A MAMBA FOUNDATION MODEL FOR TIME SERIES FORECASTING

Haoyu Ma, Yushu Chen [*]**, Wenlai Zhao, Jinzhe Yang** [†]
Department of Computer Science and Technology
Tsinghua University
Beijing, China, 100084

Yingsheng Ji
Peng Cheng Laboratary
Shenzhen, China, 518000

Xinghua Xu
Naval University of Engineering,
Wuhan, China, 430033

Xiaozhu Liu
Beijing Institute of Technology
Beijing, China, 100081

Hao Jing
Earth System Modeling and Prediction Center
China Meteorological Administration
Beijing, China, 100081

Shengzhuo Liu
College of Computer Science and Mathematics
Fujian University of Technology
Fuzhou, China, 350118

Guangwen Yang [‡]
Department of Computer Science and Technology
Tsinghua University
Beijing, China, 100084

ABSTRACT

Time series foundation models have demonstrated strong performance in zero-shot learning, making them well-suited for predicting rapidly evolving patterns in real-world applications where relevant training data are scarce. However, most of these models rely on the Transformer architecture, which incurs quadratic complexity as input length increases. To address this, we introduce TSMamba, a linear-complexity foundation model for time series forecasting built on the Mamba architecture. The model captures temporal dependencies through both forward and backward Mamba encoders, achieving high prediction accuracy. To reduce reliance on large datasets and lower training costs, TSMamba employs a two-stage transfer learning process that leverages pretrained Mamba LLMs, allowing

[*] First Author and Second Author contribute equally to this work. Email: chenyushu@mail.tsinghua.edu.cn

[†] Also at National Supercomputing Center and Tecorigin in Wuxi, Jiangsu, China.

[‡] Corresponding author. Also at National Supercomputing Center in Wuxi, Jiangsu, China, and Zhejiang Lab, Hongzhou, China.

effective time series modeling with a moderate training set. In the first stage, the forward and backward backbones are optimized via patch-wise autoregressive prediction; in the second stage, the model trains a prediction head and refines other components for long-term forecasting. While the backbone assumes channel independence to manage varying channel numbers across datasets, a channel-wise compressed attention module is introduced to capture cross-channel dependencies during fine-tuning on specific multivariate datasets. Experiments show that TSMamba's zero-shot performance is comparable to state-of-the-art time series foundation models, despite using significantly less training data. It also achieves competitive or superior full-shot performance compared to task-specific prediction models. The code will be made publicly available.

1 Introduction

Time series forecasting predicts future data based on historical chronological information, offering a valuable tool for anticipating changes, formulating strategies, and mitigating risks. This technique is widely used across various sectors, including energy, finance, healthcare, manufacturing, retail, and traffic management.

Given the dynamic and ever-evolving nature of real-world data, forecasting models should be capable of adapting to changing patterns. However, traditional supervised models trained or even designed for each individual dataset or tasks (referred to as specialized models hereinafter), which are commonly used for time series forecasting, are often static and struggle to accommodate evolving patterns. This issue stems from three main challenges: first, these models require specific datasets for training, yet relevant data for emerging patterns may be unavailable or difficult to collect; second, they lack the ability to generalize across different datasets or applications, making it expensive and time-consuming to adapt models from one domain to another; and third, they often exhibit low data efficiency, increasing the risk of overfitting when training data are limited.

In contrast, time series foundation models, which are pretrained on vast domains of data, have demonstrated strong generalization capabilities across a wide range of scenarios and tasks. These models also exhibit high data efficiency in fine-tuning, enabling them to adapt to specific datasets with minimal samples. Such advantages make them effective for forecasting emerging patterns in web data, even when relevant data are unavailable or scarce. Comparison of specialized models and time series foundation models are presented in figure 1.

The development of foundational models for time series forecasting draws inspiration from the success of large language models (LLMs, e.g., Devlin et al. 2018; Brown et al. 2020; Touvron et al. 2023) in natural language processing (NLP), though it faces additional challenges.

The first challenge is the significant heterogeneity of time series data. Data from different domains exhibit diverse dependencies, and data collected at varying frequencies or sampling rates present distinct patterns. Additionally, data gathered from different devices show varying noise levels. Multivariate time series also differ in their channel characteristics, with each dataset potentially containing a different number of channels. In contrast, text data for NLP models generally do not involve concepts like frequencies or sampling rates and typically consist of a single channel.

Secondly, acquiring large-scale time series data is more challenging than in NLP. For example, the Large-scale Open Time Series Archive (LOTSA, Woo et al. 2024), the largest publicly available time series dataset to our knowledge, contains around 27 billion time points, whereas large NLP datasets, such as RedPajama-Data-v2 Together (2023), include tens of trillions of tokens.

Lastly, training foundational models for time series on large datasets imposes enormous computational demands, leading to long training times and high resource consumption. As a result, the process is both time-intensive and costly, requiring a significant budget for computing power.

Motivated by these challenges, some emerging time series foundation models (e.g., Zhou et al. 2023; Jin et al. 2024; Liu et al. 2024a;b) leverage existing LLMs and adapt them to time series tasks through cross-modality transfer learning. This approach allows these models to harness the knowledge acquired during LLM pretraining on vast datasets using significant computational resources. Other models are trained directly on large-scale time series datasets (e.g., Garza & Mergenthaler-

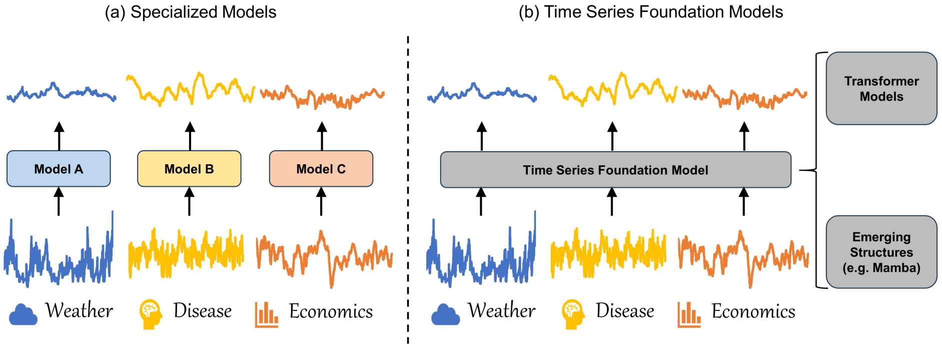

Figure 1: Comparison of specialized time series models and foundation models: (a) Specialized models are trained separately for specific tasks using relevant datasets. These models lack the ability to generalize across different domains and frequencies. (b) Time series foundation models, trained on large datasets, generalize well across a wide range of scenarios and tasks.

Canseco 2023; Das et al. 2023b; Woo et al. 2024), many of which are collected by the researchers themselves. However, several of the largest and most valuable datasets remain inaccessible to the public, and training with such large datasets also results in enormous computational costs.

Existing time series foundation models are predominantly based on the Transformer architecture Vaswani et al. (2017), which suffers from two main drawbacks: quadratic complexity with respect to input length and a lack of inductive biases Ascoli et al. (2021), which are advantageous for leveraging the chronological order of data. Recently, structured state space sequence models (SSMs) Gu et al. (2021; 2022) have emerged as an efficient approach for sequence modeling, offering linear complexity. Mamba Gu & Dao (2023) further enhances these SSMs by making the parameters functions of the inputs, allowing the model to selectively propagate or forget information along the sequence dimension based on the current data. Additionally, Mamba implements a hardware-aware parallel algorithm for efficient computation. The model achieves performance comparable to Transformers in NLP Waleffe et al. (2024) and CV Zhu et al. (2024), offering a strong alternative architecture for time series foundation models.

This paper introduces TSMamba, a time series foundation model based on the Mamba architecture for multivariate forecasting. The model combines forward and backward Mamba encoders to capture temporal dependencies with linear complexity, achieving high predictive accuracy. To address the challenges of limited dataset sizes and training budgets, we propose a two-stage transfer learning process that leverages knowledge from large-scale pretraining of Mamba LLMs, allowing efficient adaptation to the time series modality. Additionally, while the pretrained model assumes channel independence (CI) to handle varying channel numbers, we introduce a compressed cross-channel attention module to capture cross-channel dependencies during fine-tuning on specific datasets.

Our key contributions are as follows:

- We propose TSMamba, a linear-complexity foundation model for time series forecasting, applicable to prediction tasks across different domains and frequencies.
- The two-stage transfer learning process allows the model to leverage relationships distilled from large-scale LLM pretraining, enabling effective adaptation to time series data while mitigating the need for large datasets and extensive training costs.
- We introduce a channel-wise compressed attention module that enables the model to extract cross-channel dependencies during fine-tuning, outperforming channel independent approaches in most datasets and settings.
- The model achieves state-of-the-art (SOTA) performance on multiple mainstream datasets in zero-shot and full-shot forecasting scenarios.

2 Related Work

Over the last decade, time series forecasting has evolved from traditional statistical approaches to more advanced deep neural network-based techniques. Traditional models such as Autoregressive (AR), ARIMA, and VAR (Box et al., 2015), as well as kernel methods (Chen et al., 2008) and Gaussian processes (Frigola & Rasmussen, 2014), have been widely used. However, the advent of deep learning, fueled by the rapid evolution of computing capabilities and neural network architectures, has marked a paradigm shift in this field.

Various deep neural network architectures, including recurrent neural networks (RNNs), convolutional neural networks (CNNs), graph neural networks (GNNs), multi-layer perceptrons (MLPs), and Transformers, have been extensively applied to time series forecasting. RNNs (e.g., Hochreiter & Schmidhuber 1997; Qin et al. 2017; Rangapuram et al. 2018; Salinas et al. 2020), specifically designed for sequential data, were among the first deep learning models utilized in this domain. However, RNNs encountered challenges such as vanishing gradients Pascanu et al. (2012), which limited their effectiveness in capturing long-term dependencies. Similarly, CNNs, originally developed for image processing, were adapted to achieve state-of-the-art performance in time series forecasting (Bai et al., 2018; Borovykh et al., 2017; Sen et al., 2019; Wang et al., 2023; Wu et al., 2022; Donghao & Xue, 2024). While CNNs excel at identifying local patterns, their limited receptive field size constrained their ability to capture long-term dependencies. GNNs are increasingly being employed to enhance the recognition of both temporal and dimensional patterns in time series data (Wu et al., 2020; Cao et al., 2020). Additionally, recent advancements (Challu et al., 2023; Li et al., 2023b; Zeng et al., 2022; Das et al., 2023a; Ekambaram et al., 2023) suggest that MLP-based architectures remain competitive in forecasting tasks.

Following the remarkable success of Transformers (Vaswani et al., 2017) in NLP (Kalyan et al., 2021), computer vision (Khan et al., 2021), and speech processing (Karita et al., 2019), Transformers have become a mainstream approach in time series forecasting, delivering promising results (Wen et al., 2022). Their attention mechanisms are highly effective at capturing long-term dependencies. However, Transformers suffer from quadratic complexity in both computation and memory. Additionally, their flexibility leads to a lack of certain inductive biases Ascoli et al. (2021), which are beneficial for extracting sequential temporal dependencies.

Various Transformer-based approaches have been proposed to simultaneously enhance forecasting performance and reduce computational costs. For example, Informer (Zhou et al., 2021) and Autoformer (Wu et al., 2021) reduce complexity to $O(L\log(L))$, where L is the input length. Several improved Transformers (Wang et al., 2020; Ma et al., 2021; Xiong et al., 2021; Choromanski et al., 2021; Liu et al., 2022; Zhou et al., 2022; Chen et al., 2024) even achieve linear complexity. PatchTST Nie et al. (2023) applies the patching technique (Dosovitskiy et al., 2021; Bao et al., 2022; He et al., 2021) to the context of time series, dividing time series into overlapping or non-overlapping continuous patches and embedding each patch instead of individual time points. Although patching does not reduce the theoretical quadratic complexity, it substantially lowers the actual computational costs.

The success of foundation models in NLP, which utilize large-scale pre-training to tackle diverse tasks with minimal labeled data, has inspired similar strategies in time series forecasting. Although pretraining requires extensive computational resources, these models can be fine-tuned and deployed for specific prediction tasks with moderate training budgets.

To address the scarcity of large datasets for training time series foundation models, recent efforts have adapted pre-trained large language models (LLMs) to create a unified framework for various time series tasks, effectively leveraging the ability of transformer-based models to generalize across different domains. Among these efforts, Zhou et al. (2023) demonstrated that the self-attention mechanism functions similarly to PCA, offering a deeper understanding of the universality of transformer-based models. They leveraged a primarily frozen GPT-2 backbone Radford et al. (2019) to achieve competitive performance across a range of time series tasks. TIME-LLM (Jin et al., 2024) converts input time series data into text prototype representations and enhances input context by incorporating declarative prompts to effectively guide the LLM's reasoning process. Additionally, Chang et al. (2023) developed a two-stage fine-tuning approach to adapt the GPT-2 backbone model for time series forecasting tasks.

On the other hand, some works Garza & Mergenthaler-Canseco (2023); Woo et al. (2024); Liu et al. (2024c); Goswami et al. (2024) focus on collecting large datasets and training models directly on these datasets, achieving prominent zero-shot or few-shot performance across a variety of tasks, even closely matching the accuracy of supervised forecasting models tailored to each dataset.

Almost all these time series foundation models are based on the Transformer architecture, which means they share the drawbacks of quadratic complexity and lack of inductive bias.

The emergence of state space models (SSMs, Gu et al. 2021; 2022; Wang et al. 2022; Smith et al. 2023), offering linear or near-linear scaling and improved long-range dependency capture, has spurred significant advances in sequence modeling. By combining principles from RNNs and CNNs, SSMs enable efficient computation and excel in continuous signal domains like audio and vision (Goel et al., 2022; Saon et al., 2023). However, a key weakness of these models is their inability to perform content-based reasoning. To address this issue, Mamba (Gu & Dao, 2023) introduces a selective mechanism that efficiently filters and retains relevant information. It also presents a hardware-aware parallel algorithm, ensuring both theoretical linear complexity and improved practical computational efficiency. Dao & Gu (2024) further elucidate the theoretical connections between SSMs and variants of attention. The promising Mamba model has been applied in various scenarios (e.g., Zhu et al. 2024; Waleffe et al. 2024; Wang et al. 2024) and offers an alternative architecture for time series foundation models.

3 METHOD

This section applies the advanced state space model, Mamba, to construct a foundational model for time series forecasting, called TSMamba. We begin with an introduction to the time series forecasting problem, followed by a description of the Mamba model. Next, we outline the structure of the foundational model and propose a two-stage transfer learning approach to adapt the model to time series data across different domains and frequencies. Finally, we present the fine-tuning process designed to extract relationships within specific datasets, incorporating a compressed channel-wise attention module to leverage cross-channel dependencies.

3.1 THE TIME SERIES FORECASTING PROBLEM

We consider the multivariate time series forecasting problem, which involves predicting the future values of a time series based on historical data. The input series is denoted by $\mathbf{x}_{1:L} = \{\mathbf{x}_1, \cdots, \mathbf{x}_L\}$, where L represents the look-back window (input length). The value at the ith time step is $\mathbf{x}_i \in \mathcal{R}^D$, where D is the number of channels. The model maps the input to the prediction $\mathbf{Y} \in \mathcal{R}^{D\times T}$, where T is the prediction length, also known as the target window. The goal of the model is to understand the input series and minimize the prediction error relative to the actual future values $\mathbf{x}_{L+1:L+T}$.

3.2 PRELIMINARIES OF MAMBA

Originating from the classic Kalman filter model Kalman (1960), the state space model (SSM) has recently garnered significant interest. Among these methods, structured state space sequence models (S4) and Mamba represent a recent class of sequence models inspired by the following continuous system

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

This system maps a one-dimensional function or sequence $\mathbf{x}(t) \in \mathcal{R}$ to $\mathbf{y}(t) \in \mathcal{R}$ through an implicit latent state $\mathbf{h}(t) \in \mathcal{R}^{N_{\text{st}}}$.

The system (1) can be discretized through a zero-order hold (ZOH) rule into

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

where

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

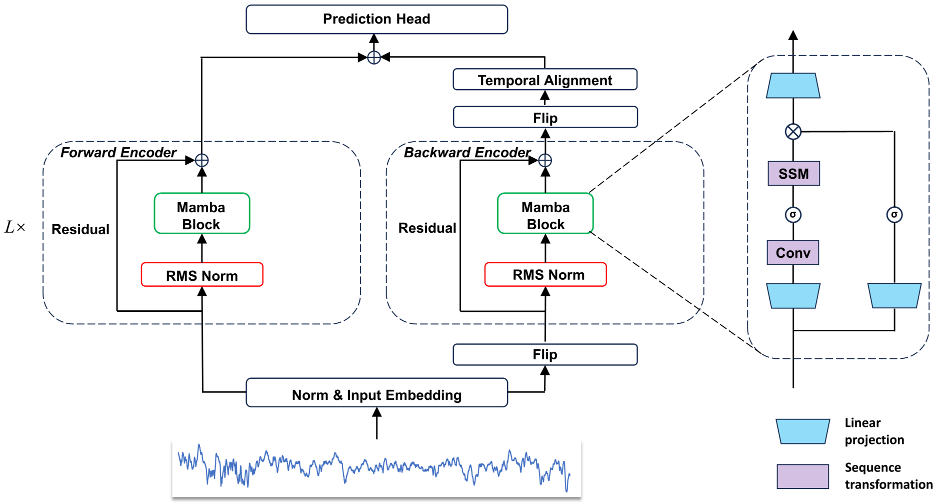

Figure 2: TSMamba Architecture: The input time series are preprocessed and then fed into the forward and backward encoder to extract internal dependencies. The representations are combined and subsequently mapped to forecasts by the prediction head.

and Δ is the time step.

Mamba improves upon S4 by making the parameters that affect interactions along the sequence input-dependent, enabling the model to selectively propagate or forget information based on the current inputs, thereby enhancing its capability to perform content-based reasoning.

Specifically, it sets

$$\begin{aligned} \mathbf{B} &= \text{Linear}_{N_{\text{st}}}(x), \\ \mathbf{C} &= \text{Linear}_{N_{\text{st}}}(x), \\ \Delta &= \text{softplus}(\text{Parameter} + \text{Broadcast}_{D_{\text{mb}}}(\text{Linear}_1(x))), \end{aligned} \tag{4}$$

where Linear_d is a parameterized projection to dimension d, and D_{mb} is the model dimension.

Gu & Dao (2023) also designed the Mamba block, which integrates an SSM into the main branch of the Gated MLP block and serves as a foundational module for building LLMs of different scales.

The modifications in Mamba introduce some drawbacks in parallel computation. S4 can be computed in two ways: either by using the linear recurrence form (2), which is efficient for autoregressive inference, or by employing a global convolution mode, which can be parallelized effectively during training Gu et al. (2022). However, when Mamba introduces input-dependent parameters, it becomes incompatible with the convolution mode, making it difficult to fully utilize the strong parallel processing power of hardware accelerators such as GPUs.

To address this, Gu & Dao (2023) designed a hardware-aware parallel algorithm in recurrent mode. Dao & Gu (2024) further improves the Mamba model, enabling the implementation of tensor parallelism. Consequently, Mamba not only achieves theoretically linear scaling with sequence length but also benefits from fast training and inference in practice.

3.3 Architecture of the Foundation Model

The architecture of TSMamba is illustrated in Figure 2. The model encodes preprocessed data using a backbone comprised of forward and backward Mamba encoders. These encoders consist of homogeneously stacked Mamba blocks, interspersed with standard normalization and residual connections. While the forward encoder extracts sequential causal dependencies, the backward encoder enriches the representation by capturing inverse time relations from the flipped embedding. The backward representation is then flipped and processed through a temporal convolution module to align with the forward representation in the time dimension. Finally, the combined representations are mapped to the forecasting output by a prediction head.

The preprocessing module consists of normalization and input embedding. Given that the number of channels varies across datasets, the multivariate time series are processed in a channel-independent (CI) setting, where each variate is treated as a univariate series. Since normalization is crucial for effective knowledge extraction, we incorporate reverse instance normalization Kim et al. (2021), which normalizes the input time series using mean and variance.

The input embedding is implemented using a 1D convolution layer, which functions similarly to patching Nie et al. (2023) but offers greater convenience. In the patching approach, each univariate series is divided into either overlapping or non-overlapping continuous segments, each of which is then embedded into a vector via linear mapping. This technique preserves local semantic information within the embedding, enhancing the model's ability to capture comprehensive semantic details that might be overlooked at the individual point level. Additionally, it reduces the input length for encoders. We apply a 1D convolution to provide a simplified implementation of patching. In this implementation, the input channel is set to 1, the output channel is set to the model dimension D_m, and the stride is set to the patch length p_l. This module effectively maps non-overlapping segments of p_l time points to a vector embedding of dimension D_m.

TSMamba employs both a forward and a backward encoder (referred to as the backbone hereinafter) to extract temporal dependencies. The forward encoder captures sequential causal relations, while the backward encoder provides additional information by leveraging inverse time relations. The output of the backward encoder is flipped and aligned with the forward representation using a convolution along the time dimension.

Both the forward and backward encoders leverage the backbone of Mamba language models, where each layer consists of a Mamba block equipped with RMSNorm and a residual connection. The Mamba block integrates a state space model (SSM) into the main branch of the Gated MLP block. The block expands the input model dimension D_m to the inner dimension D_{mb} by a factor of 2, then contracts it back to D_m, allowing for homogeneous stacking of blocks. The activation function used is SiLU Elfwing et al. (2018). Unlike the commonly used Transformer encoder or decoder blocks, which require a KV cache proportional to the number of historical tokens, the Mamba block propagates only a fixed-size internal state along the time dimension. This design enables linear complexity in both time and space.

The prediction head generates forecasts based on the historical data representations extracted by the two encoders. In a canonical Transformer decoder, predictions are made autoregressively, where a linear head maps the representation of the last token to outputs, which are then added to the inputs to predict subsequent tokens. However, Nie et al. (2023) show that using a larger linear head to map the representations of all historical patches to the entire target window at once can improve predictions by reducing error accumulation compared to the autoregressive approach.

In a foundation model, where the model dimension is significantly larger than in specialized models, using a linear head that considers all historical representations would result in an excessive number of parameters, making the model prone to overfitting. To address this, the prediction head first compresses the model dimension using a linear projection with GELU Hendrycks & Gimpel (2016) activation. It then maps the compressed representations to the target window with a much smaller linear head. Finally, the outputs are denormalized to restore the original mean and variance.

The model is trained with the Huber loss function, which offers enhanced robustness to outliers in the data compared to the Mean Squared Error (MSE) loss.

3.4 Two-stage transfer learning approach

To harness the knowledge of existing language models, enabling TSMamba to adapt to time series data across different domains and frequencies with low training costs, we designed a two-stage transfer learning approach for training the model.

The first stage involves refining the backbone and training the input embedding through autoregressive forecasting or backcasting tasks.

As shown in Figure 3, the model architecture undergoes slight modifications during this stage. The prediction head, which originally handled compressed historical patches, is replaced with a smaller head that uses the representation of the current patch to predict the next one with the forward rep-

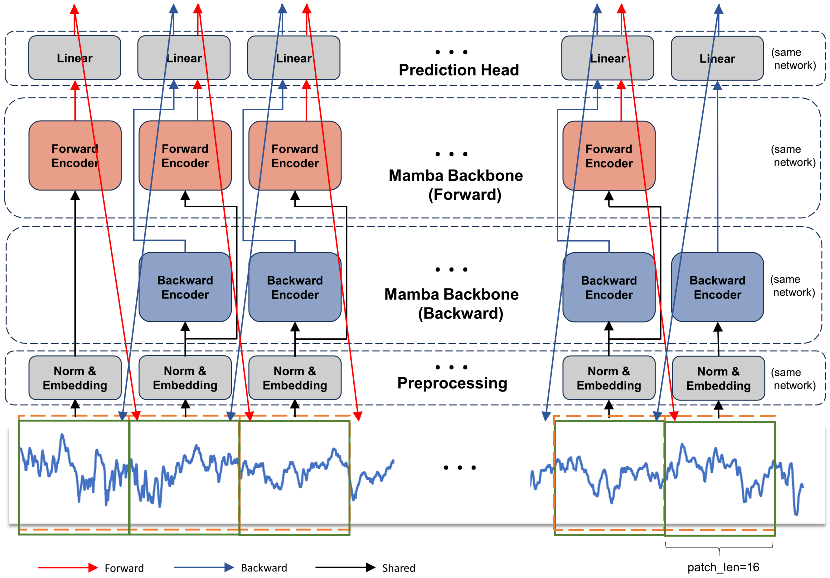

Figure 3: The first stage of transfer learning involves refining the backbone and training the input embedding through autoregressive forecasting or backcasting tasks. A small linear head is temporarily added to predict the next patch.

resentation and the last one with the backward representation. Although a merely autoregressive forecasting is not ideal for long-term forecasting due to error accumulation, this stage is crucial for refining the representation of each patch. This is because the supervision signal for predicting the next patch or the last patch is focused on the current patch's representation, rather than being dispersed across all historical patches as in the original large head configuration.

The backbone is initialized with the Mamba language model, specifically Mamba-130M, while the input embedding and prediction head are trained from scratch.

The second stage focuses on training the prediction head and further refining the other structures. The original TSMamba architecture is restored, with the backbone and input embedding loaded from the results of the first stage, while the prediction head is randomly initialized. This stage produces the TSMamba foundation model for forecasting, which can be used for zero-shot predictions directly or fine-tuned to further enhance performance on specific datasets.

In this stage, the newly initialized prediction head and temporal alignment module are trained with a larger learning rate, while the existing backbone and embedding are updated with a smaller learning rate to fully leverage the pretrained model from the first stage.

The two-stage transfer learning approach also extends to downstream tasks beyond forecasting, such as imputation, classification, and anomaly detection. In these cases, the first stage provides an input embedding and a refined backbone that produce a robust representation of the time series data. The second stage can then be adapted to accommodate specific tasks, achieving zero-shot performance using the information from the training sets.

3.5 Fine-tuning with cross-channel relation extraction

Fine-tuning focuses on learning the unique dependencies of a specified dataset, typically with limited training data. During fine-tuning, we freeze the Mamba blocks in the backbone, which contain most of the model's parameters, to preserve the relationships extracted during pretraining and the two-stage transfer learning. The input embedding, RMSNorm, and prediction head are adjusted to learn from the new dataset. Additionally, we introduce a compressed channel-wise Transformer encoder to extract cross-channel relations.

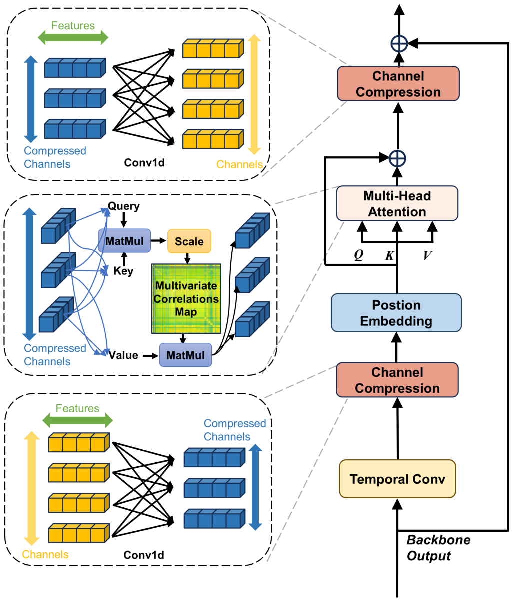

Figure 4: Compressed Channel-Wise Attention Module for Cross-Channel Dependency. The process starts with a per-channel temporal convolution to align the backbone outputs along the time dimension, followed by linear compression of the channel count. The attention module then extracts relationships between these compressed channels, and the result is linearly mapped to restore the original number of channels. Finally, the output is added back to the backbone as a correction.

The compressed channel-wise attention module is added before the prediction head to extract cross-channel dependencies.

Although leveraging cross-channel information is intuitive, previous works Nie et al. (2023); Chen et al. (2024) have shown that it is challenging to utilize these relationships effectively to improve predictions, given the strong baseline established by channel-independent (CI) models. The difficulty arises because typical time series datasets are not large enough to disentangle time-wise and channel-wise modeling effectively. CI models, which treat each variate as an independent sequence, provide more samples of univariate series, thereby reducing the risk of overfitting.

As experiments have shown that linear mixing of channels or adding a channel-wise Transformer encoder often degrades performance, we chose to apply an attention module on a compressed channel dimension, as illustrated in Figure 4. The module takes the representation extracted by the backbone as input. First, we apply a temporal convolution for each channel, acting as a time shift to align channels, since the most relevant data across different channels may exhibit time lags. Next, the number of channels is reduced from D to $\lceil \log_2(D) \rceil$ via linear projection, where $\lceil \cdot \rceil$ denotes the ceiling operation. The attention module is then applied to extract dependencies along the compressed channel dimension. The output is subsequently linearly mapped back to restore the D channels and is added as a correction to the backbone representation. This channel compression acts as a regularization mechanism, filtering out noise in cross-channel relations while reducing computational costs, particularly when D is large. In practice, the linear compression and expansion of channels are performed using 1D convolutions with a kernel size of 1, preventing any permutations.

Since the process of extracting cross-channel dependencies is prone to overfitting, the module is activated only when sufficient training data are available.

4 Experiments

We compare TSMamba with 16 different baselines, which represents state-of-the-art models in long-term forecasting. Our baselines could be divided into two parts: zero-shot forecasting and full-shot forecasting. The zero-shot forecasting includes many pretrained foundation time-series models such as Time-MoE (Shi et al., 2024), Moirai (Woo et al., 2024), TimeFM (Das et al., 2023b), Timer (Liu et al., 2024c), Moment (Goswami et al., 2024), and Chronos (Ansari et al., 2024). As for the full-shot forecasting evaluation includes a collection of Transformer-based models such as PatchTST (Nie et al., 2023), Autoformer (Wu et al., 2021), FEDformer (Zhou et al., 2022), and Crossformer (Zhang & Yan, 2023). Additionally, we incorporate large language models (LLMs) like GPT4TS (Zhou et al., 2023) and CALF (Liu et al., 2024a), as well as other recent competitive approaches, including LightTS (Zhang et al., 2022), DLinear (Zeng et al., 2022), RLinear (Li et al., 2023a), and MambaTS (Cai et al., 2024).

The configuration of TSMamba includes 3 encoder layers, an embedding size of 768, and a fixed input sequence length of 512.

4.1 ZERO-SHOT FORECASTING

The zero-shot forecasting results assess the generalizability of foundation models by examining each model's ability to adapt to previously unseen data.

The models are evaluated on widely recognized long-term forecasting benchmarks that differ from those used during pre-training. As this work is in progress, TSMamba is evaluated on ETTm2 and Weather, two commonly used medium-sized datasets. For each dataset, we considered four different forecasting horizons: {96, 192, 336, 720}. Model performance is measured using two standard evaluation metrics: mean squared error (MSE) and mean absolute error (MAE). The results of other methods for comparison are obtained from (Shi et al., 2024).

Results. Our brief zero-shot results are shown in table 1, comparing TSMamba with state-of-the-art models. While TSMamba does not consistently outperform all baselines, it excels at longer prediction lengths (336 and 720) and achieves competitive average performance. Notably, despite being pre-trained on significantly less data, TSMamba performs comparably to models with larger pre-training datasets, demonstrating its data efficiency and robustness in zero-shot forecasting.

Table 1: Full results of zero-shot forecasting experiments. TimesFM, pre-trained on Weather datasets, is excluded from evaluation on these datasets, and its results are represented by a dash (−). **Red**: the best, Blue: the 2nd best.

Models		**Zero-shot Time Series Models**																					
		TSMamba		**Time-MoE**$_{base}$		**Time-MoE**$_{large}$		**Moirai**$_{small}$		**Moirai**$_{base}$		**Moirai**$_{large}$		**TimesFM**		**Moment**		**Chronos**$_{small}$		**Chronos**$_{base}$		**Chronos**$_{large}$	
Metrics		**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**
ETTm2	96	0.201	0.287	0.201	0.291	**0.197**	0.286	0.214	0.288	0.205	0.273	0.211	0.274	0.202	**0.270**	0.260	0.335	0.209	0.291	0.199	0.274	**0.197**	0.271
	192	0.259	0.325	0.258	0.334	**0.250**	0.322	0.284	0.332	0.275	0.316	0.281	0.318	0.289	0.321	0.289	0.350	0.280	0.341	0.261	0.322	0.254	**0.314**
	336	0.315	0.361	0.324	0.373	0.337	0.375	0.331	0.362	0.329	**0.350**	0.341	0.355	0.360	0.366	0.324	0.369	0.354	0.390	0.326	0.366	**0.313**	0.353
	720	0.406	0.417	0.488	0.464	0.480	0.461	0.402	**0.408**	0.437	0.411	0.485	0.428	0.462	0.430	**0.394**	0.409	0.553	0.499	0.455	0.439	0.416	0.415
	AVG	**0.295**	0.347	0.317	0.365	0.316	0.361	0.307	0.347	0.311	**0.337**	0.329	0.343	0.328	0.346	0.316	0.365	0.349	0.380	0.310	0.350	**0.295**	0.338
Weather	96	0.179	0.235	0.160	0.214	**0.159**	0.213	0.198	0.222	0.220	0.217	0.199	**0.211**	-	-	0.243	0.255	0.211	0.243	0.203	0.238	0.194	0.235
	192	0.227	0.278	**0.210**	0.260	0.215	0.266	0.247	0.265	0.271	0.259	0.246	**0.251**	-	-	0.278	0.329	0.263	0.294	0.256	0.290	0.249	0.285
	336	0.278	0.315	**0.274**	0.309	0.291	0.322	0.283	0.303	0.286	0.297	**0.274**	0.291	-	-	0.306	0.346	0.321	0.339	0.314	0.336	0.302	0.327
	720	0.342	0.358	0.418	0.405	0.415	0.400	0.373	0.354	0.373	0.354	**0.337**	**0.340**	-	-	0.350	0.374	0.404	0.397	0.397	0.396	0.372	0.378
	AVG	**0.256**	0.296	0.265	0.297	0.270	0.300	0.275	0.286	0.287	0.281	0.264	**0.273**	-	-	0.294	0.326	0.300	0.318	0.292	0.315	0.279	0.306
Average		**0.275**	0.321	0.291	0.331	0.293	0.331	0.291	0.317	0.299	0.309	0.297	**0.308**	0.328	0.346	0.305	0.346	0.325	0.349	0.301	0.333	0.287	0.322

4.2 FULL-SHOT FORECASTING

To assess TSMamba's adaptability to specific datasets through fine-tuning, we compare its forecasting results on three widely used datasets: ILI, ETTm2, and Weather. TSMamba is tested with four different prediction horizons $\{96, 192, 336, 720\}$. The results of other methods for comparison are obtained from (Nie et al., 2023; Zhou et al., 2023; Cai et al., 2024).

Results. Our brief full-shot results are shown in Table 2, where TSMamba demonstrates superior performance across multiple datasets and prediction horizons. On average, our model achieves a 15% gain in performance compared to GPT4TS (Zhou et al., 2023), which is a recent LLM based on GPT2. Additionally, TSMamba outperforms the state-of-the-art task-specific time-series model PatchTST (Nie et al., 2023).

Table 2: Full results of in-domain forecasting experiments. A lower MSE or MAE indicates a better prediction. **Red**: the best, Blue: the 2nd best.

Models		Full-shot Time Series Models																									
		TSMamba		CALF		PatchTST		MambaTS		GPT4TS		Autoformer		FEDformer		LightTS		Crossformer		DLinear		TimesNet		RLinear	
Metrics		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm2	96	0.168	0.256	0.178	0.256	0.166	0.256	0.174	0.269	0.173	0.262	0.255	0.339	0.180	0.271	0.209	0.308	0.421	0.461	0.167	0.260	0.187	0.267	**0.164**	**0.253**
	192	0.222	0.292	0.242	0.297	0.223	0.296	0.235	0.309	0.229	0.301	0.281	0.340	0.252	0.318	0.311	0.382	0.503	0.519	0.224	0.303	0.249	0.309	**0.219**	**0.290**
	336	0.279	0.335	0.307	0.339	0.274	0.329	0.288	0.346	0.286	0.341	0.339	0.372	0.324	0.364	0.442	0.466	0.611	0.580	0.281	0.342	0.321	0.351	**0.273**	**0.326**
	720	**0.357**	**0.383**	0.397	0.393	0.362	0.385	0.360	0.393	0.378	0.401	0.433	0.432	0.410	0.420	0.675	0.587	0.996	0.750	0.397	0.421	0.497	0.403	0.366	0.385
	AVG	0.257	0.317	0.281	0.321	**0.256**	0.317	0.264	0.329	0.266	0.326	0.327	0.371	0.292	0.343	0.409	0.436	0.632	0.578	0.267	0.332	0.291	0.333	**0.256**	**0.314**
Weather	96	**0.144**	**0.193**	0.164	0.204	0.149	0.198	0.145	0.196	0.162	0.212	0.266	0.336	0.238	0.314	0.182	0.242	0.153	0.217	0.152	0.237	0.172	0.220	0.175	0.225
	192	**0.192**	**0.236**	0.214	0.250	0.194	0.241	0.193	0.241	0.204	0.248	0.307	0.367	0.275	0.329	0.227	0.287	0.197	0.269	0.220	0.282	0.219	0.261	0.218	0.260
	336	**0.242**	**0.276**	0.269	0.291	0.245	0.282	0.246	0.283	0.254	0.286	0.359	0.395	0.339	0.377	0.282	0.334	0.252	0.311	0.265	0.319	0.280	0.306	0.265	0.294
	720	**0.313**	**0.328**	0.355	0.352	0.314	0.334	0.314	0.331	0.326	0.337	0.419	0.428	0.389	0.409	0.352	0.386	0.318	0.363	0.323	0.362	0.365	0.359	0.329	0.339
	AVG	**0.222**	**0.258**	0.250	0.274	0.226	0.264	0.225	0.263	0.237	0.270	0.338	0.382	0.310	0.357	0.261	0.312	0.230	0.290	0.240	0.300	0.259	0.287	0.247	0.279
ILI	96	**1.189**	**0.659**	-	-	1.319	0.754	-	-	2.063	0.881	2.906	1.182	2.624	1.095	8.313	2.144	3.040	1.186	2.215	1.081	2.317	0.934	4.337	1.507
	192	**1.227**	**0.687**	-	-	1.430	0.834	-	-	1.868	0.892	2.585	1.038	2.516	1.021	6.631	1.902	3.356	1.230	1.963	0.963	1.972	0.920	4.205	1.481
	336	**1.246**	**0.700**	-	-	1.553	0.815	-	-	1.790	0.884	3.024	1.145	2.505	1.041	7.299	1.982	3.441	1.223	2.130	1.024	2.238	0.940	4.257	1.484
	720	**1.357**	**0.762**	-	-	1.470	0.788	-	-	1.979	0.957	2.761	1.114	2.742	1.122	7.283	1.985	3.608	1.302	2.368	1.096	2.027	0.928	4.278	1.487
	AVG	**1.255**	**0.702**	-	-	1.443	0.798	-	-	1.925	0.903	2.819	1.120	2.597	1.070	7.382	2.003	3.361	1.235	2.169	1.041	2.139	0.931	4.269	1.490

5 Conclusions

This paper introduces a Mamba-based foundation model for time series forecasting. The model employs both forward and backward Mamba encoders to capture temporal dependencies. To meet the challenges of forecasting heterogeneous time series data across different domains and frequencies with limited training resources, it leverages knowledge from the Mamba language model and adapts to time series datasets through a two-stage transfer learning approach. While the foundation model processes each channel of multivariate time series independently, it can capture cross-channel dependencies during fine-tuning on specific datasets using an additional compressed cross-channel attention module. TSMamba performs comparably to state-of-the-art models in zero-shot and full-shot settings while requiring significantly less training data, underscoring its potential to enhance downstream forecasting tasks.

References

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, et al. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*, 2024.

Stéphane Ascoli, Hugo Touvron, Matthew L Leavitt, Ari S Morcos, Giulio Biroli, and Levent Sagun. Convit: Improving vision transformers with soft convolutional inductive biases. In *International conference on machine learning*, pp. 2286–2296. PMLR, 2021.

Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. BEiT: BERT pre-training of image transformers. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=p-BhZSz59o4`.

Anastasia Borovykh, Sander M. Bohté, and Cornelis W. Oosterlee. Conditional time series forecasting with convolutional neural networks. *arXiv: Machine Learning*, 2017.

George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Xiuding Cai, Yaoyao Zhu, Xueyao Wang, and Yu Yao. Mambats: Improved selective state space models for long-term time series forecasting. *arXiv preprint arXiv:2405.16440*, 2024.

Defu Cao, Yujing Wang, Juanyong Duan, Ce Zhang, Xia Zhu, Congrui Huang, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, and Qi Zhang. Spectral temporal graph neural network for multivariate time-series forecasting. *ArXiv*, abs/2103.07719, 2020.

Cristian Challu, Kin G Olivares, Boris N Oreshkin, Federico Garza Ramirez, Max Mergenthaler Canseco, and Artur Dubrawski. Nhits: Neural hierarchical interpolation for time series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 6989–6997, 2023.

Ching Chang, Wen-Chih Peng, and Tien-Fu Chen. Llm4ts: Two-stage fine-tuning for time-series forecasting with pre-trained llms. *arXiv preprint arXiv:2308.08469*, 2023.

S. Chen, X. X. Wang, and C. J. Harris. Narxbased nonlinear system identification using orthogonal least squares basis hunting. *IEEE Transactions on Control Systems*, pp. 78–84, 2008.

Yushu Chen, Shengzhuo Liu, Jinzhe Yang, Hao Jing, Wenlai Zhao, and Guangwen Yang. A joint time-frequency domain transformer for multivariate time series forecasting. *Neural Networks*, 176:106334, 2024.

Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamás Sarlós, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, David Benjamin Belanger, Lucy J. Colwell, and Adrian Weller. Rethinking attention with performers. In *9th International Conference on Learning Representations (ICLR), Virtual Event, Austria, May 3-7, 2021*, 2021.

Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. *arXiv preprint arXiv:2405.21060*, 2024.

Abhimanyu Das, Weihao Kong, Andrew Leach, Rajat Sen, and Rose Yu. Long-term forecasting with tide: Time-series dense encoder. *arXiv preprint arXiv:2304.08424*, 2023a.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. *arXiv preprint arXiv:2310.10688*, 2023b.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Luo Donghao and Wang Xue. ModernTCN: A modern pure convolution structure for general time series analysis. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=vpJMJerXHU`.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=YicbFdNTTy`.

Vijay Ekambaram, Arindam Jati, Nam Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. Tsmixer: Lightweight mlp-mixer model for multivariate time series forecasting. *arXiv preprint arXiv:2306.09364*, 2023.

Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural networks*, 107:3–11, 2018.

R. Frigola and C. E. Rasmussen. Integrated pre-processing for Bayesian nonlinear system identification with Gaussian processes. *IEEE Conference on Decision and Control*, pp. 552—560, 2014.

Azul Garza and Max Mergenthaler-Canseco. Timegpt-1. *arXiv preprint arXiv:2310.03589*, 2023.

Karan Goel, Albert Gu, Chris Donahue, and Christopher Ré. It's raw! audio generation with state-space models. In *International Conference on Machine Learning*, pp. 7616–7633. PMLR, 2022.

Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. Moment: A family of open time-series foundation models. *arXiv preprint arXiv:2402.03885*, 2024.

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

Albert Gu, Isys Johnson, Karan Goel, Khaled Saab, Tri Dao, Atri Rudra, and Christopher Ré. Combining recurrent, convolutional, and continuous-time models with linear state space layers. *Advances in neural information processing systems*, 34:572–585, 2021.

Albert Gu, Karan Goel, and Christopher Re. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross B. Girshick. Masked autoencoders are scalable vision learners. *CoRR*, abs/2111.06377, 2021. URL `https://arxiv.org/abs/2111.06377`.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9:1735–1780, 1997.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, et al. Time-llm: Time series forecasting by reprogramming large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

RE Kalman. A new approach to linear filtering and prediction problems. *Trans. ASME, D*, 82:35–44, 1960.

Katikapalli Subramanyam Kalyan, Ajit Rajasekharan, and Sivanesan Sangeetha. Ammus: A survey of transformer-based pretrained models in natural language processing. *arXiv preprint arXiv:2108.05542*, 2021.

Shigeki Karita, Nanxin Chen, Tomoki Hayashi, Takaaki Hori, Hirofumi Inaguma, Ziyan Jiang, Masao Someki, Nelson Enrique Yalta Soplin, Ryuichi Yamamoto, Xiaofei Wang, et al. A comparative study on transformer vs rnn in speech applications. In *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 449–456. IEEE, 2019.

Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *ACM Computing Surveys (CSUR)*, 2021.

Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*, 2021.

Zhe Li, Shiyi Qi, Yiduo Li, and Zenglin Xu. Revisiting long-term time series forecasting: An investigation on linear mapping. *arXiv preprint arXiv:2305.10721*, 2023a.

Zhe Li, Zhongwen Rao, Lujia Pan, and Zenglin Xu. Mts-mixers: Multivariate time series forecasting via factorized temporal and channel mixing. *arXiv preprint arXiv:2302.04501*, 2023b.

Peiyuan Liu, Hang Guo, Tao Dai, Naiqi Li, Jigang Bao, Xudong Ren, Yong Jiang, and Shu-Tao Xia. Calf: Aligning llms for time series forecasting via cross-modal fine-tuning, 2024a. URL `https://arxiv.org/abs/2403.07300`.

Shizhan Liu, Hang Yu, Cong Liao, Jianguo Li, Weiyao Lin, Alex X Liu, and Schahram Dustdar. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *International Conference on Learning Representations*, 2022.

Xu Liu, Junfeng Hu, Yuan Li, Shizhe Diao, Yuxuan Liang, Bryan Hooi, and Roger Zimmermann. Unitime: A language-empowered unified model for cross-domain time series forecasting. In *Proceedings of the ACM on Web Conference 2024*, pp. 4095–4106, 2024b.

Yong Liu, Haoran Zhang, Chenyu Li, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer: Generative pre-trained transformers are large time series models. In *Forty-first International Conference on Machine Learning*, 2024c.

Xuezhe Ma, Xiang Kong, Sinong Wang, Chunting Zhou, Jonathan May, Hao Ma, and Luke Zettlemoyer. Luna: Linear unified nested attention. *CoRR*, abs/2106.01540, 2021.

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

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, 2012.

Yao Qin, Dongjin Song, Haifeng Cheng, Wei Cheng, Guofei Jiang, and Garrison W. Cottrell. A dual-stage attention-based recurrent neural network for time series prediction. In *International Joint Conference on Artificial Intelligence*, pp. 2627–2633, 2017.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Syama Sundar Rangapuram, Matthias W. Seeger, Jan Gasthaus, Lorenzo Stella, Bernie Wang, and Tim Januschowski. Deep state space models for time series forecasting. In *Neural Information Processing Systems*, 2018.

David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36: 1181–1191, 2020.

George Saon, Ankit Gupta, and Xiaodong Cui. Diagonal state space augmented transformers for speech recognition. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. IEEE, 2023.

Rajat Sen, Hsiang-Fu Yu, and Inderjit S. Dhillon. Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting. In *Neural Information Processing Systems*, 2019.

Xiaoming Shi, Shiyu Wang, Yuqi Nie, Dianqi Li, Zhou Ye, Qingsong Wen, and Ming Jin. Time-moe: Billion-scale time series foundation models with mixture of experts. *arXiv preprint arXiv:2409.16040*, 2024.

Jimmy T.H. Smith, Andrew Warrington, and Scott Linderman. Simplified state space layers for sequence modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Ai8Hw3AXqks.

Together. Redpajama: an open dataset for training large language models, 2023. URL https://github.com/togethercomputer/RedPajama-Data.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Roger Waleffe, Wonmin Byeon, Duncan Riach, Brandon Norick, Vijay Korthikanti, Tri Dao, Albert Gu, Ali Hatamizadeh, Sudhakar Singh, Deepak Narayanan, et al. An empirical study of mamba-based language models. *arXiv preprint arXiv:2406.07887*, 2024.

Chloe Wang, Oleksii Tsepa, Jun Ma, and Bo Wang. Graph-mamba: Towards long-range graph sequence modeling with selective state spaces. *arXiv preprint arXiv:2402.00789*, 2024.

Huiqiang Wang, Jian Peng, Feihu Huang, Jince Wang, Junhui Chen, and Yifei Xiao. MICN: Multi-scale local and global context modeling for long-term series forecasting. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=zt53IDUR1U.

Junxiong Wang, Jing Nathan Yan, Albert Gu, and Alexander M Rush. Pretraining without attention. *arXiv preprint arXiv:2212.10544*, 2022.

Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *CoRR*, abs/2006.04768, 2020.

Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125*, 2022.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. *arXiv preprint arXiv:2402.02592*, 2024.

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

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *The Eleventh International Conference on Learning Representations*, 2022.

Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020.

Yunyang Xiong, Zhanpeng Zeng, Rudrasis Chakraborty, Mingxing Tan, Glenn Fung, Yin Li, and Vikas Singh. Nyströmformer: A nyström-based algorithm for approximating self-attention. In *Thirty-Fifth AAAI Conference on Artificial Intelligence*, pp. 14138–14148, 2021.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? *arXiv preprint arXiv:2205.13504*, 2022.

Tianping Zhang, Yizhuo Zhang, Wei Cao, Jiang Bian, Xiaohan Yi, Shun Zheng, and Jian Li. Less is more: Fast multivariate time series forecasting with light sampling-oriented mlp structures. *arXiv preprint arXiv:2207.01186*, 2022.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The eleventh international conference on learning representations*, 2023.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 11106–11115, 2021.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *Proc. 39th International Conference on Machine Learning*, 2022.

Tian Zhou, Peisong Niu, Xue Wang, Liang Sun, and Rong Jin. One fits all: Power general time series analysis by pretrained LM. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL `https://openreview.net/forum?id=gMS6FVZvmF`.

Lianghui Zhu, Bencheng Liao, Qian Zhang, Xinlong Wang, Wenyu Liu, and Xinggang Wang. Vision mamba: Efficient visual representation learning with bidirectional state space model. *arXiv preprint arXiv:2401.09417*, 2024.