MambaMixer: Efficient Selective State Space Models with Dual Token and Channel Selection

Ali Behrouz
Cornell University
ab2947@cornell.edu

Michele Santacatterina
NYU Grossman School of Medicine
santam13@nyu.edu

Ramin Zabih
Cornell University
rdz@cs.cornell.edu

Project Page (Code & Models)

Abstract

Recent advances in deep learning have mainly relied on Transformers due to their data dependency and ability to learn at scale. The attention module in these architectures, however, exhibit quadratic time and space in input size, limiting their scalability for long-sequence modeling. Despite recent attempts to design efficient and effective architecture backbone for multi-dimensional data, such as images and multivariate time series, existing models are either data independent, or fail to allow inter- and intra-dimension communication. Recently, State Space Models (SSMs), and more specifically Selective State Space Models (S6), with efficient hardware-aware implementation, have shown promising potential for long sequence modeling. Motivated by the recent success of SSMs, we present MambaMixer block, a new architecture with data dependent weights that uses a dual selection mechanism across tokens and channels–called Selective Token and Channel Mixer. MambaMixer further connects the sequential selective mixers using a weighted averaging mechanism, allowing layers to have direct access to different layers' input and output. As a proof of concept, we design Vision MambaMixer (ViM2) and Time Series MambaMixer (TSM2) architectures based on MambaMixer block and explore their performance in various vision and time series forecasting tasks. Our results underline the importance of selectively mixing across both tokens and channels. In ImageNet classification, object detection, and semantic segmentation tasks, ViM2 achieves competitive performance with well-established vision models, i.e., ViT, MLP-Mixer, ConvMixer, and outperforms SSM-based vision models, i.e., ViM and VMamba. In time series forecasting, TSM2, an attention and MLP-free architecture, achieves outstanding performance compared to state-of-the-art methods while demonstrating significantly improved computational cost. These results show that while Transformers, cross-channel attention, and cross-channel MLPs are sufficient for good performance in practice, neither is necessary.

1 Introduction

In recent years, Transformers (Vaswani et al., 2017) have been the pivotal backbone architecture behind deep learning's success, enabling a number of breakthrough advances in language modeling (Wolf et al., 2019), vision (Dosovitskiy et al., 2021), time series (Zhou et al., 2021), healthcare (Tang et al., 2023), and several other domains (Radford et al., 2023; Behrouz et al., 2023). The attention modules in Transformers are crucial for their data dependency and enable them to generalize to unseen data and tasks given the context as input. They, however, are difficult to scale efficiently to long sequences due to their quadratic time and space complexity. Breaking this quadratic computational cost is a key step towards new possibilities for deep learning such as long-range context learning (Gu & Dao, 2023), large object detection (Zhu et al., 2024), and long-range time series forecasting (Liu et al., 2021a).

To alleviate this computational complexity bottleneck, several recent studies have focused on designing sub-quadratic sequence models motivated by a diverse range of objectives: i.e., MLP-Mixer (Tolstikhin et al., 2021) and ConvMixer (Trockman & Kolter, 2023) are motivated as simpler alternatives to attention modules, MonarchMixer (M2) (Fu et al., 2023a) tackles efficiency without losing quality by using sparse Monarch matrices, and efficient attentions (Xiao et al., 2024; Kacham et al., 2023; Ding et al., 2023; Chen et al., 2021) sparsify or approximate the full attention matrix. These methods, however, either ① are based on data-independent parameters, ② introduce a trade-off between expressivity and speed, underperforming relative to Transformers when are scalable and efficient, or ③ are actually slow in practice, due to low hardware utilization (Dao et al., 2022; Chen et al., 2021).

Recently, structured State Space Models (SSMs) have emerged as a promising class of architectures for sequence modeling (Gu et al., 2022b; Fu et al., 2023b; Smith et al., 2023). SSMs can be seen as a combination of Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs), making them very efficient in training (as a CNN) and inference (as an RNN) (Gu et al., 2020). Despite their efficiency, primary general SSM architectures, e.g., S4 (Gu et al., 2022b), S4D (Gu et al., 2022a), are based on data-independent parameters, limiting their effectiveness in compressing context into a smaller state (Gu & Dao, 2023). To alleviate this limitation, recently, Gu & Dao (2023) present Mamba, a selective SSMs (S6) that effectively selects relevant context by making the SSM weights time variant (i.e., data dependent). Mamba achieves on par performance with Transformer-based state-of-the-art methods in language modeling while having less parameters and scaling near-linearly in sequence length (Gu & Dao, 2023), addressing all the three abovementioned limitations. The success of Mamba motivates several studies to adapt its design to different domains and modalities, e.g., vision (Zhu et al., 2024; Liu et al., 2024), graphs (Behrouz & Hashemi, 2024), videos (Li et al., 2024), DNA modeling (Schiff et al., 2024), etc.

Surprisingly, Mamba and its variants independently apply the S6 block to each channel, overlooking information flow across channels (also known as *Channel Mixing*). The lack of channel mixing in Mamba not only results in stability issues while scaling to large-sized networks (Patro & Agneeswaran, 2024), but it also cause missing the relationships among feature maps, limiting Mamba's ability to model global information in multi-dimensional data such as images and multivariate time series. Using additional channel mixing blocks for S6 blocks, however, might be challenging in large-scale networks as due to their recurrent nature, increasing the number of blocks for both token and channel mixing can damage the information flow, limiting their ability to use early features.

In this paper, we present MambaMixer, an efficient selective state space models with dual selection across both channels and tokens. MambaMixer sequentially uses *Selective Token Mixer* and *Selective Channel Mixer*, each of which consists of a bidirectional S6 block (Gu & Dao, 2023), to efficiently select and mix (resp. filter) informative (resp. irrelevant) tokens and channels. Contrary to original Mamba block that uses simple skip connections between consecutive blocks, inspired by DenseNet (Huang et al., 2017) and DenseFormer (Pagliardini et al., 2024), MambaMixer block allows layers to have direct access to earlier features (i.e., inputs and outputs of earlier layers) and further enhances the information flow between selective channel and token mixer blocks as well as different layers using a weighted averaging mechanism, enabling MambaMixer-based models to use a large number of layers and making training more stable for large networks.

As a proof of concept, we employ MambaMixer block to design **Vi**sion **M**amba**M**ixer (ViM2) and **T**ime **S**eries **M**amba**M**ixer (TSM2) for vision and time series forecasting tasks, respectively. ViM2 first tokenizes the images and uses Cross-Scan Module (CSM) (Liu et al., 2024) to vertically and horizontally scan images, determining the order of tokens. It then, uses a MambaMixer block to selectively mixes tokens and channels. While TSM2 shares very similar architecture with ViM2, Contrary to ViM2, it uses unidirectional S6 blocks for time dimension due to its causal nature and inspired by (Chen et al., 2023), it uses an additional MambaMixer block to selectively mix auxiliary information of time series (whenever is available) as well as a 2-dimensional normalization across both time and variate dimensions. We further explore the performance of proposed models in various vision and time series forecasting tasks. In ImageNet classification, object detection, and semantic segmentation tasks, ViM2 achieves competitive performance with well-established vision models, and outperforms SSM-based vision models. In time series forecasting, TSM2 achieves outstanding performance compared to state-of-the-art methods on various datasets with diverse domains.

Table 1: Comparison of the architecture design of MambaMixer and existing models.

Backbone	**Token Mixing**		**Channel Mixing**		**Complexity**	**Models**
	Data-dependent	**Module**	**Data-dependent**	**Module**		
MLP-Mixer	–	MLP	–	MLP	$O(L^2)$	MLP-Mixer (Tolstikhin et al., 2021) DynaMixer (Wang et al., 2022b)
ConvMixer	–	Conv	–	Conv	$O(L)$	ConvMixer (Trockman & Kolter, 2023)
Convolution	–	ConvNet	–	MLP	$O(L)$	AlexNet (Krizhevsky et al., 2012) ResNet (He et al., 2016)
Transformers	✓	Attention	–	MLP	$O(L^2)$	ViT (Dosovitskiy et al., 2021) DeIT (Touvron et al., 2021)
	✓	Attention + Conv	–	MLP		SwinTransformer (Liu et al., 2021b)
SSMs	–	SSM	–	–	$O(L)$	Hyena (Poli et al., 2023) H3 (Fu et al., 2023b)
	✓	Selective SSM	–	–		Mamba (Gu & Dao, 2023) Vim (Zhu et al., 2024)
	✓	Selective SSM	✓	Selective SSM		MambaMixer (Ours)

Contributions. To summarize, our contributions are: ① Presenting MambaMixer block, a new SSM-based architecture with dual selection that efficiently and effectively selects and mixes (resp. filters) informative (resp. irrelevant) tokens and channels in a data-dependent manner, allowing connections across both channel and token dimensions. ② Demonstrating the ability and effectiveness of bidirectional S6 blocks to focus on or ignore particular channels using ablation studies. ③ Enhancing information flow in multi-layer MambaMixer-based architectures by allowing direct access to earlier features by a weighted averaging mechanism. ④ Presenting ViM2 and TSM2 models based on MambaMixer for vision and time series forecasting tasks, with outstanding performance compared to baselines.

2 Related Work

To situate our contributions in a broader context, we discuss related studies in three groups:

2.1 Sequence Modeling

Transformers (Vaswani et al., 2017) have been the pivotal backbone architecture behind deep learning's success. Despite their outstanding success in various domains, their attention module has quadratic space and time complexity, which limits their scalability. To this end, recently, several studies aim to design attention-free models with competitive performance to Transformers (Karami & Ghodsi, 2024; De et al., 2024). To this end, State-space Models (SSMs) have recently emerged as a powerful and attention-free tools for modeling long input sequences (Poli et al., 2023; Fu et al., 2023b). More specifically, recently, Gu & Dao (2023) present Mamba, a selective state space model that using input-dependent weights can effectively focus on or ignore some particular tokens. While these sequence models show promising performance on 1D data, they overlook the channel-wise dependencies in sequences. Although this channel-wise dependencies in some tasks on 1D input data like language modeling might not significantly damage the performance, its lackness makes the adaption of these sequence models for multi-dimensional data challenging. Our MambaMixer block uses selective state space models (Gu & Dao, 2023) across both channel and tokens to selectively mix and fuse information across these dimensions.

2.2 Architectures for Generic Vision Backbone

Convolutional Neural Networks and Mixer Architectures. CNNs have served as the de-facto standard backbone in computer vision since the AlexNet model (Krizhevsky et al., 2012) outperforms vision models designed based on hand-crafted image features (Pinz et al., 2006). Several studies have focused on improving the design of CNNs: He et al. (2016) present residual networks using skip connection and Ioffe & Szegedy (2015) introduce batch normalization, both enabling the design of very deep CNNs. Several studies have focused on using sparse convolutions (Xie et al., 2017; Liu et al., 2015), e.g., depth-wise convolutions (Guo et al., 2019). Using the idea of convolutions in the extreme case of small kernels, (Tolstikhin et al., 2021) present MLP-Mixer that use MLP across both patch and channel directions to fuse information in both spatial and feature directions. The success of

MLP-Mixer motivated several studies to use matrix multiplication with efficient and sparse matrices across spatial and channel directions (Wang et al., 2022b; Tang et al., 2022; Fu et al., 2023a). All these architectures, however, are based on data-independent weights, limiting their generalizability and in-context learning ability.

Vision Transformers. Motivated by the success of Transformers (Vaswani et al., 2017) in various domains, Vision Transformers (ViT) (Dosovitskiy et al., 2021) are designed purely based on Transformers. This new backbone architecture has inspired a new paradigm of "isotropic" architectures, which are architectures that use patch embeddings for the first layer and have equal size and shape throughout the network. ViTs due to their ability to learn at scales became popular models over time and various of their variants are designed for a wide array of vision tasks (Liu et al., 2021b, 2022a, 2021c; Touvron et al., 2021). While many ViT architectures use MLPs for channel mixing, very recently, several studies suggest that a simple MLP is not able to effectively filter irrelevant features and discuss the need of channel-wise attentions (Zamir et al., 2022; Zhang et al., 2022). Despite their outstanding performance, Transformers' time and memory scales quadratic with respect to the input, making utilizing them challenging for high-resolution images. This is even more challenging, specifically for channel-wise attention in large-networks, where the number of channels is large. Our ViM2 model's time scales linearly with respect to the input and shows the same pattern for its memory usage (see Section 6.1). It further uses selective state space models across both channel and token dimension, enabling effectively select (resp. filter) informative (resp. irrelevant) tokens or channels.

SSM-based Generic Vision Models. S4ND (Nguyen et al., 2022) is pioneer study to employ SSM blocks for visual tasks, handling visual data as continuous signals across 1D, 2D, and 3D domains. Recently, motivated by the success of Mamba (Gu & Dao, 2023), Vmamba (Liu et al., 2024) and Vim (Zhu et al., 2024) adapt Mamba block for generic vision tasks by addressing the directional sensitivity challenge in SSMs based on bi-directional and cross-scan mechanisms. Subsequently, several studies suggest more sophisticated scan mechanisms to improve the performance of these models (Li et al., 2024; Huang et al., 2024; Hu et al., 2024). Surprisingly, existing adaptions mostly focus on different scanning strategies and treat each channel separately, missing cross-channel dependency. Our ViM2 model, using MambaMixer blocks, can effectively and selectively fuse information across both dimensions.

2.3 Architectures for Generic Timeseries Backbone

Not surprisingly, Transformer-based models have been common choices for time series forecasting, when modeling the complex relationships of covariates are required (Zhou et al., 2021; Liu et al., 2021a; Wu et al., 2021; Ilbert et al., 2024; Nie et al., 2023). Several studies have focused on making the design of Transformers for time series more efficient (Zhou et al., 2021; Wu et al., 2021). Some other studies have focused on extracting long-term information for better forecasting (Nie et al., 2023; Zhou et al., 2022a). To this end, Zhou et al. (2022a) and Zhou et al. (2022b) suggest decomposing the sequences using Fast Fourier Transformation, enabling better extraction of long-term information. Recently, Chen et al. (2023) present TSMixer, an all-MLP architecture for time series forecasting, with state-of-the-art performance and show that to achieve competitive performance in practice, Transformers are not necessarily. Our TSM2, is an alternative MLP- and attention-free architecture that is designed based on MambaMixer block. In this design, S6 blocks, in both directions of variates and time, is used to select (resp. filter) important (resp. irrelevant) variates as well as time stamps.

3 Model: MambaMixer

In this section, we first discuss preliminary concepts about SSMs and then introduce MambaMixer block in details.

3.1 Preliminaries

SSMs are known as linear time-invariant systems that map input sequence $x(t) \in \mathbb{R}^L$ to response sequence $y(t) \in \mathbb{R}^L$ (Aoki, 2013). To this end, SSMs use a latent state $h(t) \in \mathbb{R}^{N \times L}$, parameter

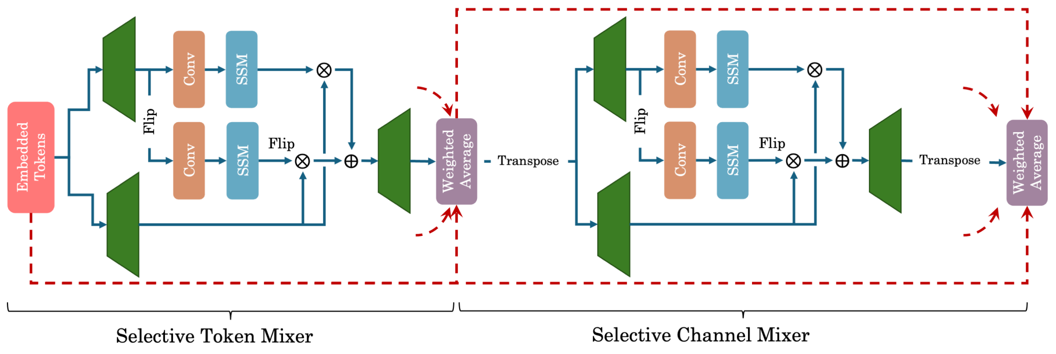

Figure 1: Architecture design of MambaMixer. For further potential architectures see Appendix B.

$\mathbf{A} \in \mathbb{R}^{N\times N}$, and projection parameters $\mathbf{B} \in \mathbb{R}^{N\times 1}$, $\mathbf{C} \in \mathbb{R}^{1\times N}$ such that:

$$\begin{aligned} h'(t) &= \mathbf{A}\, h(t) + \mathbf{B}\, x(t), \\ y(t) &= \mathbf{C}\, h(t). \end{aligned} \tag{1}$$

The above differential equation in deep learning settings, however, is hard to solve and so discrete space state models (Gu et al., 2020; Zhang et al., 2023) suggests discretizing the above system using a parameter $\mathbf{\Delta}$, i.e.,

$$\begin{aligned} h_t &= \bar{\mathbf{A}}\, h_{t-1} + \bar{\mathbf{B}}\, x_t, \\ y_t &= \mathbf{C}\, h_t, \end{aligned} \tag{2}$$

where

$$\begin{aligned} \bar{\mathbf{A}} &= \exp\left(\mathbf{\Delta}\mathbf{A}\right), \\ \bar{\mathbf{B}} &= \left(\mathbf{\Delta}\mathbf{A}\right)^{-1}\left(\exp\left(\mathbf{\Delta}\mathbf{A} - I\right)\right)\ .\ \mathbf{\Delta}\mathbf{B}. \end{aligned} \tag{3}$$

Discrete SSMs can be interpreted as a combination of CNNs and RNNs and are equivalent to the following convolution (Gu et al., 2020):

$$\begin{aligned} \bar{\mathbf{K}} &= \left(\mathbf{C}\bar{\mathbf{B}}, \mathbf{C}\bar{\mathbf{A}}\bar{\mathbf{B}}, \dots, \mathbf{C}\bar{\mathbf{A}}^{L-1}\bar{\mathbf{B}}\right), \\ y &= x * \bar{\mathbf{K}}, \end{aligned} \tag{4}$$

which makes them very efficient in both training, as a CNN, and inference, as an RNN.

Despite discrete SSMs efficiency, they are based on data-independent parameters, meaning that parameters $\bar{\mathbf{A}}$, $\bar{\mathbf{B}}$, and $\mathbf{C}$ are time invariant and the same for any input, limiting their effectiveness in compressing context into a smaller state (Gu & Dao, 2023). To alleviate this limitation, recently, Gu & Dao (2023) present Mamba, a selective SSMs (S6) that effectively selects relevant context by enabling dependence of the parameters $\bar{\mathbf{B}}$, $\bar{\mathbf{C}}$, and $\mathbf{\Delta}$ on the input x_t, i.e.,:

$$\bar{\mathbf{B}}_t = \texttt{Linear}_{\mathbf{B}}(x_t), \qquad \bar{\mathbf{C}}_t = \texttt{Linear}_{\mathbf{C}}(x_t), \qquad \text{and} \qquad \mathbf{\Delta}_t = \texttt{Softplus}\left(\texttt{Linear}_{\mathbf{\Delta}}(x_t)\right), \tag{5}$$

where $\texttt{Linear}(.)$ is a linear projection and $\texttt{Softplus}(.) = \log(1 + \exp(.))$. This selection (or data dependency) comes at the cost of the model not being able to be trained as a CNN (Equation 4), causing challenges for the model efficiency and scalability. To overcome this challenge, (Gu & Dao, 2023) show that the linear recurrence in Equation 2 can be formulated as an associative scan (Martin & Cundy, 2018), which accepts efficient parallel algorithms, resulting in logarithmic complexity in sequence length.

3.2 MambaMixer

As discussed earlier, despite the success of S6 in language modeling (Gu & Dao, 2023), employing them for non-causal and/or multi-dimensional data is challenging. Despite recent attempts to adapt S6 block and Mamba model to non-causal and/or multi-dimensional data (Li et al., 2024; Zhu et al., 2024), surprisingly, existing adaptions mostly focus on different scanning strategies and treat each channel separately, missing cross-channel dependency. To overcome this challenge and allow S6 block to select cross channels and tokens, we present MambaMixer block which has three main modules (See Figure 1): *Selective Token Mixer*, *Selective Channel Mixer*, and *Weighted Averaging of Earlier Features*.

Selective Token Mixer. The first module of MambaMixer is selective token mixer, which aim to mix and fuse information across tokens while having the ability of focus on or ignore particular tokens. Let $x \in \mathbb{R}^{\mathtt{B}\times\mathtt{L}\times\mathtt{D}}$ represents the input, where $\mathtt{B}$ is the batch size, $\mathtt{L}$ is the sequence length and $\mathtt{D}$ is the number of channels. Inspired by Mamba architecture, we use a S6 block with gated MLP as well as a convolution layer. While Mamba is designed for 1D data and so uses a 1D convolution, we later discuss how the choice of convolution should depend on the input data, i.e., depth-wise convolution (Chollet, 2017) for images and 1D convolution for time series. Therefore, unidirectional Selective Token Mixer module can be summarized as follows:

$$\begin{aligned}
\bar{\mathbf{x}} &= \sigma\left(\mathtt{Conv}\left(\mathtt{Linear}\left(x\right)\right)\right), && (6)\\
\bar{\mathbf{x}}_{\mathtt{MLP}} &= \mathtt{MLP}\left(x\right), && (7)\\
\bar{\mathbf{B}}_x &= \mathtt{Linear}_{\mathbf{B}}(x), && (8)\\
\bar{\mathbf{C}}_x &= \mathtt{Linear}_{\mathbf{C}}(x) && (9)\\
\mathbf{\Delta}_x &= \mathtt{Softplus}\left(\mathtt{Linear}_{\mathbf{\Delta}}(x)\right) && (10)\\
\mathbf{y}_{\mathtt{Token}} &= \mathtt{SSM}_{\bar{\mathbf{A}},\bar{\mathbf{B}}_x,\mathbf{C}_x,\mathbf{\Delta}_x}\left(\bar{\mathbf{x}}\right) \otimes \bar{\mathbf{x}}_{\mathtt{MLP}} \qquad \text{(Using Equation 2)}, && (11)
\end{aligned}$$

where $\mathtt{Conv}(.)$ is a convolution and $\otimes$ is non-linearity. The special case of selective token mixer module can be seen as the Mamba block (Gu & Dao, 2023), when we use 1D convolution as $\mathtt{Conv}(.)$. This design allows the model to not only mix and fuse information across tokens, but also focus on or ignore particular tokens in this process. For multi-dimensional data (e.g., images, videos, etc.), however, tokens are not causal and requires a multi-directional mixing process with multiple scans to allow information flow between each pair of tokens. In these cases, with d possible scanning, we simply extend the above block to d-directional selective token mixer by using d unidirectional S6 blocks (instead of one) each of which getting the input sequence based on different scanning, and then added their corresponding outputs together. We further discuss this in Section 4, when we adapt MambaMixer for vision tasks.

Selective Channel Mixer. In multi-dimensional data, learning the dependencies of features is an important step toward training an effective and robust model. For example, it is known that images in different domains share cross-channel correlations while having domain-specific spatial correlations (Guo et al., 2019). In processing high-resolution images, to avoid high computational cost, instead of explicitly modeling pairwise pixel interactions, one can fuse information across feature channels (Zamir et al., 2022). In multivariate time series, features are different variates and to take advantage of complementary information provided by different variates, learning their dependencies is required.

Learning the dependencies of features (channel mixing), however, comes with several major challenges: It ① requires additional parameters, potentially limiting model's scalability, ② needs to be selective and data-dependent to select (resp. filter) informative (resp. non-informative) features, and ③ requires additional attention to training stability and access to early features, due to additional model depth. To overcome ① and ② we use selective channel mixer block that selectively focus on or ignore some particular features, depending on the downstream tasks. In the next part, we further discuss how to address ③.

The architecture design is similar to the selective token mixer module with some slight differences. First, due to the non-causal nature of features, we use a bidirectional method to enhance information flow between each pair of features. Second, since this module is a sequence model and is applied across channel dimension, we use `1D-Conv` as the convolution module. That is,

$$\tilde{\mathbf{x}}_{\texttt{forward}} = \sigma\left(\texttt{1D-Conv}\left(\texttt{Linear}\left(x^\top\right)\right)\right), \tag{12}$$

$$\tilde{\mathbf{B}}_{x^\top} = \texttt{Linear}_{\mathbf{B}}(x^\top), \qquad \bar{\mathbf{C}}_{x^\top} = \texttt{Linear}_{\mathbf{C}}(x^\top), \qquad \text{and} \quad \boldsymbol{\Delta}_{x^\top} = \texttt{Softplus}\left(\texttt{Linear}_{\boldsymbol{\Delta}}(x^\top)\right), \tag{13}$$

$$z = \texttt{Flip}\,(x)\,, \tag{14}$$

$$\tilde{\mathbf{x}}_{\texttt{backward}} = \sigma\left(\texttt{1D-Conv}\left(\texttt{Linear}\left(z^\top\right)\right)\right), \tag{15}$$

$$\tilde{\mathbf{B}}_{z^\top} = \texttt{Linear}_{\mathbf{B}}(z^\top), \qquad \bar{\mathbf{C}}_{z^\top} = \texttt{Linear}_{\mathbf{C}}(z^\top), \qquad \text{and} \quad \boldsymbol{\Delta}_{z^\top} = \texttt{Softplus}\left(\texttt{Linear}_{\boldsymbol{\Delta}}(z^\top)\right), \tag{16}$$

$$\tilde{\mathbf{x}}_{\texttt{MLP}} = \texttt{MLP}\left(x^\top\right), \tag{17}$$

$$\mathbf{y}_{\texttt{Channel}} = \texttt{SSM}_{\bar{\mathbf{A}},\bar{\mathbf{B}}_{x^\top},\mathbf{C}_{x^\top},\boldsymbol{\Delta}_{x^\top}}\left(\tilde{\mathbf{x}}_{\texttt{forward}}\right) \otimes \tilde{\mathbf{x}}_{\texttt{MLP}} + \texttt{Flip}\left(\texttt{SSM}_{\bar{\mathbf{A}},\bar{\mathbf{B}}_{z^\top},\mathbf{C}_{z^\top},\boldsymbol{\Delta}_{z^\top}}\left(\tilde{\mathbf{x}}_{\texttt{backward}}\right)\right) \otimes \tilde{\mathbf{x}}_{\texttt{MLP}}. \tag{18}$$

This design is similar to Mamba (Gu & Dao, 2023), when we make Mamba bidirectional by using two modules that are responsible for forward and backward selective scanning and instead of applying it across tokens, we apply it across channels. This channel mixing enhances information flow between each pair of features and can simulate attention-like functionality, while keeping the complexity linear. For example, in vision tasks, while existing selective SSM-based vision models (e.g., ViM (Zhu et al., 2024) and VMamba (Liu et al., 2024)) are able to select informative image patches, they cannot focus on a particular part of a single patch. Using the selective channel mixer by selecting informative features, MambaMixer is able to focus on a particular part of a single patch, allowing more flexibility and enhancing generalizibility.

Weighted Averaging of Earlier Features. Inspired by DenseNet (Huang et al., 2017) and DenseFormer (Pagliardini et al., 2024), for all MambaMixer layers as well as Selective Token and Channel Mixer blocks, we directly connect the outputs of earlier blocks to their input using a weighted averaging mechanism. This mechanism allows directly re-using early features, capturing complex dynamics and making training more stable. Given $\ell \in \{1, \dots, \mathcal{L}\}$, in ℓ-th MambaMixer block, let $\mathbf{y}^{(\ell)}_{\texttt{Token}}$ and $\mathbf{y}^{(\ell)}_{\texttt{Channel}}$ be the output of selective token and channel mixer blocks, respectively. We compute the input of ℓ-th selective token mixer block, $x^{(\ell)}_{\texttt{Token}}$ as:

$$x^{(\ell)}_{\texttt{Token}} = \sum_{i=0}^{\ell-1} \alpha_{\ell,i}\, \mathbf{y}^{(i)}_{\texttt{Token}} + \sum_{i=0}^{\ell-1} \beta_{\ell,i}\, \mathbf{y}^{(i)}_{\texttt{Channel}}. \tag{19}$$

Similarly, for the input of ℓ-th selective channel mixer block, $x^{(\ell)}_{\texttt{Channel}}$, we have:

$$x^{(\ell)}_{\texttt{Channel}} = \sum_{i=0}^{\ell} \theta_{\ell,i}\, \mathbf{y}^{(i)}_{\texttt{Token}} + \sum_{i=0}^{\ell-1} \gamma_{\ell,i}\, \mathbf{y}^{(i)}_{\texttt{Channel}}. \tag{20}$$

Note that in the above, $\alpha_{i,j}$, $\beta_{i,j}$, $\theta_{i,j}$, and $\gamma_{i,j}$ are learnable parameters and $\mathbf{y}^{(0)}_{\texttt{Token}} = \mathbf{y}^{(0)}_{\texttt{Token}} = x$, where x is the input of the model.

3.3 Computational Complexity

To understand the computational complexity (space and time) of MambaMixer, we discuss each of its selective mixers separately.

Hardware-aware Implementation. The main challenge of making the weights of SSMs input-dependent is that the Equation 4 in training is not valid anymore, making the training recurrent and slow. Thanks to the hardware-aware implementation of S6 block by Gu & Dao (2023), MambaMixer uses their hardware-aware parallel algorithm in recurrent mode for its S6 blocks in both Selective Channel and Token Mixer blocks. Accordingly, there is no restriction for hardware-efficient parallelization of MambaMixer training.

Time Complexity. Let $\texttt{B}$ be the batch size, $\texttt{L}$ be the length of sequence, $\texttt{D}$ be the hidden state dimension, $\texttt{E}$ be the expanded state dimension, and $\texttt{N}$ be the channel dimension. Using hardware-aware implementation of S6 block, its time complexity for token mixing is $\mathcal{O}(\texttt{BLE} + \texttt{EN})$. Similarly,

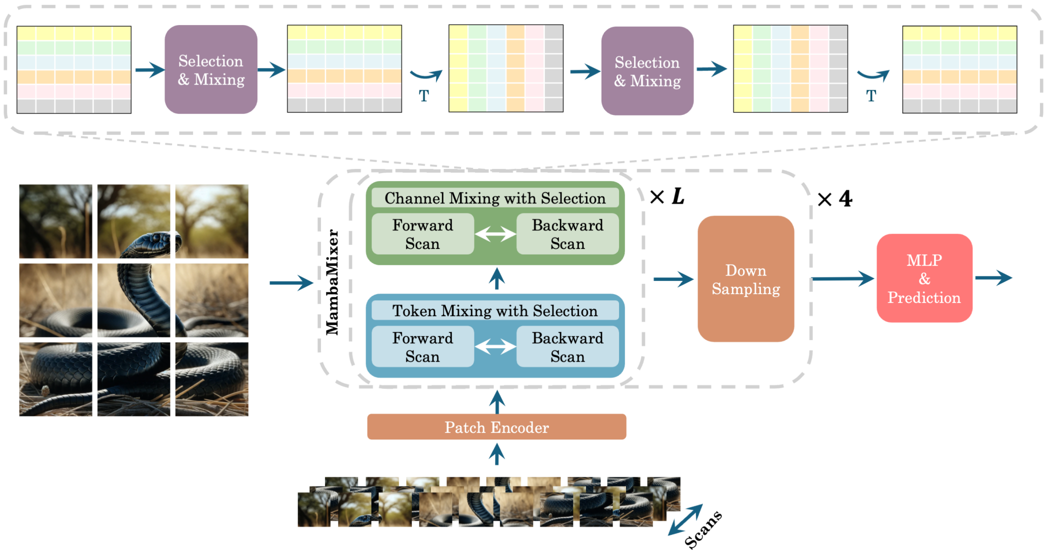

Figure 2: Architecture design and overview of the ViM2's pipeline.

its time complexity for channel mixing is $\mathcal{O}(\texttt{BNE} + \texttt{EL})$. Overall, the time complexity of MambaMixer is $\mathcal{O}(\texttt{EB}(\texttt{N} + \texttt{L}))$, which is linear with respect to both sequence length and its number of channels.

Space Complexity. One of the main limitation of Transformers (Vaswani et al., 2017), and their attention modules is quadratic space complexity with respect to the sequence length. That is, given $\texttt{E} = 2\texttt{D}$, the space complexity of a self-attention block is $\mathcal{O}\left(\texttt{LD}^2 + \texttt{L}^2\texttt{D}\right)$. For a pure Mamba block for vision, which misses the dependencies of channels, this complexity is $\mathcal{O}(\texttt{LDN})$ (Zhu et al., 2024). Our MambaMixer similarly requires $\mathcal{O}(2\texttt{LDN} + \mathcal{L}(2\mathcal{L} + 3))$ space ($\mathcal{L}$ is the number of layers), resulting in a linear space complexity with respect to sequence length and number of channels. Note that $\mathcal{L}(2\mathcal{L} + 3)$ is the required memory for our weighted averaging module. In extreme cases, even using 50 MambaMixer blocks, the overhead memory is negligible.

4 Vision MambaMixer (ViM2)

While in the previous section we focus on designing MambaMixer as a generic backbone that is able to selectively mix both tokens and channels, in this section, we modify and use it to design a vision model.

4.1 ViM2 Architecture

As discussed earlier in Section 3.2, The original design of Selective Token Mixer is suitable for causal tokens, while images are non-causal data and requires model modification. In this section, we first describe the modifications of MambaMixer block to adapt it for vision tasks, and then present its overall architecture.

Scanning. To adapt the primary design of MambaMixer for images, we use cross-scan presented by Liu et al. (2024). In this scan, we scan each image in four directions: top-left to bottom-right, bottom-right to top-left, top-right to bottom-left, and bottom-left to top-right. Then, each scan is passed to a S6 block (selective SSM) and outputs are merged to form new embeddings.

Depth-wise Convolution. Due to the 2 dimensional nature of images, to preserve the its structure, in Selective Token Mixer, we do not flat patches to 1D sequences. Accordingly, each patch has a 2D shape and so we apply depth-wise convolution (Guo et al., 2019) as the choice of convolution module in the architecture of Selective Token Mixer.

Selective Channel Mixing. In the selective Channel Mixer, instead of its primary bidirectional design, we use the same scans as Selective Token Mixer, but pass them to the S6 blocks across their channels and finally merged them to form new features.

Overall Architecture. The overall architecture of ViM2 is illustrated in Figure 2. Using a similar pipeline as Liu et al. (2024), we first, patchify the input images using a stem module, and keep it in the 2D shape without flattening into 1D sequences for token mixing. In the selective channel mixer block, we first flat each token and at the end reshape it to its 2D shape. To learn the hierarchical representation of each image, we then stack multiple modified MambaMixer blocks (described above) with down sampling in four main stages. In fact, each stage (except the first) includes a down sampling in the beginning, resulting in dividing image dimensions by two in each stage, followed by stacks of MambaMixer blocks with weighted averaging modules. We limit these weighted residual connections to each individual stage. Using this design, not only ViM2 can learn hierarchical representations of images, but it also can selectively mix and fuse information across both tokens and channels.

4.2 Connection to MLP-Mixer and VMamba

In an extreme case of using SSMs with zero dimension, i.e., removing S6 blocks from both selective token and channel mixer modules, as well as removing depth-wise convolution, ViM2 is equivalent to MLP-Mixer architecture, where $\texttt{MLP}(.)$ is used to fuse information across both channels and tokens. When freezing all $\beta_{i,j} = 0$, $\alpha_{i,j} = 0$ for $j \leq i-2$, and $\alpha_{i,i-1} = 1$, and/or removing the selective channel mixer block, ViM2 is equivalent to VMamba architecture. Therefore, one can interpret ViM2 as the generalization of VMamba and MLP-Mixer such that it selectively fuses information across both channels and tokens and allows direct access to early features, making the model more robust and stable.

5 Time Series MambaMixer (TSM2)

In learning complex temporal patterns in multivariate time series, capturing the dependencies of variates is a key step. To this end, several studies suggest enhancing cross-variate information flow by using a feature mixer module (Chen et al., 2023; Behrouz & Hashemi, 2023). Existing methods, however, treat all the variates the same in feature mixing phase, including unimportant and/or noisy variates. This becomes a significant challenge when different variates come from different sources and might be noisy (e.g., medical health records), or when some variates are irrelevant to the forecasting task (e.g., traffic forecasting). To overcome this challenge, we adapt our MambaMixer block for multivariate time series forecasting. The Selective Token Mixer (here is also called Selective Time Mixer), can effectively mix and fuse information across time dimension with the ability of focusing on or ignoring specific time stamps. Further, Selective Channel Mixer block (here is also called Selective Variate Mixer) can effectively select (resp. filter) informative (resp. irrelevant) variates. Next, we discuss the details of the TSM2 architecture.

5.1 TSM2 Architecture

The overall architecture of TSM2 is illustrated in Figure 3. TSM2 first patchify each variate of the time series separately, and then uses a MambaMixer block with a unidirectional Selective Token (Time) Mixer and a bidirectional Selective Channel (Variate) Mixer to selectively flow information and mix across both time and channels. The main reason for these choices is the causal nature of time dimension and non-causal nature of variates/features. Similar to the primary design of MambaMixer, we also use weighted averaging to allow the model directly access to early features, improving the stability and allowing the model to effectively ignore unnecessary time and channel mixing operations. We apply 2D normalization on both time and feature dimensions and finally, for the choice of convolution in Selective Token (Time) Mixer, we use a 1D convolution, due to the causal nature of the data.

Auxiliary Information. Chen et al. (2023) show the importance of using auxiliary information in time series forecasting, whenever it is available. That is, in addition to the historical observations (multivariate time series), several real-world scenarios provide us with static $\mathbf{S} \in \mathbb{R}^{M \times C_S}$ and future time-varying features $\mathbf{Z} \in \mathbb{R}^{M \times T_Z \times C_Z}$, where M is the number of variates, T_Z is the time dimension

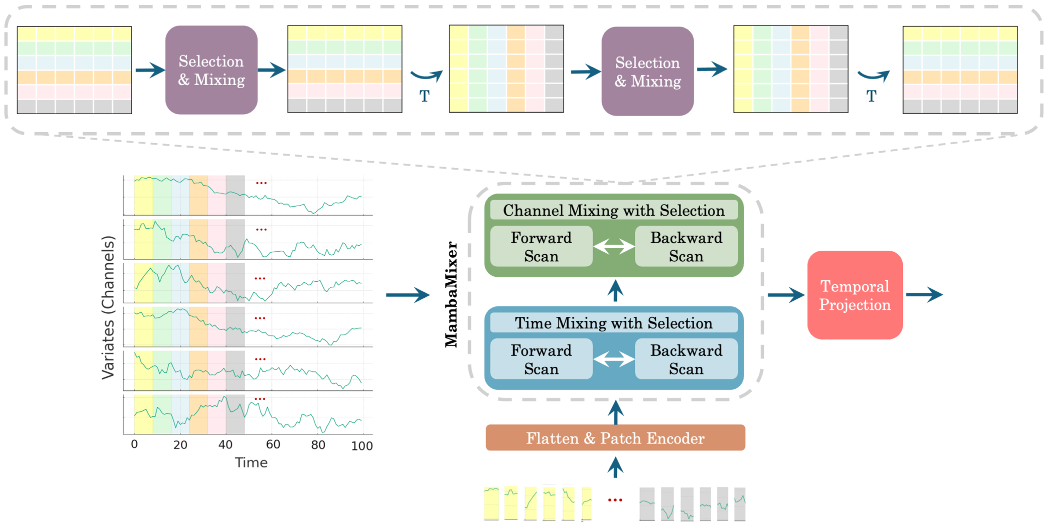

Figure 3: Architecture design and overview of the TSM2's pipeline.

of future features, and C_S and C_Z are channel sizes for static and future features, respectively. For example, the static features can be the location of streets in traffic forecasting, the functionality of brain regions in brain activity analysis, meta-data in medical records for survival analysis, etc. Similarly, future time-varying features can be promotion in subsequent weeks, weather forecasting for traffic prediction, hidden symptoms in clinical latency, etc.

The main challenges to incorporate this information in forecasting tasks is ① how to align them with the historical observations, and ② how to effectively encode them to capture their complex dynamic patterns in forecasting tasks. In our TSM2 model, we follow Chen et al. (2023) pipeline to align historical, future, and static features. That is, given historical multivariate time series data $x \in \mathbb{R}^{M \times T}$ and its corresponding static $\mathbf{S} \in \mathbb{R}^{M \times C_S}$ and future time-varying features $\mathbf{Z} \in \mathbb{R}^{M \times T_Z \times C_Z}$, we first use Selective Channel Mixer on $\mathbf{Z}$ and project it to size $M \times T$, the same size as the historical observation. Similarly, we project $\mathbf{S}$ to the same size and concatenate all the three matrices to use as the input of TSM2. That is,

$$\mathbf{S}_{\text{proj}} = \texttt{Linear}_{\text{stat}}(\mathbf{S}) \in \mathbb{R}^{M \times T}, \tag{21}$$

$$\mathbf{Z}_{\text{proj}} = \texttt{Linear}_{\text{future}}\left(\texttt{SelectiveChannelMixer}\left(\mathbf{Z}\right)\right) \in \mathbb{R}^{M \times T}, \tag{22}$$

$$x_{\text{input}} = x \parallel \mathbf{Z}_{\text{proj}} \parallel \mathbf{S}_{\text{proj}} \in \mathbb{R}^{M \times 3T} \tag{23}$$

$$\mathbf{y} = \texttt{TSM2}\left(x_{\text{input}}\right). \tag{24}$$

Again, due to the causal nature of time-varying auxiliary future features, we use a unidirectional Selective Token Mixer and a bidirectional Selective Channel Mixer.

6 Experiments

In this section, we evaluate the performance of MambaMixer, ViM2, and TSM2 in various tasks of ImageNet classification, object detection, semantic segmentation, and time series forecasting tasks.

ViM2 and TSM2 Models. In our experimental evaluations, we use different variants of our model. While we tune the hyperparameters of TSM2 using grid search to obtain the best performance, for ViM2 we follow the existing studies and use three scales of our model: i.e., ViM2-Tiny, ViM2-Small, and ViM2-Base, referred to as ViM2-T, ViM2-S, and ViM2-B. The details of these architectures are reported in Table 8. For ablation studies, we further replace our selective channel mixer with MLP without weighted averaging, and refer to it as ViM2-MLP and TSM2-MLP.

Table 2: Accuracy comparison across various models on ImageNet-1K[†].

Method	Image Size	#Parameters (M)	ImageNet Top-1 Accuracy
ConvNets			
ResNet-18 (He et al., 2016)	224^2	12	69.8
ResNet-50 (He et al., 2016)	224^2	25	76.2
ResNet-101 (He et al., 2016)	224^2	45	77.4
ResNet-152 (He et al., 2016)	224^2	60	78.3
RegNetY-4G (Radosavovic et al., 2020)	224^2	21	80.0
RegNetY-8G (Radosavovic et al., 2020)	224^2	39	81.7
RegNetY-16G (Radosavovic et al., 2020)	224^2	84	82.9
EffNet-B3 (Tan & Le, 2019)	300^2	12	81.6
EffNet-B4 (Tan & Le, 2019)	380^2	19	82.9
EffNet-B5 (Tan & Le, 2019)	456^2	30	83.6
EffNet-B6 (Tan & Le, 2019)	528^2	43	84.0
Mixer			
Mixer-B/16 (Tolstikhin et al., 2021)	224^2	59	76.4
Mixer-L/16 (Tolstikhin et al., 2021)	224^2	207	71.8
ConvMixer-768/32 (Trockman & Kolter, 2023)	224^2	21	80.2
ConvMixer-1536/20 (Trockman & Kolter, 2023)	224^2	52	81.4
M2-ViT-b (Fu et al., 2023a)	224^2	45	79.5
Transformers			
ViT-b + Monarch (Fu et al., 2023a)	224^2	33	78.9
ViT-B/16 (Dosovitskiy et al., 2021)	384^2	86	77.9
ViT-L/16 (Dosovitskiy et al., 2021)	384^2	307	76.5
DeiT-S (Touvron et al., 2021)	224^2	22	79.8
DeiT-B (Touvron et al., 2021)	224^2	86	81.8
DeiT-B (Touvron et al., 2021)	384^2	86	83.1
Swin-T (Liu et al., 2021b)	224^2	29	81.3
Swin-S (Liu et al., 2021b)	224^2	50	83.0
Swin-B (Liu et al., 2021b)	224^2	88	83.5
SSMs			
S4ND-ConvNeXt-T (Nguyen et al., 2022)	224^2	30	82.2
S4ND-ViT-B (Nguyen et al., 2022)	224^2	89	80.4
VMamba-T (Liu et al., 2024)	224^2	22	82.2
VMamba-S (Liu et al., 2024)	224^2	44	83.5
VMamba-B (Liu et al., 2024)	224^2	75	83.2
ViM-T (Zhu et al., 2024)	224^2	7	76.1
ViM-S (Zhu et al., 2024)	224^2	26	80.5
ViM2-MLP	224^2	40	79.1
ViM2-T	224^2	20	82.7
ViM2-S	224^2	43	83.7
ViM2-B	224^2	74	83.9

[†] Reported results are preliminary results and might be changed in next versions.

6.1 Image Classification on ImageNet

Setup. In this experiment, we evaluate and compare ViM2 classification performance on ImageNet-1K (Deng et al., 2009) with other baselines. Following Liu et al. (2024), we use the pipeline of Liu et al. (2022a) and train our models for 300 epochs (with the first 20 epochs to warm-up), with batch size of 1024 and using AdamW optimizer (Loshchilov & Hutter, 2019) with a momentum of 0.9, learning rate of 0.001, and a weight decay of 0.05.

Results. Table 2 reports the ImageNet classification results for ViM2 and various baselines based on ConvNets (i.e., ResNet (He et al., 2016), RegNetY (Radosavovic et al., 2020), and EffNet (Tan & Le, 2019)), Mixer layers (i.e., MLP-Mixer (Tolstikhin et al., 2021), ConvMixer (Trockman & Kolter, 2023), and M2-ViT (Fu et al., 2023a)), Transformer-based (i.e., ViT (Dosovitskiy et al., 2021), DeiT-S (Touvron et al., 2021), and Swin (Liu et al., 2021b)), and finally SSM-based baselines (i.e., VMamba (Liu et al., 2024) and ViM (Zhu et al., 2024)). ViM2-T, ViM2-S, and ViM2-B achieve 82.7%, 83.7%, and 83.9% classification accuracy, respectively. These results show that with similar

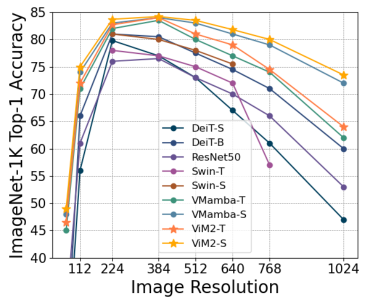

Figure 4: A comparison of input scaling evaluation for ViM2 and baselines. All models have trained with 224 × 224 inputs.

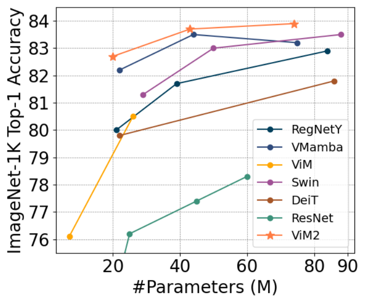

Figure 5: The effect of number of parameters and ViM2 performance comparison with baselines on ImageNet-1K.

scale and size, ViM2 surpasses all well-established vision models, except EffNet (Tan & Le, 2019). More specifically, it surpasses ViM-T by 6.6%, ViM-S by 2.2%, Vmamba-T by 0.5%, Swin-T by 1.4%, MLP-Mixer-B/16 by 6.5%, and ResNet-50 by 6.5%. This pattern persists across different scales: i.e., ViM2-B surpasses VMamba-B by 0.7%, Swin-B by 0.4%, DeiT-B by 0.8%, ResNet-152 by 5.6%, and EffNet-B5 by 0.3%.

Finally, comparing ViM2-MLP with standard ViM2, results show that despite using 100% more parameters, ViM2-T surpasses ViM2-MLP by 3.6%. This shows the significance of the selective channel mixer module as replacing it with a simple MLP damages the performance significantly.

Input Scaling Evaluation and #Parameters. We further evaluate ViM2 on input scaling and compare it with baselines. We follow Liu et al. (2024) and assess the inference performance of ViM2, which is trained with a 224×224 input size. The results are reported in Figure 4. ViM2 and VMamba are the only models that show improvement when we increase image resolution from 224 to 384. Compared to other models, ViM2 shows the most stable performance and less performance drop when using image resolution of 1024. We further report the performance of models on ImageNet-1K with respect to their number of parameters. The results are reported in Figure 5. ViM2 shows the best performance with less number of parameters. There are two main reasons for the superior performance of ViM2. First, its hierarchical architecture allow understanding the image at different level of granularity. Second, its selective channel mixer block as well as its weighted averaging module, which allow direct access to early features, enhance the information flow, resulting in more efficient models.

6.2 Semantic Segmentation

Setup. In this experiment, we evaluate the performance of ViM2 model on ADE20K dataset (Zhou et al., 2019) and compare it with state-of-the-art baselines. Following Liu et al. (2024), we construct a UperHead (Xiao et al., 2018b) on top of the pre-trained model. We use crop size of 512×512. While again using AdamW optimizer (Loshchilov & Hutter, 2019), we set the learning rate as 0.00006, and use a batch size of 16 in fine-tuning.

Results. The results of the performance comparison of ViM2 with baselines are reported in Table 3. Results show that with similar scale and size, ViM2 outperforms all the baselines: i.e., ViM2-T surpasses VMamba-T with 1.3, ViM-S with 3.7, Swin-T with 4.2, and ResNet-50 with 6.5 mIoU (single-scale) improvement. The similar trend can be seen for ViM-S model as well as mIoU (multi-scale) measure.

6.3 Object Detection on COCO

Setup. In this experiment, we evaluate and compare the performance of ViM2 with baselines on object detection using the MSCOCO 2017 dataset (Lin et al., 2014). Following the experimental setup of Liu et al. (2024), we use the training framework on the mmdetection library (Chen et al.,

Table 3: Semantic segmentation results on ADE20K using UperNet (Xiao et al., 2018a)[†].

Method	Crop size	mIoU (Single-scale)	mIoU (Multi-scale)	#Parameters (M)
ResNet-50 (He et al., 2016)	512^2	42.1	42.8	67
ResNet-101 (He et al., 2016)	512^2	42.9	44.0	85
DeiT-S + MLN (Touvron et al., 2021)	512^2	43.8	45.1	58
DeiT-B + MLN (Touvron et al., 2021)	512^2	45.5	47.2	144
Swin-T (Liu et al., 2021b)	512^2	44.4	45.8	60
Swin-S (Liu et al., 2021b)	512^2	47.6	49.5	81
Swin-B (Liu et al., 2021b)	512^2	48.1	49.7	121
ConvNeXt-T (Liu et al., 2022b)	512^2	46.0	46.7	60
ConvNeXt-S (Liu et al., 2022b)	512^2	48.7	49.6	82
ConvNeXt-B (Liu et al., 2022b)	512^2	49.1	49.9	122
ViM-T (Zhu et al., 2024)	512^2	41.0	-	13
ViM-S (Zhu et al., 2024)	512^2	44.9	-	46
VMamba-T (Liu et al., 2024)	512^2	47.3	48.3	55
VMamba-S (Liu et al., 2024)	512^2	49.5	50.5	76
VMamba-B (Liu et al., 2024)	512^2	50.0	51.3	110
ViM2-T	512^2	48.6	49.9	51
ViM2-S	512^2	50.2	51.4	75

[†] Reported results are preliminary results and might be changed in next versions.

Table 4: Object detection and instance segmentation results on COCO dataset[†]. Here, AP^b and AP^m denote box AP and mask AP, respectively.

Method	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}	#Parameters (M)
ResNet-50 (He et al., 2016)	38.2	58.8	41.4	34.7	55.7	37.2	44
ResNet-101 (He et al., 2016)	38.2	58.8	41.4	34.7	55.7	37.2	63
Swin-T (Liu et al., 2021b)	42.7	65.2	46.8	39.3	62.2	42.2	48
Swin-S (Liu et al., 2021b)	44.8	66.6	48.9	40.9	63.2	44.2	69
Swin-B (Liu et al., 2021b)	46.9	-	-	42.3	-	-	107
ConvNeXt-T (Liu et al., 2022b)	44.2	66.6	48.3	40.1	63.3	42.8	48
ConvNeXt-S (Liu et al., 2022b)	45.4	67.9	50.0	41.8	65.2	45.1	70
ConvNeXt-B (Liu et al., 2022b)	47.0	69.4	51.7	42.7	66.3	46.0	108
PVTv2-B2 (Wang et al., 2022a)	45.3	67.1	49.6	41.2	64.2	44.4	45
PVTv2-B3 (Wang et al., 2022a)	47.0	68.1	51.7	42.5	65.7	45.7	65
PVTv2-B5 (Wang et al., 2022a)	47.4	68.6	51.9	42.5	65.7	46.0	102
ViT-Adapter-S (Dosovitskiy et al., 2021)	44.7	65.8	48.3	39.9	62.5	42.8	48
ViT-Adapter-B (Dosovitskiy et al., 2021)	47.0	68.2	51.4	41.8	65.1	44.9	102
ViM-T (Zhu et al., 2024)	45.7	63.9	49.6	26.1	49.0	63.2	-*
VMamba-T (Liu et al., 2024)	46.5	68.5	50.7	42.1	65.5	45.3	42
VMamba-S (Liu et al., 2024)	48.2	69.7	52.5	43.0	66.6	46.4	64
VMamba-B (Liu et al., 2024)	48.5	69.6	53.0	43.1	67.0	46.4	96
ViM2-T	47.1	68.7	50.9	42.4	65.6	45.5	39
ViM2-S	48.5	69.9	52.8	43.1	66.8	46.5	62

[*] The number of parameters is not reported (Zhu et al., 2024).
[†] Reported results are preliminary results and might be changed in next versions.

2019). We fine-tune the pre-trained classification models on ImageNet-1K for both 12 and 36 epochs. Again, we use AdamW optimizer with learning rate of 0.0001.

Results. The result of the performance comparison of ViM2 and baselines in object detection are reported in Table 4. We found that the ability of S6 block in capturing long-range contexts results in the superior performance of ViM2 as it enables ViM2 to capture large object that other transformer-based methods fails to capture. ViM2-S achieves on par performance with VMamba-B while having almost 30% (34 M) less parameters. Again, with similar scale and size, ViM2 surpasses baselines performance including Swin-B, ViM-T, ViT, and ResNet-101.

Table 5: Performance comparison between our MambaMixer and baselines for multivariate long-term forecasting with different horizons H. Results of baselines are obtained from Ilbert et al. (2024). We display the MSE of methods. Best results are bolded and highlighted in blue, second best are highlighted in gray[†].

Dataset	H	TSMambaMixer (Ours)	SAMFormer (Ilbert et al., 2024)	Transformer (Ilbert et al., 2024)	TSMixer (Chen et al., 2023)	Informer (Zhou et al., 2021)	Autoformer (Wu et al., 2021)	FEDFormer (Zhou et al., 2022b)	Pyraformer (Liu et al., 2021a)	LogTransformer (Li et al., 2019)
ETTh1	96	**0.375**	0.381	0.509	0.398	0.941	0.435	0.376	0.664	0.878
	192	**0.398**	0.409	0.535	0.426	1.007	0.456	0.423	0.790	1.037
	336	**0.419**	0.423	0.570	0.435	1.038	0.486	0.444	0.891	1.238
	720	**0.422**	0.427	0.601	0.498	1.144	0.515	0.469	0.963	1.135
ETTh2	96	**0.253**	0.295	0.396	0.308	1.549	0.332	0.332	0.645	2.116
	192	**0.334**	0.340	0.413	0.352	3.792	0.426	0.407	0.788	4.315
	336	**0.347**	0.350	0.414	0.360	4.215	0.477	0.400	0.907	1.124
	720	0.401	**0.391**	0.424	0.409	3.656	0.453	0.412	0.963	3.188
ETTm1	96	**0.322**	0.329	0.384	0.336	0.626	0.510	0.326	0.543	0.600
	192	**0.349**	0.353	0.400	0.362	0.725	0.514	0.365	0.557	0.837
	336	**0.366**	0.382	0.461	0.391	1.005	0.510	0.392	0.754	1.124
	720	**0.407**	0.429	0.463	0.450	1.133	0.527	0.446	0.908	1.153
ETTm2	96	**0.173**	0.181	0.200	0.211	0.355	0.205	0.180	0.435	0.768
	192	**0.230**	0.233	0.273	0.252	0.595	0.278	0.252	0.730	0.989
	336	**0.279**	0.285	0.310	0.303	1.270	0.343	0.324	1.201	1.334
	720	0.388	**0.375**	0.426	0.390	3.001	0.414	0.410	3.625	3.048
Electricity	96	**0.142**	0.155	0.182	0.173	0.304	0.196	0.186	0.386	0.258
	192	**0.153**	0.168	0.202	0.204	0.327	0.211	0.197	0.386	0.266
	336	**0.175**	0.183	0.212	0.217	0.333	0.214	0.213	0.378	0.280
	720	**0.209**	0.219	0.238	0.242	0.351	0.236	0.233	0.376	0.283
Exchange	96	0.163	0.161	0.292	0.343	0.847	0.197	**0.139**	-	0.968
	192	**0.229**	0.246	0.372	0.342	1.204	0.300	0.256	-	1.040
	336	0.383	**0.368**	0.494	0.484	1.672	0.509	0.426	-	1.659
	720	**0.999**	1.003	1.323	1.204	2.478	1.447	1.090	-	1.941
Traffic	96	**0.396**	0.407	0.420	0.409	0.733	0.597	0.576	2.085	0.684
	192	**0.408**	0.415	0.441	0.637	0.777	0.607	0.610	0.867	0.685
	336	0.427	**0.421**	0.501	0.747	0.776	0.623	0.608	0.869	0.734
	720	**0.449**	0.456	0.468	0.688	0.827	0.639	0.621	0.881	0.717
Weather	96	**0.161**	0.197	0.227	0.214	0.354	0.249	0.238	0.896	0.458
	192	**0.208**	0.235	0.256	0.231	0.419	0.325	0.275	0.622	0.658
	336	**0.252**	0.276	0.278	0.279	0.583	0.351	0.339	0.739	0.797
	720	0.337	**0.334**	0.353	0.343	0.916	0.415	0.389	1.004	0.869

[†] Reported results are preliminary results and might be changed in the next versions.

6.4 Multivariate Long-term Forecasting

Setup. We perform experiments on 8 publicly available benchmark datasets of real-world multivariate time series, commonly used in studies on long-term forecasting (Ilbert et al., 2024; Chen et al., 2023; Nie et al., 2023). We compare the performance of TSM2 with the state-of-the-art models on multivariate time series forecasting: i.e., SAMFormer (Ilbert et al., 2024), simple cross-variate Transformer (Ilbert et al., 2024), TSMixer (Chen et al., 2023), Informer (Zhou et al., 2021), Autoformer (Wu et al., 2021), FEDFormer (Zhou et al., 2022b), Pyraformer (Liu et al., 2021a), and LogTransformer (Li et al., 2019). All time series are segmented with input length $L = 512$, prediction horizons $H \in \{96, 192, 336, 720\}$, and a stride of 1.

Results. We compare the MSE results of TSM2 with baselines' in Table 5. TSM2 outperforms all the baselines in most cases (26 out of 32 cases as the best model and 5 as the second best model). The reason for the superior performance of TSM2 is two folds:① While some baselines have focused more on cross-time dependencies (Zhou et al., 2021, 2022b), some others are designed to focus more on cross-variate dependencies (Ilbert et al., 2024). This choice, however, depends on the nature of the dataset. TSM2 using weighted averaging module can learn whether it needs to focus more on cross-variate or cross-time dependencies in a data-driven manner. ② The dual selection mechanism in MambaMixer not only allows TSM2 to capture long-range dependencies but it also uses data-dependent weights, making model more generalizable; while some baselines like TSMixer (Chen et al., 2023) uses data-independent simple MLPs to fuse information, lacking the selection ability.

Forecasting with Auxiliary Information. Chen et al. (2023) discussed the importance of auxiliary information in time series forecasting. To show the ability of our TSM2 in using auxiliary features, we follow Chen et al. (2023) and perform an experiment on M5 dataset (Makridakis et al., 2022). We use two additional baselines that are capable of using auxiliary features: i.e., DeepAR (Salinas et al., 2020) and TFT (Lim et al., 2021). The results are reported in Table 6. TSM2 significantly outperforms baselines with 7.43% improvement over the second best model, i.e., TSMixer (Chen et al., 2023).

Table 6: Evaluation of TSM2 and baselines on M5 dataset with auxiliary information.

Measure	TSMambaMixer (Ours)	TSMixer (Chen et al., 2023)	DeepAR (Salinas et al., 2020)	TFT (Lim et al., 2021)
Test WRMSSE	**0.591**	0.640	0.789	0.670
Val WRMSSE	**0.527**	0.568	0.611	0.579

6.5 Significance of Dual Selection

We further evaluate the significance of dual selection and the architectural design of MambaMixer block in TSM2. To this end, we use two variants of TSM2: ① TSM2-MLP replaces the selective channel mixer block with MLP. ② Mamba + Linear Time: use unidirectional Mamba (Gu & Dao, 2023) as the channel mixer and linear layer as the token mixer (similar to TSMixer architecture (Chen et al., 2023)). The results are reported in Table 7. Compared to TSM2-MLP, the superior performance of TSM2 shows the significance of selective channel mixer block in TSM2. Compared to the second baseline, the superior performance of TSM2 shows the importance of bidirectionality in the design of selective channel mixer as well as the significance of the selective token mixer block.

Table 7: Ablation study on the architecture of TSM2 on ETTh1, ETTm1, Exchange, and Electricity datasets.

Model	ETTh1	ETTm1	Exchange	Electricity
TSM2	**0.375**	**0.322**	**0.163**	**0.142**
TSM2-MLP	0.386	0.339	0.197	0.173
Mamba + Linear Time	0.388	0.334	0.220	0.151

7 Conclusion

We present MambaMixer, an efficient selective state space model with dual token and channel selection. MambaMixer uses selective state space models (S6 blocks) in both token and channel directions, enabling the model to effectively and selectively fuse information across both of these dimensions. To enhance the information flow and capturing the complex dynamics of features, MambaMixer uses a learnable weighted averaging mechanism on early features, which allows each block to directly access to early features. As proof of concept, we further present ViM2 and TSM2 models based on MambaMixer block for vision and time series forecasting tasks. Our experimental evaluations show that In ImageNet classification, object detection, and semantic segmentation tasks, ViM2 achieves competitive performance with well-established vision models, i.e., ViT, MLP-Mixer, ConvMixer, and outperforms SSM-based vision models, i.e., ViM and VMamba. In time series forecasting, TSM2 outperforms all the baselines in most datasets and achieve state-of-the-art performance while demonstrating significantly improved computational cost.

References

Aoki, M. *State space modeling of time series*. Springer Science & Business Media, 2013.

Behrouz, A. and Hashemi, F. Learning temporal higher-order patterns to detect anomalous brain activity. In Hegselmann, S., Parziale, A., Shanmugam, D., Tang, S., Asiedu, M. N., Chang, S., Hartvigsen, T., and Singh, H. (eds.), *Proceedings of the 3rd Machine Learning for Health Symposium*, volume 225 of *Proceedings of Machine Learning Research*, pp. 39–51. PMLR, 10 Dec 2023. URL https://proceedings.mlr.press/v225/behrouz23a.html.

Behrouz, A. and Hashemi, F. Graph mamba: Towards learning on graphs with state space models. *arXiv preprint arXiv:2402.08678*, 2024.

Behrouz, A., Delavari, P., and Hashemi, F. Unsupervised representation learning of brain activity via bridging voxel activity and functional connectivity. In *NeurIPS 2023 AI for Science Workshop*, 2023. URL https://openreview.net/forum?id=HSvg7qFFd2.

Chen, B., Dao, T., Winsor, E., Song, Z., Rudra, A., and Ré, C. Scatterbrain: Unifying sparse and low-rank attention. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=SehIKudiIo1.

Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

Chen, S.-A., Li, C.-L., Yoder, N., Arik, S. O., and Pfister, T. Tsmixer: An all-mlp architecture for time series forecasting. *arXiv preprint arXiv:2303.06053*, 2023.

Chollet, F. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1251–1258, 2017.

Dao, T., Chen, B., Sohoni, N. S., Desai, A., Poli, M., Grogan, J., Liu, A., Rao, A., Rudra, A., and Ré, C. Monarch: Expressive structured matrices for efficient and accurate training. In *International Conference on Machine Learning*, pp. 4690–4721. PMLR, 2022.

De, S., Smith, S. L., Fernando, A., Botev, A., Cristian-Muraru, G., Gu, A., Haroun, R., Berrada, L., Chen, Y., Srinivasan, S., et al. Griffin: Mixing gated linear recurrences with local attention for efficient language models. *arXiv preprint arXiv:2402.19427*, 2024.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Ding, J., Ma, S., Dong, L., Zhang, X., Huang, S., Wang, W., Zheng, N., and Wei, F. Longnet: Scaling transformers to 1,000,000,000 tokens. *arXiv preprint arXiv:2307.02486*, 2023.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Fu, D. Y., Arora, S., Grogan, J., Johnson, I., Eyuboglu, S., Thomas, A. W., Spector, B. F., Poli, M., Rudra, A., and Re, C. Monarch mixer: A simple sub-quadratic GEMM-based architecture. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023a. URL https://openreview.net/forum?id=cB0BImqSS9.

Fu, D. Y., Dao, T., Saab, K. K., Thomas, A. W., Rudra, A., and Re, C. Hungry hungry hippos: Towards language modeling with state space models. In *The Eleventh International Conference on Learning Representations*, 2023b. URL https://openreview.net/forum?id=COZDy0WYGg.

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

Gu, A., Dao, T., Ermon, S., Rudra, A., and Ré, C. Hippo: Recurrent memory with optimal polynomial projections. *Advances in neural information processing systems*, 33:1474–1487, 2020.

Gu, A., Goel, K., Gupta, A., and Ré, C. On the parameterization and initialization of diagonal state space models. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022a. URL https://openreview.net/forum?id=yJE7iQSAep.

Gu, A., Goel, K., and Re, C. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022b. URL https://openreview.net/forum?id=uYLFoz1vlAC.

Guo, Y., Li, Y., Wang, L., and Rosing, T. Depthwise convolution is all you need for learning multiple visual domains. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 8368–8375, 2019.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Hu, V. T., Baumann, S. A., Gui, M., Grebenkova, O., Ma, P., Fischer, J., and Ommer, B. Zigma: Zigzag mamba diffusion model. *arXiv preprint arXiv:2403.13802*, 2024.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.

Huang, T., Pei, X., You, S., Wang, F., Qian, C., and Xu, C. Localmamba: Visual state space model with windowed selective scan. *arXiv preprint arXiv:2403.09338*, 2024.

Ilbert, R., Odonnat, A., Feofanov, V., Virmaux, A., Paolo, G., Palpanas, T., and Redko, I. Unlocking the potential of transformers in time series forecasting with sharpness-aware minimization and channel-wise attention. *arXiv preprint arXiv:2402.10198*, 2024.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. pmlr, 2015.

Kacham, P., Mirrokni, V., and Zhong, P. Polysketchformer: Fast transformers via sketches for polynomial kernels. *arXiv preprint arXiv:2310.01655*, 2023.

Karami, M. and Ghodsi, A. Orchid: Flexible and data-dependent convolution for sequence modeling. *arXiv preprint arXiv:2402.18508*, 2024.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.-X., and Yan, X. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in neural information processing systems*, 32, 2019.

Li, S., Singh, H., and Grover, A. Mamba-nd: Selective state space modeling for multi-dimensional data. *arXiv preprint arXiv:2402.05892*, 2024.

Lim, B., Arık, S. Ö., Loeff, N., and Pfister, T. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4):1748–1764, 2021.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pp. 740–755. Springer, 2014.

Liu, B., Wang, M., Foroosh, H., Tappen, M., and Pensky, M. Sparse convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 806–814, 2015.

Liu, S., Yu, H., Liao, C., Li, J., Lin, W., Liu, A. X., and Dustdar, S. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *International conference on learning representations*, 2021a.

Liu, Y., Tian, Y., Zhao, Y., Yu, H., Xie, L., Wang, Y., Ye, Q., and Liu, Y. Vmamba: Visual state space model. *arXiv preprint arXiv:2401.10166*, 2024.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021b.

Liu, Z., Rodriguez-Opazo, C., Teney, D., and Gould, S. Image retrieval on real-life images with pre-trained vision-and-language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2125–2134, 2021c.

Liu, Z., Hu, H., Lin, Y., Yao, Z., Xie, Z., Wei, Y., Ning, J., Cao, Y., Zhang, Z., Dong, L., et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12009–12019, 2022a.

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., and Xie, S. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11976–11986, 2022b.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Makridakis, S., Spiliotis, E., and Assimakopoulos, V. M5 accuracy competition: Results, findings, and conclusions. *International Journal of Forecasting*, 38(4):1346–1364, 2022.

Martin, E. and Cundy, C. Parallelizing linear recurrent neural nets over sequence length. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=HyUNwulC-.

Nguyen, E., Goel, K., Gu, A., Downs, G., Shah, P., Dao, T., Baccus, S., and Ré, C. S4nd: Modeling images and videos as multidimensional signals with state spaces. *Advances in neural information processing systems*, 35:2846–2861, 2022.

Nie, Y., Nguyen, N. H., Sinthong, P., and Kalagnanam, J. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Jbdc0vTOcol.

Pagliardini, M., Mohtashami, A., Fleuret, F., and Jaggi, M. Denseformer: Enhancing information flow in transformers via depth weighted averaging. *arXiv preprint arXiv:2402.02622*, 2024.

Patro, B. N. and Agneeswaran, V. S. Simba: Simplified mamba-based architecture for vision and multivariate time series, 2024.

Pinz, A. et al. Object categorization. *Foundations and Trends® in Computer Graphics and Vision*, 1 (4):255–353, 2006.

Poli, M., Massaroli, S., Nguyen, E., Fu, D. Y., Dao, T., Baccus, S., Bengio, Y., Ermon, S., and Ré, C. Hyena hierarchy: Towards larger convolutional language models. In *International Conference on Machine Learning*, pp. 28043–28078. PMLR, 2023.

Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., and Sutskever, I. Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning*, pp. 28492–28518. PMLR, 2023.

Radosavovic, I., Kosaraju, R. P., Girshick, R., He, K., and Dollár, P. Designing network design spaces. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10428–10436, 2020.

Salinas, D., Flunkert, V., Gasthaus, J., and Januschowski, T. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International journal of forecasting*, 36(3):1181–1191, 2020.

Schiff, Y., Kao, C.-H., Gokaslan, A., Dao, T., Gu, A., and Kuleshov, V. Caduceus: Bi-directional equivariant long-range dna sequence modeling. *arXiv preprint arXiv:2403.03234*, 2024.

Smith, J. T., Mello, S. D., Kautz, J., Linderman, S., and Byeon, W. Convolutional state space models for long-range spatiotemporal modeling. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=1ZvEtnrHS1.

Tan, M. and Le, Q. V. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, pp. 6105–6114, 2019.

Tang, C., Zhao, Y., Wang, G., Luo, C., Xie, W., and Zeng, W. Sparse mlp for image recognition: Is self-attention really necessary? In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pp. 2344–2351, 2022.

Tang, J., Du, M., Vo, V., LAL, V., and Huth, A. Brain encoding models based on multimodal transformers can transfer across language and vision. In Oh, A., Neumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S. (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 29654–29666. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/5ebbbac62b968254093023f1c95015d3-Paper-Conference.pdf.

Tolstikhin, I. O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in neural information processing systems*, 34:24261–24272, 2021.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and Jégou, H. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pp. 10347–10357. PMLR, 2021.

Trockman, A. and Kolter, J. Z. Patches are all you need? *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=rAnB7JSMXL. Featured Certification.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Wang, W., Xie, E., Li, X., Fan, D.-P., Song, K., Liang, D., Lu, T., Luo, P., and Shao, L. Pvt v2: Improved baselines with pyramid vision transformer. *Computational Visual Media*, 8(3):415–424, 2022a.

Wang, Z., Jiang, W., Zhu, Y. M., Yuan, L., Song, Y., and Liu, W. Dynamixer: a vision mlp architecture with dynamic mixing. In *International conference on machine learning*, pp. 22691–22701. PMLR, 2022b.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

Wu, H., Xu, J., Wang, J., and Long, M. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, 34:22419–22430, 2021.

Xiao, G., Tian, Y., Chen, B., Han, S., and Lewis, M. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=NG7sS51zVF.

Xiao, T., Liu, Y., Zhou, B., Jiang, Y., and Sun, J. Unified perceptual parsing for scene understanding. In *ECCV*, pp. 418–434, 2018a.

Xiao, T., Liu, Y., Zhou, B., Jiang, Y., and Sun, J. Unified perceptual parsing for scene understanding. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 418–434, 2018b.

Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1492–1500, 2017.

Zamir, S. W., Arora, A., Khan, S., Hayat, M., Khan, F. S., and Yang, M.-H. Restormer: Efficient transformer for high-resolution image restoration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5728–5739, 2022.

Zhang, H., Wu, C., Zhang, Z., Zhu, Y., Lin, H., Zhang, Z., Sun, Y., He, T., Mueller, J., Manmatha, R., et al. Resnest: Split-attention networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2736–2746, 2022.

Zhang, M., Saab, K. K., Poli, M., Dao, T., Goel, K., and Re, C. Effectively modeling time series with simple discrete state spaces. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=2EpjkjzdCAa.

Zhou, B., Zhao, H., Puig, X., Xiao, T., Fidler, S., Barriuso, A., and Torralba, A. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 127:302–321, 2019.

Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., and Zhang, W. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 11106–11115, 2021.

Zhou, T., Ma, Z., Wen, Q., Sun, L., Yao, T., Yin, W., Jin, R., et al. Film: Frequency improved legendre memory model for long-term time series forecasting. *Advances in Neural Information Processing Systems*, 35:12677–12690, 2022a.

Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., and Jin, R. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International conference on machine learning*, pp. 27268–27286. PMLR, 2022b.

Zhu, L., Liao, B., Zhang, Q., Wang, X., Liu, W., and Wang, X. Vision mamba: Efficient visual representation learning with bidirectional state space model. *arXiv preprint arXiv:2401.09417*, 2024.

A Architectural Overview

Table 8: Architectural overview of the ViM2 in different scales.

layer name	output size		Tiny	Small	Base
Stem	112×112		conv 4×4, 96, stride 4	conv 4×4, 96, stride 4	conv 4×4, 128, stride 4
Stage 1	56×56	Token Mixer	$\begin{bmatrix}\text{Linear 96} \rightarrow 2\times96 \\ \text{DWConv } 3\times3, 2\times96 \\ \text{S6}, 2\times96 \\ \text{Linear } 2\times96\end{bmatrix}\times2$	$\begin{bmatrix}\text{Linear 96} \rightarrow 2\times96 \\ \text{DWConv } 3\times3, 2\times96 \\ \text{S6}, 2\times96 \\ \text{Linear } 2\times96\end{bmatrix}\times2$	$\begin{bmatrix}\text{Linear 128} \rightarrow 2\times128 \\ \text{DWConv } 3\times3, 2\times128 \\ \text{S6}, 2\times128 \\ \text{Linear } 2\times128\end{bmatrix}\times2$
		Channel Mixer	$\begin{bmatrix}\text{CW-Linear 192} \\ \text{CW-S6, 192} \\ \text{CW-Linear 192}\end{bmatrix}\times1$	$\begin{bmatrix}\text{CW-Linear 192} \\ \text{CW-S6, 192} \\ \text{CW-Linear 192}\end{bmatrix}\times1$	$\begin{bmatrix}\text{CW-Linear 256} \\ \text{CW-S6, 256} \\ \text{CW-Linear 256}\end{bmatrix}\times1$
Stage 2	28×28	Token Mixer	$\begin{bmatrix}\text{Linear 192} \rightarrow 2\times192 \\ \text{DWConv } 3\times3, 2\times192 \\ \text{S6}, 2\times192 \\ \text{Linear } 2\times192\end{bmatrix}\times2$	$\begin{bmatrix}\text{Linear 192} \rightarrow 2\times192 \\ \text{DWConv } 3\times3, 2\times192 \\ \text{S6}, 2\times192 \\ \text{Linear } 2\times192\end{bmatrix}\times2$	$\begin{bmatrix}\text{Linear 256} \rightarrow 2\times256 \\ \text{DWConv } 3\times3, 2\times256 \\ \text{S6}, 2\times256 \\ \text{Linear } 2\times256\end{bmatrix}\times2$
		Channel Mixer	$\begin{bmatrix}\text{CW-Linear 384} \\ \text{CW-S6, 384} \\ \text{CW-Linear 384}\end{bmatrix}\times1$	$\begin{bmatrix}\text{CW-Linear 384} \\ \text{CW-S6, 384} \\ \text{CW-Linear 384}\end{bmatrix}\times1$	$\begin{bmatrix}\text{CW-Linear 512} \\ \text{CW-S6, 512} \\ \text{CW-Linear 512}\end{bmatrix}\times1$
Stage 3	14×14	Token Mixer	$\begin{bmatrix}\text{Linear 384} \rightarrow 2\times384 \\ \text{DWConv } 3\times3, 2\times384 \\ \text{S6}, 2\times384 \\ \text{Linear } 2\times384\end{bmatrix}\times6$	$\begin{bmatrix}\text{Linear 384} \rightarrow 2\times384 \\ \text{DWConv } 3\times3, 2\times384 \\ \text{S6}, 2\times384 \\ \text{Linear } 2\times384\end{bmatrix}\times18$	$\begin{bmatrix}\text{Linear 512} \rightarrow 2\times512 \\ \text{DWConv } 3\times3, 2\times512 \\ \text{S6}, 2\times512 \\ \text{Linear } 2\times512\end{bmatrix}\times18$
		Channel Mixer	$\begin{bmatrix}\text{CW-Linear 768} \\ \text{CW-S6, 768} \\ \text{CW-Linear 768}\end{bmatrix}\times3$	$\begin{bmatrix}\text{CW-Linear 768} \\ \text{CW-S6, 768} \\ \text{CW-Linear 768}\end{bmatrix}\times9$	$\begin{bmatrix}\text{CW-Linear 1024} \\ \text{CW-S6, 1024} \\ \text{CW-Linear 1024}\end{bmatrix}\times9$
Stage 4	7×7	Token Mixer	$\begin{bmatrix}\text{Linear 768} \rightarrow 2\times768 \\ \text{DWConv } 3\times3, 2\times768 \\ \text{S6}, 2\times768 \\ \text{Linear } 2\times768\end{bmatrix}\times2$	$\begin{bmatrix}\text{Linear 768} \rightarrow 2\times768 \\ \text{DWConv } 3\times3, 2\times768 \\ \text{S6}, 2\times768 \\ \text{Linear } 2\times768\end{bmatrix}\times2$	$\begin{bmatrix}\text{Linear 1024} \rightarrow 2\times1024 \\ \text{DWConv } 3\times3, 2\times1024 \\ \text{S6}, 2\times1024 \\ \text{Linear } 2\times1024\end{bmatrix}\times2$
		Channel Mixer	$\begin{bmatrix}\text{CW-Linear 768} \\ \text{CW-S6, 768} \\ \text{CW-Linear 768}\end{bmatrix}\times1$	$\begin{bmatrix}\text{CW-Linear 768} \\ \text{CW-S6, 768} \\ \text{CW-Linear 768}\end{bmatrix}\times1$	$\begin{bmatrix}\text{CW-Linear 1024} \\ \text{CW-S6, 1024} \\ \text{CW-Linear 1024}\end{bmatrix}\times1$
Output	1×1		Average Pooling, 1000D FC, Softmax		

B Possible Architecture Design of Selective Channel and Token Mixer Blocks

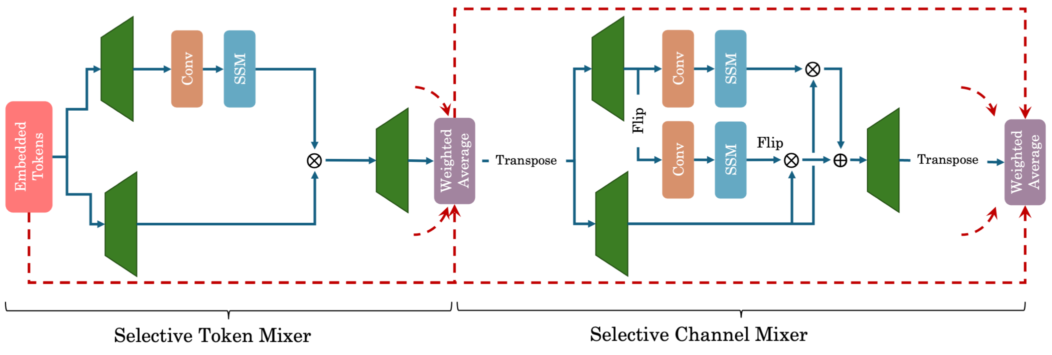

Figure 6: Architecture design of MambaMixer (unidirectional token mixer).

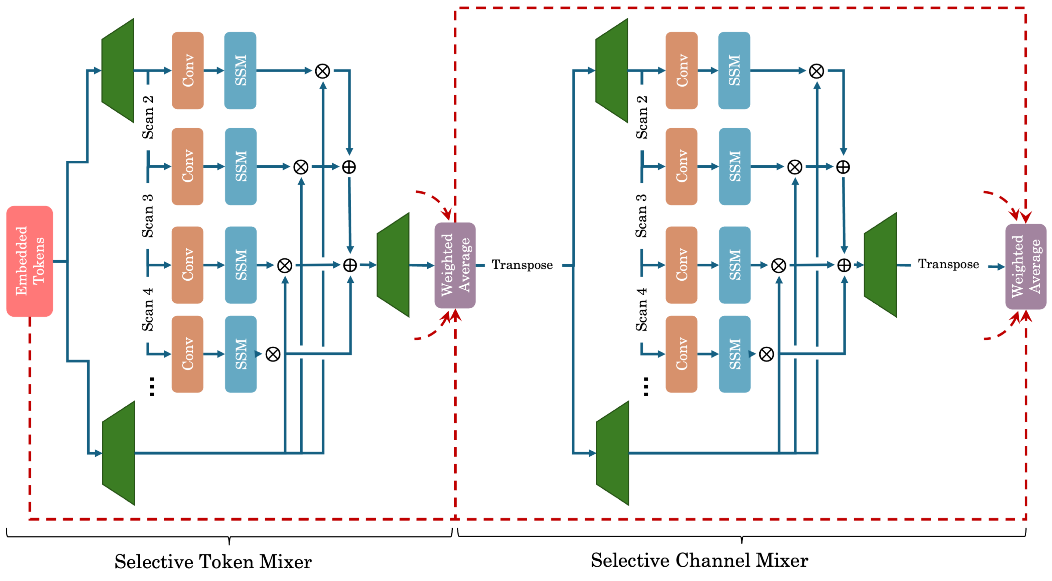

Figure 7: Architecture design of MambaMixer (with n scans).

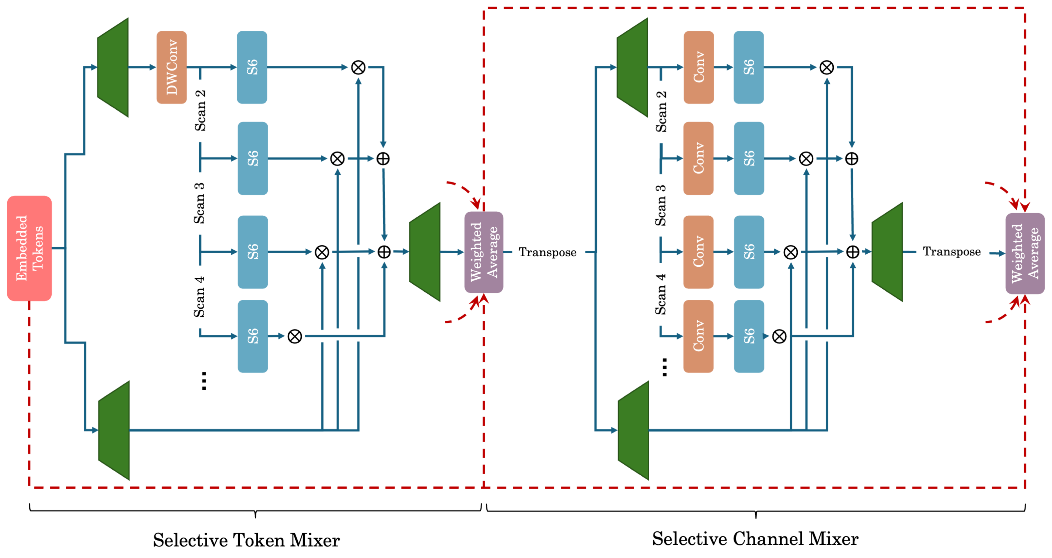

Figure 8: Architecture design of MambaMixer used in ViM2 with depth-wise Conv.