TransXSSM: A Hybrid Transformer–State Space Model with Unified Rotary Position Embedding

Bingheng Wu[1]   **Jingze Shi**[1]   **Yifan Wu**[1]
Nan Tang[1]   **Yuyu Luo**[1]
[1]HKUST (Guangzhou)

Abstract

Transformers exhibit proficiency in capturing long-range dependencies, whereas State Space Models (SSMs) facilitate linear-time sequence modeling. Notwithstanding their synergistic potential, the integration of these architectures presents a significant challenge, primarily attributable to a fundamental incongruity in their respective positional encoding mechanisms: Transformers rely on explicit Rotary Position Embeddings (RoPE), while SSMs leverage implicit positional representations via convolutions. This divergence often precipitates discontinuities and suboptimal performance. To address this impediment, we propose a unified rotary position embedding (**Unified RoPE**) methodology, thereby establishing a consistent positional encoding framework for both self-attention and state-space components. Using this Unified RoPE, we introduce **TransXSSM**, a hybrid architecture that coherently integrates the Transformer and SSM layers under this unified positional encoding scheme. At a 4K sequence length, TransXSSM exhibits training and inference speeds that are **42.3% and 29.5% faster**, respectively, relative to standard Transformer models. It also delivers higher accuracy: under comparable settings, it surpasses a Transformer baseline by over 4% on language modeling benchmarks. TransXSSM furthermore scales more effectively: TransXSSM-1.3B gains **7.22%** in average accuracy over its 320M version (versus about 6% gains for equivalent Transformers or SSMs). Our results show that unified positional encoding resolves positional incompatibility in hybrid models, enabling efficient, high-performance long-context modeling.

1 Introduction

Effective positional encoding is crucial for sequence modeling in language tasks, as it underpins a model's ability to understand order, perform reasoning, and handle long contexts. Currently, three representative paradigms dominate the landscape of sequence modeling: Transformer-based models (*e.g.,* Llama3 [1], state-space models (SSMs, *e.g.,* Mamba2 [2]), and hybrid architectures combining both approaches (*e.g.,* Jamba [3]). However, as summarized in Table 1, each approach exhibits critical limitations. Transformers utilize explicit positional encodings such as Rotary Position Embedding (RoPE) to effectively handle positional information but suffer from severe computational inefficiencies, especially on long sequences, due to quadratic self-attention complexity [4]. By contrast, SSM-based models achieve linear computational complexity and high throughput for extremely long sequences but implicitly encode positions, thus significantly limiting their reasoning and few-shot learning abilities. Hybrid models (*e.g.,* Jamba [3], which combines a Transformer and an Mamba SSM) aim to get the best of both worlds, *i.e.,* the reasoning capability of Transformers with the efficiency of SSMs, and have shown promise on extended-context tasks [5]. However, integrating Transformers and SSMs presents non-trivial challenges, particularly due to fundamental differences in how they encode positional information. Given these limitations, **our goal** is to design a hybrid

Table 1: Comparison of TransXSSM and related model paradigms across key features.

Models	Mechanism	Position Encoding	Unified Encoding	Spectrum Continuity	Long-Seq Efficiency	Training Throughput	Inference Speed
Transformer (e.g., Llama3)	Self-Attention	Rotary Position Embedding	✔	High	Low	Low	Low
State-Space Models (e.g., Mamba2)	State-Space	Implicit (Conv+Recursion)	✔	Low	High	**Highest**	High
Hybrid (e.g., Jamba)	Transformer + Mamba	Separate	✘	Moderate	Moderate	Moderate	Moderate
TransXSSM (ours)	Self-Attention + State-Space	**Unified** Rotary Position Embedding	✔	High	High (near-linear)	High (close to SSM)	High

model that effectively combines the powerful positional reasoning capabilities of Transformers with the computational efficiency and scalability of SSMs.

Challenges: Positional Encoding Incompatibility and Information Discontinuity. Transformer architectures require explicit positional embeddings, such as Rotary Position Embedding (RoPE) [6], due to their permutation-invariant self-attention mechanism. Conversely, SSMs encode positional information implicitly through convolutional recurrence and internal state dynamics, without explicit positional embeddings. When stacking Transformer and SSM layers naively, this fundamental mismatch in positional encoding methods creates a *positional encoding inconsistency*, causing an *information gap* between module interfaces. Recent work suggests hybrid models can operate without explicit positional encodings by relying solely on SSMs' implicit ordering [2]. However, this approach can deprive Transformer layers of crucial positional cues, severely impairing performance on tasks that demand precise positional reasoning. Fundamentally, the absence of a unified positional encoding scheme hinders coherent propagation of positional information throughout hybrid models, representing a critical barrier to effectively integrating Transformer and SSM architectures [7].

Proposed Approach: Unified RoPE and the TransXSSM Architecture. To address the above challenges, we propose a *unified rotational position embedding* (Unified RoPE) and building a new hybrid architecture around it. Our method extends RoPE, originally created for Transformers to encode absolute positions with complex rotations [6], to state-space models. By applying the same rotary positional transformations to the state update signals of an SSM as we do to the query/key vectors of self-attention, we establish a single consistent positional encoding that is shared across both module types. Equipping SSM layers with RoPE-based position embeddings (or adjusting their parameterization to incorporate RoPE) allows positional phase information to flow continuously between attention and SSM layers. The result is a spectrally continuous position representation throughout the hybrid network: no more resets or misalignments of positional phase when transitioning between layers. Leveraging Unified RoPE, we further propose the **TransXSSM** architecture, a *Hybrid State-space Attention Model* that integrates self-attention and SSM layers under this unified positional scheme. As summarized in Table 1, TransXSSM preserves the near-linear runtime, high throughput, and rapid inference typical of state-space models, and, by incorporating explicit relational position encoding together with a lightweight self-attention layer, it matches the deep contextual reasoning capabilities of Transformers. We make the following contributions:

- We propose **Unified RoPE**, a novel positional encoding mechanism that bridges the gap between self-attention and state-space models (Section 2).

- We design **TransXSSM**, a new hybrid Transformer+SSM model that leverages Unified RoPE for coherence between modules. TransXSSM is carefully structured to combine the strengths of both approaches – it achieves near-linear scaling in sequence length and high throughput (comparable to pure SSMs) while maintaining the robust contextual reasoning ability of Transformers. Key architectural choices (*e.g.,* stacking ratio of SSM to attention layers, specialized normalization) enable TransXSSM to maximize efficiency without sacrificing performance (Section 3).

- **State-of-the-Art Long-Context Performance**. We extensively evaluate TransXSSM on language modeling and long-context benchmarks, comparing it against strong baselines (Llama3, Mamba2, Jamba). TransXSSM consistently outperforms these models in both accuracy and speed; notably, our TransXSSM-1.3B model surpasses all baselines by over 2 points on seven diverse tasks. In addition, in a challenging long-context "needle-in-a-haystack" retrieval task, TransXSSM demonstrates superior accuracy, underscoring the effectiveness of our unified positional encoding for multi-hop reasoning and deep context modeling (Section 4).

2 Unified Rotary Position Embedding

2.1 Background: Self-Attention and RoPE

Self-Attention Mechanism. Self-attention computes pairwise relevance scores between all tokens in a sequence, enabling long-range dependency modeling [4]. For a sequence with query, key, and value representations Q, K, V, the basic scaled dot-product attention is: $Y = \text{softmax}(QK^\top) \cdot V$.

In causal language modeling, a binary lower-triangular mask L is applied to ensure each position only attends to previous positions (preventing information leakage). This masking restricts the $QK^\top$ matrix to zero out future positions. Many efficient attention variants re-factor or approximate the $QK^\top$ term. For example, linear attention uses kernel feature maps to rewrite $(QK^\top)V$ as $Q \cdot (K^\top V)$, enabling linear complexity. When incorporating the causal mask L, the quadratic attention can be implemented recursively. Notably, the masked attention operation can be viewed as a semiseparable matrix application, which connects self-attention to state-space models (SSMs). In fact, [2] show a state-space update can produce the same result as masked attention by an appropriate choice of matrices C, B. Formally, one can define a semiseparable matrix $M = L \circ (CB^\top)$ (where "$\circ$" denotes elementwise product) analogous to $M = L \circ (QK^\top)$. Applying this M to an input sequence X (for SSM) or values V (for attention) yields an equivalent result: $(L \circ QK^\top) \cdot V = (L \circ CB^\top) \cdot X$.

Rotary Position Embedding. Transformers require explicit positional encoding because the attention operation itself is permutation-invariant. Rotary Position Embedding (RoPE) is a technique that encodes absolute positions as complex rotations applied to query and key vectors. It introduces functions f_Q and f_K such that the inner product of a position-encoded query q at position m and key k at position n depends only on their relative position $(m - n)$. In other words, RoPE achieves an implicit relative position encoding from absolute positions. Formally, for some function g:

$$< f_Q(Q, m), f_K(K, n) > = g(Q, K, m - n) \tag{1}$$

This means the attention score between token m and token n can be made a function of $m - n$ (their distance), even though each vector is encoded with its absolute position. RoPE was originally proposed for Transformer attention. However, hybrid architectures that intermix Transformers with SSM layers lack a unified positional scheme – Transformers use explicit encodings like RoPE, whereas SSMs typically rely on implicit positional information (*e.g.,* via convolutional recurrence). This mismatch can lead to inconsistencies when passing information between layers. We seek to extend RoPE to SSMs so that both module types share the same positional encoding mechanism.

2.2 Unified Rotary Position Embedding

Rotations for Hybrid Models. We propose a unified rotary position encoding that applies the same rotational embedding to both self-attention (Transformer) and state-space (SSM) components. The key idea is to treat the state-space's recurrence weights analogously to attention's Q, K vectors. In practice, we define four position-encoding functions f_Q, f_K, f_C, f_B for queries Q, keys K, and the analogous state update vectors (which we denote C and B for the SSM's internal update).

Each function multiplies its vector by a complex phase $e^{i,\theta,m}$ or $e^{i,\theta,n}$ determined by the absolute position index. Let m be the position of a "query-like" vector (Q or C) and n for a "key-like" vector (K or B). The unified Rotary Position Embedding (Unified RoPE) is defined as:

$$f_Q(q, m) = qe^{i,m,\theta}, \quad f_K(k, n) = ke^{i,n,\theta}, \quad f_C(c, m) = ce^{i,m,\theta}, \quad f_B(b, n) = be^{i,n,\theta} \tag{2}$$

Here θ is a base angular frequency (we will define a set of multiple frequencies $\Theta = \theta_i$ shortly). Intuitively, multiplying a vector by $e^{i,m\theta}$ rotates it in the complex plane by an angle proportional to its absolute position m. Applying the same type of rotation to Q, K in the Transformer, and to C, B in the SSM, produces a consistent positional phase across both modules. This ensures that wherever a sequence goes, *i.e.,* attention or SSM, the notion of position is carried by these rotations.

Real-valued Implementation. To use RoPE in practice, the complex representation (Eq. (2)) is converted to real-valued matrix form. For a 2D vector $[x^{(1)}, x^{(2)}]^\top$, this involves multiplication by a 2D rotation matrix for $f_{\{Q,C\}}$ and $f_{\{K,B\}}$, as shown in Eq. (3):
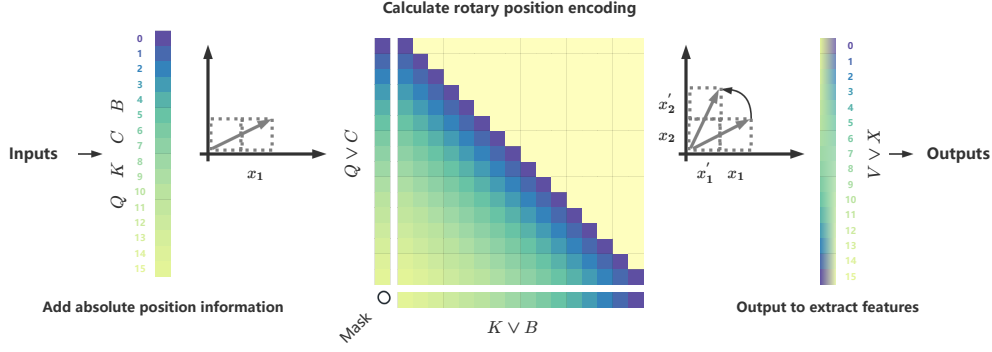
Figure 1: **Rotary Position Embedding Application**. Application of Unified RoPE. Input vectors (Q, C, K, B) are enriched with absolute positions (m, n) via rotation matrices $\mathbb{R}^d_{\Theta,m}$ or $\mathbb{R}^d_{\Theta,n}$ (Eq. (3) and Eq.(5)), yielding position-encoded vectors. Their inner product captures relative position $m - n$ (Eq. (4)). An optional masking step can follow.

$$f_{\{Q,C\}}(x_m, m) = \begin{pmatrix} \cos(m\theta) & -\sin(m\theta) \\ \sin(m\theta) & \cos(m\theta) \end{pmatrix} \begin{pmatrix} x_m^{(1)} \\ x_m^{(2)} \end{pmatrix}, \; f_{\{K,B\}}(x_n, n) = \begin{pmatrix} \cos(n\theta) & -\sin(n\theta) \\ \sin(n\theta) & \cos(n\theta) \end{pmatrix} \begin{pmatrix} x_n^{(1)} \\ x_n^{(2)} \end{pmatrix}$$
(3)

The inner product of these RoPE-modified vectors $f_Q(q_m, m), f_K(k_n, n)$ (or $f_C(c_m, m), f_B(b_n, n)$) yields a score dependent on the relative position $m - n$ via rotation matrix $R^d_{\Theta,m-n}$, as shown in Eq. (4). This reflects relative positions for both self-attention and SSMs. Appendix A justifies the application of RoPE for the CB term in State-Space Duality.

$$attn_{score} = < f_Q(q_m, m), f_K(k_n, n) > = q_m R^d_{\Theta,m-n} k_n^\top$$
$$ssd_{score} = < f_C(c_m, m), f_B(b_n, n) > = c_m R^d_{\Theta,m-n} b_n^\top$$
(4)

In other words, the positional difference $m - n$ is now reflected in the phase of the dot-product. This unifies how positional information is encoded: both the self-attention scores and the SSM update scores can be interpreted through the same relative-position rotation $R^d_{\Theta,m-n}$. As a result, a Transformer layer and an SSM layer will "see" positional shifts in the same way, facilitating seamless information flow between the two.

Rotation Matrices and Frequencies. For dimensions $d > 2$, the vector is partitioned into $d/2$ 2D blocks. Each block is rotated independently (similar to Eq. (3)) using distinct angles θ_i from Θ, forming a block-diagonal rotation matrix $R^d_{\Theta,m}$ (Eq. (5)):

$$R^d_{\Theta,m} = \bigoplus_{i=0}^{\frac{d}{2}-1} \begin{pmatrix} \cos(m\theta_i) & -\sin(m\theta_i) \\ \sin(m\theta_i) & \cos(m\theta_i) \end{pmatrix}, \quad \theta_i = 10000^{-2i/d}.$$
(5)

The set of distinct rotation angles $\Theta = \{\theta_i\}$ for $R^d_{\Theta,m}$ (Eq. (5)) is defined for $i \in [0, \ldots, \frac{d}{2} - 1]$ as follows: $\Theta = \{\theta_i = 10000^{-2i/d}, i \in [0, 1, \ldots, \frac{d}{2} - 1]\}$.

Compatibility with Linear-Time Inference. Finally, a causal mask L is applied to the position-encoded score matrices from Eq. (4). These masked scores weight and combine value vectors V (attention) or input vectors X (SSMs) to produce output features y, as detailed in Eq. (6). Specifically, we multiply elementwise by the mask L (to zero out forbidden positions) and then use the resulting masked scores to weight the corresponding value vectors. For a Transformer attention layer, the values are V; for an SSM, the "values" are the input sequence X itself. In both cases, we obtain output features y as follows:

$$y = (Attn_{score} \circ L) \cdot V, \quad y = (SSD_{score} \circ L) \cdot X \tag{6}$$

Crucially, applying RoPE does not sacrifice the linear-time inference capability of either model. Because RoPE encodes position by a simple rotation of each input vector (Eqs. (2), (3)) before computing attention or SSM scores, it does not change the asymptotic complexity of those operations. A Transformer with RoPE can still perform auto-regressive generation in linear time by caching past K, V and rotating a new query q_{m+1} on the fly. Similarly, an SSM layer can still update its state recurrently one step at a time, incorporating the rotation $e^{i,\theta,m}$ into its state-update at step m. In summary, Unified RoPE preserves the fast inference properties of state-space models and efficient attention mechanisms, while providing a consistent positional encoding throughout a hybrid Transformer+SSM architecture.

$$Attn(Q,K,V)_i = \frac{\sum_{j=1}^{i}[\phi(R_{\Theta,i}q_i)][\varphi(R_{\Theta,j}k_j)]^{\top}v_j}{\sum_{j=1}^{i}\phi(R_{\Theta,i}q_i)\varphi(R_{\Theta,j}k_j)^{\top}}, \quad SSD(C,B,X)_i = \sum_{j=1}^{i}c_i R_{\Theta,i-j}^{d} b_j^{\top} x_j \tag{7}$$

Figure 1 illustrates the overall process, showing how absolute position rotations lead to relative-position-aware attention maps, which are then masked and used to produce outputs.

3 TransXSSM Architecture Design

With a unified positional embedding in place, we design an architecture that fully capitalizes on it. TransXSSM (Hybrid Transformer–SSM model) combines the high-level transformer block structure (self-attention + feed-forward network with residual connections) with state-space sequence modeling blocks in a single model. All modules share the same Unified RoPE encoding (as introduced in Section 2), so that positional information is seamlessly understood by every layer (see Figure 2).

Overall Architecture. For language modeling, TransXSSM follows the standard Transformer architecture outline with modifications to include SSM layers. As shown in Figure 3, input tokens are first embedded into vectors and then processed by a backbone of N stacked modules. Each module (analogous to a "layer") contains multiple State-Space (SS) components and a Self-Attention (SA) component. We adopt a 7:1 ratio of SS to SA blocks per module – that is, each module consists of 7 SSM-based sub-layers followed by 1 Transformer attention sub-layer (this ratio is motivated by prior studies on hybrid models [8] and our own experiments). Each sub-layer (SS or SA) is followed by a position-wise feed-forward network (FFN) to refine features, and wrapped with residual connections and normalization. We use RMSNorm normalization and add a residual skip connection around each SS/SA block and around its subsequent FFN, which is important for stabilizing training with long sequences (mitigating issues like state collapse observed in long-sequence RNNs [9]). Finally, the network output goes through a last normalization and a linear language model head to predict the next-token probabilities.

We highlight four **key design principles** of TransXSSM.

Principle 1: Unified Positional Rotary Encoding. Every SS and SA sub-layer in TransXSSM uses *the same* Unified RoPE as described in Section 2.2. Before computing attention scores or state-space updates, the inputs to each layer are endowed with the RoPE phase appropriate for their position. This guarantees that all parts of the model share a common positional frame of reference.

Principle 2: Hybrid Module Stacking. We intermix a larger number of state-space layers with occasional self-attention layers. In our design, each attention layer is placed after a block of seven SSM layers (7:1 ratio). This strategy, inspired by [5] and validated by our ablation studies, provides an excellent trade-off: the SSM layers efficiently handle the bulk of sequence length (contributing near-linear time complexity and high throughput), while the periodic attention layers inject global context mixing and strong relational reasoning. The result is a model that is both fast on long sequences and capable of complex in-context learning.

Principle 3: Feature Refinement via FFNs. After each attention or SSM sub-layer, we include a standard Transformer-style feed-forward network. These two-layer FFNs further transform and mix
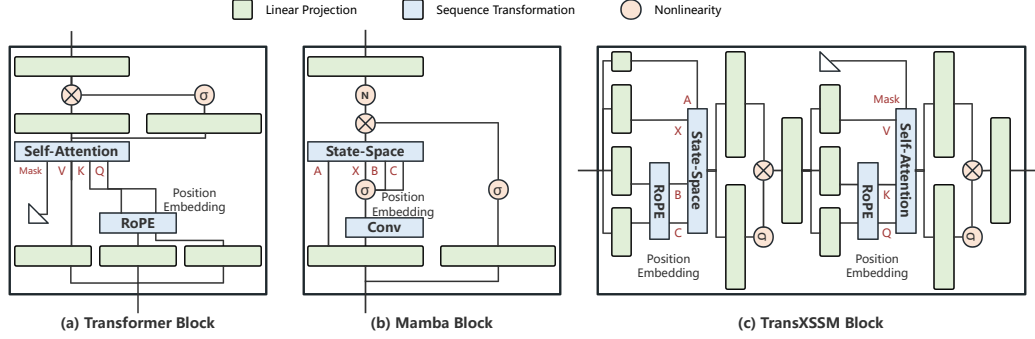
Figure 2: **TransXSSM Block**. Structure of the TransXSSM block, integrating State-Space (SS), Multi-Layer Perceptron (MLP), and Transformer-style Self-Attention (SA) blocks. All blocks utilize the proposed Unified RoPE.
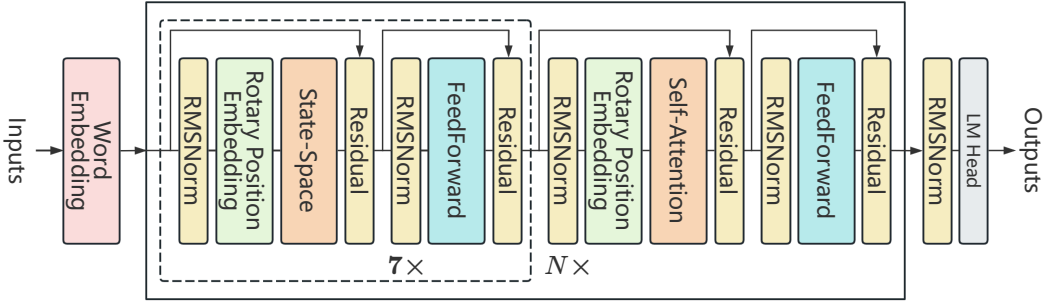

Figure 3: **TransXSSM Language Modeling Architecture**. Each input token is first converted to an embedding, then passes through N stacked hybrid modules. Each module contains 7 state-space (SS) layers and 1 self-attention (SA) layer, all employing the unified RoPE positional embedding. Every SS or SA layer is followed by an FFN (feed-forward network), and each pair (SS/SA + FFN) is wrapped with residual connections and RMSNorm normalization.

the features. They help "refine" the outputs of the SS or SA layers, and are important for maintaining model capacity and expressiveness at each layer of the hybrid model.

Principle 4: Stability and Regularization. Long sequence training can lead to instabilities such as state-vector drift or "state collapse" in SSMs. To ensure robust training, we employ RMSNorm normalization and residual skip connections around every sub-layer (both SS and SA) and its FFN. These measures keep activations bounded and gradients well-behaved over long sequences. Empirically, we found this design choice crucial for training TransXSSM on contexts up to 16K tokens without divergence. The hybrid architecture thus inherits the normalization practices that have proven effective in deep Transformers, applied uniformly across all its components.

By following these principles, TransXSSM effectively unifies two paradigms of sequence modeling. The SSM layers carry the heavy load of long-sequence processing with minimal cost, and the occasional attention layers inject powerful global modeling capabilities – all the while, Unified RoPE ensures that the Transformer and SSM components speak the same positional *language*. In the next section, we empirically demonstrate the benefits of this design, including improved speed, accuracy, and scalability compared to existing approaches.

4   Experiments

We conduct comprehensive experiments to validate the effectiveness, efficiency, and scalability of the proposed *Unified RoPE* and *TransXSSM* architecture. We aim to answer the following research questions: (**RQ1**) Does Unified RoPE effectively unify position encoding across Transformer and SSM modules, and how does it compare with other positional encodings? (Section 4.1) (**RQ2**) How does TransXSSM compare with state-of-the-art baselines on standard benchmarks? (Section 4.2) (**RQ3**) Does TransXSSM maintain its advantages at larger model scales? (Section 4.3)

6

Table 2: **Comparative Position Encoding Performance**. Models with $d_{model} = 256$ and sequence length 8192 were trained for 8000 steps. RoPE demonstrates superior perplexity and competitive throughput compared to Conv1d + D and a_t for State-Space modules and the hybrid setup. Self-attention modules inherently use RoPE in this comparison.

Modules	Training Steps	Conv1d + D PPL ↓ / S/S ↑	a_t PPL ↓ / S/S ↑	RoPE PPL ↓ / S/S ↑
Self-Attention	8,000	—	—	8.38 / 6.8
State-Space	8,000	8.56 / 7.6	8.62 / 9.4	8.33 / 9.2
State-Space + Self-Attention	8,000	8.48 / 7.2	8.56 / 8.5	8.18 / 8.4

4.1 Effectiveness of Unified RoPE and Comparison with Alternatives

This subsection addresses **RQ1**. To assess different position encoding methods, we conducted a comparative study. Models with $d_{model} = 256$ and sequence length 8192 were trained for 8000 steps. We evaluated Self-Attention, State-Space, and a hybrid State-Space + Self-Attention setup using three schemes: Conv1d + D (1D convolution + dense layer, State-Space only), a_t (recursive, State-Space only), and our proposed *Unified RoPE*. Conv1d + D and a_t are not directly applicable to standard Self-Attention modules.

The results in Table 2 show that **Unified RoPE** achieves the best perplexity for both standalone State-Space modules and hybrid configurations when processing long sequences, while maintaining competitive training efficiency. Specifically, in the hybrid State-Space + Self-Attention setup, TransXSSM with RoPE achieved the lowest perplexity (PPL=8.18), with throughput near top-performing state-space models.

Ablation studies further compared RoPE against alternatives (Conv1d + D and a_t). The results indicated that these alternatives were ineffective in hybrid scenarios due to incompatibility with self-attention modules, underscoring the necessity of unified positional encoding across modules. This study confirms that Unified RoPE uniquely provides spectral continuity and seamless integration across heterogeneous module types.

4.2 Performance Against State-of-the-Art Baselines

This subsection addresses **RQ2**. To ensure a fair comparison and demonstrate TransXSSM's advantages in efficiency and effectiveness, we retrained LlaMa3, Mamba2, Jamba, and our proposed TransXSSM. All models were trained under identical conditions, utilizing the Smollm-Corpus dataset, NeoX tokenizer, and consistent key hyperparameters (details in Appendix B.1).

Computational Efficiency and Throughput. Figure 4 shows training and evaluation throughput for 1.3B parameter models across sequence lengths. TransXSSM exhibits superior computational efficiency over LlaMa3 (Self-Attention) and Jamba (hybrid), especially at longer sequence lengths. Although Mamba2 (pure State-Space) is marginally more efficient, TransXSSM's integration of Self-Attention's modeling capabilities justifies this small difference.

Performance on Downstream Benchmarks. Downstream task evaluations (Table 3) further substantiate TransXSSM's effectiveness.

- **Cross-Task Stability and Superiority**: TransXSSM shows consistently strong performance across diverse tasks (e.g., MMLU, TriviaQA, PIQA, HellaSwag), often outperforming baselines. This suggests that unified position embedding effectively harmonizes State-Space (efficient long-sequence processing) and Self-Attention (accurate long-dependency capture) strengths.
- **Enhanced Reasoning Capabilities**: TransXSSM leads on tasks requiring commonsense reasoning and contextual understanding (e.g., HellaSwag, PIQA, Winogrande). On Winogrande, the TransXSSM-1.3B outperforms its LlaMa3 counterpart by nearly 6.7 points (62.09 vs 55.40), highlighting the reasoning benefits of our hybrid architecture with unified position encoding.

The improvement of TransXSSM over Jamba (another hybrid model) underscores our unified position embedding strategy's importance. While both mix Transformer and SSM blocks, TransXSSM's RoPE-facilitated consistent position representation across components is crucial for coherent contextual
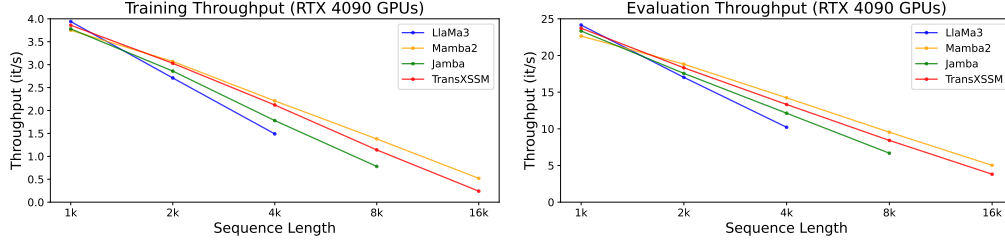
Figure 4: **Throughput Evaluation**. Training (left) and evaluation (right) throughput (iterations/second) for LlaMa3, Mamba2, Jamba, and TransXSSM at the 1.3B parameter scale across varying sequence lengths. TransXSSM surpasses LlaMa3 and Jamba in efficiency, while being slightly less efficient than Mamba2.
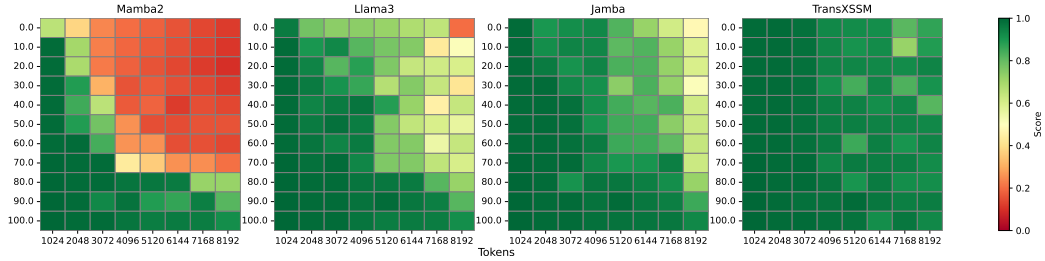

Figure 5: **Needle in a Haystack Performance (1.3B Models)**. Performance comparison of LlaMa3, Mamba2, Jamba, and TransXSSM (1.3B scale) on the "needle in a haystack" task. TransXSSM, with its unified position encoding, exhibits strong performance.

Table 3: **Downstream Task Performance**. Validation results for LlaMa3, Mamba2, Jamba, and TransXSSM on various downstream tasks, trained under identical conditions. Best results per parameter scale are in **bold**, second best are underlined. TransXSSM generally outperforms other models across most tasks and scales.

MODEL	MMLU ACC ↑	TRIVIAQA QEM ↑	ARC ACC ↑	PIQA ACC ↑	HELLASWAG ACC ↑	OBQA ACC ↑	WINOGRANDE ACC ↑	AVG
LlaMa3-320M	<u>33.65</u>	8.86	**51.68**	71.42	52.30	<u>37.02</u>	53.15	43.99
Mamba2-320M	33.10	<u>9.36</u>	50.72	70.24	48.62	35.16	54.17	43.07
Jamba-320M	33.12	9.32	50.80	<u>71.88</u>	<u>52.92</u>	36.73	<u>55.24</u>	<u>44.31</u>
TransXSSM-320M	**34.45**	**10.38**	<u>51.57</u>	**73.32**	**53.79**	**37.42**	**55.61**	**45.22**
LlaMa3-1.3B	<u>37.86</u>	20.66	**59.82**	76.05	61.65	**41.15**	55.40	50.36
Mamba2-1.3B	36.28	21.28	58.02	72.26	59.48	37.98	58.72	49.07
Jamba-1.3B	37.43	<u>21.60</u>	59.33	<u>76.58</u>	<u>62.33</u>	40.82	<u>59.20</u>	<u>51.07</u>
TransXSSM-1.3B	**39.08**	**23.02**	<u>59.69</u>	**78.15**	**63.63**	<u>41.12</u>	**62.09**	**52.44**

understanding and superior performance, vital for tasks needing multi-hop reasoning or precise positional awareness.

Long-Context Retrieval (Needle-in-a-Haystack Task). We further evaluated architectures on the "needle in a haystack" synthetic retrieval task. This task tests long-context extraction by embedding a "needle" (random sentence) in a "haystack" (long document) for retrieval. Figure 5 shows that TransXSSM exhibited strong long-context retrieval, outperforming baselines in this challenging scenario.

Task-Specific Strengths and Overall Balance. As shown in Table 3, it shows pure Transformers (LlaMa3) excel in directed reasoning (e.g., ARC), while pure State-Space models (Mamba2) are strong in direct knowledge extraction (e.g., TriviaQA). TransXSSM, with its 7:1 State-Space to Self-Attention ratio and unified position embedding, retains these respective strengths while achieving a superior overall performance balance, often leading on most tasks.

4.3   Scalability of TransXSSM Advantages at Larger Model Scales

To address **RQ3**, we analyze TransXSSM's advantages when scaling from 320M to 1.3B parameters, comparing its performance progression against baselines (Llama3, Mamba2, Jamba) using configurations from Table 4 of Appendix B.1.

Table 3 shows that when scaling from 320M to 1.3B parameters, **TransXSSM's average performance gain (+7.22 points, from 45.22 to 52.44) surpasses that of LlaMa3 (+6.37), Mamba2 (+6.0), and Jamba (+6.76)**. This demonstrates TransXSSM's superior scalability, with its hybrid modeling advantages becoming more pronounced at larger scales. This consistent outperformance during scaling confirms the robustness and effective scalability of TransXSSM's architectural advantages. The unified position embedding provides a solid foundation for effectively fusing different architectural paradigms, leading to a better balance of computational efficiency and model capability at larger scales.

5   Related Work

Hybrid modeling seeks to merge the strengths of different architectures. Recently, architectures combining Transformers and State Space Models (SSMs) have gained attention for their potential to overcome the limitations of unimodal systems. Early hybrid models explored various integration strategies. For instance, studies combined S4 layers with local attention [10] or utilized SSM layers preceding Transformer blocks [11], primarily showing promise on small to medium-scale models. However, simple interleaving of Mamba and attention layers [12], or replacing some attention layers with SSM variants like H3 [13] and Hyena [14], sometimes struggled to outperform pure Mamba or attention models when scaled, as seen with StripedHyena versus Mistral-7B [12, 15, 16]. Other attempts included varying the combination order of SSM and self-attention in specific domains like speech recognition [17, 18], or replacing MLP layers in Transformers with Mamba layers [19]. The Jamba architecture [3] marked a significant advancement by achieving a balance between performance, throughput, and memory usage through a carefully designed interleaving ratio of Transformer-Mamba layers, establishing it as a pioneering production-level large-scale attention-SSM hybrid model. Jamba leverages Transformer layers for robust modeling and Mamba layers for efficient long-sequence processing. Our TransXSSM architecture builds upon these efforts by introducing a novel *unified* Rotary Position Embedding (RoPE) strategy, compatible with both self-attention and state-space duality. This consistent positional understanding across heterogeneous blocks aims to address a critical, often overlooked issue in prior hybrid designs: positional spectrum discontinuity. We posit this as a key factor in our model's enhanced performance and scalability.

6   Conclusion

This paper introduces a **Unified RoPE** method, addressing the core challenge of inconsistent position encoding in hybrid sequence transformation architectures. We theoretically demonstrate the effective application of Rotary Position Embedding to state-space duality algorithms and, based on this, designed the **TransXSSM** architecture, achieving an efficient fusion of self-attention and state-space models. Experimental results show that TransXSSM significantly outperforms pure self-attention models, pure state-space models, and other hybrid architectures across various downstream tasks, particularly excelling in knowledge-intensive and reasoning-intensive tasks. Moreover, TransXSSM's advantages grow with model scale, indicating strong scalability and suggesting it can effectively leverage increased capacity.

A   RoPE for SSD

Proof of Equation 4. by definition, $h_0 = B_0 x_0$. By induction,

$$h_t = A_t \ldots A_1 B_0 x_0 + A_t \ldots A_2 B_1 x_1 + \cdots + A_t A_{t-1} B_{t-2} x_{t-2} + A_t B_{t-1} x_{t-1} + B_t x_t$$

$$= \sum_{s=0}^{t} A_{t:s}^{\times} B_s x_s$$

Multiplying by C_t to produce y_t, and vectorizing the equation to $t \in [\mathtt{T}]$ ($\mathtt{T}$ is the sequence length), we derive the matrix transformation form of SSD.

$$y_t = \sum_{s=0}^{t} C_t^{\top} A_{t:s}^{\times} B_s x_s$$
$$y = \mathsf{SSD}(A, B, C)(x) = Mx$$
$$M_{ji} := C_j^{\top} A_j \cdots A_{i+1} B_i$$

Then the matrix form of SSD is represented using SSS (Sequentially Semiseparable) as $M = \mathsf{SSS}(A, B, C)$, where $M_{ji} = C_j^{\top} A_{j:i} B_i$, and then considering A is just a scalar, rearranged as:

$$M_{ji} = A_{j:i} \cdot (C_j^{\top} B_i)$$

Vectorized as:

$$L := \mathsf{1SS}(a)$$
$$M = L \circ (CB^{\top})$$

Finally, it is proved that the matrix transformation form of SSD is equivalent to Attention $(L \circ QK^{\top}) \cdot V = (L \circ CB^{\top}) \cdot X$.

Now we have enough theoretical support to give rotational positional encoding to the C and B matrices in SSD.

$$C_m = f_C(x_m, m)$$
$$B_n = f_B(x_n, n)$$

C_m represents the output weight matrix of the m-th token corresponding to the word vector x_m integrated with the position information m, B_n represents the input weight matrix of the n-th token corresponding to the word vector x_n integrated with the position information n.

To utilize the relative positional information between tokens, we assume that the inner product operation between the C_m vector and the B_n vector can be represented by a function g, where the input of the function g is the word embedding vectors x_m and x_n, and their relative positional information $m - n$, the inner product of C_m and B_n and their relative positional information $m - n$ is defined as

$$< f_C(x_m, m), f_B(x_n, n) >= g(x_m, x_n, m - n)$$

Now, assuming the word embedding vector dimension is $d = 2$, we have $f_C(x_m, n) = (W_C x_m)e^{im\theta}$, for the first half of the formula $W_C x_m$, we know that W_C is a two-dimensional matrix, x_m is a two-dimensional vector, the result of the multiplication is naturally a two-dimensional vector, represented by C_m

$$C_m = \begin{bmatrix} C_m^{(1)} \\ C_m^{(2)} \end{bmatrix} = W_C x_m = \begin{bmatrix} W_C^{(11)} & W_C^{(12)} \\ W_C^{(21)} & W_C^{(22)} \end{bmatrix} \begin{bmatrix} x_m^{(1)} \\ x_m^{(2)} \end{bmatrix}$$

For the second half $e^{im\theta}$, according to Euler's formula $e^{ix} = \cos(x) + i\sin(x)$, we have

$$e^{im\theta} = \cos(m\theta) + i\sin(m\theta)$$

We know

$$f_C(x_m, m) = (W_C x_m)e^{im\theta} = C_m e^{im\theta}$$

C_m is represented in complex form,

$$C_m = \left[C_m^{(1)}, C_m^{(2)}\right] = \left[C_m^{(1)} + iC_m^{(2)}\right]$$

Thus,

$$f_C(x_m, m) = C_m e^{im\theta} = \left[C_m^{(1)} + iC_m^{(2)}\right]e^{im\theta}$$

According to the above derivation, we know that $f_C(x_m, m)$ is the product of two complex numbers,

$$f_C(x_m, m) = C_m e^{im\theta} = \left[C_m^{(1)} + iC_m^{(2)}\right] \times (\cos(m\theta) + i\sin(m\theta))$$

Considering the following two formulas about complex numbers

$$(a + ib) \times (c + id) = ac + ibc + iad + i^2bd = (ac - bd) + i(bc + ad)$$
$$i^2 = -1$$

We have

$$
\begin{aligned}
C_m e^{im\theta} &= \left[C_m^{(1)} + iC_m^{(2)}\right] \times (\cos(m\theta) + i\sin(m\theta)) \\
&= \left[C_m^{(1)}\cos(m\theta) - C_m^{(2)}\sin(m\theta)\right] + i\left[C_m^{(2)}\cos(m\theta) + C_m^{(1)}\sin(m\theta)\right]
\end{aligned}
$$

Expressing this result as a real vector,

$$C_m e^{im\theta} = \left[C_m^{(1)}\cos(m\theta) - C_m^{(2)}\sin(m\theta), C_m^{(2)}\cos(m\theta) + C_m^{(1)}\sin(m\theta)\right]$$

Therefore, C_m multiplied by a rotation matrix is obtained.

$$
\begin{aligned}
f_C(x_m, m) &= (W_C x_m)e^{im\theta} = C_m e^{im\theta} \\
&= \left[C_m^{(1)}\cos(m\theta) - C_m^{(2)}\sin(m\theta), C_m^{(2)}\cos(m\theta) + C_m^{(1)}\sin(m\theta)\right] \\
&= \begin{bmatrix} \cos(m\theta) & -\sin(m\theta) \\ \sin(m\theta) & \cos(m\theta) \end{bmatrix} \begin{bmatrix} C_m^{(1)} \\ C_m^{(2)} \end{bmatrix}
\end{aligned}
$$

Similarly, B_n vector can be obtained

$$
\begin{aligned}
f_B(x_n, n) &= (W_B x_n)e^{in\theta} = B_n e^{in\theta} \\
&= \left[B_n^{(1)}\cos(n\theta) - B_n^{(2)}\sin(n\theta), B_n^{(2)}\cos(n\theta) + B_n^{(1)}\sin(n\theta)\right] \\
&= \begin{bmatrix} \cos(n\theta) & -\sin(n\theta) \\ \sin(n\theta) & \cos(n\theta) \end{bmatrix} \begin{bmatrix} B_n^{(1)} \\ B_n^{(2)} \end{bmatrix}
\end{aligned}
$$

The function g can be represented as

$$g(x_m, x_n, m - n) = R\left[(W_C x_m)(W_B x_n)^* e^{i(m-n)\theta}\right]$$

where R represents the real part of the complex number x, $(W_C x_m)(W_B x_n)^*$ represents the conjugate of the product of two complex numbers. Considering

$$z = a + ib$$
$$z^* = a - ib$$

we have

$$W_C x_m = C_m = C_m^{(1)} + iC_m^{(2)}$$
$$W_B x_n = B_n = B_n^{(1)} + iB_n^{(2)}$$
$$(W_B x_n)^* = B_n^* = B_n^{(1)} - iB_n^{(2)}$$
$$e^{i(m-n)\theta} = \cos((m-n)\theta) + i\sin((m-n)\theta)$$

We now want to prove that

$$
\begin{aligned}
&g(x_m, x_n, m - n) \\
&= R\left[(W_C x_m)(W_B x_n)^* e^{i(m-n)\theta}\right] \\
&= R\left[(C_m^{(1)} + iC_m^{(2)})(B_n^{(1)} - iB_n^{(2)})(\cos((m-n)\theta) + i\sin((m-n)\theta))\right] \\
&= R\left[((C_m^{(1)} B_n^{(1)} + C_m^{(2)} B_n^{(2)}) + i(C_m^{(2)} B_n^{(1)} - C_m^{(1)} B_n^{(2)}))(\cos((m-n)\theta) + i\sin((m-n)\theta))\right] \\
&= (C_m^{(1)} B_n^{(1)} + C_m^{(2)} B_n^{(2)})\cos((m-n)\theta) - (C_m^{(2)} B_n^{(1)} - C_m^{(1)} B_n^{(2)})\sin((m-n)\theta)
\end{aligned}
$$

Recalling the vectorized form of SSD, the C vector at position m and the B vector at position n will perform an inner product operation, that is,

$$f_C(x_m, m) = \left[C_m^{(1)}\cos(m\theta) - C_m^{(2)}\sin(m\theta), C_m^{(2)}\cos(m\theta) + C_m^{(1)}\sin(m\theta)\right]$$
$$f_B(x_n, n) = \left[B_n^{(1)}\cos(n\theta) - B_n^{(2)}\sin(n\theta), B_n^{(2)}\cos(n\theta) + B_n^{(1)}\sin(n\theta)\right]$$

We have

$$
\begin{aligned}
< f_C(x_m, m), f_B(x_n, n) > = &\left[C_m^{(1)}\cos(m\theta) - C_m^{(2)}\sin(m\theta)\right]\left[B_n^{(1)}\cos(n\theta) - B_n^{(2)}\sin(n\theta)\right] \\
&+ \left[C_m^{(2)}\cos(m\theta) + C_m^{(1)}\sin(m\theta)\right]\left[B_n^{(2)}\cos(n\theta) + B_n^{(1)}\sin(n\theta)\right] \\
= &\, C_m^{(1)}\cos(m\theta)B_n^{(1)}\cos(n\theta) - C_m^{(1)}\cos(m\theta)B_n^{(2)}\sin(n\theta) \\
&- C_m^{(2)}\sin(m\theta)B_n^{(1)}\cos(n\theta) + C_m^{(2)}\sin(m\theta)B_n^{(2)}\sin(n\theta) \\
&+ C_m^{(2)}\cos(m\theta)B_n^{(2)}\cos(n\theta) + C_m^{(2)}\cos(m\theta)B_n^{(1)}\sin(n\theta) \\
&+ C_m^{(1)}\sin(m\theta)B_n^{(2)}\cos(n\theta) + C_m^{(1)}\sin(m\theta)B_n^{(1)}\sin(n\theta)
\end{aligned}
$$

Considering

$$\sin(a + b) = \sin(a)\cos(b) + \cos(a)\sin(b)$$
$$\sin(a - b) = \sin(a)\cos(b) - \cos(a)\sin(b)$$
$$\cos(a + b) = \cos(a)\cos(b) - \sin(a)\sin(b)$$
$$\cos(a - b) = \cos(a)\cos(b) + \sin(a)\sin(b)$$

We have

$$
\begin{aligned}
&< f_C(x_m, m), f_B(x_n, n) > \\
&= C_m^{(1)} B_n^{(1)} (\cos(m\theta)\cos(n\theta) + \sin(m\theta)\sin(n\theta)) \\
&+ C_m^{(1)} B_n^{(2)} (-\cos(m\theta)\sin(n\theta) + \sin(m\theta)\cos(n\theta)) \\
&+ C_m^{(2)} B_n^{(1)} (-\sin(m\theta)\cos(n\theta) + \cos(m\theta)\sin(n\theta)) \\
&+ C_m^{(2)} B_n^{(2)} (\sin(m\theta)\sin(n\theta) + \cos(m\theta)\cos(n\theta)) \\
&= C_m^{(1)} B_n^{(1)} \cos((m-n)\theta) + C_m^{(1)} B_n^{(2)} \sin((m-n)\theta) \\
&- C_m^{(2)} B_n^{(1)} \sin((m-n)\theta) + C_m^{(2)} B_n^{(2)} \cos((m-n)\theta) \\
&= (C_m^{(1)} B_n^{(1)} + C_m^{(2)} B_n^{(2)}) \cos((m-n)\theta) + (C_m^{(1)} B_n^{(2)} - C_m^{(2)} B_n^{(1)}) \sin((m-n)\theta) \\
&= (C_m^{(1)} B_n^{(1)} + C_m^{(2)} B_n^{(2)}) \cos((m-n)\theta) - (C_m^{(2)} B_n^{(1)} - C_m^{(1)} B_n^{(2)}) \sin((m-n)\theta) \\
&= g(x_m, x_n, m-n)
\end{aligned}
$$

It is proved that the inner product of the C vector at position m and the B vector at position n is the function g.

Finally, using the matrix-vector multiplication form

$$
\begin{aligned}
&< f_C(x_m, m), f_B(x_n, n) > \\
&= \left[\begin{bmatrix} \cos(m\theta) & -\sin(m\theta) \\ \sin(m\theta) & \cos(m\theta) \end{bmatrix} \begin{bmatrix} C_m^{(1)} \\ C_m^{(2)} \end{bmatrix} \right]^T \left[\begin{bmatrix} \cos(n\theta) & -\sin(n\theta) \\ \sin(n\theta) & \cos(n\theta) \end{bmatrix} \begin{bmatrix} B_n^{(1)} \\ B_n^{(2)} \end{bmatrix} \right] \\
&= \begin{bmatrix} C_m^{(1)} & C_m^{(2)} \end{bmatrix} \begin{bmatrix} \cos(m\theta) & \sin(m\theta) \\ -\sin(m\theta) & \cos(m\theta) \end{bmatrix} \begin{bmatrix} \cos(n\theta) & -\sin(n\theta) \\ \sin(n\theta) & \cos(n\theta) \end{bmatrix} \begin{bmatrix} B_n^{(1)} \\ B_n^{(2)} \end{bmatrix}
\end{aligned}
$$

Expanding the product of the two rotary matrices, we have

$$
\begin{bmatrix} \cos(m\theta)\cos(n\theta) + \sin(m\theta)\sin(n\theta) & -\cos(m\theta)\sin(n\theta) + \sin(m\theta)\cos(n\theta) \\ -\sin(m\theta)\cos(n\theta) + \cos(m\theta)\sin(n\theta) & \sin(m\theta)\sin(n\theta) + \cos(m\theta)\cos(n\theta) \end{bmatrix}
$$

Finally, we get

$$
< f_C(x_m, m), f_B(x_n, n) > = \begin{bmatrix} C_m^{(1)} & C_m^{(2)} \end{bmatrix} \begin{bmatrix} \cos((m-n)\theta) & -\sin((m-n)\theta) \\ \sin((m-n)\theta) & \cos((m-n)\theta) \end{bmatrix} \begin{bmatrix} B_n^{(1)} \\ B_n^{(2)} \end{bmatrix}
$$

The above derivation is only for the case of word embedding dimension $d = 2$, when $d > 2$, the two-dimensional case can be extended to any dimension as follows

$$
f_{\{C,B\}}(x_m, m) = R_{\Theta,m}^d W_{\{C,B\}} x_m
$$

The inner product satisfies linearity, so for any even-dimensional RoPE, we can represent it as a concatenation of the two-dimensional case, that is, grouping the elements of the word embedding vector in pairs

$$
R_{\Theta,m}^d = \begin{bmatrix}
\cos m\theta_0 & -sinm\theta_0 & 0 & 0 & \dots & 0 & 0 \\
\sin m\theta_0 & \cos m\theta_0 & 0 & 0 & \dots & 0 & 0 \\
0 & 0 & \cos m\theta_1 & -sinm\theta_1 & \dots & 0 & 0 \\
0 & 0 & \sin m\theta_1 & \cos m\theta_1 & \dots & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & 0 & \dots & \cos m\theta_{d/2} & -sinm\theta_{d/2-1} \\
0 & 0 & 0 & 0 & \dots & \sin m\theta_{d/2} & \cos m\theta_{d/2-1}
\end{bmatrix}
$$

Each group applies the same rotation operation and the rotation angle of each group is calculated as follows:

$$\Theta = \{\theta_i = 10000^{-2(i-1)/d}, i \in [1, 2, \ldots, d/2]\}$$

RoPE is a kind of relative positional encoding, and relative positional encoding is a special form of Toeplitz matrix

$$\begin{bmatrix} 0 & & & & & \\ 1 & 0 & & & & \\ 2 & 1 & 0 & & & \\ 3 & 2 & 1 & 0 & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \\ n-1 & n-2 & n-3 & \ldots & 1 & 0 \end{bmatrix}$$

We can know that the position distribution after RoPE is unbalanced, 0 appears most frequently as the low bit, and n-1 appears least frequently as the high bit, which also leads to a problem of RoPE, its high bit is not sufficiently trained, and the generalization ability is not as good as the low bit. We take the average value of the effective sequence length of the training data as the base, for the sequence length greater than the base, we have

$$C \times \max(1, \log_{base} n)$$

The part of the sequence length within the base is not affected, and the part of the sequence length greater than the base is expanded according to the ratio of $\log_{base} n$, so that the problem of insufficient generalization ability of the high bit can be solved.

$\square$

B Additional Experimental Details

B.1 Experimental Setup for Baseline Comparison

Model Architectures Settings.   To ensure a fair comparison and avoid discrepancies from varying training data, we retrained four distinct model architectures: Llama3 [20] (Self-Attention based), Mamba2 [2] (State-Space based), Jamba [3] (a hybrid of Transformer and Mamba2 blocks), and our proposed TransXSSM. All models were trained from scratch on the Smollm-Corpus [21] dataset using the NeoX tokenizer [22].

Model Scales Settings.   We experimented with two model scales, 320M and 1.3B parameters, detailed in Table 4. For the 320M scale, models have $d_{model} = 768$, 24 layers, 12 attention heads, a learning rate of 3e-4, and a batch size of 1M tokens. For the 1.3B scale, $d_{model} = 2048$, 24 layers, 32 attention heads, a learning rate of 2e-4, and a batch size of 2M tokens. State-Space components within Mamba2, Jamba, and TransXSSM uniformly use $d_{state} = 128$ and $chunk\_len = 256$.

Model Training Settings.   Training was conducted using Nvidia's open-source PyTorch container (version 24.2), compatible with the mamba-ssm library's CUDA kernels, and managed by the Trainer class from the Transformers library [23]. We employed the AdamW optimizer ($\beta_1 = 0.9, \beta_2 = 0.999, weight\_decay = 0.01$), with a linear warmup for 10% of total steps followed by a cosine decay to 10% of the initial learning rate. No bias terms were used, and LayerNorm was replaced by RMSNorm.

Table 4: **Model Parameters**. Key hyperparameters for Llama3, Mamba2, Jamba, and TransXSSM models at 320M and 1.3B scales. Parameters were aligned where possible to ensure comparable total and active parameter counts.

MODEL	d_{model}	n_{layers}	n_{heads}	d_{state}	chunk_len	LEANING RATE	BATCH SIZE
LlaMa3-320M	768	24	12	—	—	3e-4	1M tokens
Mamba2-320M	768	24	12	128	256	3e-4	1M tokens
Jamba-320M	768	24	12	128	256	3e-4	1M tokens
TransXSSM-320M	768	24	12	128	256	3e-4	1M tokens
LlaMa3-1.3B	2048	24	32	—	—	2e-4	2M tokens
Mamba2-1.3B	2048	24	32	128	256	2e-4	2M tokens
Jamba-1.3B	2048	24	32	128	256	2e-4	2M tokens
TransXSSM-1.3B	2048	24	32	128	256	2e-4	2M tokens

Downstream Tasks Settings.   Downstream task evaluation utilized the EleutherAI LM Evaluation Harness [24]. The benchmark suite included MMLU [25], TriviaQA [26], ARC [27], PIQA [28], HellaSwag [29], OBQA [30], and Winogrande [31]. Consistent evaluation settings were applied across all 320M and 1.3B models to ensure fair and comparable results.

References

[1] Z. Ghahramani and G. E. Hinton, "Variational learning for switching state-space models," *Neural Computation*, vol. 12, no. 4, pp. 831–864, 2000.

[2] T. Dao and A. Gu, "Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality," in *International Conference on Machine Learning (ICML)*, 2024.

[3] O. Lieber, B. Lenz, H. Bata, G. Cohen, J. Osin, I. Dalmedigos, E. Safahi, S. Meirom, Y. Belinkov, S. Shalev-Shwartz *et al.*, "Jamba: A hybrid transformer-mamba language model," *arXiv preprint arXiv:2403.19887*, 2024.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[5] R. Waleffe, W. Byeon, D. Riach, B. Norick, V. Korthikanti, T. Dao, A. Gu, A. Hatamizadeh, S. Singh, D. Narayanan *et al.*, "An empirical study of mamba-based language models," *arXiv preprint arXiv:2406.07887*, 2024.

[6] J. Su, Y. Lu, S. Pan, A. Murtadha, B. Wen, and Y. Liu, "Roformer: Enhanced Transformer with rotary position embedding," *arXiv preprint arXiv:2104.09864*, 2021.

[7] B. N. Patro and V. S. Agneeswaran, "Mamba-360: Survey of state space models as transformer alternative for long sequence modelling: Methods, applications, and challenges," *arXiv preprint arXiv:2404.16112*, 2024.

[8] A. Blakeman, A. Basant, A. Khattar, A. Renduchintala, A. Bercovich, A. Ficek, A. Bjorlin, A. Taghibakhshi, A. S. Deshmukh, A. S. Mahabaleshwarkar *et al.*, "Nemotron-h: A family of accurate and efficient hybrid mamba-transformer models," *arXiv preprint arXiv:2504.03624*, 2025.

[9] Y. Chen, X. Zhang, S. Hu, X. Han, Z. Liu, and M. Sun, "Stuffed mamba: State collapse and state capacity of rnn-based long-context modeling," *arXiv preprint arXiv:2410.07145*, 2024.

[10] S. Zuo, X. Liu, J. Jiao, D. Charles, E. Manavoglu, T. Zhao, and J. Gao, "Efficient long sequence modeling via state space augmented Transformer," *arXiv preprint arXiv:2212.08136*, 2022.

[11] J. Pilault, M. Fathi, O. Firat, C. Pal, P.-L. Bacon, and R. Goroshin, "Block-state transformers," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[12] A. Gu and T. Dao, "Mamba: Linear-time sequence modeling with selective state spaces," *arXiv preprint arXiv:2312.00752*, 2023.

[13] D. Y. Fu, T. Dao, K. K. Saab, A. W. Thomas, A. Rudra, and C. Re, "Hungry hungry hippos: Towards language modeling with state space models," in *The Eleventh International Conference on Learning Representations*, 2022.

[14] M. Poli, S. Massaroli, E. Nguyen, D. Y. Fu, T. Dao, S. Baccus, Y. Bengio, S. Ermon, and C. Ré, "Hyena hierarchy: Towards larger convolutional language models," in *The International Conference on Machine Learning (ICML)*, 2023.

[15] M. Poli, J. Wang, S. Massaroli, J. Quesnelle, R. Carlow, E. Nguyen, and A. Thomas, "StripedHyena: Moving Beyond Transformers with Hybrid Signal Processing Models," https://github.com/togethercomputer/stripedhyena, 2023.

[16] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. l. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier *et al.*, "Mistral 7b," *arXiv preprint arXiv:2310.06825*, 2023.

[17] Y. Fathullah, C. Wu, Y. Shangguan, J. Jia, W. Xiong, J. Mahadeokar, C. Liu, Y. Shi, O. Kalinli, M. Seltzer, and M. J. F. Gales, "Multi-head state space model for speech recognition," in *Proceedings of INTERSPEECH 2023*, 2023, pp. 241–245.

[18] G. Saon, A. Gupta, and X. Cui, "Diagonal state space augmented Transformers for speech recognition," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[19] J. Park, J. Park, Z. Xiong, N. Lee, J. Cho, S. Oymak, K. Lee, and D. Papailiopoulos, "Can mamba learn how to learn? a comparative study on in-context learning tasks," in *The International Conference on Machine Learning (ICML)*, 2024.

[20] A. 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, B. Roziere, B. Biron, B. Tang, B. Chern, C. Caucheteux, C. Nayak, C. Bi, C. Marra, C. McConnell, C. Keller, C. Touret, C. Wu, C. Wong, C. C. Ferrer, C. Nikolaidis, D. Allonsius, D. Song, D. Pintz, D. Livshits, D. Wyatt, D. Esiobu, D. Choudhary, D. Mahajan, D. Garcia-Olano, D. Perino, D. Hupkes, E. Lakomkin, E. AlBadawy, E. Lobanova, E. Dinan, E. M. Smith, F. Radenovic, F. Guzmán, F. Zhang, G. Synnaeve, G. Lee, G. L. Anderson, G. Thattai, G. Nail, G. Mialon, G. Pang, G. Cucurell, H. Nguyen, H. Korevaar, H. Xu, H. Touvron, I. Zarov, I. A. Ibarra, I. Kloumann, I. Misra, I. Evtimov, J. Zhang, J. Copet, J. Lee, J. Geffert, J. Vranes, J. Park, J. Mahadeokar, J. Shah, J. van der Linde, J. Billock, J. Hong, J. Lee, J. Fu, J. Chi, J. Huang, J. Liu, J. Wang, J. Yu, J. Bitton, J. Spisak, J. Park, J. Rocca, J. Johnstun, J. Saxe, J. Jia, K. V. Alwala, K. Prasad, K. Upasani, K. Plawiak, K. Li, K. Heafield, K. Stone, K. El-Arini, K. Iyer, K. Malik, K. Chiu, K. Bhalla, K. Lakhotia, L. Rantala-Yeary, L. van der Maaten, L. Chen, L. Tan, L. Jenkins, L. Martin, L. Madaan, L. Malo, L. Blecher, L. Landzaat, L. de Oliveira, M. Muzzi, M. Pasupuleti, M. Singh, M. Paluri, M. Kardas, M. Tsimpoukelli, M. Oldham, M. Rita, M. Pavlova, M. Kambadur, M. Lewis, M. Si, M. K. Singh, M. Hassan, N. Goyal, N. Torabi, N. Bashlykov, N. Bogoychev, N. Chatterji, N. Zhang, O. Duchenne, O. Çelebi, P. Alrassy, P. Zhang, P. Li, P. Vasic, P. Weng, P. Bhargava, P. Dubal, P. Krishnan, P. S. Koura, P. Xu, Q. He, Q. Dong, R. Srinivasan, R. Ganapathy, R. Calderer, R. S. Cabral, R. Stojnic, R. Raileanu, R. Maheswari, R. Girdhar, R. Patel, R. Sauvestre, R. Polidoro, R. Sumbaly, R. Taylor, R. Silva, R. Hou, R. Wang, S. Hosseini, S. Chennabasappa, S. Singh, S. Bell, S. S. Kim, S. Edunov, S. Nie, S. Narang, S. Raparthy, S. Shen, S. Wan, S. Bhosale, S. Zhang, S. Vandenhende, S. Batra, S. Whitman, S. Sootla, S. Collot, S. Gururangan, S. Borodinsky, T. Herman, T. Fowler, T. Sheasha, T. Georgiou, T. Scialom, T. Speckbacher, T. Mihaylov, T. Xiao, U. Karn, V. Goswami, V. Gupta, V. Ramanathan, V. Kerkez, V. Gonguet, V. Do, V. Vogeti, V. Albiero, V. Petrovic, W. Chu, W. Xiong, W. Fu, W. Meers, X. Martinet, X. Wang, X. Wang, X. E. Tan, X. Xia, X. Xie, X. Jia, X. Wang, Y. Goldschlag, Y. Gaur, Y. Babaei, Y. Wen, Y. Song, Y. Zhang, Y. Li, Y. Mao, Z. D. Coudert, Z. Yan, Z. Chen, Z. Papakipos, A. Singh, A. Srivastava, A. Jain, A. Kelsey, A. Shajnfeld, A. Gangidi, A. Victoria, A. Goldstand, A. Menon, A. Sharma, A. Boesenberg, A. Baevski, A. Feinstein, A. Kallet, A. Sangani, A. Teo, A. Yunus, A. Lupu, A. Alvarado, A. Caples, A. Gu, A. Ho, A. Poulton, A. Ryan, A. Ramchandani, A. Dong, A. Franco, A. Goyal, A. Saraf, A. Chowdhury, A. Gabriel, A. Bharambe, A. Eisenman, A. Yazdan, B. James, B. Maurer, B. Leonhardi, B. Huang, B. Loyd, B. D. Paola, B. Paranjape, B. Liu, B. Wu, B. Ni, B. Hancock, B. Wasti, B. Spence, B. Stojkovic, B. Gamido, B. Montalvo, C. Parker, C. Burton, C. Mejia, C. Liu, C. Wang, C. Kim, C. Zhou, C. Hu, C.-H. Chu, C. Cai, C. Tindal, C. Feichtenhofer, C. Gao, D. Civin, D. Beaty, D. Kreymer, D. Li, D. Adkins, D. Xu, D. Testuggine, D. David, D. Parikh, D. Liskovich, D. Foss, D. Wang, D. Le, D. Holland, E. Dowling, E. Jamil, E. Montgomery, E. Presani, E. Hahn, E. Wood, E.-T. Le, E. Brinkman, E. Arcaute, E. Dunbar, E. Smothers, F. Sun, F. Kreuk, F. Tian, F. Kokkinos, F. Ozgenel, F. Caggioni, F. Kanayet, F. Seide, G. M. Florez, G. Schwarz, G. Badeer, G. Swee, G. Halpern, G. Herman, G. Sizov, Guangyi, Zhang, G. Lakshminarayanan, H. Inan, H. Shojanazeri, H. Zou, H. Wang, H. Zha, H. Habeeb, H. Rudolph, H. Suk, H. Aspegren, H. Goldman, H. Zhan, I. Damlaj, I. Molybog, I. Tufanov, I. Leontiadis, I.-E. Veliche, I. Gat, J. Weissman, J. Geboski, J. Kohli, J. Lam, J. Asher, J.-B. Gaya, J. Marcus, J. Tang, J. Chan, J. Zhen, J. Reizenstein, J. Teboul, J. Zhong, J. Jin, J. Yang, J. Cummings, J. Carvill, J. Shepard, J. McPhie, J. Torres, J. Ginsburg, J. Wang, K. Wu, K. H. U, K. Saxena, K. Khandelwal, K. Zand, K. Matosich, K. Veeraraghavan, K. Michelena, K. Li, K. Jagadeesh, K. Huang, K. Chawla, K. Huang, L. Chen, L. Garg, L. A, L. Silva, L. Bell, L. Zhang, L. Guo, L. Yu, L. Moshkovich, L. Wehrstedt, M. Khabsa, M. Avalani, M. Bhatt, M. Mankus, M. Hasson, M. Lennie, M. Reso, M. Groshev, M. Naumov, M. Lathi, M. Keneally, M. Liu, M. L. Seltzer, M. Valko, M. Restrepo, M. Patel, M. Vyatskov, M. Samvelyan, M. Clark, M. Macey, M. Wang, M. J. Hermoso, M. Metanat, M. Rastegari, M. Bansal, N. Santhanam, N. Parks, N. White, N. Bawa, N. Singhal, N. Egebo, N. Usunier, N. Mehta, N. P. Laptev, N. Dong, N. Cheng, O. Chernoguz, O. Hart, O. Salpekar, O. Kalinli, P. Kent, P. Parekh, P. Saab, P. Balaji, P. Rittner, P. Bontrager, P. Roux, P. Dollar, P. Zvyagina, P. Ratanchandani, P. Yuvraj, Q. Liang, R. Alao, R. Rodriguez, R. Ayub, R. Murthy, R. Nayani, R. Mitra, R. Parthasarathy, R. Li, R. Hogan, R. Battey, R. Wang, R. Howes, R. Rinott, S. Mehta, S. Siby, S. J. Bondu, S. Datta, S. Chugh, S. Hunt, S. Dhillon, S. Sidorov, S. Pan, S. Mahajan, S. Verma, S. Yamamoto, S. Ramaswamy, S. Lindsay, S. Lindsay, S. Feng, S. Lin, S. C. Zha, S. Patil, S. Shankar, S. Zhang, S. Zhang, S. Wang, S. Agarwal, S. Sajuyigbe, S. Chintala, S. Max, S. Chen, S. Kehoe, S. Satterfield, S. Govindaprasad, S. Gupta, S. Deng, S. Cho, S. Virk, S. Subramanian, S. Choudhury, S. Goldman, T. Remez, T. Glaser, T. Best, T. Koehler, T. Robinson, T. Li, T. Zhang, T. Matthews, T. Chou, T. Shaked, V. Vontimitta, V. Ajayi, V. Montanez, V. Mohan, V. S. Kumar, V. Mangla, V. Ionescu, V. Poenaru, V. T. Mihailescu, V. Ivanov, W. Li, W. Wang, W. Jiang, W. Bouaziz, W. Constable, X. Tang, X. Wu, X. Wang, X. Wu, X. Gao, Y. Kleinman, Y. Chen, Y. Hu, Y. Jia, Y. Qi, Y. Li, Y. Zhang, Y. Zhang, Y. Adi, Y. Nam, Yu, Wang, Y. Zhao, Y. Hao, Y. Qian, Y. Li, Y. He, Z. Rait, Z. DeVito, Z. Rosnbrick, Z. Wen, Z. Yang, Z. Zhao, and Z. Ma, "The llama 3 herd of models," 2024. [Online]. Available: https://arxiv.org/abs/2407.21783

[21] L. Ben Allal, A. Lozhkov, G. Penedo, T. Wolf, and L. von Werra, "Smollm-corpus," 2024. [Online]. Available: https://huggingface.co/datasets/HuggingFaceTB/smollm-corpus

[22] S. Black, S. Biderman, E. Hallahan, Q. Anthony, L. Gao, L. Golding, H. He, C. Leahy, K. Mc-Donell, J. Phang *et al.*, "Gpt-neox-20b: An open-source autoregressive language model," *arXiv preprint arXiv:2204.06745*, 2022.

[23] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. [Online]. Available: https://www.aclweb.org/anthology/2020.emnlp-demos.6

[24] L. Gao, J. Tow, S. Biderman, S. Black, A. DiPofi, C. Foster, L. Golding, J. Hsu, K. McDonell, N. Muennighoff, J. Phang, L. Reynolds, E. Tang, A. Thite, B. Wang, K. Wang, and A. Zou, "A framework for few-shot language model evaluation," Sep. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.5371628

[25] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt, "Measuring massive multitask language understanding," in *International Conference on Learning Representations*, 2021.

[26] M. Joshi, E. Choi, D. S. Weld, and L. Zettlemoyer, "Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension," 2017.

[27] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord, "Think you have solved question answering? try ARC, the AI2 reasoning challenge," *arXiv preprint arXiv:1803.05457*, 2018.

[28] Y. Bisk, R. Zellers, J. Gao, Y. Choi *et al.*, "PIQA: Reasoning about physical commonsense in natural language," in *Proceedings of the AAAI conference on Artificial Intelligence*, vol. 34, 2020.

[29] R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi, "HellaSwag: Can a machine really finish your sentence?" in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

[30] T. Mihaylov, P. Clark, T. Khot, and A. Sabharwal, "Can a suit of armor conduct electricity? a new dataset for open book question answering," *arXiv preprint arXiv:1809.02789*, 2018.

[31] K. Sakaguchi, R. L. Bras, C. Bhagavatula, and Y. Choi, "Winogrande: An adversarial Winograd schema challenge at scale," *Communications of the ACM*, vol. 64, no. 9, pp. 99–106, 2021.