UmambaTSF: A U-shaped Multi-Scale Long-Term Time Series Forecasting Method Using Mamba

Li Wu, Wenbin Pei, Jiulong Jiao, Qiang Zhang

***Abstract*—Multivariate Time series forecasting is crucial in domains such as transportation, meteorology, and finance, especially for predicting extreme weather events. State-of-the-art methods predominantly rely on Transformer architectures, which utilize attention mechanisms to capture temporal dependencies. However, these methods are hindered by quadratic time complexity, limiting the model's scalability with respect to input sequence length. This significantly restricts their practicality in the real world. Mamba, based on state space models (SSM), provides a solution with linear time complexity, increasing the potential for efficient forecasting of sequential data. In this study, we propose UmambaTSF, a novel long-term time series forecasting framework that integrates multi-scale feature extraction capabilities of U-shaped encoder-decoder multilayer perceptrons (MLP) with Mamba's long sequence representation. To improve performance and efficiency, the Mamba blocks introduced in the framework adopt a refined residual structure and adaptable design, enabling the capture of unique temporal signals and flexible channel processing. In the experiments, UmambaTSF achieves state-of-the-art performance and excellent generality on widely used benchmark datasets while maintaining linear time complexity and low memory consumption.**

***Index Terms*—Multivariate time series forecasting, Mamba, Multi-scale features, linear scalability.**

I. Introduction

TIME series forecasting remains a fundamental problem in deep learning, with a wide range of real-world applications where future sequence behaviors are inferred from historical data. Models structured founded on convolutional neural networks (CNNs) and recurrent neural networks (RNNs) have traditionally addressed the complexities of temporal dependencies [1]–[3]. However, the advent of transformer-based architectures has gained prominence due to their exceptional self-attention capabilities [4]. In Particular, the introduction of patch-based transformers, e.g., PatchTST [5], effectively captures both short-term and long-term temporal dependencies, establishing them as exemplary models in long-term forecasting.

With growing demands in practical scenarios and advancements in deep learning techniques, there has been a concerted effort to extend the forecasting horizon of these models. Autofomer [6] automates periodic decomposition and aggregates similar subsequences to mitigate the entanglement and inefficiencies in self-attention mechanisms, thereby extending forecast durations. Subsequently, FEDformer [7], non-stationary Transformers [8] and iTranformer [9] are exemplary models that continue the exploration of long forecasting horizons. However, transformer-based models have been typically criticized for their quadratic time complexity, despite a substantial body of work aimed at addressing this challenge [10]–[12]. The computation of the self-attention mechanism is limited to the receptive window, making it unable to directly understand elements outside this window. In general, a small window size results in poor model performance, while a large window size dramatically increases computational complexity. Meanwhile, linear models [13]–[15], due to their simple and efficient temporal representation and strong interpretability, have successfully demonstrated competitive prediction accuracy. Despite that, empirical comparisons reveal that linear models encounter challenges in complex temporal contexts and tasks requiring extended forecasting horizons. This is primarily due to their limited capacity for non-linear expression and insufficient historical context windows to effectively capture intricate sequence dependencies.

The Mamba model, based on a state-space model (SSM), has recently shown performance comparable to the Transformer in sequence data representation [16]–[18]. It also excels in graph-structured data [19], image processing [20]–[22], and multimodal learning [23], [24], offering superior performance and computational efficiency. Mamba captures long-term correlations and features a context-aware selective mechanism with hidden attention, making it easier to infer long sequences with linear time complexity [25]. This advancement creates new opportunities for efficient long-sequence forecasting. However, there are several open challenges in leveraging the Mamba model for time series forecasting.

Challenge 1: The Mamba model is initially designed to handle long sequence inputs, where efficiently capturing both short-term and long-term temporal dependencies is challenging. Patch-based and sliding window methods disrupt the long cyclic signals crucial to the Mamba model. Additionally, when directly processing input sequences, the Mamba model also

This work was supported by the National Key Research and Development Program of China under grant 2021ZD0112400, the National Natural Science Foundation of China under grants 42265010, 62162053, 62206041, 12371516 and U21A20491, and the NSFC-Liaoning Province United Foundation under grant U1908214, the 111 Project under grant D23006, the Liaoning Revitalization Talents Program under grant XLYC2008017, and China University Industry-University-Research Innovation Fund under grants 2022IT174, Natural Science Foundation of Liaoning Province under grant 2023-BSBA-030, and an Open Fund of National Engineering Laboratory for Big Data System Computing Technology under grant SZU-BDSC-OF2024-09.

Li Wu and Jiulong Jiao are with the School of Computer Science and Technology, Dalian University of Technology, Dalian, 116024, China; Qinghai University, Xining, 810016, China. (e-mail: wuli777@mail.dlut.edu.cn;jiaojiulong@mail.dlut.edu.cn)

Wenbin Pei and Qiang Zhang are with the School of Computer Science and Technology, Dalian University of Technology, Dalian 116024, China; Key Laboratory of Social Computing and Cognitive Intelligence (Dalian University of Technology), Ministry of Education, Dalian 116024, China. (e-mail: peiwenbin@dlut.edu.cn;zhangq@dlut.edu.cn)

struggles to capture multi-scale temporal features, leading to suboptimal use of the information density in time series data. Since time series data often show varying cycles and trends [26]–[28], developing a Mamba model for multi-scale information extraction is essential.

Challenge 2: Time series data often contain overlapping signals, including cyclical, trend-based, seasonal, and stochastic [29], [30], making feature extraction at each scale challenging. A standalone Mamba model with a limited state space struggles to capture complex signal variations [18]. The major challenge is how a simple Mamba configuration can extract the unique information in each cyclic signal, balancing the accuracy of temporal feature representation while improving model efficiency.

Challenge 3: Most studies employ either channel independence or channel parallelism to process multivariate time series, but real-world variable interactions are often more complex. PatchTST [5] and GPT4TS [31] adopt channel independence, processing each channel separately. Differently, iTransformer [9] utilizes channel parallelism, treating channels as multi-dimensional features. For strongly correlated datasets, channel parallelism may need refinement to better capture variable interactions. Therefore, a more flexible processing approach is essential to address diverse relationships.

This paper introduces an innovative framework, UmambaTSF, based on a combination of mamba and linear layers, for multivariate time series forecasting. To address Challenge 1, we introduce a U-shaped multi-scale feature extraction module in UmambaTSF, incorporating Mamba structures. This module leverages Mamba's capacity to capture temporal dependencies across multiple scales, extracting time-series features at each scale to fully exploit the input data. For Challenge 2, inspired by the N-Beats model [32], we propose residual mamba layers, which utilize residual information to iteratively remove noise and redundant signals, uncovering unique information at each scale. Furthermore, to balance training efficiency, the state expansion factor in Mamba is maintained as small as possible. To tackle Challenge 3, UmambaTSF features a flexible Channel-Adaptable Mamba module, consisting of a single Mamba block. This module adjusts to varying channel correlations, allowing for different processing methods that enhance the model's adaptability across diverse datasets. Our contributions are summarized as:

1). We design a multi-feature extractor that captures multiple periodic patterns from time series data across different scales, incorporating both long-term and short-term temporal dependencies.
2). We develop residual Mamba layers to eliminate overlapping and noisy signals in time series data by leveraging residual information, enabling more accurate temporal feature representation at each scale.
3). We propose a flexible Channel-Adaptable Mamba module that effectively adapts to complex channel relationships in multivariate time series, optimizing information transformation for forecasting across various scenarios.

According to the results of the experiments, UmambaTSF achieves state-of-the-art (SOTA) predictive accuracy (as shown in Fig. 1) with the complexity of $O(L)$ for length-L series and minimal memory usage on public real-world datasets. We also conduct comprehensive analyses on the generalization ability of UmambaTSF, highlighting the potential of SSM-based methods in time series forecasting.

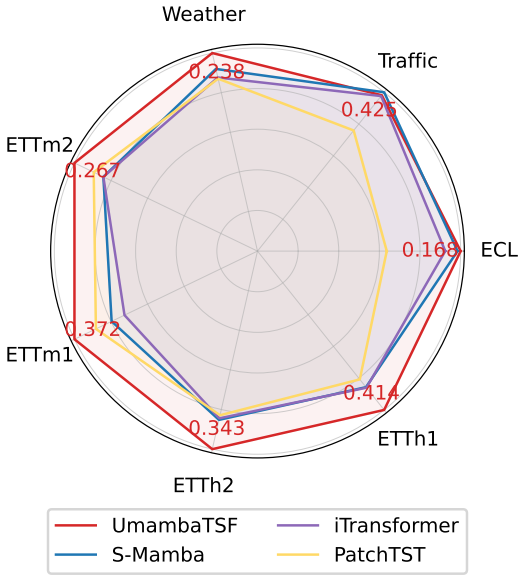


Fig. 1. Average performance (MSE) between UmambaTSF and the latest SOTA models on seven public real-world datasets. The center of the circle denotes the maximum error, while points nearer to the boundary indicate better performance.

II. Background

In this section, we introduce the problem definition of multivariate time series forecasting, followed by the related works in Transformer-based Time Series Forecasting, MLP-based Time Series Forecasting and SSM-based sequence modeling.

A. Multivariate Time Series Forecasting Problem Definition

In multivariate time series forecasting task, historical sequence is defined as $X = [x_1, x_2, \ldots, x_L] \in \mathcal{R}^{N \times L}$, where L represents the length of the historical data and N indicates the number of variables. The ground truth future sequence of length T is $Y = [x_{L+1}, x_{L+2}, \ldots, x_{L+T}] \in \mathcal{R}^{N \times T}$. Based on the sequential information and patterns in X, a predictive model F is expected to be designed to produce the forecast $Y' = F(X)$, aiming to minimize the discrepancy between the predicted Y' and the actual Y, thereby improving the accuracy of the forecasts.

B. Transformer-based Time Series Forecasting

Transformer models have become mainstream methods for time series forecasting due to their self-attention mechanism, which captures long-term dependencies without the vanishing or exploding gradient issues of RNNs. However, the canonical Transformer model has quadratic complexity [33]. Informer [10] reduces the complexity to $O(LlogL)$ with a sparse attention mechanism, enhancing performance for long sequences. Autoformer [6] addresses bottlenecks in sparse attention by

proposing a deep decomposition architecture for better information extraction. Crossformer [34] introduces dual-phase attention for temporal and variable dimensions, improving multivariate time series handling. PatchTST [5] segments the input sequence into smaller patches, allowing the model to process longer sequences while reducing computational complexity. iTransformer [9] modifies the Transformer by inverting the roles of self-attention and feed-forward mechanisms, making it more suitable for time series tasks.

Although complexity has been progressively reduced by various model explorations in Transformer-based time series forecasting, limitations remain due to the inherent computational demands of the self-attention mechanism and the necessity to process complete sequence information. Various improvement efforts have been made, including sparse patterns, kernelization methods, and chunking techniques. However, when managing long sequences, these approaches still encounter the following challenges, including the trade-off between efficiency and accuracy, reliance on extensive training data, and difficulty capturing subtle dependencies within time series.

C. MLP-based Time Series Forecasting

MLP-based models have high interpretability, low computational costs, and ability to model long inputs [35], while they face challenges to the Transformer series. DLinear [13] has initially demonstrated that linear models can perform comparably to, if not better, Transformer architectures in time series forecasting, particularly in capturing trends and residuals. It retains its effectiveness with increasing lengths of the retrospective window, contrasting with the poor performance of classic Transformer architecture models. TiDE [36] consisting of an encoder and a decoder implemented via multilayer perceptrons, takes advantage of the simplicity and speed of linear models while addressing challenges such as long-term dependencies, noise, and uncertainties inherent in time series forecasting. RLinear [14] proposes a single-layer linear model integrating reversible instance normalization (RevIN) with channel independence, which has demonstrated exceptional predictive capabilities across a variety of datasets. However, according to comparative analyses in the iTransformer literature, the predictive difficulty of linear models increases with longer forecast lengths due to a reduced ratio of input data.

Although MLP-based models are adept at detecting periodic patterns within extensive input data, real-world scenarios frequently involve prediction windows substantially longer than the inputs. This highlights the need for further research to extend forecasting lengths while maintaining predictive accuracy.

D. SSM-Based Sequence Modeling

To date, sequence modeling methods based on SSM have garnered increasing attention. SSM represents system dynamics through latent states evolving over time and is renowned for its linear complexity and capacity to handle long sequences. It incorporates both observed data and hidden states, allowing for a comprehensive understanding of temporal dependencies and underlying processes. SSM is adaptable to various forms of sequential data, including those with irregularities and noise. By integrating domain knowledge and handling multivariate sequences, it enhances prediction accuracy. SSM is extensively applied in fields such as economics, engineering, and environmental science, offering robust and interpretable predictions. S4 [37] (Structured State Space Sequence) combines linear SSMs, the HiPPO framework, and deep learning to achieve high performance. S5 [38] replaces the frequency-domain methods used in S4 with a purely recurrent time-domain approach that leverages parallel scanning. S5 maintains the computational efficiency of S4 while achieving superior performance. Mamba [16], building on S4, introduces an information-selective mechanism and hardware-aware acceleration algorithms, further enhancing the performance and computational efficiency of sequence modeling. It has achieved success across multiple domains. S-Mamba [17] utilizes a bidirectional Mamba layer to extract inter-variate correlations, while TimeMachine [18] employs an ensemble of four Mamba models, both representing significant advancements in time series researches.

However, studies using Mamba on time series forecasting models have only recently emerged and require further improvements to enhance its ability to extract both short-term and long-term features, as well as to improve prediction accuracy across different channel scenarios. This paper focuses on addressing the extraction of multi-scale temporal features and expanding the predictive capabilities of Mamba across various channel scenarios while maintaining lower memory usage.

III. The Proposed Method: UmambaTSF

In this section, we first introduce the overall architecture of the UmambaTSF model. Then, we provide a detailed explanation of the multi-scale feature extractor, with a focus on its core component, the Mamba-based temporal signal processor (MTSP).

A. Architecture Overview

The architecture of UmambaTSF is illustrated in Fig. 2. The time series input data is first processed through instance normalization, utilizing standard Z-score normalization [31] and RevIN techniques [39]. This normalization step avoids trend variations due to moment statistics while retaining essential statistical information needed to reconstruct accurate forecasting outcomes. Next, a linear tokenization layer is introduced to map the normalized data to an expanded feature dimension. A multi-scale feature extractor then captures temporal correlations across various scales, after which the multi-scale data is projected to the forecast horizon using a projection layer. Ultimately, instance denormalization is applied to generate the final predictive sequence.

The multi-scale feature extractor is a crucial module within our framework, structured as a U-shaped architecture. This structure facilitates the rich expression of features [40]. In Fig. 2, the left side works as an encoder, utilizing linear layers to map data progressively to shorter lengths of time

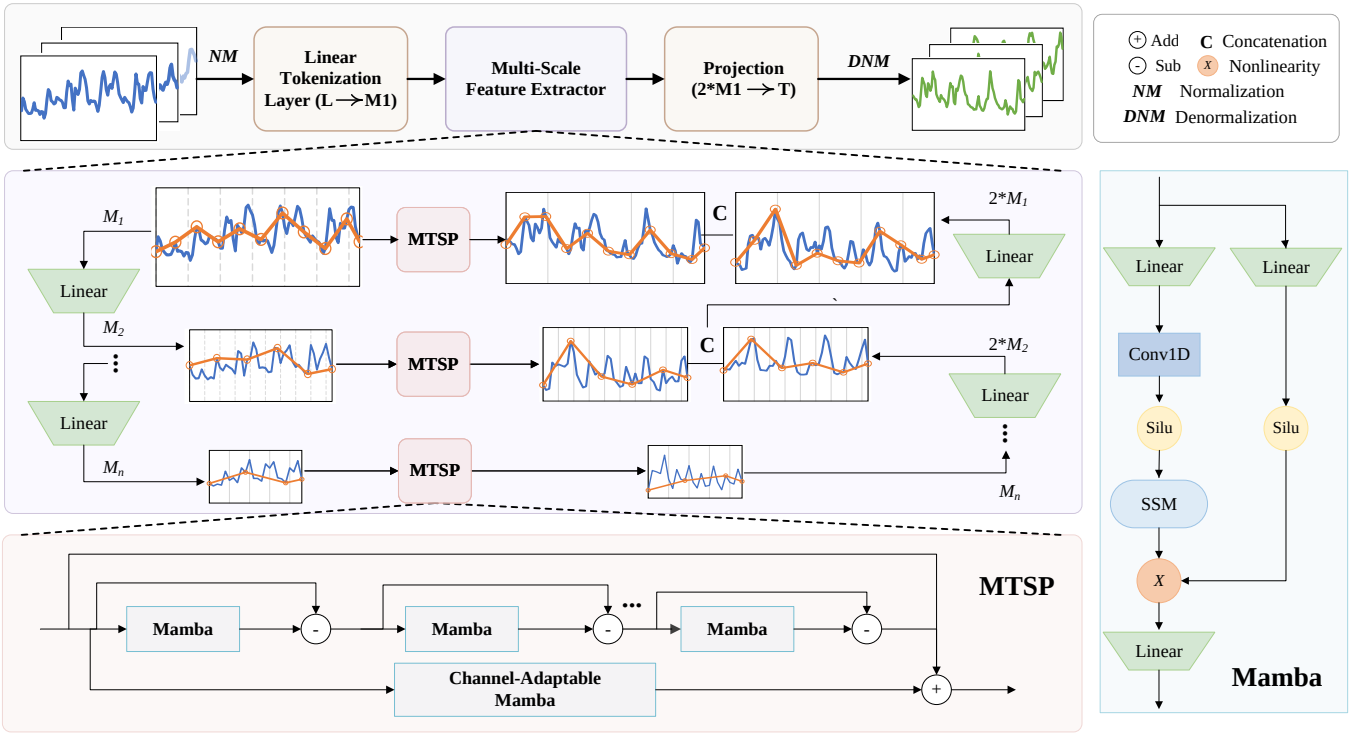


Fig. 2. Overall framework of UmambaTSF, where the top-left section illustrates the overall architecture of our model. The center-left and bottom-left sections introduce the multi-scale feature extractor and the Mamba-based temporal signal processor (MTSP), which are used to extract temporal features at each scale. The top-right section provides an explanation of the operation symbols, while the bottom-right section describes the structure of the Mamba model.

series, while the right side serves as a decoder, incrementally increasing the dimensional features of temporal data to deeply mine multi-scale characteristics.

The Mamba-based temporal signal processor, a core component of the multi-scale feature extractor, employs residual Mamba layers and channel-adaptable Mamba block to flexibly and efficiently extract long-range contextual features from each dimensional feature. These components are designed to extract periodic signals and trend information from time series data, while capturing interactions among multivariate elements under different scenario conditions.

B. U-shaped Multi-scale Feature Extractor

Time series data typically exhibit diverse periodicities and trends. Our predictive model, UmambaTSF, is based on a U-shaped architecture to extract multi-scale features, allowing it to effectively capture complex data fluctuations.

As illustrated in Fig. 2, prior to the multi-scale feature extraction module, the Linear tokenizer layer linearly maps the input data X with the length of L to a higher feature dimension M_1 (M_1 is the data dimension of the first layer in the multi-scale feature extractor). In the encoder part of the multi-scale feature extractor, the feature dimensions are progressively reduced through linear layers from $[M_1, M_2, \ldots, M_n]$, where $M_1 > M_2 > \cdots > M_n$, n represents the number of layers in the multi-scale feature extractor.

In the multi-scale feature extractor, X_i is the feature of the i-th layer in the downsampling process of the left-side encoder, from top to bottom. $X_m[i]$ is the temporal association at a certain scale, indicating the lateral skip connections of each layer. X_i and $X_m[i]$ can be defined as follows:

$$X_i = DO(Linear(X_{i-1}, M_{i-1} \rightarrow M_i)), \tag{1}$$

$$X_m[i] = \overrightarrow{MTSP}(X_i), \tag{2}$$

where $i \in \{1, 2, 3, \ldots, n\}$. $Linear$ is achieved through MLP. $M_{i-1} \rightarrow M_i$ represents the mapping from dimension M_{i-1} to dimension M_i. DO stands for the dropout operation. $\overrightarrow{MTSP}$ indicates that the features in each encoder-decoder layer are further processed by the Mamba-based temporal signal processor to extract temporal dependencies at a specific scale. X_0 and M_0 represent the initial time series X and input length L, respectively.

In the decoder part, the feature dimensions progressively increase through a series of expanding linear layers, with dimensions ranging from $[M_n, M_{n-1}, \ldots, M_1]$. X'_j represents the feature of the j-th layer in the upsampling process of the right-side decoder, from bottom to top. X'_0 is the value of the last layer X_m in the downsampling on the left side. If $j = 1$, X'_j is calculated by:

$$X'_1 = Linear(X'_0, M_n \rightarrow M_{n-1}) \tag{3}$$

The feature obtained from the decoder layer X' is concatenated with the corresponding layer's X_m along the temporal length dimension. Upsampling is then applied to linearly map the concatenated feature to the dimension of the previous layer, as shown in equations (4) and (5):

$$X'_j = Concat(X'_j, X_m[n-j]), \tag{4}$$

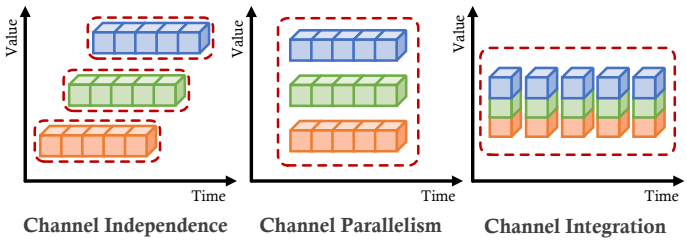


Fig. 3. Schematic Diagrams of Channel Processing in Three Different Scenarios.

$$X'_{j+1} = Linear(X'_j, 2 * M_{n-j} \rightarrow M_{n-j-1}), \tag{5}$$

where $j \in \{1, 2, \ldots, n-1\}$. When $j = n-1$, M_0 is the forecast length T. The projection layer maps the feature dimension of $2 \times M_1$ obtained from the U-shaped multi-scale feature extractor to the forecast length T.

C. Mamba-based Temporal Signal Processor

The structure of the MTSP is composed of fundamental Mamba blocks, representing a state space modeling method with the capability to capture any cyclic process within latent states. The processor consists of three paths: the first, known as residual Mamba layers, is designed to extract cyclical and trend information from time series data; the second, termed channel-adaptable Mamba, handles the mixed processing of multi-dimensional variables; and the third path directly processes the input time series data. The outputs from the three paths are summed, enabling the acquisition of unique temporal signals at each scale while facilitating the flexible processing of multi-channel data. Next, we introduce the fundamental **Mamba block**, followed by the **residual Mamba layers** and the **channel-adaptable Mamba block**.

*a) **Mamba Block**:* Within the Mamba block, as shown in the bottom right part of Fig. 2, both branches first undergo linear mapping. The first branch then proceeds through the one-dimensional causal convolution and SiLU [41] activation, followed by the structured SSM. This is combined with the activated residual connection of the second branch. Finally, the output is obtained through a linear transformation. Continuous-time SSM maps an input function or sequence $x(t)$ to output $y(t)$ through the latent state $h(t)$ as presented in equation (6):

$$\begin{aligned} dh(t)/dt &= \boldsymbol{A}h(t) + \boldsymbol{B}x(t) \\ y(t) &= \boldsymbol{C}h(t), \end{aligned} \tag{6}$$

where A, B, and C are learnable matrices, and the continuous sequence is discretized using a step size Δ. The discretized SSM model is illustrated in equation (7):

$$\begin{aligned} h_t &= \overline{\boldsymbol{A}}h_{t-1} + \overline{\boldsymbol{B}}x_t \\ y_t &= \boldsymbol{C}h_t \end{aligned} \tag{7}$$

To perform calculations using continuous recursive methods, discrete form $\overline{\boldsymbol{A}}$ and $\overline{\boldsymbol{B}}$ are obtained based on continuous form A and B by equation (8):

$$\overline{\boldsymbol{A}} = \exp(\Delta \boldsymbol{A}) \quad \overline{\boldsymbol{B}} = (\Delta \boldsymbol{A})^{-1}(\exp(\Delta \boldsymbol{A}) - I) \cdot \Delta \boldsymbol{B} \tag{8}$$

Additionally, Mamba offers an option to increase the model dimensions using a controllable expansion factor. This enables the model to use coefficient matrices that allow for selective propagation or forgetting of information along the input token sequence, depending on the context and the current token [16].

*b) **Residual Mamba Layers**:* As the residuals can preserve temporal information at different granularities [42], the residuals from multiple Mamba blocks are used to continuously identify the hidden trend information within the temporal features. The input to the first Mamba block is the original input feature, while its output is the reconstructed value corresponding to that input. For the remaining Mamba blocks, the input is the reconstruction value from the previous Mamba block minus the input value of that block. On one hand, the next block complements the previous one by continually fitting the residual information that the earlier block did not capture. On the other hand, this process can be viewed as a decomposition of the time series data, enabling continuous learning of trend information within the temporal dataset. The reconstructed value $rml[k]$ from the k-th Mamba block and residual-updated X_i are defined as follows:

$$rml[k] = Mamba(X_i), \tag{9}$$

$$X_i = rml[k] - X_i, \tag{10}$$

where $k \in \{1, 2, \ldots, K\}$, K represents the number of layers in the residual Mamba layer, and each layer is a standard Mamba Block.

*c) **Channel-Adaptable Mamba Block**:* The channel-adaptable Mamba block consists of a single Mamba designed to process multivariate time series feature information. It is structured as a flexible module to handle various scenarios, including channel independence, channel interdependence, and channel integration. In different scenarios, the input sequences are transformed into various shapes, which correspondingly alter the input dimension d_model setting within the Mamba block.

As shown in Fig. 3, when channels are independent, the dimensions of all time points for each variable are concatenated together and sequentially input as vectors into the Mamba block. When channels are parallel, tokens are formed by variables, with the distinction that multiple variables are input into the Mamba block simultaneously. During channel integration, variables at each time point are concatenated and sequentially fed into the Mamba block.

For datasets with fewer channels and lower complementarity, the channel independence method enables multivariate predictions unaffected by other variables. Conversely, when datasets have numerous channels with complex interrelations, the channel parallelism method reduces computational costs while incorporating interactions among channels. When multiple channels are strongly interrelated, the channel integration method emphasizes relationships between variables at each time step.

The input sequence X_i undergoes a transformation based on different channel processing methods, and is represented as T_X_i. The features cam obtained from the channel-adaptable Mamba block must ultimately be subjected to a dimensional transformation to T_cam to accommodate the

summation operation across the three paths. Therefore, cam and $\overrightarrow{MTSP}(X_i)$ are defined as follows:

$$cam = Mamba(T\_X_i) \tag{11}$$

$$\overrightarrow{MTSP}(X_i) = rml[K] + T\_cam + X_i \tag{12}$$

IV. Experimental Design

A. Datasets

Real-world datasets often contain noise and complex patterns that are absent in synthetic data. Time series modeling on these datasets helps improve the model's capability to handle and predict under realistic and frequently non-ideal conditions. We evaluate our model on seven real-world datasets, including Weather [6], Electricity (ECL) [6], Traffic [6], and four ETT [10] datasets (i.e., ETTh1, ETTh2, ETTm1, ETTm2). These datasets have been widely used for long-term time series forecasting. The Weather dataset comprises 21 meteorological factors, meticulously collected every 10 minutes throughout 2020 from the Weather Station of the Max Planck Biogeochemistry Institute. ECL captures the hourly electricity consumption of 321 clients. The Traffic dataset comprises hourly road occupancy rates recorded by 862 sensors on freeways in the San Francisco Bay Area from January 2015 to December 2016. The ETT dataset includes data on seven factors related to electricity transformers, spanning from July 2016 to July 2018. ETT is divided into four subsets: ETTh1 and ETTh2 (they contain hourly recordings), as well as ETTm1 and ETTm2, which feature recordings every 15 minutes. The relevant statistics for these datasets are shown in TABLE I.

TABLE I
THE STATISTICAL CHARACTERISTICS OVERVIEW OF THE SEVEN DATASETS.

Dataset	Channels	Time Points	Frequency
Weather	21	52696	10 Minutes
Traffic	862	17544	Hourly
Electricity	321	26304	Hourly
ETTh1	7	17420	Hourly
ETTh2	7	17420	Hourly
ETTm1	7	69680	15 Minutes
ETTm2	7	69680	15 Minutes

B. Experimental Setting and Metrics

Similar to most classic time series forecasting models, we divided the dataset into training, validation, and testing segments in a 7:2:1 ratio [6], [9], [30]. UmambaTSF is implemented using Pytorch and trained on an Ubuntu 20.04.6 system with an Nvidia A100 GPU (40GB). The input sequence length is fixed at $L = 96$, and the forecasting sequence length $T \in \{96, 192, 336, 720\}$. UmambaTSF utilizes the Adam optimizer combined with L2 loss, and the batch size is variable depending on the dataset, but all training sessions are consistent at 20 epochs. UmambaTSF's multi-scale feature extractor layers are set to 2 or 3, with 3 or 4 residual Mamba layers. The state expansion factor of each small Mamba is capped at 16. To ensure model reproducibility, the random seed is fixed. Mean Squared Error (MSE) and Mean Absolute Error (MAE) are employed to assess the effectiveness of long-term predictions. The calculation is shown in equations (13) and (14):

$$MSE = \frac{\sum_{i=1}^{N}(Y - \hat{Y})^2}{N} \tag{13}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}(Y - \hat{Y}), \tag{14}$$

where N represents the number of variables, Y and Y' refers to the ground truth and corresponding prediction, respectively.

C. Baseline Methods

In the experiments, 10 state-of-the-art (SOTA) models are selected as baselines, introduced as follows:

S-Mamba [17]: It employs a bidirectional Mamba layer to extract inter-variate correlations.
iTransformer [9]: It adapts the classic Transformer by swapping self-attention and FNN roles and encoding each time series variable as a token.
PatchTST [5]: It breaks down sequences into smaller patches for efficient processing, using a channel-wise approach.
Crossformer [34]: It uses a dual-phase attention mechanism for temporal and variable dimensions, boosting its ability to manage multivariate time series.
Autoformer [6]: It employs a deep decomposition architecture with $O(L \log L)$ complexity to mitigate sparse attention bottlenecks.
FEDformer [7]: It converts time-domain data to frequency-domain, using low-rank approximation to enhance performance and reduce computational complexity.
DLinear [13]: It shows that linear models can match or surpass Transformer architectures in time series forecasting.

TiDE [36]: It addresses long-term dependencies, noise, and uncertainties in time series forecasting.
RLinear [14]: It combines a single-layer linear model with RevIN Instance normalization, showing strong predictive performance across multiple datasets.
TimesNet [43]: It reshapes sequences in two-dimensional space to capture intra-cycle and inter-cycle variations.

Among these, the Mamba-based model is S-Mamba; the Transformer-based models include iTransformer, PatchTST, Crossformer, AutoFormer, and FEDformer; the MLP-based models consist of DLinear, TiDE, and RLinear; and the CNN-based method is TimesNet.

TABLE II

FULL RESULTS IN MSE AND MAE (THE LOWER THE BETTER) FOR THE LONG-TERM FORECASTING TASK. WE COMPARE EXTENSIVELY WITH BASELINES UNDER DIFFERENT PREDICTION LENGTHS, $T \in \{96, 192, 336, 720\}$. THE LENGTH OF THE INPUT SEQUENCE L IS SET TO 96 FOR ALL MODELS. THE BEST RESULTS ARE HIGHLIGHTED IN BOLD RED FONT, AND THE SECOND BEST ARE IN UNDERLINED BLUE FONT. THE RESULTS OF S-MAMBA ARE FROM S-MAMBA [17], WHILE THE RESULTS OF OTHER BASELINES ARE REPORTED BY ITRANSFORMER [9].

Methods		UmambaTS		S-Mamba		iTransformer		Rlinear		PatchTST		Crossformer		TiDE		TimesNet		Dlinear		FEDformer		Autoformer	
D	T	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	**0.157**	**0.204**	0.165	0.210	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	0.217	0.296	0.266	0.336
	192	**0.205**	**0.248**	0.214	0.252	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	0.276	0.336	0.307	0.367
	336	**0.251**	**0.288**	0.274	0.297	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	0.339	0.380	0.359	0.395
	720	**0.340**	**0.344**	0.350	0.345	0.358	0.349	0.364	0.353	0.354	0.348	0.398	0.418	0.351	0.386	0.365	0.359	0.345	0.381	0.403	0.428	0.419	0.428
Traffic	96	0.391	0.266	**0.382**	**0.261**	0.395	0.268	0.649	0.389	0.544	0.359	0.522	0.290	0.805	0.493	0.593	0.321	0.650	0.396	0.587	0.366	0.613	0.388
	192	0.413	0.274	**0.396**	**0.267**	0.417	0.276	0.601	0.366	0.540	0.354	0.530	0.293	0.756	0.474	0.617	0.336	0.598	0.370	0.604	0.373	0.616	0.382
	336	0.431	0.282	**0.417**	**0.276**	0.433	0.283	0.609	0.369	0.551	0.358	0.558	0.305	0.762	0.477	0.629	0.336	0.605	0.373	0.621	0.383	0.622	0.337
	720	0.466	0.302	**0.460**	**0.300**	0.467	0.302	0.647	0.387	0.586	0.375	0.589	0.328	0.719	0.449	0.640	0.350	0.645	0.394	0.626	0.382	0.660	0.408
ECL	96	0.140	0.236	**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	0.193	0.308	0.201	0.317
	192	**0.157**	**0.251**	0.159	0.255	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	0.201	0.315	0.222	0.334
	336	**0.173**	**0.268**	0.176	0.272	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	0.214	0.329	0.231	0.338
	720	**0.203**	**0.296**	0.204	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	0.246	0.355	0.254	0.361
ETTh1	96	**0.358**	**0.391**	0.386	0.405	0.386	0.405	0.386	0.395	0.414	0.419	0.423	0.448	0.479	0.464	0.384	0.402	0.386	0.400	0.376	0.419	0.449	0.459
	192	**0.415**	**0.420**	0.443	0.437	0.441	0.436	0.437	0.424	0.460	0.445	0.471	0.474	0.525	0.492	0.436	0.429	0.437	0.432	0.420	0.448	0.500	0.482
	336	**0.428**	**0.426**	0.489	0.468	0.487	0.458	0.479	0.446	0.501	0.466	0.570	0.546	0.565	0.515	0.491	0.469	0.481	0.459	0.459	0.465	0.521	0.496
	720	**0.454**	**0.456**	0.502	0.489	0.503	0.491	0.481	0.470	0.500	0.488	0.653	0.621	0.594	0.558	0.521	0.500	0.519	0.516	0.506	0.507	0.514	0.512
ETTh2	96	**0.274**	**0.334**	0.296	0.348	0.297	0.349	0.288	0.338	0.302	0.348	0.745	0.584	0.400	0.440	0.340	0.374	0.333	0.387	0.358	0.397	0.346	0.388
	192	**0.347**	**0.381**	0.376	0.396	0.381	0.400	0.374	0.390	0.388	0.400	0.877	0.656	0.528	0.509	0.402	0.414	0.477	0.476	0.429	0.439	0.456	0.452
	336	**0.340**	**0.381**	0.424	0.431	0.428	0.432	0.415	0.426	0.426	0.433	1.043	0.731	0.643	0.571	0.452	0.452	0.594	0.541	0.496	0.487	0.482	0.486
	720	**0.409**	**0.432**	0.426	0.444	0.427	0.445	0.420	0.440	0.431	0.446	1.104	0.763	0.874	0.679	0.462	0.468	0.831	0.657	0.463	0.474	0.515	0.511
ETTm1	96	**0.316**	**0.356**	0.333	0.368	0.334	0.368	0.355	0.376	0.329	0.367	0.404	0.426	0.364	0.387	0.338	0.375	0.345	0.372	0.379	0.419	0.505	0.475
	192	**0.356**	**0.378**	0.376	0.390	0.377	0.391	0.391	0.392	0.367	0.385	0.450	0.451	0.398	0.404	0.374	0.387	0.380	0.389	0.426	0.441	0.553	0.496
	336	**0.374**	**0.400**	0.408	0.413	0.426	0.420	0.424	0.415	0.399	0.410	0.532	0.515	0.428	0.425	0.410	0.411	0.413	0.413	0.445	0.459	0.621	0.537
	720	**0.440**	**0.435**	0.475	0.448	0.491	0.459	0.487	0.450	0.454	0.439	0.666	0.589	0.487	0.461	0.478	0.450	0.474	0.453	0.543	0.490	0.671	0.561
ETTm2	96	**0.175**	**0.256**	0.179	0.263	0.180	0.264	0.182	0.265	**0.175**	0.259	0.287	0.366	0.207	0.305	0.187	0.267	0.193	0.292	0.203	0.287	0.255	0.339
	192	**0.238**	**0.298**	0.250	0.309	0.250	0.309	0.246	0.304	0.241	0.302	0.414	0.492	0.290	0.364	0.249	0.309	0.284	0.362	0.269	0.328	0.281	0.340
	336	**0.286**	**0.331**	0.312	0.349	0.311	0.348	0.307	0.342	0.305	0.343	0.597	0.542	0.377	0.422	0.321	0.351	0.369	0.427	0.325	0.366	0.339	0.372
	720	**0.370**	**0.383**	0.411	0.406	0.412	0.407	0.407	0.398	0.402	0.400	1.730	1.042	0.558	0.524	0.408	0.403	0.554	0.522	0.421	0.415	0.433	0.432

TABLE III
THE IMPROVEMENT OF UMAMBATSF OVER ITRANSFORMER AND S-MAMBA, EVALUATED USING THE AVERAGE MSE AND MAE FOR FORECAST LENGTHS $T \in \{96, 192, 336, 720\}$ WITH AN INPUT LENGTH OF $L = 96$.

Models	iTransformer [9]		S-Mamba [17]		UmambaTSF **(ours)**		Promotion over iTransformer		Promotion over S-Mamba	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	0.258	0.278	0.251	0.276	0.238	0.271	7.75% ↑	2.52% ↑	5.18% ↑	1.81% ↑
Traffic	0.428	0.282	0.414	0.276	0.425	0.281	0.70% ↑	0.35% ↑	-2.66% ↓	-1.81% ↓
ECL	0.178	0.270	0.179	0.265	0.168	0.263	5.62% ↑	2.59% ↑	6.15% ↑	0.75% ↑
ETTh1	0.454	0.448	0.455	0.450	0.414	0.423	8.81% ↑	5.58% ↑	9.01% ↑	6.00% ↑
ETTh2	0.383	0.407	0.381	0.405	0.343	0.382	10.44% ↑	6.14% ↑	9.97% ↑	5.68% ↑
ETTm1	0.407	0.410	0.398	0.405	0.372	0.392	8.60% ↑	4.39% ↑	6.53% ↑	3.21% ↑
ETTm2	0.288	0.332	0.288	0.332	0.267	0.317	7.29% ↑	4.52% ↑	7.29% ↑	4.52% ↑

V. RESULTS AND ANALYSIS

A. Main Results

TABLE II presents the overall forecasting performance of UmambaTSF and the baseline models across the seven datasets. Based on the results in the table, we summarize the following observations and provide a detailed analysis:

(1). UmambaTSF outperforms existing SOTA models, including iTransformer and S-Mamba, on the Weather, ECL, ETTh1, ETTh2, ETTm1, and ETTm2 datasets. As shown in TABLE III, UmambaTSF achieves improvements in MSE of 7.75% and 5.18% over iTransformer and S-Mamba, respectively on the Weather dataset, along with improvements of 5.62% and 6.15% on the ECL dataset. Furthermore, on the four ETT datasets, the average improvements are 8.79% and 8.20%, respectively.

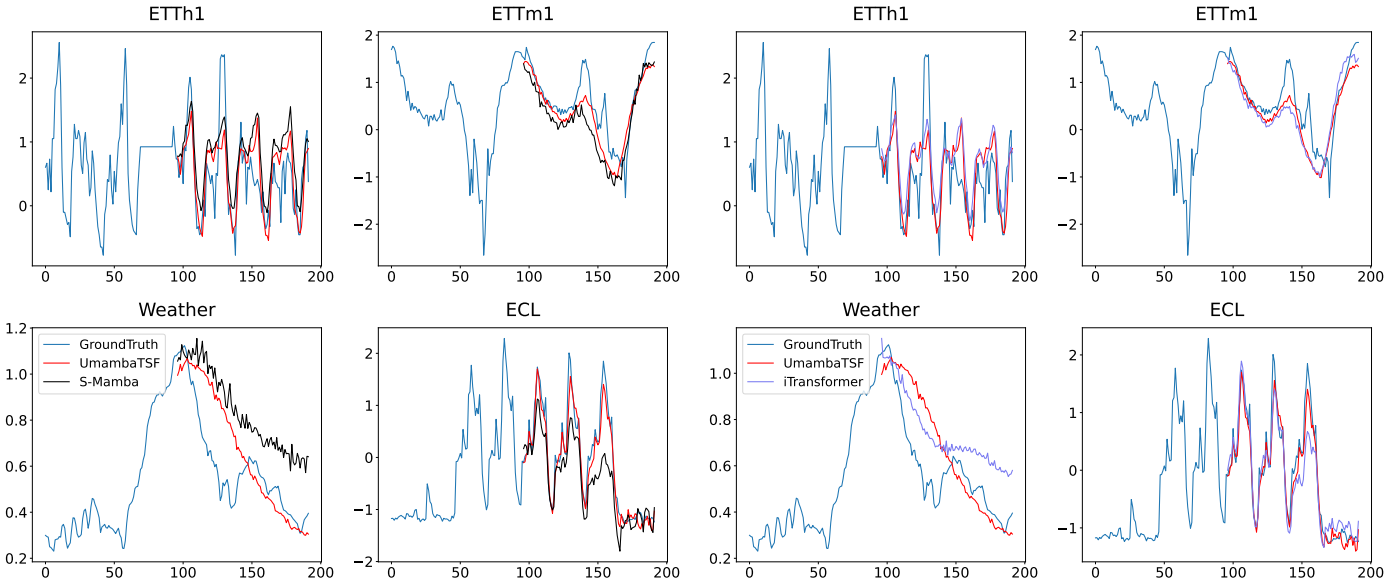
(2). UmambaTSF significantly exceeds existing SOTA models on both the channel-rich Weather and ECL datasets as well as the channel-sparse ETT datasets, demonstrating improvements in MSE and MAE. This enhancement is not only attributed to the multi-scale feature extractor's ability to capture temporal dependencies from multiple periodic signals but also to the flexible channel processing techniques provided by the proposed channel-adaptable approach in UmambaTSF. This accommodates transformations across various data scenarios and enhances the model's generalizability.

(3). UmambaTSF surpasses the results of iTransformer on the Traffic dataset but falls short of S-Mamba's performance. This is because UmambaTSF is designed to be highly minimalist and computationally efficient, accommodating both low and high channel counts. For the Traffic dataset, where directly using parallel channel methods involves too much noisy information, we plan to further design channel data filtering capabilities in future work.

Additionally, to more intuitively evaluate the predictive capability of UmambaTSF, we visually compare UmambaTSF with leading baseline models S-Mamba and iTransformer across four datasets (ETTh1, ETTm1, Weather, and ECL) through graphical representation. We randomly select a variable and input its lookback sequence to showcase the predicted results alongside the actual sequence trend. As shown in Fig. 4, the actual input and subsequent sequences are represented by a blue line, while the predictions from UmambaTSF, S-Mamba, and iTransformer are represented by red, black, and purple lines, respectively. It is evident that UmambaTSF's predictions closely align with the actual values, showing nearly perfect consistency on the ECL dataset, a significant advantage on the Weather dataset, and improved alignment with the extremes within the periodic sequences of the ETT datasets.

In time series prediction, time complexity is a crucial factor because it directly affects the scalability and efficiency of the model. Time complexity determines the computational time required to process data, which is particularly vital for real-time or near-real-time applications. The proposed method, UmambaTSF, leverages both the linearly scalable linear layers and the Mamba structure, achieving a time complexity of $O(L)$. This efficiency is crucial for applications that require immediate responses, such as stock trading and weather forecasting, where the model must process data rapidly. However, the mainstream algorithms for time series are still based on the transformer structure. The original Transformer model [4], relying on full self-attention mechanisms, has a time complexity of $O(L^2)$, significantly increasing computational costs for long sequence data. To address this challenge, Informer [10] employs sparse self-attention techniques, reducing necessary computations through probabilistic sparsification and entropy coding, thus lowering the time complexity to $O(LlogL)$. PatchTST [5] segments time series data into blocks and applies the Transformer structure to each block, effectively distributing the computational load and enhancing processing efficiency, despite a time complexity of $O((L/p)^2)$, where p denotes the patch size. These innovations are particularly crucial for large-scale and real-time prediction needs, each balancing computational resource consumption and predictive performance in different ways.

Fig. 5 presents a comparison of the prediction accuracy, time complexity, and GPU memory usage of UmambaTSF, S-Mamba, iTransformer, PatchTST, Informer, and Transformer models on the ETTh1 and Weather datasets. The specific memory usage values are shown in TABLE IV. Note that the complexities $O((L/p)^2)$ and $O(LlogL)$ are presented here just for an illustrative purposes. As shown in the figure, UmambaTSF achieves commendable results in both accuracy and computational cost with its purely linear complexity.



(a) Forecasting comparisons between UmambaTSF and S-Mamba (b) Forecasting comparisons between UmambaTSF and iTransformer

Fig. 4. Comparison of forecasts among UmambaTSF, iTransformer, and S-Mamba on four datasets, using an input length of 96 and a forecast length of 96. The blue line depicts the ground truth, while the red, black, and purple lines represent the predictions from UmambaTSF, S-Mamba, and iTransformer, respectively.

TABLE IV
MEMORY USAGE (GIB) OF UMAMBATSF, S-MAMBA, AND ITRANSFORMER WHEN BOTH INPUT LENGTH AND FORECAST LENGTH ARE 96.

Model	UmambaTSF	S-Mamba	iTransformer
ETTh1	1.48	1.98	1.93
Weather	1.67	3.77	3.62
ECL	1.78	2.51	2.61

TABLE V
RESULTS OF THE ABLATION STUDY ON ETTH1, ETTM1 AND WEATHER DATASETS. MSE AND MAE ARE THE AVERAGE VALUES ACROSS FORECAST LENGTHS T.

ULL	RML	CAM	Weather		ETTh1		ETTm1	
			MSE	MAE	MSE	MAE	MSE	MAE
×	×	×	0.266	0.290	0.432	0.430	0.410	0.403
✓	✓	×	0.248	0.277	0.414	0.423	0.373	0.393
✓	×	✓	0.244	0.274	0.430	0.428	0.408	0.403
✓	×	×	0.266	0.29	0.532	0.490	0.520	0.467
*	✓	✓	0.241	0.274	0.423	0.427	0.372	0.392
✓	*	✓	0.245	0.276	0.428	0.430	0.379	0.397
✓	✓	✓	0.238	0.271	0.414	0.423	0.372	0.392

B. Ablation Study

To better evaluate the functionality and effectiveness of each module in UmambaTSF, we divide the Multi-Scale Feature Extractor into three components: 1) U-shape Linear Layers (ULL); 2) Residual Mamba Layers (RML); and 3) Channel Adaptable Mamba Block (CAM). As shown in TABLE V, the "×" and "✓" indicate the absence or presence of a specific module, respectively. When ULL is marked with an "*", it indicates that both the encoding and decoding layers of ULL consist of only one layer. When RML is marked with an "*", it signifies that the RML contains only one Mamba block. Due to the coupling relationships between the model components, the RML and CAM modules cannot exist independently of ULL, unless all three modules are absent. All "×"s represent the model without the Multi-Scale Feature Extractor, while all "✓"s indicate the complete UmambaTSF model.

Across multiple experiments, UmambaTSF consistently delivered the best results, with each module contributing uniquely to its overall performance. Notably, the CAM module significantly improved results on the weather dataset, where a high degree of inter-variable correlation is present. On the other hand, the RML module proved particularly important for the ETTh1 and ETTm1 datasets, due mainly to the relatively small number of channels in the ETT series, where the temporal feature information of each channel plays a more critical role.

The model that includes only the ULL module, without RML and CAM, and lacks the concatenation operation in the decoder on the right side of ULL, performs the worst. This indicates that multi-scale representation alone, without detailed temporal signals at each scale, is detrimental to the predictive model. However, when the U-shape structure contains only one layer that includes RML and CAM, or when the RML consists of just one Mamba block, the model still produces reasonable results, though these are not as strong as those achieved by the full UmambaTSF. This demonstrates that the multi-layer feature extraction and the method of using multiple Mamba blocks for residual learning effectively enhance the temporal information, improving performance.

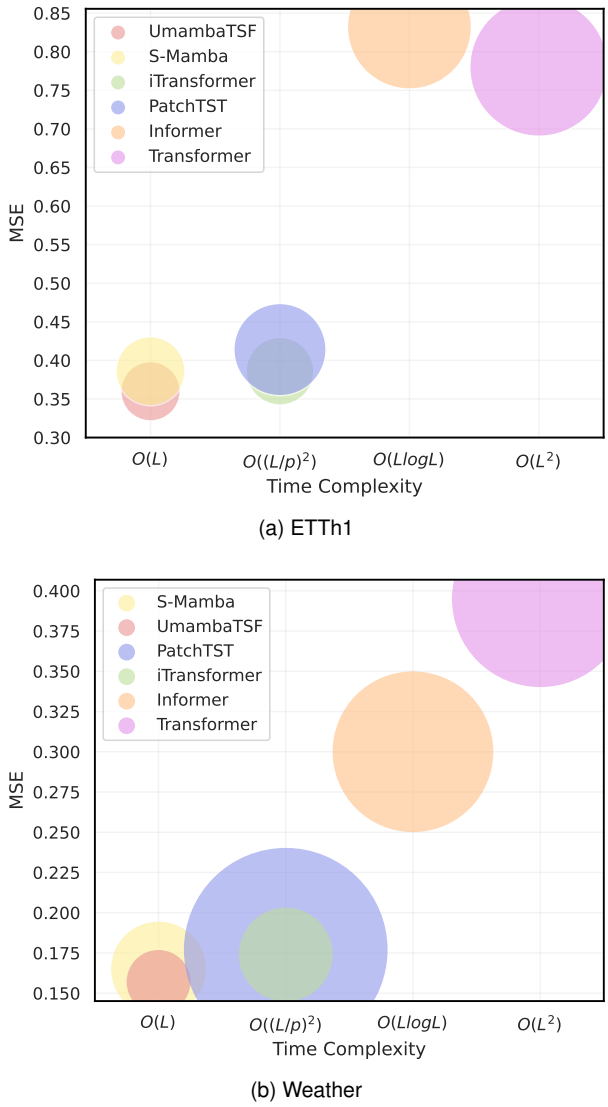


Fig. 5. Comparison of UmambaTSF and five models on MSE, Computational Complexity, and GPU Memory. The diameters of the circles represent the memory sizes.

C. Hyperparameter Sensitivity Analysis

As illustrated in the overall framework depicted in Fig. 2, our multi-scale feature extraction architecture is primarily composed of linear layers. The encoder begins with a dimension being larger than the input sequence length L and progressively reduces the length. The decoder's scale incrementally rises and merges the temporal dependencies extracted from the same layer by the mamba block. To assess the impact of multi-scale configurations, we conduct experiments on the ETTh1 and ETTh2 datasets with various scale settings, as shown in Fig. 6. The results reveal that model performance remains relatively stable across different configurations, with both datasets achieving strong performance under multiple settings. This suggests that once the U-shaped architecture is established, there are several viable scale combinations, and prediction accuracy is not reliant on any single specific configuration.

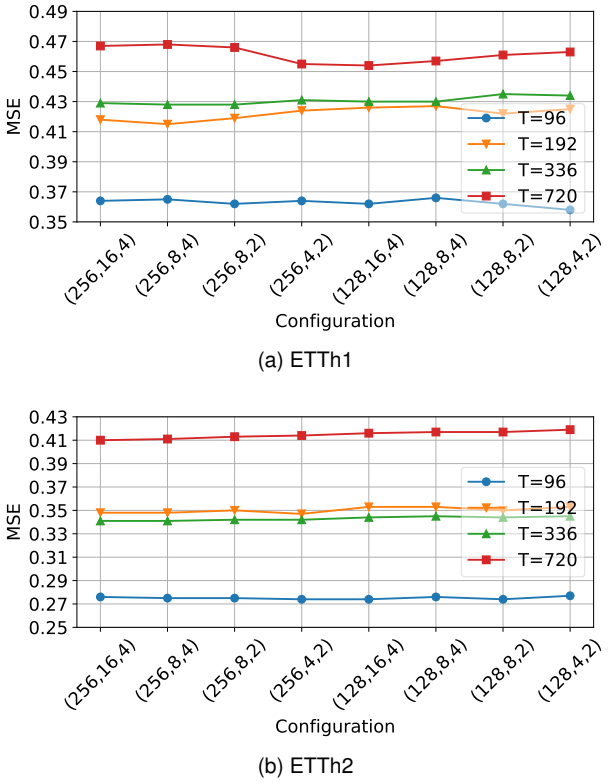


Fig. 6. MSE comparison with combinations of multi-scale feature dimensions for input sequence length $L = 96$ and forecast length $T \in \{96, 192, 336, 720\}$ for the ETTh1 and ETTh2 datasets.

D. The Generality of UmambaTSF

1) Increasing Lookback Length: Previous studies have shown that the forecasting performance of models like Transformers does not necessarily improve with an increasing lookback length due to the problem of distracted attention from the expanding input [5], [13]. To validate whether our linear-complexity model, UmambaTSF, maintains its advantage across various lookback lengths, we conduct comparative analyses with three recent advanced models: S-Mamba, iTransformer, and TimesNet. These are tested on the ETTh1 and Weather datasets with lookback lengths set at 48, 96, 192, 336, and 720, and a forecasting window of 96. The results, depicted in Fig. 7, indicate that UmambaTSF consistently performs best on these datasets across all tested input windows. Notably, on the Weather dataset, as the lookback length increases, there is a corresponding enhancement in forecasting accuracy. This improvement is attributed not only to Mamba's ability to address long-range dependency issues but also to the MLP-based linear layers, which are better suited for leveraging larger historical datasets.

2) Model Robustness: Time series forecasting is conducted on recorded sequential data, which typically does not allow for large-scale training sets. Solutions based on the Transformer architecture are often contested due to insufficient data volumes necessary for training robust models. To validate the robustness of our method, we conduct experiments on the Weather, ECL, and Traffic datasets to compare the performance of UmambaTSF, S-Mamba, and iTransformer across

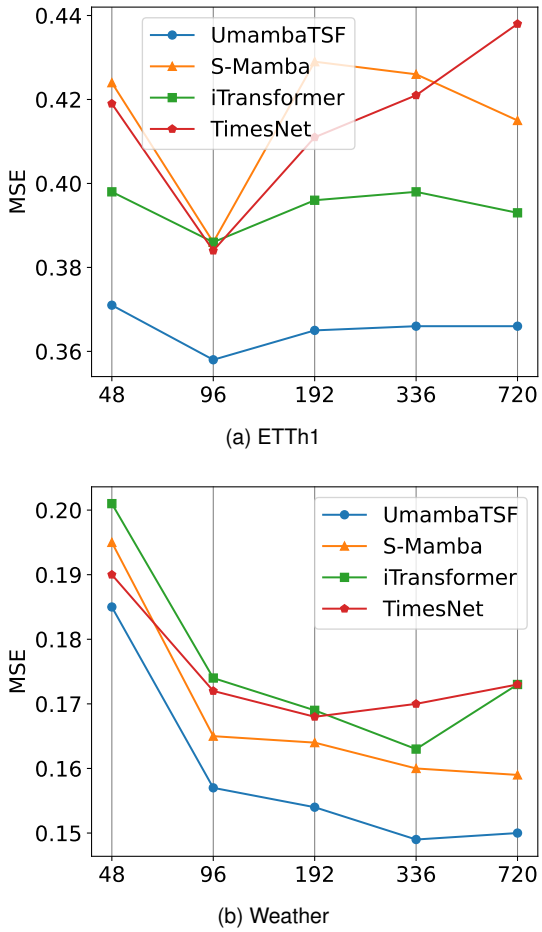


Fig. 7. Forecast performance with input length $L \in \{48, 96, 192, 336, 720\}$ and fixed prediction length $T = 96$. UmambaTSF demonstrates a significant advantage in MSE on the ETTh1 dataset, while on the Weather dataset, UmambaTSF gains more performance improvement from the extended look-back window length.

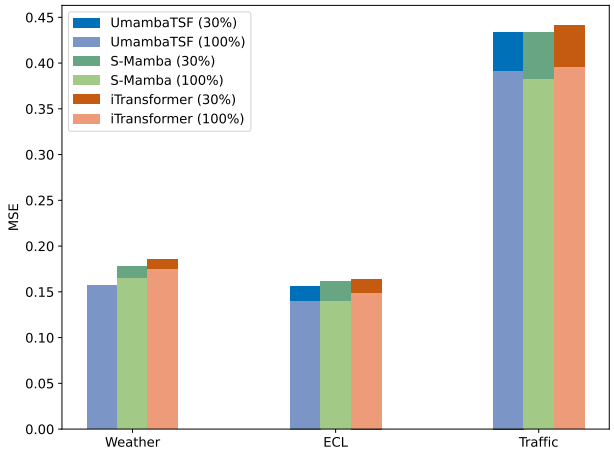


Fig. 8. Forecasting performance comparison among UMambaTSF, S-Mamba and iTransformer trained on 100% samples with on 30% samples. The lookback length $L = 96$ and the forecast length $T = 96$ for all datasets.

the full dataset and with 30% of the original dataset. The results, depicted in Fig. 8, demonstrate that UmambaTSF maintains strong robustness, consistently outperforming or matching the advanced models iTransformer and S-Mamba, even when the dataset is reduced to only 30%. Remarkably, on the Weather dataset, its performance using only 30% of the data is identical to that achieved with the full dataset. This superior performance is likely attributed to UmambaTSF's powerful feature extraction capabilities and its streamlined, effective architecture.

VI. CONCLUSION

The goal of this paper is to achieve a win-win outcome in terms of prediction accuracy and computational cost for long-term time series forecasting using a linearly scalable approach. This goal has been realized by the proposed innovative method, UmambaTSF. The model employs a multi-scale feature extraction module, capable of capturing time series information across various scales. Integrating residual Mamba layers and a multi-scenario flexible transformation Mamba module, it significantly enhances both accuracy and versatility.

In the experiments, we evaluate UmambaTSF on the 7 real-world datasets against 10 advanced baseline models. The results demonstrate that UmambaTSF achieves SOTA performance while maintaining computational costs low. Moreover, its compact and efficient design further enhances the model's generalization capability. This paper highlights Mamba's potential for time series forecasting from the perspectives of predictive performance, computational efficiency, and generalization ability. In the future, we plan to explore more channel processing techniques to further improve the accuracy of the forecasting model in noisy, multivariate environments.

REFERENCES

[1] M. Liu, A. Zeng, M. Chen, Z. Xu, Q. Lai, L. Ma, and Q. Xu, "Scinet: Time series modeling and forecasting with sample convolution and interaction," *Advances in Neural Information Processing Systems*, vol. 35, pp. 5816–5828, 2022.

[2] J. Su, D. Xie, Y. Duan, Y. Zhou, X. Hu, and S. Duan, "Mdcnet: Long-term time series forecasting with mode decomposition and 2d convolution," *Knowledge-Based Systems*, p. 111986, 2024.

[3] D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski, "Deepar: Probabilistic forecasting with autoregressive recurrent networks," *International journal of forecasting*, vol. 36, no. 3, pp. 1181–1191, 2020.

[4] A. Vaswani, "Attention is all you need," *Advances in Neural Information Processing Systems*, 2017.

[5] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A time series is worth 64 words: Long-term forecasting with transformers," in *The Eleventh International Conference on Learning Representations*, 2023. [Online].

[6] H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting," *Advances in neural information processing systems*, vol. 34, pp. 22 419–22 430, 2021.

[7] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, "Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting," in *International conference on machine learning*. PMLR, 2022, pp. 27 268–27 286.

[8] Y. Liu, H. Wu, J. Wang, and M. Long, "Non-stationary transformers: Exploring the stationarity in time series forecasting," *Advances in Neural Information Processing Systems*, vol. 35, pp. 9881–9893, 2022.

[9] Y. Liu, T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma, and M. Long, "itransformer: Inverted transformers are effective for time series forecasting," in *The Twelfth International Conference on Learning Representations*, 2024.

[10] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 12, 2021, pp. 11 106–11 115.

[11] S. Liu, H. Yu, C. Liao, J. Li, W. Lin, A. X. Liu, and S. Dustdar, "Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting," in *International conference on learning representations*, 2021.
[12] H. Wu, J. Wu, J. Xu, J. Wang, and M. Long, "Flowformer: Linearizing transformers with conservation flows" in *Proceedings of the 39th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., vol. 162. PMLR, 17–23 Jul 2022, pp. 24 226–24 242.
[13] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting?" in *Proceedings of the AAAI conference on artificial intelligence*, vol. 37, no. 9, 2023, pp. 11 121–11 128.
[14] Z. Li, S. Qi, Y. Li, and Z. Xu, "Revisiting long-term time series forecasting: An investigation on linear mapping," *arXiv preprint arXiv:2305.10721*, 2023.
[15] Z. Wang, S. Ruan, T. Huang, H. Zhou, S. Zhang, Y. Wang, L. Wang, Z. Huang, and Y. Liu, "A lightweight multi-layer perceptron for efficient multivariate time series forecasting," *Knowledge-Based Systems*, vol. 288, p. 111463, 2024.
[16] A. Gu and T. Dao, "Mamba: Linear-time sequence modeling with selective state spaces," *arXiv preprint arXiv:2312.00752*, 2023.
[17] Z. Wang, F. Kong, S. Feng, M. Wang, H. Zhao, D. Wang, and Y. Zhang, "Is mamba effective for time series forecasting?" *arXiv preprint arXiv:2403.11144*, 2024.
[18] M. A. Ahamed and Q. Cheng, "Timemachine: A time series is worth 4 mambas for long-term forecasting," *arXiv preprint arXiv:2403.09898*, 2024.
[19] A. Behrouz and F. Hashemi, "Graph mamba: Towards learning on graphs with state space models," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024, pp. 119–130.
[20] L. Zhu, B. Liao, Q. Zhang, X. Wang, W. Liu, and X. Wang, "Vision mamba: Efficient visual representation learning with bidirectional state space model," in *Forty-first International Conference on Machine Learning*, 2024.
[21] J. Liu, H. Yang, H.-Y. Zhou, Y. Xi, L. Yu, Y. Yu, Y. Liang, G. Shi, S. Zhang, H. Zheng *et al.*, "Swin-umamba: Mamba-based unet with imagenet-based pretraining," *arXiv preprint arXiv:2402.03302*, 2024.
[22] B. N. Patro and V. S. Agneeswaran, "Simba: Simplified mamba-based architecture for vision and multivariate time series," *arXiv preprint arXiv:2403.15360*, 2024.
[23] H. Zhao, M. Zhang, W. Zhao, P. Ding, S. Huang, and D. Wang, "Cobra: Extending mamba to multi-modal large language model for efficient inference," *arXiv preprint arXiv:2403.14520*, 2024.
[24] Y. Qiao, Z. Yu, L. Guo, S. Chen, Z. Zhao, M. Sun, Q. Wu, and J. Liu, "Vl-mamba: Exploring state space models for multimodal learning," *arXiv preprint arXiv:2403.13600*, 2024.
[25] A. Ali, I. Zimerman, and L. Wolf, "The hidden attention of mamba models," *arXiv preprint arXiv:2403.01590*, 2024.
[26] R. A. Angryk, P. C. Martens, B. Aydin, D. Kempton, S. S. Mahajan, S. Basodi, A. Ahmadzadeh, X. Cai, S. Filali Boubrahimi, S. M. Hamdi *et al.*, "Multivariate time series dataset for space weather data analytics," *Scientific data*, vol. 7, no. 1, p. 227, 2020.
[27] H. Wang, J. Peng, F. Huang, J. Wang, J. Chen, and Y. Xiao, "MICN: Multi-scale local and global context modeling for long-term series forecasting," in *The Eleventh International Conference on Learning Representations*, 2023.
[28] S. Feng, C. Miao, K. Xu, J. Wu, P. Wu, Y. Zhang, and P. Zhao, "Multi-scale attention flow for probabilistic time series forecasting," *IEEE Transactions on Knowledge and Data Engineering*, 2023.
[29] Z. Yue, Y. Wang, J. Duan, T. Yang, C. Huang, Y. Tong, and B. Xu, "Ts2vec: Towards universal representation of time series," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 8, 2022, pp. 8980–8987.
[30] T. Dai, B. Wu, P. Liu, N. Li, J. Bao, Y. Jiang, and S.-T. Xia, "Periodicity decoupling framework for long-term series forecasting," in *The Twelfth International Conference on Learning Representations*, 2024.
[31] T. Zhou, P. Niu, X. Wang, L. Sun, and R. Jin, "One fits all: Power general time series analysis by pretrained LM," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.
[32] B. N. Oreshkin, D. Carpov, N. Chapados, and Y. Bengio, "N-beats: Neural basis expansion analysis for interpretable time series forecasting," in *International Conference on Learning Representations*, 2020.
[33] S. Li, X. Jin, Y. Xuan, X. Zhou, W. Chen, Y.-X. Wang, and X. Yan, "Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting," *Advances in neural information processing systems*, vol. 32, 2019.
[34] Y. Zhang and J. Yan, "Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting," in *The Eleventh International Conference on Learning Representations*, 2023.
[35] V. Ekambaram, A. Jati, N. Nguyen, P. Sinthong, and J. Kalagnanam, "Tsmixer: Lightweight mlp-mixer model for multivariate time series forecasting," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023, pp. 459–469.
[36] A. Das, W. Kong, A. Leach, S. K. Mathur, R. Sen, and R. Yu, "Long-term forecasting with tiDE: Time-series dense encoder," *Transactions on Machine Learning Research*, 2023.
[37] A. Gu, K. Goel, and C. Ré, "Efficiently modeling long sequences with structured state spaces," *arXiv preprint arXiv:2111.00396*, 2021.
[38] J. T. Smith, A. Warrington, and S. W. Linderman, "Simplified state space layers for sequence modeling," *arXiv preprint arXiv:2208.04933*, 2022.
[39] T. Kim, J. Kim, Y. Tae, C. Park, J.-H. Choi, and J. Choo, "Reversible instance normalization for accurate time-series forecasting against distribution shift," in *International Conference on Learning Representations*, 2022.
[40] X. Ma, X. Li, L. Fang, T. Zhao, and C. Zhang, "U-mixer: An unet-mixer architecture with stationarity correction for time series forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 13, 2024, pp. 14 255–14 262.
[41] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," *Neural networks*, vol. 107, pp. 3–11, 2018.
[42] M. Hou, C. Xu, Z. Li, Y. Liu, W. Liu, E. Chen, and J. Bian, "Multi-granularity residual learning with confidence estimation for time series prediction," in *Proceedings of the ACM Web Conference 2022*, 2022, pp. 112–121.
[43] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, "Timesnet: Temporal 2d-variation modeling for general time series analysis," in *The Eleventh International Conference on Learning Representations*, 2023.

Li Wu received her M.Eng. degree from Tsinghua University in Beijing, China. She is currently pursuing her Ph.D. at Dalian University of Technology. Her primary research interests include time series modeling, spatiotemporal forecasting, and weather prediction. She is also a lecturer at Qinghai University and has published more than 10 papers, including in PPoPP and ICASSP. She is currently leading the National Natural Science Foundation project on "Precipitation Forecasting Methods Based on Multi-Source Data Using Deep Learning and Its Application in the Sanjiangyuan Region", and participated in several national key research and development programs.

Wenbin Pei received her Ph.D. degree at Victoria University of Wellington, New Zealand, in 2021. She is currently an assistant professor at Dalian University of Technology. Her research interests include evolutionary computation and machine learning. She has published more than 30 papers in fully refereed international journals and conferences, including IEEE Transactions on Evolutionary Computation and AAAI. She was a program committee member for 6 international conferences and a co-chair of special sessions in 4 international conferences. She has been serving as reviewers for international journals, including IEEE Transactions on Evolutionary Computation, IEEE Transactions on Cybernetics, Evolutionary Computation journal (MIT press), IEEE Transactions on Emerging Topics in Computational Intelligence, Applied Soft Computing, and Artificial Intelligence in Medicine.

JiuLong Jiao received his B.Eng. and M.Eng. degrees from Harbin Institute of Technology. He is currently pursuing his Ph.D. at Dalian University of Technology. His primary research focuses on machine learning and multimodal few-shot learning.

Qiang Zhang received the Ph.D. degree in Circuits and Systems from Xidian University, Xi'an, in 2002. He is currently a professor and the dean of the School of Computer Science and Technology, at Dalian University of Technology. His research interest includes bio-inspired computing and the related applications. Prof. Zhang has published more than 70 papers in fully refereed international journals and conferences. He was awarded National Science Fund for Distinguished Young Scholars in 2014, and was also selected as one of the state department special allowance experts. He has been serving as editorial board for 7 international journals and chairs of special issues in journals, such as Neurocomputing and International Journal of Computer Applications in Technology.