Re-Examine Hybrid State-Space and Self-Attention for In-Context Learning

Jingze Shi [*1] **Bingheng Wu** [*1]

Abstract

Recent research has shown that combining the state-space algorithm-driven Mamba with the self-attention algorithm-driven Transformer outperforms using Mamba or Transformer alone in most language modeling tasks. However, the performance of the mainstream hybrid modeling architecture models in in-context learning tasks is not ideal. We re-examine the advantages and disadvantages of these two algorithms and redesign the structure of this hybrid modeling from the principle. The finally redesigned architecture improves the performance by 1.3% in standard short text tasks, 20.86% in natural long text tasks, and 27.06% in synthetic long text tasks.

1. Introduction

The self-attention algorithm of the Transformers (Wolf et al., 2020) architecture can directly capture the relationship between any two elements in a sequence, effectively handle long-distance dependencies. However, it is limited by quadratic complexity. The state-space algorithm of the Mamba (Gu & Dao, 2023) architecture can achieve linear scaling of sequence length during training and maintain a constant state size during generation, but it leads to bias in capturing long-distance dependencies. Hybrid modeling architecture models, such as Wonderful Matrices (Shi & Wu, 2024), Jamba (Lieber et al., 2024), etc., use state-space and self-attention for hybrid modeling, making the model have efficiency similar to The Mamba and effect similar to The Transformer. However, these models still have a significant gap in performance in in-context learning tasks compared to the original Transformer.

We propose Self-Attention before LM-Head 1, which is a simple change to the existing hybrid stacked architecture models, modifying the state-space and self-attention to use the same positional encoding, and using a Transformer block composed of self-attention and feed-forward networks before the LM-Head predicts the probability distribution. This method allows the model to continue to leverage the advantages of the efficient context summary of the state-space and the effective associative recall of self-attention without bias in the final token prediction.

Our research and evaluation show that Self-Attention before LM-Head can achieve better performance on the in-context learning task benchmark compared to the baseline hybrid model with only a few structural and parameter adjustments. For example, in standard short text tasks, our model improves performance by 1.3%, in natural long text tasks by 20.86%, in synthetic long text tasks by 27.06%, and achieves state-of-the-art performance on the needle in a haystack task.

2. Background

2.1. State-Space

In the field of natural language processing (NLP), SSM is used to model the dynamic characteristics of text sequences. Its advantage is that it can efficiently handle long sequence data and capture the time dependency in the sequence through the update mechanism of the state variables. However, traditional SSM has some limitations, such as the matrix parameters (such as A, B, C) of the linear time-invariant (LTI) system remain fixed throughout the entire sequence generation process, making it difficult for the model to perform content-aware reasoning. For example, when dealing with tasks that require selective attention or ignoring specific inputs, the performance of traditional SSM is not as good as the Transformer model based on the self-attention mechanism. State-space models are mathematical models used to describe the dynamic behavior of systems, and their basic form is:

$$\begin{aligned} h'(t) &= Ah(t) + Bx(t) \\ y(t) &= Ch(t) + Dx(t) \end{aligned} \tag{1}$$

where $h(t)$ is the state vector, $x(t)$ is the input vector, $y(t)$ is the output vector, and A, B, C, D are system matrices.

$$\begin{aligned} h_t &= Ah_{t-1} + Bx_t \\ y_t &= Ch_t \end{aligned} \tag{2}$$

Starting from the state-space model of continuous-time systems, the discrete-time state update formula can be obtained through discretization methods. Equations (2a) and (2b) represent the discrete form of the state-space model. They describe how the state vector and output vector are updated

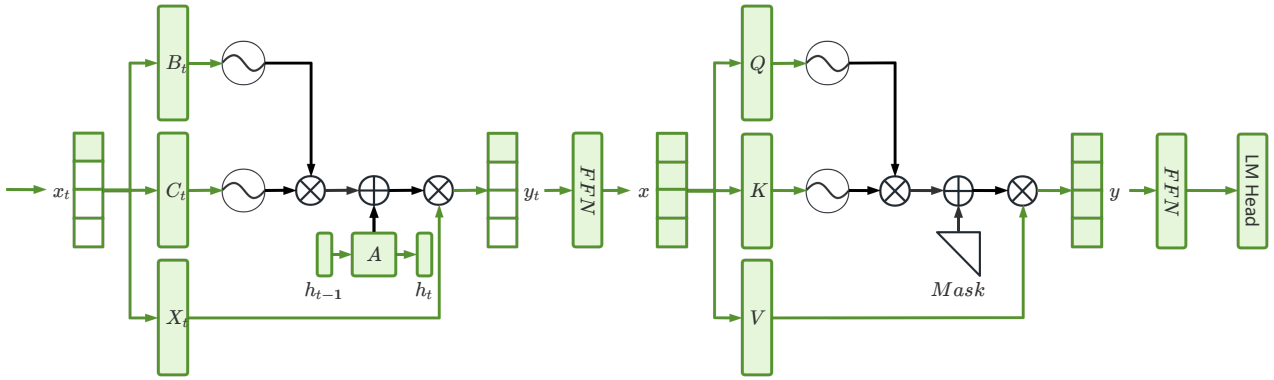

Figure 1. **Self-Attention before LM-Head**. The state-space and self-attention both use the same positional encoding, and a Transformer block composed of self-attention and feed-forward networks is used before the LM-Head, regardless of how the other parts of the model backbone are combined.

based on the previous state and current input at discrete time intervals.

The above is the process of SSM-S4. All structured SSMs are LTI models, and LTI models have basic efficiency limitations. To overcome these limitations, researchers have proposed various improvement methods. For example, the Mamba (**?**) model introduces the Selective Scan algorithm and the Hardware-Aware Algorithm, allowing SSM to dynamically adjust parameters to better handle long sequence data and improve computational efficiency. This improved SSM, also known as Selective State Space Model (Selective SSM, referred to as S6), aims to build efficient sequence processing modules like Transformer. However, even so, it is still not as efficient as CNN and Transformer in computation, and the development of modern parallel computing devices makes this computational efficiency gap more obvious, thereby limiting the training of SSM on larger-scale sequence data. Therefore, it is very important to reveal the deeper relationship between Selective SSM and the attention mechanism and propose a new hybrid modeling architecture to overcome these limitations.

2.2. Self Attention

In the Transformer model, the self-attention mechanism determines the importance weights of each position by calculating the similarity between queries, keys, and values. Specifically, the self-attention mechanism first calculates the dot product between the query and the key, then converts these dot products into attention weights through the softmax function, and finally applies these weights to the value vector to generate a weighted context representation.

$$Y = \text{softmax}(QK^\top) \cdot V \tag{3}$$

Through the self-attention mechanism, the Transformer model can directly capture the relationship between any two elements in the sequence, effectively handling long-distance dependency problems. However, the quadratic complexity of the self-attention mechanism limits the computational efficiency of the Transformer model, making it perform poorly when handling long sequence data. To overcome this limitation, researchers have proposed many improvement methods. For example, Lin et al. (**?**) proposed a self-attention mechanism based on local sensitive hashing, which reduces the computational cost of dot products between queries and keys, thereby improving the computational efficiency of the Transformer model. Similarly, this improvement method still cannot completely solve the computational efficiency problem of the Transformer model. Therefore, it is very important to reveal the deeper relationship between the self-attention mechanism and the state space and propose a new hybrid modeling architecture to overcome these limitations.

2.3. Hybrid Modeling

Hybrid modeling architecture models are models that combine state-space and self-attention for hybrid modeling, with the goal of making the model have efficiency similar to Mamba and effect similar to Transformer. For example, the Wonderful Matrices (Shi & Wu, 2024) model combines state-space and self-attention for hybrid modeling, allowing the model to leverage the advantages of the efficient context summary of the state-space and the effective associative recall of self-attention. However, these models still do not perform as well as the original Transformer model in in-context learning tasks, which indicates the limitations of

hybrid modeling architecture models in in-context learning tasks.

3. Method

In the past hybrid modeling architecture models, a one-dimensional convolution in the Mamba (Gu & Dao, 2023) block is used to implicitly perform positional encoding, while the rotational positional encoding used by self-attention is removed. The rotational positional encoding does not directly operate on the attention score matrix. For good extrapolation performance, we can add it to the state space. Let's review the state-space algorithm **??**, first, the matrix B controls whether to let the input x_t enter the state h_t, we add rotational positional encoding to the matrix B, the matrix B as an input gate filters x_t after h_t has absolute positional information, then, the matrix A controls whether to let the state h_{t-1} enter the state h_t, that is, it also allows the absolute positional information of the past state to affect the current state, finally, the matrix C controls whether to let the state h_t enter the output y_t, we add rotational positional encoding to the matrix C, the matrix C as an output gate filters h_t after y_t has relative positional information. However, it is worth noting that due to the recursive nature of the state space, that is, only one state cache is retained, the positional encoding will be affected by the superposition state, even the original one-dimensional convolution positional encoding is the same, which is also its difficulty in context learning defect. So using the position information processed by the state space directly as the positional encoding of self-attention will inevitably lead to the confusion of position information. To solve this problem, we should retain the rotational positional encoding of self-attention and provide the original position information for self-attention.

The computational complexity of the state space grows linearly with the sequence length T, and its data-dependent related parameters can enable the model to have the ability to selectively process information. The linear recursion gives the aggregated state that allows us to calculate only the last token. Of course, by approximating the function to generate the optimal solution of the state matrix A to remember history, it can only accurately capture the state change from h_{t-1} to h_t and gradually decay the old state information, which leads to the need for strong replication, context learning, or long-context reasoning capabilities. In tasks that require powerful capabilities, before predicting the token, the information summary of the state space will produce bias. To solve this problem, we should use self-attention and feed-forward networks that do not compress context information before the LM-Head, which can produce context-aware states by weighting elements without distance restrictions, allowing the model to continue to leverage the efficient context summary of the state space and the effective associative recall of self-attention without bias in the final token prediction.

References

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

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

Shi, J. and Wu, B. Wonderful matrices: Combining for a more efficient and effective foundation model architecture, 2024. URL https://arxiv.org/abs/2412.11834.

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

A. You *can* have an appendix here.

You can have as much text here as you want. The main body must be at most 8 pages long. For the final version, one more page can be added. If you want, you can use an appendix like this one.

The `\onecolumn` command above can be kept in place if you prefer a one-column appendix, or can be removed if you prefer a two-column appendix. Apart from this possible change, the style (font size, spacing, margins, page numbering, etc.) should be kept the same as the main body.