MAMBA and Transformers fine-tune on Question answering with context

Hieu Minh Nguyen
22520440@gm.uit.edu.vn

January 11, 2025

Abstract

In this experiment, we apply the MAMBA language model to predict answers based on questions and context. We employ the prompting method to address the issue of capturing context and to improve model performance. Additionally, we compare MAMBA's performance with that of a language model based on Transformers.

1 Introduction

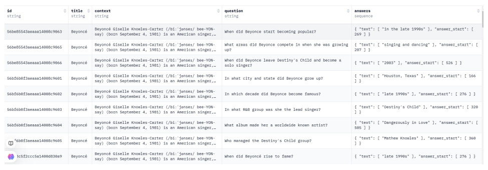
Foundation models (FMs), or large models pretrained on massive data then adapted for downstream tasks, have emerged as an effective paradigm in modern machine learning. The backbone of these FMs are often sequence models, operating on arbitrary sequences of inputs from a wide variety of domains such as language, images, speech, audio, time series, and genomics [2],[3],[5],[7],[8],[10] . While this concept is agnostic to a particular choice of model architecture, modern FMs are predominantly based on a single type of sequence model: the Transformer[12] and its core attention layer[1].The efficacy of self-attention is attributed to its ability to route information densely within a context window, allowing it to model complex data. However, this property brings fundamental drawbacks: an inability to model anything outside of a finite window, and quadratic scaling with respect to the window length. An enormous body of research has appeared on more efficient variants of attention to overcome these drawbacks [11], but often at the expense of the very properties that makes it effective. As of yet, none of these variants have been shown to be empirically effective at scale across domains. Recognizing these limitations, Tridao et al developed a selective state-space model named MAMBA[4], which advances prior work across several dimensions to achieve the modeling capabilities of Transformers while maintaining linear scaling with sequence length.And in this report , we will use Mamba for Question Answering with context to evaluate model performance.

2 Dataset

2.1 overview

In this study, we selected the SQuAD 2.0 dataset [9](Stanford Question Answering Dataset), which was constructed from carefully curated and reliable textual passages from diverse sources. This dataset, developed by researchers at Stanford University, serves as a benchmark for question answering tasks, providing a collection of questions and answers based on context. SQuAD 2.0 comprises over 150,000 question-answer pairs, with its primary distinction from SQuAD 1.1 being the inclusion of unanswerable questions.

SQuAD 2.0 not only challenges models to extract appropriate answers from passages but also requires the ability to detect when a question cannot be answered. Compared to traditional pre-processed datasets like the Penn Treebank (PTB) [6], SQuAD 2.0 is more practical and retains complete contextual integrity, including special characters, punctuation, and complex textual structures.



id string	title string	context string	question string	answers sequence
56be85543aeaaa14008c9063	Beyoncé	Beyoncé Giselle Knowles-Carter (/biːˈjɒnseɪ/ bee-YON-say) (born September 4, 1981) is an American singer,…	When did Beyonce start becoming popular?	{ "text": ["in the late 1990s"], "answer_start": [269] }
56be85543aeaaa14008c9065	Beyoncé	Beyoncé Giselle Knowles-Carter (/biːˈjɒnseɪ/ bee-YON-say) (born September 4, 1981) is an American singer,…	What areas did Beyonce compete in when she was growing up?	{ "text": ["singing and dancing"], "answer_start": [207] }
56be85543aeaaa14008c9066	Beyoncé	Beyoncé Giselle Knowles-Carter (/biːˈjɒnseɪ/ bee-YON-say) (born September 4, 1981) is an American singer,…	When did Beyonce leave Destiny's Child and become a solo singer?	{ "text": ["2003"], "answer_start": [526] }
56bf6b0f3aeaaa14008c9601	Beyoncé	Beyoncé Giselle Knowles-Carter (/biːˈjɒnseɪ/ bee-YON-say) (born September 4, 1981) is an American singer,…	In what city and state did Beyonce grow up?	{ "text": ["Houston, Texas"], "answer_start": [166] }
56bf6b0f3aeaaa14008c9602	Beyoncé	Beyoncé Giselle Knowles-Carter (/biːˈjɒnseɪ/ bee-YON-say) (born September 4, 1981) is an American singer,…	In which decade did Beyonce become famous?	{ "text": ["late 1990s"], "answer_start": [276] }
56bf6b0f3aeaaa14008c9603	Beyoncé	Beyoncé Giselle Knowles-Carter (/biːˈjɒnseɪ/ bee-YON-say) (born September 4, 1981) is an American singer,…	In what R&B group was she the lead singer?	{ "text": ["Destiny's Child"], "answer_start": [320] }
56bf6b0f3aeaaa14008c9604	Beyoncé	Beyoncé Giselle Knowles-Carter (/biːˈjɒnseɪ/ bee-YON-say) (born September 4, 1981) is an American singer,…	What album made her a worldwide known artist?	{ "text": ["Dangerously in Love"], "answer_start": [505] }
6b0f3aeaaa14008c9605	Beyoncé	Beyoncé Giselle Knowles-Carter (/biːˈjɒnseɪ/ bee-YON-say) (born September 4, 1981) is an American singer,…	Who managed the Destiny's Child group?	{ "text": ["Mathew Knowles"], "answer_start": [360] }
3c5f2ccc5a1400d830a9	Beyoncé	Beyoncé Giselle Knowles-Carter (/biːˈjɒnseɪ/ bee-YON-say) (born September 4, 1981) is an American singer,…	When did Beyoncé rise to fame?	{ "text": ["late 1990s"], "answer_start": [276] }

Figure 1: Structures of dataset Squad 2.0

2.2 Analysis of the Squad 2.0 dataset

2.2.1 Dataset Composition:

Over 150,000 question-answer pairs.
Context: Extracted from Wikipedia articles.
Questions: Some have answers present within the context, while others are explicitly unanswerable.
Answers:

- If answerable: Extracted spans of text from the context.
- If unanswerable: Labeled as "unanswerable."

Data Distribution:

- Answerable questions: Approximately 66%.
- Unanswerable questions: Approximately 34%.

Notable Characteristics:

- Context passages are often lengthy, spanning multiple sentences or paragraphs.
- Answers typically consist of 3-5 words, often noun phrases or short clauses.

	samples
Train	130000
Validation	11900
Test	1000

Table 1: Data split

3 Method

3.1 Mamba model

Mamba is a novel neural network architecture designed for efficient sequence processing, especially for long sequences. It is considered a potential alternative to Transformer architectures in various applications. Mamba is built upon the ideas of State Space Models (SSMs) but incorporates significant enhancements.

State-space Models (S4)

Structured state-space sequence (S4) models are a type of State-Space Model (SSM) that leverages linear time-invariant (LTI) systems to combine the computational advantages of Transformers with those of recurrent neural networks (RNNs). Specifically, S4 models aim to achieve highly parallelizable training and efficient autoregressive inference using recurrence.

Within the S4 layer, an input signal is first discretized, and latent dynamics are learned through the LTI parameters. The crucial aspect of S4 is that its latent dynamics can be compactly represented as a single convolution between the input and an SSM convolution kernel. This convolution kernel is actually a matrix whose entries are products of LTI learnable parameters. This formulation allows the model to be efficiently unrolled, avoiding the need to explicitly compute the state space equations at every step.

This approach allows for hardware efficiency and improved long-dependency modeling, and these LTI-based S4 models achieved similar performance to Transformers with comparable parameter-sizes for natural language tasks, even when augmenting them with attention layers for hybrid architectures. However, they are limited by the LTI assumption.

General Architecture

Innovating on these previous S4 approaches, Mamba utilizes time-varying parameters to model latent dynamics, thus broadening the ability to capture nuanced changes evolving in discrete-time. Without the time invariance of LTI dynamics, the input-to-output mapping can no longer be expressed as a convolution using an SSM convolution kernel, thus voiding hardware optimizations originally introduced with S4 models. Mamba introduces extensive use of customized CUDA kernels, implementing highly parallelized prefix sums to compute recurrent states, in order to enable hardware efficiency with time-varying parameters.

1. **Input:** Mamba receives an input sequence $x = (x_1, x_2, ..., x_L)$, where x_i is the input vector at time step i.

2. **RMS Normalization:** The input is first passed through an RMS Normalization layer, which stabilizes training by normalizing the input features, and preventing exploding gradient problems.

3. **Projection Layers:** Following the normalization, the input is then passed through two separate projection layers.

    - **Purpose:** These projection layers serve to transform the input into different representations that are suitable for the following operations. They increase the dimensionality of the input in order to provide more information for subsequent layers.
    - **Linear Transformations:** Each projection layer typically involves a linear transformation (matrix multiplication), potentially followed by a non-linear activation function, though this is not shown explicitly in the diagram, it is implied that these are linear projection.
    - These projections are often parameterized by different weights, so information that has gone through each branch are transformed differently.

4. **Convolution:** The input that has gone through one of the projection layers is sent into a one dimensional convolution layer.

5. **State Space Model (SSM):**

   For model dimension d and maximum input sequence length T, the MambaBlock defines state-space parameters $\mathbf{A}, \mathbf{B}, \mathbf{C}, \Delta_t \in \mathbb{R}^{d \times d}$ for $t \in \{1, \ldots, T\}$. The matrix Δ_t controls the discrete step-size. Given an input sequence $u_1, \ldots, u_T \in \mathbb{R}^d$, the following linear mapping through latent states $x_1, \ldots, x_T \in \mathbb{R}^d$ is used to produce the output $y_1, \ldots, y_T \in \mathbb{R}^d$:

   $$\mathbf{x}_t = \bar{\mathbf{A}}_t \mathbf{x}_{t-1} + \bar{\mathbf{B}}_t \mathbf{u}_t \tag{1}$$
   $$\mathbf{y}_t = \mathbf{C}_t \mathbf{x}_t \tag{2}$$

   where

   $$\begin{aligned} \Delta_t &= \text{softplus}(\text{Linear}(\Delta_t)) \in \mathbb{R}^{d \times d} \\ \bar{\mathbf{A}}_t &= \exp(\Delta_t \mathbf{A}) \\ \bar{\mathbf{B}}_t &= \Delta_t^{-1}(\exp(\Delta_t \mathbf{A}) - \mathbf{I})\mathbf{B} \end{aligned}$$

   In practice, $\mathbf{A}, \mathbf{B}, \mathbf{C}$, and Δ are diagonal matrices.

6. **Selection Mechanism:** Mamba uses a special selection mechanism to determine which parts of the input sequence should be emphasized during computation. This helps the model to reduce computational load and improve efficiency when handling long sequences.

7. **Combination:** The output from the selective SSM block and the projection layer is combined using an element wise multiplication or activation depending on implementation.

8. **Projection Layer:** The aggregated information is subsequently transformed into the desired output dimensionality through a projection layer, which maps it into a different space to align with the target output size.

9. **Addition:** The output of the projection layer and the input from the initial RMS Norm layer via a skip connection are then added together using element-wise addition.

10. **RMS Normalization:** The output of addition is passed through another RMS Normalization layer.

11. **Linear + Softmax:** Finally, the output is passed through a linear layer followed by a softmax activation function to generate the final output.

12. **Output:** The final output of the mamba block is obtained. This Mamba block is repeated n times in practice.

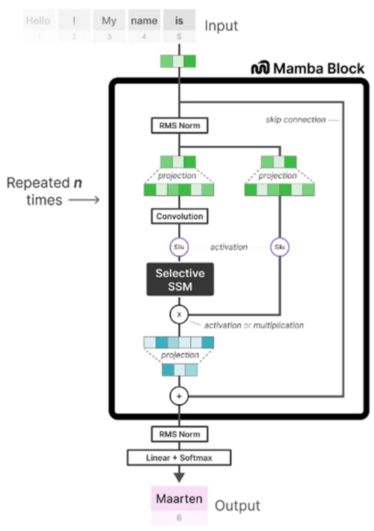


Figure 2: Mamba model architecture

3.1.1 Hardware-Aware Algorithm

A disadvantage of recent GPUs is their limited transfer (IO) speed between their small but highly efficient SRAM and their large but slightly less efficient DRAM. Frequently copying information between SRAM and DRAM becomes a bottleneck.

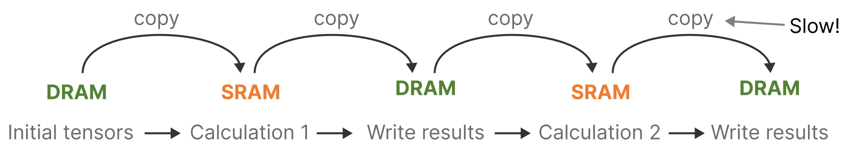


Figure 3: Frequently switching from SRAM and DRAM caused bottleneck

MAMBA, attempts to limit the number of times we need to go from DRAM to SRAM and vice versa. It does so through kernel fusion which allows the model to prevent writing intermediate results and continuously performing computations until it is done.

The intermediate states are not saved but are necessary for the backward pass to compute the gradients. Instead, the authors recompute those intermediate states during the backward pass.

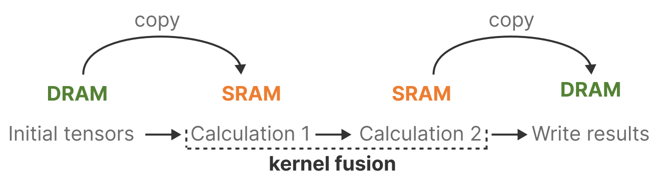


Figure 4: Kernel Fusion reduce the number of switching SRAM and DRAM

3.1.2 Parameters Initialization

Most prior SSMs also suggest special initializations, particularly in the complex-valued case, which can help in several settings such as low-data regimes. The initialization stage of this model is based on HiPPO theory [**hippo**]. The theory describes the difficulty of RNN while capturing long dependencies. In their research, the authors suggested a solution by projecting continuous signals or discrete time series onto orthogonal polynomial bases, providing an efficient way to compress and represent historical information in a compact form. for handling long-range dependencies. These define the initialization n-th element of $\mathbf{A}$ as $-\frac{1}{2} + n$ and $-(n+1)$ for the complex and real numerical part of the element, respectively. The Δ is initialized to uniform$([0.001, 0.1])$.

Advantages of Mamba

- **Efficient Handling of Long Sequences:** Mamba can process long sequences more efficiently than traditional Transformer models.
- **Fast Computation:** Mamba can be computed in parallel, improving processing speed.
- **Resource Efficiency:** Mamba typically requires fewer computational resources than similar Transformer models.
- **Flexibility:** It can be applied to various tasks such as natural language processing, computer vision, and time-series prediction.

Footnotes are inserted with the `\footnote` command.[1]

Prompting Methods

- **Basic Prompting:**

[1]This is a footnote.

  - A simple prompt, such as "You are a trivia bot. Answer the following trivia question.", is used to initiate a question-answering behavior.
  - This initial prompt is used to evaluate the performance of the raw, pre-trained language model.

- **N-Shot Prompting:**
  - Multiple question-answer pairs are included in the prompt, demonstrating the desired behavior.
  - This aims to guide the model to generalize to similar questions and answer them correctly by following the provided pattern.
  - Example Question Answer Pair Prompt:

```
You are a Trivia QA bot.
Answer the following question
succinctly and accurately.
Q: What is the capital of France?
A: Paris
Q: Who invented the segway?
A: Dean Kamen
Q: What is the fastest animal?
A: Cheetah
Q: {user_input}
A: {model_response}
```

- **Contextual Prompting:**
  - The prompt provides a context, a question based on the context, and an expected answer, or the explicit option to state "I don't know".
  - This is designed to extract information and answers from a given text passage.

Evaluate

We use Exact Match (EM) and F1 Score for evaluate model's performance.

- **Exact Match (EM):** This metric measures whether the predicted answer matches the ground truth exactly, without any deviation in terms of wording, order, or format.

- **Formula:**
$$EM = \frac{\text{Number of exact matches}}{\text{Total number of questions}}$$

- **F1 Score:** This metric assesses the overlap of tokens between the predicted answer and the ground truth by balancing **Precision** and **Recall**.

  - **Precision:** The proportion of correct tokens in the predicted answer.
  $$\text{Precision} = \frac{\begin{array}{c}\text{Number of}\\\text{correct tokens}\end{array}}{\begin{array}{c}\text{Number of tokens}\\\text{in the predicted answer}\end{array}}$$

  - **Recall:** The proportion of correct tokens in the predicted answer compared to the ground truth.
  $$\text{Recall} = \frac{\begin{array}{c}\text{Number of}\\\text{correct tokens}\end{array}}{\begin{array}{c}\text{Number of tokens}\\\text{in the ground truth}\end{array}}$$

  - **F1 Score:** The harmonic mean of Precision and Recall.
  $$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Experiments

- **Data Loading:**
  - The model is trained with a custom data loader (‘SFTDataset‘) using a dataset formatted as context, question and answer tuples.
  - The data loader creates both positive and negative training examples by pairing a question with a relevant answer (positive), and with "I don’t know" when paired with a random, irrelevant context (negative).
  - This method aims to improve the Mamba model’s ability to either extract relevant answers when it has access to correct context or reject incorrect contexts.

- **Fine-Tuning Process:**

- The pre-trained Mamba model is fine-tuned on the custom dataset that is outputted by the SFTDataset dataloader.
- The fine-tuning is for a set number of epochs, in this case it was set to 10 epochs.

3.2 Result

Model	**F1**	**EXACT MATCH**
mamba-130m-finetune	0.2217	**12.1%**
mamba-2.8b-finetune	**0.2357**	11.5%
mamba-2.8b	0.0157	0.07%
Roberta-base	0.7115	74.67%

The results table demonstrates that the prompting and fine-tuning methods yield better performance compared to the base model. However, their performance remains significantly lower than that achieved by models utilizing Transformers.

4 Disscussions

Mamba has made a breakthrough by removing activation functions, enabling RNNs to train in parallel. However, due to the absence of an attention mechanism for storing more information, Mamba compensates by increasing the dimension of the feature vector. The lack of attention also limits Mamba's performance on extremely long contexts. To address this limitation, the authors reintroduced the attention mechanism and combined it with SSM in Mamba2.

In the future, given that this is an extractive question answering task, we believe that the prompting method may not be the most suitable approach. Instead, we plan to extract features from the context and subsequently fine-tune the Mamba model using these features.

References

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. *Neural Machine Translation by Jointly Learning to Align and Translate*. 2016. arXiv: 1409.0473 [cs.CL]. URL: https://arxiv.org/abs/1409.0473.

[2] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. "Language Models are Few-Shot Learners". In: *CoRR* abs/2005.14165 (2020). arXiv: 2005.14165. URL: https://arxiv.org/abs/2005.14165.

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. arXiv: 2010.11929 [cs.CV]. URL: https://arxiv.org/abs/2010.11929.

[4] Albert Gu and Tri Dao. *Mamba: Linear-Time Sequence Modeling with Selective State Spaces*. 2024. arXiv: 2312.00752 [cs.LG]. URL: https://arxiv.org/abs/2312.00752.

[5] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. "Deep learning for time series classification: a review". In: *Data Mining and Knowledge Discovery* 33.4 (Mar. 2019), 917–963. ISSN: 1573-756X. DOI: 10.1007/s10618-019-00619-1. URL: http://dx.doi.org/10.1007/s10618-019-00619-1.

[6] Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. "Building a Large Annotated Corpus of English: The Penn Treebank". In: *Computational Linguistics* 19.2 (1993). Ed. by Julia Hirschberg, pp. 313–330. URL: https://aclanthology.org/J93-2004/.

[7] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. *WaveNet: A Generative Model for Raw Audio*. 2016. arXiv: 1609.03499 [cs.SD]. URL: https://arxiv.org/abs/1609.03499.

[8] Michael Poli, Stefano Massaroli, Eric Nguyen, Daniel Y. Fu, Tri Dao, Stephen Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Ré. *Hyena Hierarchy: Towards Larger Convolutional Language Models*. 2023. arXiv: 2302.10866 [cs.LG]. URL: https://arxiv.org/abs/2302.10866.

[9] Pranav Rajpurkar, Robin Jia, and Percy Liang. *Know What You Don't Know: Unanswerable Questions for SQuAD*. 2018. arXiv: 1806.03822 [cs.CL]. URL: https://arxiv.org/abs/1806.03822.

[10] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. *Sequence to Sequence Learning with Neural Networks*. 2014. arXiv: 1409.3215 [cs.CL]. URL: https://arxiv.org/abs/1409.3215.

[11] Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. *Efficient Transformers: A Survey*. 2022. arXiv: 2009.06732 [cs.LG]. URL: https://arxiv.org/abs/2009.06732.

[12] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL]. URL: https://arxiv.org/abs/1706.03762.