cup B_m$ for $m \in$ top-K$(\text{score}(\boldsymbol{q}_i, B_m))$ are used to produce selective attention results A_{slc}. Intuitively, tokens in blocks with high attention score between $\boldsymbol{q}_i$ and mean-pooled keys are selected, and the whole network is then trained end-to-end.

In this way, the probing step can be carried out by only N/M attention computations. Given the probing results $\mathcal{I}_i^{(slc)}$, with only a small constant number MK of tokens attended, the actual attention time complexity is kept constant. We further adopt the stronger and more sophisticated Native Sparse Attention (NSA) (Yuan et al., 2025). NSA uses a learned MLP as ϕ to produce compressed keys $\boldsymbol{k}_m^B$ and values $\boldsymbol{v}_m^B$. In addition to A_{slc}, NSA further computes a compressive attention branch $A_{cmp} = A(\boldsymbol{q}_i, \boldsymbol{k}_m^B, \boldsymbol{v}_m^B)$ directly using the compressed KVs, as well as an SWA branch A_{swa}. It then predicts three gating weights $g_{\{slc,cmp,swa\}}$ for each query, and outputs

$$A_{NSA} = g_{slc} \cdot A_{slc} + g_{cmp} \cdot A_{cmp} + g_{swa} \cdot A_{swa}. \tag{2}$$

Other learned variants directly predict block- or token-level attention scores (Gao et al., 2025; DeepSeek-AI, 2025) via a lightweight scoring branch, approximate attention scores in lower rank (Singhania et al., 2024; Tan et al., 2025), combine the NSA branches (Zhao et al., 2025), or use a "landmark" token per block as $\boldsymbol{k}^B$ and $\boldsymbol{v}^B$ (Mohtashami & Jaggi, 2023). Many other methods focus on training-free scenario, and heuristically take block statistics such as mean-pooling as $\boldsymbol{k}^B$ (Tang et al., 2024; Chen et al., 2024a; Jiang et al., 2024a; Xiao et al., 2024; Lu et al., 2024; Zhang et al., 2025; Lee et al., 2024); while we focus on exploring the trained scenario to best fit the model to sparse context.

2.2 Linear attention

The recent revival of recurrent models beginning with Linear Attention (Katharopoulos et al., 2020) highlights an efficient alternative to transformers with limited performance degradation. Although both compress past information into a fixed-sized memory, such modern variants differ from classical RNNs in that training can be parallelized as there is no nonlinear transform in the timewise memory transition. Concretely, given $\boldsymbol{q}, \boldsymbol{k}, \boldsymbol{v}$, the output of each Linear Attention step is roughly

$$\boldsymbol{o}_t = \sum_{j=1}^{t} (\boldsymbol{q}_t^T \boldsymbol{k}_j) \boldsymbol{v}_j = (\sum_{j=1}^{t} \boldsymbol{v}_j \boldsymbol{k}_j^T) q_t = \boldsymbol{S}_t \boldsymbol{q}_t, \tag{3}$$

where a matrix-valued hidden state is conceptually updated by accumulating past KVs, i.e., $\boldsymbol{S}_t = \boldsymbol{S}_{t-1} + \boldsymbol{v}_t \boldsymbol{k}_t^T$. More sophisticated update rules have been further introduced, e.g. exponential decay of S_t in RWKV (Peng et al., 2023) and RetNet (Sun et al., 2023), or the parameterized input transforms in S4 (Gu et al., 2022). In these models, past information will be gradually overwritten without explicit mechanisms to decide whether the current input should be stored. Later models including Mamba (Gu & Dao, 2024), Mamba2 (Dao & Gu, 2024), and GLA (Yang et al., 2024c) use input-dependent gating to ignore unimportant inputs and better retain critical past memory. Particularly, DeltaNet improves Mamba by delta rule updates (Schlag et al., 2021; Yang et al., 2024d), and Gated DeltaNet adds additional gating and achieves strong long-context retrieval performance, hence we choose it as a strong baseline and the model backbone in our study. In Gated DeltaNet, the memory update is performed as

$$\boldsymbol{S}_t = \boldsymbol{S}_{t-1}(\alpha_t(\boldsymbol{I} - \beta_t \boldsymbol{k}_t \boldsymbol{k}_t^T)) + \beta_t \boldsymbol{v}_t \boldsymbol{k}_t^T, \tag{4}$$

where the input-dependent gates α_t, β_t control the update and retention of past memory. Beyond gating for better selectivity, other works focus on boosting expressiveness of limited memory via improved memory parameterization (Qin et al., 2024a; Peng et al., 2025) or a mixture of routed memories (Du et al., 2025). However, without direct access to past tokens, forgetfulness is inevitable for recurrent models with limited memory.

2.3 Hybrid models

To alleviate the forgetfulness of linear attention, a straightforward remedy is to put attention back. Adding a small number of full attention layers can significantly improve retrieval performance (Lenz et al., 2025; Waleffe et al., 2024; MiniMax et al., 2025; Wang et al., 2025) but at increased computational demands. By contrast, we prioritize preserving time and space complexity by interleaving token mixers of intermediate costs. SE-Attn explores parameter efficient fine-tuning of attention layers in hybrid Mamba2 into query-aware sparse attention (Nunez et al., 2024), and laNSA utilizes the more sophisticated token mixer in a different pretraining scenario. Furthermore, it has been observed that interleaving SWA layers can markedly enhance linear attention performance (Ren et al.,

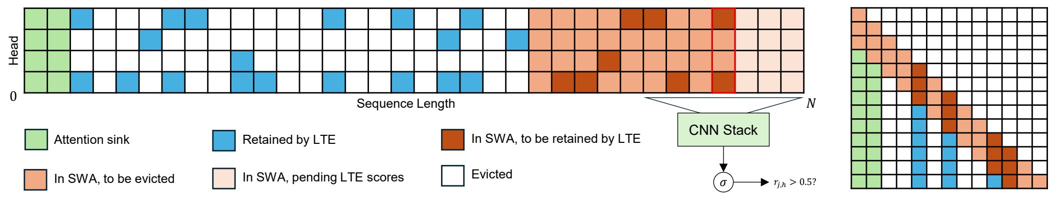


Figure 2: Left: Illustration of our KV eviction scheme, combining LTE, an attention sink (size $s = 2$), and SWA ($w = 12$), with 4 KV heads. A CNN stack is trained end-to-end to predict a per-token, per-head retention score $r_{j,h}$ to decide whether a KV-pair will be evicted when moved out of SWA. With recent KVs cached by SWA, the CNN can read short-range past and future context. r_j can be computed once the CNN receptive field is fully covered by SWA. Right: Resulting per-head A-plus-column sparsity pattern; colored tiles indicate selected tokens at each step.

2024; Zuo et al., 2024; Yang et al., 2024b; Arora et al., 2024b; Cabannes et al., 2025). Closer to laLTE, Ren et al. (2023) and Nguyen et al. (2024) study hybrid models that learn to select tokens as queries to perform SWA. These works nevertheless pursue aims distinct from ours: While SWA can be viewed as an eviction policy, it does not grant access to distant tokens, even when stacked together (Xiao, 2025). Most similar to our work, the concurrent HAXA

3 Methods

3.1 Learnable Token Eviction

3.1.1 LTE module

We retain KV pairs using a per-token, per-head retention score $r_{j,h}$ predicted by the LTE module and restrict attention by combining LTE with an attention sink and sliding-window attention (SWA). Concretely, for the decoding step i and head h, the index set is

$$\mathcal{I}_{i,h} = \{j : j \leq i, (j \leq s) \vee (r_{j,h} > 0.5) \vee (j \geq i - w)\}, \tag{5}$$

which induces an A+column-shaped attention pattern as shown in Figure 2. To determine retention accurately and efficiently, we anchor our LTE module design around four key properties:

1. Fine-grained: Patterns of attention maps vary significantly across layers and heads, some have sharp focus on a few tokens, some demonstrate recognizable local clusters, while others are rather dispersed. Therefore, instead of blockwise and layerwise decision, we choose to make an eviction decision of each KV pair on each head to capture head-specific patterns and make the best use of each head's own cache budget.
2. Contextualized: In addition to each token's own KV, we try to leverage local context including both past and future tokens to better understand each token and its eventual utility. Information from future tokens can not be captured by the causal/recurrent layers used in our model. Hence we use a 3-layer 1D CNN with kernel size=3, dilation=2, and a total receptive field $R = 13$ tokens to ingest local context from both sides. This "look-ahead" is made possible by SWA during decoding: recent KVs are stored in a window $w \gg R$ (e.g., with $w = 768$). We can defer computing r_j by $R//2$ steps until all necessary KVs within the receptive field $[j - R//2, j + R//2]$ are readily available. This computation can be even further deferred, since r_j is not needed as long as token j remains inside the window.
3. Independent: Inspired by ReMoE (Wang et al., 2024), we decide whether a token is important per se, independently without global scheduling such as picking the top-K tokens per head. This not only reduces computational overheads, but leads to simplicity, flexibility, and fine-grained budgeting compared to pre-allocating cache budget K per head base on certain heuristics in other sparse attention approaches described above.
4. Lightweight: We prioritize efficient and parallel inference, hence we choose to apply a small CNN module with only 3 layers and roughly 1% extra parameters in our experiments, and with the channel width halved at each layer.

Figure 2 sketches the full eviction scheme. More specifically, the LTE module consumes the concatenation of key and value vectors per head, prior to rotary positional encodings (RoPE) (Su et al., 2024) for better position generalization. Convolutions are applied independently per head, but executed in parallel across heads using grouped 1D convolutions. Each convolution is followed by Swish activation and a dropout, and a final linear layer maps the per-head feature to scalar scores, also implemented by grouped convolution. Unlike ReMoE, we pass the scalar through a sigmoid to obtain $r_{j,h}$, which stabilizes training. We then binarize the scores with a 0.5 decision threshold. Inputs are zero-padded on the left, and thanks to the deferred computation with SWA, padding is not necessary on the right. The first $s = 4$ tokens are further forcefully retained as the attention sink. We can then simply leverage FlexAttention (Dong et al., 2024) for efficient attention computation with token and head masking during training.

3.1.2 Adaptive training

The model training is a challenge as LTE outputs only r scores to decide a discrete attention mask, hence the gradient cannot be directly back-propagated into LTE parameters. Instead, we leverage straight-through estimator by conceptually applying an all-one mask m to the value vectors so that $v'_{j,h} := v_{j,h} \cdot m_{j,h}$, and use gradients on $m_{j,h}$ as the surrogate gradient, That is to say,

$$\frac{\partial \mathcal{L}}{\partial r_{j,h}} := \frac{\partial \mathcal{L}}{\partial m_{j,h}} = \left\langle \boldsymbol{v}_{j,h}, \frac{\partial \mathcal{L}}{\partial \boldsymbol{v}_{j,h}} \right\rangle, \tag{6}$$

In this way, the gradient on $r_{j,h}$ is tied with the contribution to the loss of the underlying $v_{j,h}$ it controls. Stable end-to-end training is observed in practice under this scheme.

Some heads mainly focus on a few tokens, and their attention patterns are naturally sparsified by LTE without any additional constraints: On unimportant $v_{j,h}$, the all-one $m_{j,h}$ is pulled to zero by a negative gradient, which is copied to $r_{j,h}$. Some other heads have a dispersed attention pattern and benefit from an explicit sparsity prior to avoid excessive density. Inspired by ReMoE (Wang et al., 2024), we impose a dynamically adjusted L1-style penalty to encourage sparsity on retained tokens:

$$\mathcal{L}_{sparse} = \sum_h \lambda_h \sum_i \mathrm{ReLU}(r_{i,h} - 0.5) \tag{7}$$

We target a cap $b = 512$ on the number of retained tokens, anticipating $c_h = \sum_i \mathbf{1}[r_{i,h} > 0.5] \leq b$. To adapt the model to this cap, we adjust λ_h with a feedback loop during training: increase when the moving average of c_h exceeds b, decrease when it falls below $0.95 \cdot b$, with an upper bound $\lambda_h \leq 1$ for stability. Empirically, many heads quickly drive λ_h toward zero, suggesting that LTE gradient signals alone suffice to induces adequate sparsity. In this way, the attention density of each head is determined from end-to-end training using gradient signals on the values, without the need of handcrafted rules for budget allocation. More details are described in Appendix A.

3.1.3 Inference implementation

Capped capacity Our models are trained on fixed-length sequences ($N = 4096$ in our experiments). Although the number of retained tokens is regularized by the sparsity penalty, the actual computation depends on the retention score of each KV predicted from the input, and no hard cap is enforced as mentioned above. The case is different during inference as we expect guaranteed time and space complexity given arbitrary input with arbitrary length. Therefore we still have to resort to a global budget cap: the number of cached KVs out of window is restricted to at most b per head.

Cached prefilling and decoding We first allocate and fill a fixed-size cache as shown in Figure 3 during prefilling, consisting of two segments: one holds the last w KVs within the SWA window, and one gathers KVs already out-of-window but retained by LTE (or as the attention sink). This yields contiguous KVs amenable to tiled attention implementation, avoids memory management intricacies due to cache length imbalance between heads, and prevents worst-case regressions when a head is unusually dense. More specifically, since we store post-RoPE KVs, the physical locations in the cache are irrelevant and we can freely reorder or compact them. This allows us to use the SWA segment as a circular buffer with a `next_ptr` maintained, and compact out-of-window KVs into a contiguous span. Similar to Pagliardini et al. (2023), we leverage this compaction to achieve faster

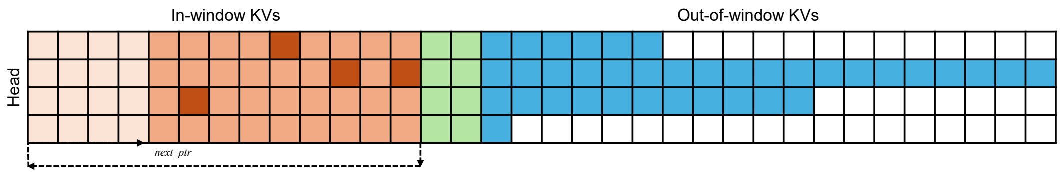


Figure 3: Illustration of the KV cache layout in LTE inference, coloring identical to Figure 2: a circular SWA segment for in-window KVs, and an out-of-window segment with capped capacity for LTE-retained KVs and the attention sink. LTE scoring is batched and deferred as late as possible.

KV-sparse attention. When the capacity happens to exceed on certain head, we keep those with top-b scores. We then implement a Triton kernel for SWA+KV-sparse attention in the tiled Flash Attention manner. Attention outputs of each query tile is computed in two stages: 1) SWA using tiles from original KVs; and 2) KV-sparse attention using tiles from the compacted KV cache. Compacted KV entries inside the sliding window of each query are masked out to avoid double-counting, and tiles completely inside the window (or breaking causality) are skipped for efficiency. During decoding, when a new KV pair is pushed into the SWA buffer, the leftmost KV pair is popped out. If its $r > 0.5$, we will copy it to the next available slot of each head in the out-of-window segment. If the out-of-window segment is already full, we will evict the lowest-score entry if it scores higher. A fused Triton kernel for this cache replacement process is implemented to obtain best efficiency. Now there is only one query token, and its relevant KVs are already compacted into a contiguous segment when the SWA segment is full. Hence we can simply utilize the existing highly-efficient `flash_attn_with_kvcache` API that supports `cache_seqlens` during decoding.

Lazy scoring We do not need to compute r_j immediately when the computation is possible (i.e. current n reaches $j + R//2$. Instead, we can defer and batchify the LTE computation as long as all the tokens within the receptive field remain in-window. Therefore, we trigger LTE scoring only when uncomputed LTE score r_j requires a token that is about to leave the window, i.e. $N - w = j - R//2$. We then rotate the buffer to move *next_ptr* to 0, and compute the LTE score of tokens from $N - w + R//2$ to $N - R//2$ altogether. Another problem is that LTE consumes pre-RoPE keys, while KV cache stores post-RoPE keys, hence we need to undo RoPE on cached keys by applying the corresponding inverse rotation. With RoPE implemented using cached sinusoids, this is equivalent to applying RoPE on the keys with negated sine values at offset $N - w$. With this trick, LTE scoring is carried out only every $W - R$ steps in parallel with minimal amortized overhead.

3.2 Hybrid modeling

We use Gated DeltaNet as our baseline and backbone linear attention for laLTE and laNSA, thanks to its relatively strong performance on retrieval tasks. We apply a 1:1 interleaving of linear attention and alternative token mixers layers, following the Gated DeltaNet + SWA design by (Yang et al., 2024b) known as Gated DeltaNet-H1 for a simplified comparison. We use $w = 1024$ for SWA and $w = 768$ for LTE. Although the LTE cache size is capped to $b = 512$, the observed per-head average retained counts is around 256, so their effective total memory capacity is comparable.

4 Empirical studies

We evaluate hybrid linear-attention models that interleave Gated DeltaNet (GDN) with various mixers from the hierarchy in Figure 1: sliding-window attention (GDN+SWA), learnable token eviction (laLTE), native sparse attention (laNSA), full-attention (GDN+Attn.), plus several ablated variants of LTE. We compare against pure GDN, NSA, and a strong transformer (Transf.++) following Gu & Dao (2024). Given limited computational resources available, we train 0.4B- and 1.4B-sized models on 10B and 30B tokens from FineWeb-Edu (Penedo et al., 2024), following Yang et al. (2024b). We implement LTE layers, hybrid models, as well as inference kernels for LTE and NSA on top of Flash Linear Attention (Yang & Zhang, 2024). Below we leverage both short-context and long-context/retrieval-intensive benchmarks to assess models above and to demonstrate the effectiveness of our laLTE and laNSA approaches, and compare the prefilling and decoding efficiency. More details are available in Appendix A. Additional results on KV retention rates are given in Appendix B.

Table 1: Results (perplexity and accuracy%) on short-context tasks. Best numbers are bolded.

Model	Wiki.	LMB	LMB	PIQA	Hella	Winog.	ArcE	ArcC	BoolQ	SciQ	SIQA	Avg.↑
	ppl↓	ppl↓	acc↑	acc↑	acc_n↑	acc↑	acc↑	acc_n↑	acc↑	acc↑	acc↑	
0.4B Params., 10B Tokens												
GDN	27.56	36.06	31.9	66.4	39.1	50.9	56.8	25.9	58.4	71.0	37.5	48.7
GDN+SWA	26.99	33.87	33.7	66.4	39.6	51.5	56.4	27.9	60.2	72.4	38.5	49.6
laLTE	26.65	**30.03**	34.4	66.1	**40.2**	**52.8**	57.2	26.3	**61.8**	**75.6**	38.1	**50.3**
laNSA	26.84	36.19	33.2	66.1	39.9	51.8	**58.0**	**28.3**	61.6	73.1	37.7	50.0
GDN+Attn.	**26.61**	33.50	**34.6**	**66.5**	39.8	51.6	57.4	27.0	55.8	74.9	38.2	49.5
NSA	27.70	39.33	33.5	66.4	38.7	50.8	55.8	28.1	59.7	69.4	37.3	48.8
Transf.++	27.97	40.36	32.8	65.9	38.3	51.1	56.8	28.0	61.2	71.0	**38.6**	49.3
1.4B Params, 30B Tokens												
GDN	18.31	15.27	43.2	70.9	52.4	54.6	67.8	34.4	58.0	**83.9**	40.4	56.2
GDN+SWA	18.53	14.77	43.7	70.4	51.7	55.0	66.8	35.2	61.4	81.5	40.2	56.2
laLTE	**17.99**	14.79	43.9	71.1	**54.0**	**55.2**	**68.4**	**35.8**	60.6	83.4	39.4	**56.9**
laNSA	18.16	**14.73**	**44.9**	**71.2**	52.4	54.6	67.3	35.5	59.9	82.4	**40.8**	56.6
GDN+Attn.	18.68	16.10	43.2	69.6	51.0	53.5	66.3	32.8	53.8	81.5	40.1	54.6
NSA	19.53	18.48	42.2	68.8	50.1	53.0	65.6	34.0	59.1	80.5	39.1	54.7
Transf.++	20.68	20.17	39.1	68.1	47.2	52.0	63.8	33.5	**62.5**	80.5	38.8	54.0

4.1 Short-context benchmarks

We first evaluate zero-shot commonsense tasks using LM-evaluation-harness (Gao et al., 2024). Given short inputs (generally shorter than the SWA window), we do not expect denser attention to confer any advantage. Thus performance should be broadly similar across models, and differences most likely reflect training quality rather than access to distant tokens. This is confirmed by results in Table 1. At 1.4B, GDN slightly outperforms Transformer++, while NSA and GDN+Attn. fall in between; the ordering is not consistent at 0.4B. GDN+SWA performs similar or slightly better, while adding LTE or NSA further yields small gains. Overall, the gaps are small and can only corroborate Yang et al. (2024b)'s observations that linear attention and hybrid models (including our laLTE and laNSA) are comparable to or slightly better than transformers on short-context tasks.

4.2 Retrieval-intensive benchmarks

We then assess on single needle-in-a-haystack (S-NIAH) from RULER (Hsieh et al., 2024) and the retrieval-intensive EVAPORATE suite (Arora et al., 2024b). S-NIAH spans tasks from retrieving numbers in synthetic texts (S1) to the rather challenging extraction of long UUIDs from real essays (S3). We evaluate contexts of 1K–4K tokens, where most models attain non-trivial accuracy. EVAPORATE comprises realistic QA-style retrieval over passages up to 4K tokens, which is challenging for linear-attention models and is widely used in the area. With the demand to retrieve from distant past, we expect that hybrid models with more direct and accurate access to past tokens will have better performance, for which laLTE and laNSA have an edge when time and/or space complexity is constrained. Nevertheless, we acknowledge that access to past tokens alone does not always translate into successful retrieval, higher computational complexity does not monotonically improve accuracy, and the actual outcomes can be task-dependent as observed by Wang et al. (2025).

Results in Table 2 and Table 3 confirm the effectiveness of hybrid models, laLTE and laNSA in particular. Specifically, pure GDN excels on S-NIAH; at 1.4B it approaches full transformers, echoing Yang et al. (2024b). However, GDN lags on EVAPORATE. GDN+SWA considerably improves on it, but is much weaker on S-NIAH, especially beyond its 1K window size and at 1.4B. At the cost of efficiency, GDN+Attn. consistently achieves best EVAPORATE results across scales, surpassing full transformers, and performs comparably to full transformers and pure GDN on S-NIAH. As for our models, despite similar time and space complexity, laLTE outperforms pure GDN and GDN+SWA on both suites, and approaches GDN+Attn. on S-NIAH at 1.4B. This confirms laLTE's effectiveness and versatility for long-context retrieval under strict complexity budgets. With full history kept in memory, laNSA further attains the best results among linear-time models on EVAPORATE, approaches full transformers on 0.4B scale S-NIAH, and outperforms pure NSA. However it trails full transformer and GDN+Attn. on EVAPORATE and weaker on 1.4B S-NIAH, despite theoretical all-history access. It also has lower performance than laLTE on retrieval from shorter context (e.g. S3-1K), possibly due to conflict between different attention branches (Hu et al., 2025). Notably, NSA requires at least 16 group size for grouped-query attention. In our models with rel-

Table 2: Results (%) on single needle-in-a-haystack benchmarks from the RULER suite. Numbers higher than 80% are bolded.

Model	S1-1K	S1-2K	S1-4K	S2-1K	S2-2K	S2-4K	S3-1K	S3-2K	S3-4K	Avg.
0.4B Params., 10B Tokens										
GDN	**99.8**	**92.8**	45.2	**100.0**	**94.8**	32.4	0.2	0.2	0.0	51.7
GDN+SWA	**100.0**	52.0	27.0	61.2	67.2	27.4	75.4	64.8	17.4	54.7
laLTE	**99.8**	**86.8**	36.4	**99.8**	74.2	25.0	**87.2**	42.8	17.8	63.3
laNSA	**100.0**	**100.0**	65.0	**88.6**	**100.0**	37.6	**97.6**	73.0	11.8	74.8
GDN+Attn.	**100.0**	**100.0**	77.0	**100.0**	**99.6**	52.6	**95.4**	**87.0**	12.2	80.4
NSA	**100.0**	**98.8**	46.6	**100.0**	**95.8**	36.4	**94.4**	64.6	18.6	72.8
Transf.++	**100.0**	**99.6**	50.8	**100.0**	**100.0**	55.4	62.8	71.4	35.8	75.1
LTE-MLP	**95.6**	50.0	23.0	**100.0**	56.6	24.4	25.4	32.0	16.2	47.0
Pure LTE	**94.4**	37.8	21.4	74.2	54.6	22.6	**85.8**	37.6	15.0	49.3
1.4B Params, 30B Tokens										
GDN	**100.0**	**100.0**	**100.0**	**93.6**	**99.0**	**88.0**	**93.0**	**77.6**	49.0	88.9
GDN+SWA	**100.0**	52.0	29.6	**93.4**	**83.2**	34.2	**92.8**	57.0	19.4	62.4
laLTE	**100.0**	**99.2**	**95.0**	**100.0**	**98.0**	**81.4**	**85.4**	55.6	33.2	83.1
laNSA	**100.0**	**99.8**	57.8	**80.6**	**100.0**	68.2	60.4	**94.2**	34.0	77.2
GDN+Attn.	**99.8**	**100.0**	78.4	**100.0**	**100.0**	**90.6**	**83.8**	**71.0**	58.0	86.8
NSA	**100.0**	**93.8**	36.2	**100.0**	**99.4**	40.2	**97.8**	**78.6**	26.8	74.8
Transf.++	**100.0**	**100.0**	**98.4**	**100.0**	**100.0**	**85.6**	**89.4**	**84.2**	64.8	91.4
TOVA	**99.8**	**81.6**	41.0	**100.0**	11.4	8.0	**83.8**	0.0	0.8	47.4
Unif.+SWA	**94.2**	37.6	20.2	**100.0**	54.4	24.2	**84.8**	37.2	23.0	52.8

Table 3: Results (%) on recall-intensive EVAPORATE tasks. Best linear-time results are bolded.

Model	FDA	SWDE	NQ	SQUAD	TQA	DROP	Avg.
0.4B Params., 10B Tokens							
GDN	4.17	8.37	10.90	27.95	47.69	17.73	19.47
GDN+SWA	5.08	18.27	9.34	**34.52**	49.11	20.51	22.81
laLTE	15.52	19.26	**11.15**	32.98	48.99	**23.67**	25.26
laNSA	**17.33**	23.13	10.71	31.60	**49.17**	21.56	**25.58**
GDN+Attn.	29.31	26.19	11.53	30.53	46.92	18.69	27.19
NSA	7.99	**26.28**	10.01	33.91	47.16	19.74	24.18
Transf.++	28.58	26.46	11.09	31.94	46.68	17.30	27.01
LTE-MLP	12.70	18.63	11.15	31.27	49.76	20.17	23.95
Pure LTE	14.25	18.72	7.44	33.31	46.15	19.45	23.22
1.4B Params., 30B Tokens							
GDN	20.96	25.83	**17.96**	35.02	56.46	20.75	29.50
GDN+SWA	26.68	27.99	16.53	41.09	58.95	**23.29**	32.42
laLTE	28.68	28.98	17.68	**43.00**	**60.60**	22.62	33.59
laNSA	**31.85**	**36.99**	17.20	38.07	58.06	20.60	**33.80**
GDN+Attn.	46.64	41.13	18.09	40.21	58.47	22.57	37.85
NSA	25.59	34.83	13.91	40.65	56.28	23.05	32.38
Transf.++	38.84	34.92	16.19	37.60	55.92	21.66	34.19
TOVA	20.96	30.78	13.49	40.18	58.59	22.62	31.10
Unif.+SWA	15.97	24.21	11.82	22.32	57.17	15.81	24.55

atively small hidden sizes and head counts, NSA is rendered closer to multi-query attention, which can be a performance bottleneck and a factor of instability.

We further ablate laLTE against GDN+following token mixers with constant time and space complexity: **LTE-MLP** replaces CNN with a 2-layer MLP with no context access (especially no future access); **Pure LTE** is a pure transformer with all layers sparsified by LTE; **Unif.+SWA** retains out-of-window tokens at a uniform stride; **TOVA** is one of the state-of-the-art eviction heuristics based on accumulated attention scores (Oren et al., 2024); the latter two are training-free. Although LTE-MLP and TOVA achieve moderate results on EVAPORATE, better than pure GDN, all of them

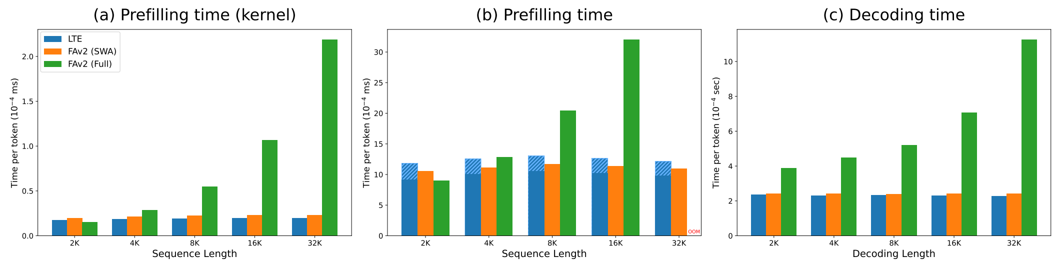


Figure 4: Comparing time costs between LTE and SWA or full attention with Flash Attention-2, on (a) attention kernels prefilling with artificial KVs; and hybrid layers with real inputs during (b) prefilling and (c) decoding with a 4K-token prompt, using our 1.4B models with batch size=32. Shadowed part of the LTE bar in (b) indicate time spent on scoring. Note that decoding takes much longer time than prefilling. Overall, LTE and SWA take similar time, much lower than full attention.

underperform laLTE, especially on S-NIAH; while Unif.+SWA performs much worse than all other models. This confirms the importance of an accurate eviction rule, and underscores the effectiveness of our design of contextualized and learnable eviction. The observation is similar on Pure LTE, which supports our choice to combine sparse attention with linear attention to compensate for the evicted memory. To summarize, all our results confirm the importance of direct and accurate access of past tokens on retrieval-intensive tasks. From GDN, GDN+SWA/laLTE, laNSA, to GDN+Attn., more complete direct access to past tokens generally allows better long-context performance, though task-level outcomes vary; while our laLTE proves its strength for accurate access under rigorous time and space complexity constraints.

4.3 Efficiency

We then benchmark the actual computational efficiency of our LTE approach. All results are measured on a single H100-SXM GPU following our 1.4B configuration with batch size=32. Figure 4 (a) shows the runtime of prefilling kernels at different sequence lengths, compared with SWA and full attention using the highly-optimized FlashAttention-2 (Dao, 2024) kernels. As for LTE we use an artificial half-full out-of-window KV-cache, with 1024 tokens identical to the SWA window size. Our SWA+KV-sparse attention kernel exhibits a linear-growing runtime comparable to SWA, and much faster than full attention. While LTE depends on real context for token eviction decision; many layers and heads retain only a few tokens, but the LTE scoring and caching introduce some additional overheads. We therefore also measure the prefilling and decoding time of our actual 1.4B models on real inputs, implemented based on Flash Linear Attention. To better isolate the speed of our targets, we only measure the time costs of hybrid layers (i.e. excluding other layers like GDN). Results shown in Figure 4 (b)-(c) similarly confirm the efficiency of our implementation. Scoring causes some overheads during prefilling (indicated by the shadowed part), which are largely compensated for by faster attention computation. Thanks to lazy scoring, the overhead becomes ignorable and LTE runs slightly faster than SWA during the decoding stage, which dominates inference time. It is further noteworthy that LTE uses a constant-size KV-cache equal to 1280 tokens, i.e. 2.5MiB per sample per layer under our 1.4B configuration. Compared to full KV cache under longer context, this allows larger batch sizes and higher token throughput with fixed size memory. Taken together, these results support adopting LTE in lieu of SWA for better performance without much efficiency degradation.

5 Conclusion

We study hybrid models with a series of token mixers to improve the retrieval capability of linear attention methods, trying to trade off between time/space complexity and accuracy. We further propose two approaches with different efficiency: laNSA to interleave query-aware native sparse attention of sub-quadratic time complexity when memory capacity permits, and laLTE for token eviction using a learnable contextualized module that maintains constant space complexity and supports an efficient decoding algorithm. Empirical results confirm the effectiveness of our approaches.

REFERENCES

Joshua Ainslie, Santiago Ontanon, Chris Alberti, Vaclav Cvicek, Zachary Fisher, Philip Pham, Anirudh Ravula, Sumit Sanghai, Qifan Wang, and Li Yang. ETC: Encoding Long and Structured Inputs in Transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020*, pp. 268–284, Online, November 2020. Association for Computational Linguistics. URL `https://aclanthology.org/2020.emnlp-main.19`.

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai. GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023*, pp. 4895–4901, Singapore, December 2023a. Association for Computational Linguistics. URL `https://aclanthology.org/2023.emnlp-main.298`.

Joshua Ainslie, Tao Lei, Michiel de Jong, Santiago Ontañón, Siddhartha Brahma, Yury Zemlyanskiy, David Uthus, Mandy Guo, James Lee-Thorp, Yi Tay, Yun-Hsuan Sung, and Sumit Sanghai. CoLT5: Faster Long-Range Transformers with Conditional Computation. arXiv preprint arXiv:2303.09752, October 2023b. URL `http://arxiv.org/abs/2303.09752`.

Sotiris Anagnostidis, Dario Pavllo, Luca Biggio, Lorenzo Noci, Aurelien Lucchi, and Thomas Hofmann. Dynamic Context Pruning for Efficient and Interpretable Autoregressive Transformers. In *Proceedings of the 37th International Conference on Neural Information Processing Systems, NeurIPS 2023*, November 2023. URL `https://openreview.net/forum?id=uvdJgFFzby`.

Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Re. Zoology: Measuring and Improving Recall in Efficient Language Models. In *12th International Conference on Learning Representations, ICLR 2024*, October 2024a. URL `https://openreview.net/forum?id=LY3ukUANko`.

Simran Arora, Sabri Eyuboglu, Michael Zhang, Aman Timalsina, Silas Alberti, James Zou, Atri Rudra, and Christopher Re. Simple linear attention language models balance the recall-throughput tradeoff. In *Proceedings of the 41st International Conference on Machine Learning, ICML 2024*, pp. 1763–1840. PMLR, July 2024b. URL `https://proceedings.mlr.press/v235/arora24a.html`. ISSN: 2640-3498.

Simran Arora, Aman Timalsina, Aaryan Singhal, Benjamin Spector, Sabri Eyuboglu, Xinyi Zhao, Ashish Rao, Atri Rudra, and Christopher Ré. Just read twice: Closing the recall gap for recurrent language models. arXiv preprint arXiv:2407.05483, arXiv, July 2024c. URL `http://arxiv.org/abs/2407.05483`.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The Long-Document Transformer. arXiv preprint arXiv:2004.05150, December 2020. URL `http://arxiv.org/abs/2004.05150`.

Loïc Cabannes, Maximilian Beck, Gergely Szilvasy, Matthijs Douze, Maria Lomeli, Jade Copet, Pierre-Emmanuel Mazaré, Gabriel Synnaeve, and Hervé Jégou. Short window attention enables long-term memorization. arXiv preprint arXiv:2509.24552, arXiv, September 2025. URL `http://arxiv.org/abs/2509.24552`.

Zefan Cai, Yichi Zhang, Bofei Gao, Yuliang Liu, Tianyu Liu, Keming Lu, Wayne Xiong, Yue Dong, Baobao Chang, Junjie Hu, and Wen Xiao. PyramidKV: Dynamic KV Cache Compression based on Pyramidal Information Funneling. arXiv preprint arXiv:2406.02069, October 2024. URL `http://arxiv.org/abs/2406.02069`.

Renze Chen, Zhuofeng Wang, Beiquan Cao, Tong Wu, Size Zheng, Xiuhong Li, Xuechao Wei, Shengen Yan, Meng Li, and Yun Liang. ArkVale: Efficient Generative LLM Inference with Recallable Key-Value Eviction. In *Proceedings of the 38th International Conference on Neural Information Processing Systems, NeurIPS 2024*, November 2024a. URL `https://openreview.net/forum?id=4oAt5L4lYe`.

Yilong Chen, Guoxia Wang, Junyuan Shang, Shiyao Cui, Zhenyu Zhang, Tingwen Liu, Shuohuan Wang, Yu Sun, Dianhai Yu, and Hua Wu. NACL: A General and Effective KV Cache Eviction Framework for LLM at Inference Time. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024*, pp. 7913–7926, Bangkok, Thailand, August 2024b. Association for Computational Linguistics. URL `https://aclanthology.org/2024.acl-long.428/`.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating Long Sequences with Sparse Transformers. arXiv preprint arXiv:1904.10509, April 2019. URL `http://arxiv.org/abs/1904.10509`.

Tri Dao. FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning. In *12th International Conference on Learning Representations, ICLR 2024*, October 2024. URL `https://openreview.net/forum?id=mZn2Xyh9Ec`.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized Models and Efficient Algorithms Through Structured State Space Duality. In *Proceedings of the 41st International Conference on Machine Learning, ICML 2024*, June 2024. URL `https://openreview.net/forum?id=ztn8FCR1td`.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Re. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. In *Proceedings of the 36th International Conference on Neural Information Processing Systems, NeurIPS 2022*, October 2022. URL `https://openreview.net/forum?id=H4DqfPSibmx`.

DeepSeek-AI. DeepSeek-V3.2-Exp: Boosting Long-Context Efficiency with DeepSeek Sparse Attention. Technical report, September 2025. URL `https://github.com/deepseek-ai/DeepSeek-V3.2-Exp`.

Juechu Dong, Boyuan Feng, Driss Guessous, Yanbo Liang, and Horace He. Flex Attention: A Programming Model for Generating Optimized Attention Kernels. arXiv preprint arXiv:2412.05496, December 2024. URL `http://arxiv.org/abs/2412.05496`.

Jusen Du, Weigao Sun, Disen Lan, Jiaxi Hu, and Yu Cheng. MoM: Linear Sequence Modeling with Mixture-of-Memories. arXiv preprint arXiv:2502.13685, February 2025. URL `http://arxiv.org/abs/2502.13685`.

Yuan Feng, Junlin Lv, Yukun Cao, Xike Xie, and S. Kevin Zhou. Ada-KV: Optimizing KV Cache Eviction by Adaptive Budget Allocation for Efficient LLM Inference. arXiv preprint arXiv:2407.11550, January 2025. URL `http://arxiv.org/abs/2407.11550`.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, July 2024. URL `https://zenodo.org/records/12608602`.

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, February 2025. URL `http://arxiv.org/abs/2410.13276`.

Suyu Ge, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, and Jianfeng Gao. Model Tells You What to Discard: Adaptive KV Cache Compression for LLMs. In *The 12th International Conference on Learning Representations, ICLR 2024*, October 2023. URL `https://openreview.net/forum?id=uNrFpDPMyo`.

Albert Gu and Tri Dao. Mamba: Linear-Time Sequence Modeling with Selective State Spaces. In *1st Conference on Language Modeling, COLM 2024*, May 2024. URL `https://openreview.net/forum?id=tEYskw1VY2`.

Albert Gu, Karan Goel, and Christopher Re. Efficiently Modeling Long Sequences with Structured State Spaces. In *10th International Conference on Learning Representations, ICLR 2022*, October 2022. URL `https://openreview.net/forum?id=uYLFoz1vlAC`.

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: Human Language Technologies (Volume 1: Long Papers), NAACL-HLT 2024*, pp. 3991–4008, Mexico City, Mexico, June 2024. Association for Computational Linguistics. URL `https://aclanthology.org/2024.naacl-long.222/`.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, and Boris Ginsburg. RULER: What's the Real Context Size of Your Long-Context Language Models? In *1st Conference on Language Modeling, COLM 2024*, August 2024. URL `https://openreview.net/forum?id=kIoBbc76Sy`.

Yuxuan Hu, Jianchao Tan, Jiaqi Zhang, Wen Zan, Pingwei Sun, Yifan Lu, Yerui Sun, Yuchen Xie, Xunliang Cai, and Jing Zhang. Optimizing Native Sparse Attention with Latent Attention and Local Global Alternating Strategies. arXiv preprint arXiv:2511.00819, November 2025. URL `http://arxiv.org/abs/2511.00819`. arXiv:2511.00819 [cs].

Samy Jelassi, David Brandfonbrener, Sham M. Kakade, and Eran Malach. Repeat After Me: Transformers are Better than State Space Models at Copying. In *Proceedings of the 41st International Conference on Machine Learning, ICML 2024*, pp. 21502–21521. PMLR, July 2024. URL `https://proceedings.mlr.press/v235/jelassi24a.html`. ISSN: 2640-3498.

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 *Proceedings of the 38th International Conference on Neural Information Processing Systems, NeurIPS 2024*, November 2024a. URL `https://openreview.net/forum?id=fPBACAbqSN`.

Yikun Jiang, Huanyu Wang, Lei Xie, Hanbin Zhao, Chao Zhang, Hui Qian, and John C. S. Lui. D-LLM: A Token Adaptive Computing Resource Allocation Strategy for Large Language Models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, November 2024b. URL `https://openreview.net/forum?id=UIOjGTKHQG`.

Hyun-rae Jo and Dongkun Shin. A2SF: Accumulative Attention Scoring with Forgetting Factor for Token Pruning in Transformer Decoder. arXiv preprint arXiv:2407.20485, July 2024. URL `http://arxiv.org/abs/2407.20485`.

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, ICML 2020*, pp. 5156–5165. PMLR, November 2020. URL `https://proceedings.mlr.press/v119/katharopoulos20a.html`.

Minsoo Kim, Kyuhong Shim, Jungwook Choi, and Simyung Chang. InfiniPot: Infinite Context Processing on Memory-Constrained LLMs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024*, pp. 16046–16060, Miami, Florida, USA, November 2024. Association for Computational Linguistics. URL `https://aclanthology.org/2024.emnlp-main.897/`.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The Efficient Transformer. In *The 8th International Conference on Learning Representations, ICLR 2020*, October 2020. URL `https://openreview.net/forum?id=rkgNKkHtvB`.

Heejun Lee, Geon Park, Youngwan Lee, Jaduk Suh, Jina Kim, Wonyong Jeong, Bumsik Kim, Hyemin Lee, Myeongjae Jeon, and Sung Ju Hwang. A Training-Free Sub-quadratic Cost Transformer Model Serving Framework with Hierarchically Pruned Attention. In *The Thirteenth International Conference on Learning Representations*, October 2024. URL `https://openreview.net/forum?id=PTcMzQgKmn`.

Barak Lenz, Opher Lieber, Alan Arazi, Amir Bergman, Avshalom Manevich, Barak Peleg, Ben Aviram, Chen Almagor, Clara Fridman, Dan Padnos, Daniel Gissin, Daniel Jannai, Dor Muhlgay, Dor Zimberg, Edden M. Gerber, Elad Dolev, Eran Krakovsky, Erez Safahi, Erez Schwartz, Gal Cohen, Gal Shachaf, Haim Rozenblum, Hofit Bata, Ido Blass, Inbal Magar, Itay Dalmedigos, Jhonathan Osin, Julie Fadlon, Maria Rozman, Matan Danos, Michael Gokhman, Mor Zusman, Naama Gidron, Nir Ratner, Noam Gat, Noam Rozen, Oded Fried, Ohad Leshno, Omer Antverg, Omri Abend, Or Dagan, Orit Cohavi, Raz Alon, Ro'i Belson, Roi Cohen, Rom Gilad, Roman Glozman, Shahar Lev, Shai Shalev-Shwartz, Shaked Haim Meirom, Tal Delbari, Tal Ness, Tomer Asida, Tom Ben Gal, Tom Braude, Uriya Pumerantz, Josh Cohen, Yonatan Belinkov, Yuval Globerson, Yuval Peleg Levy, and Yoav Shoham. Jamba: Hybrid Transformer-Mamba Language Models. In *13th International Conference on Learning Representations, ICLR 2025*, October 2025. URL `https://openreview.net/forum?id=JFPaD7lpBD`.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. SnapKV: LLM Knows What You are Looking for Before Generation. In *Proceedings of the 38th International Conference on Neural Information Processing Systems, NeurIPS 2024*, November 2024. URL `https://openreview.net/forum?id=poE54GOq2l`.

Zichang Liu, Aditya Desai, Fangshuo Liao, Weitao Wang, Victor Xie, Zhaozhuo Xu, Anastasios Kyrillidis, and Anshumali Shrivastava. Scissorhands: Exploiting the Persistence of Importance Hypothesis for LLM KV Cache Compression at Test Time. In *Proceedings of the 37th International Conference on Neural Information Processing Systems, NeurIPS 2023*, November 2023. URL `https://openreview.net/forum?id=JZfg6wGi6g`.

Enzhe Lu, Zhejun Jiang, Jingyuan Liu, Yulun Du, Tao Jiang, Chao Hong, Shaowei Liu, Weiran He, Enming Yuan, Yuzhi Wang, Zhiqi Huang, Huan Yuan, Suting Xu, Xinran Xu, Guokun Lai, Yanru Chen, Huabin Zheng, Junjie Yan, Jianlin Su, Yuxin Wu, Neo Y. Zhang, Zhilin Yang, Xinyu Zhou, Mingxing Zhang, and Jiezhong Qiu. MoBA: Mixture of Block Attention for Long-Context LLMs. arXiv preprint arXiv:2502.13189, February 2025. URL `http://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, Miami, Florida, USA, November 2024. Association for Computational Linguistics. URL `https://aclanthology.org/2024.findings-emnlp.417/`.

MiniMax, Aonian Li, Bangwei Gong, Bo Yang, Boji Shan, Chang Liu, Cheng Zhu, Chunhao Zhang, Congchao Guo, Da Chen, Dong Li, Enwei Jiao, Gengxin Li, Guojun Zhang, Haohai Sun, Houze Dong, Jiadai Zhu, Jiaqi Zhuang, Jiayuan Song, Jin Zhu, Jingtao Han, Jingyang Li, Junbin Xie, Junhao Xu, Junjie Yan, Kaishun Zhang, Kecheng Xiao, Kexi Kang, Le Han, Leyang Wang, Lianfei Yu, Liheng Feng, Lin Zheng, Linbo Chai, Long Xing, Meizhi Ju, Mingyuan Chi, Mozhi Zhang, Peikai Huang, Pengcheng Niu, Pengfei Li, Pengyu Zhao, Qi Yang, Qidi Xu, Qiexiang Wang, Qin Wang, Qiuhui Li, Ruitao Leng, Shengmin Shi, Shuqi Yu, Sichen Li, Songquan Zhu, Tao Huang, Tianrun Liang, Weigao Sun, Weixuan Sun, Weiyu Cheng, Wenkai Li, Xiangjun Song, Xiao Su, Xiaodong Han, Xinjie Zhang, Xinzhu Hou, Xu Min, Xun Zou, Xuyang Shen, Yan Gong, Yingjie Zhu, Yipeng Zhou, Yiran Zhong, Yongyi Hu, Yuanxiang Fan, Yue Yu, Yufeng Yang, Yuhao Li, Yunan Huang, Yunji Li, Yunpeng Huang, Yunzhi Xu, Yuxin Mao, Zehan Li, Zekang Li, Zewei Tao, Zewen Ying, Zhaoyang Cong, Zhen Qin, Zhenhua Fan, Zhihang Yu, Zhuo Jiang, and Zijia Wu. MiniMax-01: Scaling Foundation Models with Lightning Attention. arXiv preprint arXiv:2501.08313, January 2025. URL `http://arxiv.org/abs/2501.08313`.

Amirkeivan Mohtashami and Martin Jaggi. Random-Access Infinite Context Length for Transformers. In *Proceedings of the 37th International Conference on Neural Information Processing Systems, NeurIPS 2023*, November 2023. URL `https://openreview.net/forum?id=7eHn64wOVy`.

Chien Van Nguyen, Huy Huu Nguyen, Thang M. Pham, Ruiyi Zhang, Hanieh Deilamsalehy, Puneet Mathur, Ryan A. Rossi, Trung Bui, Viet Dac Lai, Franck Dernoncourt, and Thien Huu Nguyen. Taipan: Efficient and Expressive State Space Language Models with Selective Attention. arXiv preprint arXiv:2410.18572, October 2024. URL `http://arxiv.org/abs/2410.18572`.

Elvis Nunez, Luca Zancato, Benjamin Bowman, Aditya Golatkar, Wei Xia, and Stefano Soatto. Expansion Span: Combining Fading Memory and Retrieval in Hybrid State Space Models.

arXiv preprint arXiv:2412.13328, December 2024. URL http://arxiv.org/abs/2412.13328. arXiv:2412.13328 [cs].

Matanel Oren, Michael Hassid, Nir Yarden, Yossi Adi, and Roy Schwartz. Transformers are Multi-State RNNs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024*, pp. 18724–18741, Miami, Florida, USA, November 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.emnlp-main.1043/.

Matteo Pagliardini, Daniele Paliotta, Martin Jaggi, and François Fleuret. Fast Attention Over Long Sequences With Dynamic Sparse Flash Attention. In *Proceedings of the 37th International Conference on Neural Information Processing Systems, NeurIPS 2023*, November 2023. URL https://openreview.net/forum?id=UINHuKeWUa.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The FineWeb Datasets: Decanting the Web for the Finest Text Data at Scale. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, November 2024. URL https://openreview.net/forum?id=n6SCkn2QaG.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Leon Derczynski, Xingjian Du, Matteo Grella, Kranthi Gv, Xuzheng He, Haowen Hou, Przemyslaw Kazienko, Jan Kocon, Jiaming Kong, Bartłomiej Koptyra, Hayden Lau, Jiaju Lin, Krishna Sri Ipsit Mantri, Ferdinand Mom, Atsushi Saito, Guangyu Song, Xiangru Tang, Johan Wind, Stanisław Woźniak, Zhenyuan Zhang, Qinghua Zhou, Jian Zhu, and Rui-Jie Zhu. RWKV: Reinventing RNNs for the Transformer Era. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 14048–14077, Singapore, December 2023. Association for Computational Linguistics. URL https://aclanthology.org/2023.findings-emnlp.936.

Bo Peng, Ruichong Zhang, Daniel Goldstein, Eric Alcaide, Xingjian Du, Haowen Hou, Jiaju Lin, Jiaxing Liu, Janna Lu, William Merrill, Guangyu Song, Kaifeng Tan, Saiteja Utpala, Nathan Wilce, Johan S. Wind, Tianyi Wu, Daniel Wuttke, and Christian Zhou-Zheng. RWKV-7 "Goose" with Expressive Dynamic State Evolution. In *2nd Conference on Language Modeling, COLM 2025*, August 2025. URL https://openreview.net/forum?id=ayB1PACN5j.

Piotr Piękos, Róbert Csordás, and Jürgen Schmidhuber. Mixture of Sparse Attention: Content-Based Learnable Sparse Attention via Expert-Choice Routing. arXiv preprint arXiv:2505.00315, arXiv, May 2025. URL http://arxiv.org/abs/2505.00315.

Zhen Qin, Songlin Yang, Weixuan Sun, Xuyang Shen, Dong Li, Weigao Sun, and Yiran Zhong. HGRN2: Gated Linear RNNs with State Expansion. In *1st Conference on Language Modeling, COLM 2024*, August 2024a. URL https://openreview.net/forum?id=y6SqbJfCSk.

Ziran Qin, Yuchen Cao, Mingbao Lin, Wen Hu, Shixuan Fan, Ke Cheng, Weiyao Lin, and Jianguo Li. CAKE: Cascading and Adaptive KV Cache Eviction with Layer Preferences. In *13th International Conference on Learning Representations, ICLR 2025*, October 2024b. URL https://openreview.net/forum?id=EQgEMAD4kv.

David Raposo, Sam Ritter, Blake Richards, Timothy Lillicrap, Peter Conway Humphreys, and Adam Santoro. Mixture-of-Depths: Dynamically allocating compute in transformer-based language models. arXiv preprint arXiv:2404.02258, arXiv, April 2024. URL http://arxiv.org/abs/2404.02258.

Liliang Ren, Yang Liu, Shuohang Wang, Yichong Xu, Chenguang Zhu, and ChengXiang Zhai. Sparse Modular Activation for Efficient Sequence Modeling. In *Proceedings of the 37th International Conference on Neural Information Processing Systems, NeurIPS 2023*, November 2023. URL https://openreview.net/forum?id=TfbzX6I14i.

Liliang Ren, Yang Liu, Yadong Lu, Yelong Shen, Chen Liang, and Weizhu Chen. Samba: Simple Hybrid State Space Models for Efficient Unlimited Context Language Modeling. In *The 13th*

International Conference on Learning Representations, ICLR 2025, October 2024. URL `https://openreview.net/forum?id=bIlnpVM4bc`.

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, February 2021. ISSN 2307-387X. URL `https://doi.org/10.1162/tacl_a_00353`.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear Transformers Are Secretly Fast Weight Programmers. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021*, pp. 9355–9366. PMLR, July 2021. URL `https://proceedings.mlr.press/v139/schlag21a.html`.

Aman Sharma, Saeed Najafi, Parsa Farinneya, Benyamin Jamialahmadi, Marzieh S. Tahaei, Yuhe Fan, Mehdi Rezagholizadeh, Boxing Chen, and Aref Jafari. DTRNet: Dynamic Token Routing Network to Reduce Quadratic Costs in Transformers. arXiv preprint arXiv:2509.00925, arXiv, August 2025. URL `http://arxiv.org/abs/2509.00925`.

Jingze Shi, Yifan Wu, Bingheng Wu, Yiran Peng, Liangdong Wang, Guang Liu, and Yuyu Luo. Trainable Dynamic Mask Sparse Attention. arXiv preprint arXiv:2508.02124, August 2025. URL `http://arxiv.org/abs/2508.02124`.

Prajwal Singhania, Siddharth Singh, Shwai He, Soheil Feizi, and Abhinav Bhatele. Loki: Low-rank Keys for Efficient Sparse Attention. In *Proceedings of the 38th International Conference on Neural Information Processing Systems, NeurIPS 2024*, November 2024. URL `https://openreview.net/forum?id=raABeiV71j`.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced transformer with Rotary Position Embedding. *Neurocomputing*, 568:127063, February 2024. ISSN 0925-2312. URL `https://www.sciencedirect.com/science/article/pii/S0925231223011864`.

Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive Network: A Successor to Transformer for Large Language Models. arXiv preprint arXiv:2307.08621, August 2023. URL `http://arxiv.org/abs/2307.08621`.

Xin Tan, Yuetao Chen, Yimin Jiang, Xing Chen, Kun Yan, Nan Duan, Yibo Zhu, Daxin Jiang, and Hong Xu. DSV: Exploiting Dynamic Sparsity to Accelerate Large-Scale Video DiT Training. arXiv preprint arXiv:2502.07590, arXiv, March 2025. URL `http://arxiv.org/abs/2502.07590`.

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, ICML 2024*, pp. 47901–47911. PMLR, July 2024. URL `https://proceedings.mlr.press/v235/tang24l.html`. ISSN: 2640-3498.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS 2017*, volume 30. Curran Associates, Inc., 2017. URL `https://papers.nips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html`.

Roger Waleffe, Wonmin Byeon, Duncan Riach, Brandon Norick, Vijay Korthikanti, Tri Dao, Albert Gu, Ali Hatamizadeh, Sudhakar Singh, Deepak Narayanan, Garvit Kulshreshtha, Vartika Singh, Jared Casper, Jan Kautz, Mohammad Shoeybi, and Bryan Catanzaro. An Empirical Study of Mamba-based Language Models. arXiv preprint arXiv:2406.07887, June 2024. URL `http://arxiv.org/abs/2406.07887`.

Dustin Wang, Rui-Jie Zhu, Steven Abreu, Yong Shan, Taylor Kergan, Yuqi Pan, Yuhong Chou, Zheng Li, Ge Zhang, Wenhao Huang, and Jason Eshraghian. A Systematic Analysis of Hybrid Linear Attention. arXiv preprint arXiv:2507.06457, July 2025. URL `http://arxiv.org/abs/2507.06457`.

Hanrui Wang, Zhekai Zhang, and Song Han. SpAtten: Efficient Sparse Attention Architecture with Cascade Token and Head Pruning. In *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pp. 97–110, February 2021. URL `https://ieeexplore.ieee.org/document/9407232`.

Ziteng Wang, Jun Zhu, and Jianfei Chen. ReMoE: Fully Differentiable Mixture-of-Experts with ReLU Routing. In *13th International Conference on Learning Representations, ICLR 2025*, October 2024. URL `https://openreview.net/forum?id=4D0f16Vwc3`.

Kaiyue Wen, Xingyu Dang, and Kaifeng Lyu. RNNs are not Transformers (Yet): The Key Bottleneck on In-Context Retrieval. In *13th International Conference on Learning Representations, ICLR 2025*, October 2025. URL `https://openreview.net/forum?id=h3wbI8Uk1Z`.

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 *Proceedings of the 38th International Conference on Neural Information Processing Systems, NeurIPS 2024*, November 2024. URL `https://openreview.net/forum?id=bTHFrqhASY`.

Guangxuan Xiao. Why Stacking Sliding Windows Can't See Very Far, August 2025. URL `https://guangxuanx.com/blog/stacking-swa`.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient Streaming Language Models with Attention Sinks. In *The 12th International Conference on Learning Representations, ICLR 2024*, October 2023. URL `https://openreview.net/forum?id=NG7sS51zVF`.

Dongjie Yang, Xiaodong Han, Yan Gao, Yao Hu, Shilin Zhang, and Hai Zhao. PyramidInfer: Pyramid KV Cache Compression for High-throughput LLM Inference. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 3258–3270, Bangkok, Thailand, August 2024a. Association for Computational Linguistics. URL `https://aclanthology.org/2024.findings-acl.195/`.

Songlin Yang and Yu Zhang. FLA: A Triton-Based Library for Hardware-Efficient Implementations of Linear Attention Mechanism, January 2024. URL `https://github.com/fla-org/flash-linear-attention`.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated Delta Networks: Improving Mamba2 with Delta Rule. In *The 13th International Conference on Learning Representations, ICLR 2025*, October 2024b. URL `https://openreview.net/forum?id=r8H7xhYPwz`.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated Linear Attention Transformers with Hardware-Efficient Training. In *Proceedings of the 41st International Conference on Machine Learning, ICML 2024*, pp. 56501–56523. PMLR, July 2024c. URL `https://proceedings.mlr.press/v235/yang24ab.html`. ISSN: 2640-3498.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing Linear Transformers with the Delta Rule over Sequence Length. In *Proceedings of the 38th International Conference on Neural Information Processing Systems, NeurIPS 2024*, November 2024d. URL `https://openreview.net/forum?id=y8Rm4VNRPH`.

Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, Yuxing Wei, Lean Wang, Zhiping Xiao, Yuqing Wang, Chong Ruan, Ming Zhang, Wenfeng Liang, and Wangding Zeng. 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), ACL 2025*, pp. 23078–23097, Vienna, Austria, July 2025. Association for Computational Linguistics. URL `https://aclanthology.org/2025.acl-long.1126/`.

Manzil Zaheer, Guru Guruganesh, Kumar 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 *Proceedings of the 34th International Conference on*

Neural Information Processing Systems, NeurIPS 2020, volume 33, pp. 17283–17297. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/hash/c8512d142a2d849725f31a9a7a361ab9-Abstract.html`.

Zihao Zeng, Bokai Lin, Tianqi Hou, Hao Zhang, and Zhijie Deng. In-context KV-Cache Eviction for LLMs via Attention-Gate. arXiv preprint arXiv:2410.12876, October 2024. URL `http://arxiv.org/abs/2410.12876`.

Chaoran Zhang, Lixin Zou, Dan Luo, Xiangyang Luo, Zihao Li, Min Tang, and Chenliang Li. Efficient Sparse Attention needs Adaptive Token Release. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 14081–14094, Bangkok, Thailand, August 2024. Association for Computational Linguistics. URL `https://aclanthology.org/2024.findings-acl.837/`.

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 *Forty-second International Conference on Machine Learning*, June 2025. URL `https://openreview.net/forum?id=74c3Wwk8Tc`.

Yu Zhang and Songlin Yang. Flame: Flash language modeling made easy, January 2025. URL `https://github.com/fla-org/flame`.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Re, Clark Barrett, Zhangyang Wang, and Beidi Chen. H2O: Heavy-Hitter Oracle for Efficient Generative Inference of Large Language Models. In *Proceedings of the 37th International Conference on Neural Information Processing Systems, NeurIPS 2023*, November 2023. URL `https://openreview.net/forum?id=RkRrPp7GKO`.

Weilin Zhao, Zihan Zhou, Zhou Su, Chaojun Xiao, Yuxuan Li, Yanghao Li, Yudi Zhang, Weilun Zhao, Zhen Li, Yuxiang Huang, Ao Sun, Xu Han, and Zhiyuan Liu. InfLLM-V2: Dense-Sparse Switchable Attention for Seamless Short-to-Long Adaptation. arXiv preprint arXiv:2509.24663, arXiv, September 2025. URL `http://arxiv.org/abs/2509.24663`.

Youpeng Zhao, Di Wu, and Jun Wang. ALISA: Accelerating Large Language Model Inference via Sparsity-Aware KV Caching. In *2024 ACM/IEEE 51st Annual International Symposium on Computer Architecture (ISCA)*, pp. 1005–1017, June 2024. URL `https://ieeexplore.ieee.org/document/10609626`.

Xiabin Zhou, Wenbin Wang, Minyan Zeng, Jiaxian Guo, Xuebo Liu, Li Shen, Min Zhang, and Liang Ding. DynamicKV: Task-Aware Adaptive KV Cache Compression for Long Context LLMs. arXiv preprint arXiv:2412.14838, February 2025. URL `http://arxiv.org/abs/2412.14838`.

Simiao Zuo, Xiaodong Liu, Jian Jiao, Denis X. Charles, Eren Manavoglu, Tuo Zhao, and Jianfeng Gao. Efficient Hybrid Long Sequence Modeling with State Space Augmented Transformers. In *1st Conference on Language Modeling, COLM 2024*, August 2024. URL `https://openreview.net/forum?id=uUIFTjBREk`.

Adrian Łańcucki, Konrad Staniszewski, Piotr Nawrot, and Edoardo M. Ponti. Inference-Time Hyper-Scaling with KV Cache Compression. arXiv preprint arXiv:2506.05345, June 2025. URL `http://arxiv.org/abs/2506.05345`.

A Implementation details

We train our models using Flame (Zhang & Yang, 2025) and refer to their training recipes, while adapted to our computational resources available. All the models are implemented based on Flash Linear Attention (FLA). Particularly, FLA implements the prefilling kernel for NSA, and we develop the decoding kernel based on it. It is noteworthy that NSA implemented by FLA is simplified by directly using mean pooling to produce the compressed $\boldsymbol{k}_m^B$ and $\boldsymbol{v}_m^B$. We also develop the LTE prefilling kernel upon the FLA attention kernel. Both the 0.4B and 1.4B models have 24 layers, with hidden size = 1024 and 2048 respectively. As for GDN layers, we use 8 heads and expand_v = 1. As for attention layers, we set number of heads $H_q = H_{kv} = 16$ at 0.4B, and grouped-query attention (Ainslie et al., 2023a) with $H_q = 32, H_{kv} = 8$ at 1.4B. Similar configurations are also adopted by Wang et al. (2025). The exception is that NSA requires at least 16 group size due to hardware constraints, hence we use $H_q = 32, H_{kv} = 2$. Token embeddings are tied on 0.4B models. The LTE module consists of a 3-layer 1D grouped convolution stack, using kernel size = 3 and dilation = 2, yielding a receptive field $R = 13$. Each layer halves the channel width. Each layer is followed by a Swish activation and a dropout layer with $p = 0.1$. It then passes linear projection per head implemented as grouped convolution with kernel size = 1, and produces the retention score r after sigmoid. We implement directly with Conv2D to avoid tensor manipulation overheads. The number of groups equals the number of KV heads to process each head independently and efficiently. The LTE module will add ∼1.7% and ∼0.7% parameters on 0.4B and 1.4B models respectively.

We use the AdamW optimizer with learning rate = 3e-4 under cosine decay. At 0.4B/1.4B, we use global batch sizes of 0.5M/1.0M tokens, context length 4096, for 20480/30720 steps with 1024/512 warmup steps. As for LTE training, we initialize the regularization weight matrix λ to 1e-9, and update it via a feedback loop every $u = 32$ steps for stability, following the pseudocode given in Algorithm 1. The update is based on the exponential moving average $\bar{c}$ of the retention count c, with coefficient $\alpha_{ema} = 2/(1 + u/2)$ to approximate the average within the update cycle.

Algorithm 1 Pseudo-code for the LTE training loop.

```
 1: $\lambda_{:} \leftarrow 10^{-9}$                    ▷ Initialize $\lambda$ matrix (shaped [#layers, #heads])
 2: **for** step $\leftarrow 0$ to $T$ **do**                    ▷ $T$ is total training steps
 3:     $r, c, \ell \leftarrow$ model(fetch_inputs())          ▷ Retention score, retention count, and LM loss
 4:     $\ell \leftarrow \ell + \lambda \cdot \mathrm{ReLU}(r)$          ▷ Augment loss with L1 penalty on $r$
 5:     optimizer.step()
 6:     $\bar{c} \leftarrow \alpha_{ema} \cdot \bar{c} + (1 - \alpha_{ema}) \cdot c$     ▷ Exponential moving average of retained token counts
 7:     **if** step mod $u = 0$ **then**                ▷ Every $u$ steps, update $\lambda$
 8:         Synchronize $\bar{c}$ across devices
 9:         $s \leftarrow \mathbf{1}(\bar{c} > L) - \mathbf{1}(\bar{c} < 0.95 \cdot L)$     ▷ Direction to adjust $\lambda$
10:         $\lambda \leftarrow \lambda \cdot \alpha^{s}$
11:         $\lambda \leftarrow \min(\lambda, 1.0)$                ▷ Clamp $\lambda$ to at most 1.0
12:         $\lambda[\lambda < 10^{-9}] \leftarrow 0$              ▷ Eliminate near-zero values
13:         $\lambda[\bar{c} > L \;\&\; \lambda = 0] \leftarrow 10^{-9}$     ▷ Re-enable penalty when too many retained
14:     **end if**
15: **end for**
```

During LTE decoding, we directly use the `flash_attention_kv_cache` API by Flash Attention, which supports different past KV cache length per sample in a batch by passing in `cache_seqlens`. Since we have different cache length per head, we merge the dimensions for KV head and batch size, leaving a KV cache with the number of query heads equals to the GQA group size. As for evaluation, we use LM-eval-harness for short-context and RULER evaluations, and leverage the reference implementation by Arora et al. (2024c) for EVAPORATE benchmarks. We follow their input truncation scheme, but only truncate the input length to 4K tokens when it exceeds 4K to better reflect long-context performance. Inference is carried out using greedy decoding in FP32 for stability, except that GDN and Flash Attention kernels support FP16 only, hence we perform all token-mixing ops in FP16 for parity. Training-free sparse attention mechanisms including Unif.+SWA and TOVA are evaluated by applying them to pretrained GDN + Attn. We use an identical 1024 total cache size for both of them. For compute-cost measurements, we report the total elapsed time between the CUDA event pairs at layer entry after a full warm-up phase; for prefilling we average the middle three of five runs. Samples from the training dataset are used as inputs.

B Retention pattern

Figure 5 shows the average retention rate per head per layer from the out-of-SWA part of 16 samples drawn from the training data produced by our l.4B laLTE model. We find that retention rates are highly varied across layers and heads, while the higher layers are often denser. This is similar to findings in other works, e.g. Łańcucki et al. (2025), except that dense heads do not present in lowest

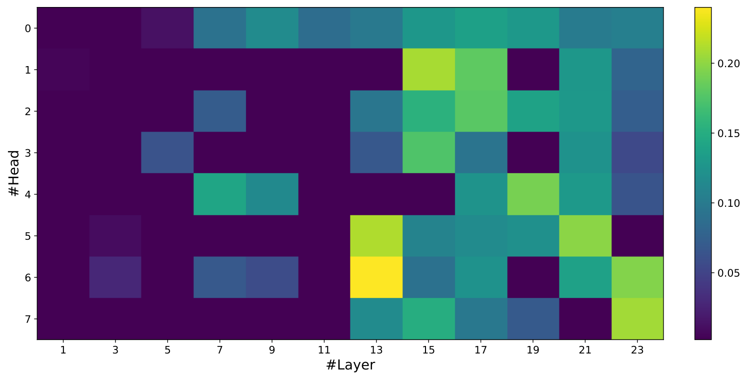


Figure 5: Average retention rate per head per layer produced by a laLTE model.

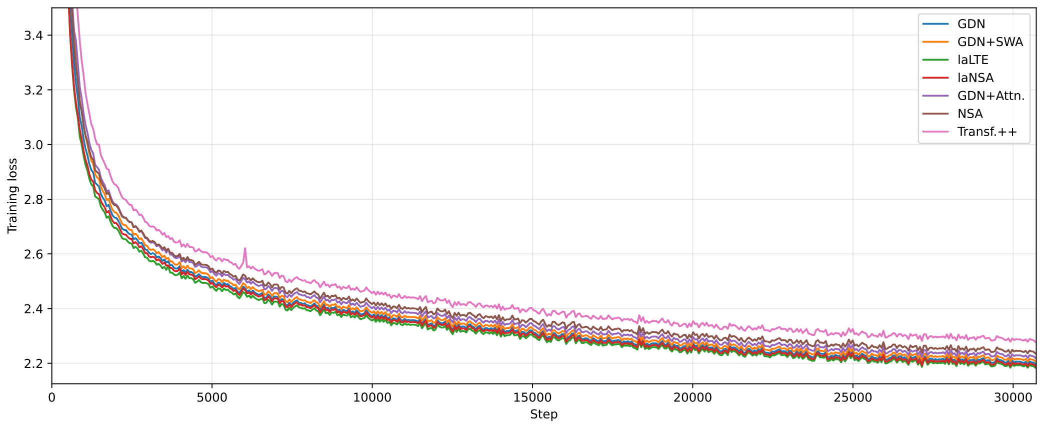


Figure 6: Training loss curve of 1.4B models.

layers; The interleaved GDN layers may have a role in it. The observation also supports the necessity of having a fine-grained cache budget different from head to head and from layer to layer.

C Loss curves

Figure 6 shows the loss curves of our 1.4B models, which match our expectation, observations in previous works, and short-context evaluations: both NSA and GDN have lower perplexity than pure transformers, while hybrid models (specifically, laLTE and laNSA) will further improve the results, though the training loss doesn't indicate long-context capabilities.