FMamba: Mamba based on Fast-attention for Multivariate Time-series Forecasting

Shusen Ma[1], **Yu Kang**[1, 2, 3], **Peng Bai**[2], **Yun-Bo Zhao**[1, 2, 3*]

[1]Institute of Advanced Technology, USTC
[2]Department of Automation, USTC
[3]Institute of Artificial Intelligence, Hefei Comprehensive National Science Center
ybzhao@ustc.edu.cn

Abstract

In multivariate time-series forecasting (MTSF), extracting the temporal correlations of the input sequences is crucial. While popular Transformer-based predictive models can perform well, their quadratic computational complexity results in inefficiency and high overhead. The recently emerged Mamba, a selective state space model, has shown promising results in many fields due to its strong temporal feature extraction capabilities and linear computational complexity. However, due to the unilateral nature of Mamba, channel-independent predictive models based on Mamba cannot attend to the relationships among all variables in the manner of Transformer-based models. To address this issue, we combine fast-attention with Mamba to introduce a novel framework named FMamba for MTSF. Technically, we first extract the temporal features of the input variables through an embedding layer, then compute the dependencies among input variables via the fast-attention module. Subsequently, we use Mamba to selectively deal with the input features and further extract the temporal dependencies of the variables through the multi-layer perceptron block (MLP-block). Finally, FMamba obtains the predictive results through the projector, a linear layer. Experimental results on eight public datasets demonstrate that FMamba can achieve state-of-the-art performance while maintaining low computational overhead.

Introduction

Multivariate time-series forecasting (MTSF) tasks involve discerning inter-variable correlations and intra-series temporal dependencies by learning the historical observations of diverse variables. The learned prior knowledge is then used to predict the future states of these variables over a certain period. Inter-variable correlations refer to the correlations between different variables, such as how traffic flow might decrease on rainy days because people perceive it as more dangerous to travel. Intra-series temporal dependencies refer to the dependencies between different time points within a sequence. For example, if the temperature is high on a particular day, it might tend to fluctuate around that value in the following days. Learning the inter-variable correlations and intra-series dependencies can help the model make better predictions about the future states of the variables (Liu et al. 2024; Ma et al. 2024).

Currently, the most popular models for MTSF are those based on Transformers, with most adopting a channel-mixing approach (Zhou et al. 2022; Zhang and Yan 2023). However, due to the quadratic computational complexity of self-attention mechanisms, these models' computational cost and efficiency increase and decrease dramatically with the increasing input length. Although DLinear (Zeng et al. 2023) demonstrates that simple models based on linear networks could outperform Transformer-based models in prediction performance, subsequent studies, such as PatchTST (Nie et al. 2023) and iTransformer (Liu et al. 2024), reaffirm the effectiveness of Transformer-based models by adopting channel-independent data processing methods, despite not fundamentally solve the issue of quadratic computational complexity. The model proposed in this paper is also built on the foundation of input channel independence.

To find a model structure that can replace the Transformer, Gu et al. have been promoting the development of State Space Models (SSM) (Gu, Goel, and Ré 2021). SSM can be regarded as a combination of convolutional neural network (CNN) and recurrent neural network (RNN), achieving linear computational efficiency. However, SSM cannot dynamically adjust internal parameters based on different inputs, making it inefficient at extracting temporal features from input sequences. To address this issue, Mamba (Gu and Dao 2023) parameterizes the input of SSM to enable selective processing of input sequences. Nevertheless, due to the unilateral nature of Mamba, it cannot attend to the global variable correlations in the manner of the self-attention mechanism.

To solve the above problem, we apply a fast-attention mechanism to the Mamba, introducing a new prediction model with linear computational complexity called FMamba shown in Figure 1. In FMamba, the input sequence first passes through an Embedding layer to preliminarily extract temporal information. Then, the fast-attention mechanism (Choromanski et al. 2021) efficiently captures the correlations between diverse variables. After that, Mamba parameterizes the input features, allowing FMamba to focus on or ignore information between variables selectively. Subsequently, a multi-layer perceptron block (MLP-block) module extracts deeper temporal information. Finally, a linear layer, called the projector, generates the outcome. Experimentally, the proposed FMamba has demonstrated its effec-

*Corresponding author.

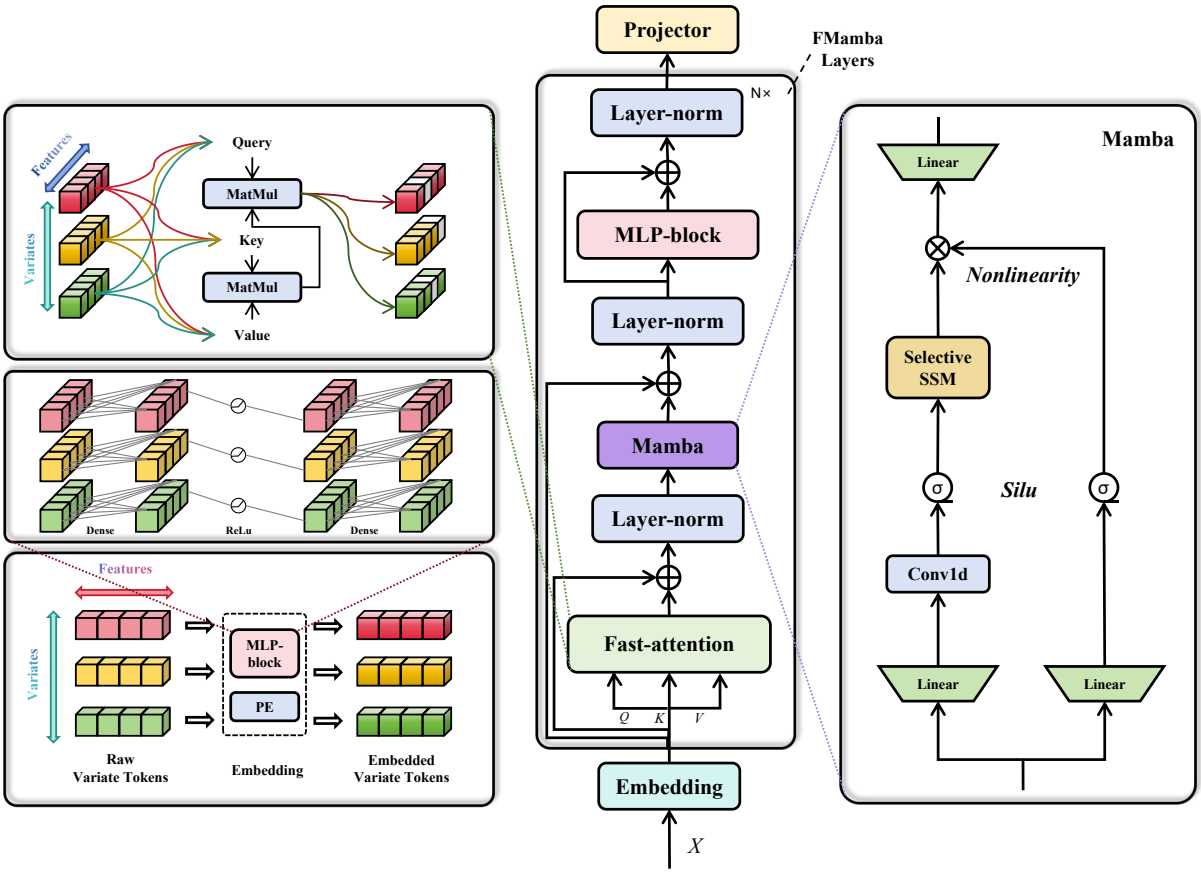

Figure 1: The structure of FMamba.

tiveness across various real-world forecasting benchmarks compared with the current state-of-the-art (SOTA) model. The main contributions of this paper are summarized as follows:

- We propose a novel and simple prediction framework called FMamba, which achieves SOTA-level performance and obtains linear computational complexity, significantly reducing the computational overhead and enhancing the model's computational efficiency.
- We innovatively combine the fast-attention mechanism with the Mamba. The fast-attention mechanism can prevent the model from failing to attend to the correlations of global variables due to the unilateral nature of Mamba, while the introduction of Mamba helps the model better focus on valuable information between variables.
- Experimental results demonstrate that the proposed FMamba achieves SOTA performance on eight public datasets.

Related Work

The traditional MTSF models are based on statistical methods, such as AR (autoregression), MA (moving average), ARMA (autoregression moving average), and ARIMA (autoregression integrated moving average). Although highly interpretable, these methods cannot effectively handle complex nonlinear relationships between various variables. On the other hand, due to the robust nonlinear fitting capability of deep neural networks (DNN), prediction models based on DNN have excelled in MTSF tasks, with classic models being based on RNN. However, the cyclic structure of RNN makes it unsuitable for processing long sequences as it may encounter gradient explosion or gradient vanishing issues. While variants like long short-term memory (LSTM) and gated recurrent unit (GRU) mitigate these problems to some extent through gating mechanisms, they still retain the cyclic structure. Another class of predictive models is based on CNN, which utilizes local perception capabilities to capture the temporal characteristics of variables. Additionally, hybrid models (Wang et al. 2023; Ma et al. 2023) are also introduced widely.

With the booming development of Transformer (Vaswani et al. 2017) in natural language processing and computer vision, many Transformer-based models have been proposed, such as FEDformer (Zhou et al. 2022) and Crossformer (Zhang and Yan 2023). However, recent studies (Zeng et al. 2023; Das et al. 2023) have questioned the necessity of Transformer-based models for MTSF, as simple linear layers can achieve or even surpass the performance of Transformer-based models. To address these concerns,

models like iTransformer (Liu et al. 2024) and PatchTST (Nie et al. 2023) have been proposed. iTransformer treats each input variable as a token, a method also adopted by PCDformer (Ma et al. 2024), and then uses the self-attention mechanism to calculate the correlations between variables. PatchTST divides the input data into diverse channels, allowing different variables to share the same Transformer backbone. Experiments have shown that this new data processing method enables Transformer-based models to once again achieve SOTA-level performance.

However, the self-attention mechanism of Transformer models results in quadratic computational complexity. To find a better framework, SSM has been proposed, reducing computational costs while maintaining strong long-term temporal dependency extraction capabilities. Mamba introduces a selective mechanism based on SSM, enabling the model to selectively focus on or ignore certain inputs during inference through parameterizing the input. TimeMachine (Ahamed and Cheng 2024) is the first to leverage purely SSM modules to capture long-term dependencies in MTSF tasks, employing channel-mixing and channel-independent strategies. To address the limitation of Mamba in MTSF tasks, where it cannot attend to global variables due to its unilateral nature, S-Mamba (Wang et al. 2024) reconfigures Mamba blocks for bidirectional scanning. However, this approach cannot directly calculate the correlation between different variables, as the self-attention mechanism does. To better capture the correlations between variables and the temporal features within sequences, this paper attempts to combine fast-attention and Mamba. By leveraging the advantages of both fast-attention and Mamba while maintaining linear computational complexity, we aim to create a novel, simple, and effective prediction framework for MTSF tasks.

Methodology

In this section, we first briefly explain the problem statement and then provide a detailed introduction to the structure of FMamba.

Problem Statement

Consider a dataset $\mathbf{X} \in \mathbb{R}^{l \times n}$, where l denotes the total length of the dataset and n denotes the number of variables. Given a fixed-length look-back window L, the objective of the MTSF task is to predict the future sequence $\widetilde{\mathbf{X}}_{t+1:t+\tau} = \{\mathbf{x}_{t+1}, \ldots, \mathbf{x}_{t+\tau}\}$, leveraging historical observations $\mathbf{X}_{t-L+1:t} = \{\mathbf{x}_{t-L+1}, \ldots, \mathbf{x}_t\}$. Here, τ denotes the prediction horizon, and $\mathbf{x_t} \in \mathbb{R}^n$ signifies the system's state at time t. In this work, we adopt a channel-independent approach (Ma et al. 2024; Liu et al. 2024) to process the input sequence. Therefore, the input for FMamba can be represented as $\mathbf{X} = \{\mathbf{X}_{\text{token}}, \mathbf{X}_0\} \in \mathbb{R}^{n \times (L+\tau)}$, where $\mathbf{X}_{\text{token}} \in \mathbb{R}^{n \times L}$ and $\mathbf{X}_0 \in \mathbb{R}^{n \times \tau}$ that is initialized by zero (Zhou et al. 2021). Channel independence means that different input variables correspond to various channels.

Embedding Layer

The embedding layer comprises positional embedding (PE) (Vaswani et al. 2017) and an MLP-block. PE aims to add positional information for each time step, aiding the model in understanding the temporal relationships within sequential data. The purpose of the MLP-block is to transform the input of each variable into a high-dimensional representation to capture temporal features. The specific computation formula is as follows:

$$\text{MLP-block}(x) = \text{ReLU}(\mathbf{W}_1 x + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2, \quad (1)$$

where $\mathbf{W}_1, \mathbf{W}_2$ are weight matrices, and $\mathbf{b}_1, \mathbf{b}_2$ are bias vectors. The final output of the embedding layer is the sum of the outputs of PE and MLP-block.

Fast-attention Mechanism

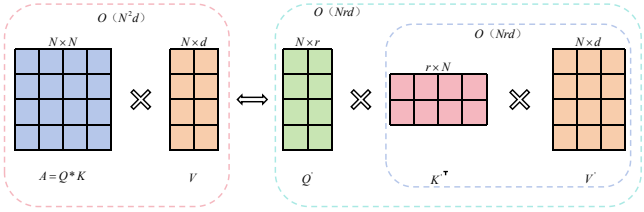

Figure 2: The illustration of canonical self-attention and fast-attention.

Canonical self-attention shown on the left side of Figure 2, also known as scaled dot-product attention, is a mechanism that allows a model to focus on different parts of an input sequence when computing a representation for each element in the sequence. The specific computation formula is as follows:

$$\begin{aligned} &\mathbf{Q} = \mathbf{X}\mathbf{W}_Q, \quad \mathbf{K} = \mathbf{X}\mathbf{W}_K, \quad \mathbf{V} = \mathbf{X}\mathbf{W}_V, \\ &\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}. \end{aligned} \quad (2)$$

The $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ correspond to the query, key, and value matrices, respectively. The $\mathbf{W}_Q$, $\mathbf{W}_K$, and $\mathbf{W}_V$ are all learnable parameters. The d_k is the dimension of the key.

However, due to its quadratic computational complexity, its computational overhead increases rapidly with the increase of the input sequence's length. Inspired by (Choromanski et al. 2021), we attempt to use the fast-attention mechanism, shown on the right side of Figure 2, to reduce computational overhead while enabling the model to discern the correlation between input variables. In this paper, we use the Gaussian kernel function $\mathcal{F}$ to transform the input nonlinearly, obtaining the query and key. It can effectively smooth data, and capture local patterns and nonlinear relationships. Unlike self-attention, fast-attention first multiplies $\mathbf{K'}^\top$ and $\mathbf{V'}$ and then multiplies the product with $\mathbf{Q'}$, achieving linear computational complexity. The specific cal-

culation formula is as follows:

$$\begin{aligned}
&\mathcal{F}(x) = \textbf{exp}(\frac{-x^2}{2}),\\
&\mathbf{Q}' = \mathcal{F}_{\mathbf{Q}'}(\mathbf{X}), \quad \mathbf{K}' = \mathcal{F}_{\mathbf{K}'}(\mathbf{X}), \quad \mathbf{V}' = \mathbf{X}\mathbf{W}_{V'}, \quad (3)\\
&\text{Attention}(\mathbf{Q}', \mathbf{K}', \mathbf{V}') = \frac{\mathbf{Q}'}{k_dim}\left(\mathbf{K}'^{\top}\mathbf{V}'\right),
\end{aligned}$$

where k_dim denotes the kernel dimension of the fast-attention.

Mamba

The SSM represents the internal state evolution of the system through first-order differential equations and controls the system's output through latent state representations, defined as follows:

$$\begin{aligned}
h(t)' &= \boldsymbol{A}h(t) + \boldsymbol{B}x(t),\\
y(t) &= \boldsymbol{C}h(t),
\end{aligned} \tag{4}$$

where $h(t)$ represents the latent state representation at any given time t, and $x(t)$ represents the input at time t. $\boldsymbol{A} \in \mathbb{R}^{N\times N}$, $\boldsymbol{B} \in \mathbb{R}^{N\times D}$, and $\boldsymbol{C} \in \mathbb{R}^{N\times D}$ are all learnable parameter matrices. In real-world applications, especially in MTSF tasks, the data are usually discrete time series. Therefore, the model needs to be discretized to accommodate these discrete-time data. The following formula illustrates how to transform a continuous SSM into a discrete SSM by using zero-order holding and time sampling at intervals of Δ:

$$\begin{aligned}
\overline{\boldsymbol{A}} &= \exp(\Delta\boldsymbol{A}),\\
\overline{\boldsymbol{B}} &= (\Delta\boldsymbol{A})^{-1}(\exp(\Delta\boldsymbol{A}) - \boldsymbol{I}) \cdot \Delta\boldsymbol{B},\\
h_k &= \overline{\boldsymbol{A}}h_{k-1} + \overline{\boldsymbol{B}}x_k,\\
y_k &= \boldsymbol{C}h_k.
\end{aligned} \tag{5}$$

As a parameterized mapping from input data to output data, SSM can be seen as a combination of CNN and RNN. Specifically, a CNN structure is used during training, and an RNN structure is used during inference. Since $\overline{\boldsymbol{A}}$ only remembers the previously captured state information, Gu et al. introduced HiPPO (Gu et al. 2020) to efficiently solve long-range dependency problems in sequence modeling with limited storage space. This constitutes the core of the structured state space model (S4) (Gu, Goel, and Ré 2021).

However, there is a problem with SSM: the matrices $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{C}$ do not vary with diverse inputs, making it impossible to perform targeted inference for various inputs. To address this issue, Mamba (Gu and Dao 2023) designed a simple selection mechanism that parameterizes the input of SSM, allowing the model to selectively focus on or ignore input information during inference. Moreover, Mamba employs a hardware-aware parallel algorithm to ensure computational efficiency. The structure of Mamba is shown on the right side of Figure 1, and its specific process is illustrated in Algorithm 1. The input sequence received by Mamba is $X \in \mathbb{R}^{B\times V\times D}$, where B represents the batch size, V represents the number of input variables, and D represents the feature representation dimension. Similar to the Transformer, the output dimension of the Mamba remains consistent with the input dimension.

Algorithm 1: The process of Mamba Block

Input: $X : (B, V, D)$
Output: $Y : (B, V, D)$

$x : (B, V, E) \leftarrow \texttt{Linear}_x(X)$ ▷ Linear projection
$z : (B, V, E) \leftarrow \texttt{Linear}_z(X)$
$x' : (B, V, E) \leftarrow \texttt{SiLU}(\texttt{Conv1D}(x))$
$\boldsymbol{A} : (D, N) \leftarrow \texttt{Parameter}$ ▷ Structured state matrix
$\boldsymbol{B} : (B, V, N) \leftarrow \texttt{Linear}_B(x')$
$\boldsymbol{C} : (B, V, N) \leftarrow \texttt{Linear}_C(x')$
$\Delta : (B, V, D) \leftarrow \texttt{Softplus}(\texttt{Parameter} + \texttt{Broadcast}(\texttt{Linear}(x')))$
$\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}} : (B, V, D, N) \leftarrow \texttt{discretize}(\Delta, \boldsymbol{A}, \boldsymbol{B})$ ▷ Input-dependent parameters and discretization
$y : (B, V, E) \leftarrow \texttt{SelectiveSSM}(\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}}, \boldsymbol{C})(x')$
$y' : (B, V, E) \leftarrow y \otimes \texttt{SiLU}(z)$
$Y : (B, V, D) \leftarrow \texttt{Linear}(y')$

Specifically, the input of Mamba undergoes linear projections to generate x and z respectively. The further one-dimensional convolution operation is applied to x, followed by a SiLU activation function to produce x'. $\boldsymbol{A}$ is initialized and represents the structured state matrix. $\boldsymbol{B}$ and $\boldsymbol{C}$ are generated from x' through linear transformation. Δ is computed by applying a Softplus function to a combination of the parameter and a broadcasted linear transformation of x'. $\overline{\boldsymbol{A}}$ and $\overline{\boldsymbol{B}}$ are produced by discretization operation. Through the above operations, Mamba makes $\boldsymbol{B}$, $\boldsymbol{C}$, and Δ functions of the input sequence. Although $\boldsymbol{A}$ is not data-dependent, $\overline{\boldsymbol{A}} = \exp(\Delta\boldsymbol{A})$ makes it true. This transformation allows Mamba to adapt its behavior based on the input content, ultimately making Mamba a data-dependent selective model.

Experimental Results

Datasets and Baselines

Table 1: The overall information of the eight datasets.

Datasets	Variants	Dataset Size	Granularity
PEMS03	358	(15617, 5135, 5135)	5min
PEMS04	307	(10172, 3375, 3375)	5min
PEMS07	883	(16911, 5622, 5622)	5min
PEMS08	170	(10690, 3548, 3548)	5min
Electricity	321	(18317, 2633, 5261)	1hour
SML2010	22	(2752, 368, 780)	15min
Weather	21	(36792, 5271, 10540)	10min
Solar-Energy	137	(36601, 5161, 10417)	10min

The primary experiments are carried out on the subsequent eight publicly available datasets: (1) PEMS (PEMS03, PEMS04, PEMS07, and PEMS08): containing California's public traffic network data, collected in 5-minute intervals; (2) Electricity: the hourly data on electricity consumption from 321 clients; (3) SML2010: collecting from 22 sensors in a room for about 40 days with an average sampling time of 15 minutes; (4) Weather: including 21 meteorological variables recorded every 10 minutes at the Weather Station of the Max Planck Biogeochemistry Institute in 2020;

(5) Solar-Energy: recording the solar power production of 137 PV plants in 2006, which are sampled every 10 minutes. More details about the datasets are shown in Table 1.

We primarily compare FMamba with 11 current SOTA models, including S-Mamba (Wang et al. 2024), iTransformer (Liu et al. 2024), RLinear (Li et al. 2023), PatchTST (Nie et al. 2023), Crossformer (Zhang and Yan 2023), TiDE (Das et al. 2023), TimesNet (Wu et al. 2023), DLinear (Zeng et al. 2023), SCINet (LIU et al. 2022), FEDformer (Zhou et al. 2022), and Stationary (Liu et al. 2022).

Experimental Details

We adhere to the standard approach for dataset partitioning, splitting all datasets into training, validation, and testing sets as detailed in Table 1. Our model employs the L2 loss function and utilizes the ADAM optimizer for iterative parameter updates. The training process is set to run for 10 epochs, incorporating early stopping as needed. All experiments are executed using PyTorch on a single NVIDIA GeForce RTX 3090. Algorithm 2 presents the prediction process of FMamba.

Algorithm 2: The Forecasting Procedure of FMamba

Input: $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_L\} \in \mathbb{R}^{B \times L \times n}$
Output: $Y = \{\mathbf{x}_{L+1}, \ldots, \mathbf{x}_{L+\tau}\} \in \mathbb{R}^{B \times \tau \times n}$

$X^\top \in \mathbb{R}^{B \times n \times L} \leftarrow \texttt{Permute}(X)$
$X_{\text{norm}} \leftarrow \texttt{Norm}(X^\top)$
$X_{emb} \leftarrow \texttt{Embedding}(X_{\text{norm}})$
for i **in** FMamba Layers **do**:
    $Y' \leftarrow \texttt{Fast-attention}(X_i)$ ▷ X_i represents the i_{th} FMamba Layer's input
    $Y' \leftarrow \texttt{Layer-norm}(Y' + X_i)$
    $Y' \leftarrow \texttt{Mamba}(Y')$
    $Y' \leftarrow \texttt{Layer-norm}(Y' + X_i)$
    $Y' \leftarrow Y' + \texttt{MLP-block}(Y')$
    $Y' \leftarrow \texttt{Layer-norm}(Y')$
end for
$Y' \in \mathbb{R}^{B \times n \times \tau} \leftarrow \texttt{Projector}(Y')$
$Y \in \mathbb{R}^{B \times \tau \times n} \leftarrow \texttt{Permute}(Y')$

Results and Analysis

The detailed experimental results on eight datasets are presented in Table 2, with the best results highlighted in red and the second-best results underlined. It illustrates the comparative forecasting performance of FMamba and other baselines across various datasets and tasks. To facilitate reproduction, we also provide the model hyperparameters corresponding to different tasks shown in the Appendix. Through analysis, we can find that FMamba achieves lower MSE and MAE metrics than other baselines on most tasks.

Specifically, compared to S-Mamba based on Mamba, FMamba demonstrates an overall improvement in average MSE across all datasets: **22.8%** (0.136→0.105) in PEMS03, **2.1%** (0.096→0.094) in PEMS04, **10.2%** (0.088→0.079) in PEMS07, **26.3%** (0.156→0.115) in PEMS08, **1.2%** (0.171→0.169) in Electricity, **17.6%** (0.415→0.342) in SML2010, **2.0%** (0.252→0.247) in Weather, and **12.7%** (0.244→0.213) in Solar-Energy. In terms of the average MAE indicator, FMamba performs better on most datasets except for Electricity and Weather (0.267→0.269, 0.277→0.293). Compared to iTransformer based on Transformer, FMamba also shows an overall improvement in average MSE across all datasets: **7.1%** (0.113→0.105) in PEMS03, **15.3%** (0.111→0.094) in PEMS04, **21.8%** (0.101→0.079) in PEMS07, **23.3%** (0.150→0.115) in PEMS08, **5.1%** (0.178→0.169) in Electricity, **17.8%** (0.416→0.342) in SML2010, **4.3%** (0.258→0.247) in Weather, and **8.6%** (0.233→0.213) in Solar-Energy. In terms of the average MAE indicator, FMamba performs better on most datasets except for Solar-Energy (0.262→0.270).

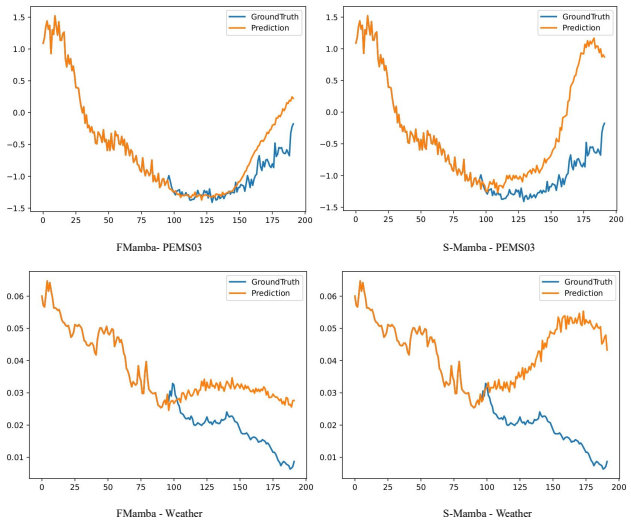

Figure 3: Comparison of forecasting between FMamba and S-Mamba on PEMS03 and Weather when the input length is 96 and the forecasting length is 96.

We attribute FMamba's success to the following two factors: 1) Compared to the bidirectional scanning employed by S-Mamba, FMamba's fast-attention can more comprehensively discern dependencies between variables, aiding the model in learning complex relationships between them; 2) Compared to the traditional Transformer, Mamba not only reduces computational overhead but also enhances the model's robustness by selectively processing or ignoring input information through parameterizing the model's input. To visually demonstrate the prediction performance of FMamba, we visualize its predictions across all datasets and compare them with those of S-Mamba. Part of the visualizations is shown in Figure 3 and the full visualizations can be found in the Appendix.

Ablation Study

To validate the effectiveness of the key components of the model, we design four ablation experiments: 1) To verify whether fast-attention can achieve the same effect as self-attention, we replace fast-attention with self-attention;

Table 2: Full results of diverse MTSF tasks. Extensive baselines are compared under different forecasting lengths following the setting of iTransformer. The input length is set to 96 for all baselines. Avg is the average of all four prediction lengths. * denotes reimplementation and the other baselines' results refer to iTransformer.

Models		**FMamba**		S-Mamba*		iTransformer		RLinear		PatchTST		Crossformer		TiDE		TimesNet		DLinear		SCINet		FEDformer		Stationary	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
PEMS03	12	**0.063**	**0.166**	0.066	0.170	0.071	0.174	0.126	0.236	0.099	0.216	0.090	0.203	0.178	0.305	0.085	0.192	0.122	0.243	0.066	0.172	0.126	0.251	0.081	0.188
	24	**0.080**	**0.189**	0.088	0.197	0.093	0.201	0.246	0.334	0.142	0.259	0.121	0.240	0.257	0.371	0.118	0.223	0.201	0.317	0.085	0.198	0.149	0.275	0.105	0.214
	48	**0.111**	**0.224**	0.165	0.277	0.125	0.236	0.551	0.529	0.211	0.319	0.202	0.317	0.379	0.463	0.155	0.260	0.333	0.425	0.127	0.238	0.227	0.348	0.154	0.257
	96	0.166	0.279	0.226	0.321	**0.164**	**0.275**	1.057	0.787	0.269	0.370	0.262	0.367	0.490	0.539	0.228	0.317	0.457	0.515	0.178	0.287	0.348	0.434	0.247	0.336
	Avg	**0.105**	**0.215**	0.136	0.241	0.113	0.221	0.495	0.472	0.180	0.291	0.169	0.281	0.326	0.419	0.147	0.248	0.278	0.375	0.114	0.224	0.213	0.327	0.147	0.249
PEMS04	12	**0.071**	**0.175**	0.072	0.177	0.078	0.183	0.138	0.252	0.105	0.224	0.098	0.218	0.219	0.340	0.087	0.195	0.148	0.272	0.073	0.177	0.138	0.262	0.088	0.196
	24	**0.082**	**0.188**	0.084	0.192	0.095	0.205	0.258	0.348	0.153	0.275	0.131	0.256	0.292	0.398	0.103	0.215	0.224	0.340	0.084	0.193	0.177	0.293	0.104	0.216
	48	**0.099**	**0.210**	0.101	0.212	0.120	0.233	0.572	0.544	0.229	0.339	0.205	0.326	0.409	0.478	0.136	0.250	0.355	0.437	**0.099**	0.211	0.270	0.368	0.137	0.251
	96	0.125	0.240	0.127	0.236	0.150	0.262	1.137	0.820	0.291	0.389	0.402	0.457	0.492	0.532	0.190	0.303	0.452	0.504	**0.114**	**0.227**	0.341	0.427	0.186	0.297
	Avg	0.094	0.203	0.096	0.204	0.111	0.221	0.526	0.491	0.195	0.307	0.209	0.314	0.353	0.437	0.129	0.241	0.295	0.388	**0.092**	**0.202**	0.231	0.337	0.127	0.240
PEMS07	12	**0.057**	**0.153**	0.060	0.157	0.067	0.165	0.118	0.235	0.095	0.207	0.094	0.200	0.173	0.304	0.082	0.181	0.115	0.242	0.068	0.171	0.109	0.225	0.083	0.185
	24	**0.069**	**0.165**	0.077	0.178	0.088	0.190	0.242	0.341	0.150	0.262	0.139	0.247	0.271	0.383	0.101	0.204	0.210	0.329	0.119	0.225	0.125	0.244	0.102	0.207
	48	**0.084**	**0.183**	0.095	0.197	0.110	0.215	0.562	0.541	0.253	0.340	0.311	0.369	0.446	0.495	0.134	0.238	0.398	0.458	0.149	0.237	0.165	0.288	0.136	0.240
	96	**0.105**	**0.204**	0.118	0.218	0.139	0.245	1.096	0.795	0.346	0.404	0.396	0.442	0.628	0.577	0.181	0.279	0.594	0.553	0.141	0.234	0.262	0.376	0.187	0.287
	Avg	**0.079**	**0.176**	0.088	0.188	0.101	0.204	0.504	0.478	0.211	0.303	0.235	0.315	0.380	0.440	0.124	0.225	0.329	0.395	0.119	0.234	0.165	0.283	0.127	0.230
PEMS08	12	**0.072**	**0.171**	0.076	0.178	0.079	0.182	0.133	0.247	0.168	0.232	0.165	0.214	0.227	0.343	0.112	0.212	0.154	0.276	0.087	0.184	0.173	0.273	0.109	0.207
	24	**0.091**	**0.192**	0.110	0.216	0.115	0.219	0.249	0.343	0.224	0.281	0.215	0.260	0.318	0.409	0.141	0.238	0.248	0.353	0.122	0.221	0.210	0.301	0.140	0.236
	48	**0.121**	**0.221**	0.165	0.252	0.186	0.235	0.569	0.544	0.321	0.354	0.315	0.355	0.497	0.510	0.198	0.283	0.440	0.470	0.189	x0.270	0.320	0.394	0.211	0.294
	96	**0.175**	**0.261**	0.274	0.327	0.221	0.267	1.166	0.814	0.408	0.417	0.377	0.397	0.721	0.592	0.320	0.351	0.674	0.565	0.236	0.300	0.442	0.465	0.345	0.367
	Avg	**0.115**	**0.211**	0.156	0.243	0.150	0.226	0.529	0.487	0.280	0.321	0.268	0.307	0.441	0.464	0.193	0.271	0.379	0.416	0.158	0.244	0.286	0.358	0.201	0.276
Electricity	96	**0.137**	0.238	0.139	**0.235**	0.148	0.240	0.201	0.281	0.195	0.285	0.219	0.314	0.237	0.329	0.168	0.272	0.197	0.282	0.247	0.345	0.193	0.308	0.169	0.273
	192	**0.157**	0.255	0.161	0.258	0.162	**0.253**	0.201	0.283	0.199	0.289	0.231	0.322	0.236	0.330	0.184	0.289	0.196	0.285	0.257	0.355	0.201	0.315	0.182	0.286
	336	**0.174**	0.276	0.181	0.278	0.178	**0.269**	0.215	0.298	0.215	0.305	0.246	0.337	0.249	0.344	0.198	0.300	0.209	0.301	0.269	0.369	0.214	0.329	0.200	0.304
	720	0.208	0.308	**0.201**	**0.298**	0.225	0.317	0.257	0.331	0.256	0.337	0.280	0.363	0.284	0.373	0.220	0.320	0.245	0.333	0.299	0.390	0.246	0.355	0.222	0.321
	Avg	**0.169**	0.269	0.171	**0.267**	0.178	0.270	0.219	0.298	0.216	0.304	0.244	0.334	0.251	0.344	0.192	0.295	0.212	0.300	0.268	0.365	0.214	0.327	0.193	0.296
SML2010	48	**0.244**	**0.279**	0.266	0.290	0.293	0.300	0.289	0.327	0.295	0.323	0.277	0.332	0.408	0.411	0.313	0.307	0.324	0.363	N/A	N/A	0.313	0.370	0.399	0.376
	96	**0.319**	**0.338**	0.415	0.364	0.346	0.347	0.327	0.357	0.345	0.342	0.323	0.365	0.436	0.426	0.382	0.367	0.359	0.387	N/A	N/A	0.374	0.411	0.454	0.411
	192	**0.386**	**0.384**	0.457	0.411	0.447	0.413	0.403	0.404	0.401	0.396	0.483	0.466	0.499	0.461	0.471	0.417	0.428	0.429	N/A	N/A	0.471	0.463	0.550	0.455
	336	**0.418**	**0.411**	0.523	0.451	0.576	0.485	0.440	0.432	0.480	0.441	0.823	0.668	0.568	0.510	0.596	0.486	0.482	0.464	N/A	N/A	0.905	0.678	0.798	0.551
	Avg	**0.342**	**0.353**	0.415	0.379	0.416	0.386	0.365	0.380	0.380	0.376	0.477	0.458	0.478	0.452	0.441	0.394	0.398	0.411	N/A	N/A	0.516	0.481	0.550	0.448
Weather	96	0.164	0.222	0.165	**0.209**	0.174	0.214	0.192	0.232	0.177	0.218	**0.158**	0.230	0.202	0.261	0.172	0.220	0.196	0.255	0.221	0.306	0.217	0.296	0.173	0.223
	192	0.214	0.265	0.215	0.255	0.221	**0.254**	0.240	0.271	0.225	0.259	**0.206**	0.277	0.242	0.298	0.219	0.261	0.237	0.296	0.261	0.340	0.276	0.336	0.245	0.285
	336	**0.267**	0.319	0.273	**0.296**	0.278	**0.296**	0.292	0.307	0.278	0.297	0.272	0.335	0.287	0.335	0.280	0.306	0.283	0.335	0.309	0.378	0.339	0.380	0.321	0.338
	720	**0.341**	0.364	0.353	0.349	0.358	**0.347**	0.364	0.353	0.354	0.348	0.398	0.418	0.351	0.386	0.365	0.359	0.345	0.381	0.377	0.427	0.403	0.428	0.414	0.410
	Avg	**0.247**	0.293	0.252	**0.277**	0.258	0.279	0.272	0.291	0.259	0.281	0.259	0.315	0.271	0.320	0.259	0.287	0.265	0.317	0.292	0.363	0.309	0.360	0.288	0.314
Solar-Energy	96	**0.195**	0.251	0.208	0.246	0.203	**0.237**	0.322	0.339	0.234	0.286	0.310	0.331	0.312	0.399	0.250	0.292	0.290	0.378	0.237	0.344	0.242	0.342	0.215	0.249
	192	**0.219**	0.275	0.240	0.272	0.233	**0.261**	0.359	0.356	0.267	0.310	0.734	0.725	0.339	0.416	0.296	0.318	0.320	0.398	0.280	0.380	0.285	0.380	0.254	0.272
	336	**0.218**	0.276	0.262	0.290	0.248	**0.273**	0.397	0.369	0.290	0.315	0.750	0.735	0.368	0.430	0.319	0.330	0.353	0.415	0.304	0.389	0.282	0.376	0.290	0.296
	720	**0.221**	0.279	0.267	0.293	0.249	**0.275**	0.397	0.356	0.289	0.317	0.769	0.765	0.370	0.425	0.338	0.337	0.356	0.413	0.308	0.388	0.357	0.427	0.285	0.295
	Avg	**0.213**	0.270	0.244	0.275	0.233	**0.262**	0.369	0.356	0.270	0.307	0.641	0.639	0.347	0.417	0.301	0.319	0.330	0.401	0.282	0.375	0.291	0.381	0.261	0.381
1st Count		**34**	**22**	1	6	1	11	0	0	0	0	2	0	0	0	0	0	0	0	3	2	0	0	0	0

2) To compare the impact of Mamba and self-attention on FMamba, we replace Mamba with self-attention; 3) To verify the necessity of fast-attention and Mamba for FMamba, we remove fast-attention and Mamba from FMamba separately. The ablation results are listed in Table 3 and detailed ablation results are shown in the Appendix. Here are some noteworthy points: 1) Replacing fast-attention with self-attention in FMamba has little impact on the model's predictive performance, indicating that fast-attention can achieve the same effect as canonical self-attention while maintaining linear computational complexity; 2) Replacing Mamba with self-attention in FMamba significantly impacts the model's predictive performance, suggesting that the inclusion of Mamba helps the model selectively process input features, enhancing its robustness; 3) Analyzing the results of the third and fourth ablation experiments, we can see the indispensable role of fast-attention and Mamba in FMamba. This is likely due to the ability of fast-attention to attend to

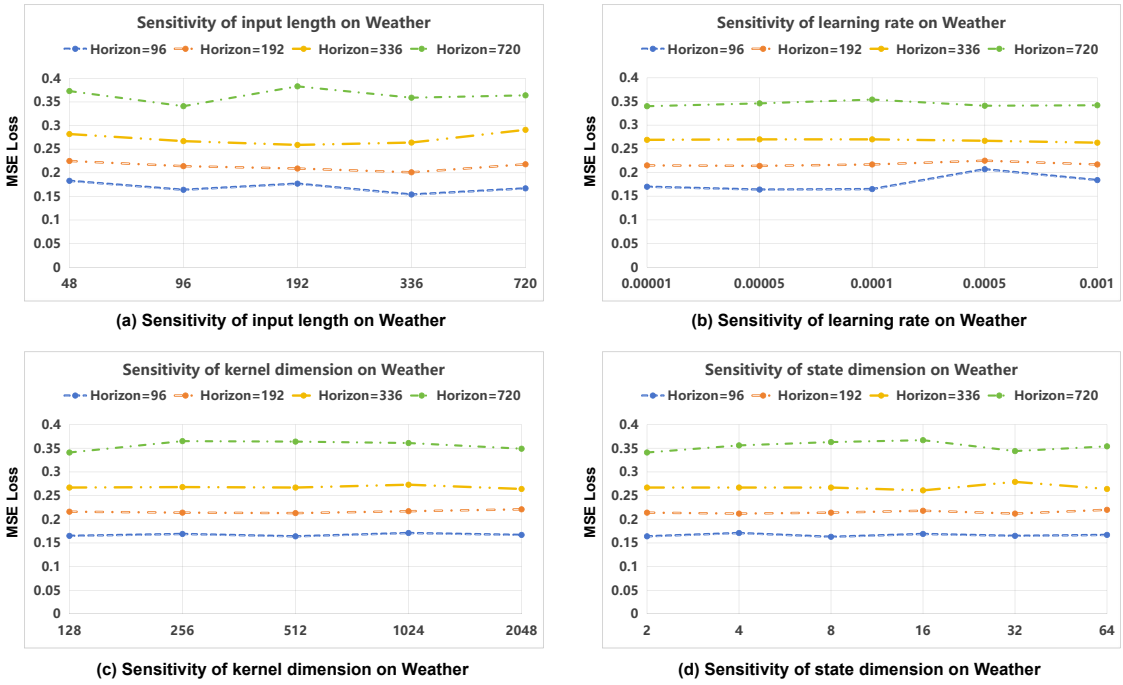

Figure 4: The parameter sensitivity of four components in FMamba.

global variations and Mamba's selective processing capability.

Table 3: Ablations on FMamba. The average results of all predicted lengths are listed here.

Design	Cross-variate Encoding	Selected Attention	PEMS08		Electricity		Solar-Energy	
			MSE	MAE	MSE	MAE	MSE	MAE
FMamba	**Fast-attention**	**Mamba**	**0.115**	**0.211**	0.169	0.269	0.213	0.270
Replace	Self-attention	**Mamba**	0.116	0.214	0.173	0.275	**0.199**	**0.264**
	Fast-attention	Self-attention	0.131	0.227	**0.166**	**0.266**	0.218	0.275
w/o	**Fast-attention**	w/o	0.124	0.215	0.170	0.270	0.213	0.271
	w/o	**Mamba**	0.116	0.214	0.180	0.278	0.201	0.265

Parameter Sensitivity Analysis

To determine whether FMamba is sensitive to the length of the input sequence and various hyper-parameters, we conduct sensitivity analysis experiments using the Weather dataset. Specifically, we examine the model's Input Length, Learning Rate, Kernel Dimension, and State Dimension. For each parameter setting, we adhere to the hyper-parameters listed in Table 5 and keep the other hyper-parameters constant. The detailed results are presented in Figure 4. Overall, the impact of these parameters on the model's predictive performance is minimal, underscoring the robustness of FMamba.

Conclusion

This paper proposes a novel model for MTSF, named FMamba. It primarily comprises an Embedding layer, fast-attention, Mamba, and an MLP-block. The Embedding layer and the MLP-block of the FMamba layer are respectively utilized to extract lower-level and higher-level temporal features. The fast-attention mechanism helps the model overcome the unilateral nature of Mamba, enabling it to attend to the mutual dependencies of various variates like self-attention, without incurring quadratic computational complexity. The Mamba parameterizes the input features, allowing FMamba to focus on or ignore information between variables selectively. Extensive experiments on various real-world datasets demonstrate that FMamba outperforms SOTA methods in addressing MTSF problems.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62173317).

References

Ahamed, M. A.; and Cheng, Q. 2024. Timemachine: A time series is worth 4 mambas for long-term forecasting. *arXiv preprint arXiv:2403.09898*.

Choromanski, K. M.; Likhosherstov, V.; Dohan, D.; Song, X.; Gane, A.; Sarlós, T.; Hawkins, P.; Davis, J. Q.; Mohiuddin, A.; Kaiser, L.; Belanger, D. B.; Colwell, L. J.; and Weller, A. 2021. Rethinking Attention with Performers. In *International Conference on Learning Representations*.

Das, A.; Kong, W.; Leach, A.; Sen, R.; and Yu, R. 2023. Long-term Forecasting with TiDE: Time-series Dense Encoder. *arXiv preprint arXiv:2304.08424*.

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

Gu, A.; Dao, T.; Ermon, S.; Rudra, A.; and Ré, C. 2020. HiPPO: Recurrent Memory with Optimal Polynomial Projections. In *Advances in Neural Information Processing Systems*, volume 33, 1474–1487.

Gu, A.; Goel, K.; and Ré, C. 2021. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*.

Li, Z.; Qi, S.; Li, Y.; and Xu, Z. 2023. Revisiting long-term time series forecasting: An investigation on linear mapping. *arXiv preprint arXiv:2305.10721*.

LIU, M.; Zeng, A.; Chen, M.; Xu, Z.; LAI, Q.; Ma, L.; and Xu, Q. 2022. SCINet: Time Series Modeling and Forecasting with Sample Convolution and Interaction. In *Advances in Neural Information Processing Systems*, volume 35, 5816–5828.

Liu, Y.; Hu, T.; Zhang, H.; Wu, H.; Wang, S.; Ma, L.; and Long, M. 2024. iTransformer: Inverted Transformers Are Effective for Time Series Forecasting. In *International Conference on Learning Representations*.

Liu, Y.; Wu, H.; Wang, J.; and Long, M. 2022. Non-stationary Transformers: Exploring the Stationarity in Time Series Forecasting. In *Advances in Neural Information Processing Systems*, volume 35, 9881–9893.

Ma, S.; Zhang, T.; Zhao, Y.-B.; Kang, Y.; and Bai, P. 2023. TCLN: A Transformer-based Conv-LSTM network for multivariate time series forecasting. *Applied Intelligence*, 53(23): 28401–28417.

Ma, S.; Zhao, Y.-B.; Kang, Y.; and Bai, P. 2024. Multivariate Time Series Modeling and Forecasting with Parallelized Convolution and Decomposed Sparse-Transformer. *IEEE Transactions on Artificial Intelligence*, 1–11.

Nie, Y.; H. Nguyen, N.; Sinthong, P.; and Kalagnanam, J. 2023. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. In *International Conference on Learning Representations*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L. u.; and Polosukhin, I. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30.

Wang, X.; Wang, Y.; Peng, J.; Zhang, Z.; and Tang, X. 2023. A hybrid framework for multivariate long-sequence time series forecasting. *Applied Intelligence*, 53(11): 13549–13568.

Wang, Z.; Kong, F.; Feng, S.; Wang, M.; Zhao, H.; Wang, D.; and Zhang, Y. 2024. Is Mamba Effective for Time Series Forecasting? *arXiv preprint arXiv:2403.11144*.

Wu, H.; Hu, T.; Liu, Y.; Zhou, H.; Wang, J.; and Long, M. 2023. TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis. In *International Conference on Learning Representations*.

Zeng, A.; Chen, M.; Zhang, L.; and Xu, Q. 2023. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, 11121–11128.

Zhang, Y.; and Yan, J. 2023. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *International Conference on Learning Representations*.

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

Zhou, T.; Ma, Z.; Wen, Q.; Wang, X.; Sun, L.; and Jin, R. 2022. FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting. In *International Conference on Machine Learning*, 27268–27286.

Appendix

Table 4: The hyper-parameters of FMamba on PEMS03, PEMS04, PEMS07, and PEMS08 datasets for MTSF tasks.

Models		PEMS03				PEMS04				PEMS07				PEMS08			
Horizon		12	24	48	96	12	24	48	96	12	24	48	96	12	24	48	96
Hyperparameter	el	4	4	4	4	4	4	4	4	2	2	2	2	2	2	2	2
	bs	32	32	32	32	32	32	32	32	16	16	16	16	32	32	32	32
	lr	1e-3	1e-3	1e-3	1e-3	5e-4	5e-4	5e-4	5e-4	5e-4	5e-4	5e-4	5e-4	8e-4	8e-4	8e-4	8e-4
	d_model	512	512	512	512	1024	1024	1024	1024	512	512	512	512	512	512	512	512
	dropout	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	d_state	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
	k_dim	128	128	128	128	128	128	128	128	512	512	512	512	512	512	512	512

[1] $el, bs, lr, d_{\text{model}}, d\_\text{state}$, and k_dim denote the number of encoder layers, batch size, learning rate, feature representation dimension, state dimension of Mamba, and kernel dimension of fast-attention respectively.

Table 5: The hyper-parameters of FMamba on Electricity, SML2010, Weather, and Solar-Energy datasets for MTSF tasks.

Models		Electricity				SML2010				Weather				Solar-Energy			
Horizon		96	192	336	720	48	96	192	336	96	192	336	720	96	192	336	720
Hyperparameter	el	3	3	3	3	3	3	3	3	3	3	3	3	2	2	2	2
	bs	16	16	16	16	32	32	32	32	16	16	16	16	32	32	32	32
	lr	8e-4	1e-3	1e-3	1e-3	8e-4	8e-4	8e-4	8e-4	5e-5	5e-5	7e-5	7e-5	1e-4	1e-4	1e-4	1e-4
	d_model	512	512	512	512	512	512	512	512	512	512	512	512	512	512	512	512
	dropout	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
	d_state	16	16	16	16	2	2	2	2	2	2	2	2	2	2	2	2
	k_dim	512	512	512	512	128	128	128	256	512	256	128	128	256	256	256	256

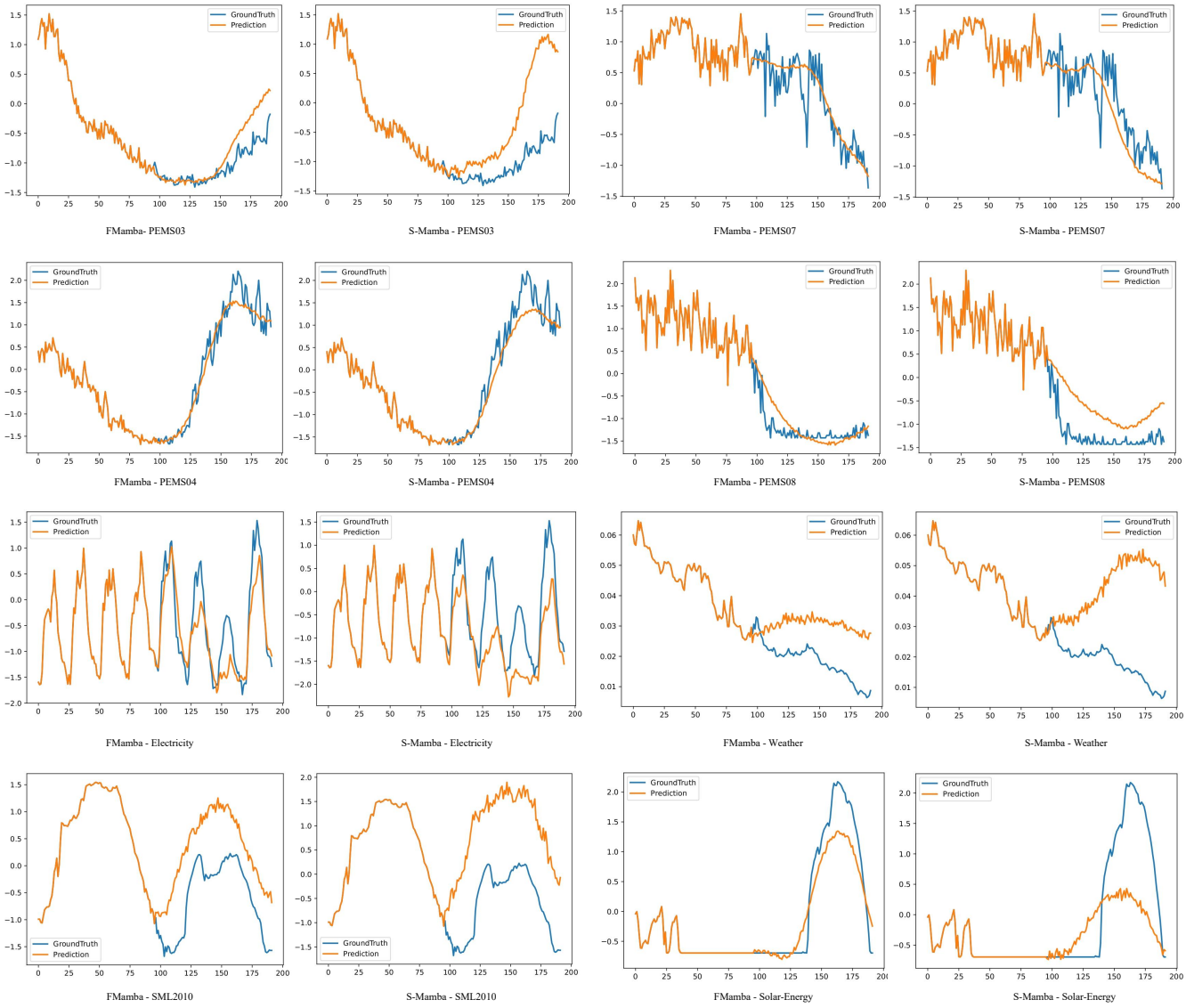

Figure 5: Comparison of forecasts between FMamba and S-Mamba on eight datasets when the input length is 96 and the forecast length is 96. The blue line represents the ground truth and the orange line represents the forecast.

Table 6: Full results of the ablation on FMamba. We replace different components on the respective dimensions and remove the specific component of FMamba.

Design	Cross-variate Encoding	Selected Attention	Forecasting Lengths	PEMS08 MSE	PEMS08 MAE	Forecasting Lengths	Electricity MSE	Electricity MAE	Solar-Energy MSE	Solar-Energy MAE
FMamba	**Fast-attention**	**Mamba**	12	0.072	0.171	96	0.137	0.237	0.195	0.251
			24	0.091	0.192	192	0.157	0.255	0.219	0.275
			48	0.121	0.221	336	0.174	0.276	0.218	0.276
			96	0.177	0.260	720	0.208	0.308	0.221	0.279
			Avg	**0.115**	**0.211**	Avg	0.169	0.269	0.213	0.270
Replace	Self-attention	**Mamba**	12	0.071	0.170	96	0.140	0.241	0.177	0.239
			24	0.092	0.194	192	0.162	0.264	0.196	0.263
			48	0.122	0.223	336	0.183	0.286	0.210	0.273
			96	0.180	0.270	720	0.207	0.310	0.212	0.279
			Avg	0.116	0.214	Avg	0.173	0.275	**0.199**	**0.264**
	Fast-attention	Self-attention	12	0.073	0.176	96	0.139	0.237	0.199	0.257
			24	0.095	0.202	192	0.155	0.254	0.221	0.279
			48	0.136	0.240	336	0.170	0.274	0.225	0.283
			96	0.219	0.290	720	0.199	0.300	0.225	0.281
			Avg	0.131	0.227	Avg	**0.166**	**0.266**	0.218	0.275
w/o	**Fast-attention**	w/o	12	0.071	0.171	96	0.140	0.240	0.195	0.253
			24	0.097	0.193	192	0.154	0.253	0.212	0.271
			48	0.127	0.225	336	0.172	0.275	0.223	0.280
			96	0.199	0.269	720	0.214	0.313	0.222	0.279
			Avg	0.124	0.215	Avg	0.170	0.270	0.213	0.271
	w/o	**Mamba**	12	0.073	0.174	96	0.147	0.247	0.176	0.242
			24	0.092	0.194	192	0.166	0.263	0.200	0.265
			48	0.126	0.227	336	0.185	0.284	0.214	0.276
			96	0.171	0.262	720	0.222	0.318	0.212	0.276
			Avg	0.116	0.214	Avg	0.180	0.278	0.201	0.265