Affirm: Interactive Mamba with Adaptive Fourier Filters for Long-term Time Series Forecasting

Yuhan Wu*, Xiyu Meng*, Huajin Hu, Junru Zhang, Yabo Dong†, Dongming Lu

College of Computer Science and Technology, Zhejiang University, Hangzhou, 310027, China
{wuyuhan, mengxiyu, huajinhu, junruzhang, dongyb, ldm}@zju.edu.cn

Abstract

In long-term series forecasting (LTSF), it is imperative for models to adeptly discern and distill from historical time series data to forecast future states. Although Transformer-based models excel at capturing long-term dependencies in LTSF, their practical use is limited by issues like computational inefficiency, noise sensitivity, and overfitting on smaller datasets. Therefore, we introduce a novel time series lightweight interactive Mamba with an adaptive Fourier filter model (**Affirm**). Specifically, (i) we propose an adaptive Fourier filter block. This neural operator employs Fourier analysis to refine feature representation, reduces noise with learnable adaptive thresholds, and captures inter-frequency interactions using global and local semantic adaptive Fourier filters via element-wise multiplication. (ii) A dual interactive Mamba block is introduced to facilitate efficient intra-modal interactions at different granularities, capturing more detailed local features and broad global contextual information, providing a more comprehensive representation for LTSF. Extensive experiments on multiple benchmarks demonstrate that Affirm consistently outperforms existing SOTA methods, offering a superior balance of accuracy and efficiency, making it ideal for various challenging scenarios with noise levels and data sizes. The codes and data are publicly available at https://github.com/zjuml/Affirm.

Introduction

Time series data, collected daily and continuously by IoT sensors and wearable devices, is inherently sequential and time-dependent. Time series forecasting (TSF) predicts future values based on historical observations and is widely applied in finance, meteorology, healthcare, useful life, and transportation (Huang, Chen, and Qiao 2024; Wu et al. 2024b; Wang et al. 2024d; Guo et al. 2024; Long et al. 2024).

Nowadays, deep learning models for TSF are primarily in four families: Transformer, MLP, CNN, and RNN. Transformers are the mainstream due to their ability to capture long-term dependencies through self-attention, such as iTransformer (Liu et al. 2024b) and PatchTST (Nie et al. 2023). However, they struggle with small datasets, easily falling into overfitting and computational inefficiencies due to large parameter sizes and quadratic complexity. Besides, some studies also questioned the effectiveness of Transformers, revealing the permutation invariant nature in attention may compromise temporal information (Zeng et al. 2023). Their experiments show that a simple linear layer can surprisingly outperform complex Transformers in TSF. Yet, linear models have their issues: struggling with complex, noisy data, and failing to capture long-term correlations. Similarly, CNNs are limited by small receptive fields, hampering effectiveness for long-term dependency (Zeng et al. 2024). RNNs, like LSTM and GRU, address long-term dependency but suffer from computational inefficiency and lack of parallelization, leading to slower training and inference.

Recently, state-space-based models (SSMs) (Gu et al. 2020) have garnered attention for their potential in sequence modeling. SSMs excel with lengthy sequences by utilizing ordinary differential equations to dynamically evolve states over time. It incorporates hidden-attention mechanisms and context-aware selectivity with linear complexity, making them ideal for time-series analysis (Cai et al. 2024). Mamba (Gu and Dao 2023), building on SSM and S4, has become powerful in formal language learning (Park et al. 2024), visual representation (Zhu et al. 2024), and image haze removal (Zheng and Wu 2024). Mamba advances SSM capabilities with a selective scanning mechanism that tailors parameters to inputs, enhancing feature compression and information extraction. It also employs a unique hardware-aware algorithm for better parallel processing, offering faster inference and scalability than Transformers (Wang et al. 2024b; Zhu et al. 2024). We replaced the Transformer block with Mamba in PatchTST, and compared it to SOTA iTransformer and PatchTST on various datasets, as shown in Figure 1 and Table 1, revealing that Mamba's performance varies with data frequency. It performs comparably on short-frequency datasets like weather (10 minutes) and ETTm1 (15 minutes) but underperforms on longer-frequency datasets like ETTh2 (hourly) and Exchange (daily). This suggests Mamba struggles with time variations at lower frequencies, prompting us to consider how to enhance Mamba's performance in TSF. Clearly, improvements can be made by better learning short-term and long-term dependencies in time series data.

To this end, we introduce **Affirm** (**A**daptive **f**ourier **f**ilter **i**nte**r**active **m**amba), a lightweight time series model

*These authors contributed equally.

†Corresponding author is Yabo Dong.

Models	Mamba		PatchTST		iTransformer	
	MSE	MAE	MSE	MAE	MSE	MAE
Weather	0.346	0.342	0.329	0.338	0.358	0.349
ETTm1	0.462	0.446	0.422	0.423	0.491	0.459
ETTh2	0.432	0.447	0.395	0.427	0.427	0.445
Exchange	0.913	0.722	0.901	0.714	0.847	0.691

Table 1: Comparison between vanilla Mamba and Transformer-based architectures for forecasting task.

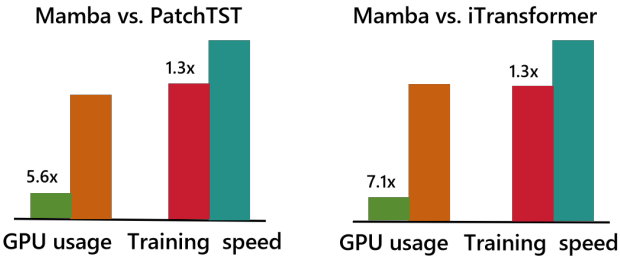

Figure 1: GPU usage and Training speed comparisons.

that scales like Transformers but replaces the self-attention mechanism with the proposed Adaptive Fourier Filter Block (AFFB). Inspired by the convolution theorem (Rabiner and Gold 1975; Huang et al. 2023), AFFB leverages the mathematical equivalence between time domain convolution and the element-wise product in the frequency domain. In this neural operator, Affirm replaces the self-attention layer with four key steps: (i) **Fourier transform**: convert signals from time domain to frequency domain. (ii) **adaptive thresholding**: attenuate high or low frequencies using learnable threshold, a strategy to reduce noise and improve signal clarity. (iii) **adaptive filtering**: apply learnable global and local filters for adaptive filtering across all frequencies via the element-wise product, capturing both long-term and short-term interactions similar to circular convolution. (iv) **inverse Fourier transform**: inverse frequency back to the time domain. We replace the traditional feedforward network with the proposed Interactive Dual Mamba Block (IDMB), where Mamba with dual causal convolutional kernel sizes promotes interactive learning to extract temporal patterns. We also incorporate self-supervised pre-training to boost performance, especially on large datasets.

Affirm is lightweight, reducing complexity from $O(N^2)$ to $O(NlogN)$ using the computationally efficient Fast Fourier Transform (FFT), making it more efficient than self-attention. In summary, our contributions are as follows:

- We introduce Affirm, a generalized lightweight time series forecasting model that uses an interactive Mamba mechanism and three adaptive Fourier filters to capture both long-term and short-term relationships in data.
- We develop an Adaptive Frequency Filtering Block that utilizes Fourier transforms, local and global filters to effectively encompass all frequencies. The block adaptively reduces noise with adaptive appropriate thresholds. Additionally, we introduce the Interactive Dual Mamba Block, which excels in learning complex temporal and spatial features, enhancing adaptability and generalization across various domains.
- Extensive experiments demonstrate that Affirm surpasses SOTA methods, proving its effectiveness and superiority in time series forecasting.

Related Work

State Space Models

State-space models (SSMs) effectively tackle long-range dependencies but are constrained by high computational and memory demands of state representations (Gu et al. 2021). To mitigate this, researchers have developed several variants, such as S4 (Gu, Goel, and Ré 2021), S5 (Smith, Warrington, and Linderman 2022), and SSDNet (Lin, Koprinska, and Rana 2021). Mamba (Gu and Dao 2023) stands out by implementing an S4-based data-dependent selection mechanism to filter inputs and using hardware-aware algorithms for parallel processing. Its successful application in computer vision (Tang et al. 2024), recommendations (Liu et al. 2024a), and graphs (Wang et al. 2024a) demonstrates its effectiveness and adaptability. Moreover, it achieves linear-time efficiency with long sequences and outperforms Transformers in benchmark evaluations, positioning it as a potent alternative for time series (Cai et al. 2024). For example, MambaTS (Cai et al. 2024) uses time-varying scanning to arrange historical information and solve the scan order sensitivity issue. TimeMachine (Ahamed and Cheng 2024) proposes a multiscale Mamba to address channel mixing and independence. MambaMixer (Behrouz, Santacatterina, and Zabih 2024) uses bi-directional blocks for inter- and intra-sequence analysis. To further enhance its ability in TSF, we introduce IDMB to learn both local and global features.

Frequency-Aware Time Series

Frequency analysis is a staple in traditional signal processing, offering valuable insights into frequency representation learning (Wu et al. 2024a; Liu et al. 2023). For instance, Autoformer (Wu et al. 2021) replaces self-attention with an FFT-based auto-correlation mechanism, while FEDformer (Zhou et al. 2022b) enhances long-term periodic patterns using a DFT-based attention mechanism. Similarly, FiLM (Zhou et al. 2022a) leverages Fourier analysis to filter noise while preserving historical data. FreTS (Yi et al. 2024) captures channel and time dependencies by analyzing both real and imaginary frequency components, and FITS (Xu, Zeng, and Xu 2023) uses rFFT and low-pass filters for compact representation. TSLANet (Eldele et al. 2024) employs adaptive spectral blocks to effectively capture time series features.

However, many of these methods rely on feature engineering for cycle selection, focusing on dominant cycles and harmonics, which may hinder representation learning and lead to inefficiency or overfitting (Xu, Zeng, and Xu 2023). This paper handles this inspired by frequency filters in digital image processing and computer vision (Pitas 2000; Huang et al. 2023), where learnable Fourier filters facilitate mutual information learning, improve semantic adaptability, and reduce computational costs and parameters.

Method

Preliminaries

Discrete Fourier Transform. The discrete nature of DFT aligns seamlessly with digital processing, enabling efficient numerical calculations with $O(N \log N)$ complexity. It converts a sequence of N complex numbers $x[n]$, where $0 \leq$ n $\leq N-1$, from the time domain to the frequency domain through a 1D transformation.

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j(2\pi/N)kn} := \sum_{n=0}^{N-1} x[n] W_N^{kn} \quad (1)$$

where j is the imaginary unit with $W_N = e^{-j(2\pi/N)}$. It transforms a sequence $x[n]$ into its frequency spectrum $X[k]$ with frequencies $\omega_k = 2\pi k/N$, where are periodic for length N. Thus, only the first N points are considered.

DFT is a bijective transformation, enabling exact recovery of the original sequence $x[n]$ via the Inverse DFT (IDFT):

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j(2\pi/N)kn} \quad (2)$$

State Space Models. SSMs function as linear time-invariant (LTI) systems that map continuous input signals $x(t) \mapsto y(t)$ via a hidden state $h(t)$. This state space captures the evolution of the state over time and is typically described using ordinary differential equations as follows:

$$h'(t) = \mathbf{A}h(t) + \mathbf{B}x(t), y(t) = \mathbf{C}h(t). \quad (3)$$

where $h'(t) = \frac{dh(t)}{dt}$, A denotes state evolution, B and C are projection parameters.

S4 and Mamba are discrete SSMs, employing a timescale parameter Δ and discretization methods like Euler's or Zero-Order Hold (ZOH) to derive their discrete-time matrices $\overline{\mathbf{A}}$ and $\overline{\mathbf{B}}$ from the continuous-time matrices A and B:

$$\overline{\mathbf{A}} = \exp(\Delta\mathbf{A}), \overline{\mathbf{B}} = (\Delta\mathbf{A})^{-1}(\exp(\Delta\mathbf{A}) - \mathbf{I}) \cdot \Delta\mathbf{B} \quad (4)$$

Then, SSM can discretize continuous signals into sequences with a time interval of Δ:

$$h_t = \overline{\mathbf{A}}h_{t-1} + \overline{\mathbf{B}}x_t, y_t = \mathbf{C}h_t. \quad (5)$$

Here, h_t and x_t denote the state vector and input vector.

Finally, the model computes the output by a global convolution for training as the following:

$$\overline{K} = (C\overline{B}, C\overline{AB}, ..., C\overline{A}^{M-1}\overline{B}), y = x * \overline{K}. \quad (6)$$

Overall Architecture

Our framework integrates two new components: the Adaptive Frequency Filtering Block (AFFB) and the Interactive Dual-Mamba Block (IDMB), illustrated in Figure 2. AFFB employs Fourier analysis to transform the time series data to the frequency domain, where it applies adaptive thresholding to attenuate high-frequency and low-frequency noises through learnable Low Pass Filtering (LPF) and High Pass Filtering (HPF), highlighting relevant spectral features and compacting model size. Then the combined signals are captured through a learnable filter using a 1×1 convolutional (linear) layer, a ReLU function, and another linear layer. After processing, the IFFT reconstructs the temporal representations with reduced noise and enhanced features. IDMB interactively refines local and global features with different kernel sizes and dropout mechanisms, enhancing adaptability to temporal dynamics in TSF. Together, these components effectively balance local and global temporal features for robust time series analysis.

Embedding Layer

Given an input time series $X \in R^{C \times L}$, where C is the channel number and L is the sequence length, X is divided into N non-overlapping patches $\{P_1, P_2, ..., P_N\}$, each of length p, totally patches $P_i \in R^{C*p}$. Each patch is mapped to a new dimension d, resulting in $P_i \rightarrow P_i' \in R^{C \times d}$. Position embeddings E_i are added to each patch to preserve temporal order, giving $X_{PE_i} = P_i' + E_i$. The complete set of latent patches is $X_{PE} = X_{PE_1}, X_{PE_2}, ..., X_{PE_N}$, with the positional embeddings enhancing temporal correlation capture.

Adaptive Fourier Filter Block

High frequencies are often discarded as noise, and the denoised signal is obtained by reconstructing only the low-frequency components. This approach has drawbacks: (i) it ignores some useful information in high frequencies, and (ii) it may reconstruct weak noise in low frequencies as part of the output, leading to reduced signal-to-noise ratios and increased errors (Li, Bu, and Yang 2023). Inspired by (Rao et al. 2021; Huang et al. 2023), we propose the AFF Block, which aims to learn spatial information through global circular convolution operations.

Fast Fourier Transformations. For a given X_{PE}, its representation is calculated as:

$$X_F = \mathcal{F}[X_{PE}] \in \mathcal{C}^{C \times L_F'} \quad (7)$$

Here, $\mathcal{F}[\cdot]$ denotes the 1D FFT operation, and L_F' is the sequence length after Fourier transformation.

Adaptive Low Pass Filter. High-frequency components often indicate rapid fluctuations or deviations from the underlying trend, making the signal more random and harder to interpret (Eldele et al. 2024). Removing high-frequency noise can help to learn the trend and periodic patterns, which is crucial in TSF. Besides, this operation can simplify the model, accelerate training, and lower computational costs. Thus, we propose an adaptive low-pass filter with a learnable threshold that dynamically adjusts the filtering degree to suppress irrelevant noise while preserving vital information, resulting in a more concise frequency representation.

The trainable threshold θ_{high} is adaptively tuned to match the frequency characteristics, optimized through backpropagation to effectively distinguish valuable information from high-frequency noise and eliminate components exceeding the threshold. The formula is as follows:

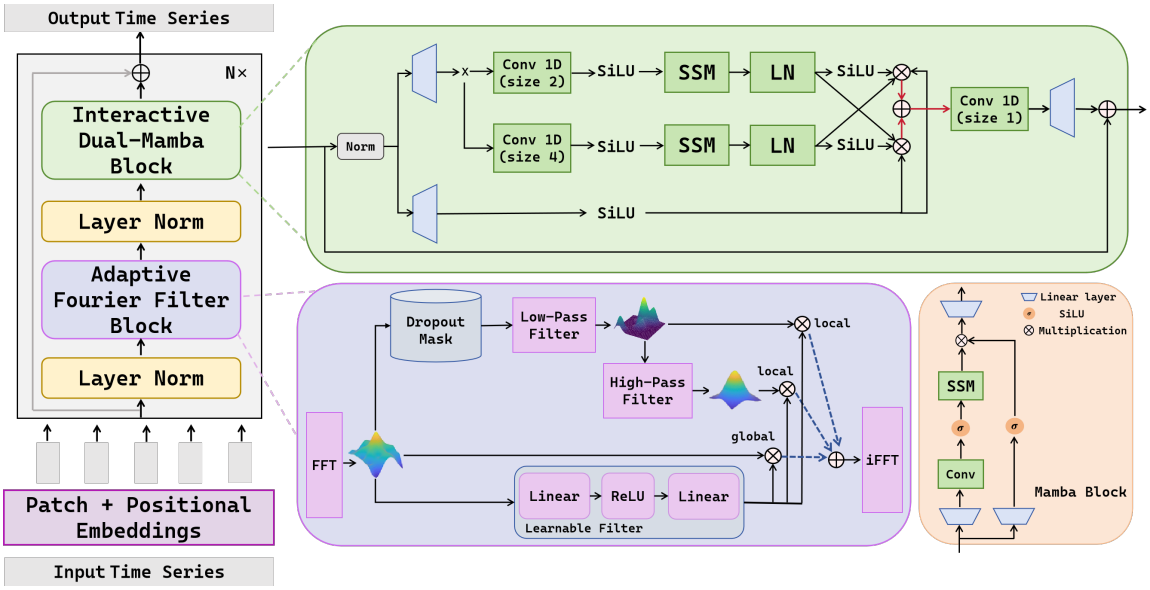

Figure 2: The structure of our proposed Affirm.

$$X_{Filter}^{high} = X_F \odot (|F| \leq \theta_{high}) \tag{8}$$

where $\odot$ denotes element-wise multiplication (the Hadamard product), and $|F| \leq \theta_{high}$ represents a binary mask where frequencies below the threshold θ_{high} are retained, while others are filtered out.

Adaptive High Pass Filter. Low-frequency components generally encapsulate the underlying trends and cycles in time series data. However, they may still contain noise, particularly systematic errors from data collection or processing, such as sensor calibration issues, algorithmic biases, and data entry mistakes (Li, Bu, and Yang 2023). Consequently, we implemented an adaptive low-frequency denoising branch, which is applied after handling high-frequency noise. This method offers several benefits: (1) It removes unnecessary noise, allowing the model to focus on crucial frequency components, and enhancing generalization. (2) It helps capture essential features, reducing overfitting and improving performance. (3) It improves the model's ability to handle non-smooth data, stabilizing the data structure. The process with a learnable threshold θ_{low} is:

$$X_{Filter}^{low} = X_F \odot (|F| > \theta_{low}) \tag{9}$$

Similarly, $|F| > \theta_{low}$ means frequencies above the threshold θ_{low} are retained, while others are discarded.

Learnable Linear Filters. After adaptive filtering in the frequency domain, Affirm employs three learnable linear filters: a global filter $\mathcal{M}(X_F)_G$ from the original frequency X_F; a local filter $\mathcal{M}(X_F^{high})_L$ from high frequency; and a local filter $\mathcal{M}(X_F^{low})_L$ from low frequency. Each filter is tailored to match the corresponding frequency features X_F, X_F^{high}, and X_F^{low}. The process is as follows:

$$X_G = \mathcal{M}(X_F)_G \odot X_F \tag{10}$$

$$X_L^{high} = \mathcal{M}(X_F^{high})_L \odot X_F^{high} \tag{11}$$

$$X_L^{low} = \mathcal{M}(X_F^{low})_L \odot X_F^{low} \tag{12}$$

As shown in Figure 2, to make the network as lightweight as possible, $M(\cdot)$ is implemented by a $1 * 1$ convolutional (linear) layer, followed by ReLU, and another linear layer. The three filtered spectra are then integrated to capture the full spectral feature details, calculated as:

$$X_{mixed} = X_G + X_L^{high} + X_L^{low} \tag{13}$$

The X_{mixed} denotes the global and local adaptive frequency mixing of X_F. The multiplication operation is mathematically equivalent to the dynamic circular convolutional process.

Inverse Fourier Transform. We further use IFFT to transfer the mixed frequency back to the time domain:

$$X_T = \mathcal{F}^{-1}[X_{mixed}] \in \mathcal{R}^{C \times d} \tag{14}$$

where $\mathcal{F}^{-1}(\cdot)$ represents IFFT, ensuring the combined features remain consistent with the original time series data.

Interactive Dual Mamba Block

The original Mamba structure is shown in the bottom right of Figure 2. Apply LayerNorm first, then process the features through two parallel branches: the left SSM branch for sequence modeling and the right for a gated nonlinear layer.

$$h_t = SSM(Conv(Linear(X_T))) + \sigma(Linear(X_T)) \tag{15}$$

We design an IDMB with causal convolutions of different kernel sizes to process X_T after AFFB. X_T is passed

through these convolutions with LayerNorm to fully extract features, capturing both local features and long-term dependencies through parallel interactions. Specifically, the first convolution, using a 2x2 depth-wise filter f_{2*2}: $h_{2*2} = LN(SSM(\sigma(Conv1(Linear(X_T)))))$, captures fine-grained local patterns, while the second, with a 4x4 depth-wise filter f_{4*4}: $h_{4*4} = LN(SSM(\sigma(Conv2(Linear(X_T)))))$, identifies broader dependencies. Conv1($\odot$) and Conv2($\odot$) are 1D-convolution layers. IDMB allows each layer to modulate the other, enhancing comprehensive feature extraction by promoting interaction between different scales, computed as:

$$\mathcal{H}_1 = \sigma(h_{2*2}) \bigodot h_{4*4} \bigodot \sigma(Linear(X_T)) \quad (16)$$

$$\mathcal{H}_2 = \sigma(h_{4*4}) \bigodot h_{2*2} \bigodot \sigma(Linear(X_T)) \quad (17)$$

$\mathcal{H}_1$ and $\mathcal{H}_2$ are then added and passed through a linear convolution layer Conv3($\odot$) tailored to specific task:

$$\mathcal{O}_{IDMB} = Conv3(\mathcal{H}_1 + \mathcal{H}_2) \quad (18)$$

where O_{IDMB} is the output features.

Self-Supervised Pretraining

Self-supervised learning utilizes data's inherent structure to extract valuable information without needing external labels, which is vital in scenarios where data acquisition is costly or annotations are scarce (Nie et al. 2023). Inspired by methods in computer vision and NLP, we introduced a self-supervised pre-training phase for Affirm using a masked autoencoder approach for time series data (He et al. 2022). By partially masking input series, Affirm is trained to detect hidden patterns and reconstruct complete information, enhancing its robustness, generalization, and feature extraction. Affirm focuses on broader data patches rather than single-point masking, encouraging thorough sequence analysis and optimizing patch reconstruction through MSE minimization.

Experiments

Experimental Setup

Datasets. To validate our model, we evaluate Affirm on 8 benchmarks: 4 ETT datasets (ETTh1, ETTh2, ETTm1, ETTm2), Electricity (ECL), Exchange, Traffic, and Weather (Wu et al. 2021).

Baselines. We assessed Affirm against ten baselines, including: 1) Transformers: iTransformer (Liu et al. 2024b), PatchTST (Nie et al. 2023), Crossformer (Zhang and Yan 2023); 2) MLPs: Dlinear (Zeng et al. 2023), Rlinear (Li et al. 2023), TimeMixer (Wang et al. 2024c); 3) CNNs: TSLANet (Eldele et al. 2024); 4) Mambas: DTMamba (Wu, Gong, and Zhang 2024); 5) and LLMs: Time-LLM (Jin et al. 2023), GPT4TS (Zhou et al. 2023).

Implementation Details. We conduct all experiments on an NVIDIA GeForce RTX 4090 Ti GPU with 64-bit Linux 5.15.0-56-generic, with 60/20/20 train/validation/test split for ETTs, and 70/10/20 for other datasets. Similar to (Zhou et al. 2023) settings, we use the look-back window of 336 for ETTs, 96 for exchange, 512 for Traffic and Weather, and 96 for Electricity. All datasets are normalized during training (Kim et al. 2021). For baselines, we report their best results if setups match; otherwise, we rerun their code.

Results

As shown in Table 2, the Time-LLM model excels by leveraging the robust llama-7B model, effectively capturing complex patterns. Affirm outshines in almost all benchmarks except Time-LLM, achieving second-best performances on seven out of eight datasets and improving MSE 4.0% and 6.9% over SOTA PatchTST in ETTh1 and ECL (avg), confirming its robustness in diverse scenarios. Affirm and TSLANet both use the neural operator, with Affirm achieving superior MSE improvements of 1.6% and 3.5% on ETTm2 and Exchange (avg), indicating its effectiveness in learning long-term dependencies. Furthermore, results also show Affirm excels beyond specialized Transformers and MLPs. These models, such as iTransformer and Dlinear, perform competitive performance in certain cases but fall short in others. GPT4TS also showcases the capability of GPT models in forecasting by achieving the second-best in some datasets. Despite Time-LLM's slightly better performance, it requires much higher computational resources. For instance, on the ETTh1 dataset, Affirm presents a nearly equivalent performance to Time-LLM with an MSE of 0.411 against Time-LLM's 0.408 but uses far less computational costs—9.78e+07 FLOPS against Time-LLM's 7.3e+12, highlights Affirm's efficient balance between performance and computational demand.

Ablation Study

Various Variants of Affirm. To assess the impact of Affirm's various components, we conduct ablation studies as detailed in Table 3 . Removing the IDMB (i.e., w/o IDMB) and AFFB (i.e., w/o AFFB) significantly impairs performance, with AFFB's absence resulting in more pronounced declines. Specifically, eliminating IDMB decreased MSE by 2.6% and 5.4% and MAE by 2.5% and 4.4% on ETTh1 and Weather. The removal of AFFB led to larger reductions in MSE by 5.1% and 5.8%, emphasizing its importance in feature extraction and denoising. We also investigate the effects of omitting local adaptive components within AFFB—LPF and HPF—on noise filtering. Removing both adversely impacted performance, especially the LPF, indicating that high-frequency noise more severely affects outcomes. A scenario without both local filters (i.e., w/o LPF+HPF) showed deteriorating performance over time with only global filtering, confirming the essential role of local filters in noise management. Similarly, the importance of pretraining is confirmed, as its absence marginally reduces the model's predictive performance.

Varying Look-back Window. In principle, a longer look-back window should improve prediction accuracy by expanding the receptive field, but this is not typically observed in most Transformer-based models (Zeng et al. 2023). As in Figure 3, these models including TSLANet, show limited benefits from extended look-back periods, indicating a weak grasp of temporal information. In contrast, the MSE of PatchTST and our Affirm decreases as the look-back window lengthens, with Affirm consistently outperforming PatchTST across all settings, demonstrating its superior ability in both short-term and long-term prediction tasks.

Methods		Affirm		DTMamba		TSLANet		Time-LLM		GPT4TS		iTransformer		PatchTST		Crossformer		Rlinear		Dlinear		TimeMixer	
Metrics		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.363	**0.392**	0.386	0.399	0.370	0.394	**0.362**	0.392	0.376	0.397	0.386	0.405	0.382	0.401	0.423	0.448	0.386	0.395	0.375	0.399	0.375	0.400
	192	0.408	0.421	0.426	0.424	0.412	0.417	**0.398**	0.418	0.416	0.418	0.441	0.436	0.428	0.425	0.471	0.474	0.437	0.424	0.405	**0.416**	0.429	0.421
	336	0.424	0.426	0.480	0.450	**0.399**	**0.416**	0.430	0.427	0.442	0.433	0.487	0.458	0.451	0.436	0.570	0.546	0.479	0.446	0.439	0.443	0.484	0.458
	720	0.450	**0.453**	0.484	0.470	0.472	0.475	**0.442**	0.457	0.477	0.456	0.503	0.491	0.452	0.459	0.653	0.621	0.481	0.470	0.472	0.490	0.498	0.482
	Avg.	0.411	**0.423**	0.444	0.435	0.413	0.426	**0.408**	0.423	0.428	0.426	0.454	0.448	0.428	0.430	0.529	0.522	0.446	0.434	0.423	0.437	0.447	0.440
ETTh2	96	0.276	0.343	0.290	0.340	0.280	0.341	**0.268**	**0.328**	0.285	0.342	0.297	0.349	0.285	0.340	0.745	0.584	0.288	0.338	0.289	0.353	0.289	0.341
	192	**0.316**	**0.372**	0.366	0.392	0.330	0.375	0.329	0.375	0.354	0.389	0.380	0.400	0.356	0.386	0.877	0.656	0.374	0.390	0.383	0.418	0.372	0.392
	336	0.341	0.383	0.380	0.409	**0.317**	**0.374**	0.368	0.409	0.373	0.407	0.428	0.432	0.350	0.395	1.043	0.731	0.415	0.426	0.448	0.465	0.386	0.414
	720	0.391	0.429	0.416	0.437	0.404	0.440	**0.372**	**0.420**	0.406	0.441	0.427	0.445	0.395	0.427	1.104	0.763	0.420	0.440	0.605	0.551	0.412	0.434
	Avg.	**0.331**	**0.381**	0.363	0.395	0.333	0.383	0.334	0.383	0.355	0.395	0.383	0.407	0.347	0.387	0.942	0.684	0.374	0.399	0.431	0.447	0.364	0.395
ETTm1	96	0.285	0.344	0.325	0.360	0.289	0.349	**0.272**	**0.334**	0.292	0.346	0.334	0.368	0.291	0.340	0.404	0.426	0.355	0.376	0.299	0.343	0.320	0.357
	192	0.323	0.365	0.375	0.386	0.328	0.370	**0.310**	**0.358**	0.332	0.372	0.377	0.391	0.328	0.365	0.450	0.451	0.391	0.392	0.335	0.365	0.361	0.381
	336	**0.351**	**0.384**	0.396	0.405	0.355	0.389	0.352	0.384	0.366	0.394	0.426	0.420	0.365	0.389	0.532	0.515	0.424	0.415	0.369	0.386	0.390	0.404
	720	0.418	0.414	0.454	0.442	0.421	0.425	**0.383**	**0.411**	0.417	0.421	0.491	0.459	0.422	0.423	0.666	0.589	0.287	0.450	0.425	0.421	0.454	0.441
	Avg.	0.344	0.377	0.388	0.399	0.348	0.383	**0.329**	**0.372**	0.352	0.383	0.407	0.410	0.352	0.379	0.513	0.495	0.414	0.408	0.357	0.379	0.381	0.395
ETTm2	96	0.167	0.260	0.177	0.259	0.169	0.259	**0.161**	**0.253**	0.173	0.262	0.180	0.264	0.169	0.254	0.287	0.366	0.182	0.265	0.167	0.260	0.175	0.258
	192	0.221	0.296	0.240	0.300	0.224	0.297	**0.219**	**0.293**	0.229	0.301	0.250	0.309	0.230	0.294	0.414	0.492	0.246	0.304	0.224	0.303	0.237	0.299
	336	**0.271**	**0.328**	0.310	0.345	0.275	0.329	0.271	0.329	0.286	0.341	0.311	0.348	0.280	0.329	0.597	0.542	0.307	0.342	0.281	0.342	0.298	0.340
	720	**0.351**	**0.377**	0.395	0.394	0.354	0.380	0.352	0.379	0.378	0.401	0.412	0.407	0.378	0.386	1.730	1.042	0.407	0.398	0.397	0.421	0.391	0.396
	Avg.	0.252	0.315	0.281	0.325	0.256	0.316	**0.251**	**0.313**	0.267	0.326	0.288	0.332	0.264	0.316	0.757	0.611	0.286	0.327	0.267	0.332	0.275	0.323
Electricity	96	**0.129**	**0.223**	0.166	0.256	0.136	0.229	0.131	0.224	0.139	0.238	0.148	0.240	0.138	0.230	0.219	0.314	0.201	0.281	0.140	0.237	0.153	0.247
	192	**0.146**	**0.239**	0.178	0.268	0.152	0.244	0.152	0.241	0.153	0.251	0.162	0.253	0.149	0.243	0.231	0.322	0.201	0.283	0.153	0.249	0.166	0.256
	336	0.162	0.252	0.197	0.289	0.168	0.262	**0.160**	**0.248**	0.169	0.266	0.178	0.269	0.169	0.262	0.246	0.337	0.215	0.298	0.169	0.267	0.185	0.277
	720	**0.191**	**0.288**	0.243	0.326	0.205	0.293	0.192	0.298	0.206	0.297	0.225	0.317	0.211	0.299	0.280	0.363	0.257	0.331	0.203	0.301	0.225	0.310
	Avg.	**0.157**	**0.250**	0.196	0.285	0.165	0.257	0.158	0.252	0.167	0.263	0.178	0.270	0.167	0.259	0.244	0.334	0.219	0.298	0.166	0.264	0.182	0.272
Exchange	96	**0.080**	**0.198**	0.083	0.201	0.083	0.201	0.123	0.251	0.082	0.199	0.086	0.206	0.088	0.205	0.256	0.367	0.093	0.217	0.081	0.203	0.091	0.215
	192	0.169	0.296	0.173	0.295	0.177	0.299	0.224	0.344	0.171	**0.293**	0.177	0.299	0.176	0.299	0.470	0.509	0.184	0.307	**0.157**	0.293	0.197	0.318
	336	0.325	0.411	0.346	0.427	0.331	0.417	0.377	0.451	0.354	0.428	0.331	0.417	**0.301**	**0.397**	1.268	0.883	0.351	0.432	0.305	0.414	0.416	0.472
	720	0.852	0.690	0.868	0.698	0.888	0.739	1.018	0.771	0.877	0.704	0.847	0.691	0.901	0.714	1.767	1.068	0.886	0.714	**0.643**	**0.601**	0.968	0.725
	Avg.	0.356	0.399	0.368	0.405	0.370	0.414	0.436	0.454	0.371	0.406	0.360	0.403	0.367	0.404	0.940	0.707	0.379	0.418	**0.297**	**0.378**	0.418	0.433
Traffic	96	0.364	0.255	0.487	0.317	0.372	0.261	**0.362**	**0.248**	0.388	0.282	0.395	0.268	0.401	0.267	0.522	0.290	0.649	0.389	0.410	0.282	0.462	0.285
	192	0.381	0.262	0.498	0.325	0.388	0.266	**0.374**	**0.247**	0.407	0.290	0.417	0.276	0.406	0.268	0.530	0.293	0.601	0.366	0.423	0.287	0.473	0.296
	336	0.392	**0.268**	0.511	0.334	0.394	0.269	**0.385**	0.271	0.412	0.294	0.433	0.283	0.421	0.277	0.558	0.305	0.609	0.369	0.436	0.296	0.498	0.296
	720	0.433	0.290	0.533	0.326	0.430	0.289	**0.430**	**0.288**	0.450	0.312	0.467	0.302	0.452	0.297	0.589	0.328	0.647	0.387	0.466	0.315	0.506	0.313
	Avg.	0.392	0.268	0.507	0.326	0.396	0.271	**0.388**	**0.264**	0.414	0.295	0.428	0.282	0.420	0.277	0.550	0.403	0.627	0.378	0.434	0.295	0.484	0.297
Weather	96	**0.146**	**0.196**	0.171	0.218	0.148	0.197	0.147	0.201	0.162	0.212	0.174	0.214	0.160	0.204	0.158	0.230	0.192	0.232	0.176	0.237	0.163	0.209
	192	0.192	0.239	0.220	0.257	0.193	0.241	**0.189**	**0.234**	0.204	0.248	0.221	0.254	0.204	0.245	0.206	0.277	0.240	0.271	0.220	0.282	0.208	0.250
	336	**0.244**	**0.278**	0.274	0.296	0.245	0.282	0.262	0.279	0.254	0.286	0.278	0.296	0.257	0.285	0.272	0.335	0.292	0.307	0.265	0.319	0.251	0.287
	720	0.321	0.332	0.349	0.346	0.325	0.337	**0.304**	**0.316**	0.326	0.337	0.358	0.349	0.329	0.338	0.298	0.418	0.364	0.353	0.323	0.362	0.339	0.341
	Avg.	0.226	0.261	0.254	0.279	0.228	0.264	**0.225**	**0.257**	0.237	0.271	0.258	0.278	0.238	0.268	0.259	0.315	0.272	0.291	0.246	0.300	0.240	0.271

Table 2: Multivariate long-term series forecasting results on different prediction lengths $\in \{96, 192, 336, 720\}$. A lower value indicates better performance. **Bold**: best, underlined: second best.

Variants	ETTh1 (avg.)		Weather (avg.)	
	MSE	MAE	MSE	MAE
w/o IDMB	0.423	0.434	0.239	0.273
w/o AFFB	0.434	0.435	0.240	0.275
w/o LPF	0.421	0.432	0.236	0.271
w/o HPF	0.417	0.430	0.229	0.265
w/o LPF+HPF	0.422	0.433	0.237	0.272
w/o pretraining	0.415	0.426	0.228	0.263
Affirm	0.411	0.423	0.226	0.261

Table 3: Ablation study of each component in Affirm.

Efficiency of Adaptive Filter in Noise Reduction

We further explore the effectiveness of adaptive filters (LPF, HPF) in reducing noise, as illustrated in Figure 4. Specifically, it depicts how the Transformer, PatchTST, Affirm, and Affirm w/o filters perform under varying degrees of Gaussian noise. While the Transformer's performance declines sharply with increased noise, PatchTST is more sensitive to noise than ETTh1 in large Weather, and both Affirm variants exhibit stable and robust behavior, especially Affirm (with filters). This indicates the significant advantage of adaptive filters in managing noisy environments. Notably, Affirm w/o adaptive filters still outperforms Transformer and PatchTST, emphasizing the benefits of IDMB and global filters.

Scaling Efficiency

To evaluate Affirm's scalability, we compared it against one of the best-performing Transformers in TSF, i.e., PatchTST (Nie et al. 2023) on ETTm1 with varying sizes and layers. Figure 5 shows that Affirm consistently outperforms PatchTST across all conditions. As data size increases, the performance of both models slightly decreases with layers increase, but PatchTST's drop is more pronounced. Notably, on a smaller dataset (1%), Affirm remains stable despite increased layers, while PatchTST's curve suffers a marked upswing, indicating a tendency toward overfitting or inefficiency as model complexity rises with limited data. This

Figure 3: MSE on 3 large datasets.

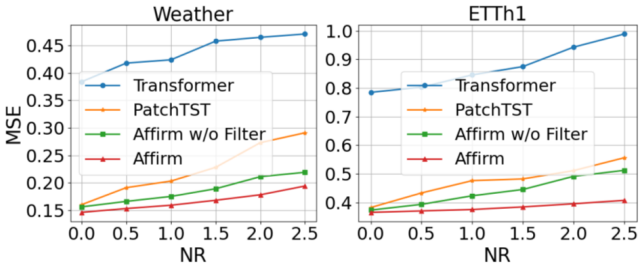

Figure 4: Effectiveness of Adaptive Filter in noise reduction.

may stem from PatchTST's inherent design, which might yield diminishing returns as model depth increases, posing optimization challenges that impede the learning of effective features. In contrast, Affirm's performance remains steady or improves with larger datasets and increased layer counts, showcasing its capacity to effectively utilize extensive data.

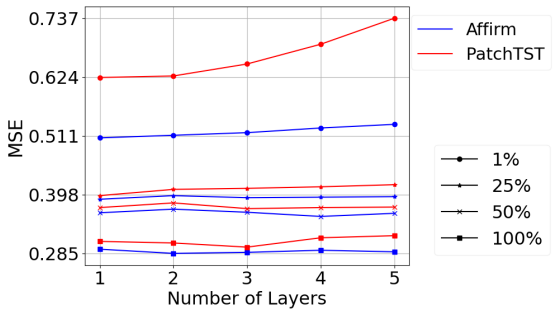

Figure 5: Scaling comparison of Affirm vs. PatchTST across different layer counts and data percentages on ETTm1.

Complexity Analysis

We assessed Affirm's complexity against various TSF models on the ECL dataset, focusing on training parameters, MACs, and MSE over a 96 look-back window and a 720 prediction length. Table 4 highlights Affirm's efficiency and superior performance. Models like Transformer, Informer, Autoformer, FEDformer, and FiLM have parameter counts ranging from 13.61M to 20.68M and MACs between 3.93G to 5.97G, yet showing higher MSEs compared to Affirm. In contrast, against lightweight PatchTST and TSLANet, Affirm aligns in parameter size but significantly decreases MACs by 66.1% and 77.4%, and MSE by 12.9% and 9.6%, respectively. Remarkably, Affirm slashes both parameters and MACs by over 99% compared to TimesNet, which has a parameter count of 301.7M and MACs of 1226.49G and reduces MSE by 14.3%. This drastic cut in computational demand, coupled with improved performance, positions Affirm as a lightweight yet powerful alternative.

Method	MACs	Parameters	MSE
Transformer	4.03G	13.61M	0.491
Informer	3.93G	14.38M	0.399
Autoformer	4.41G	14.91M	0.412
FEDfomrer	4.41G	20.68M	0.264
PatchTST	5.07G	1.5M	0.248
FiLM	5.97G	14.91M	0.268
TimesNet	1226.49G	301.7M	0.255
TSLANet	7.6G	**1.4M**	0.239
Affirm	**1.72G**	1.7M	**0.216**

Table 4: Number of training parameters, MACs, and MSE of TSF models under look-back window =96 and forecasting horizon=720 on the large Electricity dataset.

Sensitivity Analysis

Affirm involves several hyperparameters requiring careful tuning for optimal performance. We conducted a sensitivity analysis of mask_ratio and dropout on ETTh1 and Weather. Table 5 reveals that the best performance is achieved with a mask_ratio of 0.4. Reasonable parameters can improve generalization and enhance robustness to adapt to unstable and data loss scenarios.

mask_ratio	p=0.01	p=0.1	p=0.2	p=0.4	p=0.5	p=0.6
ETTh1	0.371	0.369	0.366	**0.363**	0.365	0.367
Weather	0.159	0.152	0.150	**0.146**	0.149	0.151

Table 5: Sensitivity experiments of mask_ratio.

Conclusion

We introduce Affirm, a novel lightweight framework for time series forecasting, with an innovative combination of Mamba with adaptive Fourier filters as a potent alternative to Transformers. It introduces a dual interactive Mamba and uses dropout regularization to improve parameter selectivity and prevent overfitting. Besides, it integrates adaptive global and local Fourier filters in the frequency domain, where the local neural operator utilizes learnable thresholds to filter noise. Experiments demonstrate Affirm's ability to achieve SOTA ability and efficiency trade-offs in TSF, particularly under noisy conditions and various data sizes. Complexity analysis confirms Affirm's superiority with significantly lower computational costs. Moreover, our in-depth layer-wise scaling analysis reveals that Affirm outperforms Transformers on smaller datasets and exhibits enhanced scalability with increased layers, especially in larger datasets. Affirm opens up new avenues for TSF as a foundation model.

Acknowledgments

We thank the anonymous reviewers for their helpful feedbacks. The work is supported by Zhejiang Provincial Science and Technology Plan Project (No. 2023C03183), Key Scientific Research Base for Digital Conservation of Cave Temples (Zhejiang University).

References

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

Behrouz, A.; Santacatterina, M.; and Zabih, R. 2024. Mambamixer: Efficient selective state space models with dual token and channel selection. *arXiv preprint arXiv:2403.19888*.

Cai, X.; Zhu, Y.; Wang, X.; and Yao, Y. 2024. MambaTS: Improved Selective State Space Models for Long-term Time Series Forecasting. *arXiv preprint arXiv:2405.16440*.

Eldele, E.; Ragab, M.; Chen, Z.; Wu, M.; and Li, X. 2024. Tslanet: Rethinking transformers for time series representation learning. *arXiv preprint arXiv:2404.08472*.

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

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

Gu, A.; Goel, K.; and Ré, C. 2021. Efficiently modeling long sequences with structured state spaces. *In International Conference on Learning Representations, 2021*.

Gu, A.; Johnson, I.; Goel, K.; Saab, K.; Dao, T.; Rudra, A.; and Ré, C. 2021. Combining recurrent, convolutional, and continuous-time models with linear state space layers. *Advances in neural information processing systems*, 34: 572–585.

Guo, H.; Zhu, H.; Wang, J.; Prahlad, V.; Ho, W. K.; de Silva, C. W.; and Lee, T. H. 2024. Remaining Useful Life Prediction via Frequency Emphasizing Mix-Up and Masked Reconstruction. *IEEE Transactions on Artificial Intelligence*.

He, K.; Chen, X.; Xie, S.; Li, Y.; Dollár, P.; and Girshick, R. 2022. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 16000–16009.

Huang, H.; Chen, M.; and Qiao, X. 2024. Generative Learning for Financial Time Series with Irregular and Scale-Invariant Patterns. In *The Twelfth International Conference on Learning Representations (ICLR)*.

Huang, Z.; Zhang, Z.; Lan, C.; Zha, Z.-J.; Lu, Y.; and Guo, B. 2023. Adaptive Frequency Filters As Efficient Global Token Mixers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 6049–6059.

Jin, M.; Wang, S.; Ma, L.; Chu, Z.; Zhang, J. Y.; Shi, X.; Chen, P.-Y.; Liang, Y.; Li, Y.-F.; Pan, S.; et al. 2023. Time-llm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728*.

Kim, T.; Kim, J.; Tae, Y.; Park, C.; Choi, J.-H.; and Choo, J. 2021. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*.

Li, G.; Bu, W.; and Yang, H. 2023. Research on noise reduction method for ship radiate noise based on secondary decomposition. *Ocean Engineering*, 268: 113412.

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

Lin, Y.; Koprinska, I.; and Rana, M. 2021. SSDNet: State space decomposition neural network for time series forecasting. In *2021 IEEE International Conference on Data Mining (ICDM)*, 370–378. IEEE.

Liu, C.; Lin, J.; Wang, J.; Liu, H.; and Caverlee, J. 2024a. Mamba4rec: Towards efficient sequential recommendation with selective state space models. *arXiv preprint arXiv:2403.03900*.

Liu, Y.; Hu, T.; Zhang, H.; Wu, H.; Wang, S.; Ma, L.; and Long, M. 2024b. itransformer: Inverted transformers are effective for time series forecasting. *In International Conference on Learning Representations (ICLR), 2024*.

Liu, Z.; Ma, Q.; Ma, P.; and Wang, L. 2023. Temporal-frequency co-training for time series semi-supervised learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 8923–8931.

Long, Q.; Fang, Z.; Fang, C.; Chen, C.; Wang, P.; and Zhou, Y. 2024. Unveiling Delay Effects in Traffic Forecasting: A Perspective from Spatial-Temporal Delay Differential Equations. In *Proceedings of the ACM on Web Conference 2024*, 1035–1044.

Nie, Y.; Nguyen, N. H.; Sinthong, P.; and Kalagnanam, J. 2023. A time series is worth 64 words: Long-term forecasting with transformers. *In International Conference on Learning Representations (ICLR), 2023*.

Park, J.; Park, J.; Xiong, Z.; Lee, N.; Cho, J.; Oymak, S.; Lee, K.; and Papailiopoulos, D. 2024. Can mamba learn how to learn? a comparative study on in-context learning tasks. *In International Conference on Machine Learning (ICML), 2024*.

Pitas, I. 2000. *Digital image processing algorithms and applications*. John Wiley & Sons.

Rabiner, L. R.; and Gold, B. 1975. *Theory and Application of Digital Signal Processing:(by) Lawrence R. Rabiner (and) Bernard Gold*. Prentice-Hall.

Rao, Y.; Zhao, W.; Zhu, Z.; Lu, J.; and Zhou, J. 2021. Global filter networks for image classification. *Advances in neural information processing systems*, 34: 980–993.

Smith, J. T.; Warrington, A.; and Linderman, S. W. 2022. Simplified state space layers for sequence modeling. *arXiv preprint arXiv:2208.04933*.

Tang, Y.; Dong, P.; Tang, Z.; Chu, X.; and Liang, J. 2024. Vmrnn: Integrating vision mamba and lstm for efficient and accurate spatiotemporal forecasting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5663–5673.

Wang, C.; Tsepa, O.; Ma, J.; and Wang, B. 2024a. Graph-mamba: Towards long-range graph sequence modeling with selective state spaces. *arXiv preprint arXiv:2402.00789*.

Wang, L.; Li, D.; Dong, S.; Meng, X.; Zhang, X.; and Hong, D. 2024b. PyramidMamba: Rethinking Pyramid Feature Fusion with Selective Space State Model for Semantic Segmentation of Remote Sensing Imagery. *arXiv preprint arXiv:2406.10828*.

Wang, S.; Wu, H.; Shi, X.; Hu, T.; Luo, H.; Ma, L.; Zhang, J. Y.; and Zhou, J. 2024c. Timemixer: Decomposable multiscale mixing for time series forecasting. *The Twelfth International Conference on Learning Representations (ICLR)*.

Wang, Y.; Han, Y.; Wang, H.; and Zhang, X. 2024d. Contrast everything: A hierarchical contrastive framework for medical time-series. *Advances in Neural Information Processing Systems*, 36.

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

Wu, Y.; Meng, X.; He, Y.; Zhang, J.; Zhang, H.; Dong, Y.; and Lu, D. 2024a. Multi-view Self-Supervised Contrastive Learning for Multivariate Time Series. In *Proceedings of the 32nd ACM International Conference on Multimedia*, 9582–9590.

Wu, Y.; Meng, X.; Zhang, J.; He, Y.; Romo, J. A.; Dong, Y.; and Lu, D. 2024b. Effective LSTMs with seasonal-trend decomposition and adaptive learning and niching-based backtracking search algorithm for time series forecasting. *Expert Systems with Applications*, 236: 121202.

Wu, Z.; Gong, Y.; and Zhang, A. 2024. DTMamba: Dual Twin Mamba for Time Series Forecasting. *arXiv preprint arXiv:2405.07022*.

Xu, Z.; Zeng, A.; and Xu, Q. 2023. FITS: Modeling time series with $10k$ parameters. *arXiv preprint arXiv:2307.03756*.

Yi, K.; Zhang, Q.; Fan, W.; Wang, S.; Wang, P.; He, H.; An, N.; Lian, D.; Cao, L.; and Niu, Z. 2024. Frequency-domain MLPs are more effective learners in time series forecasting. *Advances in Neural Information Processing Systems*, 36.

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

Zeng, C.; Liu, Z.; Zheng, G.; and Kong, L. 2024. C-Mamba: Channel Correlation Enhanced State Space Models for Multivariate Time Series Forecasting. *arXiv preprint arXiv:2406.05316*.

Zhang, Y.; and Yan, J. 2023. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The eleventh international conference on learning representations (ICLR)*.

Zheng, Z.; and Wu, C. 2024. U-shaped vision mamba for single image dehazing. *arXiv preprint arXiv:2402.04139*.

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

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

Zhou, T.; Niu, P.; Sun, L.; Jin, R.; et al. 2023. One fits all: Power general time series analysis by pretrained lm. *Advances in neural information processing systems*, 36: 43322–43355.

Zhu, L.; Liao, B.; Zhang, Q.; Wang, X.; Liu, W.; and Wang, X. 2024. Vision mamba: Efficient visual representation learning with bidirectional state space model. *In International Conference on Machine Learning (ICML), 2024*.