

FTMixer: Frequency and Time Domain Representations Fusion for Time Series Forecasting

Zhengen Li

lzhengnan389@gmail.com
Communication University of China
China, Beijing

Xilong Cheng

xilongcheng330@gmail.com
Communication University of China
China, Beijing

Yunxiao Qin

qinyunxiao@cuc.edu.cn
Communication University of China
China, Beijing

Yuting Tan

YutingTan@cuc.edu.cn
Communication University of China
China, Beijing

Abstract

Time series data can be represented in both the time and frequency domains, with the time domain emphasizing local dependencies and the frequency domain highlighting global dependencies. To harness the strengths of both domains in capturing local and global dependencies, we propose a novel Frequency and Time Domain Mixer (FTMixer) method. To exploit the global characteristics of the frequency domain, we introduce a novel Frequency Channel Convolution (FCC) module, designed to capture global inter-series dependencies. Inspired by the windowing concept in frequency domain transformations, we further propose a novel Windowed Frequency-Time Convolution (WFTC) module, which captures local dependencies by leveraging both frequency domain representations obtained from windowed transformations and time domain representations. Notably, FTMixer employs the Discrete Cosine Transformation (DCT) with real numbers instead of the complex-number-based Discrete Fourier Transformation (DFT), enabling direct utilization of modern deep learning operators in the frequency domain. Extensive experimental results across seven real-world long-term time series datasets demonstrate the superiority of FTMixer, in terms of both forecasting performance and computational efficiency. Code is available here.

CCS Concepts

• **Mathematics of computing** → **Time series analysis**.

Keywords

Time series forecasting

ACM Reference Format:

Zhengen Li, Yunxiao Qin, Xilong Cheng, and Yuting Tan. 2018. FTMixer: Frequency and Time Domain Representations Fusion for Time Series Forecasting. In . ACM, New York, NY, USA, 12 pages. <https://doi.org/XXXXXXX.XXXXXXX>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

Conference'17, July 2017, Washington, DC, USA

© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-XXXX-X/18/06
<https://doi.org/XXXXXXX.XXXXXXX>

1 Introduction

Time series forecasting finds wide application across various domains, including traffic flows [40], ECL consumption [16, 36, 46], and weather forecasting [4, 20, 22, 25]. In recent years, advancements in deep learning have revolutionized time series forecasting [24, 26, 31, 44]. Among these advancements, Transformer-based methods [22, 30, 40, 49, 53] and MLP-based methods [7, 12, 15, 35, 48] dominate this field. Most previous methods have concentrated on learning time series in the time domain and have achieved promising performance [32].

On the other hand, recent studies [6, 11, 18, 39, 50, 52] have demonstrated that, under certain conditions (e.g., early stopping or large step size), deep neural networks (DNNs) tend to gravitate towards simpler solutions. This phenomenon, known as implicit sparse regularization [18, 52], suggests that deep regression models focus on the most influential data points within the input sequence for regression tasks.

In the context of time series forecasting, when the model operating in the time domain, these influential data points correspond to specific time instants, enabling the model to focus on the critical moments that are most predictive of future values. In contrast, when operating in the frequency domain, implicit sparse regularization directs the model's focus towards the most significant frequency components. Since each frequency component represents a sinusoid in the time domain, this focus allows the model to capture the primary periodicities of the data, thereby preserving essential patterns while effectively filtering out noise [45, 47, 53]. Figure 1(a) shows that the weights of the frequency domain Fully Connected layer reveal prominent diagonal patterns, highlighting the model's ability to capture periodicity by focusing on significant frequencies. In contrast, the time domain Fully Connected layer's weights must manage data across periodic intervals to identify periodic patterns, resulting in more complex and less sparse representations. This increased sparsity in the frequency domain enhances Deep Neural Network (DNN) learning by improving feature extraction and reducing overfitting [18, 28]. Figure 1(b) further illustrates that outputs from the frequency domain are smoother and capture more periodic information, while time domain outputs emphasize local dependencies.

Several studies have leveraged the frequency domain to analyze time series data [9, 38, 43, 45, 47, 53]. For example, TSLA-Net [9] employs frequency domain adaptive denoising to enhance the

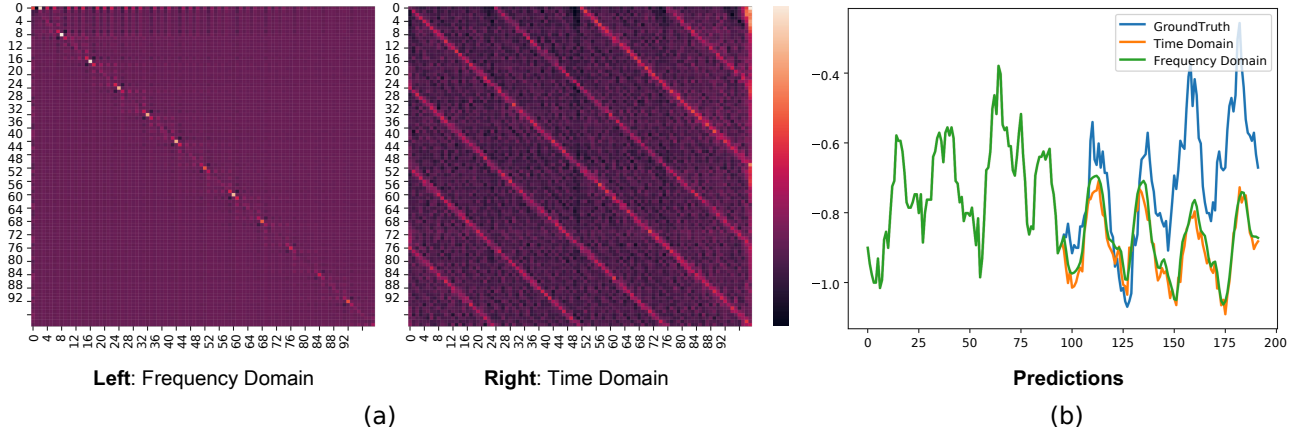


Figure 1: (a) Visualizations of the Fully Connected (FC) layer weights learned in the time and frequency domains on the ETTh1 dataset, with both the input and output length equal to 96, resulting in a 96×96 weight matrix (y -axis: the output, x -axis: the input). Note that we train the frequency domain FC layer by employing the Discrete Cosine Transform (DCT). From the FC layer weight visualizations, we can see that learning in the frequency domain identifies clearer diagonal dependencies and key patterns than in the time domain. (b) Predictions of the frequency domain FC layer and the time domain FC layer. The frequency domain output is smoother and emphasizes periodic information with smaller MSE=0.379, while the time domain output captures more local dependencies with larger MSE=0.383.

model’s capability to identify long-term periodic patterns and improve computational efficiency. Similarly, TimesNet [41] utilizes the Fast Fourier Transform (FFT) to detect periodicities in time series data and performs convolution based on these identified periodic components.

Despite advancements in leveraging the frequency domain for time series analysis, two major challenges still remain: **1) Handling Complex Number Representations.** Existing methods often rely on the Discrete Fourier Transform (DFT) [41, 43, 53], which introduces complex representations of time series data. Deep learning techniques such as Batch Normalization [14] and activation functions [10] are not well-suited for these complex numbers. Although it is possible to process the real and imaginary parts separately with distinct models to adapt complex numbers to deep learning techniques, this approach increases the number of parameters and computational complexity, and may perform not well. The experimental result in Table 5 demonstrate the unsatisfactory performance of this approach. **2) Loss of Local information.** Global frequency domain transformations mainly capture global dependencies, potentially masking critical variations and phenomena, such as sudden spikes and irregular patterns [8, 13, 33, 34, 42], which are essential for accurate predictions and understanding time series dynamics [27].

To address aforementioned challenges, we propose a method that effectively combines insights from both the time domain and the frequency domain: Frequency and Time domain Mixer (FTMixer). First, to fully utilize the frequency domain with deep learning models, we employ the Discrete Cosine Transformation (DCT) [2]. Unlike the Discrete Fourier Transform (DFT) [2, 34], which involves complex numbers, the DCT operates exclusively on real numbers, making it more compatible with modern deep learning techniques.

Additionally, to capture inter-series global dependencies, we propose a novel Frequency Channel Convolution (FCC) module. The FCC embeds the entire sequence in the frequency domain before performing convolution, allowing for a comprehensive analysis of global patterns. To enhance the capture of local dependencies, we draw inspiration from the windowed Discrete Fourier Transform (DFT) [34, 42] and introduce Windowed Frequency-Time Convolution (WFTC) module. The WFTC segments the time series into patches of varying scales, applies frequency domain transformations within each patch, and then performs convolution across these patches to effectively capture local variations. After extracting frequency domain representations, we transform them back to the time domain and integrate them with the results of the convolution performed directly in the time domain on the patches. We use Depth-Wise Separable Convolution to process features extracted by WFTC, balancing efficiency with performance. The outputs of the Depth-Wise Separable Convolution and FCC are added together and passed through a projection layer to yield the final model output.

Moreover, we propose the Dual-Domain Loss Function (DDLDF), which computes losses separately in the time and frequency domains. Leveraging the DCT’s ability to concentrate energy into fewer coefficients and operate with real numbers, this loss function improves the model’s ability to capture and utilize domain-specific features effectively.

Contribution. In this work, we explore the potential of integrating time and frequency domains for time series forecasting and propose a novel approach, FTMixer. We incorporate the Discrete Cosine Transform (DCT) into time series forecasting and introduce the Frequency Capture Convolution (FCC) module to capture global dependencies. Inspired by windowed DCT, we propose the Windowed

Frequency-Time Convolution (WFTC) module to capture local dependencies across both time and frequency domains. Additionally, we introduce the Dual-Domain Loss Function (DDLDF) to leverage the strengths of both domains. Extensive experiments across seven datasets demonstrate that FTMixer outperforms state-of-the-art methods.

2 Related Work

2.1 Time Series Forecasting

Time series forecasting plays a crucial role in various domains, including finance, public health, and weather forecasting [40]. Recent years have witnessed significant development in this field driven by deep learning models specifically designed for time series tasks. Among these models, three prominent architectures have garnered considerable attention: Multi-Layer Perceptrons (MLPs), Transformers, and Temporal Convolutional Networks (TCNs).

Inspired by their success in natural language processing, Transformers have been adapted for time series analysis with remarkable results (e.g., [17], [29], [45]). Examples include Autoformer [40], which utilizes attention mechanisms to decompose sequences, PatchTST [30] which segments sequences inspired by the Vision Transformer (ViT) architecture, and iTransformer [22] that embeds the entire sequence then computing attention across channel dimensions.

Known for their simplicity and effectiveness, MLPs have also found application in time series analysis (e.g., [15], [23], [35], [19], [48], [43], [7], [3]). DLinear [48], for instance, performs trend-season decomposition and learns using two MLPs. RLinear [15] implements reversible instance norm and achieves impressive performance. Additionally, FITS [43] directly learns in the frequency domain, leading to surprising results.

Temporal Convolutional Networks (TCNs) are another class of deep learning models excelling at capturing local dependencies within time series data (e.g., [4], [41], [37], [26]). TimesNet [41] utilizes CNN for feature extraction, with a particular focus on leveraging Fast Fourier Transform (FFT) for periodicity extraction. ModernTCN [26], drawing inspiration from transformers, captures inter-series and cross-time information simultaneously. ConvTimeNet [4] proposes a novel patch method to determine the suitable length of the patch window, enhancing the adaptability of TCNs to various time series datasets.

2.2 Frequency-Aware Time Series Forecasting

Several successful approaches have demonstrated the value of incorporating frequency domain information. TSLA-Net [9] employs frequency domain adaptive denoising to enhance the model's capability to identify long-term periodic patterns and improve computational efficiency. Similarly, TimesNet [41] utilizes the Fast Fourier Transform (FFT) to detect periodicities in time series data and performs convolution based on these identified periodic components. FreTS [47] forecasts time series by leveraging both inner-series and inter-series information. On the other hand, FITS [43] achieves improved performance by training sequences directly in the frequency domain using a fully connected layer. However, a single linear model often proves insufficient for capturing non-linear patterns in the frequency domain. Additionally, the effectiveness of

traditional deep learning techniques like activation functions and batch normalization on complex number data (used in the DFT) remains uncertain.

This work addresses these limitations by introducing the Discrete Cosine Transform (DCT) for the first time in time series analysis. Compared to the Discrete Fourier Transform (DFT) [2], DCT operates exclusively on real numbers, making it more suitable for modern deep learning techniques. Furthermore, DCT utilizes only amplitude to represent the frequency domain information, simplifying the computation of the loss function in the frequency domain. These advantages of DCT pave the way for a novel and potentially more effective approach to frequency-aware time series forecasting.

2.3 Implicit Sparse Regularisation

Recent studies [11, 18, 39, 50, 52] have shown that, under specific conditions such as early stopping or large step sizes, deep neural networks (DNNs) naturally evolve towards simpler solutions. Specifically, [50] shows that when gradient descent is applied directly to the residual sum of squares with sufficiently small initial values, and proper early stopping rules are employed, the iterates converge to a nearly sparse, rate-optimal solution that often surpasses explicitly regularized approaches. Similarly, [18] proves that early stopping tends to lead models towards sparser solutions. Additionally, [6] demonstrates that if an exact solution exists, vanilla gradient flow for the overparameterized loss functional converges to a good approximation of the solution with minimal ℓ_1 -norm.

3 Methodology

3.1 Preliminary

3.1.1 Problem Definition. Let $[X_1, X_2, \dots, X_T] \in \mathbb{R}^{N \times T}$ stand for the regularly sampled multi-channel time series dataset with N series and T timestamps, where $X_t \in \mathbb{R}^N$ denotes the multi-channel values of N distinct series at timestamp t . We consider a time series lookback window of length- L at each timestamp t as the model input, namely $\mathbf{X}_t = [X_{t-L+1}, X_{t-L+2}, \dots, X_t] \in \mathbb{R}^{N \times L}$; also, we consider a horizon window of length- τ at timestamp t as the prediction target, denoted as $\mathbf{Y}_t = [X_{t+1}, X_{t+2}, \dots, X_{t+\tau}] \in \mathbb{R}^{N \times \tau}$. Then the time series forecasting formulation is to use historical observations \mathbf{X}_t to predict future values \mathbf{Y}_t . For simplicity, we shorten the model input \mathbf{X}_t as $\mathbf{X} = [X_1, X_2, \dots, X_L] \in \mathbb{R}^{N \times L}$ and reformulate the prediction target as $\mathbf{Y} = [X_{L+1}, X_{L+2}, \dots, X_{L+\tau}] \in \mathbb{R}^{N \times \tau}$, in the rest of the paper.

3.1.2 Discrete Cosine Transformation. Our methodology utilizes the Discrete Cosine Transform (DCT) to convert input data into the frequency domain. This section provides an overview of the DCT and its relationship to the Discrete Fourier Transform (DFT).

The Cosine Transform is a variant of the Fourier Transform that focuses exclusively on the cosine components [1]. It is particularly advantageous for functions with symmetry, simplifying the transformation process compared to the Fourier Transform, which includes both sine and cosine components.

The continuous Fourier Transform of a function $f(t)$ is given by:

$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-i\omega t} dt,$$

where $F(\omega)$ represents the frequency domain representation of $f(t)$. For even functions $f_e(t)$, the Fourier Transform can be expressed purely in terms of cosine functions:

$$F(\omega) = \int_{-\infty}^{\infty} f_e(t) \cos(\omega t) dt.$$

This relationship highlights the efficiency of cosine components for symmetric functions. We formalize this connection between the Discrete Cosine Transform (DCT) and the Discrete Fourier Transform (DFT) with the following theorem:

The Discrete Cosine Transform (DCT) of a sequence can be derived from the Discrete Fourier Transform (DFT) of a symmetrically extended version of the sequence [1]. The DCT for a sequence \mathbf{x} of length L is defined as:

$$\bar{x}_k = \sum_{n=0}^{L-1} x_n \cos\left(\frac{\pi}{L} \left(n + \frac{1}{2}\right) k\right), \quad (1)$$

where x_n is the n -th element of the sequence \mathbf{x} , and \bar{x}_k denotes the k -th frequency component in the DCT frequency domain coefficients, with $k \in \{0, 1, \dots, L-1\}$.

Using Eq. 1, we obtain $\bar{\mathbf{x}} = [\bar{x}_0, \bar{x}_1, \dots, \bar{x}_{L-1}]$, representing the frequency features of \mathbf{x} .

The DCT is reversible, allowing the transformation of frequency domain coefficients back to the time domain through the inverse Discrete Cosine Transform (iDCT):

$$x_n = \frac{1}{2} \bar{x}_0 + \sum_{k=1}^{L-1} \bar{x}_k \cos\left(\frac{\pi}{L} \left(k + \frac{1}{2}\right) n\right). \quad (2)$$

By employing the DCT, our methodology effectively transitions input data into the frequency domain. The DCT, renowned in signal processing, emphasizes cosine components and operates efficiently with real numbers, making it well-suited for integration with deep learning frameworks.

3.2 Overall Architecture

To address the challenge of capturing both local and global patterns in time series data, we introduce the **Frequency and Time domain Mixer** (FTMixer) method. As shown in Figure 2, FTMixer incorporates two key modules: Frequency Channel Convolution (FCC) and Windowed Frequency-Time Convolution (WFTC).

The FCC module is designed to capture inter-series dependencies in the frequency domain, enhancing the model's ability to detect global patterns that may be missed in the time domain. Meanwhile, the WFTC module employs multi-scale windowing to capture detailed local frequency information, addressing the limitation of traditional methods that rely solely on global frequency representations.

These components work together to balance local and global feature extraction, improving overall performance in time series forecasting.

The model structure of FTMixer is summarized as follows:

$$\begin{cases} \mathbf{Z}_{\text{FCC}} = f_{\text{FCC}}(\mathbf{X}), \\ \mathbf{Z}_{\text{DS}} = f_{\text{DS}}(\text{Concate}(f_{\text{WFTC}}(\mathbf{X}))), \\ \mathbf{Z} = \mathbf{Z}_{\text{FCC}} + \mathbf{Z}_{\text{DS}}, \\ \hat{\mathbf{Y}} = f_{\text{Pre}}(\mathbf{Z}), \end{cases} \quad (3)$$

Here, $\hat{\mathbf{Y}}$ denotes the model's output, and \mathbf{X} represents the input time series. f_{WFTC} applies the WFTC module to each channel of \mathbf{X} , with f_{DS} and f_{Pre} representing depth-wise separable convolution (DS-Conv) and the model predictor, respectively.

3.3 Frequency Channel Convolution

The FCC module is designed to capture global inter-series dependencies in the frequency domain. Standard convolution tends to emphasize local dependencies, which usually overlook broader, global patterns due to its inherent focus on local features. To address this limitation, we apply the Discrete Cosine Transform (DCT) to each channel of the input sequence, converting it into the frequency domain, which can be formulated as:

$$\mathbf{X}_f = \text{Embedding}(\text{DCT}(\mathbf{X})) \quad (4)$$

\mathbf{X}_f represents the frequency domain representation of the input \mathbf{X} . The Discrete Cosine Transform (DCT) is applied to the input \mathbf{X} , and the resulting frequency domain representation is then embedded along the sequence dimension. Following this transformation, we perform convolution with kernel sizes equal to the variable dimensions, effectively allowing convolution across the entire variable dimension.

$$\mathbf{Z}_{\text{FCC}} = \text{iDCT}(\text{Linear}(\text{Conv1d}(\mathbf{X}_f))) \quad (5)$$

This approach allows the FCC module to effectively capture global dependencies and periodic patterns, enhancing the model's ability to understand long-term trends in time series data.

3.4 Windowed Frequency-Time Convolution

Existing frequency-domain models often concentrate solely on the global frequency representation of entire sequences, which may result in similar representations for distinct time-domain sequences. Inspired by the windowing technique in frequency domain transformations [27, 33], we propose the Windowed Frequency-Time Convolution (WFTC) module to capture fine-grained information by applying the Discrete Cosine Transform (DCT) within multi-scale windows.

In the WFTC module, as illustrated in Figure 2, each channel of the input sequence is initially segmented into patches of various scales. The DCT is then applied within each patch to derive the local frequency domain representation. To capture local dependencies, we perform convolution on these patches. Subsequently, we transform the frequency domain embeddings back to the time domain and add them to the result of the convolution performed directly in the time domain on the patches. This approach enhances the model's ability to capture local dependencies. The overall process can be formulated as:

$$\begin{cases} \mathbf{F}_{P_j} = \text{iDCT}(\text{Conv}(\text{DCT}(\mathbf{P}_j))), \\ \mathbf{Z}_{P_j} = \mathbf{F}_{P_j} + \text{Conv}(\mathbf{P}_j) \\ \tilde{\mathbf{Z}}_{P_j} = \text{Embedding}(\mathbf{Z}_{P_j}) \\ \mathbf{Z}_{\text{WFTC}} = \text{Concate}(\tilde{\mathbf{Z}}_{P_1}, \tilde{\mathbf{Z}}_{P_2}, \dots, \tilde{\mathbf{Z}}_{P_n}), \end{cases} \quad (6)$$

where \mathbf{P}_j denotes the j -th patch (e.g., \mathbf{P}_1 or \mathbf{P}_2 in Figure 2) in the WFTC module. \mathbf{Z}_{P_j} represents the features extracted for the j -th patch following the embedding layer, where embedding is applied along the sequence dimension of each patch in a manner similar to a feed-forward layer. \mathbf{Z}_{WFTC} is the output of the WFTC module

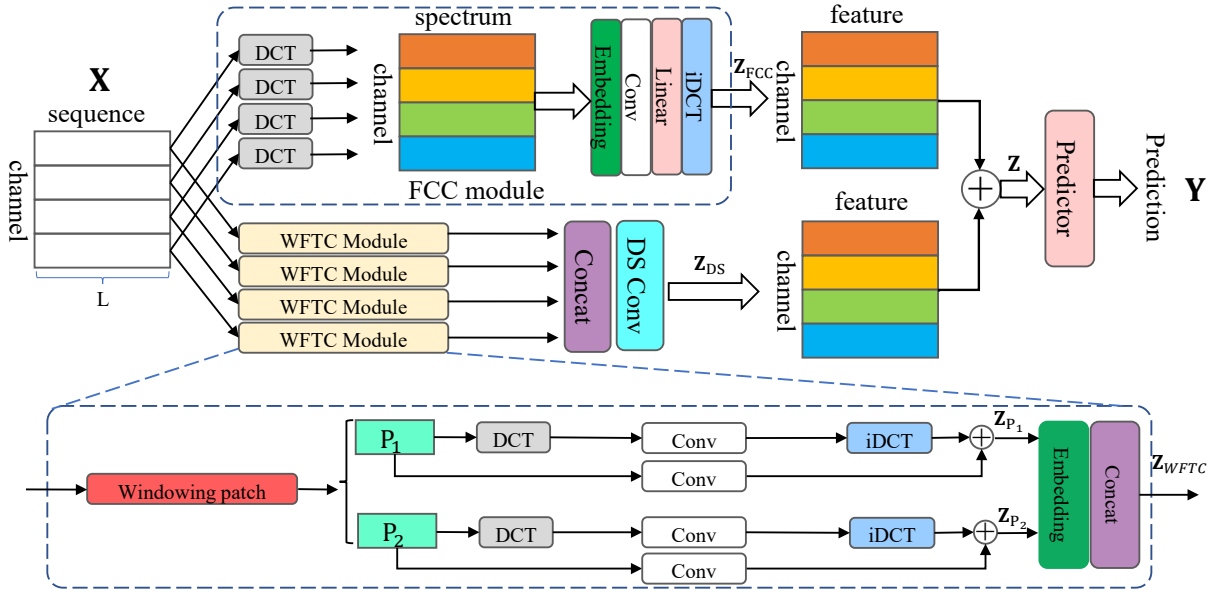


Figure 2: The framework of FTMixer. FTMixer comprises two main modules: FCC and WFTC. An example X containing four channels is visualized here for easier understanding. The model predictor is a Linear layer.

for the current input channel. Note that we separately apply the WFTC module on each input channel, which is shown in Figure 2.

3.5 Depth-Wise Separable Convolution

After obtain Z_{WFTC} for all input channel, we first concatenate them and then apply Depth-Wise Separable Convolution (DS Convolution) [5], to further process the obtained feature. DS Convolution decouples the learning of intra-patch dimensions from the learning of inter-patch dimensions. It is more computationally efficient than vanilla convolution and comprises two components: Depth-Wise Convolution (DW Conv) and Point-Wise Convolution (PW Conv). DW Conv aggregates inter-patch information through grouped convolution, while PW Conv operates akin to a Feed Forward Network (FFN) to extract intra-patch information.

3.6 Dual-Domain Loss Function

To fully leverage the advantages of both the frequency and time domains, we propose a Dual-Domain Loss Function (DDLf) that computes losses separately in each domain. For the time domain, we use Mean Squared Error (MSE), consistent with most time-domain-based methods [22, 26, 30, 40, 48]. In the frequency domain, we employ Mean Absolute Error (MAE), following [38], due to its effectiveness in handling varying magnitudes of frequency components and its stability compared to squared loss functions. The use of Discrete Cosine Transform (DCT) facilitates this approach by concentrating energy into fewer coefficients and utilizing real numbers, which enhances the capture of frequency domain information. The overall loss function of our method is thus formulated as:

$$\begin{cases} \mathcal{L}_{\text{time}} = \text{MSE}(Y - F(X)), \\ \mathcal{L}_{\text{fre}} = \text{MAE}(\text{DCT}(Y) - \text{DCT}(F(X))), \\ \mathcal{L}_{\text{total}} = \mathcal{L}_{\text{time}} + \mathcal{L}_{\text{fre}}. \end{cases} \quad (7)$$

Where $F(X)$ represents the prediction of the model.

4 Experiment

4.1 Experiment Settings

In this section, we evaluate the efficacy of FTMixer on time series forecasting, and anomaly detection tasks. We show that our FTMixer can serve as a foundation model with competitive performance on these tasks.

Table 1: The Statistics of the seven datasets used in our experiments.

Datasets	ETTh1&2	ETTM1&2	Traffic	ECL	Weather
Channels	7	7	862	321	21
Timesteps	17,420	69,680	17,544	26,304	52,696
Granularity	1 hour	5 min	1 hour	1 hour	10 min

Datasets. We perform all experiments on seven widely-used real-world multi-channel time series forecasting datasets. These datasets encompass diverse domains, including ECL Transformer Temperature (ETTh1, ETTh2, ETTm1, and ETTm2) [51], ECL, Traffic, and Weather, as utilized by Autoformer [40]. For fairness in comparison, we adhere to a standardized protocol [22], dividing all forecasting datasets into training, validation, and test sets. Specifically, we employ a ratio of 6:2:2 for the ETT dataset and 7:1:2 for the remaining datasets, in line with [9, 22, 26, 30, 41]. Refer to Table 1 for an overview of the characteristics of these datasets.

Evaluation protocol. Our evaluation framework, inspired by TimesNet [41], is based on two key metrics: Mean Squared Error (MSE) and Mean Absolute Error (MAE). To ensure fair comparison,

Table 2: Full forecasting results on different prediction lengths $\in \{96, 192, 336, 720\}$. Lower MSE and MAE indicate better performance. We highlight the best performance with red bold text.

Methods		FTMixer		ModernTCN		TSLANet		iTransformer		PatchTST		Crossformer		FEDformer		RLinear		Dlinear		TimesNet	
Methods		—		ICLR 2024		ICML 2024		ICLR 2024		ICLR 2023		ICLR 2023		ICML 2022		ICLR 2022		AAAI 2023		ICLR 2023	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.356	0.388	0.391	0.410	0.382	0.406	0.386	0.405	0.382	0.401	0.423	0.448	0.376	0.419	0.386	0.395	<u>0.375</u>	<u>0.399</u>	0.384	0.402
	192	0.401	0.410	0.416	0.423	0.422	0.435	0.441	0.436	0.428	0.425	0.471	0.474	0.420	0.448	0.437	0.424	<u>0.405</u>	<u>0.416</u>	0.436	0.429
	336	0.422	0.425	<u>0.437</u>	<u>0.435</u>	0.443	0.451	0.487	0.458	0.451	0.436	0.570	0.546	0.459	0.465	0.479	0.446	0.439	0.443	0.491	0.469
	720	0.430	0.454	0.461	0.470	0.461	0.498	0.503	0.491	<u>0.452</u>	<u>0.459</u>	0.653	0.621	0.506	0.507	0.481	0.470	0.472	0.490	0.521	0.500
	Avg	0.402	0.419	0.426	<u>0.434</u>	0.434	0.447	0.454	0.448	0.428	0.430	0.529	0.522	0.440	0.460	0.446	0.434	<u>0.423</u>	0.437	0.458	0.450
ETTh2	96	0.275	0.335	<u>0.279</u>	0.344	0.289	0.351	0.297	0.349	0.285	<u>0.340</u>	0.745	0.584	0.358	0.397	0.288	0.338	0.289	0.353	0.340	0.374
	192	0.336	0.375	<u>0.342</u>	0.387	<u>0.342</u>	0.388	0.380	0.400	0.356	<u>0.386</u>	0.877	0.656	0.429	0.439	0.374	0.390	0.383	0.418	0.402	0.414
	336	0.359	0.398	<u>0.363</u>	<u>0.402</u>	0.371	0.413	0.428	0.432	0.365	0.405	1.043	0.731	0.496	0.487	0.415	0.426	0.448	0.465	0.452	0.452
	720	0.388	0.427	<u>0.390</u>	<u>0.428</u>	0.417	0.450	0.427	0.445	0.395	0.427	1.104	0.763	0.463	0.474	0.420	0.440	0.605	0.551	0.462	0.468
	Avg	0.339	0.383	<u>0.344</u>	0.390	0.355	0.401	0.383	0.407	0.347	<u>0.387</u>	0.942	0.684	0.437	0.449	0.374	0.399	0.431	0.447	0.414	0.427
ETTm1	96	0.284	0.334	0.293	0.345	<u>0.286</u>	<u>0.340</u>	0.334	0.368	0.291	<u>0.340</u>	0.404	0.426	0.379	0.419	0.355	0.376	0.299	0.343	0.338	0.375
	192	0.321	0.361	0.336	0.372	0.329	0.372	0.377	0.391	<u>0.328</u>	<u>0.365</u>	0.450	0.451	0.426	0.441	0.391	0.392	0.335	<u>0.365</u>	0.374	0.387
	336	0.355	0.384	0.370	0.391	<u>0.356</u>	0.387	0.426	0.420	0.365	0.389	0.532	0.515	0.445	0.459	0.424	0.415	0.369	<u>0.386</u>	0.410	0.411
	720	0.415	0.417	0.422	<u>0.419</u>	<u>0.417</u>	0.418	0.491	0.459	0.422	0.423	0.666	0.589	0.543	0.490	0.487	0.450	0.425	0.421	0.478	0.450
	Avg	0.343	0.373	0.355	0.382	<u>0.347</u>	<u>0.380</u>	0.407	0.410	0.352	0.379	0.513	0.495	0.448	0.452	0.414	0.408	0.357	0.379	0.400	0.406
ETTm2	96	0.163	0.252	0.168	0.257	<u>0.167</u>	0.262	0.180	0.264	0.169	<u>0.254</u>	0.287	0.366	0.203	0.287	0.182	0.265	0.167	0.260	0.187	0.267
	192	0.219	0.287	<u>0.225</u>	0.297	0.230	0.305	0.250	0.309	0.230	<u>0.294</u>	0.414	0.492	0.269	0.328	0.246	0.304	0.224	0.303	0.249	0.309
	336	0.269	0.320	<u>0.273</u>	<u>0.328</u>	0.284	0.337	0.311	0.348	0.280	0.329	0.597	0.542	0.325	0.366	0.307	0.342	0.281	0.342	0.321	0.351
	720	0.351	0.377	0.370	0.390	<u>0.368</u>	0.391	0.412	0.407	0.378	<u>0.386</u>	1.730	1.042	0.421	0.415	0.407	0.398	0.397	0.421	0.408	0.403
	Avg	0.250	0.309	<u>0.259</u>	0.318	0.262	0.324	0.288	0.332	0.264	<u>0.316</u>	0.757	0.611	0.305	0.349	0.286	0.327	0.267	0.332	0.291	0.333
Traffic	96	0.362	0.238	0.425	0.298	<u>0.372</u>	<u>0.261</u>	0.395	0.268	0.401	0.267	0.522	0.290	0.587	0.366	0.649	0.389	0.410	0.282	0.593	0.321
	192	0.382	0.252	0.435	0.302	<u>0.388</u>	<u>0.266</u>	0.417	0.276	0.406	0.268	0.530	0.293	0.604	0.373	0.601	0.366	0.423	0.287	0.617	0.336
	336	0.389	0.256	0.446	0.306	<u>0.394</u>	<u>0.269</u>	0.433	0.283	0.421	0.277	0.558	0.305	0.621	0.383	0.609	0.369	0.436	0.296	0.629	0.336
	720	0.425	0.281	0.452	0.311	<u>0.430</u>	<u>0.289</u>	0.467	0.302	0.452	0.297	0.589	0.328	0.626	0.382	0.647	0.387	0.466	0.315	0.640	0.350
	Avg	0.390	0.257	0.440	0.304	<u>0.396</u>	<u>0.271</u>	0.428	0.282	0.420	0.277	0.550	0.304	0.610	0.376	0.627	0.378	0.434	0.295	0.620	0.336
Weather	96	0.143	0.187	0.150	0.204	<u>0.148</u>	<u>0.198</u>	0.174	0.214	0.160	0.204	0.158	0.230	0.217	0.296	0.192	0.232	0.176	0.237	0.172	0.220
	192	0.188	0.232	0.196	0.247	<u>0.194</u>	<u>0.242</u>	0.221	0.254	0.204	0.245	0.206	0.277	0.276	0.336	0.240	0.271	0.220	0.282	0.219	0.261
	336	0.241	0.276	0.247	0.286	<u>0.245</u>	<u>0.282</u>	0.278	0.296	0.257	0.285	0.272	0.335	0.339	0.380	0.292	0.307	0.265	0.319	0.280	0.306
	720	0.318	0.332	0.330	0.339	<u>0.325</u>	<u>0.337</u>	0.358	0.349	0.329	0.338	0.398	0.418	0.403	0.428	0.364	0.353	0.323	0.362	0.365	0.359
	Avg	0.223	0.257	0.231	0.269	<u>0.228</u>	<u>0.265</u>	0.258	0.278	0.238	0.268	0.259	0.315	0.309	0.360	0.272	0.291	0.246	0.300	0.259	0.287
ECL	96	0.127	0.217	0.142	0.345	<u>0.136</u>	<u>0.229</u>	0.148	0.240	0.138	0.230	0.219	0.314	0.193	0.308	0.201	0.281	0.140	0.237	0.168	0.272
	192	0.145	0.235	0.156	0.25	0.152	0.244	0.162	0.253	<u>0.149</u>	<u>0.243</u>	0.231	0.322	0.201	0.315	0.201	0.283	0.153	0.249	0.184	0.289
	336	0.163	0.262	0.174	0.269	<u>0.168</u>	<u>0.262</u>	0.178	0.269	0.169	0.262	0.246	0.337	0.214	0.329	0.215	0.298	0.169	0.267	0.198	0.300
	720	0.199	0.285	0.211	0.297	0.205	<u>0.293</u>	0.225	0.317	0.211	0.299	0.280	0.363	0.246	0.355	0.257	0.331	<u>0.203</u>	0.301	0.220	0.320
	Avg	0.159	0.249	0.171	0.290	<u>0.165</u>	<u>0.257</u>	0.178	0.270	0.167	0.259	0.244	0.334	0.214	0.327	0.219	0.298	0.166	0.264	0.193	0.295

we use prediction lengths of $\{96, 192, 336, 720\}$ and set the historical horizon length to $T = 336$ for our model. For other models, we follow [30, 41] by treating the historical horizon T as a hyper-parameter and using the settings from their original papers, as some models may exhibit performance degradation with increasing historical horizons. For baselines, we report the best results from their original works if their settings match ours; otherwise, we re-run their official codes to ensure fair comparison. Notably, some official codes of baselines contain a bug that drops the last batch during testing¹. We fixed this issue and re-run these baselines. All reported results are averaged over 10 random seeds.

¹We found this bug at: <https://github.com/yuqinie98/PatchTST/issues/7>.

Baseline setting. We compare FTMixer against a variety of state-of-the-art baselines. Transformer-based baselines include iTransformer, PatchTST, Crossformer, FEDformer, and Autoformer. MLP-based baselines include RLinear [15] and DLinear [48]. Besides, we also consider the Convolutional-based baseline SCINet [21], the general-purpose time series models TimesNet, another frequency domain related baseline TSLANet [9], and ModernTCN [26].

4.2 Experimental Results

Quantitative Comparison. Table 2 presents the comprehensive forecasting results, with the top-performing results highlighted in bold red and the second-best results underlined in blue. Lower

Table 3: The ablation experimental results about WFTC and FCC (MSE).

	ETTh1	ECL	Weather	ETTh2
w/o WFTC	0.447	0.186	0.255	0.263
w/o FCC	0.427	0.171	0.242	0.258
Ours	0.402	0.159	0.223	0.250

values of Mean Squared Error (MSE) and Mean Absolute Error (MAE) indicate better predictive performance. It is evident that FTMixer consistently demonstrates the most promising predictive performance across all datasets. On the *ETTh1* dataset, FTMixer outperforms ModernTCN and TSLANet, achieving noticeable reductions in MSE and MAE, which underscores its effectiveness in capturing complex patterns. This trend continues on the *ETTh2* dataset, where FTMixer excels beyond Crossformer and FEDformer, especially in longer forecasting horizons such as 720. Additionally, on the *Traffic* dataset, FTMixer consistently delivers the lowest loss compared to Dlinear and TimesNet.

To provide a deeper understanding of why our method outperforms others, we analyze its performance on three representative datasets: *ETTh1 Dataset*: The ETTh1 dataset features both global and complex local multi-scale dependencies [45]. In this context, FTMixer excels by leveraging its ability to capture both global dependencies through FCC and local dependencies through WFTC, resulting in superior performance. *Weather Dataset*: Characterized by local dependencies and significant noise [45], this dataset poses a challenge. FTMixer effectively captures intricate patterns despite the noise, demonstrating its promising performance in noisy environments. *ECL Dataset*: With significant global dependencies [36], this dataset highlights FTMixer’s capability to excel even when global patterns are prevalent. FTMixer consistently outperforms other models, showcasing its effectiveness in capturing both global and local dependencies across diverse datasets.

5 Model Analysis

5.1 Ablation Study

In this subsection, we assess the contributions of each component of our FTMixer method by removing it from FTMixer. The results, presented in Tables 3 and 4, underscore the significance of each component. The experimental settings for this ablation study are consistent with those employed in the main experiments.

The Effectiveness of WFTC. The Windowed Frequency-Time Convolution (WFTC) module is designed to capture local dependencies. It is particularly effective on datasets with pronounced local patterns, such as Weather and ETTh1. As demonstrated in Table 3, the removal of WFTC leads to a marked decrease in performance. For instance, without WFTC, the mean squared error (MSE) increases to 0.447 on the ETTh1 dataset and to 0.186 on the ECL dataset. This decline emphasizes the WFTC’s **critical role in extracting complex multi-scale local features** and improving the model’s ability to handle intricate local dependencies.

The Effectiveness of FCC. The Frequency Channel Convolution (FCC) module focuses on capturing global dependencies and

Table 4: The ablation experiments about loss function (MSE).

	ETTh1	Weather	ECL	ETTh2
w/o Fre Loss	0.419	0.231	0.169	0.262
w/o Time Loss	0.418	0.246	0.164	0.256
Ours	0.402	0.223	0.159	0.250

Table 5: Experimental results comparing the DCT and DFT versions of our model (MSE). Although the DCT version shows only a marginal improvement over the DFT version in terms of performance, it is more efficient as it avoids the additional complexity of separately processing the real and imaginary parts of complex numbers.

	ETTh1	ECL	Weather	ETTh2
Ours (DFT Version)	0.407	0.164	0.226	0.254
Ours (DCT Version)	0.402	0.159	0.223	0.250

inter-series relationships, which are crucial for datasets with complex inter-series structures. As shown in Table 3, omitting FCC results in significant performance degradation, particularly on the ECL dataset, where the MSE rises from 0.159 to 0.171 when FCC is removed. This indicates FCC’s essential role in **leveraging global frequency domain information to enhance forecasting accuracy**.

The Effectiveness of DDLF. We assess the effectiveness of the Dual-Domain Loss Function (DDLDF) through ablation experiments on the ETTh1 and Weather datasets. For ETTh1, which features complex seasonal patterns and long-term trends [40, 45], excluding the frequency domain loss component, \mathcal{L}_{fre} , results in an increased MSE from 0.402 to 0.419. This suggests that frequency domain information is crucial for capturing periodic trends. Removing the time domain loss component, $\mathcal{L}_{\text{time}}$, also degrades performance, raising the MSE to 0.418, indicating the importance of capturing local dependencies. Similarly, in the Weather dataset, which involves less pronounced periodic patterns but has significant local variations [45], the MSE increases from 0.223 to 0.231 when \mathcal{L}_{fre} is omitted, and to 0.246 when $\mathcal{L}_{\text{time}}$ is removed. These findings underscore the necessity of integrating both frequency and time domain losses to effectively capture the diverse features present in different datasets, thereby enhancing overall forecasting accuracy.

5.2 DCT vs DFT

In this section, we replace DCT with DFT to compare their performance under the same experimental setup as the main experiments. Since DFT produces complex numbers, we separately predict the real and imaginary parts. These components are then combined and transformed back to the time domain using the inverse DFT (IDFT). Processing real and imaginary parts independently introduces additional parameters and computational overhead, making this approach less efficient compared to using DCT. As shown in Table 5, the DCT version of the model consistently outperforms the DFT version. For example, on the ETTh1 dataset, DCT achieves a Mean Squared Error (MSE) of 0.402, whereas DFT results in an MSE of 0.407. We hypothesize that the superior performance of

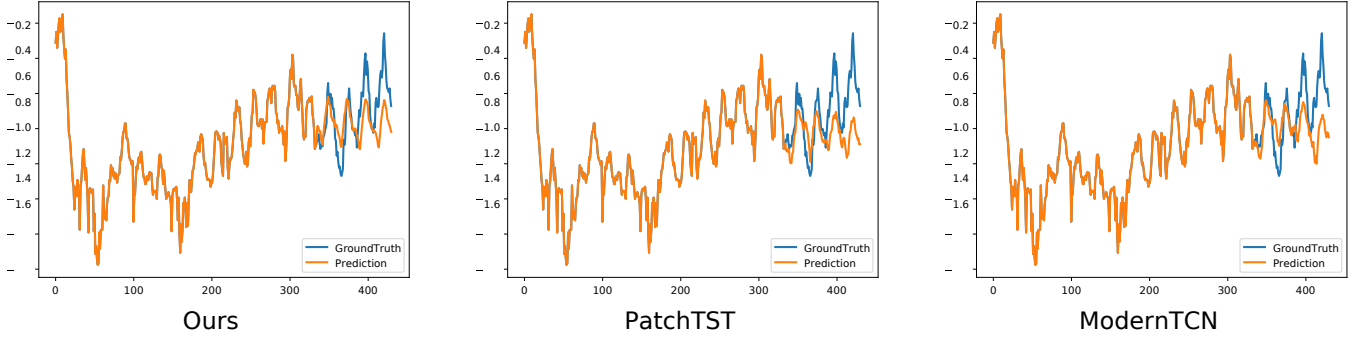


Figure 3: The visualization of predictions by FTMixer, PatchTST, and ModernTCN on the ETTh1 dataset shows that the proposed FTMixer achieves the best performance with an MSE of 0.356, compared to PatchTST’s 0.382 and ModernTCN’s 0.375.

Table 6: The computational cost comparison between the proposed FTMixer, iTransformer, and PatchTST. The batch size for ETTh1 dataset here is 32, while the batch size for ECL dataset here is 1.

Cost	Benchmark	FTMixer	ModernTCN	PatchTST
Time	ETTh1	0.021s	0.043s	0.027s
	ECL	0.023s	0.141s	0.036s
Memory	ETTh1	272MB	316MB	448MB
	ECL	304MB	3350MB	828MB

Table 7: The performance of the proposed FTMixer under diverse input lengths.

	ETTh1		Weather		Traffic	
	MSE	MAE	MSE	MAE	MSE	MAE
96	0.368	0.390	0.165	0.207	0.392	0.261
192	0.360	0.389	0.153	0.195	0.376	0.247
336	0.356	0.388	0.143	0.187	0.362	0.238
720	0.355	0.387	0.142	0.189	0.354	0.231

DCT over DFT is due to the inefficiency and additional complexity involved in separately processing real and imaginary components, which impacts the overall forecasting accuracy.

5.3 Computational Efficiency

In this subsection, we delve into the computational efficiency analysis of the proposed FTMixer. FTMixer stands out as a purely Temporal Convolutional Network (TCN)-based method, distinguished by its streamlined computational overhead. Unlike Transformer-based methodologies, which typically entail a computational complexity of $O(T^2)$ per layer, FTMixer significantly mitigates this burden to $O(KT)$, where K denotes the size of the convolution kernel. The computational complexity of the Feature Vision Convolution (FCC) component within FTMixer is expressed as $O(TMK)$, where M signifies the number of channels and K denotes the size of the convolution kernel. To substantiate the efficiency claims of FTMixer, experiments were conducted with an input length of 336 and an output length of 96 on the ETTh1 and ECL datasets. As

delineated in Table 6, FTMixer showcased superior efficiency when compared to two prominent state-of-the-art methods ModernTCN and PatchTST. ModernTCN’s efficiency diminishes as the number of channels escalates due to the linear complexity of its convolution kernel with respect to channel count. For instance, the ETTh1 dataset comprises 7 channels, while the ECL dataset comprises 321 channels. In contrast, FTMixer **exhibits lower memory costs** under similar conditions, demonstrating its efficiency across varying channel counts.

5.4 Visualization

Here, we visualize the predictions on the ETTh1 dataset. As illustrated in Figure 3, our FTMixer model more effectively captures the primary trends, aligning more closely with the ground truth compared to the two baselines, PatchTST and ModernTCN. In contrast, ModernTCN and PatchTST exhibit more localized features and are less accurate in reflecting the overall trend.

5.5 Varying Input Length

In this section, we evaluate the performance of our model with varying input lengths $L \in \{96, 192, 336, 720\}$ while maintaining a fixed prediction length of 96. As illustrated in Table 7, the performance of FTMixer improves with longer input lengths. For instance, on the ETTh1 dataset, the MSE decreases from 0.368 at $L = 96$ to 0.355 at $L = 720$, demonstrating the model’s enhanced ability in capturing long-term dependencies. Similarly, on the Weather dataset, MSE drops from 0.165 at $L = 96$ to 0.142 at $L = 720$, indicating improved performance in managing complex seasonal and trend patterns. These improvements can be attributed to FTMixer’s Windowed Frequency-Time Convolution (WFTC) and Frequency Channel Convolution (FCC), which effectively capture both local and global patterns. Longer input lengths provide the model with a more comprehensive temporal context, enabling it to better manage and integrate various dependencies present in the data.

6 Conclusion

This paper investigates the potential of combining information from both the time domain and the frequency domain for time series forecasting tasks. We propose a novel method called FTMixer,

which integrates the Discrete Cosine Transform (DCT). Our approach includes the Frequency Convolution Component (FCC) for capturing global dependencies and the Windowed Frequency-Time Convolution (WFTC) module for local dependency extraction in both domains. Extensive experiments demonstrate the effectiveness of FTMixer, showcasing its state-of-the-art performance and computational efficiency. These findings underscore the significant value of frequency domain information and the combined approach of FCC and WFTC in enhancing time series forecasting. We believe these results can inspire further exploration into the frequency domain's role in time series forecasting tasks.

References

- [1] N. Ahmed, T. Natarajan, and K. Rao. Discrete cosine transform. *IEEE Transactions on Computers*, C-23(1):90–93, 1974. doi: 10.1109/T-C.1974.223784.
- [2] N. Ahmed, T. Natarajan, and K. R. Rao. Discrete cosine transform. *IEEE transactions on Computers*, 100(1):90–93, 1974.
- [3] C. Challu, K. G. Olivares, B. N. Oreshkin, F. Garza, M. Mergenthaler-Canseco, and A. Dubrawski. N-hits: Neural hierarchical interpolation for time series forecasting, 2022.
- [4] M. Cheng, J. Yang, T. Pan, Q. Liu, and Z. Li. ConvtimeNet: A deep hierarchical fully convolutional model for multivariate time series analysis. *arXiv preprint arXiv:2403.01493*, 2024.
- [5] F. Chollet. Xception: Deep learning with depthwise separable convolutions, 2017. URL <https://arxiv.org/abs/1610.02357>.
- [6] H.-H. Chou, J. Maly, and H. Rauhut. More is less: Inducing sparsity via overparameterization, 2023. URL <https://arxiv.org/abs/2112.11027>.
- [7] A. Das, W. Kong, A. Leach, R. Sen, and R. Yu. Long-term forecasting with tide: Time-series dense encoder. *arXiv preprint arXiv:2304.08424*, 2023.
- [8] I. Daubechies. The wavelet transform, time-frequency localization and signal analysis. *IEEE Transactions on Information Theory*, 36(5):961–1005, 1990. doi: 10.1109/18.57199.
- [9] E. Eldele, M. Ragab, Z. Chen, M. Wu, and X. Li. Tslanet: Rethinking transformers for time series representation learning. In *International Conference on Machine Learning*, 2024.
- [10] K. Fukushima. Visual feature extraction by a multilayered network of analog threshold elements. *IEEE Transactions on Systems Science and Cybernetics*, 5(4): 322–333, 1969. doi: 10.1109/TSSC.1969.300225.
- [11] D. Gissin, S. Shalev-Shwartz, and A. Daniely. The implicit bias of depth: How incremental learning drives generalization, 2019. URL <https://arxiv.org/abs/1909.12051>.
- [12] L. Han, X.-Y. Chen, H.-J. Ye, and D.-C. Zhan. Softs: Efficient multivariate time series forecasting with series-core fusion, 2024.
- [13] F. Harris. On the use of windows for harmonic analysis with the discrete fourier transform. *Proceedings of the IEEE*, 66(1):51–83, 1978. doi: 10.1109/PROC.1978.10837.
- [14] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015. URL <https://arxiv.org/abs/1502.03167>.
- [15] T. Kim, J. Kim, Y. Tae, C. Park, J.-H. Choi, and J. Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*, 2021.
- [16] G. Lai, W.-C. Chang, Y. Yang, and H. Liu. Modeling long- and short-term temporal patterns with deep neural networks, 2018.
- [17] B. Li, W. Cui, L. Zhang, C. Zhu, W. Wang, I. W. Tsang, and J. T. Zhou. Diffformer: Multi-resolutional differencing transformer with dynamic ranging for time series analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(11): 13586–13598, 2023. doi: 10.1109/TPAMI.2023.3293516.
- [18] J. Li, T. V. Nguyen, C. Hegde, and R. K. W. Wong. Implicit sparse regularization: The impact of depth and early stopping, 2021. URL <https://arxiv.org/abs/2108.05574>.
- [19] Z. Li, R. Cai, Z. Yang, H. Huang, G. Chen, Y. Shen, Z. Chen, X. Song, Z. Hao, and K. Zhang. When and how: Learning identifiable latent states for nonstationary time series forecasting, 2024.
- [20] B. Lim, S. O. Arik, N. Loeff, and T. Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting, 2020.
- [21] M. Liu, A. Zeng, M. Chen, Z. Xu, Q. Lai, L. Ma, and Q. Xu. Scinet: Time series modeling and forecasting with sample convolution and interaction. *Advances in Neural Information Processing Systems*, 35:5816–5828, 2022.
- [22] Y. Liu, T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma, and M. Long. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625*, 2023.
- [23] Y. Liu, C. Li, J. Wang, and M. Long. Koopa: Learning non-stationary time series dynamics with koopman predictors, 2023.
- [24] Y. Liu, H. Wu, J. Wang, and M. Long. Non-stationary transformers: Exploring the stationarity in time series forecasting, 2023.
- [25] Y. Liu, H. Zhang, C. Li, X. Huang, J. Wang, and M. Long. Timer: Transformers for time series analysis at scale, 2024.
- [26] D. Luo and X. Wang. ModernTCN: A modern pure convolution structure for general time series analysis. In *The Twelfth International Conference on Learning Representations*, 2024.
- [27] G. Michau, G. Frusque, and O. Fink. Fully learnable deep wavelet transform for unsupervised monitoring of high-frequency time series. *Proceedings of the National Academy of Sciences*, 119(8), Feb. 2022. ISSN 1091-6490. doi: 10.1073/pnas.2106598119. URL <http://dx.doi.org/10.1073/pnas.2106598119>.
- [28] R. Mukherji, M. Schöne, K. K. Nazeer, C. Mayr, D. Kappel, and A. Subramoney. Weight sparsity complements activity sparsity in neuromorphic language models, 2024.
- [29] Z. Ni, H. Yu, S. Liu, J. Li, and W. Lin. Basisformer: Attention-based time series forecasting with learnable and interpretable basis, 2024.
- [30] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*, 2022.
- [31] B. N. Oreshkin, D. Carpo, N. Chapados, and Y. Bengio. N-beats: Neural basis expansion analysis for interpretable time series forecasting, 2020.
- [32] S. Qi, Z. Xu, Y. Li, L. Wen, Q. Wen, Q. Wang, and Y. Qi. Pdatetime: Rethinking long-term multivariate time series forecasting from the perspective of partial differential equations, 2024.
- [33] L. F. Richardson and W. F. Eddy. Algorithm 991: The 2d tree sliding window discrete fourier transform. *ACM Transactions on Mathematical Software*, 45(1): 1–12, Feb. 2019. ISSN 1557-7295. doi: 10.1145/3264426. URL <http://dx.doi.org/10.1145/3264426>.
- [34] R. Stasiński. Dct computation using real-valued dft algorithms. In *2002 11th European Signal Processing Conference*, pages 1–4, 2002.
- [35] W. Toner and L. Darlow. An analysis of linear time series forecasting models, 2024.
- [36] A. Trindade. ElectricityLoadDiagrams20112014. UCI Machine Learning Repository, 2015. DOI:<https://doi.org/10.24432/C58C86>.
- [37] H. Wang, J. Peng, F. Huang, J. Wang, J. Chen, and Y. Xiao. Micn: Multi-scale local and global context modeling for long-term series forecasting. In *The Eleventh International Conference on Learning Representations*, 2022.
- [38] H. Wang, L. Pan, Z. Chen, D. Yang, S. Zhang, Y. Yang, X. Liu, H. Li, and D. Tao. Fredf: Learning to forecast in frequency domain. *arXiv preprint arXiv:2402.02399*, 2024.
- [39] B. Woodworth, S. Gunasekar, J. D. Lee, E. Moroshko, P. Savarese, I. Golan, D. Soudry, and N. Srebro. Kernel and rich regimes in overparametrized models, 2020. URL <https://arxiv.org/abs/2002.09277>.
- [40] H. Wu, J. Xu, J. Wang, and M. Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, 34:22419–22430, 2021.
- [41] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *The eleventh international conference on learning representations*, 2022.
- [42] K. Xu, M. Qin, F. Sun, Y. Wang, Y.-K. Chen, and F. Ren. Learning in the frequency domain. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1740–1749, 2020.
- [43] Z. Xu, A. Zeng, and Q. Xu. Fits: Modeling time series with 10k parameters. *arXiv preprint arXiv:2307.03756*, 2023.
- [44] W. Xue, T. Zhou, Q. Wen, J. Gao, B. Ding, and R. Jin. Card: Channel aligned robust blend transformer for time series forecasting, 2024.
- [45] H. Ye, J. Chen, S. Gong, F. Jiang, T. Zhang, J. Chen, and X. Gao. Atfnet: Adaptive time-frequency ensembled network for long-term time series forecasting, 2024.
- [46] J. Ye, W. Zhang, K. Yi, Y. Yu, Z. Li, J. Li, and F. Tsung. A survey of time series foundation models: Generalizing time series representation with large language model, 2024.
- [47] K. Yi, Q. Zhang, W. Fan, S. Wang, P. Wang, H. He, D. Lian, N. An, L. Cao, and Z. Niu. Frequency-domain mlps are more effective learners in time series forecasting, 2023.
- [48] A. Zeng, M. Chen, L. Zhang, and Q. Xu. Are transformers effective for time series forecasting?, 2022.
- [49] Y. Zhang and J. Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The eleventh international conference on learning representations*, 2022.
- [50] P. Zhao, Y. Yang, and Q.-C. He. High-dimensional linear regression via implicit regularization. *Biometrika*, 109(4):1033–1046, Feb. 2022. ISSN 1464-3510. doi: 10.1093/biomet/asac010. URL <http://dx.doi.org/10.1093/biomet/asac010>.
- [51] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Virtual Conference*, volume 35, pages 11106–11115. AAAI Press, 2021.
- [52] M. Zhou and R. Ge. Implicit regularization leads to benign overfitting for sparse linear regression, 2023. URL <https://arxiv.org/abs/2302.00257>.
- [53] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International conference on machine learning*, pages 27268–27286. PMLR, 2022.

A Visualization of results

Figure 4 visualizes the output of FTMixer on six real benchmarks. FTMixer demonstrates exceptional accuracy, producing predictions highly consistent with ground truth.

B Impact of Window Size on WFTC Performance

In this section, we analyze the impact of window size on the ETTh1 dataset, with input lengths of 96, 192, 336, 720 and an output length of 96. The results are reported in Table 8. We find that the proposed FTMixer performs the best with a window size of 24.

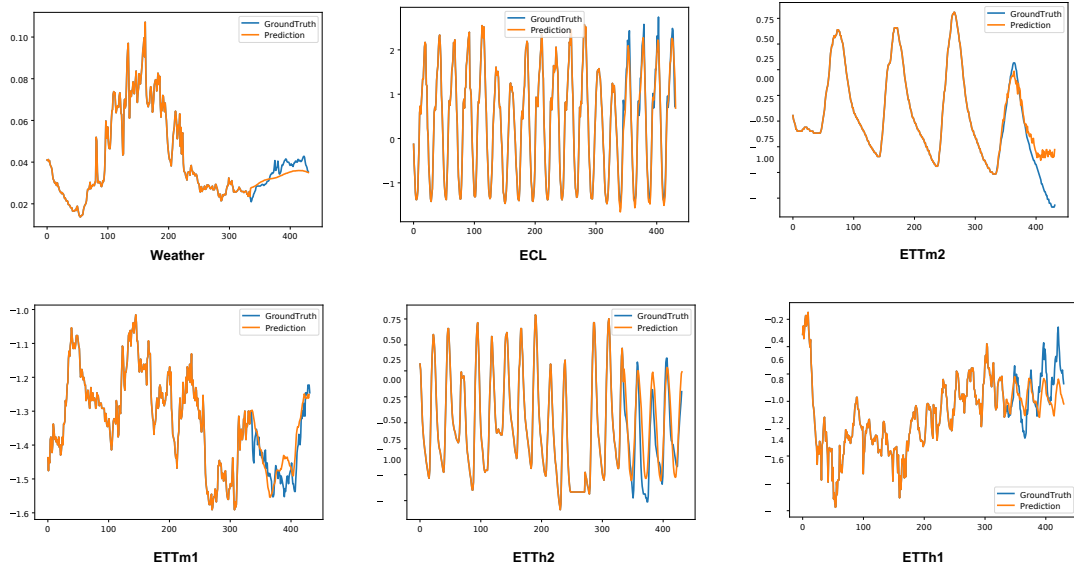


Figure 4: Visualization of Meta-Tuner forecasting on six dataset.

Table 8: The performance of the proposed FTMixer under diverse WFTC window size.

Input length \ Window Size						
	12		24		48	
	MSE	MAE	MSE	MAE	MSE	MAE
96	0.373	0.391	0.368	0.390	0.370	0.391
192	0.364	0.390	0.360	0.389	0.363	0.390
336	0.360	0.389	0.356	0.388	0.359	0.389
720	0.357	0.388	0.355	0.387	0.356	0.388