MaBERT: A Padding-Safe Interleaved Transformer-Mamba Hybrid Encoder for Efficient Extended-Context Masked Language Modeling

Abstract

Encoder-only Transformers such as BERT learn strong bidirectional representations through masked language modeling (MLM), but their quadratic self-attention cost limits long-input scalability. Recent Transformer-Mamba hybrids improve efficiency by combining attention with state-space models (SSMs), yet they mainly target decoder-based causal language modeling, whereas encoder-only MLM must preserve bidirectional context and control padding propagation in variable-length batches. We propose MaBERT, an encoder-only hybrid model that interleaves Transformer-based global context mixing with Mamba-based sequential state updates and applies two-stage padding-safe masking to mitigate padding-induced representation drift. Under controlled pretraining and fine-tuning with representative encoder baselines, MaBERT achieves the highest macro-average score across the eight General Language Understanding Evaluation (GLUE) tasks and the best performance on seven of them, with a 14.8% improvement over the strongest baseline on the Corpus of Linguistic Acceptability (CoLA) task. On 11-class arXiv academic research topic classification with long-document inputs up to 4,096 tokens, MaBERT attains the highest macro-F1 score while reducing training time and inference latency by $4.23\times$ and $4.59\times$, respectively, compared with the average of the baselines. These results suggest that MaBERT provides a practical framework for efficient long-context language understanding by preserving strong bidirectional representations while substantially reducing training and inference costs. Code is available at `https://anonymous.4open.science/r/mabert-0DF3`.

1 Introduction

BERT (Devlin et al., 2019) and other Transformer-based encoder models learn bidirectional contextual representations through masked language modeling (MLM) and have become standard backbones for natural language understanding. Encoder-only pretrained models are effective for tasks requiring fine-grained sequence-level representations, including sentence classification, sentence-pair inference, and document understanding (Wang et al., 2018). They also incur lower training and inference costs than large decoder-only generative models, making them practical in resource-constrained settings (Warner et al., 2024).

However, self-attention computes all pairwise token interactions, resulting in $O(n^2)$ computational and memory complexity with respect to the input length n (Vaswani et al., 2017). This limits scalability for long documents or extended contexts. RoBERTa (Liu et al., 2019) and DeBERTa (He et al., 2021) improve encoder representations through better pretraining and attention mechanisms, while Longformer (Beltagy et al., 2020) and BigBird (Zaheer et al., 2020) reduce long-input cost through sparse attention. Nevertheless, these methods remain attention-based or restrict global interactions, making it difficult to jointly achieve representational capacity and computational efficiency for long sequences (Tay et al., 2022).

State space models (SSMs) (Gu et al., 2022) offer an alternative by processing long inputs in linear time through recurrent hidden-state updates. Mamba (Gu and Dao, 2024) further improves SSMs with a selective scan mechanism that performs input-dependent state updates. However, recent Transformer-Mamba hybrids (Lieber et al., 2024; Fu et al., 2023) have mainly targeted decoder-based causal language modeling, where left-to-right prediction naturally aligns with Mamba's sequential scan-based state updates. In contrast, encoder-only MLM must predict masked tokens by integrating both left and right contexts, making decoder-oriented hybrid designs difficult to directly transfer to BERT-style encoders (Devlin et al., 2019). Accordingly, encoder-side Mamba

integration requires a structure that preserves self-attention-based bidirectional token interactions while incorporating efficient sequential state updates.

A further challenge is padding in variable-length mini-batches. Transformer encoders can exclude padding tokens through attention masks (Vaswani et al., 2017), but Mamba performs sequential state updates in input order, allowing padding-position activations to enter the SSM computation (Gu and Dao, 2024). These activations can remain in residual and feed-forward pathways and propagate to upper layers, causing padding-induced representation drift in sequence-level representations as the padded length changes. This makes padding propagation control essential for stable encoder-only Mamba integration.

In this paper, we propose MaBERT, an encoder-only Transformer-Mamba hybrid model. MaBERT extends BERT-style encoders into an interleaved hybrid encoder composed of Transformer context blocks and Mamba sequential blocks. Transformer blocks provide direct global context mixing through self-attention, while Mamba blocks provide efficient sequential representation updates through a selective state-space mechanism. MaBERT further applies two-stage padding-safe masking at the input and output stages of each Mamba block to mitigate padding-induced representation drift.

Our contributions are summarized as follows:

- We introduce MaBERT, an encoder-only Transformer-Mamba hybrid for masked language modeling. MaBERT interleaves attention-based context blocks and Mamba-based sequential blocks within a BERT-style encoder, combining direct bidirectional context mixing with efficient sequential representation updates.
- We identify padding-induced representation drift as a practical issue in Mamba-containing encoder batches and propose two-stage padding-safe masking (PSM). The proposed masking controls pad activations at both the Mamba input and block output stages, reducing the propagation of padding information through Mamba-containing encoder layers.
- We conduct a controlled architecture-level analysis of hybrid encoder design under matched training conditions. The results show that interleaving pattern, Mamba directionality, and padding-safe masking substantially affect encoder-only language understanding, while MaBERT improves both downstream performance and long-input efficiency.

The rest of this paper is organized as follows. Section 2 reviews related work. Section 3 describes MaBERT and padding-safe masking. Section 4 presents the experimental results and analyses. Section 5 concludes the paper.

2 Related Work

2.1 Transformer Encoder Models

Since BERT (Devlin et al., 2019) demonstrated the effectiveness of bidirectional contextual representations through MLM, encoder-based pretrained models have advanced by improving representational capacity and training recipes. RoBERTa (Liu et al., 2019) improves BERT-style encoders without architectural changes by removing next sentence prediction (NSP) and refining pretraining settings such as dynamic masking and large-batch training. DeBERTa (He et al., 2021) further improves token interaction modeling through disentangled attention, which separates content and relative position representations.

However, these models remain fundamentally based on self-attention, whose computational and memory costs scale as $O(n^2)$ with the input length (Vaswani et al., 2017). Longformer (Beltagy et al., 2020) and BigBird (Zaheer et al., 2020) reduce long-input cost through sparse attention, while recent optimized encoder systems improve practical efficiency through long-context training recipes, position encoding choices such as rotary position embeddings (RoPE) (Su et al., 2024), kernel-level attention acceleration via FlashAttention (Dao et al., 2022), and padding-removal strategies (Warner et al., 2024). Although these methods improve the efficiency of encoder models, sparse-attention approaches reduce cost by restricting interaction patterns, while optimized Transformer encoders remain centered on self-attention. Thus, jointly improving long-input efficiency and preserving strong contextual representations remains structurally challenging (Tay et al., 2022, 2020).

2.2 State Space Models

SSMs have been studied as efficient alternatives for long-sequence modeling. Unlike self-attention,

SSMs propagate sequence information through hidden states and can process sequences with linear complexity in the input length (Gu et al., 2021). Early SSMs based on linear time-invariant (LTI) systems used input-independent state transitions, limiting their ability to selectively retain or discard context-dependent information. S4 (Gu et al., 2022) improved the stability and efficiency of SSMs through structured parameterization, and H3 (Fu et al., 2023) extended SSMs to language modeling by combining SSM operations with gating mechanisms.

Mamba (Gu and Dao, 2024) addresses the representational limitations of conventional SSMs through a selective scan mechanism for input-dependent state updates. By adjusting state transitions and discretization parameters according to the input, Mamba selectively preserves important information, suppresses irrelevant information, and models long-range dependencies with $O(n)$ complexity. However, Mamba-based language models have mainly been validated in decoder-based causal language modeling settings (Lieber et al., 2024; Wang et al., 2024). How to integrate and stabilize Mamba in an encoder-MLM setting for bidirectional contextual representation learning remains underexplored.

2.3 Hybrid Attention-SSM Models

Recent studies have explored hybrid architectures that combine the global context mixing of self-attention with the linear sequence modeling efficiency of SSMs. Jamba (Lieber et al., 2024) interleaves Transformer and Mamba blocks to reduce memory and computational cost while preserving attention-based global interactions. Hymba (Dong et al., 2024) introduces a hybrid-head structure that places attention heads and SSM heads in parallel within a layer, combining global context extraction and sequential state modeling. Nemotron-H (Blakeman et al., 2025) scales hybrid Mamba-Transformer architectures to large language models and improves accuracy-efficiency trade-offs.

However, most existing hybrid Attention-SSM models target decoder-based generative models with causal masking (Lieber et al., 2024; Dong et al., 2024). In contrast, encoder-MLM requires full-context integration for masked token prediction (Devlin et al., 2019) and must handle padding introduced by variable-length batching. Since SSMs scan sequences and update states sequentially, padding-position activations can create nonzero states that persist through residual and feed-forward pathways if they are not explicitly suppressed (Gu and Dao, 2024). Therefore, encoder-based hybrid Attention-SSM models require a design that jointly considers global context mixing, efficient sequential state updates, and padding propagation control.

3 MaBERT

This section describes MaBERT, an encoder-only Transformer-Mamba hybrid for MLM, as illustrated in Figure 1. MaBERT converts input tokens into token, position, and segment embeddings, processes them through an interleaved stack of Transformer context blocks and Mamba sequential blocks, and passes the final encoder output to the MLM head during pretraining or to a task-specific classification head during downstream evaluation. Transformer blocks provide direct global context mixing, while Mamba blocks provide efficient sequential state updates.

3.1 Interleaved Encoder Design

Part 1 of Figure 1 illustrates the MaBERT encoder. Let the input hidden states to the l-th layer be

$$H^l \in \mathbb{R}^{B\times T\times D}, \tag{1}$$

where B, T, and D denote the batch size, sequence length, and hidden dimension, respectively. The padding mask is denoted by $m \in \{0,1\}^{B\times T}$, where $m_{b,t} = 1$ indicates a valid token and $m_{b,t} = 0$ indicates a padding position.

MaBERT replaces the homogeneous self-attention stack of a standard Transformer encoder with an interleaved stack of Transformer context blocks and Mamba sequential blocks. Transformer blocks provide direct global context mixing through self-attention, while Mamba blocks update token representations through a selective state-space mechanism. Despite their different internal operations, both blocks preserve the $B \times T \times D$ hidden-state interface: the Mamba block expands the hidden representation for convolution and selective scanning, then projects it back to dimension D. This enables Transformer and Mamba blocks to be interleaved without changing the hidden-state format.

We use a 12-layer MaBERT encoder with the MMTMMTMMTMMT pattern, where M and T denote Mamba sequential and Transformer context blocks, respectively. This pattern inserts one

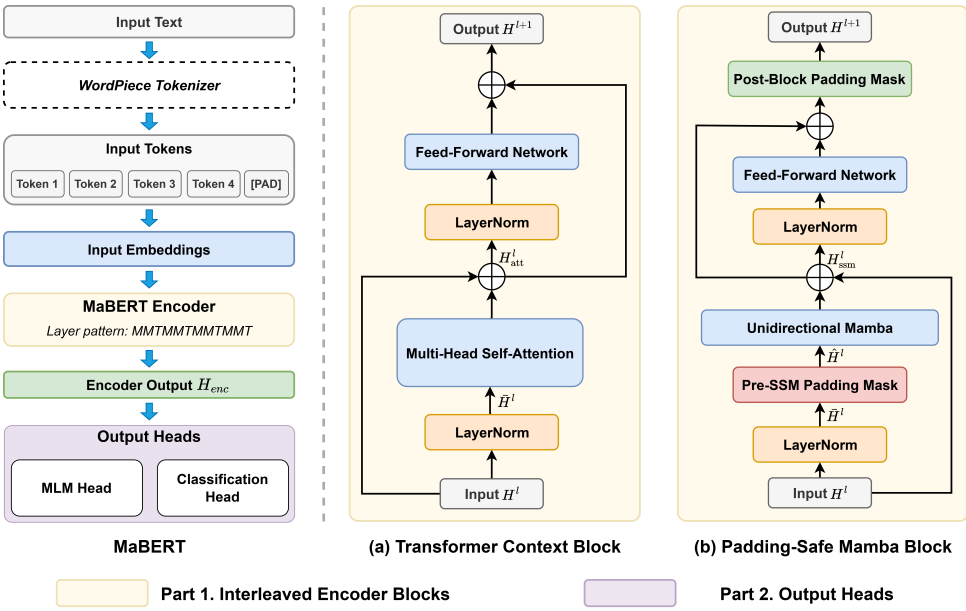


Figure 1: Overall architecture of MaBERT.

Transformer block after every two Mamba blocks and showed the best overall performance in the structural analysis of Section 4.3. Each Mamba block performs unidirectional sequential updates, while the periodically inserted Transformer blocks provide explicit bidirectional context mixing.

MaBERT applies a Pre-LayerNorm residual update rule to all blocks for consistent integration of attention-based and state-space-based transformations. Each block normalizes its input before the core operation and adds the transformed output to the original representation through a residual connection. Sections 3.2 and 3.3 detail the Transformer context block and the padding-safe Mamba sequential block, respectively.

3.2 Transformer Context Block

Figure 1(a) illustrates the computational flow of the Transformer context block. The Transformer context block computes direct contextual interactions across all tokens through self-attention. Given the input H^l to the l-th Transformer block, the input representation is first normalized by LayerNorm. Multi-head self-attention is then performed with the padding mask m:

$$\bar{H}^l = \mathrm{LN}(H^l), \tag{2}$$

$$H^l_{\mathrm{att}} = H^l + \mathrm{MHSA}(\bar{H}^l; m). \tag{3}$$

The attention residual output H^l_{att} is then processed by another LayerNorm and a feed-forward network:

$$H^{l+1} = H^l_{\mathrm{att}} + \mathrm{FFN}(\mathrm{LN}(H^l_{\mathrm{att}})), \tag{4}$$

where $\mathrm{LN}(\cdot)$, $\mathrm{MHSA}(\cdot)$, and $\mathrm{FFN}(\cdot)$ denote LayerNorm, multi-head self-attention, and a position-wise feed-forward network, respectively. In MaBERT, the Transformer block provides direct self-attention-based cross-token interaction and complements global context information between the sequential updates of Mamba blocks.

3.3 Padding-Safe Mamba Sequential Block

Figure 1(b) illustrates the computational flow of the padding-safe Mamba sequential block. The Mamba block updates hidden states along the sequence direction through a selective state-space mechanism. Unlike self-attention, which directly computes all pairwise token interactions, Mamba updates token representations through a sequential state scan.

When Mamba is used in an encoder setting, handling padding in variable-length mini-batches becomes important. In Transformer blocks, padding positions can be excluded through attention masks. In contrast, because Mamba blocks perform sequential state updates, padding positions can produce nonzero hidden states and keep propagating through residual and feed-forward pathways. To mitigate this issue, MaBERT applies two-stage padding-safe masking to each Mamba block.

The padding mask m is broadcast along the hidden dimension and used as $\tilde{m} \in \{0, 1\}^{B \times T \times 1}$. The

input H^l is first normalized by LayerNorm, and activations at padding positions are removed before being fed into the Mamba SSM core:

$$\bar{H}^l = \mathrm{LN}(H^l), \quad \hat{H}^l = \tilde{m} \odot \bar{H}^l, \tag{5}$$

$$H^l_{\mathrm{ssm}} = H^l + \tilde{m} \odot \mathrm{Mamba}(\hat{H}^l), \tag{6}$$

where $\mathrm{Mamba}(\cdot)$ denotes a shape-preserving transformation that expands the hidden representation for convolution and selective scanning and projects it back to dimension D, yielding $\mathrm{Mamba}(\hat{H}^l) \in \mathbb{R}^{B\times T\times D}$. The symbol $\odot$ denotes element-wise multiplication. This dimensional consistency enables the residual addition in Eq. 6. This operation removes padding-position activations before the SSM core and zeroes the Mamba output at padding positions before the residual update. A Pre-LayerNorm FFN update is then applied to H^l_{ssm}, and the same padding mask is applied to the final block output:

$$H^{l+1} = \tilde{m} \odot \left(H^l_{\mathrm{ssm}} + \mathrm{FFN}(\mathrm{LN}(H^l_{\mathrm{ssm}})) \right). \tag{7}$$

Rather than resetting the internal SSM state at padding boundaries, MaBERT controls padding effects by removing padding-position activations before the Mamba core and by masking the block output before it is propagated to upper layers. Thus, although the selective scan may still process zero-valued padded inputs, padding-position representations are prevented from contributing to residual, FFN, and subsequent-layer computations. Therefore, two-stage padding-safe masking suppresses both the injection of padding activations into the SSM core and their layer-wise propagation, thereby mitigating padding-induced representation drift in variable-length encoder batches. The drift analysis in Section 4.5 empirically verifies that this masking strategy substantially reduces padding-induced representation drift.

3.4 Output Heads

Part 2 of Figure 1 illustrates the output heads attached to the shared MaBERT encoder backbone. Following the standard encoder-only pretraining and fine-tuning paradigm, MaBERT uses an MLM head during pretraining and a lightweight classification head during downstream evaluation. The final encoder output is denoted as follows:

$$H_{\mathrm{enc}} \in \mathbb{R}^{B\times T\times D}. \tag{8}$$

The MLM head maps token-level contextual representations to vocabulary logits and is trained to predict the original tokens only at masked positions.

For downstream classification, the contextual representation of the [CLS] token is used as the sequence-level representation:

$$h_{\mathrm{cls}} = H_{\mathrm{enc},[\mathrm{CLS}]}. \tag{9}$$

The representation is then passed through dropout and a linear classifier to produce class logits:

$$z = \mathrm{Dropout}(h_{\mathrm{cls}})W_c + b_c, \tag{10}$$

where $W_c \in \mathbb{R}^{D\times C}$ and $b_c \in \mathbb{R}^C$ are classifier parameters, and C denotes the number of classes. The output $z \in \mathbb{R}^{B\times C}$ denotes the final class logits. This standard head design keeps the empirical comparison focused on the shared encoder architecture rather than task-specific prediction layers.

4 Experiments

The experiments are organized around four aspects: general language understanding, hybrid encoder structure, long-input scalability, and padding-safe masking. Specifically, we compare MaBERT with representative encoder baselines on the General Language Understanding Evaluation (GLUE) benchmark, analyze interleaving patterns and Mamba directionality, evaluate arXiv long-document topic classification across 512–4096-token inputs, and examine padding-induced representation drift.

4.1 Experimental Setup

Baselines. We compare MaBERT with BERT (Devlin et al., 2019), Longformer (Beltagy et al., 2020), BigBird (Zaheer et al., 2020), and DeBERTa (He et al., 2021), covering dense self-attention, sparse long-context attention, and disentangled-attention encoders. All models are implemented with HuggingFace Transformers (Wolf et al., 2020) and trained from scratch under the same MLM pretraining and downstream fine-tuning protocol. Each baseline uses its default tokenizer and input preprocessing pipeline to preserve its standard interface. This setup isolates architectural differences under matched training conditions rather than ranking publicly released checkpoints, which may differ in corpora, objectives, optimization recipes, tokenizer design, and system-level implementations.

Datasets and metrics. General language understanding is evaluated on eight GLUE tasks: Corpus of Linguistic Acceptability (CoLA), SST-2, MRPC, QQP, MNLI-m, MNLI-mm, QNLI, and RTE (Wang et al., 2018). We report Matthews correlation coefficient for CoLA, F1-score for MRPC and QQP, and accuracy for the remaining tasks, and compute the macro-average over the eight tasks. Long-input performance is evaluated on arXiv topic classification (He et al., 2019), an 11-class task that predicts the topic category of academic papers, at maximum input lengths of 512, 1024, 2048, and 4096 tokens. We use macro-F1 as the primary metric and accuracy as a secondary metric.

Training and evaluation. All models are pretrained on BookCorpus and English Wikipedia using only the MLM objective (Devlin et al., 2019), with 900K steps at length 128 and 100K steps at length 512. For long-input experiments, position embeddings are extended to 4096 tokens when required, and all models receive the same 10K continued MLM steps at length 4096. For learned absolute position embeddings, we use repeated-copy extension to avoid model-specific length-extension tuning. Downstream results are reported over five random seeds using the shared [CLS]-based classification head in Section 3.4. Efficiency is measured under matched hardware, input length, batch size, precision, warm-up, and GPU synchronization settings. Full details are provided in Appendix A, Tables A1– A4.

4.2 Main Results

This section compares the GLUE benchmark performance of the final MaBERT encoder against baseline encoder models. All models are evaluated under the same pretraining and fine-tuning protocol described in Section 4.1, and the results are presented in Table 1.

As shown in Table 1, MaBERT achieves the highest GLUE average among all models. Compared with the strongest baseline, DeBERTa, MaBERT improves the macro-average score by 1.86% and shows relative improvements of 4.45%, 3.53%, and 3.66% over BERT, Longformer, and BigBird, respectively. This shows that MaBERT provides consistent gains across dense self-attention encoders, sparse-attention-based long-context encoders, and disentangled-attention encoders under matched pretraining and fine-tuning conditions. Although MaBERT has more parameters than BERT, it is slightly smaller than DeBERTa while achieving a higher GLUE macro-average, suggesting that improvement is not solely explained by parameter count.

At the task level, MaBERT achieves the best performance on seven out of the eight GLUE tasks. The largest gain is observed on CoLA, where MaBERT improves the MCC score over DeBERTa by 14.8%, while it also improves RTE accuracy by 3.65%. Although DeBERTa achieves the highest F1-score on MRPC, MaBERT obtains the best overall average and the strongest results on most task-level metrics. These results suggest that the Transformer-Mamba hybrid encoder is effective not only for general encoder-only language understanding, but also for fine-grained sentence-level judgment tasks such as grammatical acceptability.

4.3 Structural Analysis

This section analyzes two design factors that determine the MaBERT encoder configuration: Transformer-Mamba interleaving patterns and Mamba scan direction. All comparisons use the same 12-layer depth, MLM pretraining corpus, pretraining budget, and downstream fine-tuning protocol. Additional details on layer composition, parameter counts, Mamba configuration, and scan-direction implementation are provided in Appendix B, Tables B1 and B2.

We first compare Mamba-only, Transformer-only, and hybrid interleaving patterns, where all hybrid patterns use unidirectional Mamba scans. As shown in Table 2, the Mamba-only encoder performs worst, while most hybrid patterns outperform the Transformer-only encoder. This indicates that simply replacing attention with Mamba is insufficient for encoder-only language understanding, and that Mamba is more effective when integrated into an attention-supported encoder stack. Among the hybrid patterns, MMTMMTMMTMMT achieves the highest GLUE average of 0.822 and the best results on all eight tasks, showing that layer ordering matters even with the same number of Mamba and Transformer layers.

We next analyze the Mamba scan direction. Bidirectional scanning can compensate for the lack of attention-based bidirectional context mixing in a pure Mamba stack, whereas its role may differ in hybrid encoders where Transformer layers already provide bidirectional context. Therefore, we compare unidirectional and bidirectional Mamba in the pure Mamba stack and representative hybrid pat-

Model	CoLA	SST-2	MRPC	QQP	MNLI-m	MNLI-mm	QNLI	RTE	Avg.
BERT	0.522±0.017	0.912±0.012	0.853±0.017	0.856±0.006	0.826±0.014	0.829±0.011	0.876±0.012	0.618±0.027	0.787
Longformer	0.534±0.018	0.924±0.012	0.863±0.016	0.858±0.006	0.830±0.013	0.831±0.014	0.882±0.012	0.626±0.031	0.794
BigBird	0.528±0.016	0.926±0.011	0.864±0.015	0.857±0.005	0.831±0.014	0.832±0.012	0.881±0.013	0.624±0.029	0.793
DeBERTa	0.589±0.015	0.923±0.013	**0.895±0.014**	0.872±0.004	0.828±0.015	0.831±0.013	0.887±0.019	0.631±0.034	0.807
MaBERT	**0.676±0.018**	**0.933±0.010**	0.869±0.017	**0.879±0.005**	**0.835±0.016**	**0.837±0.017**	**0.893±0.012**	**0.654±0.033**	**0.822**

Table 1: GLUE performance of MaBERT and baseline encoder models. Avg. denotes the macro-average over the eight tasks. Boldface indicates the best performance in each column.

Pattern	CoLA	SST-2	MRPC	QQP	MNLI-m	MNLI-mm	QNLI	RTE	Avg.
MMMMMMMMMMMM	0.438±0.020	0.889±0.011	0.812±0.019	0.840±0.005	0.760±0.017	0.755±0.018	0.812±0.013	0.570±0.034	0.735
TTTTTTTTTTTT	0.522±0.017	0.912±0.012	0.853±0.017	0.856±0.006	0.826±0.014	0.829±0.011	0.876±0.012	0.618±0.027	0.787
MTTMTTMTTMTT	0.653±0.018	0.929±0.010	0.858±0.016	0.873±0.005	0.830±0.015	0.831±0.017	0.887±0.011	0.640±0.032	0.813
TMTMTMTMTMTM	0.620±0.017	0.920±0.011	0.860±0.015	0.875±0.004	0.828±0.014	0.829±0.016	0.889±0.010	0.642±0.031	0.808
MTMTMTMTMTMT	0.662±0.019	0.922±0.009	0.846±0.018	0.874±0.005	0.829±0.016	0.832±0.015	0.884±0.012	0.638±0.033	0.811
TMMTMMTMMTMM	0.640±0.018	0.926±0.010	0.864±0.016	0.877±0.005	0.831±0.015	0.832±0.017	0.890±0.011	0.648±0.032	0.814
MMTMMTMMTMMT	**0.676±0.018**	**0.933±0.010**	**0.869±0.017**	**0.879±0.005**	**0.835±0.016**	**0.837±0.017**	**0.893±0.012**	**0.654±0.033**	**0.822**
TTMTTMTTMTTM	0.612±0.016	0.924±0.010	0.851±0.017	0.871±0.004	0.826±0.015	0.830±0.016	0.891±0.011	0.636±0.030	0.805

Table 2: GLUE performance across Transformer-Mamba interleaving patterns. M and T denote Mamba and Transformer layers, respectively. Avg. denotes the macro-average over the eight tasks.

terns, as shown in Table 3.

Pattern	Direction	Avg.
MMMMMMMMMMMM	Uni	0.735
	Bi	**0.803**
TMMTMMTMMTMM	**Uni**	**0.814**
	Bi	0.800
MMTMMTMMTMMT	**Uni**	**0.822**
	Bi	0.786

Table 3: Effect of Mamba directionality in pure and hybrid encoder stacks. Avg. denotes the GLUE macro-average score. Uni and Bi denote unidirectional and bidirectional Mamba scans, respectively.

Bidirectional scanning improves the pure Mamba stack by 9.25%, confirming its usefulness when attention-based bidirectional context mixing is absent. In contrast, unidirectional scans outperform bidirectional scans by 3.15% on average in hybrid stacks, suggesting that Mamba is more effective as a complementary sequential module once Transformer layers provide bidirectional context mixing. Accordingly, the final MaBERT encoder uses unidirectional Mamba scans.

4.4 Scalability Analysis

This section evaluates MaBERT on 11-class arXiv academic research topic classification under increasing input lengths. The dataset consists of long academic documents, with 95.97% of examples exceeding 4,096 tokens after tokenization, as reported in Appendix C, Table C1. Since 4,096 tokens substantially exceeds the conventional 512-token input limit of BERT-style encoders, this setting evaluates long-context document classification beyond the standard encoder length. We evaluate maximum input lengths of 512, 1,024, 2,048, and 4,096 tokens, using macro-F1 as the primary metric and accuracy as a secondary metric. Accuracy results are provided in Appendix C, Table C2.

Table 4 compares macro-F1 scores across the four maximum input lengths.

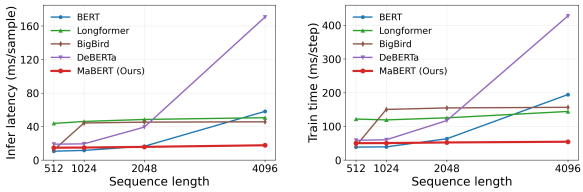
Model	F1@512	F1@1024	F1@2048	F1@4096
BERT	0.847±0.002	0.852±0.002	0.856±0.002	0.857±0.002
Longformer	0.847±0.002	0.853±0.003	0.856±0.003	0.859±0.003
BigBird	0.840±0.003	0.846±0.003	0.850±0.003	0.853±0.003
DeBERTa	**0.854±0.001**	0.855±0.002	0.857±0.002	0.858±0.002
MaBERT	0.852±0.002	**0.856±0.002**	**0.863±0.002**	**0.868±0.002**

Table 4: Long-document classification performance on arXiv topic classification. Macro-F1 is reported across maximum input lengths, where F1@L denotes macro-F1 evaluated with a maximum input length of L tokens.

MaBERT is slightly behind DeBERTa at 512 tokens but achieves the highest macro-F1 from 1,024 tokens onward. At 4,096 tokens, MaBERT improves over the strongest baseline by 1.05%, and its own macro-F1 increases by 1.88% from the 512-token setting. This indicates that MaBERT can effectively leverage additional context for long-document classification.

Figure 2 shows the efficiency trend as input length increases. Under the timing protocol detailed in Appendix C and Table C3, MaBERT shows only modest increases in inference latency and train step time from 512 to 4,096 tokens, whereas BERT and DeBERTa show rapidly increasing costs. At 4,096 tokens, MaBERT is approximately 3.3× and 9.6× faster than BERT and DeBERTa in inference latency, and approximately

3.6× and 7.8× faster in train step time, respectively. This indicates favorable runtime scaling for MaBERT at longer input lengths.



(a) (b)

Figure 2: Efficiency comparison across input lengths. (a) Inference latency. (b) Train step time.

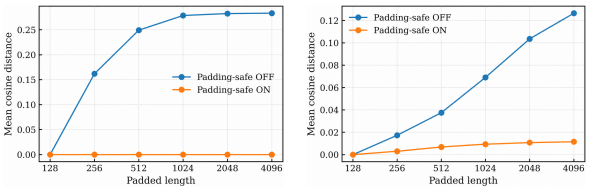
Details of the system environment, full timing results, and peak-memory measurements are provided in Appendix C, Tables C3– C5.

4.5 Masking Analysis

This section analyzes padding-safe masking in terms of downstream performance, representation stability, and masking placement. We first ablate padding-safe masking while keeping the classification head unchanged. Removing padding-safe masking leads to a 2.68% relative decrease in GLUE average, indicating that it contributes to both representation stability and downstream performance. Detailed task-level results are reported in Appendix D, Table D1.

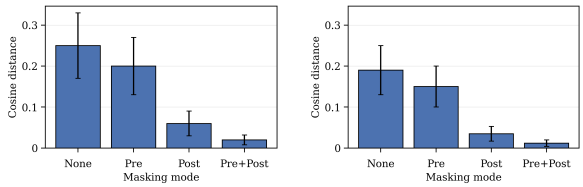
We then evaluate representation stability by increasing the padded sequence length from 128 to 4096 while keeping valid tokens fixed, as shown in Figure 3. Drift is measured on the [CLS] representation used for classification and on the valid-token mean representation. Without padding-safe masking, the mean cosine distance from the reference representation increases as padding length grows, especially for [CLS]. In contrast, padding-safe masking strongly suppresses drift for both representation types. The full protocol is summarized in Appendix D, Table D2.

Figure 4 further decomposes the effect of masking placement. Pre-masking controls padding activations before the Mamba SSM core, whereas post-masking blocks padding-position representations from propagating to the next layer. Post-masking is more effective than pre-masking alone, and combining both yields the lowest cosine distance, resulting in the most stable representations. This confirms the importance of blocking layer-wise padding propagation in two-stage padding-safe masking. Detailed quantitative results are reported in Appendix D, Table D3.



(a) (b)

Figure 3: Padding-induced representation drift averaged over the GLUE dev sets. (a) [CLS] representation. (b) Valid-token mean representation. Lower cosine distance indicates lower padding-induced representation drift.



(a) (b)

Figure 4: Masking placement ablation measured by padding-induced representation drift. (a) [CLS] representation. (b) Valid-token mean representation.

5 Conclusion

This paper presents MaBERT, a Transformer-Mamba hybrid encoder for encoder-only masked language modeling. MaBERT interleaves Transformer context blocks and Mamba sequential blocks to combine bidirectional context mixing with efficient sequential state updates, and applies two-stage padding-safe masking to control padding activations in variable-length encoder batches.

MaBERT improves GLUE performance and long-document classification efficiency over encoder baselines. The results also show that periodically inserting one Transformer layer after every two Mamba layers forms an effective interleaving pattern. Unidirectional Mamba scans are particularly effective when Transformer layers already provide bidirectional context mixing. In addition, two-stage padding-safe masking mitigates padding-induced representation drift during Transformer-Mamba hybrid encoding. MaBERT provides a practical framework for extending encoder-only masked language models to long-context understanding while preserving contextual representation quality and improving computational efficiency.

6 Limitations

This study compares models under the same MLM pretraining budget and downstream fine-tuning protocol to isolate the architectural effects of MaBERT. This controlled setting should be interpreted as architecture-level evidence under matched training conditions, rather than as final performance rankings among publicly released checkpoints that may differ in corpora, objectives, optimization recipes, tokenizer design, and system-level implementations. In addition, each baseline uses its default tokenizer and input preprocessing pipeline; while this preserves each architecture's standard interface, tokenizer differences may partially affect absolute downstream performance.

For long-context evaluation, learned position embeddings are extended to 4,096 tokens when required, and the same 10K-step continued MLM adaptation is applied to all models under matched conditions. Although this protocol avoids model-specific tuning, stronger position-extension or adaptation strategies may further improve individual baselines. The long-document analysis is mainly conducted on arXiv topic classification, so the results should be viewed as evidence from long-document topic classification rather than as a comprehensive evaluation of all long-context natural language understanding scenarios. Future work should extend the evaluation to more diverse long-document benchmarks, including legal, biomedical, scientific, and multi-document understanding tasks.

Finally, this study focuses on encoder-only pretraining and sequence-level classification with a unified [CLS]-based classifier. Future work should further examine MaBERT and padding-safe masking in broader encoder-based settings, such as retrieval, reranking, token-level prediction, and architecture-specific classification heads.

References

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

Aaron Blakeman, Aarti Basant, Abhinav Khattar, Adithya Renduchintala, Akhiad Bercovich, Aleksander Ficek, Alexis Bjorlin, Ali Taghibakhshi, Amala Sanjay Deshmukh, Ameya Sunil Mahabaleshwarkar, and 1 others. 2025. Nemotron-h: A family of accurate and efficient hybrid mamba-transformer models. arXiv preprint arXiv:2504.03624.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. In *Advances in Neural Information Processing Systems*, volume 35, pages 16344–16359.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186.

Xin Dong, Yonggan Fu, Shizhe Diao, Wonmin Byeon, Zijia Chen, Ameya Sunil Mahabaleshwarkar, Shih-Yang Liu, Matthijs Van Keirsbilck, Min-Hung Chen, Yoshi Suhara, Yingyan Lin, Jan Kautz, and Pavlo Molchanov. 2024. Hymba: A hybrid-head architecture for small language models. arXiv preprint arXiv:2411.13676.

Daniel Y. Fu, Tri Dao, Khaled K. Saab, Armin W. Thomas, Atri Rudra, and Christopher Ré. 2023. Hungry hungry hippos: Towards language modeling with state space models. In *International Conference on Learning Representations*.

Albert Gu and Tri Dao. 2024. Mamba: Linear-time sequence modeling with selective state spaces. In *Conference on Language Modeling*.

Albert Gu, Karan Goel, and Christopher Ré. 2022. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*.

Albert Gu, Isys Johnson, Karan Goel, Khaled Saab, Tri Dao, Atri Rudra, and Christopher Ré. 2021. Combining recurrent, convolutional, and continuous-time models with linear state space layers. In *Advances in Neural Information Processing Systems*, volume 34, pages 572–585.

Jun He, Liqun Wang, Liu Liu, Jiao Feng, and Hao Wu. 2019. Long document classification from local word glimpses via recurrent attention learning. *IEEE Access*, 7:40707–40718.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: Decoding-enhanced BERT with disentangled attention. In *International Conference on Learning Representations*.

Opher Lieber, Barak Lenz, Hofit Bata, Gal Cohen, Jhonathan Osin, Itay Dalmedigos, Erez Safahi, Shaked Meirom, Yonatan Belinkov, Shai Shalev-Shwartz, Omri Abend, Raz Alon, Tomer Asida, Amir Bergman, Roman Glozman, Michael Gokhman, Avashalom Manevich, Nir Ratner, Noam Rozen, and 3 others. 2024. Jamba: A hybrid transformer-mamba language model. arXiv preprint arXiv:2403.19887.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. 2020. Long range arena: A benchmark for efficient transformers. arXiv preprint arXiv:2011.04006.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2022. Efficient transformers: A survey. *ACM Computing Surveys*, 55(6):1–28.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355.

Junxiong Wang, Tushaar Gangavarapu, Jing Nathan Yan, and Alexander M. Rush. 2024. Mambabyte: Token-free selective state space model. arXiv preprint arXiv:2401.13660.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Nathan Cooper, Griffin Adams, Jeremy Howard, and Iacopo Poli. 2024. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. arXiv preprint arXiv:2412.13663.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, and Li Yang. 2020. Big bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems*, volume 33, pages 17283–17297.

A Experimental Details

This appendix summarizes the controlled pretraining and fine-tuning settings used in Section 4.1. All models use the same corpus, objective, training budget, fine-tuning pipeline, metrics, and five random seeds.

Setting	Value
Corpus	BookCorpus + English Wikipedia
Objective	MLM only, no NSP
Masking ratio	0.15
Mask replacement	80% [MASK], 10% random, 10% unchanged
Base pretraining steps	1M
Base length schedule	900K at 128, 100K at 512
Long-context adaptation	10K MLM steps at 4096
Global batch size	256
Optimizer	AdamW
β_1, β_2	0.9, 0.98
ϵ	1e-6
Peak learning rate	6e-4
Warmup steps	24K
Learning rate schedule	Linear decay
Weight decay	0.01
Dropout	0.1

Table A1: Common MLM pretraining settings used for all models.

Setting	GLUE	arXiv topic classification
Max input length	512	512, 1024, 2048, 4096
Optimizer	AdamW	AdamW
Learning rate	2e-5	2e-5
Per-device batch size	16	2
Gradient accumulation	1	8
Effective batch size	16	16
Epochs	3	5
Warmup ratio	0.10	0.06
Weight decay	0.01	0.01
Early stopping	Not used	Patience = 3
Seeds	5	5
Evaluation metric	Task-specific	Macro-F1, accuracy
Checkpoint selection	Best validation metric	Best validation macro-F1

Table A2: Downstream fine-tuning settings.

Model	Context mechanism	Params.
BERT	Full self-attention	109.5M
Longformer	Sliding-window sparse attention	148.7M
BigBird	Block, sparse, and global attention	127.5M
DeBERTa	Disentangled self-attention	183.8M
MaBERT	Interleaved Transformer-Mamba blocks	175.4M

Table A3: Model scale and context mechanism of compared encoders.

Model	Original	Target	Extension method
BERT	512	4096	Repeated copy
Longformer	4098	4096	Native long-position embeddings
BigBird	4096	4096	Native long-position embeddings
DeBERTa	512	4096	Repeated copy when required
MaBERT	512	4096	Repeated copy

Table A4: Position embedding extension for long-context evaluation.

B Additional Structural Analysis

This appendix reports additional details for the interleaving and directionality analyses in Section 4.3. All variants use the same 12-layer depth and training protocol.

Encoder pattern	#M	#T	Params.
MMMMMMMMMMMM	12	0	208.7M
TTTTTTTTTTTT	0	12	108.9M
MTTMTTMTTMTT	4	8	142.2M
TMTMTMTMTMTM	6	6	158.8M
MTMTMTMTMTMT	6	6	158.8M
TMMTMMTMMTMM	8	4	175.4M
MMTMMTMMTMMT	8	4	175.4M
TTMTTMTTMTTM	4	8	142.2M

Table B1: Layer composition and parameter counts of interleaving patterns. M and T denote Mamba and Transformer layers, respectively.

Setting	Value
Mamba core	Mamba2
Hidden size	768
State size (d_{state})	16
Convolution width (d_{conv})	4
Expansion factor	2
Dropout	0.1

Table B2: Mamba block configuration used in MaBERT.

Unidirectional Mamba uses a single left-to-right scan. Bidirectional Mamba applies forward and backward scans and averages the two scan outputs before the residual update.

C Additional Scalability Results

This appendix provides dataset statistics, accuracy results, and full efficiency measurements for Section 4.4.

Statistic	Value
Train examples	28,388
Validation examples	2,500
Test examples	2,500
Number of classes	11
Average token length	16,200.96
Median token length	12,832
90th percentile token length	29,096
Documents over 512 tokens	100.00%
Documents over 1024 tokens	99.97%
Documents over 2048 tokens	99.64%
Documents over 4096 tokens	95.97%

Table C1: Statistics of the arXiv topic classification dataset.

Model	Acc@512	Acc@1024	Acc@2048	Acc@4096
BERT	0.8512±0.002	0.8542±0.002	0.8584±0.002	0.8598±0.002
Longformer	0.8524±0.002	0.8580±0.003	0.8616±0.003	0.8640±0.003
BigBird	0.8468±0.003	0.8528±0.003	0.8568±0.003	0.8596±0.003
DeBERTa	0.8534±0.001	0.8566±0.002	0.8608±0.002	0.8618±0.002
MaBERT	**0.8542±0.002**	**0.8588±0.002**	**0.8652±0.002**	**0.8712±0.002**

Table C2: Accuracy on arXiv topic classification across maximum input lengths.

Component	Specification
GPU	NVIDIA RTX PRO 6000, 96GB
Software stack	PyTorch 2.10.0+cu128, Transformers 4.48.3, mamba-ssm 2.3.0
Precision	bf16 mixed precision
Timing protocol	Warm-up iterations excluded from synchronized CUDA timing

Table C3: System environment and timing protocol for efficiency measurements.

Timing measurements are conducted using fixed maximum-length inputs for each setting, with caching disabled. Warm-up iterations are performed but excluded from the reported averages. Reported times are averaged over repeated synchronized CUDA runs, and no hidden truncation is applied beyond the configured maximum input length.

Model	Train@512	Train@1024	Train@2048	Train@4096	Mem@4096
BERT	**38.6**	**39.3**	63.3	194.4	20.3
Longformer	121.8	119.3	125.4	144.3	6.4
BigBird	42.4	150.3	154.8	156.6	8.0
DeBERTa	58.7	60.3	117.2	427.5	25.9
MaBERT	50.3	50.5	**52.4**	**54.5**	**3.9**

Table C4: Training step time and peak GPU memory. Training time is reported in milliseconds, and peak memory is reported in GB at 4096 tokens.

Model	Infer@512	Infer@1024	Infer@2048	Infer@4096
BERT	**10.6**	**11.6**	16.5	58.1
Longformer	43.8	46.1	48.5	50.6
BigBird	12.1	44.3	45.3	45.7
DeBERTa	19.0	19.4	39.4	170.6
MaBERT	14.8	15.0	**15.8**	**17.7**

Table C5: Inference latency across input lengths. Latency is reported in milliseconds and indicates forward-only evaluation.

D Additional Masking Analysis

This appendix reports task-level GLUE results without padding-safe masking, the padding drift protocol, and masking placement ablation for Section 4.5.

Task	MaBERT	w/o PSM
CoLA	**0.676±0.018**	0.661±0.018
SST-2	**0.933±0.010**	0.922±0.031
MRPC	**0.869±0.017**	0.841±0.023
QQP	**0.879±0.005**	0.860±0.004
MNLI-m	**0.835±0.016**	0.819±0.017
MNLI-mm	**0.837±0.017**	0.820±0.027
QNLI	**0.893±0.012**	0.878±0.014
RTE	**0.654±0.033**	0.597±0.078
Avg.	**0.822**	0.800

Table D1: Task-level GLUE results of MaBERT and its variant without padding-safe masking (w/o PSM). Avg. denotes the macro-average over the eight tasks.

Setting	Value
Evaluation split	GLUE development sets
Padding construction	Append padding after valid tokens
Target padded lengths	128, 512, 1024, 2048, 4096
Reference representation	Shortest padded length per example
Distance metric	Mean cosine distance
Representation types	[CLS]; valid-token mean
Aggregation	Averaged over examples and tasks

Table D2: Padding drift measurement protocol.

Masking mode	[CLS] rep.	Valid-token mean rep.
None	0.250±0.080	0.190±0.060
Pre	0.200±0.066	0.151±0.050
Post	0.061±0.030	0.037±0.019
Pre + Post	**0.022±0.012**	**0.013±0.008**

Table D3: Masking placement ablation measured by mean cosine distance. Lower values indicate lower padding-induced representation drift.