Bi-Mamba: Towards Accurate 1-Bit State Space Models

Anonymous authors
Paper under double-blind review

Abstract

The typical selective state-space model (SSM) of Mamba addresses several limitations of Transformers, such as quadratic computational complexity with sequence length and significant inference-time memory requirements due to the key-value cache. However, the growing size of Mamba models continues to pose training and deployment challenges and raises environmental concerns due to considerable energy consumption. In this work, we introduce `Bi-Mamba`, a scalable and powerful 1-bit Mamba architecture designed for more efficient large language models with multiple sizes across 780M, 1.3B, and 2.7B. `Bi-Mamba` models are trained from scratch on data volume as regular LLM pertaining using an autoregressive distillation loss. Extensive experimental results on language modeling demonstrate that `Bi-Mamba` achieves performance comparable to its full-precision counterparts (e.g., FP16 or BF16) and much better accuracy than post-training-binarization (PTB) Mamba and binarization-aware training (BAT) Transformer baselines, while significantly reducing memory footprint and energy consumption compared to the original Mamba model. Our study pioneers a new linear computational complexity LLM framework under low-bit representation and facilitates future design of specialized hardware tailored for efficient 1-bit Mamba-based LLMs. We will fully release our code and pre-trained weights for reproduction.

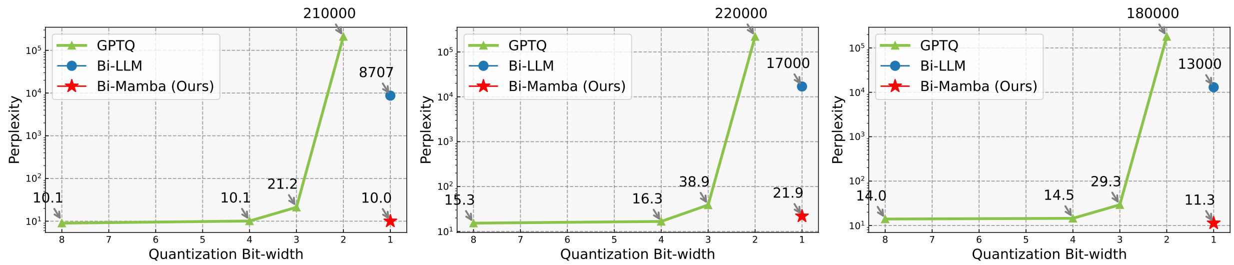
1 Introduction

The Selective State-Space Model (SSM) (Gu et al., 2021b; 2022) has recently emerged as a powerful alternative to Transformers Vaswani et al. (2017) in language modeling, demonstrating comparable or superior performance at small to moderate scales. SSMs are characterized by linear scaling in sequence length during training and a constant state size during generation, which significantly reduces computational and memory overhead, making them more efficient in terms of speed and memory usage.

The representative SSM model of Mamba (Gu & Dao, 2024; Dao & Gu, 2024) has a significant advantage when handling long context sequences due to its linear complexity. In contrast, conventional transformers suffer from quadratic complexity as the sequence length increases. This means that for tasks involving large input sequences or extended contexts, Mamba is much more efficient, as its memory and computational requirements scale linearly with the sequence length. In practice, this allows Mamba to process long sequences faster and with lower resource consumption, making it ideal for applications such as long document processing, conversational agents, and any scenario where managing large amounts of contextual information is crucial. On the other hand, transformers require exponentially more resources as the context length increases, which can quickly become a bottleneck in long-context tasks.

Prior works (Devlin et al., 2019; Radford et al., 2019; Touvron et al., 2023; Achiam et al., 2023) on Transformers have been extensively studied for several years, with numerous methods proposed to alleviate their high computational and storage costs, such as pruning (Ma et al., 2023), model quantization Frantar et al. (2023), and KV Cache (Pope et al., 2023; Kwon et al., 2023). Among these methods, quantization has proven to be highly effective Dettmers et al. (2022b). Researchers have successfully quantized transformer models from 16-bit to 8-bit, 4-bit, and even 1-bit representations, often with minimal performance degradation Frantar et al. (2023); Ma et al. (2024c;a).

However, there has been little investigation into how SSM models or Mamba behave after low-bit quantization or even binarization. In this paper, we introduce `Bi-Mamba`, a novel approach that applies extreme



(a) Wiki-2 (b) PTB (c) C4

Figure 1: Perplexity comparison of `Bi-Mamba`, GPTQ and Bi-LLM on Wiki2, PTB and C4 datasets. GPTQ and Bi-LLM show significant performance degradation when the bit is low. `Bi-Mamba` demonstrates low perplexity in 1 bit and shows similar performance as GPTQ-8bit.

quantization to SSM models by reducing weights to a binary setting. Through this extreme quantization, we demonstrate that SSM models can be effectively binarized for both training and inference, maintaining high performance while significantly reducing memory footprint and energy consumption.

Our work pioneers a new framework for 1-bit representation in linear computational complexity models, potentially facilitating the development of specialized hardware tailored for fully low-bit SSM-based language models. We provide comprehensive experiments showing that `Bi-Mamba` achieves competitive performance comparing to full-precision counterpart large language models (LLMs), thereby establishing the feasibility and benefits of binarizing SSM models. To further understand the behaviors of binarization on Mambas, we conducted extensive empirical studies to analyze the distribution of pre-trained and binarized weights in Mambas. The results of this study are detailed in Section 4 and Appendix **??**, leading to two key observations:

- As shown in Figure 2, post-training-binarization methods like BiLLM (Huang et al., 2024) and PB-LLM (Shang et al., 2023) typically tend to shift the distribution of weights on Mamba after binarization, resulting in a misaligned distribution to the optimal binary weights. Our binarization-aware training, however, ensures that the binarized weights remain close to the original weight distribution, preserving the largest capability in weight representation binarization.
- Based on our empirical experiments, applying existing LLM post-binarization methods (Frantar et al., 2023; Huang et al., 2024), even when retaining salient weights, often severely degrades the performance of the Mamba model. Without accounting for salient weights, binarization-aware training appears to be the only feasible solution for effectively binarizing models like Mamba while preserving competitive performance.

Our contributions in this paper are as follows: (1) This is the first work to successfully binarize the Mamba architecture to 1-bit from scratch while maintaining strong performance. (2) We explore the potential parameter space for binarization in Mamba, offering valuable insights for future research. (3) The `Bi-Mamba` model pretrained by our method can serve as a robust base model for downstream tasks in resource-limited scenarios and can be easily adapted for other NLP applications.[1]

2 Related Work

Post-Training Quantization. Due to their large parameter count, LLMs (Brown et al., 2020; OpenAI, 2023; Touvron et al., 2023) are resource intensive to run. Therefore they are often quantized to low bits representations during inference. This type of quantization is typically called post-training quantization (PTQ) (Dettmers et al., 2022a; Xiao et al., 2023; Wei et al., 2022; Frantar et al., 2023; Yao et al., 2022; Xiao et al., 2023; Lin et al., 2024), where the quantization operation is applied to the pretrained models.

[1] We will make all our models, code, and training datasets fully available, we aim to support further research in this promising direction and encourage the exploration of binarized SSM models for more efficient large language models.

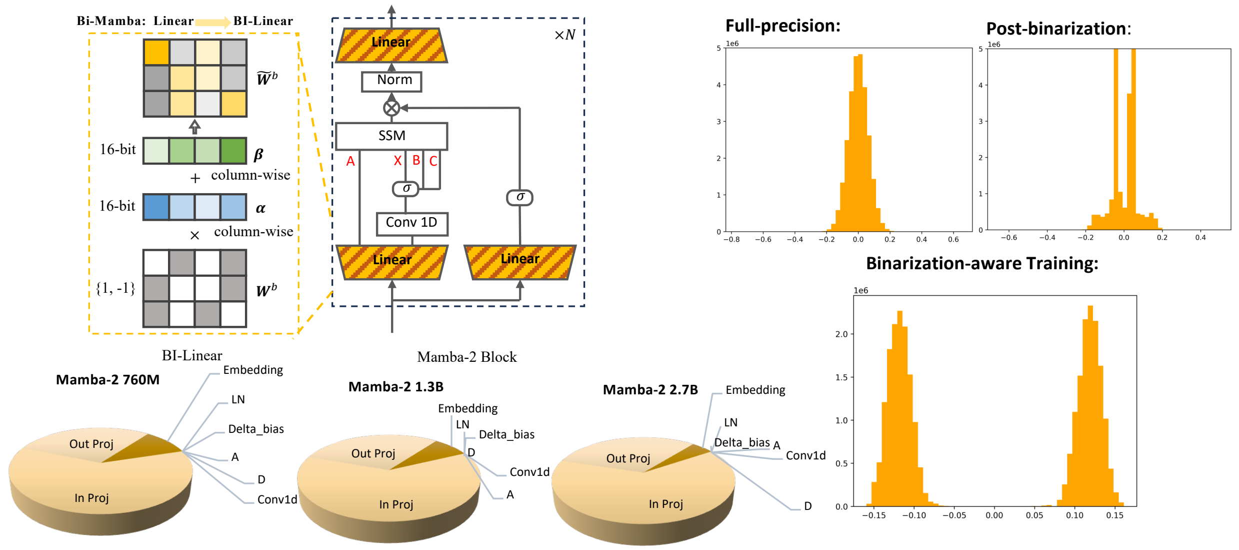


Figure 2: Illustration of the Bi-Mamba framework. Our Bi-Mamba binarizes both input and output projection matrices. Compared with the post-binarization method (Bi-LLM), our binarization-aware training method (Bi-Mamba) generates a more similar weight distribution (after scaling) on each part.

Post-training quantization methods of LLMs are typically based on the empirical observations that a small set of salient features in pretrained Transformers have significantly large magnitudes, and quantization of these features needs to be extra careful (Xiao et al., 2023; Lin et al., 2023). PTQ methods have led to near lossless accuracy drop for INT8 weights and activations quantization, and INT4 weight-only quantization. While many works (Huang et al., 2024; Egiazarian et al., 2024; Shao et al., 2023; Chee et al., 2023; Tseng et al., 2024) have been focused on pushing post-training weight quantization below 4 bits, they often leads to drastic drop of model performance.

Quantization-Aware Training. Different from post-training quantization, quantization-aware training (QAT) aims to learn the quantized models during training. In the LLM era, QAT is less investigated as compared to PTQ because LLMs typically require huge amounts of compute resources to train. Li et al. (2023b) propose a method to adopt LoRA fine-tuning during QAT, making the quantization procedure resource efficient. However, these methods still require the pretrained weights to initialize the student LLMs. Wortsman et al. (2023) propose a method to stabilize the training of low-bit large vision-language models from random initializations. Most recently, Ma et al. (2024c) trained LLMs with tenary weights from random initializations, achieving non-trivial performance. Ma et al. (2024a) further proposed a distillation based method to pretrain binarized LLMs from scratch.

State-Space Models. While LLMs are often built with Transformers (Vaswani et al., 2017), their self-attention operations suffer from quadratic time complexity. This makes Transformers inefficient to run on long sequences. Therefore there have been many research efforts aiming at addressing the efficiency issue (Peng et al., 2023; Beck et al., 2024; Katharopoulos et al., 2020; De et al., 2024). Among these efforts, SSMs (Gu & Dao, 2024; Dao & Gu, 2024; Gu et al., 2021a) are a type of recurrent neural networks. The latest advanced SSMs like Mamba (Gu & Dao, 2024) and Mamba-2 (Dao & Gu, 2024) have demonstrated comparable performance to Transformers, while having linear complexity with respect to the sequence length. Despite their promising capabilities, how to effectively quantize this architecture has rarely been investigated. Concurrent to our work, Pierro & Abreu (2024) have looked at post-training quantization of Mamba.

3 Approach

3.1 Preliminary: Mamba Series

Mamba belongs to a class of models known as State Space Models (SSMs), which is able to offer performance and scaling laws comparable to the Transformer while remaining practical at extremely long sequence lengths

Model Size	Embedding	LN	Δ_{bias}	A	D	Conv1d	In Proj.	Out Proj.
Mamba-2 760M	9.901	0.029	0.0003	0.0003	0.0003	0.102	60.936	29.031
Mamba-2 1.3B	7.664	0.022	0.0002	0.0002	0.0002	0.078	62.270	29.964
Mamba-2 2.7B	4.763	0.018	0.0002	0.0002	0.0002	0.064	64.115	31.039

Table 1: Proportional distribution of parameters across different modules in Mamba-2. Input and output matrices take up the majority of the parameters in Mamba-2 models, around 90% and more.

(e.g., one million tokens). It achieves this extended context by eliminating "quadratic bottleneck" present in the attention mechanism. The vanilla structured state space sequence models (S4) transform a 1-dimensional sequence $x \in R^T$ into another sequence $y \in R^T$ by utilizing an latent state $h \in R^{T,N}$ as follows:

$$\begin{aligned} h_t &= Ah_{t-1} + Bx_t \\ y_t &= Ch_t + Dx_t \end{aligned} \tag{1}$$

where $A \in R^{N,N}$, $B \in R^{N,1}$, $C \in R^{1,N}$, $D \in R$. h_{t-1} is the hidden state, x_t is the input, the observation that the model gets each time. h_t then represents the derivative of the hidden state, i.e. how the state is evolving.

Since the parameters in Equation 1 are constant through time, this model is linear time-invariant (LTI). The LTI model gives equal attention to all elements when processing sequences, which prevents the model from effectively understanding natural language. To address this, Mamba improved it to the Selective State Space Model, where B and C are obtained through a linear projection of x_t. Meanwhile, the above equation applies to dynamic systems with continuous input and output signals. So, it needs to do discretization when dealing with text sequences:

$$\begin{aligned} \overline{A} &= exp(\Delta A) \\ \overline{B} &= exp(\Delta A)^{-1}(exp(\Delta A) - I) \cdot \Delta B \end{aligned} \tag{2}$$

where, Δ is related to x_t and a learnable parameter Δ_{bias}. Then, the calculating process of selective SSM is as follows:

$$\begin{aligned} h_t &= \overline{A}h_{t-1} + \overline{B}x_t \\ y_t &= Ch_t + Dx_t \end{aligned} \tag{3}$$

Centered around selective SSM, combined with linear projection for input and output along with layer normalization, this forms the basic block for Mamba.

Compared to Mamba, Mamba-2 further replaces selective SSM with the state space duality (SSD). In SSD, A is simplified as scalar-times-identity structure, and SSD uses a larger head dimension which is 1 in Mamba. Meanwhile, Mamba-2 block also introduces simplifications, such as removing sequential linear projections in Mamba block, to facilitate parallel training. The basic structure of Mamba-2 is shown in Figure 2.

3.2 Bi-Mamba

Binarization Space in Mamba. To begin with, we need to identify what weight matrices can be binarized in Mamba architecture. We use the latest Mamba-2 Dao & Gu (2024) as our base architecture. According to the description above, we can interpret the SSD matrices of A, B, C, D, and Δ_{bias} more intuitively, as well as other layers including *embedding, layer normalization (LN), Conv-1d, inner linear projection, out linear projection* in Mamba-2 to better understand the effect after binarization, so that to determine the binarization space. Briefly, A is the transition state matrix, representing how the current state transitions to the next state. It answers the question of *how should the model gradually forget the less relevant parts of the state over time?* B maps the new input to the state, addressing the point of *which parts of the new input should the model retain?* C maps the state to the output of the SSM, asking that *how can the model utilize the current state to make an accurate next prediction?* D shows how the new input directly influences the output, acting as a modified skip connection and asking *how can the model incorporate the new input into the prediction?*

Considering our base architecture Mamba-2, we present the specific parameters proportion for different layers across different sizes as shown in Figure 2 and Table 1[2]. It is observed that, in Mamba-2, the vast majority of the model's parameters are in the linear modules, excluding the causal head. For instance, in Mamba-2-2.7B, these parameters account for 95.2% of the entire model. Additionally, the embedding module shares parameters with the causal head. Binarizing the embedding significantly diminishes its capability to represent token semantics, thereby reducing model performance. Therefore, in our `Bi-Mamba`, we only binarize the parameters in the linear modules (excluding the causal head). This strategy aims to maintain a high compression ratio of 90% while still allowing the binarized model to perform effectively.

Simple Learnable Scaling Factors for Better Capability. To binarize Mamba, we replace the original linear modules with the FBI-Linear module introduced by FBI-LLM (Ma et al., 2024a). Specifically, the FBI-Linear module primarily consists of two parts: a matrix $\boldsymbol{W}^b \in \mathbb{R}^{m\times n}$ made up solely of $\{1, -1\}$ and high-precision scale factors $\boldsymbol{\alpha} \in \mathbb{R}^n$ and $\boldsymbol{\beta} \in \mathbb{R}^n$. The inference process of FBI-Linear is as follows:

$$\boldsymbol{y} = \widetilde{\boldsymbol{W}}^b \boldsymbol{x} \tag{4}$$

where $\widetilde{\boldsymbol{W}}^b$ is derived by performing column-wise multiplication with $\boldsymbol{\alpha}$ and addition with $\boldsymbol{\beta}$ respectively:

$$\widetilde{\boldsymbol{W}}^b_{\cdot,i} = \alpha_i \boldsymbol{W}^b_{\cdot,i} + \beta_i \tag{5}$$

where α_i and β_i are the learnable scaling and shifting factors at i-th layer.

Objective of Training. Our training objective is a cross-entropy loss between the outputs of the target student model and the pretrained teacher model at each step of the autoregressive scheme for next-token prediction. This can be expressed as:

$$\mathcal{L}_{\texttt{Bi-Mamba}} = -\frac{1}{n}\sum_{k}^{n} \boldsymbol{p}^{\mathcal{T}}\left(x^{k+1}\right) \cdot \log \boldsymbol{p}^{\mathcal{S}}\left(x^{k+1}\right) \tag{6}$$

where n is the number of input tokens. Here, $\boldsymbol{p}^{\mathcal{T}}\left(x^{k+1}\right)$ represents the token distribution over the vocabulary at the k-th step predicted by the teacher model, and $\boldsymbol{p}^{\mathcal{S}}\left(x^{k+1}\right)$ is the corresponding predicted distribution by the student model.

Overall Design of `Bi-Mamba`. During binarization-aware training of autoregressive distillation (Ma et al., 2024a), we compute the cross-entropy between the output probability distributions of a high-precision pre-trained model and our target `Bi-Mamba` model. In this process, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are learnable parameters, while $\boldsymbol{W}^b$ is derived using the $sign(\cdot)$ function from a learnable high-precision matrix $\boldsymbol{W}^f \in \mathbb{R}^{m\times n}$. Since the $sign(\cdot)$ function is non-differentiable, we use the Straight Through Estimator (STE) (Bengio et al., 2013) as prior studies (Rastegari et al., 2016; Alizadeh et al., 2019) in BNNs to approximate the gradients of the input variables, enabling the continuation of backward propagation.

4 Experiments

In our experiments, we solely binarize the parameters in the most linear modules, while keeping other model parameters and activation at original precision. Notably, we do not strictly represent the binarized parameters with 1-bit, instead, we use high-precision values to simulate the binarized parameters. We train `Bi-Mamba` on different scales and evaluate their performance across multiple tasks.

4.1 Setup

Training Dataset. Following FBI-LLM, we train `Bi-Mamba` with the Amber dataset (Liu et al., 2023) which contains a total 1.26 Trillion tokens from RefinedWeb (Penedo et al., 2023), StarCoder (Li et al., 2023a), and RedPajama-v1 (Computer, 2023). The data is partitioned into 360 chunks, each comprising approximately 3.5B tokens on average.

[2]Since B and C are derived from the linear projection of each layer's inputs, and the language model head is tied to the embedding layer, these modules are ignored in our design.

Training details. We train `Bi-Mamba` on different scales with Mamba-2 architecture. Specifically, we binarize input projection and output projection matrices in 780M, 1.3B and 2.7B Mamba-2 models. We use LLaMA2-7B as the teacher model for all `Bi-Mambas` to calculate autoregressive distillation loss. Therefore, all `Bi-Mambas` we trained have the same vocabulary and tokenizer as LLaMA2. We train models until convergence with 32× NVIDIA A100 GPUs in total and maintain BF16 precision while training. For configuring different sizes of `Bi-Mamba`, the details can be found at Table 2. We follow the same architectures as the original Mamba-2 models and apply binarization on both input and output projection matrices. The training process uses the Adam optimizer with parameters $\beta_1 = 0.9$ and $\beta_2 = 0.95$. The initial learning rate is set at $2.5e-4$ and follows a cosine schedule, decreasing to $2.5e-5$ over 2,000 warm-up steps. Gradient clipping is set at 1.0. We train `Bi-Mamba` 780M, 1.3B, 2.7B with 30 data chunks, which are 105B tokens.

	`Bi-Mamba` 780M	`Bi-Mamba` 1.3B	`Bi-Mamba` 2.7B
d_model	1536	2,048	2,560
n_layer	48	48	64
vocabulary size	32,000	32,000	32,000
learning rate	2.5e-4	2.5e-4	2.5e-4
batch size (token)	0.5M	0.5M	0.5M
teacher model	LLaMA2-7B	LLaMA2-7B	LLaMA2-7B

Table 2: The configuration and training details for `Bi-Mamba`.

Evaluation Metrics. We evaluate the models based on their zero-shot performance in downstream tasks including BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2020), HellaSwag (Zellers et al., 2019), Winogrande (Sakaguchi et al., 2021), ARC (Clark et al., 2018), and OpenbookQA (Mihaylov et al., 2018). All downstream evaluations are done with *lm-evaluation-harness* package (Gao et al., 2024). We also use perplexity on Wikitext2 (Merity et al., 2016), PTB (Marcus et al., 1993), C4 (Raffel et al., 2020) dataset as the evaluation metric. Perplexity measures how well a probability model predicts a token, quantitatively measuring the model's generation power.

Baselines. We compare our work with quantization and binarization methods, namely GPTQ (Frantar et al., 2023) and Bi-LLM (Huang et al., 2024). GPTQ is a post-training quantization method while Bi-LLM is a post-training binarization method. We apply the official implementation of GPTQ and quantize the Mamba-2 models into 3 and 2 bits, respectively. For Bi-LLM, we also utilize their official implementation and binarize Mamba-2 models. Moreover, we add quantization-aware training methods, OneBit (Xu et al., 2024) and BitNet-1.58bit (Ma et al., 2024b) for comparison. For BitNet, we report the results in the original paper for comparison while for OneBit, since they only released the official weights of 7B, we only add the results of 7B models. Furthermore, we include results from open-source full-precision transformer-based models of various sizes, such as OPT (Zhang et al., 2022), and TinyLLaMA (Zhang et al., 2024), as references.

4.2 Main Results

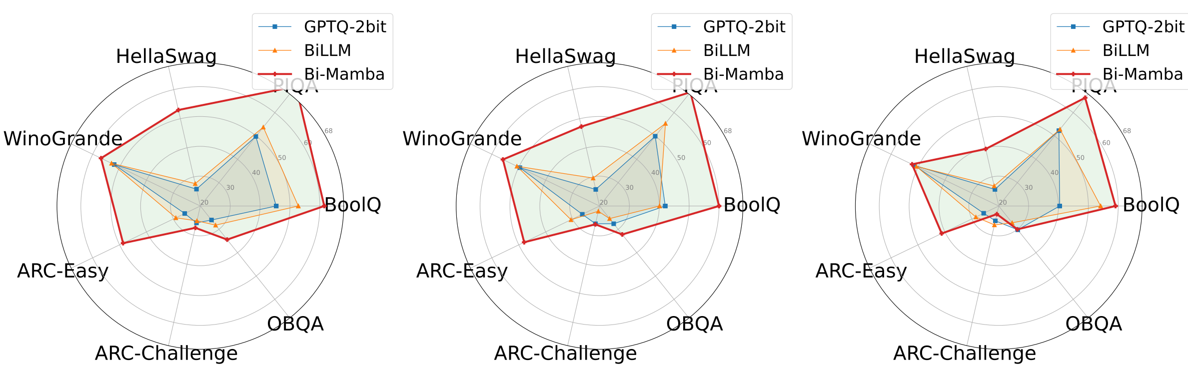
Table 3 presents the comparison of our `Bi-Mamba` model against various baselines on downstream tasks and perplexity on Wiki2, PTB, and C4 datasets. These evaluations provide insight into model generalization capabilities without further task-specific fine-tuning. The visualization of performance comparison is shown in Figure 3.

For the 780M Mamba-2 model, `Bi-Mamba` demonstrates an average downstream performance of 45.3, outperforming GPTQ-3bit and Bi-LLM, which achieve 42.6 and 38.1, respectively. In perplexity assessments, `Bi-Mamba` reports scores of 13.4, 32.4, and 14.5 on Wiki2, PTB, and C4, respectively, with the baseline models exhibiting up to 10× higher perplexity. Moreover, `Bi-Mamba` 780M surpasses the binarization-aware training method, BitNet on the zero-shot performance on downstream tasks. BitNet obtains 44.3 scores on average while `Bi-Mamba` 780M achieves 45.3 scores.

For the 1.3B model, `Bi-Mamba` achieves a notable downstream accuracy of 48.4 on average, surpassing GPTQ-2bit's 35.3 and Bi-LLM's 36.2. This performance indicates `Bi-Mamba`'s enhanced generalization

Method	Model	Size	Zero-shot Accuracy ↑								Perplexity ↓		
			BoolQ	PIQA	HS	WG	ARC-e	ARC-c	OBQA	Avg.	Wiki2	PTB	C4
Mamba-2 (Dao & Gu, 2024)	M	780M	61.5	71.8	54.9	60.2	54.3	28.5	36.2	52.5	11.8	20.0	16.5
GPTQ-3bit	M	780M	44.6	62.9	40.3	53.3	40.6	26.4	30.6	42.6	152.5	192.5	186.0
GPTQ-2bit	M	780M	40.4	52.3	25.7	51.3	25.6	25.1	30.2	35.2	1.6e+8	1.3e+8	7.3e+7
BiLLM	M	780M	54.1	52.9	26.9	50.6	28.5	26.5	27.2	38.1	1.8e+4	2.4e+4	1.5e+4
BitNet-1.58bit	T	700M	58.2	68.1	35.1	55.2	51.8	21.4	20.0	44.3	-	-	-
`Bi-Mamba`	M	780M	58.5	68.0	41.6	52.0	42.4	24.3	30.6	**45.3**	**13.4**	**32.4**	**14.5**
TinyLLaMA (Zhang et al., 2024)	T	1.3B	57.8	73.3	59.2	59.1	55.3	30.1	36.0	53.0	7.8	30.5	9.9
OPT (Zhang et al., 2022)	T	1.3B	57.8	72.5	53.7	59.5	51.0	29.5	33.4	51.1	14.6	20.3	16.1
Mamba-2 (Dao & Gu, 2024)	M	1.3B	64.3	73.7	59.9	61.0	60.4	33.1	37.8	55.8	10.4	17.7	14.8
GPTQ-3bit	M	1.3B	56.8	68.2	48.5	54.4	48.0	28.8	30.4	47.8	29.3	56.5	37.3
GPTQ-2bit	M	1.3B	42.0	49.9	25.7	49.6	26.4	26.1	27.6	35.3	1.2e+6	1.0e+6	1.3e+6
BiLLM	M	1.3B	40.1	55.4	29.6	50.7	30.6	21.8	25.4	36.2	4943.2	3540.8	4013.6
BitNet-1.58bit	T	1.3B	56.7	68.8	37.7	55.8	54.9	24.2	19.6	45.4	-	-	-
`Bi-Mamba`	M	1.3B	60.0	68.8	47.3	55.9	48.0	26.3	32.2	**48.4**	**11.7**	**29.9**	**12.9**
Mamba-2 (Dao & Gu, 2024)	M	2.7B	70.7	76.3	66.6	63.9	64.8	36.3	38.8	59.6	9.1	15.3	13.3
GPTQ-3bit	M	2.7B	54.8	69.9	54.0	56.0	51.6	33.3	32.8	50.3	21.2	39.0	29.3
GPTQ-2bit	M	2.7B	45.4	49.8	25.8	52.0	25.8	25.8	26.0	35.8	2.1e+5	2.3e+5	1.8e+5
BiLLM	M	2.7B	52.8	53.8	27.7	53.0	29.1	25.1	28.2	38.5	8707.0	1.7e+4	1.3e+4
OneBit	T	6.7B	63.3	67.7	52.5	58.1	41.6	29.3	34.0	49.5	-	-	-
BitNet-1.58bit	T	3.0B	61.5	71.5	42.9	59.3	61.4	28.3	26.6	50.2	-	-	-
`Bi-Mamba`	M	2.7B	58.0	72.5	54.3	56.1	51.4	29.1	32.6	**50.6**	**10.0**	**21.9**	**11.3**

Table 3: Performance comparison with baselines on downstream tasks and perplexity. Here, *Model* represents the architecture of the quantized model, with "M" indicating Mamba-2 (Dao & Gu, 2024) and "T" indicating Transformer (Vaswani et al., 2017). *HS*, *WG*, and *OBQA* are abbreviations for HellaSwag, Winogrande, and OpenbookQA, respectively. We divide the table into three blocks based on model size. Our `Bi-Mamba` achieves lower perplexity than Bi-LLM and GPTQ on Wiki2, PTB and C4 datasets, as well as the best average performance on downstream tasks compared with GPTQ-2bit and Bi-LLM.



(a) 2.7B (b) 1.3B (c) 780M

Figure 3: Visualization of results comparison on Mamba-2 in the scale of 2.7B, 1.3B and 780M.

across a wider range of tasks at this model size. Additionally, `Bi-Mamba` demonstrates substantially improved perplexity, registering scores of 13.2, 30.8, and 14.0 on Wiki2, PTB, and C4 datasets, respectively. In comparison, GPTQ-2bit and Bi-LLM present considerably higher perplexity values, underscoring the efficiency of `Bi-Mamba`'s binarization in maintaining linguistic coherence. Moreover, `Bi-Mamba` 1.3B model beats the training-based method, BitNet, which obtains 45.4 scores on average on the downstream tasks.

For the 2.7B model, `Bi-Mamba` further extends its lead, achieving an average downstream accuracy of 50.6, compared to 35.8 for GPTQ-2bit and 38.5 for Bi-LLM. Notably, `Bi-Mamba` maintains low perplexity across all datasets, with scores of 10.0, 21.9, and 11.3 on Wiki2, PTB, and C4, respectively. These results highlight `Bi-Mamba`'s ability to retain high-level performance in both task accuracy and linguistic fluency as model complexity scales up. Similarly, `Bi-Mamba` 2.7B model outperforms all training-based methods including

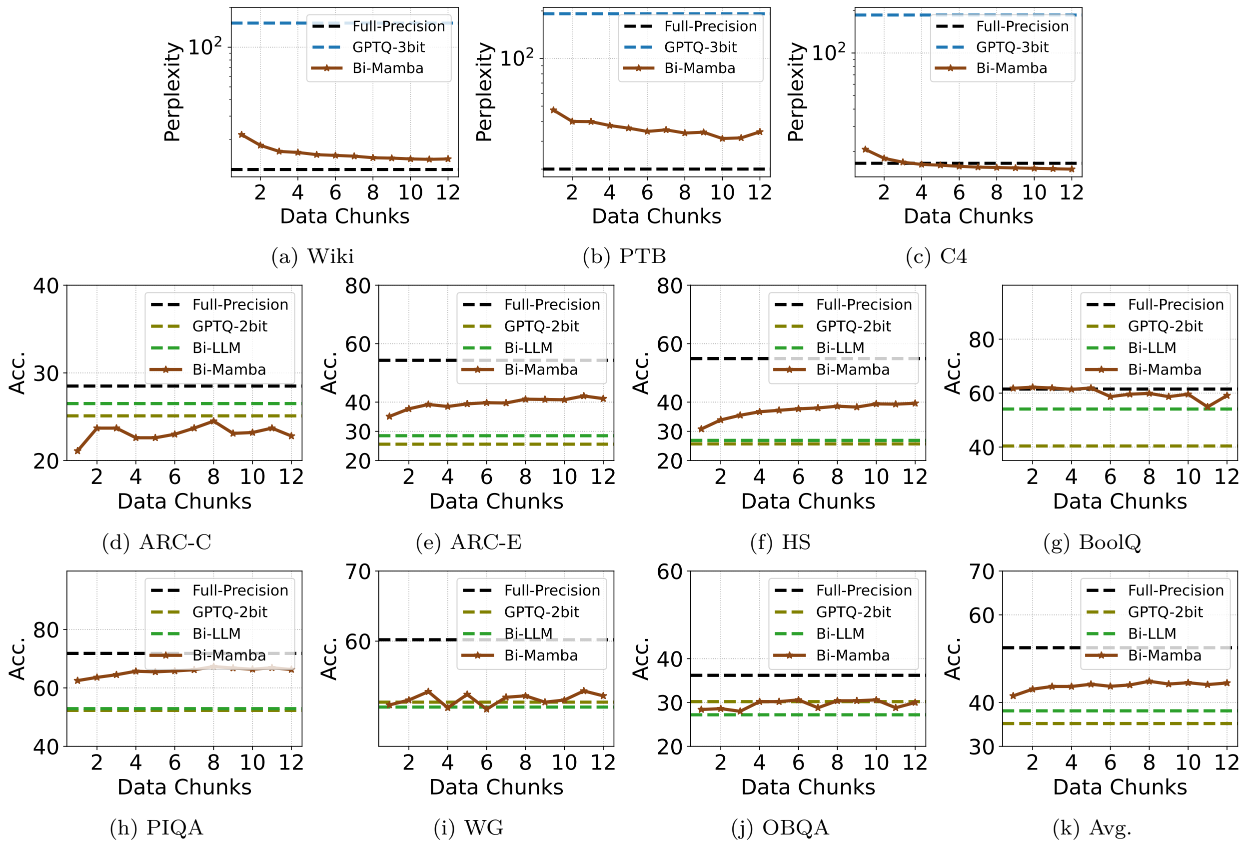


Figure 4: Downstream performance and perplexity curve of `Bi-Mamba` with different training costs.

OneBit and BitNet. OneBit only obtains 49.5 scores on average with 6.7B parameters and BitNet gains 50.2 scores with 3.0B parameters. The results demonstrate the effectiveness of `Bi-Mamba`. In summary, `Bi-Mamba` consistently demonstrates superior zero-shot performance and perplexity reduction across model sizes, substantiating its robustness and versatility compared to GPTQ and Bi-LLM, particularly in larger, more complex models.

5 Analysis

5.1 Training Result Dynamics

In this section, we discuss the performance of `Bi-Mamba` as the training progresses with different training costs/budgets. The main results are shown in Figure 4. We provide the downstream performance and perplexity curve along different training costs on the Mamba-2-780M model. More results on 2.7B and 1.3B Mamba-2 models can be found in the Appendix ??. From the figure, we can observe that the perplexity decreases quickly at the beginning of training and gradually converges to the full-precision perplexity. The perplexity of `Bi-Mamba` on the wiki2 and C4 datasets is more stable than the perplexity on the PTB dataset. Notably, on the C4 dataset, the final perplexity of `Bi-Mamba` is even lower than the full-precision model, highlighting the superior performance of `Bi-Mamba`. Interestingly, early in training, our `Bi-Mamba` model surpasses GPTQ-3bit on the perplexity, demonstrating the effectiveness of binarization-aware training. Since the perplexity of GPTQ-2bit and BiLLM is extremely high on all datasets, we omit them in the figure and refer to Table 3 for detailed results of GPTQ-2bit and BiLLM. Moving to the downstream task evaluation, we first observe the catastrophic performance degradation of the binarized models, whose performance is even lower than the random results on many benchmarks such as the results on ARC-E and ARC-C. This indicates that directly applying the naive binarization method destroys the ability of the full-precision model. However, after binarization-aware training, the model recovers the performance on all benchmarks and finally

Model	Zero-shot Acc. ↑								Perplexity ↓		
	BoolQ	PIQA	HS	WG	ARC-e	ARC-c	OBQA	Avg.	Wiki2	PTB	C4
Mamba-2-780M	61.5	71.8	54.9	60.2	54.3	28.5	36.2	52.5	11.8	20.0	16.5
Bi-Mamba (In_Proj)	59.0	68.3	41.2	53.7	42.6	24.3	29.4	45.5	13.8	30.5	14.4
Bi-Mamba (Fully)	58.5	68.0	41.6	52.0	42.4	24.3	30.6	45.3	13.4	32.4	14.5
Mamba-2-1.3B	64.3	73.7	59.9	61.0	60.4	33.1	37.8	55.8	10.4	17.7	14.8
Bi-Mamba (In_Proj)	62.1	71.7	50.4	53.8	49.5	26.8	33.0	49.6	10.7	26.0	12.0
Bi-Mamba (Fully)	60.0	68.8	47.3	55.9	48.0	26.3	32.2	48.4	11.7	29.9	12.9
Mamba-2-2.7B	70.7	76.3	66.6	63.9	64.8	36.3	38.8	59.6	9.1	15.3	13.3
Bi-Mamba (In_Proj)	62.4	74.6	58.9	57.1	54.0	29.4	35.4	53.1	9.1	19.5	10.5
Bi-Mamba (Fully)	58.0	72.5	54.3	56.1	51.4	29.1	32.6	50.6	10.0	21.9	11.3

Table 4: Performance comparison of partial-binarization and fully-binarization on perplexity and downstream tasks. The results show that the performance gap between partial-binarization and fully-binarization is not significant, indicating that the fully binarized model can maintain competitive performance with the partial binarization model.

outperforms all baselines including GPTQ-3bit, GPT-2bit and Bi-LLM. This underscores the importance of binarization-aware training in achieving competitive results.

5.2 Binarization Space

In this section, we explore the binarization space of Mamba models and discuss the effect of binarizing each part. We conduct experiments that binarize the In_Proj and binarize both In_Proj and Out_Proj. The binarized models are trained with the same data. The results are shown in Table 4. Partial and full binarization are compared with the full-precision Mamba model on perplexity and downstream tasks. First, the results in the table show that partial binarization generally retains higher zero-shot accuracy compared to full binarization. However, the performance gap between the partial and full binarized models is not significant. For instance, in the Mamba-2-2.7B model, partial binarization achieves an average accuracy of 53.1, while full binarization reduces this to 50.4. Across all model sizes, partial-binarized `Bi-Mamba` consistently outperforms full-binarized `Bi-Mamba` on most benchmarks, though shows minor performance degradation compared with full-precision models. It also suggests that fully binarization remains highly competitive and does not substantially lag. In terms of perplexity, the fully binarized model also performs comparably to the partial model. For example, in the Mamba-2-780M model, the C4 dataset perplexity for full-binarized `Bi-Mamba` (Fully) is 15.0, compared to 14.4 for partial-binarized `Bi-Mamba`, demonstrating that full binarization does not impose a significant perplexity increase. These findings highlight that the fully binarized model can maintain competitive performance with the partial binarization model, particularly in terms of perplexity, while still benefiting from greater storage and computational efficiency.

5.3 Comparison with Full Precision Small Models

Instead of binarization, one can train small models with full precision from scratch. We add the performance comparison of `Bi-Mamba` and small models pretrained with full precision, as shown in Table 5. We utilize the official pretrained weight from Mamba2 including models with 130M and 370M parameters pretrained with 300B tokens. 130M and 370M models in 16 bits are equivalent to 2.0B and 5.9B models in 1 bit, respectively. From the table, all `Bi-Mamba` models including 780M, 1.3B and 2.7B with only 105B token training are better than the 130M models with full precision on average performance. Specifically, Mamba-2 130M obtains 44.2 of accuracy on downstream tasks, and 20.0, 35.1, 25.2 of perplexity on Wiki, PTB and C4 datasets while the smallest model of `Bi-Mamba` with 780M parameters achieves 45.3 of accuracy on all downstream tasks and 13.4, 32.4 and 14.5 on Wiki, PTB and C4 datasets. Moreover, `Bi-Mamba` 1.3B and 2.7B models achieves higher average performance than full-precision 370M Mamba-2 model. Full-precision Mamba-2 370M gains 48.4 on average on downstream tasks. `Bi-Mamba` 1.3B beats the 370M Mamba-2 model with 48.4 of accuracy on downstream task. The results indicate that binarization with post-training is better than training a full-precision small model from scratch.

Model	Tokens	Zero-shot Acc. ↑								Perplexity ↓		
		BoolQ	PIQA	HS	WG	ARC-e	ARC-c	OBQA	Avg.	Wiki2	PTB	C4
Mamba-2 (130M, 16-bit)	300B	55.1	64.0	35.3	52.6	47.4	24.1	30.6	44.2	20.0	35.1	25.2
Mamba-2 (370M, 16-bit)	300B	54.0	69.2	46.9	55.4	48.7	26.7	32.4	47.6	14.1	24.2	19.0
Bi-Mamba (780M, 1-bit)	105B	58.5	68.0	41.6	52.0	42.4	24.3	30.6	45.3	13.4	32.4	14.5
Bi-Mamba (1.3B, 1-bit)	105B	60.0	68.8	47.3	55.9	48.0	26.3	32.2	48.4	11.7	29.9	12.9
Bi-Mamba (2.7B, 1-bit)	105B	58.0	72.5	54.3	56.1	51.4	29.1	32.6	50.6	10.0	21.9	11.3

Table 5: The performance comparison of Bi-Mamba and a pretrained model in small size. All Bi-Mamba models are better than Mamba-2-130M 16-bit model, which is equivalent to a 2.0B model in 1-bit. Moreover, Bi-Mamba 1.3B and 2.7B models achieve higher performance than Mamba-370M 16-bit model, which is equivalent to a 5.9B model in 1-bit, demonstrating the effectiveness of quantization-aware training instead of training a small model directly.

Model	Model Param.	Storage Size	Compress Ratio
Mamba-2	780M	1.45GB	-
Bi-Mamba (InProj)	780M	0.63GB	56.5%
Bi-Mamba (Full)	780M	0.22GB	84.8%
Mamba-2	1.3B	2.50GB	-
Bi-Mamba (InProj)	1.3B	1.01GB	59.6%
Bi-Mamba (Full)	1.3B	0.33GB	86.8%
Mamba-2	2.7B	5.03GB	-
Bi-Mamba (InProj)	2.7B	2.01GB	60.0%
Bi-Mamba (Full)	2.7B	0.55GB	89.0%

Table 6: Storage efficiency Bi-Mamba. Compared with partial binarization, full binarization can reduce the storage size significantly in all scale.

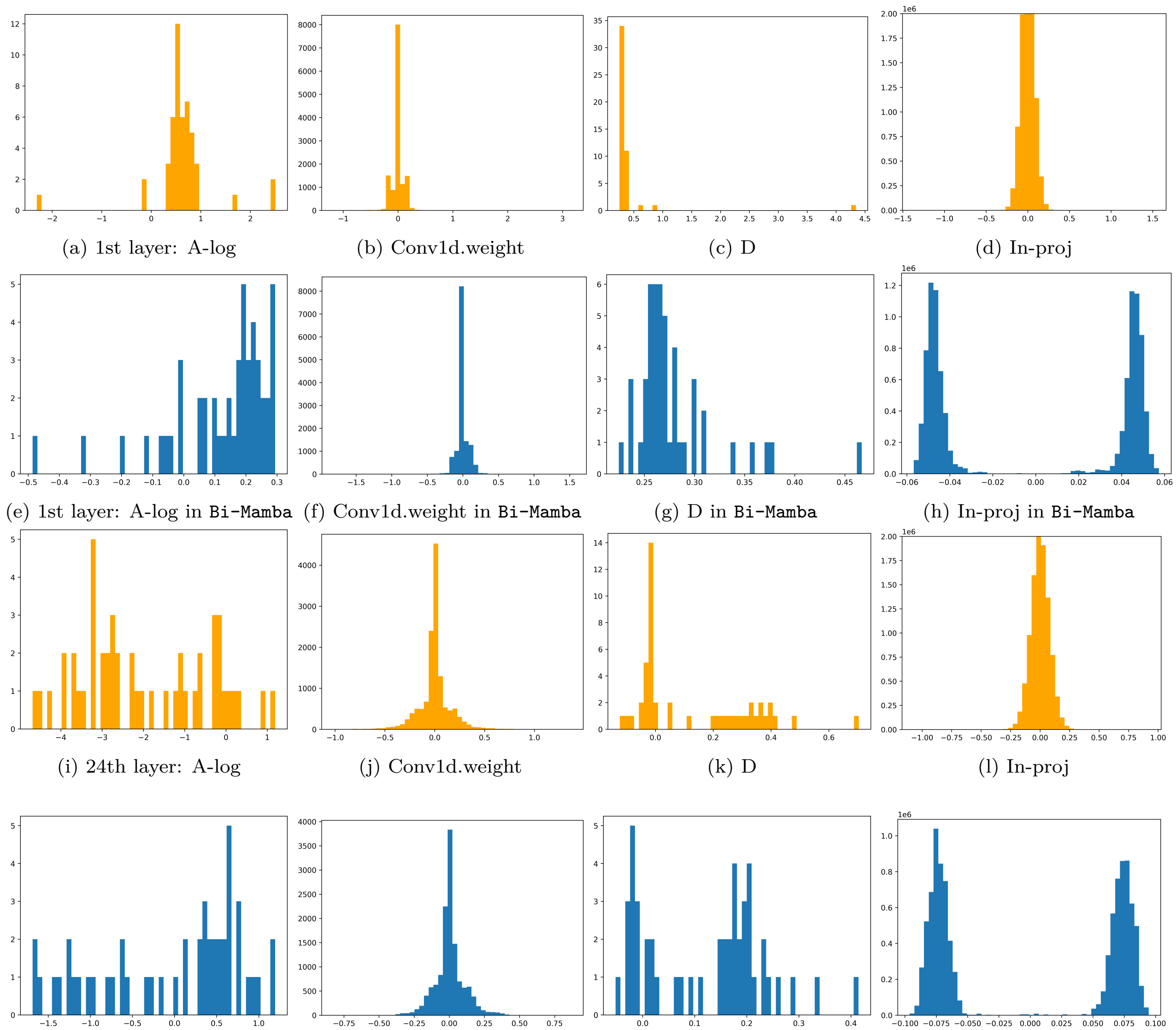
5.4 Storage Efficiency

Model binarization can significantly reduce the storage requirement in the disk. Following Bi-LLM (Huang et al., 2024), we provide the theoretical storage requirement for our Bi-Mamba in different model sizes compared with full-precision models, as shown in Table 6. For each parameter size, the storage requirements for the original full-precision Mamba-2 model are substantially larger than those for the binarized Bi-Mamba including partial and full binarization. Specifically, fully-binarized Bi-Mamba demonstrates the highest compression ratio, achieving reductions of more than 80%. In contrast, partial-binarized Bi-Mamba provides relatively moderate compression, ranging from 55% to 60%. This analysis highlights the efficiency of fully-binarization in significantly reducing storage requirements while maintaining the model parameter count, making it a highly storage-efficient alternative for large models.

5.5 Weight Distribution

We visualized the weight distributions of different modules in Mamba-2 (Orange histograms) and Bi-Mamba (Blue histograms), as shown in Figure 5. We visualize the weight parameter distributions of different modules in the first (1st), mid (24th) and final (48th) layers of the corresponding 780M models. The input and output projection matrices are the values after re-scaling. Each pair of histograms compares how Bi-Mamba modifies the distribution of weights in different modules, no matter whether the module is binarized or not, illustrating the impact of Bi-Mamba on each module.

Specifically, in the first layer, the weight distribution of the original Mamba-2 such as *Conv1d.weight*, and *D* are tightly concentrated, indicating the strong focus on specific values. In contrast, the weight distribution in Bi-Mamba in the first layer is much divergent with additional peaks in the histograms such as in *A-log*, and *D*. The divergent weight distributions in Bi-Mamba suggest that Bi-Mamba intentionally captures broader values in the initial layers to retain sufficient information for binarized modules.



(a) 1st layer: A-log (b) Conv1d.weight (c) D (d) In-proj

(e) 1st layer: A-log in `Bi-Mamba` (f) Conv1d.weight in `Bi-Mamba` (g) D in `Bi-Mamba` (h) In-proj in `Bi-Mamba`

(i) 24th layer: A-log (j) Conv1d.weight (k) D (l) In-proj

(m) 24th layers: A-log `Bi-Mamba` (n) Conv1d.weight in `Bi-Mamba` (o) D in `Bi-Mamba` (p) In-proj in `Bi-Mamba`

Figure 5: Distribution Comparison of each weight in Mamba-2 and `Bi-Mamba` modules at the 1st and 24th layers.

With more variability at the initial layers, `Bi-Mamba` can process the diverse initial features even in low-bit precision. In the mid-depth layers such as the 24th layer, the weight distribution of both the original Mamba and `Bi-Mamba` show similar patterns as in the first layer. However, the divergence is more moderate compared with the divergence in the first layer. This suggests that in the intermediate layers, `Bi-Mamba` can refine the intermediate representation with generalization with binarized weights. The distribution of all values and the final layer is present in the appendix Figure **??**, **??**, **??**. In short, in the last layer, the divergence patterns also remain while the distribution is much narrower compared with previous layers, reflecting a more concentrated range of values.

The focused distribution helps to model to generate a stable and reliable final representation. In all, our `Bi-Mamba` includes a much wider distribution to capture more information at the beginning stages while the distribution tends to be more centralized progressively to output stable final results.

Model	Zero-shot Acc. ↑								Perplexity ↓		
	BoolQ	PIQA	HS	WG	ARC-e	ARC-c	OBQA	Avg.	Wiki2	PTB	C4
Mamba-2-780M	61.5	71.8	54.9	60.2	54.3	28.5	36.2	52.5	11.8	20.0	16.5
`Bi-Mamba` (No KD)	50.5	65.8	37.8	50.9	39.7	23.8	30.4	42.7	14.9	30.9	15.6
`Bi-Mamba` (KL-Div)	56.8	66.5	38.1	51.6	39.8	22.7	28.2	43.3	15.0	27.3	15.6
`Bi-Mamba` (Phi-3.5)	49.6	66.8	39.0	53.2	40.6	23.7	30.8	43.4	14.5	27.3	16.6
`Bi-Mamba`	58.5	68.0	41.6	52.0	42.4	24.3	30.6	45.3	13.4	32.4	14.5

Table 7: Ablation study of `Bi-Mamba`. This table includes the performance comparison of different teachers and knowledge distillation strategies. Our autoregressive knowledge distillation brings improvement to the binarization-aware training regardless of the choice of teachers.

5.6 Ablation Study

In this section, we provide the ablation study of `Bi-Mamba`. As shown in Table 7, we conduct various ablation studies in the 780M model including the model without knowledge distillation (w/o KD in the table), the model utilizing the KL divergence loss as distillation loss (KL Div in the table) and the model using a different teacher model (Phi-3.5-instruct-mini (Abdin et al., 2024)). The results demonstrate the importance of each component, as removing knowledge distillation (KD) or using alternate loss functions (KL Div and Phi-3.5) significantly reduces performance compared to the full Bi-Mamba model, which achieves the best results across all metrics except PTB. Specifically, Bi-mamba trained with original autoregressive loss obtains the lowest performance compared with other models trained with KD. The average accuracy on downstream tasks is 42.7, which is surpassed by the model trained with KL-Div loss, namely 43.3. With a different teacher, Phi-3.5, `Bi-Mamba` achieves similar performance as the model trained with Llama-2-7B as the teacher, demonstrating the effectiveness of our proposed autoregressive knowledge distillation.

6 Conclusion

We introduce `Bi-Mamba`, a scalable and efficient 1-bit Mamba architecture designed for large language models in multiple sizes: 780M, 1.3B, and 2.7B parameters. We begin by identifying the binarization space within the Mamba architecture. Then, `Bi-Mamba` models are trained from scratch on large datasets, similar to standard LLM pretraining, using an autoregressive distillation loss. Extensive language modeling experiments show that `Bi-Mamba` achieves competitive performance that is slightly lower than its full-precision counterparts (e.g., FP16 or BF16), while substantially reduces memory usage and computational cost compared to the original precision Mamba. This study provided a novel, first accessible and low-bit framework with linear computational complexity, laying the foundation for developing specialized hardware optimized for efficient 1-bit Mamba-based LLMs.

Limitations and Ethical Statements

While `Bi-Mamba` achieves competitive performance to full-precision models, there may still be trade-offs in accuracy, particularly in complex tasks that rely heavily on nuanced language understanding. Also, full deployment may require specialized hardware to maximize efficiency gains, limiting accessibility on standard hardware setups. On ethical part, reducing model precision could risk oversimplifying nuanced patterns in data, potentially amplifying biases present in the training data. Moreover, while `Bi-Mamba` reduces energy consumption during inference, training binary models from scratch can still be computationally intensive.

References

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Milad Alizadeh, Javier Fernández-Marqués, Nicholas D. Lane, and Yarin Gal. A systematic study of binary neural networks' optimisation. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=rJfUCoR5KX`.

Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksandra Prudnikova, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. xlstm: Extended long short-term memory, 2024. URL `https://arxiv.org/abs/`.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher M. De Sa. Quip: 2-bit quantization of large language models with guarantees. In *NeurIPS*, 2023.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. *ArXiv*, abs/1905.10044, 2019. URL `https://api.semanticscholar.org/CorpusID:165163607`.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *ArXiv*, abs/1803.05457, 2018. URL `https://api.semanticscholar.org/CorpusID:3922816`.

Together Computer. Redpajama: An open source recipe to reproduce llama training dataset, 2023. URL `https://github.com/togethercomputer/RedPajama-Data`.

Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality, 2024. URL `https://arxiv.org/abs/2405.21060`.

Soham De, Samuel L. Smith, Anushan Fernando, Aleksandar Botev, and et al. Griffin: Mixing gated linear recurrences with local attention for efficient language models, 2024. URL `https://arxiv.org/abs/2402.19427`.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. LLM.int8(): 8-bit matrix multiplication for transformers at scale. In *NeurIPS*, 2022a.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Llm.int8(): 8-bit matrix multiplication for transformers at scale, 2022b. URL `https://arxiv.org/abs/2208.07339`.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June*

2-7, 2019, Volume 1 (Long and Short Papers), pp. 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/V1/N19-1423. URL https://doi.org/10.18653/v1/n19-1423.

Vage Egiazarian, Andrei Panferov, Denis Kuznedelev, Elias Frantar, Artem Babenko, and Dan Alistarh. Extreme compression of large language models via additive quantization, 2024. URL https://arxiv.org/abs/2401.06118.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers, 2023. URL https://arxiv.org/abs/2210.17323.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, 07 2024. URL https://zenodo.org/records/12608602.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces, 2024. URL https://arxiv.org/abs/2312.00752.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces, 2021a. URL https://arxiv.org/abs/2111.00396.

Albert Gu, Isys Johnson, Karan Goel, Khaled Saab, Tri Dao, Atri Rudra, and Christopher Ré. Combining recurrent, convolutional, and continuous-time models with linear state space layers. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 572–585, 2021b. URL https://proceedings.neurips.cc/paper/2021/hash/05546b0e38ab9175cd905eebcc6ebb76-Abstract.html.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=uYLFoz1vlAC.

Wei Huang, Yangdong Liu, Haotong Qin, Ying Li, Shiming Zhang, Xianglong Liu, Michele Magno, and Xiaojuan Qi. Billm: Pushing the limit of post-training quantization for llms, 2024. URL https://arxiv.org/abs/2402.04291.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention, 2020. URL https://arxiv.org/abs/2006.16236.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles, SOSP 2023, Koblenz, Germany, October 23-26, 2023*, pp. 611–626. ACM, 2023. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, et al. Starcoder: may the source be with you! *arXiv preprint arXiv:2305.06161*, 2023a.

Yixiao Li, Yifan Yu, Chen Liang, Pengcheng He, Nikos Karampatziakis, Weizhu Chen, and Tuo Zhao. Loftq: Lora-fine-tuning-aware quantization for large language models, 2023b. URL https://arxiv.org/abs/2310.08659.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for llm compression and acceleration. *arXiv preprint arXiv:2306.00978*, 2023.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100, 2024.

Zhengzhong Liu, Aurick Qiao, Willie Neiswanger, Hongyi Wang, Bowen Tan, Tianhua Tao, Junbo Li, Yuqi Wang, Suqi Sun, Omkar Pangarkar, et al. Llm360: Towards fully transparent open-source llms. *arXiv preprint arXiv:2312.06550*, 2023.

Liqun Ma, Mingjie Sun, and Zhiqiang Shen. Fbi-llm: Scaling up fully binarized llms from scratch via autoregressive distillation, 2024a. URL https://arxiv.org/abs/2407.07093.

Shuming Ma, Hongyu Wang, Lingxiao Ma, Lei Wang, Wenhui Wang, Shaohan Huang, Li Dong, Ruiping Wang, Jilong Xue, and Furu Wei. The era of 1-bit llms: All large language models are in 1.58 bits. *arXiv preprint arXiv:2402.17764*, 2024b.

Shuming Ma, Hongyu Wang, Lingxiao Ma, Lei Wang, Wenhui Wang, Shaohan Huang, Li Dong, Ruiping Wang, Jilong Xue, and Furu Wei. The era of 1-bit llms: All large language models are in 1.58 bits, 2024c. URL https://arxiv.org/abs/2402.17764.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. Llm-pruner: On the structural pruning of large language models, 2023. URL https://arxiv.org/abs/2305.11627.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993. URL https://aclanthology.org/J93-2004.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Conference on Empirical Methods in Natural Language Processing*, 2018. URL https://api.semanticscholar.org/CorpusID:52183757.

OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The refinedweb dataset for falcon llm: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*, 2023.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, and et al. Rwkv: Reinventing rnns for the transformer era, 2023. URL https://arxiv.org/abs/2305.13048.

Alessandro Pierro and Steven Abreu. Mamba-ptq: Outlier channels in recurrent large language models, 2024. URL https://arxiv.org/abs/2407.12397v1.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. In *Proceedings of the Sixth Conference on Machine Learning and Systems, MLSys 2023, Miami, FL, USA, June 4-8, 2023*. mlsys.org, 2023. URL https://proceedings.mlsys.org/paper_files/paper/2023/hash/c4be71ab8d24cdfb45e3d06dbfca2780-Abstract-mlsys2023.html.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European conference on computer vision*, pp. 525–542. Springer, 2016.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Yuzhang Shang, Zhihang Yuan, Qiang Wu, and Zhen Dong. Pb-llm: Partially binarized large language models. *arXiv preprint arXiv:2310.00034*, 2023.

Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. Omniquant: Omnidirectionally calibrated quantization for large language models, 2023. URL https://arxiv.org/abs/2308.13137.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023. URL https://arxiv.org/abs/2302.13971.

Albert Tseng, Jerry Chee, Qingyao Sun, Volodymyr Kuleshov, and Christopher De Sa. Quip#: Even better llm quantization with hadamard incoherence and lattice codebooks, 2024. URL https://arxiv.org/abs/2402.04396.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 5998–6008, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Xiuying Wei, Yunchen Zhang, Xiangguo Zhang, Ruihao Gong, Shanghang Zhang, Qi Zhang, Fengwei Yu, and Xianglong Liu. Outlier suppression: Pushing the limit of low-bit transformer language models. In *NeurIPS*, 2022.

Mitchell Wortsman, Tim Dettmers, Luke Zettlemoyer, Ari Morcos, Ali Farhadi, and Ludwig Schmidt. Stable and low-precision training for large-scale vision-language models, 2023. URL https://arxiv.org/abs/2304.13013.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pp. 38087–38099. PMLR, 2023.

Yuzhuang Xu, Xu Han, Zonghan Yang, Shuo Wang, Qingfu Zhu, Zhiyuan Liu, Weidong Liu, and Wanxiang Che. Onebit: Towards extremely low-bit large language models. *arXiv preprint arXiv:2402.11295*, 2024.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. In *Advances in Neural Information Processing Systems*, 2022.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Annual Meeting of the Association for Computational Linguistics*, 2019. URL https://api.semanticscholar.org/CorpusID:159041722.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. Tinyllama: An open-source small language model. *arXiv preprint arXiv:2401.02385*, 2024.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.