Overcoming Long-Context Limitations of State-Space Models via Context-Dependent Sparse Attention

Zhihao Zhan [1 2] **Jianan Zhao** [1 2] **Zhaocheng Zhu** [1 2] **Jian Tang** [1 3 4]

Abstract

Efficient long-context modeling remains a critical challenge for natural language processing (NLP), as the time complexity of the predominant Transformer architecture scales quadratically with the sequence length. While state-space models (SSMs) offer alternative sub-quadratic solutions, they struggle to capture long-range dependencies effectively. In this work, we focus on analyzing and improving the long-context modeling capabilities of SSMs. We show that the widely used synthetic task, associative recall, which requires a model to recall a value associated with a single key without context, insufficiently represents the complexities of real-world long-context modeling. To address this limitation, we extend the associative recall to a novel synthetic task, *joint recall*, which requires a model to recall the value associated with a key given in a specified context. Theoretically, we prove that SSMs do not have the expressiveness to solve multi-query joint recall in sub-quadratic time complexity. To resolve this issue, we propose a solution based on integrating SSMs with Context-Dependent Sparse Attention (CDSA), which has the expressiveness to solve multi-query joint recall with sub-quadratic computation. To bridge the gap between theoretical analysis and real-world applications, we propose locality-sensitive Hashing Attention with sparse Key Selection (HAX), which instantiates the theoretical solution and is further tailored to natural language domains. Extensive experiments on both synthetic and real-world long-context benchmarks show that HAX consistently outperforms SSM baselines and SSMs integrated with context-independent sparse attention (CISA).

[1]Mila - Québec AI Institute [2]University of Montréal [3]HEC Montréal [4]CIFAR AI Chair. Correspondence to: Jian Tang <tangjian@mila.quebec>.

Proceedings of the 42nd International Conference on Machine Learning, ES-FoMo III: 3rd Workshop on Efficient Systems for Foundation Models, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

1. Introduction

Long-context modeling is a central challenge in natural language processing (NLP), which underpins a variety of applications, such as document summarization, question answering, and machine translation (Pawar et al., 2024). Recent advances in large language models (LLMs) have broadened the landscape of long-context modeling, enabling new capabilities such as autonomous agents, retrieval-augmented generation, dialogue systems, and long-context reasoning (Liu et al., 2025). This growing demand has spurred intensive research into algorithms that can efficiently and effectively capture long-range dependencies (Tay et al., 2022).

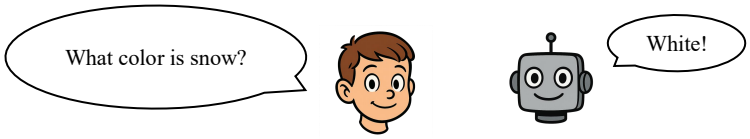
Currently, the Transformer architecture (Vaswani et al., 2017) is the dominant paradigm in sequence modeling. However, its applicability to long sequences is fundamentally constrained by the required computation that grows quadratically with the sequence length. This motivates the research direction for the invention of efficient architectures.

Recently, state-space models (SSMs) (Fu et al., 2023; Gu et al., 2022; Poli et al., 2023) have emerged as a potential alternative solution, offering sub-quadratic time complexity as well as comparable performance to Transformers on short-context NLP tasks (Gu & Dao, 2024; Dao & Gu, 2024). However, empirical evidence suggests that SSMs are less effective than Transformers in capturing long-range dependencies (Waleffe et al., 2024). Furthermore, theoretical analysis by Jelassi et al. (2024) demonstrates that SSMs are much less capable of handling long-context copying, due to the limitations of architecture representation capacity.

In this work, we aim to better understand and improve the long-context modeling abilities of SSMs. We first show that previous studies based on the widely used synthetic task, associative recall (Ba et al., 2016), might be constrained by its limited capability to simulate natural language in-context dependencies. To be specific, associative recall assumes that each key is uniquely associated with a value, regardless of the surrounding context. However, natural language often defies this assumption: the same key can correspond to different values depending on its context. Consider the example in Fig. 1, when asked on which side of the road people drive, the correct answer should depend on

Associative Recall does *not* specify context.

✓ The oceans are blue, and snow is white.



Joint Recall specifies context.

✓ In the U.S., people pay with dollars and drive cars on the right side of the road.
✓ In the U.K., people pay with pounds and drive cars on the left side of the road.

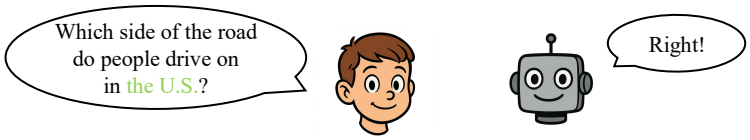


Figure 1: Comparison of joint recall and associative recall. Associative recall does not account for context. *Joint recall* extends associative recall by incorporating *context-dependency* into key-value associations. For example, while associative recall may map "pay with" to either "dollar" or "pound", joint recall allows it to map to "dollar" in the U.S. and "pound" in the U.K., depending on context. This makes joint recall a more realistic and rigorous synthetic task for both theoretical analysis and empirical benchmark for long-context modeling.

the country being referenced. Without specifying whether the context is the US or the UK, the question becomes ambiguous. This example highlights a critical shortcoming of associative recall: it lacks the capacity to simulate context-dependent key-value association, which is very common in natural language.

To address this limitation, we extend the associative recall to a more general synthetic task, joint recall. Unlike associative recall, joint recall requires the model to retrieve a value corresponding to a key conditioned on a specified context. Theoretically, we prove that standard SSMs lack the representational capacity to solve multi-query joint recall under sub-quadratic time complexity.

To overcome this expressiveness bottleneck, we propose to augment SSMs with Context-Dependent Sparse Attention (CDSA), a class of sparse attention with sparse attention patterns that are conditioned on the context representations. Locality-sensitive hashing (LSH) attention (Kitaev et al., 2020) exemplifies CDSA, while context-independent sparse attention (CISA) includes sliding window attention, A-shaped attention, and dilated attention (Ding et al., 2023a). Compared to CISA, CDSA enables dynamic content-dependent routing of information, which is essential to efficiently solve the multi-query joint recall task. We theoretically show that there exists a CDSA which, when integrated with SSMs, enables solving the multi-query joint recall task in sub-quadratic time with respect to sequence length. Moreover, we establish an expressiveness gap between SSMs integrated with CDSA and SSMs integrated with CISA on multi-query joint recall.

Building upon this insight and to bridge the gap between theory and practice, we propose a novel architecture: locality-sensitive Hashing Attention with sparse Key Selection (HAX). HAX improves the expressiveness of LSH attention by incorporating our proposed Key Selection (KS) attention, and is further integrated with state-of-the-art SSMs, Mamba and Mamba2 (Gu & Dao, 2024; Dao & Gu, 2024), instantiating the theoretically grounded solution. We validate the effectiveness of HAX through extensive experiments on both synthetic and real-world long-context modeling benchmarks. The experiment results show that HAX consistently outperforms SSM baselines as well as SSMs augmented with CISA. These findings demonstrate that CDSA, when carefully integrated with SSMs, is a critical component in unlocking their potential for long-context modeling.

Our main contributions are summarized as follows:

1. We introduce *joint recall*, a novel synthetic task that extends associative recall to context-dependent key-value association, which offers a new perspective for both theoretical analysis and empirical benchmark for long-context modeling.
2. Through theoretical analysis on *joint recall*, we demonstrate that integrating state-space models (SSMs) with context-dependent sparse attention (CDSA) has the expressiveness to solve multi-query joint recall with sub-quadratic computation.
3. Guided by this theoretical insight, we propose a novel architecture, HAX, based on SSM integrated with CDSA, which consistently outperforms SSMs and SSMs integrated with context-independent sparse attention (CISA) on both synthetic and real-world long-context benchmarks.

2. Preliminaries

In this section, we introduce two prominent approaches for efficient architecture design: sparse attention in Sec. 2.1 (exemplified by LSH attention in Sec. 2.2) and SSMs in Sec. 2.3. We also introduce associative recall, a widely-used synthetic benchmark for long-context modeling evaluation, in Sec. 2.4.

2.1. Sparse Attention

For a sequence of length l, we denote the attention scores of auto-regressive sequence modeling as:

$$\mathbf{A} = \text{Softmax}(\mathbf{M} \odot \mathbf{Q}\mathbf{K}^\top) \tag{1}$$

where $\mathbf{M} \in \{0,1\}^{l\times l}$ is the auto-regressive mask, $\mathbf{Q}, \mathbf{K} \in \mathbb{R}^{l\times d}$, d is the hidden dimension. In this work, we define the attention scores for sparse attention as:

$$\mathbf{A} = \text{Softmax}(\mathbf{S} \odot \mathbf{M} \odot \mathbf{Q}\mathbf{K}^\top) \tag{2}$$

with $\mathbf{S} \in \{0,1\}^{l\times l}$ representing the sparse attention pattern. To enforce sparsity, we impose the constraint

$$\|\mathbf{S}\|_0 \ll l^2 \tag{3}$$

To ensure per-step computational efficiency, we further tighten this constraint by requiring

$$\forall i, \|\mathbf{S}_i\|_0 \ll l \tag{4}$$

where $\mathbf{S}_i$ denotes the i-th row of $\mathbf{S}$. This implies that each query attends to at most k keys, $k \ll l$.

2.2. Locality-Sensitive Hashing Attention

Given that locality-sensitive hashing (LSH) attention represents one of the most effective input-dependent sparse attention for auto-regressive modeling (Kitaev et al., 2020), we reformulate a simple algorithm to generate the sparse attention pattern of LSH. This algorithm accepts the query and key matrices $\mathbf{Q}$ and $\mathbf{K}$ as input and outputs a binary sparse attention pattern $\mathbf{S}_{\text{LSH}}$. At each forward pass, $\mathbf{Q}$ and $\mathbf{K}$ are first centralized and normalized to $\tilde{\mathbf{Q}}$ and $\tilde{\mathbf{K}}$, respectively:

$$\tilde{\mathbf{Q}}_i = \text{normalize}(\mathbf{Q}_i - \bar{\mathbf{Q}}_i) \tag{5}$$

$$\tilde{\mathbf{K}}_i = \text{normalize}(\mathbf{K}_i - \bar{\mathbf{K}}_i) \tag{6}$$

Next, a random projection matrix $\mathbf{H} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0,1) \in \mathbb{R}^{d\times h}$ is sampled to project the normalized vectors $\tilde{\mathbf{Q}}$ and $\tilde{\mathbf{K}}$ into the hash space. Then, we consider two binning rules which assign each vector $\tilde{\mathbf{Q}}_i$ and $\tilde{\mathbf{K}}_i$ to a discrete hash bin: the argmax binning rule and the sign-bit binning rule.

The argmax binning rule assigns each vector to the index of its most aligned column in $\mathbf{H}$ by dot product:

$$\text{bin}_{Q_i} = \text{argmax}(\tilde{\mathbf{Q}}_i\mathbf{H}), \quad \text{bin}_{K_i} = \text{argmax}(\tilde{\mathbf{K}}_i\mathbf{H}) \tag{7}$$

The sign-bit binning rule constructs a binary hash code by computing the signs of the projected values and interpreting it as a binary number (Ding et al., 2024):

$$\text{bin}_{Q_i} = \sum_{j=1}^{h} \mathbf{1}[(\tilde{\mathbf{Q}}_i\mathbf{H})_j > 0] \cdot 2^{h-j} \tag{8}$$

$$\text{bin}_{K_i} = \sum_{j=1}^{h} \mathbf{1}[(\tilde{\mathbf{K}}_i\mathbf{H})_j > 0] \cdot 2^{h-j} \tag{9}$$

The argmax binning rule assigns vectors to h bins, while the sign-bit binning rule assigns vectors to 2^h bins. We will further discuss the relationship between these two binning strategies in Appendix A. Based on the assigned bins, a preliminary sparse pattern $\tilde{\mathbf{S}}_{\text{LSH}}$ is constructed by allowing each query to attend to all preceding keys within the same bin:

$$\tilde{\mathbf{S}}_{\text{LSH}_{ij}} = \mathbf{1}[\text{bin}_{Q_i} = \text{bin}_{K_j}] \tag{10}$$

Finally, to satisfy the sparsity constraint defined in Eq. 4, a per-bin sliding window mask $\mathbf{M}_{\text{LSH}}$ is applied, so that each query only attends to at most k nearest keys in the same bin:

$$\mathbf{S}_{\text{LSH}} = \mathbf{M}_{\text{LSH}} \odot \tilde{\mathbf{S}}_{\text{LSH}} \tag{11}$$

2.3. Generalized State-Space Model

Following the definitions introduced by Jelassi et al. (2024), we formulate generalized state-space models as sequence models defined by an update rule $u : \mathcal{U} \times \mathcal{V} \to \mathcal{U}$ and an output function $r : \mathcal{U} \to \mathcal{V}$, where $\mathcal{V}$ denotes the token vocabulary and $\mathcal{U}$ represents the recurrent state. Let $U_0(\varnothing) \in \mathcal{U}$ denote the initial state. Given an input sequence $v_1, ..., v_n \in \mathcal{V}$, for i in $\{1...n\}$, the state $U_i(v_1, ..., v_i) \in \mathcal{U}$ and its corresponding output $R_i(v_1, ..., v_i) \in \mathcal{V}$ are defined recursively as:

$$U_i(v_1, ..., v_i) = u(U_{i-1}(v_1, ..., v_{i-1}), v_i) \tag{12}$$

$$R_i(v_1, ..., v_i) = r(U_i(v_1, ..., v_i)) \tag{13}$$

2.4. Associative Recall

The associative recall task was originally introduced in Ba et al. (2016). Olsson et al. (2022) found that the LLM performance on this task is strongly correlated with their in-context learning abilities . Arora et al. (2024) extended associative recall to the multi-query setting: a model is first given a sequence of associated key-value pairs, and then required to recall each value when queried with the corresponding key. Associative recall has been widely adopted as a synthetic benchmark for long-context modeling (Dao & Gu, 2024; Hsieh et al., 2024).

3. Joint Recall

We discuss the motivation and formulation of joint recall in Sec. 3.1 and Sec. 3.2, respectively, and finally provide the theoretical results in Sec. 3.3.

3.1. Motivation

The motivation behind proposing joint recall is to overcome a key limitation of the setup of associative recall: each key corresponds to a single fixed value. While this setup is well-suited for studying the tasks that emphasize capturing stable lexical patterns, such as sub-word units or fixed multi-word expressions, it falls short in representing the context-sensitive nature of meaning in natural language. Consider the following examples:

- The legislative branch of the U.S. government is called Congress.
- On Monday mornings, Alice studies math.

From a philosophical perspective, definitions are often constructed using genus keys and differentia context. In the first example, the value "Congress" is identified with the genus key "the legislative branch" and the differentia context "of the U.S. government". The second example reflects a more daily scenario, where the value "math" is identifiable only when all contextual elements, "Monday" and "morning", are considered together with the key "Alice". These cases illustrate that accurate semantic interpretation in natural language often requires integrating context and keys, suggesting that models must move beyond the simplistic one-to-one mappings of associative recall to capture the compositional and context-dependent nature of meaning. This observation motivates our introduction of a novel synthetic task, which we refer to as joint recall.

3.2. Formulation

Associative recall requires a model to memorize n_k associated key-value pairs. Joint recall generalizes this task: the model is required to memorize an $n_c \times n_k$ table of context-specific key-value associations, in which n_k keys are associated with different values in each of the n_c contexts. Inspired by Arora et al. (2024), we also extend joint recall to a multi-query setting, requiring the model to recover the entire table instead of a specific entry in the table.

Fig. 2 illustrates multi-query joint recall with $n_c = 2$ and $n_k = 2$. Following the structure of natural language, the sequentialized table input consists of n_c context blocks, each beginning with a context token (e.g. uppercase letter in Fig. 2), followed by n_k key-value pairs specific to that context (e.g. lowercase-letter-digit pairs in Fig. 2). Then, the model is tasked with recalling the associated values given each context-key pair, under arbitrary permutations of the context and key ordering.

Multi-Query Associative Recall recalls each value associated with a single key without context:

Multi-Query Associative Recall	
Input	a 5 b 2 c 3 d 1 e 4 | c ? e ? a ? d ? b ?
Output	3 4 5 1 2

Multi-Query Joint Recall recalls each value associated with a key given in a specified context:

Multi-Query Joint Recall	
Input	A a 3 b 2 B b 4 a 1 | B a ? b ? A b ? a ?
Output	1 4 2 3

Figure 2: Comparison between the synthetic formulation of *multi-query joint recall* and multi-query associative recall.

Appendix C further extends the joint recall formulation to multi-level context: for example, in the sentence "On Monday mornings, Alice studies math.", “Monday” and “morning” are contexts at hierarchical levels. It also provides theoretical analyses grounded in this extended formulation.

3.3. Theoretical Analysis

3.3.1. Categorization of Sparse Attention

The sparse attention pattern $\mathbf{S}$ defined in Sec. 2.1 can be categorized as context-dependent or context-independent, depending on whether it is predetermined or dynamically inferred from the context representations. Context-independent sparse attention (CISA) patterns, such as sliding window attention, A-shaped attention, and dilated attention (Ding et al., 2023a), are fixed regardless of context. In contrast, context-dependent sparse attention (CDSA) patterns, exemplified by LSH attention (Kitaev et al., 2020), adapt according to the context representation. Appendix Fig. 8 provides an illustration of both categories.

3.3.2. Limited Expressiveness of SSMs

As a corollary of Theorem 2.7 in Jelassi et al. (2024), we demonstrate that solving the multi-query joint recall task imposes a linear growth requirement on the state dimension of SSMs with respect to the number of entries n in the joint recall table, $n = n_c \times n_k$. Let $|\mathcal{U}|$ be the number of distinct representations that the recurrent state space $\mathcal{U}$ can encode, for a state with b bits of capacity, $|\mathcal{U}| = 2^b$. We define the uniform multi-query joint recall distribution as the distribution in which all values are sampled i.i.d. from the uniform distribution over the token vocabulary $\mathcal{V}$. In this setting, we obtain the following:

Corollary 3.1 (Limited expressiveness of SSMs)**.** *Under the uniform multi-query joint recall distribution, for any n, a generalized state-space model defined in Sec. 2.3 incurs an error rate of at least $1 - \frac{|\mathcal{U}|}{|\mathcal{V}|^n}$.*

Remark 3.1. To guarantee $\Pr[err] = 0$, it is necessary that the number of representable states satisfies $|\mathcal{U}| \geq |\mathcal{V}|^n$. Taking the logarithm of both sides yields the condition $b \geq n \log |\mathcal{V}|$. This implies that the state-space dimension of the model must grow linearly with the number of entries n in the joint recall table, highlighting a fundamental limitation of the representation capacity of SSMs.

3.3.3. Improved expressiveness of SSMs integrated with CDSA

For SSMs integrated with sparse attention, we establish the following results:

Proposition 3.2 (Improved expressiveness of SSMs integrated with CDSA)**.** *There exists a 2-layer auto-regressive hybrid model consisting of an SSM layer followed by an LSH attention layer, which can solve multi-query joint recall in $O(n \log^2 n)$ time complexity with $O(\log n)$ SSM state dimensions.*

Proposition 3.3 (Limited expressiveness of SSMs integrated with CISA)**.** *There does not exist a 2-layer auto-regressive hybrid model consisting of an SSM layer followed by a CISA layer, which can solve multi-query joint recall with $o(n^2)$ time complexity, since it requires at least $O(\frac{n}{k})$ SSM state dimensions, k is the maximum number of keys that each query allowed to attend to in the sparse attention module, as defined in Eq. 4.*

Remark 3.2. Comparing **Proposition** 3.2 with **Proposition** 3.3, we see a clear representation capacity gap between the SSMs integrated with CDSA and the SSMs integrated with CISA.

Remark 3.3. In practice, with an appropriate constant k, integrating CISA with SSMs still provides an advantage: unlike SSMs, which only have access to the last state representation, CISA layers can attend to k different state representations simultaneously, at a cost of k times of computation budget.

Complete proofs are provided in the Appendix B.

4. Method

Guided by the theoretical analysis in Sec. 3.3, we propose a new architecture, locality-sensitive Hashing Attention with sparse Key Selection (HAX). HAX improves the expressiveness of LSH attention by incorporating our proposed Key Selection (KS) attention, and is further integrated with state-of-the-art SSMs, serving as an instantiation of context-dependent sparse attention (CDSA) integrated with SSMs, thereby benefiting from the theoretical advantages discussed in Sec. 3.3.

In this section, we first discuss the expressiveness limitations of LSH attention in Sec. 4.1, and then address these limitations by introducing our proposed key selection (KS) attention in Sec. 4.2. Finally, Sec. 4.3 details the architecture of HAX as well as how HAX is integrated with state-of-the-art SSM architectures, Mamba and Mamba2 (Gu & Dao, 2024; Dao & Gu, 2024).

4.1. Limitation of LSH Attention

In LLMs, certain keys (particularly those at the beginning of a sequence) often receive attention from most queries, forming distinctive "vertical-stripe" attention patterns (Vig & Belinkov, 2019), as illustrated in Appendix Fig. 9. These globally attended keys play an essential role in instruction following, where the model is expected to focus its attention on the instruction tokens (Liu et al., 2024).

Although LSH instantiates CDSA, as discussed in Sec. 2.2, it suffers from a key limitation: difficulty in capturing "vertical-stripe" attention patterns. This arises because in each hashing round, every key is mapped to a single bucket, and attention is constrained to occur only between queries and keys within the same bucket. As a result, when many queries are forced to attend to a limited set of key buckets, those buckets become overloaded, diminishing representation diversity and ultimately degrading attention quality.

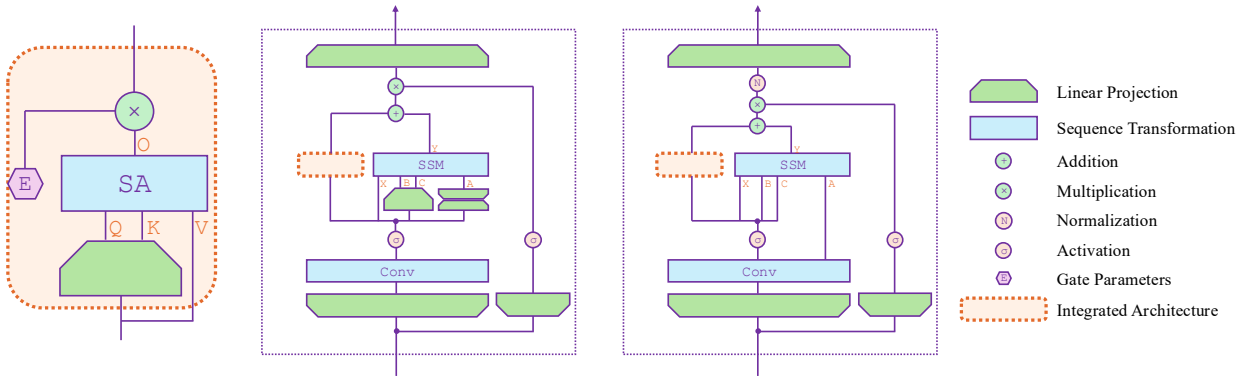
4.2. Key Selection (KS) Attention

Goals. To address the limitation of LSH attention in capturing "vertical-stripe" attention patterns, we propose to augment LSH attention by integrating it with a novel key selection (KS) attention module. This module is designed to satisfy the following desirable properties:

1. "Vertical-stripe" capability: KS attention can express "vertical-stripe" attention patterns.
2. Auto-regressive compatibility: The computation of KS attention for the current token does not depend on future queries or keys.
3. Context-dependent sparsity: KS sparse attention pattern is conditioned on the query and key representations in context and satisfies Eq. 4.

Modeling. Taking the query and the key matrices $\mathbf{Q}$ and $\mathbf{K}$ as input, KS attention operates in two phases. The first phase is key scoring, where an scoring module computes an importance score for each key based on the key itself and all previous queries:

$$x_i = f_\theta(\mathbf{K}_i, \mathbf{Q}_{1\ldots i}) \tag{14}$$



Integrated Architecture Integrated Mamba Block Integrated Mamba2 Block

Figure 3: The hybrid architectures of HAX integrated to Mamba (Gu & Dao, 2024) and Mamba2 (Dao & Gu, 2024). "SA" is short for "sparse attention", which is based on the sparse attention pattern defined in Eq. 21.

The second phase is key selection: each query attends to the k previous highest-scoring keys,

$$\mathbf{S}_{\mathrm{KS}_{ij}} = \mathbf{1}[x_j \in \text{Top-}k\{x_1, ..., x_i\}] \tag{15}$$

With an ideal key scoring module that assigns the highest scores to the globally important keys, KS attention effectively covers k "vertical-stripes" within attention patterns.

For simplicity, we use a multilayer perceptron (MLP) as the key scoring network:

$$f_\theta(\mathbf{K}_i, \mathbf{Q}_{1..i}) \triangleq \mathrm{MLP}\big(\mathbf{K}_i, \mathrm{normalize}\big(\sum_{1\le p\le i} \mathbf{Q}_p\big)\big) \tag{16}$$

Training. To train the scoring MLP, at each layer, we randomly sample k candidate keys. Their indices are denoted by $\mathcal{I}$. We compute the reference attention weights via a simple linear attention module, and calculate a pairwise ranking loss between these reference weights and the predicted scores. To be specific, we compute:

$$\mathbf{A}' = \mathbf{Q}\,\mathbf{K}[\mathcal{I}]^\top, \qquad y = \sigma(\mathbf{A}') \odot \mathbf{M}[\mathcal{I}], \tag{17}$$

where $\mathbf{K}[\mathcal{I}]$ is the selected key representations, $\mathbf{M}[\mathcal{I}]$ is the auto-regressive mask restricted to those positions, and $\sigma(\cdot)$ is the sigmoid function. With predicted scores $x \in \mathbb{R}^k$ and target scores $y \in \mathbb{R}^k$, we construct pairwise logits and targets:

$$\mathbf{P}_{ij}(x) = x_i - x_j, \qquad \mathbf{T}_{ij}(y) = \begin{cases} 1 & \text{if } y_i > y_j, \\ 0.5 & \text{if } y_i = y_j, \\ 0 & \text{if } y_i < y_j. \end{cases} \tag{18}$$

and define the ranking loss:

$$\mathcal{L}_{\text{score}}(x, y) = \frac{1}{k^2} \sum_{i,j} \mathrm{BCE}\left(\mathbf{P}_{ij}(x), \mathbf{T}_{ij}(y)\right), \tag{19}$$

where $\mathrm{BCE}(\cdot,\cdot)$ denotes binary cross-entropy. This objective encourages the scoring network to assign higher scores to informative keys that receive higher attention weights. The final training objective sums the ranking loss across layers with the auto-regressive language modeling loss $\mathcal{L}_{\mathrm{LM}}$:

$$\mathcal{L} = \mathcal{L}_{\mathrm{LM}} + \alpha \sum_{\text{layers}} \mathcal{L}_{\text{score}} \tag{20}$$

where α is a scalar hyperparameter that balances the contribution of the ranking loss.

4.3. Hybrid Block Design

Finally, we propose locality-sensitive Hashing Attention with sparse Key Selection (HAX), which combines LSH and KS attention patterns:

$$\mathbf{S}_{\mathrm{HAX}} = \max\{\mathbf{S}_{\mathrm{LSH}}, \mathbf{S}_{\mathrm{KS}}\} \in \{0,1\}^{l\times l} \tag{21}$$

When $\forall i, \|\mathbf{S}_{\mathrm{LSH}_i}\|_0 \le \frac{k}{2}, \|\mathbf{S}_{\mathrm{KS}_i}\|_0 \le \frac{k}{2}$, it satisfies

$$\forall i, \|\mathbf{S}_{\mathrm{HAX}_i}\|_0 \le k \tag{22}$$

Intuitively, LSH and KS attention are complementary, each addressing the other's limitations. LSH attention routes each query to semantically similar keys through randomized hashing, offering flexible, content-based interactions that KS attention alone lacks. In contrast, KS attention introduces broadcast connections to a small set of globally important keys, such as instructions or formatting markers, thereby recovering the "vertical-stripe" patterns that LSH attention struggles to express. While LSH attention promotes diverse contextual representations, mitigating risks of representation collapse, KS attention sharpens focus by allocating attention weights to the most informative positions, enabling stronger long-range control. Importantly, both mechanisms are inherently sparse, so their combination introduces sub-quadratic computational cost.

Figure 3 illustrates Mamba-based and Mamba2-based HAX layer. The proposed hybrid sparse attention layer mitigates the representation capacity limitations of SSMs by coupling them with a parallel sparse attention branch. A parameterized gate rescales the sparse attention output before fusion, which promotes stable optimization.

5. Experiments

5.1. Empirical Verification on Multi-Query Joint Recall

We conduct experiments from two perspectives. First, we empirically verify the theoretical findings presented in Sec. 3 on multi-query joint recall. Then, we demonstrate the effectiveness of the proposed hybrid sparse architecture across synthetic and real-world NLP benchmarks.

Data. We construct a multi-query joint recall dataset in which the number of context blocks and the number of keys per context are independently sampled from the range $[5, 16]$, and the size of the vocabulary is fixed at $|\mathcal{V}| = 16$. The dataset consists of 1M training examples, along with 10K samples each for validation and testing.

Baselines. We adopt Mamba (Gu & Dao, 2024) and Mamba2 (Dao & Gu, 2024) as base architectures and evaluate various hybrid sparse attention models built upon them, as illustrated in Figure 3. These include dilated attention (D), sliding window attention (SW), a combination of sliding window and dilated attention (SW+D) (Ding et al., 2023a), A-shaped attention (A), locality-sensitive hashing (LSH) attention, our proposed key selection (KS) attention, and HAX (a combination of LSH and KS attention). We also consider a Samba baseline. For fair comparison, we double the hidden size of SSMs, and fix k defined in Eq. 4 (the maximum number of keys each query can attend to) across all hybrid architectures.

Setup. For evaluation, we calculate mean accuracy per-sample and average over the testset. For each base architecture and hybrid sparse attention variant, we conduct experiments using 3 different random seeds for each learning rate in $\{$3e-3, 1e-3, 3e-4$\}$, and report the average performance corresponding to the best-performing learning rate.

Results. The results are summarized in Table 3. Compared to the Mamba and Mamba2 base architectures, most hybrid sparse attention models show performance improvements. In particular, our proposed HAX model consistently achieves the best performance, surpassing the base architectures by more than 100%. These findings underscore the effectiveness of our approach.

5.2. Continual Pre-training on Natural Language

Setup. To evaluate our method on real-world long-context natural language data, we perform continual pre-training based on the publicly released Mamba 130M checkpoint (Gu & Dao, 2024). As in section 5.1, we augment the Mamba architecture with sparse attention, as illustrated in Figure 3. We include SSMs integrated with CISA as baselines, along with ablated variants of our HAX model.

Validation Loss during Continual Pre-Training. Figure 4 shows the validation loss $\mathcal{L}_{LM}$ of continual pre-training. As observed, the Mamba base architecture and all baseline variants exhibit either training instability or plateau early in the training process. In contrast, our proposed HAX model is the only architecture that shows a consistent decline in validation loss throughout the training process, indicating improved stability and sustained learning.

Ruler and LongBench Evaluation. Ruler (Hsieh et al., 2024) is a synthetic long-context NLP benchmark designed to assess model performance on tasks including retrieval, multi-hop reasoning, aggregation, and question answering. LongBench (Bai et al., 2024) comprises real-world NLP long-context tasks, including question answering, summarization, few-shot learning, retrieval, aggregation, and code completion. We perform instruction tuning after the continual pre-training stage following Mamba-Chat(Mattern & Hohr, 2023), and then evaluate the models after instruction tuning. The evaluation results, summarized in Tables 1 and 2, show that among all hybrid sparse attention variants, our proposed HAX model is the only one that outperforms the Mamba baseline by a significant margin on average.

6. Related Work

6.1. State-Space Models

State-space models (SSMs) originated in control theory, exemplified by damped mass-spring systems (Patro & Agneeswaran, 2024). HiPPO (Gu et al., 2020) was one of the first efforts to adapt SSMs for machine learning applications. LSSL (Gu et al., 2021) unified convolutional neural networks (CNNs), recurrent neural networks (RNNs), and ordinary differential equations (ODEs) under the SSM framework, enabling their implementation within deep neural networks. H3 (Fu et al., 2023) integrated SSM layers with short convolutional filters to enhance sequence modeling. Mamba (Gu & Dao, 2024) advanced this line of work by making all SSM parameters *input-dependent*, which significantly increases the representation capacity of SSMs. Mamba2 (Dao & Gu, 2024) further improved the architecture and established connections between SSMs and Transformer attention.

6.2. Analysis on State-Space Models

Recent empirical studies have shown that SSM long-context modeling performance often lags behind that of Transformer architectures (Waleffe et al., 2024). Jelassi et al. (2024)

Table 1: Results of Ruler benchmark. We compare different sparse attention integrated with Mamba, including CISA methods: dilated attention (D), sliding window attention (SW), and their combination (SW+D), and A-shaped attention (A), as well as CDSA methods: LSH attention, and our proposed key selection (KS) attention and HAX. The best average performance is in **bold**.

Model	NIAHS1	NIAHS2	NIAHS3	NIAHMK1	NIAHMK2	NIAHMK3	NIAHMV	NIAHMQ	VT	CWE	FWE	QA1	Average
Mamba	99.6	53.4	7.4	20.8	0	0	17.8	4.8	1.5	1.3	100	14	26.7
+D	95.0	27.6	3.2	19.8	0	0	10.1	4.6	2.8	1.6	100	12.4	23.1
+SW	100	74	8	21.6	0.4	0	20.3	7.0	1.7	1.28	100	10.6	28.7
+SW+D	99.8	54.2	6.4	22	0	0	17.6	5.0	4.2	1.3	100	14	26.9
+A	99.8	63	4.8	24	0	0	19.8	3	2.2	1.4	100	12.2	27.5
+LSH	99.8	36.2	12.2	24	0	0	12.3	3.4	4.9	1.7	100	16.3	25.9
+KS *(ours)*	99.6	58.8	14.6	23.2	0	0	20.9	11	2.2	1.1	100	11.8	28.6
+HAX *(ours)*	100	92.4	34.6	24	0.2	0	20.4	3.8	4.9	1.7	100	12.8	**32.9**

Table 2: Results of LongBench English tasks. We compare different sparse attention integrated with Mamba, including CISA and CDSA methods as in Tab. 1. The best average performance is in **bold**.

Model	2WikiMQA	GovReport	HotPotQA	LCC	MultiNews	MultiFQA	MuSiQue	NQA	PsgCnt	PsgRet	Qasper	QMSum	Repobench	SamSum	Trec	TriviaQA	Average
Mamba	6.11	15.16	3.35	34.57	16.73	12.72	2.51	3.02	0.85	0.5	4.97	16.49	35.76	1.67	10.5	12.76	11.10
+D	4.26	10.22	2.56	32.1	11.16	9.63	2.05	2.08	0.76	1.13	4.34	12.72	32.95	2.85	14.5	12.87	9.76
+SW	5.57	15.03	3.34	34.63	16.06	11.38	2.13	2.86	1.99	0.5	6.34	16.05	35.4	1.74	12	13.14	11.14
+SW+D	6.77	14.78	2.89	34.7	16.72	12.03	2.11	2.83	0.95	0.5	5.04	16.72	35.99	1.72	10.5	14.07	11.15
+A	6.58	15.12	3.26	34.77	16.33	12.26	2.2	2.96	1.69	0.42	5.47	15.58	35.5	1.75	10.5	13.55	11.12
+LSH	3.15	9.18	1.77	26.06	9.51	5.66	1.33	1.52	1.17	0.38	3.34	8.49	26.34	1.51	10.75	8.44	7.41
+KS *(ours)*	6.32	15.92	3.22	34.1	17.11	12.03	2.56	2.71	0.95	0	5.34	16.46	35.57	1.67	10.5	13.6	11.13
+HAX *(ours)*	6.71	15.47	3.34	34.81	14.95	13.64	1.76	2.6	1.64	0.3	5.54	14.22	35.47	1.89	14	15.17	**11.34**

demonstrated that even solving simple tasks like copying requires SSM state dimensions to grow linearly with the sequence length. Furthermore, Merrill et al. (2024) established that the expressiveness of SSMs is bounded by the complexity class TC^0. Sarrof et al. (2024) further showed that SSMs and Transformers capture overlapping yet distinct subsets of TC^0, providing a theoretical basis for developing hybrid models that combine the strengths of both architectures.

6.3. Hybrid Architectures

Several works have explored architectures that mix a large proportion of SSM layers with a small number of full attention layers, and have reported performance surpassing that of standard Transformers (Waleffe et al., 2024). The effectiveness of such hybrid architectures has been further validated at billion-parameter scale (Lenz et al., 2025). In parallel, researchers have also investigated the design of hybrid sparse attention models (Ren et al., 2025; Dong et al., 2025; Nunez et al., 2024), which offer sub-quadratic computational complexity, providing a promising direction for efficient long-context modeling.

7. Conclusion

In this work, we introduce *joint recall*, a novel synthetic task that generalizes associative recall to context-dependent key-value retrieval. Theoretically, we show that both SSMs and SSMs integrated with context-independent sparse attention (CISA) could not solve multi-query joint recall within sub-quadratic time complexity, while integrating SSMs with context-dependent sparse attention (CDSA) overcomes this limitation. Guided by this insight, we propose to integrate state-of-the-art SSMs with a novel CDSA, locality-sensitive Hashing Attention with sparse Key Selection (HAX). Experiment results confirm that HAX achieves improved training stability and consistent performance gains across synthetic and real-world long-context NLP benchmarks. The *joint recall* task therefore offers a unified theoretical lens and empirical yardstick for long-context modeling, while HAX demonstrates the power of theory-driven architecture design. These results highlight the importance of aligning model design with expressiveness improvements, and demonstrate that combining efficient sequence models with CDSA is a promising direction for scalable long-context modeling.

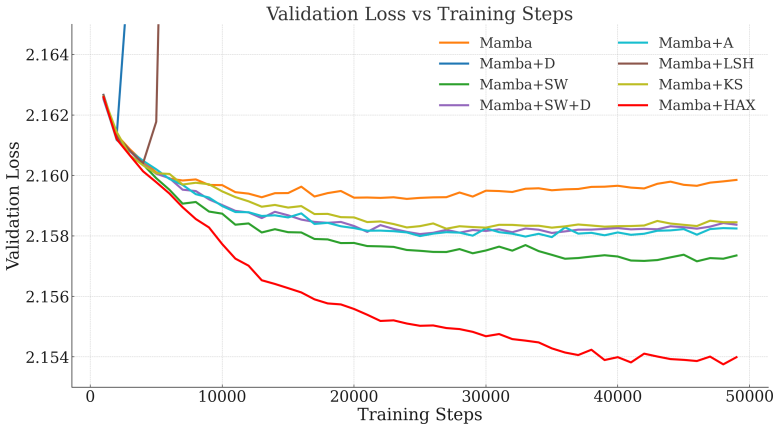


Figure 4: Validation $\mathcal{L}_{LM}$ during continual pre-training. Mamba integrated with HAX is the only architecture that consistently exhibits a decreasing validation loss over the entire training process.

Table 3: Results of multi-query joint recall. Integrating Mamba or Mamba2 with HAX achieves the best performance, which is in **bold**.

	Mamba	Mamba2
Base	16.3	36.6
+D	7.8	19.6
+SW	18.7	70.6
+SW+D	16.6	48.6
+A	16.4	49.3
+LSH	11.6	13.5
+KS *(ours)*	36.6	60.1
+HAX *(ours)*	**38.0**	**74.3**
Samba	6.3	

References

Arora, S., Eyuboglu, S., Timalsina, A., Johnson, I., Poli, M., Zou, J., Rudra, A., and Ré, C. Zoology: Measuring and improving recall in efficient language models. In *The Twelveth International Conference on Learning Representations*, 2024.

Ba, J., Hinton, G. E., Mnih, V., Leibo, J. Z., and Ionescu, C. Using fast weights to attend to the recent past. In *The Twenty-ninth Annual Conference on Neural Information Processing Systems*, 2016.

Bai, Y., Lv, X., Zhang, J., Lyu, H., Tang, J., Huang, Z., Du, Z., Liu, X., Zeng, A., Hou, L., et al. Longbench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, 2024.

Dao, T. and Gu, A. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. *arXiv preprint arXiv: 2405.21060*, 2024.

Ding, J., Ma, S., Dong, L., Zhang, X., Huang, S., Wang, W., and Wei, F. Longnet: Scaling transformers to 1,000,000,000 tokens, 2023a. URL https://arxiv.org/abs/2307.02486.

Ding, N., Chen, Y., Xu, B., Qin, Y., Zheng, Z., Hu, S., Liu, Z., Sun, M., and Zhou, B. Enhancing chat language models by scaling high-quality instructional conversations, 2023b.

Ding, N., Tang, Y., Qin, H., Zhou, Z., Xu, C., Li, L., Han, K., Heng, L., and Wang, Y. Memoryformer: Minimize transformer computation by removing fully-connected layers. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Dong, X., Fu, Y., Diao, S., Byeon, W., CHEN, Z., Mahabaleshwarkar, A. S., Liu, S.-Y., keirsbilck, M. V., Chen, M.-H., Suhara, Y., Lin, Y. C., Kautz, J., and Molchanov, P. Hymba: A hybrid-head architecture for small language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

Fu, D. Y., Dao, T., Saab, K. K., Thomas, A. W., Rudra, A., and Ré, C. Hungry hungry hippos: Towards language modeling with state space models. In *The Eleventh International Conference on Learning Representations*, 2023.

Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., Presser, S., and Leahy, C. The pile: An 800gb dataset of diverse text for language modeling, 2021. URL https://arxiv.org/abs/2101.00027.

Gao, L., Tow, J., Abbasi, B., Biderman, S., Black, S., DiPofi, A., Foster, C., Golding, L., Hsu, J., Le Noac'h, A., Li, H., McDonell, K., Muennighoff, N., Ociepa, C., Phang, J., Reynolds, L., Schoelkopf, H., Skowron, A., Sutawika, L., Tang, E., Thite, A., Wang, B., Wang, K., and Zou, A. A framework for few-shot language model evaluation, 2024.

Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., Yang, A., Fan, A., Goyal, A., Hartshorn, A., Yang, A., Mitra, A., Sravankumar, A., Korenev, A., Hinsvark, A., Rao, A., Zhang, A., Rodriguez, A., Gregerson, A., Spataru, A., Roziere, B., Biron, B., Tang, B., Chern, B., Caucheteux, C., Nayak, C., Bi, C., Marra, C., McConnell, C., Keller, C., Touret, C., Wu, C., Wong, C., Ferrer, C. C., Nikolaidis, C., Allonsius, D., Song, D., et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Gu, A. and Dao, T. Mamba: Linear-time sequence modeling with selective state spaces. In *The First Conference on Language Modeling*, 2024.

Gu, A., Dao, T., Ermon, S., Rudra, A., and RRé, C. Hippo: Recurrent memory with optimal polynomial projections. In *The Thirty-fourth Annual Conference on Neural Information Processing Systems*, 2020.

Gu, A., Johnson, I., Goel, K., Saab, K., Dao, T., Rudra, A., and RRé, C. R. Combining recurrent, convolutional, and continuous-time models with linear state-space layers. In *The Thirty-fifth Annual Conference on Neural Information Processing Systems*, 2021.

Gu, A., Goel, K., and Ré, C. Efficiently modeling long sequences with structured state spaces. In *The Tenth International Conference on Learning Representations*, 2022.

Hsieh, C.-P., Sun, S., Kriman, S., Acharya, S., Rekesh, D., Jia, F., Zhang, Y., and Ginsburg, B. Ruler: What's the real context size of your long-context language models? In *The First Conference on Language Modeling*, 2024.

Jelassi, S., Brandfonbrener, D., Kakade, S. M., and Malach, E. Repeat After Me: Transformers are Better than State Space Models at Copying. In *The Forty-first International Conference on Machine Learning*, 2024.

Kitaev, N., Kaiser, L., and Levskaya, A. Reformer: The efficient transformer. In *The Eighth International Conference on Learning Representations*, 2020.

Lenz, B., Lieber, O., Arazi, A., Bergman, A., Manevich, A., Peleg, B., Aviram, B., Almagor, C., Fridman, C., Padnos, D., Gissin, D., Jannai, D., Muhlgay, D., Zimberg, D., Gerber, E. M., Dolev, E., Krakovsky, E., Safahi,

E., Schwartz, E., Cohen, G., Shachaf, G., Rozenblum, H., Bata, H., Blass, I., Magar, I., Dalmedigos, I., Osin, J., Fadlon, J., Rozman, M., Danos, M., Gokhman, M., Zusman, M., Gidron, N., Ratner, N., Gat, N., Rozen, N., et al. Jamba: Hybrid transformer-mamba language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

Liu, J., Zhu, D., Bai, Z., He, Y., Liao, H., Que, H., Wang, Z., Zhang, C., Zhang, G., Zhang, J., Zhang, Y., Chen, Z., Guo, H., Li, S., Liu, Z., Shan, Y., Song, Y., Tian, J., Wu, W., Zhou, Z., Zhu, R., Feng, J., Gao, Y., He, S., Li, Z., Liu, T., Meng, F., Su, W., Tan, Y., Wang, Z., Yang, J., Ye, W., Zheng, B., Zhou, W., Huang, W., Li, S., and Zhang, Z. A comprehensive survey on long context language modeling, 2025. URL https://arxiv.org/abs/2503.17407.

Liu, N. F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., and Liang, P. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 2024.

Mattern, J. and Hohr, K. Mamba-chat. GitHub, 2023. URL https://github.com/havenhq/mamba-chat.

Merrill, W., Petty, J., and Sabharwal, A. The illusion of state in state-space models. In *The Forty-first International Conference on Machine Learning*, 2024.

Nunez, E., Zancato, L., Bowman, B., Golatkar, A., Xia, W., and Soatto, S. Expansion span: Combining fading memory and retrieval in hybrid state space models, 2024. URL https://arxiv.org/abs/2412.13328.

Olsson, C., Elhage, N., Nanda, N., Joseph, N., DasSarma, N., Henighan, T., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., Drain, D., Ganguli, D., Hatfield-Dodds, Z., Hernandez, D., Johnston, S., Jones, A., Kernion, J., Lovitt, L., Ndousse, K., Amodei, D., Brown, T., Clark, J., Kaplan, J., McCandlish, S., and Olah, C. In-context learning and induction heads, 2022. URL https://arxiv.org/abs/2209.11895.

Patro, B. N. and Agneeswaran, V. S. Mamba-360: Survey of state space models as transformer alternative for long sequence modelling: Methods, applications, and challenges, 2024. URL https://arxiv.org/abs/2404.16112.

Pawar, S., Tonmoy, S. M. T. I., Zaman, S. M. M., Jain, V., Chadha, A., and Das, A. The what, why, and how of context length extension techniques in large language models – a detailed survey, 2024. URL https://arxiv.org/abs/2401.07872.

Poli, M., Massaroli, S., Nguyen, E., Fu, D. Y., Dao, T., Baccus, S., Bengio, Y., Ermon, S., and Ré, C. Hyena hierarchy: Towards larger convolutional language models. In *The Fortieth International Conference on Machine Learning*, 2023.

Ren, L., Liu, Y., Lu, Y., Shen, Y., Liang, C., and Chen, W. Samba: Simple hybrid state space models for efficient unlimited context language modeling. In *The Thirteenth International Conference on Learning Representations*, 2025.

Sarrof, Y., Veitsman, Y., and Hahn, M. The expressive capacity of state space models: A formal language perspective. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Tay, Y., Dehghani, M., Bahri, D., and Metzler, D. Efficient transformers: A survey, 2022. URL https://arxiv.org/abs/2009.06732.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. In *The Thirtieth Annual Conference on Neural Information Processing Systems*, 2017.

Vig, J. and Belinkov, Y. Analyzing the structure of attention in a transformer language model. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 2019.

Waleffe, R., Byeon, W., Riach, D., Norick, B., Korthikanti, V., Dao, T., Gu, A., Hatamizadeh, A., Singh, S., Narayanan, D., Kulshreshtha, G., Singh, V., Casper, J., Kautz, J., Shoeybi, M., and Catanzaro, B. An empirical study of mamba-based language models, 2024. URL https://arxiv.org/abs/2406.07887.

Wikipedia. Harry Potter. https://en.wikipedia.org/wiki/Harry_Potter, 2025.

Yang, S. and Zhang, Y. Fla: A triton-based library for hardware-efficient implementations of linear attention mechanism, 2024. URL https://github.com/fla-org/flash-linear-attention.

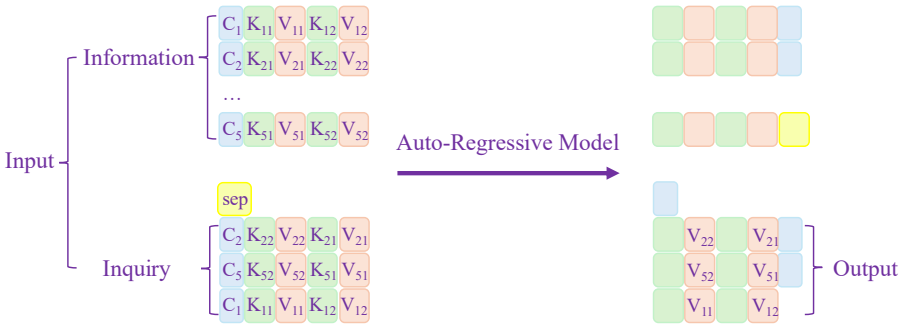


Figure 5: Input and output components of the auto-regressive multi-query joint recall task. The input sequence is further divided into an information component and an inquiry component.

A. Relationship Between the Argmax and Sign-Bit LSH Binning Rules

In this section, we show how the sign-bit LSH binning rule (Eq. 8) can be interpreted as an argmax LSH binning rule (Eq. 7) applied to an expanded projection matrix with 2^h columns. We first construct the expanded matrix, and then prove the equivalence.

Expanding the projection matrix. Let the original random projection be $\mathbf{H} = [\mathbf{H}_1, ..., \mathbf{H}_h] \in \mathbb{R}^{d\times h}, \mathbf{H}_j \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1)$. Define a codebook of 2^h signed prototypes

$$\mathcal{B} = \left\{ \mathbf{B_b} = \sum_{j=1}^{h} b_j \mathbf{H}_j \;\middle|\; \mathbf{b} = (b_1, \ldots, b_h) \in \{-1, +1\}^h \right\} \subset \mathbb{R}^d. \tag{23}$$

Stacking all $\mathbf{B_b}$ as columns yields the implicit matrix $\tilde{\mathbf{H}} \in \mathbb{R}^{d\times 2^h}$.

Equivalence of the two binning rules. For a normalized query vector $\tilde{\mathbf{Q}}_i$, we define its sign projection as $\mathbf{s} = \text{sign}(\tilde{\mathbf{Q}}_i^\top \mathbf{H}) \in \{-1, +1\}^h$. The inner product of $\tilde{\mathbf{Q}}_i$ and a prototype $\mathbf{B_b} \in \mathcal{B}$ is

$$\langle \tilde{\mathbf{Q}}_i, \mathbf{B_b} \rangle = \sum_{j=1}^{h} b_j \, \langle \tilde{\mathbf{Q}}_i, \mathbf{H}_j \rangle. \tag{24}$$

Because every term with $b_j \neq s_j$ flips the sign of the positive quantity $|\langle \tilde{\mathbf{Q}}_i, \mathbf{H}_j \rangle|$, we have the strict inequality $\langle \tilde{\mathbf{Q}}_i, \mathbf{B_s} \rangle > \langle \tilde{\mathbf{Q}}_i, \mathbf{B_b} \rangle$ for all $\mathbf{b} \neq \mathbf{s}$. Hence

$$\text{argmax}_{\mathbf{b}\in\{-1,+1\}^h} \langle \tilde{\mathbf{Q}}_i, \mathbf{B_b} \rangle = \mathbf{s} = \text{bin}_{Q_i}^{(\text{sign})}, \tag{25}$$

The sign-bit assignment is exactly the argmax rule applied to $\tilde{\mathbf{H}}$. An identical argument holds for keys $\tilde{\mathbf{K}}_j$. Thus, the sign-bit method equals the argmax method with 2^h (expanded) columns.

B. Theoretical Proof

Multi-query joint recall requires models to recall an $n_c \times n_k$ table of context-specific key-value associations, in which n_k keys are associated with different values in each of the n_c contexts, with $n = n_c \times n_k$ being the total number of entries. For clarity, we introduce some additional notations for multi-query joint recall in the auto-regressive setting, as illustrated in Appendix Figure 5. The input sequence is divided into an information component and an inquiry component. The information component provides the context-specific key-value associations. The inquiry component permutes the order of context and keys in the information component, and the model is required to predict the corresponding values given each key under every specified context.

B.1. Proof of Corollary 3.1

Corollary 3.1 (Limited expressiveness of SSMs)**.** *Under the uniform multi-query joint recall distribution, for any n, a generalized state-space model defined in Sec. 2.3 incurs an error rate of at least $1 - \frac{|\mathcal{U}|}{|\mathcal{V}|^n}$.*

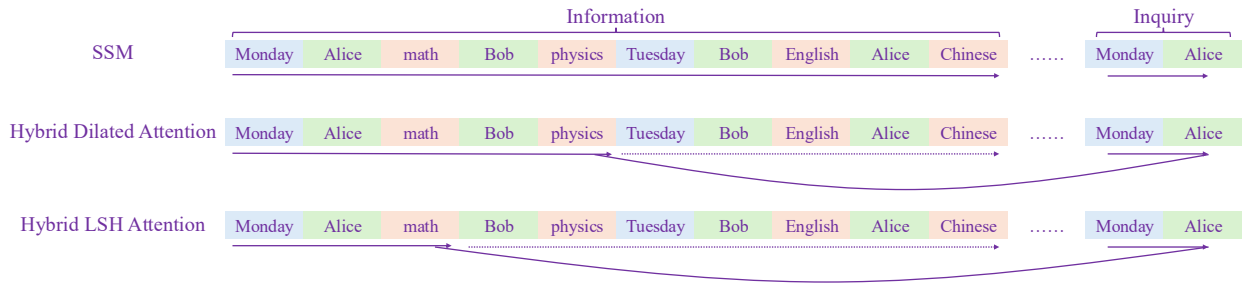


Figure 6: Comparison between SSM, hybrid dilated attention model and hybrid locality-sensitive hashing (LSH) attention model on joint recall. By selectively bypassing irrelevant context, sparse attention alleviates memory overload in SSM layers and enhances the hybrid model's capability to retrieve relevant information.

The intuition behind the proof of Corollary 3.1 is straightforward: the number of possible joint recall data instances that a model can accurately represent is fundamentally limited by the capacity of its recurrent state. Since all information from the input sequence must be encoded into a fixed recurrent state following the context during processing, the total number of distinguishable output is bounded by the representation capacity of the state space $|\mathcal{U}|$. Consequently, if the size of the output space $|\mathcal{V}|^n$ exceeds $|\mathcal{U}|$, the model inevitably incurs non-negligible error.

As a direct consequence of Theorem 2.7 in Jelassi et al. (2024), we adopt their proof strategy. We reformulate Lemma D.1 in Jelassi et al. (2024) as the following Lemma B.1. Let m denote the index of the last token in the information component. Then, for any fixed permutation $\mathcal{P}$ of the context and keys in the inquiry component, the following Lemma B.1 holds:

Lemma B.1. *Let $\mathcal{M}$ be a fixed-state generalized SSM that maps the joint recall input space $\mathcal{X}$ to the output space $\mathcal{V}^n$ under any fixed permutation $\mathcal{P}$ of the context and keys in the inquiry component. Then there exists a function $G : \mathcal{U} \to \mathcal{V}^n$ such that for all inputs $\mathbf{x} \in \mathcal{X}$, the model output satisfies $\mathcal{M}(\mathbf{x}) = G(U_m(\mathbf{x}))$, U_m is defined in Eq.12.*

Following Jelassi et al. (2024), we bound the error of the model by comparing the number of possible model states to the number of distinct input instances.

Proof.

$$1 - \Pr[err] = \Pr[\mathcal{M}(\mathbf{x}) = \mathbf{y} | \mathbf{y} \in \mathcal{V}^n] \tag{26}$$

$$= \frac{1}{|\mathcal{V}|^n} \sum_{\mathbf{y} \in \mathcal{V}^n} \mathbf{1}[\mathcal{M}(\mathbf{x}) = \mathbf{y}] \tag{27}$$

$$= \frac{1}{|\mathcal{V}|^n} \sum_{\mathbf{y} \in \mathcal{V}^n} \sum_{\mathbf{u} \in \mathcal{U}} \mathbf{1}[G(\mathbf{u}) = \mathbf{y}] \cdot \mathbf{1}[U_m(\mathbf{x}) = \mathbf{u}] \tag{28}$$

$$\leq \frac{1}{|\mathcal{V}|^n} \sum_{\mathbf{u} \in \mathcal{U}} \mathbf{1}[U_m(\mathbf{x}) = \mathbf{u}] \tag{29}$$

$$\leq \frac{|\mathcal{U}|}{|\mathcal{V}|^n} \tag{30}$$

□

To guarantee $\Pr[err] = 0$, it is necessary that the number of representable states satisfies $|\mathcal{U}| \geq |\mathcal{V}|^n$. Taking the logarithm of both sides yields the condition $b \geq n \log |\mathcal{V}|$. This implies that the state-space dimension of the model must grow linearly with the number of entries n in the joint recall table, highlighting a fundamental limitation of the representation capacity of SSMs.

In contrast, as illustrated in Fig. 6, hybrid sparse attention models mitigate this limitation by enabling information to propagate through multiple parallel paths, thereby alleviating the bottleneck imposed by sequential state updates.

B.2. Proof of Proposition 3.2

Proposition 3.2 (Improved expressiveness of SSMs integrated with CDSA)**.** *There exists a* 2*-layer auto-regressive hybrid model consisting of an SSM layer followed by an LSH attention layer, which can solve multi-query joint recall in* $O(n \log^2 n)$ *time complexity with* $O(\log n)$ *SSM state dimensions.*

Proof. We prove by construction. In the first layer, we expect the SSM state concatenates each value token representation with its associated key token representation and context token representation. To be specific, we expect the SSM state representation at each value token to be

$$[\mathbf{c}, \mathbf{k}, \mathbf{v}, is_v] \in \mathcal{U}$$

where $\mathbf{c}$ is the representation of the current associated context token, $\mathbf{k}$ is the representation of the current associated key token, and $\mathbf{v}$ is the representation of the nearest value token. is_v is a sign indicator (-1 or 1) that specifies whether the current token is a value token.

To achieve this, we first construct each vector $\mathbf{c}$, $\mathbf{k}$ and $\mathbf{v}$ be a distinct b-dimensional vector with unit norm without zero entries, i.e., $\|\mathbf{c}\|_2 = 1$, $\|\mathbf{k}\|_2 = 1$, $\|\mathbf{v}\|_2 = 1$, $\forall j, \mathbf{c}_j \neq 0, \mathbf{k}_j \neq 0, \mathbf{v}_j \neq 0$. Since the number of distinct vectors that can be drawn from the unit sphere grows exponentially with dimensionality, $O(\log n)$ embedding dimensions are sufficient to ensure that all representations are distinguishable. Then we define an embedding space in which context and value tokens are mapped to structured representations. Specifically, a context token is embedded as

$$\mathbf{e} = [\mathbf{c}, \mathbf{0}, \mathbf{0}, -1]$$

where $\mathbf{c}$ is the constructed representation of this context token on the unit sphere, and the final coordinate is set to -1 to indicate that the current token is not a value. Similarly, a key token is embedded as

$$\mathbf{e} = [\mathbf{0}, \mathbf{k}, \mathbf{0}, -1]$$

and a value token is embedded as

$$\mathbf{e} = [\mathbf{0}, \mathbf{0}, \mathbf{v}, 1]$$

where $\mathbf{k}$ and $\mathbf{v}$ are the key and value representations from the unit sphere, respectively, and the final coordinate is set to 1 only when the current token is a value token. Following Eq. 12, we define the update rule of the generalized SSM as follows:

$$U_i = u(U_{i-1}, \mathbf{e}) = U_{i-1} \odot \mathbf{1}[\mathbf{e}_j = 0] + \mathbf{e} \odot \mathbf{1}[\mathbf{e}_j \neq 0] \tag{31}$$

$$R_i = r(U_i) = U_i \tag{32}$$

where $\mathbf{e}$ is the current input embedding and $\mathbf{e}_j$ refers to its j-th dimension. The update rule operates as a conditional overwrite: if a position does not carry information (i.e., the corresponding dimension in $\mathbf{e}$ is $\mathbf{0}$), the previous state is preserved; otherwise, it is updated with the current embedding. Following this update rule, the SSM state at each value token in the information component takes the form

$$[\mathbf{c}, \mathbf{k}, \mathbf{v}, 1]$$

while the SSM state at each key token takes the form

$$[\mathbf{c}, \mathbf{k}, ?, -1]$$

In the second layer, LSH attention operates on the SSM state $[\mathbf{c}, \mathbf{k}, \mathbf{v}, is_v] \in \mathcal{U}$, using $[\mathbf{c}, \mathbf{k}, \mathbf{0}, is_v]$ as the LSH attention key representation, $[\mathbf{c}, \mathbf{k}, \mathbf{0}, 1]$ as the LSH attention query representation, and $[\mathbf{0}, \mathbf{0}, \mathbf{v}, 1]$ as the LSH attention value representation. This design ensures that value tokens in the information component and key tokens in the inquiry component that share the same context and key (i.e., identical $\mathbf{c}$ and $\mathbf{k}$ representations in the SSM state) will always be assigned to the same hash bin. With a sufficient number of, e.g. $O(n)$ hash bins, which can be efficiently constructed using sign-bit binning rule with $O(\log n)$ random projection vectors, values associated with each key in every specified context are reliably retrievable by LSH attention. This step is the bottleneck of computation with $O(n \log^2 n)$ time complexity. □

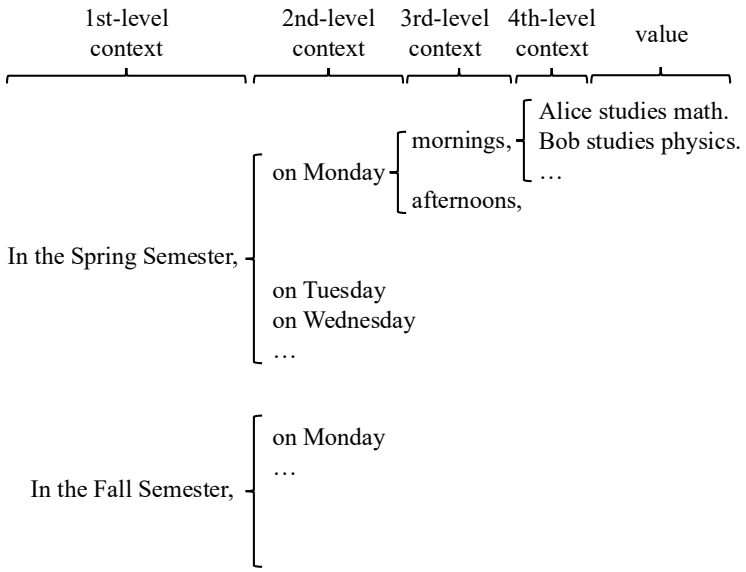


Figure 7: An example of multi-level context in natural language.

B.3. Proof of Proposition 3.3

Proposition 3.3 (Limited expressiveness of SSMs integrated with CISA)**.** *There does not exist a 2-layer auto-regressive hybrid model consisting of an SSM layer followed by a CISA layer, which can solve multi-query joint recall with $o(n^2)$ time complexity, since it requires at least $O(\frac{n}{k})$ SSM state dimensions, k is the maximum number of keys that each query allowed to attend to in the sparse attention module, as defined in Eq. 4.*

Proof. Consider a key given in the inquiry component of the auto-regressive joint recall task. The model is required to output the associated value token when this key token is provided as input. Taking the query representation from this key token, the sparse attention can attend to at most k key representations from previous tokens, where the key representations are calculated based on the SSM state representations of the first layer. To solve the joint recall task, these k key representations being attended must collectively encode the full information component. Since the full information component length is $O(n)$, by Corollary 3.1, k state representations of the generalized SSM in the first layer must use at least $O(\frac{n}{k})$ dimensions to collectively store the information component. Thus, the per-key computational cost required by the second-layer sparse attention is $O(k \cdot \frac{n}{k}) = O(n)$, and therefore the total time complexity is $O(n^2)$. □

Comparing **Proposition** 3.2 with **Proposition** 3.3, we see a clear representation capacity gap between the SSMs integrated with CDSA and the SSMs integrated with CISA. In practice, however, with an appropriate constant k, integrating CISA with SSMs still provides an advantage: unlike SSMs, which only have access to the last state representation, CISA layers can attend to k different state representations simultaneously, at a cost of k times of computation budget.

C. Extending Joint Recall to Multi-level Context

As illustrated in Figure 7, in many cases, natural language contexts exhibit hierarchical dependencies. This motivates us to extend joint recall to the multi-level context setting, in which we regard the keys as the last level of context.

C.1. Formulation

Given w different levels of context vocabulary $\mathcal{C}_1, \mathcal{C}_2, ..., \mathcal{C}_w$ and token vocabulary $\mathcal{V}$, multi-level context joint recall requires a model to recover the mapping $\mathcal{C}_1 \times \mathcal{C}_2 \times ... \times \mathcal{C}_w \to \mathcal{V}$. The context of multi-level context joint recall is hierarchically structured analogously to natural languages. It is divided into $|\mathcal{C}_1|$ first-level blocks, where each first-level block begins with a token from the first-level context vocabulary $\mathcal{C}_1$. Each first-level block is further divided into sub-blocks beginning with a second-level context token from $\mathcal{C}_2$, and this recursive sub-division continues up to the w-th level. The last-level block consists of a w-th level context token followed by a value token from $\mathcal{V}$. Note that associative recall is a special case of

multi-level joint recall with $w = 1$, and joint recall is a special case of multi-level joint recall with $w = 2$. We similarly define multi-query multi-level context joint recall, where the model is required to recall all $n = |\mathcal{C}_1| \times |\mathcal{C}_2| \times ... \times |\mathcal{C}_w|$ entries in the full context-value supertable.

C.2. Expressiveness of SSMs Integrated with CDSA on Multi-Level Context Joint Recall

On multi-query multi-level context joint recall, both Corollary 3.1 and Proposition 3.3 continue to hold under the same assumptions. We now extend Proposition 3.2 to the following Proposition C.1, which demonstrates that SSMs integrated with CDSA remain expressive even in the presence of w levels of hierarchical contexts.

Proposition C.1 (Expressiveness of SSMs integrated with CDSA on multi-level context joint recall)**.** *There exists a 2-layer auto-regressive hybrid model consisting of an SSM layer followed by an LSH attention layer, which can solve multi-query multi-level context joint recall in $O(wn \log^2 n)$ time complexity with $O(w \log n)$ SSM state dimensions.*

Proof. Similar to Proposition 3.2, we prove by construction. In the first layer, we hope the SSM state to consist of the context and value representations

$$[\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_w, \mathbf{v}, is_v] \in \mathcal{U}$$

where $\mathbf{z}_i$ denotes the representation of the nearest i-th level context token, $\mathbf{v}$ represents the nearest value token, and is_v is a sign indicator (-1 or 1) that specifies if the current token is a value token.

To achieve this, we similarly construct each vector $\mathbf{z}_i$ and $\mathbf{v}$ to be a distinct b-dimensional vector with unit norm without zero entries, i.e., $\|\mathbf{z}_i\|_2 = 1$, $\|\mathbf{v}\|_2 = 1$, $\forall j, \mathbf{z}_{ij} \neq 0, \mathbf{v}_j \neq 0$. Since the number of distinct vectors that can be drawn from the unit sphere grows exponentially with dimensionality, $O(\log n)$ embedding dimensions are sufficient to ensure that all representations are distinguishable. Consequently, the total size of the SSM state is $O(w \log n)$.

Then we define an embedding space in which context and value tokens are mapped to structured representations. Specifically, a i-th level context token is embedded as

$$\mathbf{e} = [\mathbf{0}, ..., \mathbf{0}, \mathbf{z}_i, \mathbf{0}, ..., -1]$$

where $\mathbf{z}_i$ is the context token representation, and the final coordinate is set to -1 to indicate that the token is not a value. Similarly, a value token is embedded as

$$\mathbf{e} = [\mathbf{0}, \mathbf{0}, ..., \mathbf{0}, \mathbf{v}, 1]$$

where $\mathbf{v}$ is the value token representation and the final coordinate is set to 1 to mark it as a value token. We keep the update rule of the generalized SSM as in Eq. 31:

$$U_i = u(U_{i-1}, \mathbf{e}) = U_{i-1} \odot \mathbf{1}[\mathbf{e}_j = 0] + \mathbf{e} \odot \mathbf{1}[\mathbf{e}_j \neq 0] \tag{33}$$

$$R_i = r(U_i) = U_i \tag{34}$$

which operates as a conditional overwrite. Following this update rule, the SSM state at each value token takes the form

$$[\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_w, \mathbf{v}, 1]$$

where $\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_w$ are the w levels of context representations of the current token, $\mathbf{v}$ is the value representation, and the final dimension is set to 1 to indicate that the current token is a value token.

In the second layer, LSH attention operates on the SSM state $[\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_w, \mathbf{v}, is_v]$, using $[\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_w, \mathbf{0}, is_v]$ as the LSH attention key representation, $[\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_w, \mathbf{0}, 1]$ as the LSH attention query representation, and $[\mathbf{0}, \mathbf{0}, ..., \mathbf{v}, 1]$ as the LSH attention value representation. This design ensures that tokens that share the same context (i.e., identical $[\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_w]$ in the SSM state) will always be assigned to the same hash bin. With a sufficient number of, e.g. $O(n)$ hash bins, which can be efficiently constructed using sign-bit binning rule with $O(\log n)$ random projection vectors, value representations associated with identical key combinations in the context are reliably retrievable by LSH attention. This step is the bottleneck of computation with $O(wn \log^2 n)$ time complexity. □

This construction establishes that a 2-layer hybrid model consisting of a generalized SSM followed by LSH attention can solve multi-query multi-level joint recall efficiently, with sub-quadratic time complexity and sub-linear state complexity with respect to the input sequence length.

D. Experiment Details

D.1. Empirical Verification on Joint Recall

For all models, we fix the number of layers to 2, set the hidden size to 64, and use $k = 64$. For variants that integrate multiple sparse attention components, namely SW+D, A (consists of a SW component and a sink attention component, a sink attention always attends to the first k tokens in the sequence only), and HAX (LSH+KS), we allocate $k = 32$ to each component, in order to maintain a global $k = 64$. For both LSH and HAX (LSH+KS), we adopt the sign-bit binning strategy (Eq. 8) with $h = 8$, and refresh the random hashing matrix at each training step. To ensure a fair comparison, we double the hidden size of the Mamba and Mamba2 baselines, which do not include any sparse attention mechanism, to 128. Additionally, we include a Samba baseline, consisting of 2 Mamba layers and 2 sliding window attention layers. For Samba, the hidden size and sliding window width (k) are both set to 64. We use AdamW optimizer. All models are trained for $400,000$ steps with a batch size of 64. Our implementation is based on Flash-Linear-Attention (Yang & Zhang, 2024).

D.2. Continual Pre-training on Natural Language

Following Mamba (Gu & Dao, 2024), we adopt the uncopyrighted the Pile dataset (Gao et al., 2021) as our pre-training corpus. To enhance the long-context modeling capability, we filter samples to retain only those with tokenized lengths of at least $4,096$ tokens. After continual pre-training based on the publicly released Mamba 130M checkpoint, we select the checkpoint with the lowest validation loss and perform instruction tuning on the UltraChat dataset (Ding et al., 2023b), following Mamba-Chat pipeline (Mattern & Hohr, 2023). Finally, we evaluate the instruction-tuned model on the Ruler (Hsieh et al., 2024) and LongBench (Bai et al., 2024) benchmarks.

For all experiments, we fix the sparsity parameter at $k = 128$. For variants that integrate multiple sparse attention components, namely SW+D, A, and HAX (LSH+KS), we set $k = 64$ to each component, in order to maintain a global $k = 128$. For both LSH and HAX (LSH+KS), we adopt the argmax binning strategy (Eq. 7) with $h = k$. We resample the random hashing matrix at each training step and fix a random hashing matrix during evaluation. To visualize the validation loss, we perform continual pre-training for $50,000$ steps with a context length of $2,048$ and a global batch size of 64. For downstream evaluation, we conduct continual pre-training for $10,000$ steps with a context length of $3,072$ and the same global batch size. This is followed by instruction tuning for $3,000$ steps, also with a global batch size of 64. We use AdamW optimizer.

At the beginning of continual pre-training, the $\mathbf{K}$ and $\mathbf{Q}$ projection weights are initialized using the parameters of the $\mathbf{B}$ and $\mathbf{C}$ projections from Mamba, respectively, based on state-space duality(Dao & Gu, 2024). The gating parameters $\mathbf{E}$ are initialized to zero. During continual pretraining, all models follow the cosine learning rate schedule used in Mamba, with a maximum learning rate of 3e-4 and a minimum of 1e-5. A warm-up phase of 200 steps with a learning rate of 0 precedes the cosine schedule. For instruction tuning, we apply a 200-step zero learning rate phase followed by 800 steps of linear warm-up, after which the learning rate remains constant. The peak learning rate during instruction tuning is set to 3e-6 for Mamba+D, Mamba+LSH, and Mamba+HAX, and 1e-5 for all other architectures.

E. Additional Experiments

E.1. Short-context Modeling Benchmark

We follow Mamba (Gu & Dao, 2024) to evaluate the zero-shot short context modeling performance of the continually pre-trained models on the LM evaluation harness benchmark from EleutherAI (Gao et al., 2024). Our results in Table4 show that continual pre-training on long sequences will not lead to a significant performance drop on short context benchmarks, where the Mamba w/o continual pre-training results are copied from the Mamba paper (Gu & Dao, 2024).

Table 4: LM evaluation harness benchmark for continually pre-trained Mamba models. We compare different sparse attention integrated with Mamba, including CISA and CDSA methods as in Tab. 1.

	LambdaPPL	WinoGrande	PIQA	LambdaAcc	HellaSwag	ARC-E	ARC-C	AverageAcc
Mamba	15.58	51.8	63.8	44.3	35.3	47.8	24.4	44.6
+D	15.75	52.8	64.4	43.9	35.3	47.7	24.1	44.7
+SW	15.47	52.8	63.8	44.3	35.2	47.9	24.7	44.8
+SW+D	15.35	53.0	64.0	44.7	35.2	47.6	24.4	44.8
+A	15.65	53.0	64.0	44.2	35.3	47.7	24.7	44.8
+LSH	15.73	52.6	64.3	43.8	35.3	47.7	24.0	44.6
+KS *(ours)*	15.86	52.4	63.7	44.1	35.2	47.9	24.5	44.6
+HAX *(ours)*	15.62	52.5	63.9	44.3	35.3	47.9	24.5	44.7
w/o training	16.07	51.9	64.5	44.3	35.3	48.0	24.3	44.7

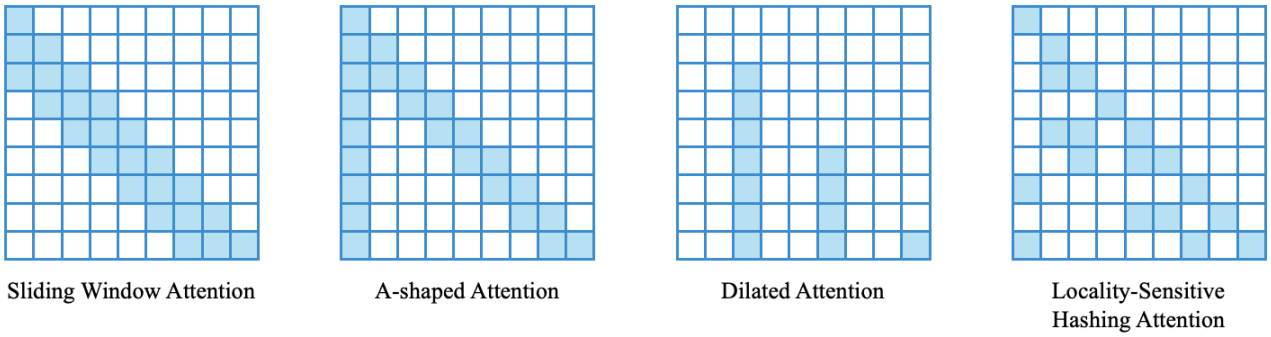


Figure 8: Examples for input-dependent and input-independent sparse attention patterns.

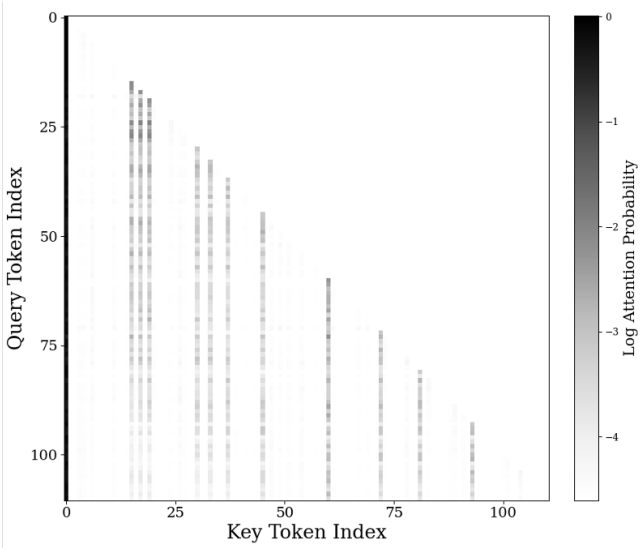


Figure 9: An example of the "vertical-stripe" attention pattern in LLM. We input the first paragraph of the Wikipedia term "Harry Potter" (Wikipedia, 2025) into the Llama 3.2 1B model (Grattafiori et al., 2024) and visualize the log attention probabilities of the last head in the first layer. The input text is: “Harry Potter is a series of seven fantasy novels written by British author J. K. Rowling. The novels chronicle the lives of a young wizard, Harry Potter, and his friends, Ron Weasley and Hermione Granger, all of whom are students at Hogwarts School of Witchcraft and Wizardry. The main story arc concerns Harry’s conflict with Lord Voldemort, a dark wizard who intends to become immortal, overthrow the wizard governing body known as the Ministry of Magic, and subjugate all wizards and Muggles (non-magical people).”