MPCformer: Multi-scale Patch Transformer and Channel Cross for Long-term Multivariate Time Series Forecasting

Journal:	*Transactions on Pattern Analysis and Machine Intelligence*
Manuscript ID	TPAMI-2024-03-0517
Manuscript Type:	Regular
Keywords:	Multi-Scale Patch, Channel Independent, Channels Cross, Time Series Forecasting

MPCformer: Multi-scale Patch Transformer and Channel Cross for Long-term Multivariate Time Series Forecasting

Linlin Yang, Ming Gao, Ying Jin, Jixiang Yu, Weiyue Li, Meiling He, Jiafu Tang, and Zhiguo Zhu

Abstract—**Multivariate time series (MTS) data has become increasingly popular across various industries due to the advancements in big data technology. Accurate multivariate predictions for distant future points in MTS data are crucial for time series forecasting. However, despite significant advancements in Transformer time series models and other methodologies, effectively modeling and incorporating both temporal and channel dependencies inherent in MTS data remains a challenge. Previous versions of Transformer time series models relied heavily on task-specific designs and preconceived 'pattern bias', thereby revealing limitations in capturing prevalent time series features such as seasonality and periodicity. To address these challenges, we introduce MPCformer, a new time series forecasting methodology that utilizes the Multi-Layer Perceptron (MLP) and the Multi-scale Patch Transformer and Channel Cross (MPTC) architecture to extract relevant features capturing both trend and periodic behavior within the time series data. Specifically, the MPTC is designed with multi-scale patch inputs, and different modules are sequentially designed to learn the dependency information of long series and the correlation between channel variables to achieve the complete modeling of MTS data. Extensive experiments on diverse real-world datasets demonstrate that MPCformer outperforms state-of-the-art (SOTA) prediction methods, showcasing significant improvements in accuracy.**

This work was supported by the Natural Science Foundation of China under Grants 72293563 and 72172025, in part by the Dalian Scientific and Technological Talents Innovation Support Plan under Grant 2022RG17, and in part by the Basic Scientific Research Project of Liaoning Provincial Department of Education under Grant JYTZD2023050. (Corresponding author: Ming Gao) All authors contribute equally to this paper.

Linlin Yang, Weiyue Li and Meiling He are with the School of Management Science and Engineering, Key Laboratory of Big Data Management Optimization and Decision of Liaoning Province, Dongbei University of Finance of Economics, Dalian, 116025, China. (e-mail: 2022100248@stumail.dufe.edu.cn, dufe_phd_stu@stumail.dufe.edu.cn, mei0_he@163.com).

Ming Gao is with the School of Management Science and Engineering, Key Laboratory of Big Data Management Optimization and Decision of Liaoning Province, Dongbei University of Finance of Economics, Dalian, 116025, China, and also with the Center for Post-doctoral Studies of Computer Science, Northeastern University, Shenyang, 110819, China (e-mail: gm@dufe.edu.cn).

Ying Jin is with the School of Management, Huazhong University of Science and Technology, Wuhan, 430074, China (e-mail: jinying.isec@gmail.com).

Jixiang Yu is with the Department of Computer Science, City University of Hong Kong, Hong Kong SAR(e-mail: jixiang.yu@my.cityu.edu.hk).

Jiafu Tang and Zhiguo Zhu is with the School of Management Science and Engineering, Key Laboratory of Big Data Management Optimization and Decision of Liaoning Province, Dongbei University of Finance of Economics, Dalian, 116025, China (e-mail: tangjiafu@dufe.edu.cn, zhuzg0628@126.com).

Index Terms—**Index Terms—Multi-Scale Patch, Channel Independent, Channels Cross, Time Series Forecasting**

I. INTRODUCTION

TIME series analysis is a critical field in both industry and academia, with forecasting being a vital component. Its applications span across numerous industries. Traditional time series forecasting methods usually rely on explicit statistical patterns/indicators, including well-known methods like linear autoregressive models [1] or state space models [2]. However, in real-world time series data, it often violates the assumptions of these methods. With the continuous development and maturity of big data technologies, where Multivariate Time Series (MTS) data is common, multivariate forecasting over longer time horizons is both practical and challenging, playing a crucial role in various fields such as energy management [3], economic and financial forecasting [4], traffic flow prediction [5], and automated business processing [6].

The continuous innovation of deep learning techniques in recent years has greatly contributed to the development of the field of time series forecasting by researchers, creating the application of deep learning-based knowledge to the field. This includes RNN/LSTM [7], CNN [8], [9], [10], Graph Neural Networks (GNN) [11], [12], WaveNet [13], Temporal Convolutional Networks (TCNs) [14], among others. In particular, transformer-based architectures have shown great potential in long-term multivariate time series forecasting (MTSF) tasks, such as FEDformer [15], Autoformer [16], Informer [17], LogTrans [18], etc.

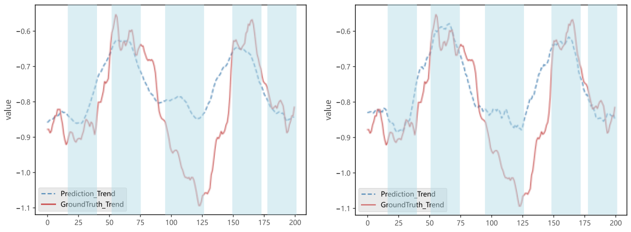
Although Transformer-based architectures have become mainstream and state-of-the-art for Multivariate Time Series Forecasting (MTSF) in recent years, these studies have mainly focused on mitigating standard quadratic complexity in time and space, such as attention mechanisms [17] or structural changes [16]. Most of the existing transformer-based methods directly embed data from multiple different channels, and then input them into a model and forecast future information for all channels and targets simultaneously. This method of processing data may destroy the time series dependency information within a single channel. This class of methods then faced a challenge from Dlinear [19], which, by introducing a channel-independent design [35] and utilizing a straightforward linear layer as the model's core, achieved prediction performance surpassing that of most previously mentioned methods. This discovery hints at a limitation of the Transformer [20] in its prior applications. However, the latest Transformer-based method, PatchTST [21], embraces the channel-independent concept similar to Dlinear, thereby elevating model performance to achieve a new SOTA. This demonstrates that the Transformer remains effective in capturing temporal dependencies.

In addition to extracting intertemporal dependencies, capturing channel correlations is crucial for MTSF. Channel correlations refer to the relationships between different variables or dimensions in the time series data. Incorporating information from related series in other dimensions can significantly enhance prediction accuracy [22]. For example, historical prices alone might not suffice for accurate forecasting when predicting future oil prices. Additional factors such as the prices of raw materials, geopolitical events, demand-supply dynamics, and the availability of alternative energy sources also play crucial roles in making reliable predictions. While PatchTST and Dlinear employ channel-independent methods to enhance the modeling of long-time dependencies, they overlook the significant interdependencies between different channels in MTSF. This oversight limits their ability to maximize forecasting performance. Models such as Crossformer [22], Client [23], and TSMixer [24], while acknowledging the correlation between different channels in MTS forecasting, struggle to effectively capture long-time dependencies. Consequently, the forecasting performance of these methods may fall short compared to PatchTST in certain datasets.

As discussed in Bengio et al [25], complex data arise from intricate interactions across multiple sources. An effective representation should decompose the various explanatory sources to enhance robustness against complex and structurally rich changes. However, most existing time series forecasting methods directly model time-lagged relationships and multivariate interactions along the observed data. As point-in-time data inherently contain unpredictable noise, this can result in capturing spurious correlations.

According to the independent mechanism hypothesis [26], [27], seasonal and trend cycle modules are assumed to operate independently of each other. This implies that if one mechanism changes due to distributional shifts, the other remains unchanged. This can be achieved by decomposing the original time series into trend cycle and seasonal cycle components, which are tailored for improved transmission or generalization in non-stationary environments. Moreover, separating the independent seasonal and trend mechanisms enables the development of specialized models for autonomous learning.

The nature of attention in Transformer is achieved by interpolating contextual history rather than extrapolating linear trends [28]. The masking experiments in Client reveal that the Transformer structure generally exhibits poor generalization of trend information in time-series data and does not effectively learn trend features. It was also verified in DifFormer [29] that seasonality was not explicitly preserved during the encoding of the self-attention mechanism.



(a) (b)

Fig.1. Comparison of trend series of predicted results and ground truth for ETTm1 data under different conditions. Fig.1(a) shows a trend plot of the MPCformer's predicted and the ground truth in the ETTm1 dataset with the Trend Decomposition module removed (entered directly into the MPTC module). Fig.1(b) shows a trend plot of predicted results and the ground truth for the MPCformer ETTm1 dataset.

Figure. 1 shows the trend comparison between the predicted values and the ground truth with and without sequence decomposition. From the comparison of the shaded portion of Fig. 1(a), it can be seen that inputting the sequences into MPCformer without sequence decomposition does not adequately extract the trend information. From the comparison of the shaded portions of Fig. 1 (a) and (b), it can be seen that the trend plots of the MPCformer prediction results after sequence decomposition are more in line with the trends of the ground truth results. This shows that the trend information of the predicted time series using the non-transformer structure can be closer to the trend information of the ground truth.

Therefore, to obtain more accurate and reliable prediction results, this paper proposes a Multi-scale Patch Transformer and Channel Cross (MPCformer) to overcome the above limitations. Firstly, the MPCformer adopts the time series decomposition method to decompose the series into trend cycle and seasonal cycle parts and proposes the MLP and MPTC modules to extract the potential features of the trend cycle and seasonal cycle series. In particular, to fully exploit the feature information under different temporal granularities in the time series, the multi-scale Patch method is proposed as an input in MPTC. Finally, in MPTC we first adopt the channel-independent modeling approach to fully extract the long-time dependence of the time series, and later propose Channel Cross MLP (CCM) to learn the correlation between different channels. This balances the channel-independent approach and the channel-cross approach to achieve a more comprehensive modeling approach for both time and variable dimensions. The MPCformer model achieves state-of-the-art accuracy in eight real-world datasets. The contributions are summarized as follows:

- In this paper, we adopt the time series decomposition method and propose a customized MLP module and MPTC module for the decomposed trend cycle series and seasonal cycle series, respectively, to extract the trend cycle and seasonal cycle feature fully.

- In the MPTC module, multi-scale patch are proposed as inputs to fully explore the potential information of different time granularities in the time series.
- The channel-independent and then channel-fusion methods are used to extract the sequence dependence and the correlation between different channels of the multivariate long time series in depth, respectively, to improve the modeling of the multivariate time series.
- Our empirical study shows that MPCformer has outperformed current SOTA models. MPCformer achieves an average improvement of 18.1% in MSE for multivariate time-series forecasting compared to the Transformer method and the Non-Transformer method.

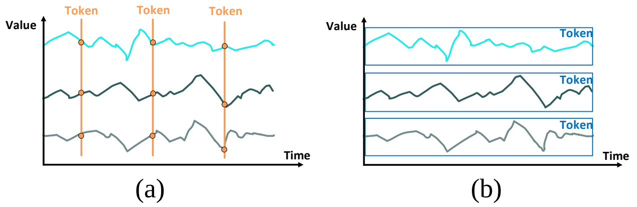
II. Related Work

A. Time Series Forecasting

Existing time series forecasting methods encompass two primary categories: classical statistical methods, deep learning-based methods, and hybrid models combining elements of both. Among them, classical statistical methods mainly include the Autoregressive Model [32], Exponential Smoothing [2], Auto-Regressive Moving Average Model (ARIMA) [1], etc. While classical statistical methods offer good interpretability, their reliance on stringent assumptions about data structure, such as smoothness and seasonality, can pose limitations. Many real-world time-series datasets are multi-dimensional and non-smooth, posing challenges for classical models to accurately capture their dynamics. Consequently, these models may not fully meet the prediction needs of existing data.

The deep learning correlation models proposed in recent years have demonstrated superior capability in capturing underlying features and multidimensional correlations within real-time-series data. While basic models such as Convolutional Neural Networks (CNN) [8] and Recurrent Neural Networks (RNN) [32] have been applied in time-series prediction, RNN methods hold a significant advantage in extracting long-term dependencies from sequences. However, they often encounter issues such as vanishing or exploding gradients, hindering their training stability and performance. On the other hand, CNN methods often face limitations related to their receptive fields, hindering their effectiveness in modeling long-term dependencies. These challenges may restrict the widespread adoption and extension of RNN and CNN methods and their variants in time-series forecasting.

Later, the proposal of the Transformer further boosted the field of temporal prediction, addressing the limitations of applying the Transformer [20] in modeling temporal dependencies. Researchers subsequently proposed several variants of the Transformer model to enhance the modeling of dependencies within long sequences, such as Informer [17], Autoformer [16], FEDformer [15], and others. These models aimed to further enhance the modeling of dependencies within long sequences and to address channel cross modeling.



(a) (b)

Fig. 2. Comparison of two input strategies for the MTSF task. Fig. 2(a) shows the strategy of the Traditional Multivariate Long Time Series Forecasting method, where all channels are taken as inputs and the predicted future values depend on the history of all channels. Fig. 2(b) shows the Channel Independent strategy, which treats multivariate series as multiple univariate series and trains a unified model on these series. The prediction of each channel depends only on its historical values and ignores the relationships between different channels.

However, these models mainly focused on reducing the complexity of modeling cross-time dependencies and couldn't effectively address long sequence dependencies and channel cross-modeling. Recent models like DLinear and PatchTST utilize channel-independent methodologies to significantly reduce interference among uncorrelated channels, thereby improving the predictive capabilities of these models. However, this approach overlooks channel correlation, which is essential for MTSF.

B. Traditional Multivariate Long Time Series Forecasting

MTS data are often characterized by complex relationships between different channels. The traditional multivariate long-time series forecasting method uses the values of the same time points in all channels as inputs and thus gets the prediction, as shown in Fig. 2(a). LogTrans [18] proposes convolutional self-attention and LogSparse with these models to improve the ability to capture local information and reduce spatial complexity. Informer [17] introduced ProbSparse self-attention alongside a distillation technique, significantly reducing the time complexity for processing extensive input sequences. Additionally, it incorporates a generative decoder capable of producing long sequences in a single step. Autoformer [16] enhances prediction performance by iteratively decomposing sequences and introducing an autocorrelation mechanism that leverages the periodic nature of the data. FEDformer [15] employs a framework akin to Autoformer, substituting the autocorrelation attention mechanism with a Fourier augmentation structure to achieve linear complexity. The Multi-scale Isometric Convolution Network (MICN) [33] is designed to forecast both the trend cycle and seasonal components separately. It accomplishes this by employing multiple branches of convolutional kernels tailored for capturing distinct pattern information within the seasonal segment, thereby facilitating comprehensive local and global modeling.

The upper method usually involves mixing data from multiple channels as input, and directly outputting future data from all channels simultaneously. This approach may compromise the time series dependency information within a single channel, and to some extent impair the modeling of long time series.

C. Channel Independent Time Series Forecasting

Recently, research has highlighted the potential of channel separation in improving the modeling of long dependencies in sequences. DLinear [19], the first to employ a channel-independent approach as illustrated in Fig. 2(b), utilizes a simple linear layer as its main module, providing experimental evidence of the limitations of the previous Traditional Multivariate Long Time Series Forecasting approach. After DLinear, PatchTST [21] also adopts a channel-independent design. It segments time series data into patches, serving as inputs to the Transformer Encoder, thereby bolstering the model's capability to capture long-range dependencies and enhancing overall performance. This further validates the efficacy of the Transformer model in the realm of time series forecasting.

The predictive performance of this class of methods often surpasses that of many traditional multivariate long-time series forecasting methods by leveraging the channel-independent approach to extract sequence dependencies, demonstrating the efficacy of this method in modeling long sequences' dependencies. This approach effectively mitigates the influence of other channels when modeling dependencies in long-time series. However, this method solely relies on the channel-independent approach, neglecting correlations among multiple channels in multivariate long-time series and consequently constraining the model's forecasting performance.

D. Both Channel- Cross and Cross-Time Time Series Forecasting

Several studies have explored both channel dependence and time dependence approaches to compensate for the shortcomings of the above two types of approaches. As shown in Table I, the columns titled Tradition refer to methods in which all channels at a point in time are used as inputs. Columns titled Channels-Cross refer to methods that focus on modeling cross-covariate correlation, and columns titled Cross-Time refer to methods that utilize only temporal correlation and assume independence between covariates. Crossformer [22] obtains a two-dimensional vector array by applying dimensional segmented embedding (DSW) processing to the time series and utilizes the two-stage attention (TSA) layer to capture the inter-temporal and inter-channel dependencies respectively, further modeling the sequence from both time and channel dimensions. Google proposed TSMixer [24], which is designed by stacking MLP, employing different Time Mixing and Feature Mixing techniques to sequentially extract time dependencies and channel correlations from the time and channel dimensions, respectively. Finally, the aggregation and prediction of features are accomplished through the Temporal Projection module. And research demonstrates that inter-channel interactions, significantly impact prediction performance. The iTransformer [34] uses a channel-independent approach to learn the correlation between different channels through self-attention by feeding individual channels as tokens into an inverted transformer structure and then learns the global dependence of the sequences for time series forecasting by using layer normalization and feed-forward network modules.

TABLE I
CLASSIFICATION OF MULTIVARIATE TIME-SERIES FORECASTING MODELS.

Category	Tradition	Channels -Cross	Cross-Time	Models
I	✓			LogTrans Informer Autoformer FEDformer MICN
II			✓	DLinear PatchTST
III		✓	✓	Crossformer TSMixer iTransformer MPCformer(ours)

These methods achieve a more comprehensive modeling approach than traditional multivariate long time series forecasting methods and channel-independent time series forecasting methods [30], and thus the prediction performance of this class of models is supposed to be higher than that of the above two classes. However, the prediction indexes of most of these models are lower than that of PatchTST, which indicates that although most of these methods have achieved a more comprehensive modeling approach, they still have problems such as not fully reflecting the time dependence of long time series or not fully learning the correlation between channels, leading to lower prediction indexes compared to PatchTST. For instance, while these methods may capture some aspects of time dependence in long-time series data, they might overlook subtle temporal patterns or fail to incorporate them effectively into the forecasting process. Similarly, although they aim to model correlations between different channels, they may not capture all nuances, resulting in suboptimal performance.

III.METHODOLOGY

A. Problem Definition

This section defines the problem. The multivariate long-time series forecasting problem aims to predict the future values of multiple target series at multiple time steps. Given a multivariate time series $X_{1:L}=\{x_1,x_2,...,x_L\}$ of length L, where $x_t\in\mathbb{R}^{d_x}$, where d_x is the dimension of the input time series variable and $d_x >= 1$. where the goal of the task is to predict future values of multiple dimensions as $\hat{Y}_{L+1:L+T}=\{\hat{y}_{L+1},\hat{y}_{L+2},...,\hat{y}_{L+T}\}$, where the predicted value is $\hat{y}_t\in\mathbb{R}^{d_y}$, is the length of the prediction and d_y is the dimension of the predicted value. The problem can then be defined as

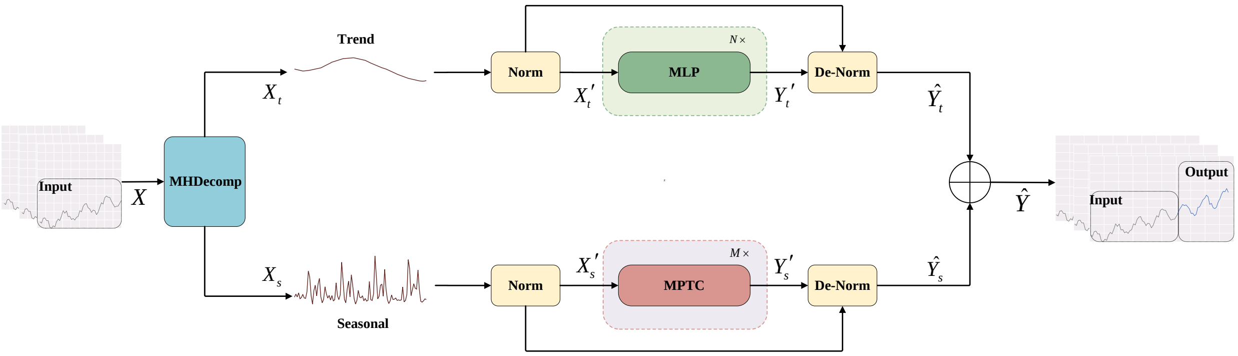


Fig. 3. The overall framework of the MPCformer model.

$$\hat{Y}_{L+1:L+T} = F\left(X_{1:L} : \Omega\right). \tag{1}$$

Where $\hat{Y}_{L+1:L+\varepsilon}$ is the prediction result, F represents the prediction model, and Ω represents the parameters required to propose the model.

B. Model Overview

As shown in Fig. 3, the overall architecture of MPCformer is similar to the MICN. However, the trend cycle learning module and seasonal learning module are completely different from MICN. MPCformer first adopts the Multi-scale Hybrid Decomposition (MHDecomp) block to separate the complex patterns of the input sequences, i.e., to decompose the original time series into two parts: the trend cycle and the seasonal part. The decomposed trend period and seasonal period sequences are fed into the Reversible Instance Normalization (RevIN) [35] module (including Norm and De-Norm) to remove the non-stationary information in the two sequences and then fed into the MLP and MPTC modules to learn the potential features of the two sequences. Finally, the output sequences are obtained by summing the outputs of the two modules. The technical details of these modules are described in detail in the following subsections.

C. Multi-scale Hybrid Decomposition Module

Moving averages are used in Autoformer to smooth out periodic fluctuations to highlight long-term trends. For the input sequence $X \in \mathbb{R}^{d_x}$, the process is

$$\begin{aligned} X_t &= AvgPool(Padding(X))_{kernel}, \\ X_s &= X - X_t. \end{aligned} \tag{2}$$

where $X_t, X_s \in \mathbb{R}^{d_x}$ is the trend-period component and the seasonal component, respectively. The length of the sequence can be kept constant by using the padding operations. However, when $AvgPod(\bullet)$ is set to a different parameter, we will get trend and seasonal series with large differences in volatility. Therefore, this decomposition module does not provide a better separation of trend and period series.

The Mixture Of Experts Decomposition block (MOEDecomp) module in FEDformer uses several different averaging filters to extract different trend patterns. Adaptive weights are then used to combine the extracted patterns into a decomposition sequence. However since the weights of each pattern cannot be determined before characterizing the individual different trend and cycle patterns, the trends and cycles under different parameters that cannot be extracted are not well output. Where MOEDecomp for the input sequence $X \in \mathbb{R}^{d_x}$, the process of decomposing the sequence is

$$\begin{aligned} X_t &= \sigma(w(X)) * (AvgPool(Padding(X))_{kernel_1} \\ &\quad \cdots \\ &\quad AvgPool(Padding(X)_{kermel_n}), \\ X_s &= X - X_t. \end{aligned} \tag{3}$$

Where $X_t, X_s \in \mathbb{R}^{d_x}$ are the trend-period component and the seasonal component, respectively. And $\sigma()$ denotes the softmax operation and $\sigma(w(X))$ denotes the weights of the mixed trend series.

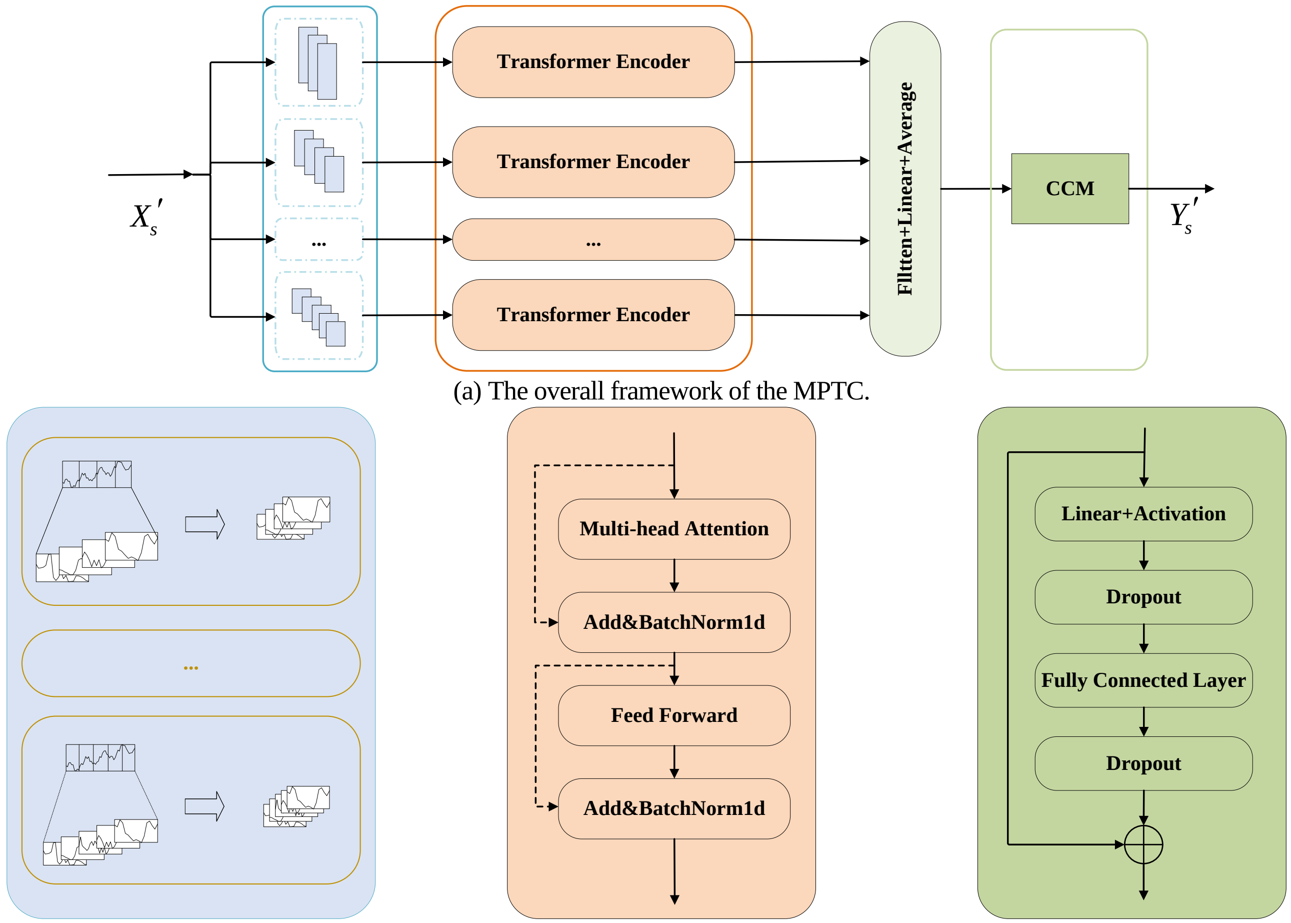
Therefore, the MPCformer in this paper uses the Multi-scale Hybrid Decomposition (MHDecomp) decomposition module. The decomposition module is similar to MOEDecomp in that it also uses multiple $AvgPool(\bullet)$ parameters to separate multiple different trends and cycle patterns. Unlike MOEDecomp, however, this module uses averages to combine multiple trends and cycles directly to ensure that the multiscale information in the seasonal cycle is consistent across different parameter settings. For the input sequence $X \in \mathbb{R}^{d_x}$, the process is

$$\begin{aligned} X_t &= mean(AvgPool(Padding(X))_{kernel_1} \\ &\quad AvgPool(Padding(X))_{kernel_2} \\ &\quad \cdots \\ &\quad AvgPool(Padding(X)_{kermel_n}), \\ X_s &= X - X_t. \end{aligned} \tag{4}$$

Where $X_t, X_s \in \mathbb{R}^{d_x}$ are the trend-period component and the seasonal component, respectively.

D. Reversible Instance Normalization

As time series forecasting models are often affected by the distribution bias problem, it can lead to inconsistency in the distribution of training data and test data of the forecasting model, which in turn seriously affects the prediction performance of the model. Therefore, several models such as TMixer, PatchTST, MICN, and so on use the RevIN module to remove the influence of the distribution bias problem. However, the application of the RevIN module in these models does not well counteract the effect



(a) The overall framework of the MPTC.

(b) Schematic diagram of a multi-scale Patch.

(c) Architecture diagram of Transformer Encoder.

(d) Architecture diagram of CCM.

Fig. 4. Detailed diagrams of the MPTC and the various parts in it.

of distributional bias on the time series prediction problem.

Therefore, this paper further explores the application of this module and finds that further decomposition of the time series, in which each sub-sequence separately adopts the RevIN module to eliminate the non-stationary information can further improve the role of the module to offset the distributional bias, which in turn improves the prediction performance of the model.

The method adopts a symmetric structure to first normalize the input series and then back-normalize the output series. Where normalization is the process of converting each data distribution to a mean-centered distribution, which in turn overlaps the training and test data distributions. The process of normalization is:

$$\begin{aligned}&\mathbb{E}_t[X]=\frac{1}{T_x}\sum_{j=1}^{T_x}X,\\&Var[X]=\frac{1}{T_x}\sum_{j=1}^{T_x}\left(X-\mathbb{E}_t[X]\right)^2,\\&\hat{X}_R=\gamma\left(\frac{X-\mathbb{E}_t[X]}{\sqrt{Var[X]+\epsilon}}\right)+\beta.\end{aligned}\tag{5}$$

Where T_x is the length of the input sequence, $\mathbb{E}_t[X]$ is the mean of the input sequence, and $Var[X]$ is the standard deviation of the input sequence. $\hat{X}_R$ is the normalized sequence, $\gamma,\beta\in\mathbb{R}^{d_x}$ is the learned affine parameter vector.

Using the normalized data as input to the model preserves the aligned training and test data distributions in the predicted output of the model. Then the role of the inverse normalization is to convert the data after the predicted output of the model to the original distribution. The process of inverse normalization is:

$$\hat{Y}_R=\sqrt{Var[X]+\epsilon}\cdot\left(\frac{\tilde{Y}_R-\beta}{\gamma}\right)+\mathbb{E}_t[X].\tag{6}$$

Where $\tilde{Y}_R$ is the model's prediction of the normalized sequence and $\hat{Y}_R$ is the result after back-normalizing $\tilde{Y}_R$.

E. Trend period learning module

Extracting the components of the time series can help us understand the underlying process and further improve the forecasting accuracy. There have been several models that use time series decomposition, such as Autoformer, FEDformer, and many other models that splice the trend part obtained from the decomposition of the original time series, and then accumulate it, and then obtain the trend feature information, but they have not proved their effectiveness or explanation for this. Meanwhile, most of the methods [23], [29] verify that the model of attention cannot effectively extract the trend information of the time series, instead the relatively simple Linear and MLP can effectively extract the trend information [23], [33].

Therefore, in this paper, the trend signal is modeled separately for the underlying features of the time series data. Considering that in the field of time series forecasting, there is always the problem of distributional bias between training data and forecasting data, and similarly, this situation also exists in the trend and cycle after decomposition[31]. Therefore, in this paper, the Reversible Instance Normalization (RevIN) module is also used for trend-periodic sequences to effectively remove and recover the non-smooth information [35], [36] that mainly exists in the trend, to eliminate further the distributional bias in the underlying time series features and improve the effectiveness of RevIN. In this paper, after exploration, it is found that the use of a customized MLP module can more effectively learn the underlying features of the trend sequence with the following formula:

$$\hat{Y}_t = RevIN(MLP(RevIN(X_t))). \qquad (7)$$

Where $\hat{Y}_t$ is the trend cycle result extracted by the trend cycle learning module.

F. Multi-scale Patch Transformer and Channel Cross

The Multi-scale Patch Transformer and Channel Cross (MPTC) consists of four parts containing a Multi-scale Patch, a Transformer Encoder, a Fusion Module, and a Channel Cross MLP (CCM) as shown in Fig. 4(a). MPTC first extracts different-scale Patch and inputs them into the channel-independent Transformer Encoder separately to extract potential long-term dependencies. The Fusion Module aggregates the multi-scale information into embedding, and then CCM extracts the correlation between different channels.

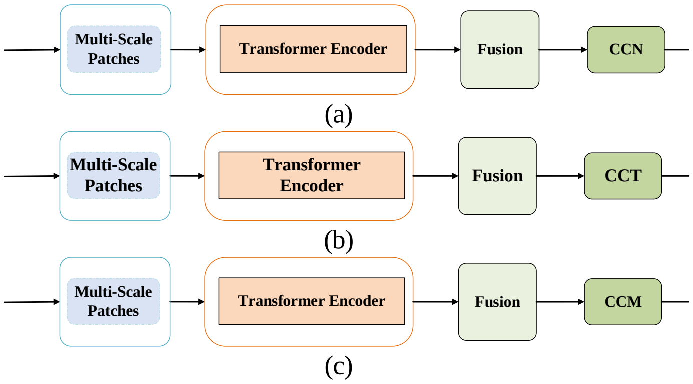
1) Multi-scale Patch

In this paper, we use the method of sliding window to divide the decomposed seasonal cycle series into Patch, which means that the original position information of the time series is preserved and the original values are not changed. Denote the Patch length as L_p and the step between two consecutive Patches as S_P, i.e., a sliding window approach is used to unfold and expand the input sequence X_s to generate the Patch sequence $x_j^{\,i} \in \mathbb{R}^{j\times N}$, where N is the number of Patch blocks divided, and where $N_p = \dfrac{(L-L_p)}{S_p}+2$. The modeling of multiple features at different time scales helps to learn the expressive representation of time series [37], [38], so this paper uses the generation of multi-scale Patch sequences as input to extract multiscale information to enhance the modeling of time series information further. Patch blocks containing different scale information can be generated when different $L_p^{\,i}$ and $S_p^{\,j}$ are set for X_s respectively. In this paper, a parallel approach is used to generate Patch sequences $P_n \in \mathbb{R}^{d_x}$ containing different scale information by applying the above method to the input sequences at the same time, as shown in Fig. 4(b). where the process of generating multi-scale Patch sequences is:

$$\begin{aligned}
&P_1 = \mathrm{Unfold}\left(\mathrm{ReplicationPad}(X_s), size = L_p^{\,1}, step = S_p^{\,1}\right),\\
&P_2 = \mathrm{Unfold}\left(\mathrm{ReplicationPad}(X_s), size = L_p^{\,2}, step = S_p^{\,2}\right),\\
&\qquad\qquad\ldots\\
&P_n = \mathrm{Unfold}\left(\mathrm{ReplicationPad}(X_s), size = L_p^{\,i}, step = S_p^{\,j}\right).
\end{aligned} \qquad (8)$$

2) Transformer Encoder

The different scale Patch sequences that have been output in the above section are fed into the Transformer Encoder module in a parallel manner. After that, the time-seasonal cycle information features in each channel are extracted in a channel-independent manner. The module uses a customized Vanilla Transformer Encoder structure, and it has been verified in several methods such as PatchTST, TSMixer, etc. that BatchNorm [39] Layers are more effective as a regularization term for temporal data compared to LayerNorm Layers. Therefore, in this paper, BatchNorm Layers are also used in this module to regularize the time, as shown in Fig. 4(c). In this paper, we use the data processing method of dividing the time series into Patch blocks, so the input to this Encoder is also a single-time Patch block in the Patch sequence. Therefore the inputs in the multi-head attention are also Patch-tokens. where the attention operates as follows:



(a)

(b)

(c)

Fig. 5. Exploration of Channel Cross Modules

$$\mathrm{Atention}(Q,K,V) = \mathrm{Sofmax}\left(\frac{QK^{\mathrm{T}}}{\sqrt{d}}\right)V. \qquad (9)$$

3) Fusion Module

This module mainly focuses on the fusion of multiple parallel branches extracted from different scales into a complete integrated time-seasonal cycle information embedding. This module focuses on the transformation of the learned feature information at each scale K_n into the same dimension T_n by using flattening, i.e., $O_n = Flatten(K_n)$ as well as linear transformations to achieve. Then the embedding of each of the same dimensions is directly applied to the mean $H_n = mean(T_n)$ to fuse multiple different scales of information, and the use of averaging can be effective in retaining information that is effectively retained at each scale. The whole process of fusion is shown below:

$$\begin{aligned}
H_n = mean(&Linear(Flatten(K_1)), Linear(Flatten(K_2)),\\
&\ldots, Linear(Flatten(K_n)).
\end{aligned} \qquad (10)$$

4) Channel Cross MLP

In this paper, Channel Cross MLP (CCM) is proposed as shown in Fig.4(d). Since the multiscale feature information of the time dimension of a single channel is only extracted in Encoder, the correlation between different channels is ignored in this module. Thus, CCM is further proposed to learn the correlation between individual channels interactively and model the multivariate time series data more comprehensively.

This paper explores several different approaches to multichannel fusion modules, as follows

a) Transformer-based Channel Cross module (CCT) as shown in Fig. 5(a).
b) Convolution-based Channel Cross Convolution module (CCN) as shown in Fig. 5(b).
c) MLP-based Channel Cross MLP (CCM), as shown in Fig. 5(c).

After exploring the above three different method modules, it is found that the MLP-based channel attention module can achieve the maximum extraction of correlation between different channels (see ablation experiments for the results). The CCM module first embeds the multiscale fusion into the information transformation dimension to transform H_n from the time dimension to the channel dimension. The process in which the extraction of seasonal cycle channel correlation is realized is shown below:

$$Y_s' = H_n + Dropout(FC(Dropout(Relu(Linear(H_n{}^T))))). \quad (11)$$

IV. Experiments

A. Datasets

We will evaluate MPCformer on the following 8 popular datasets in different domains as shown in Table II

The ETT dataset is a record of load and oil temperature characteristics of different power transformers from July 2016 to July 2018. The data consists of the target value "Oil Temperature" and 6 power load characteristics with a total of 7 variables. The time granularity of ETTh1 and ETTh2 is 1 hour. The time granularity of ETTm1 and ETTm2 is 15 minutes.

The Illness dataset records weekly influenza illness patient data from 2002 through 2021 for seven variables.

The Weather dataset records 10-minute climate data for 21 variables at nearly 1,600 locations from 2010 through mid-2013.

The Electricity dataset records the amount of electricity consumed per hour by approximately 321 consumers between 2012 and 2014, for a total of 321 variables.

The Traffic dataset records hourly roadway occupancy measured by different sensors on the freeway, with a total of 862 variables.

B. Baselines and Experimental Settings

Several models are selected to compare with MPCformer in multivariate time-series forecasting, including three transformer-based models: iTransformer [34], PatchTST [21], Crossformer [22], FEDformer [15], and Autorformer [16]. To be fair, we also include some non-transformer time-series models including MICN [33], TSMixer [24], and DLinear [19].

All experiments in this paper follow the same experimental steps, similar to previous related studies, and for the ILI dataset the prediction lengths are respectively $T \in \{24, 36, 48, 60\}$, The prediction length for the other datasets is $T \in \{96, 192, 336, 720\}$. For the Illness data, the lookback window is L=104, and for the other data, the lookback window is L=512. The baseline method selects the optimal parameters for the open-source of the method. Due to the large number of channels in the Traffic dataset, the batch size of both PatchTST and MPCformer is set to 8. The parameters for the different scales in MHDecomp are $[17, 49]$. The optimal multi-scale parameters are selected for each dataset and the model is modeled using the ADAM optimizer. The computing environment is shown in Table III.

TABLE II
The Statistics of The eight Popular Datasets For The LTSF Problem

Datasets	ETTh1& ETTh2	ETTh1& ETTh2	Illness	Weather	Electricity	Traffic
Variates	7	7	7	21	321	862
Timesteps	17,420	69,680	996	52,696	26,304	17,544
Granularity	1hour	5min	1week	10min	1hour	1hour

TABLE III
Computing Environment

Language/Environment	Version/Type
GPU	NVIDIA RTX6000(48G)
CPU	Intel(R) Xeon(R) Platinum 8375C CPU @ 2.90GHz
RAM	1024 GB
CUDA	12.1
Python	3.9
Pytorch	2.1

Metrics: Following previous studies in the same field, mean-absolute-error (MAE) and mean-square-error (MSE) were selected as evaluation metrics.

C. Main Results

The multivariate time-series forecasting results are given in Table IV. We use prediction lengths $T \in \{24, 36, 48, 60\}$ for the ILI dataset and $T \in \{96, 192, 336, 720\}$ for the others. The best results are in bold and the second best are underlined. Overall, MPCformer achieves SOTA performance across all prediction step settings for the eight datasets, outperforming all baseline methods. On average, MPCformer achieves a 9.9% improvement in MSE and a 5.1% improvement in MAE compared to the best Transformer method, PatchTST. On average, MPCformer achieves a 26.19% MSE improvement and an 8.5% MAE improvement compared to the best Non-Transformer method, TSMixer. Notably, for Dlinear, which questions the effectiveness of Transformer, MPCformer achieves 29.3% MSE improvement and a 16.1% MAE improvement. In terms of dataset size, for small noisy datasets such as the ILI dataset, MPCformer achieves an 18.9% improvement in MSE and a 13.4% improvement in MAE compared to PatchTST, demonstrating that MPCformer is

TABLE IV
MULTIVARIATE LONG-TERM FORECASTING RESULTS WITH MPCFORMER.

Models		MPCformer (our)		iTransformer (2023)		PatchTST (2023)		TSMixer (2023)		MICN (2023)		Crossformer (2023)		DLinear (2023)		FEDformer (2022)		Autoformer (2021)	
Metrics		MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
ETTh1	96	**0.359**	**0.392**	0.387	0.405	0.370	0.400	0.366	0.396	0.418	0.429	0.418	0.443	0.392	0.418	0.375	0.415	0.475	0.469
	192	**0.392**	**0.415**	0.441	0.436	0.412	0.429	0.401	0.419	0.447	0.463	0.410	0.442	0.410	0.419	0.416	0.441	0.483	0.473
	336	**0.405**	**0.420**	0.491	0.462	0.421	0.439	0.420	0.434	0.558	0.548	0.433	0.459	0.518	0.507	0.444	0.457	0.527	0.500
	720	**0.439**	**0.453**	0.509	0.494	0.447	0.468	0.449	0.467	0.625	0.597	0.514	0.518	0.492	0.504	0.474	0.492	0.510	0.512
ETTh2	96	0.277	0.343	0.301	0.350	**0.273**	**0.337**	0.274	0.339	0.297	0.362	0.404	0.442	0.290	0.353	0.339	0.381	0.386	0.416
	192	**0.330**	0.385	0.380	0.399	0.339	**0.380**	0.333	**0.380**	0.444	0.456	0.458	0.488	0.394	0.419	0.424	0.435	0.447	0.447
	336	**0.327**	0.386	0.424	0.432	0.329	**0.384**	0.366	0.408	0.515	0.498	0.514	0.524	0.443	0.458	0.445	0.461	0.478	0.482
	720	0.397	0.435	0.430	0.447	**0.379**	**0.422**	0.416	0.449	0.896	0.690	0.694	0.634	0.680	0.583	0.448	0.477	0.453	0.473
ETTm1	96	**0.284**	**0.344**	0.342	0.377	0.290	0.343	0.299	0.349	0.315	0.363	0.311	0.361	0.304	0.351	0.351	0.401	0.484	0.461
	192	**0.327**	0.371	0.383	0.396	0.333	**0.370**	0.342	0.374	0.372	0.400	0.354	0.396	0.343	0.376	0.389	0.421	0.565	0.508
	336	**0.353**	**0.390**	0.418	0.418	0.369	0.392	0.377	0.393	0.386	0.416	0.398	0.418	0.371	0.387	0.428	0.449	0.533	0.496
	720	**0.388**	**0.415**	0.487	0.457	0.416	0.420	0.428	0.421	0.448	0.457	0.519	0.496	0.425	0.421	0.476	0.475	0.598	0.537
ETTm2	96	**0.163**	**0.254**	0.186	0.272	0.165	0.256	0.166	0.256	0.178	0.272	0.397	0.446	0.166	0.258	0.188	0.280	0.232	0.312
	192	**0.218**	**0.294**	0.254	0.314	0.223	0.296	0.222	**0.294**	0.236	0.311	0.457	0.491	0.225	0.302	0.255	0.323	0.301	0.353
	336	**0.269**	**0.328**	0.316	0.351	0.273	0.329	0.278	0.332	0.300	0.351	0.527	0.535	0.284	0.345	0.323	0.363	0.359	0.384
	720	**0.343**	**0.383**	0.414	0.407	0.361	0.385	0.377	0.393	0.432	0.450	1.008	0.762	0.396	0.413	0.418	0.421	0.418	0.415
Illness	24	**1.195**	**0.667**	2.055	0.916	1.513	0.817	2.212	0.964	2.685	1.113	3.197	1.185	2.272	1.057	3.209	1.240	3.420	1.299
	36	**1.060**	**0.682**	2.006	0.925	1.482	0.844	2.216	0.946	2.668	1.068	3.646	1.265	2.349	1.089	2.581	1.048	3.372	1.251
	48	**1.471**	**0.794**	1.846	0.945	1.651	0.849	2.805	0.936	2.563	1.054	3.798	1.269	2.303	1.078	2.547	1.058	3.502	1.299
	60	**1.307**	**0.784**	2.043	0.976	1.561	0.869	2.140	0.963	2.747	1.110	3.838	1.285	2.443	1.146	2.778	1.132	2.827	1.143
Weather	96	**0.141**	**0.195**	0.176	0.216	0.148	0.198	0.146	0.200	0.169	0.236	1.793	0.788	0.175	0.235	0.237	0.322	0.280	0.348
	192	**0.186**	**0.240**	0.225	0.257	0.192	**0.240**	0.191	0.242	0.225	0.288	2.660	0.986	0.217	0.275	0.265	0.323	0.296	0.354
	336	**0.237**	**0.281**	0.281	0.299	0.245	0.282	0.243	0.281	0.275	0.334	3.638	1.214	0.262	0.312	0.318	0.354	0.378	0.410
	720	**0.308**	0.336	0.357	0.349	0.314	**0.333**	0.343	0.349	0.343	0.387	4.433	1.411	0.325	0.364	0.401	0.410	0.427	0.437
Electricity	96	**0.126**	**0.221**	0.148	0.239	0.128	0.222	0.131	0.229	0.162	0.269	0.204	0.291	0.140	0.238	0.188	0.304	0.206	0.321
	192	**0.143**	**0.239**	0.167	0.258	0.148	0.243	0.152	0.247	0.188	0.296	0.250	0.324	0.154	0.251	0.197	0.311	0.222	0.333
	336	**0.156**	**0.253**	0.177	0.269	0.161	0.258	0.163	0.261	0.190	0.301	0.327	0.364	0.169	0.268	0.208	0.323	0.271	0.369
	720	**0.183**	**0.281**	0.210	0.299	0.198	0.292	0.188	0.284	0.213	0.324	0.386	0.421	0.204	0.301	0.243	0.352	0.286	0.381
Traffic	96	**0.353**	**0.246**	0.392	0.268	0.397	0.285	0.380	0.265	0.517	0.307	0.541	0.302	0.412	0.286	0.575	0.357	0.697	0.448
	192	**0.377**	**0.257**	0.413	0.277	0.410	0.291	0.401	0.279	0.532	0.319	0.531	0.300	0.424	0.291	0.606	0.373	0.618	0.387
	336	**0.394**	**0.272**	0.426	0.284	0.419	0.296	0.415	0.289	0.535	0.315	0.597	0.342	0.438	0.298	0.621	0.380	0.610	0.376
	720	**0.435**	**0.291**	0.459	0.300	0.457	0.316	0.448	0.312	0.592	0.323	0.554	0.301	0.467	0.318	0.631	0.382	0.691	0.430
#Rank1st(total=32)		**30**	**26**	0	0	2	7	0	2	0	0	0	0	0	0	0	0	0	0
Avg		**0.417**	**0.376**	0.548	0.422	0.463	0.396	0.565	0.411	0.671	0.472	1.194	0.622	0.590	0.448	0.681	0.480	0.792	0.526

TABLE V
COMPARISON OF THE RESULTS OF MPCFORMER REMOVING DIFFERENT SCALES OF PATCH IN DIFFERENT DATASETS.

Models		MPCformer (our)		MPCformer (w/o Patch1)		MPCformer (w/o Patch2)	
Metrics		MAE	MSE	MAE	MSE	MAE	MSE
ETTh1	96	**0.359**	**0.392**	0.378	0.409	0.371	0.403
	192	**0.392**	**0.415**	0.417	0.434	0.410	0.429
	336	**0.405**	**0.420**	0.422	0.441	0.426	0.441
	720	**0.439**	**0.453**	0.456	0.472	0.460	0.474
ETTm1	96	**0.284**	**0.344**	0.288	0.344	0.291	0.347
	192	**0.327**	**0.371**	0.334	0.373	0.336	0.374
	336	**0.353**	**0.390**	0.379	0.404	0.369	0.390
	720	**0.388**	**0.415**	0.415	0.429	0.431	0.430
weather	96	**0.141**	**0.195**	0.146	0.199	0.145	0.198
	192	**0.186**	**0.240**	0.189	0.240	0.241	0.280
	336	**0.237**	**0.281**	0.240	0.281	0.282	0.310
	720	**0.308**	0.336	0.310	**0.333**	0.311	**0.333**
Electricity	96	**0.126**	**0.221**	0.127	0.222	0.131	0.228
	192	**0.143**	**0.239**	0.146	0.241	0.145	0.241
	336	**0.156**	**0.253**	0.161	0.258	0.161	0.259
	720	**0.183**	**0.281**	0.198	0.292	0.201	0.292
Traffic	96	**0.353**	**0.246**	0.363	0.255	0.362	0.254
	192	**0.377**	**0.257**	0.386	0.266	0.381	0.263
	336	**0.394**	**0.272**	0.400	0.274	0.414	0.294
	720	**0.435**	**0.291**	0.441	0.294	0.450	0.315

more robust in handling noisy data. For the larger and more stable dataset, the Traffic dataset, MPCformer still achieves a 5.1% improvement in MSE and a 7.0% improvement in MAE compared to the second-best-results TSMixer, indicating that MPCformer can capture long-term dependencies in the data more consistent.

D. Ablation Study

1) Multi-scale Patch

Under the same experimental conditions described above, we perform ablation experiments in multiple datasets (ETTh1, ETTm1, Weather, and Traffic) against the inputs of two different scale Patch modules to validate the effectiveness of extracting multi-scale information. To verify the necessity of multi-scale, MPCformer was compared with the variant models with different scales removed, respectively, as shown in Table V. Where MPCformer (w/o Patch1) denotes the MPCformer model with the first scale input module removed and MPCformer (w/o Patch2) denotes the MPCformer model with the second scale input module removed.

From Table V, it can be found that deleting any of the scale Patch modules leads to performance degradation. The prediction performance of MPCformer is best only when both different scales of information are used as inputs, verifying that the multiscale model is capable of extracting potential information at different temporal granularities.

2) Varying Look-back Window

According to the theory of statistical methods, more accurate prediction results can be obtained by utilizing more historical information. However, it was verified in the previous DLinear that the prediction performance of most Transformer methods does not necessarily improve with the increase of the look-back window, verifying that most methods are flawed in extracting temporal information. In this paper, we further improve the modeling and methodology based on the previous research, as shown in Fig. 6, it can be found that the MSE of MPCformer in multiple datasets with different output lengths is reduced with the increase of the lookback window. It proves that MPCformer can obtain effective potential information from longer lookback windows.

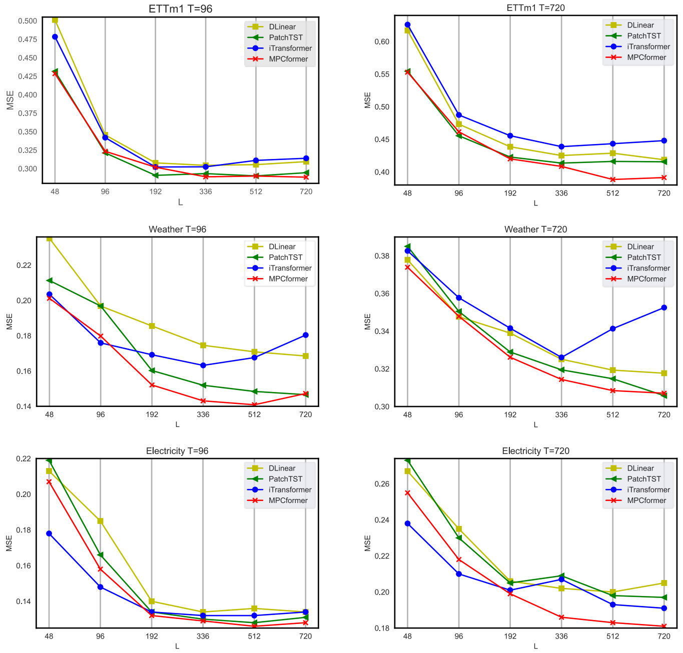


Fig. 6. For the datasets ETTm1, weather, and Electricity datasets with a different number of channels, this paper sets the lookback window to $T \in \{48, 96, 192, 336, 512, 720\}$ and compares the MSE of MPCformer with iTransformer, PatchTST, and DLinear.

3) Analysis: RevIN

This paper explores the application of RevIN in MPCformer. As shown in Table VI, MPCFormer denotes the original model proposed in this paper, in which RevIN modules are added to both trend and cycle, respectively. MPCFormer (a) denotes: applying the RevIN module only before series decomposition, MPCFormer (b) denotes: applying the RevIN module only in trend, MPCFormer (c) means: applying the RevIN module only in the seasonal cycle, MPCFormer (d) means: all RevIN modules are removed from MPCformer. As shown in Table VI, it can be found that the MSE of MPCformer is significantly lower than the other four cases. As shown in Fig. 7, in the same dataset ETTh1, the distribution of the predicted values of MPCformer that applies RevIN to the trend series and seasonal cycle series respectively is closer to the distribution of the ground truth values compared to the others, in contrast to other methods that apply RevIN only before the decomposition or only to the trend or cycle. In this paper, it is found that applying the RevIN module to trend and seasonal cycle series separately optimizes the model's forecasting performance across multiple data sets. Such an application minimizes the effects of distributional bias in the underlying series and thus outperforms the other four cases.

4) Channel Cross MLP

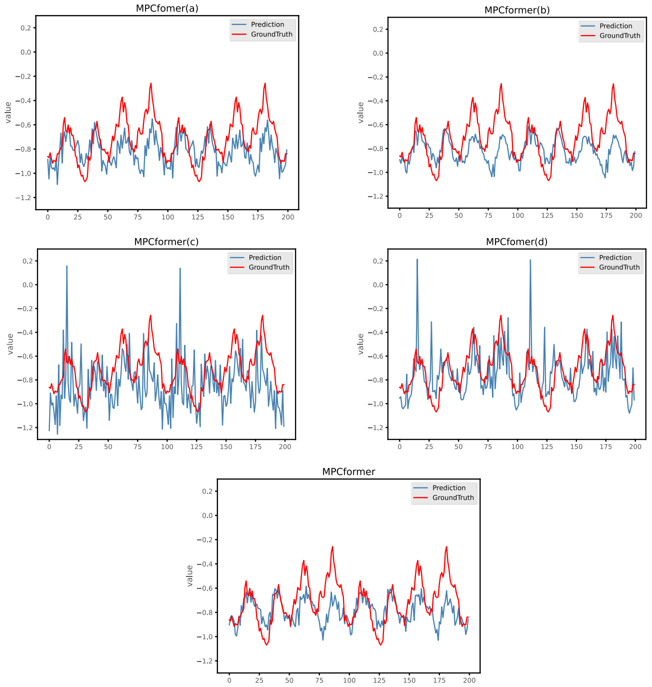


Fig. 7 Plot of predicted versus ground truth values for five different cases of RevIN in MPCformer in the ETTh1 dataset.

The experimental results of CCM ablation in MPCformer are given in Table Ⅶ. Where MPCformer$^{-\text{CCM}}$ indicates that the CCM module in the MPCformer model is removed and the other modules and parameters of the model are kept unchanged. Comparing the results with those of MPCformer, it is clear that removing the CCM module from the MPCformer model will result in a serious degradation of the model's prediction performance across multiple datasets. This shows that it is necessary to utilize the CCM module to extract the channel correlation in the timing data, and also proves the effectiveness of the CCM module.

TABLE Ⅵ
DIFFERENT REVIN APPLICATIONS IN MPCFORMER

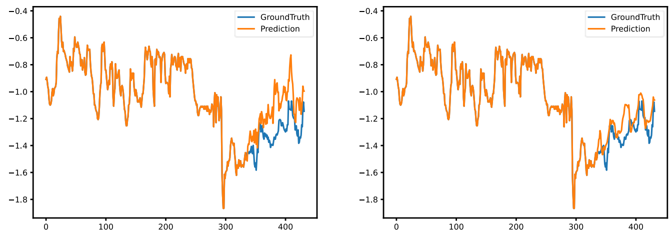
Models		MPCformer (our)		MPCformer (a)		MPCformer (b)		MPCformer (c)		MPCformer (d)	
Metrics		MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
ETTh1	96	**0.359**	**0.392**	0.368	0.399	0.364	0.393	0.541	0.507	0.397	0.422
	192	**0.392**	**0.415**	0.404	0.422	0.397	0.414	0.713	0.595	0.433	0.442
	336	**0.405**	**0.420**	0.416	0.426	0.415	0.422	0.538	0.515	0.462	0.467
	720	**0.439**	**0.453**	0.465	0.473	0.454	0.457	0.832	0.707	0.656	0.599
ETTm2	96	**0.284**	**0.344**	0.290	0.347	0.304	0.352	0.285	0.345	0.301	0.354
	192	**0.327**	**0.371**	0.331	0.373	0.341	0.376	0.330	0.373	0.349	0.384
	336	**0.353**	**0.390**	0.361	0.395	0.362	0.391	0.362	0.397	0.366	0.399
	720	**0.387**	**0.414**	0.410	0.423	0.407	0.418	0.404	0.423	0.406	0.426
Illness	24	**1.195**	**0.667**	1.310	0.714	1.842	0.848	2.344	1.109	2.492	1.011
	36	**1.060**	**0.682**	1.315	0.787	1.960	0.950	1.787	0.941	2.625	1.123
	48	**1.471**	**0.794**	1.523	0.823	2.318	0.983	2.042	0.952	2.856	1.132
	60	1.307	0.784	**1.248**	**0.769**	1.876	0.934	2.439	1.127	3.054	1.224
Electricity	96	**0.126**	**0.221**	**0.126**	**0.221**	0.127	0.224	0.128	0.224	0.128	0.225
	192	**0.143**	**0.239**	0.145	0.241	0.145	0.244	0.145	0.242	0.147	0.245
	336	**0.156**	**0.253**	0.161	0.259	0.165	0.265	0.161	0.262	0.162	0.263
	720	**0.183**	**0.281**	0.202	0.294	0.200	0.298	0.193	0.292	0.193	0.295

Meanwhile, to further validate the effectiveness of the CCM, the proposed CCM is migrated to models that only use channel-independent methods such as PatchTST and Dlinear to help these models achieve complete modeling and learn the correlation between different variables and potential features. As shown in Table Ⅶ below, where PatchTST(64)$^{+\text{CCM}}$ and DLinear$^{+\text{CCM}}$ indicate that the CCM module is migrated to PatchTST and DLinear, respectively, and the others remain unchanged. Comparing the results with those of PatchTST and DLinear under the same conditions, it can be found that the prediction performance of the model is improved to different degrees after adding the CCM module to these methods. As shown in Fig. 8, in the ETTh1 dataset, the curves of the DLinear$^{+\text{CCM}}$ predicted values are closer to the ground truth compared to the DLinear predicted value curves, which proves that the channel correlations extracted by the CCM module contribute to the model prediction performance and verifies the generality of the CCM module. This reveals the shortcomings of such models in modeling multivariate time series and extracting features of different channels, which are compensated by the addition of the CCM module, which proves the necessity of modeling and extracting features between different channels when forecasting multivariate time series.

In Table Ⅷ, it can be found that the MSE of PatchTST(64)$^{+\text{CCM}}$ has relatively little improvement over PatchTST, while on the contrary, the MSE of DLinear$^{+\text{CCM}}$ has relatively higher improvement in predictive performance over the original model DLinear. After exploration, it is found that the DLinear model also uses the sequence decomposition method and migrates the CCM to a similar position in the DLinear method and the MPCformer model, and thus DLinear$^{+\text{CCM}}$ obtains a huge improvement in prediction performance relative to DLinear. When the CCM is migrated to other locations in the DLinear method, the predictive performance of DLinear$^{+\text{CCM}}$ is not sufficiently improved. The structure of the MPCformer model and the validity of the CCM are verified from the side.

This paper explores customized CCM, CCN, and CCT as modules related to extraction channels as shown in Fig. 5. However, after several experimental findings, it can be found that it is the CCM that achieves the best prediction performance in several datasets, as shown in Fig. 9. Therefore, in this paper, the CCM module is identified as part of the MPCformer approach.

From the above, we can conclude that these components are essential and contribute significantly to our model.



(a)DLinear (b) DLinear$^{+\text{CCM}}$

Fig. 8. DLinear and DLinear$^{+\text{CCM}}$ and predicted results vs the ground truth in ETTh1.

TABLE Ⅶ
COMPARISON OF ABLATION EXPERIMENT RESULTS BETWEEN MPCFORMER AND MPFCORMER^{-CCM} IN DIFFERENT DATASETS.

Models		MPCformer		MPCformer^{-CCM}	
Metrics		MAE	MSE	MAE	MSE
ETTh1	96	**0.359**	**0.392**	0.365	0.395
	192	**0.392**	**0.415**	0.404	0.420
	336	**0.405**	**0.420**	0.415	0.427
	720	**0.439**	**0.453**	0.442	0.457
ETTh2	96	**0.277**	**0.343**	0.286	0.350
	192	**0.330**	**0.385**	0.337	0.388
	336	**0.327**	**0.386**	0.334	0.396
	720	**0.397**	**0.435**	0.409	0.447
Illness	24	**1.195**	**0.667**	1.576	0.827
	36	**1.060**	**0.682**	1.447	0.808
	48	**1.471**	**0.794**	1.802	0.872
	60	**1.307**	**0.784**	1.556	0.823
Weather	96	**0.141**	**0.195**	0.150	0.201
	192	**0.186**	**0.240**	0.243	0.284
	336	**0.237**	**0.281**	0.286	0.315
	720	**0.308**	**0.336**	0.313	0.337
Traffic	96	**0.353**	**0.246**	0.404	0.285
	192	**0.377**	**0.257**	0.409	0.286
	336	**0.394**	**0.272**	0.446	0.307
	720	**0.435**	**0.291**	0.365	0.395

TABLE Ⅷ
COMPARISON OF ABLATION EXPERIMENT RESULTS OF CCM MODULES IN PATCHTST AND DLINEAR METHODS IN MULTIPLE

Models		PatchTST		PatchTST^{+CCM}		DLinear		DLinear^{+CCM}	
Metrics		MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
ETTh1	96	0.370	0.400	**0.365**	**0.398**	0.392	0.418	**0.367**	**0.391**
	192	0.412	0.429	**0.404**	**0.424**	0.410	0.419	**0.404**	**0.417**
	336	0.421	0.439	**0.419**	**0.436**	0.518	0.507	**0.434**	**0.438**
	720	0.447	0.468	**0.449**	**0.466**	0.492	0.504	**0.479**	**0.496**
ETTm2	96	0.165	0.256	**0.164**	**0.256**	0.166	0.258	**0.166**	**0.256**
	192	0.223	0.296	**0.218**	**0.295**	0.225	0.302	**0.224**	**0.301**
	336	0.273	0.329	**0.272**	**0.329**	0.284	0.345	**0.275**	**0.345**
	720	0.361	0.385	**0.359**	**0.386**	0.396	0.413	**0.385**	**0.411**
Weather	96	0.148	0.198	**0.144**	**0.193**	0.175	0.235	**0.164**	**0.227**
	192	0.192	0.240	**0.189**	**0.239**	0.217	0.275	**0.207**	**0.267**
	336	0.245	0.282	**0.241**	**0.280**	0.262	0.312	**0.248**	**0.301**
	720	0.314	0.333	**0.307**	**0.329**	0.325	0.364	**0.295**	**0.341**
Electricity	96	**0.128**	**0.222**	0.129	0.222	0.140	0.238	**0.136**	**0.234**
	192	0.148	0.243	**0.143**	**0.238**	0.154	0.251	**0.152**	**0.253**
	336	0.161	0.258	**0.157**	**0.254**	0.169	0.268	**0.166**	**0.266**
	720	0.198	0.292	**0.192**	**0.280**	0.204	0.301	**0.196**	**0.297**

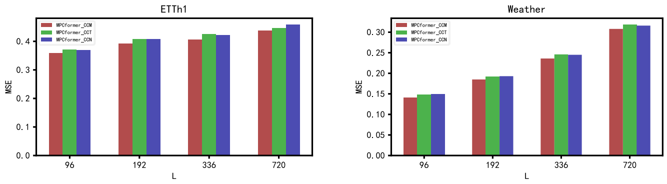


Fig. 9. Comparison of results of MPCformer application of different channel crossover modules in different datasets.

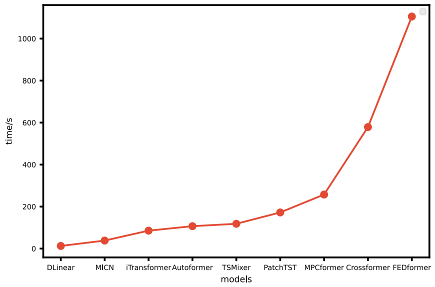


Fig. 10. Comparison plot of time consumed by Baseline and MPCformer methods for the ETTh1 dataset with a prediction length of 96.

TABLE IX
COMPUTATIONAL CONSUMPTION OF ALL METHODS IN THE ETTH1 DATASET WITH OUTPUT LENGTH 96.

Model	Training	Parameter GPU
	(s/epoch)	(M)
DLinear	1.404	346
PatchTST(64)	1.457	1018
iTransformer	3.364	522
MPCformer(our)	4.213	2914
MICN	6.668	1206
TSMixer	7	11113
Autoformer	13.972	2510
Crossformer	15.817	1716
FEDformer	49.356	1888

E. Computational Cost

Table IX represents the computational consumption of the MPCformer proposed in this paper and as a baseline method under the given conditions. Figure. 10 shows the overall time required to train and test the baseline and MPCformer under the same conditions. Although MPCformer is in the middle of the pack in terms of GPU memory required and time consumed for all baseline methods and consumes slightly more, it achieves a significant performance improvement at a less dramatic, i.e., more than 30%, computational cost. Overall MPCformer is acceptable in terms of GPU memory and training time.

F. Diebold-Mariano Forecast Evaluation Test

In Table IV, it can be found that although the MSE and MAE metrics of MPCformer outperform the other baseline datasets in most of the datasets, the performance enhancement of MPCformer over PatchTST is small in a few datasets. Therefore, the Diebold-Mariano Forecast Evaluation Test (MD) [40] is used in the ETTm1 and Electricity datasets to verify whether the forecasting performance of MPCformer is significantly better than PatchTST.

In the MD test, there is the original hypothesis that there is no significant difference in the performance of the two models in terms of prediction error, and the alternative hypothesis that at least one of the models outperforms the other in terms of prediction.

In the ETTm1 dataset, for the prediction results of MPCformer and PatchTST with a prediction length of 96, there are $DM = -3.746$ and $P = 0.00051$ when MSE is used as the loss function in MD. The original hypothesis is rejected due to

$P < 0.005$ and $DM < 0$, so the prediction performance of MPCformer is statistically superior to PatchTST.

In the Electricity dataset, for the prediction results of MPCformer and PatchTST with a prediction length of 96, there are $DM = -2.7308$ and $P = 0.0066$ when MSE is used as the loss function in MD. The original hypothesis is rejected due to $P < 0.005$ and $DM < 0$, so the prediction performance of MPCformer is statistically superior to PatchTST.

V. Conclusion

In this paper, we introduce the design of multi-scale Patch and MPTC, enhancing the Transformer model for long-time series prediction in MTS data by addressing challenges related to multi-scale analysis, long-time dependency, and multi-channel correlation. The results of ablation experiments conducted on various public datasets demonstrate that the design approaches proposed in this paper are instrumental in enabling MPCformer to outperform existing models, achieving SOTA performance.

In future research, we aim to explore more effective multi-scale modules for automatic parameter selection to enhance the extraction of richer multiscale information. Additionally, we plan to design a gating mechanism module capable of selecting channel variables to mitigate information redundancy between channels, thereby improving the prediction performance of the model.

References

[1] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, Time series analysis: forecasting and control. John Wiley & Sons, 2015.

[2] J. Durbin and S. J. Koopman, Time series analysis by state space methods. Oxford University Press, 2012.

[3] K. D. Orwig et al., "Recent Trends in Variable Generation Forecasting and Its Value to the Power System," in IEEE Transactions on Sustainable Energy, vol. 6, no. 3, pp. 924-933, July 2015

[4] R. Sen, H.-F. Yu, and I. Dhillon, "Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting," in Proc. Int. Conf. Neural Inf. Process. Syst., 2019, pp. 4837–4846.

[5] Kai Li, Weihua Bai, Shaowei Huang, Guanru Tan, Teng Zhou, Keqin Li, "Lag‐related noise shrinkage stacked LSTM network for short‐term traffic flow forecasting", IET Intelligent Transport Systems, vol.18, no.2, pp.244, 2024.

[6] D.Simchi-Levi, P. Kaminsky, E. Simchi-Levi, and R. Shankar, Designing and Managing the Supply Chain: Concepts, Strategies and Case Studies. New York, NY, USA: McGraw-Hill, 2008.

[7] G. Lai, W.-C.Chang, Y.Yang, and H.Liu, "Modeling long- and short-term temporal patterns with deep neural networks," in Proc. Int. ACM SIGIR Conf. Res. Develop. Informat. Retrieval, 2018, pp. 95–104.

[8] V. Radu et al., "Multimodal deep learning for activity and context recognition," in Proc. ACM Interactive Mobile Wearable Ubiquitous Technol., vol. 1, no. 4, pp. 1–27, 2018.

[9] S. Ding, Z. Chen, T. Zheng, and J. Luo, "RF-Net: A unified meta-learning framework for RF-enabled one-shot human activity recognition," in Proc. ACMConf. Embedded Netw. Sensor Syst., 2020, pp. 517–530.

[10] S. Li, R. R. Chowdhury, J. Shang, R. K. Gupta, and D. Hong, "Units: Short-time fourier inspired neural networks for sensory time series classification," in Proc. ACM Conf. Embedded Netw. Sensor Syst., 2021, pp. 234–247.

[11] Z. Pan, Y. Liang, W. Wang, Y. Yu, Y. Zheng, and J. Zhang, "Urban traffic prediction from spatio-temporal data using deep meta learning," in Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining, 2019, pp. 1720–1730.

[12]B.Yu, H.Yin, and Z. Zhu, "Spatiotemporal graph convolutional networks: A deep learning framework for traffic forecasting,"inProc.Int.JointConf. Artif. Intell., 2018, pp. 3634–3640.

[13] A. v. d.Oord, S.Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, andK. Kavukcuoglu, "Wavenet: A generative model for raw audio," arXiv preprint arXiv:1609.03499,2016.

[14] S. Bai, J. Z. Kolter, andV. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling,"arXivpreprintarXiv:1803.01271,2018.

[15] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, ‘‘FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting,’’ in Proc. Mach. Learn. Res.,2022, pp. 27268-27286.

[16] J. Xu et al., "Autoformer: Decomposition transformers with autocorrelation for long-term series forecasting," in Proc. Int. Conf. Neural Inf. Process. Syst., 2021, pp. 22419–22430.

[17] H. Zhou et al., "Informer: Beyond efficient transformer for long sequence time-series forecasting," in Proc. Conf. Assoc. Advance. Artif.Intell,2021, pp. 11106–11115.

[18] S. Li et al., "Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting," in Proc. Int. Conf. Neural Inf. Process. Syst., 2019, pp. 5243–5253.

[19] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting ?," in Proc. Conf. Assoc. Advance. Artif. Intell., 2023, pp. 11121–11128.

[20] A. Vaswani et al., "Attention is all you need," in Proc. Int. Conf. Neural Inf. Process. Syst., 2017, pp. 5998–6008.

[21] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A time series is worth 64 words: Long-term forecasting with transformers," in Proc. Int. Conf. Learn. Representations, 2023.

[22] Y. Zhang and J. Yan, ‘‘Crossformer: Transformer utilizing cross dimension dependency for multivariate time series forecasting,’’ in Proc.11thInt. Conf. Learn. Represent.,2022. [Online]. Available: https://openreview.net/forum?id=vSVLM2j9eie

[23] J. Gao, W. Hu, and Y. Chen, "Client: Cross-variable linear integrated enhanced trans former for multivariate long-term time series forecasting," 2022, arXiv:2305.18838

[24] S.-A. Chen, C.-L. Li, N. Yoder, S. O. Arik, and T. Pfister, "TSMixer: An all-mlp architecture for time series forecasting," 2023, arXiv:2303.06053.

[25] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," IEEE Trans. Pattern Anal. Mach. Intell., vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[26] Y.Bengio, A.Courville, and P.Vincent, "Representation learning: A review and new perspectives," 2012, arXiv:1206.5538

[27] Giambattista Parascandolo, Niki Kilbertus, Mateo Rojas-Carulla, and Bernhard Sch¨ olkopf. Learning independent causal mechanisms. in Proc. Mach. Learn. Res.,2018, pp. 4036 4044.

[28] X. Zhang et al., "First De-Trend then Attend: Rethinking Attention for Time-Series Forecasting, "2022. arXiv:2212.08151.

[29] B. Li et al., "DifFormer: Multi-Resolutional Differencing Transformer With Dynamic Ranging for Time Series Analysis," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 45, no. 11, pp. 13586-13598, 1 Nov. 2023.

[30] J. Deng, X. Chen, R. Jiang, X. Song, and I. W. Tsang, "A Multi-View Multi-Task Learning Framework for Multi-Variate Time Series Forecasting," in IEEE Transactions on Knowledge and Data Engineering, vol. 35, no. 8, pp. 7665-7680, 1 Aug. 2023.

[31]V. Le Guen and N. Thome, "Deep Time Series Forecasting With Shape and Temporal Criteria," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 45, no. 1, pp. 342-355, 1 Jan. 2023.

[32]D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," Nature, vol. 323, pp. 533-536, 1986.

[33] H. Wang, J. Peng, F. Huang, J. Wang, J. Chen, and Y. Xiao, "MICN: Multi-scale local and global context modeling for long-term series forecasting," in The Eleventh International Conference on Learning Representations, 2023. [Online]. Available: https://openreview.net/forum?id=zt53IDUR1U

[34] Y. Liu et al., ‘‘iTransformer: Inverted transformers are effective for time series forecasting,"2023, arXiv:2310.06625

[35] T. Kim, J. Kim, Y. Tae, C. Park, J.-H. Choi, and J. Choo, "Reversible instance normalization for accurate time-series forecasting against distribution shift," in International Conference on Learning Representations, 2022. [Online]. Available: https://openreview.net/ forum?id=cGDAkQo1C0p

[36] T. Zhou, et al., "Film: Frequency improved Legendre memory model for long-term time series forecasting." in Proc. Int. Conf. Neural Inf. Process. Syst., 2022, pp.12677-12690.

[37] M. A. Shabani, A. H. Abdi, L. Meng, and T. Sylvain, “Scaleformer: Iterative multi-scale refining transformers for time series forecasting,” in The Eleventh International Conference on Learning Representations, 2023. [Online]. Available: https://openreview.net/forum?id=sCrnllCtjoE

[38] M. Ferreira, D. Higdon, H. Lee, and M. West. “multi-scale and hidden resolution time series models,” Bayesian Analysis, (2006): 947-967.

[39] G. Zerveas, S.Jayaraman, D. Patel, A. Bhamidipaty, and C. hoff. “A transformer-based framework for multivariate time series representation learning,” in Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining, 2021, pp. 2114–2124.

[40]F. X. Diebold and R. S. Mariano, "Comparing predictive accuracy", J. Bus. Econ. Statist., vol. 20, no. 1, pp. 134-144, 2002.

Linlin Yang received his B.S. degree from the School of Mathematics and Information Sciences, Henan Normal University in 2021. He is currently pursuing the M.S. degree in Management Science and Engineering from Dongbei University of Finance and Economics. His research interests include machine learning-based time series forecasting, data mining, and cloud computing.

Ming Gao received the BE degree in management information system from the Department of Economic Information, Dongbei University of Finance and Economics (DUFE), Dalian, China, in 2002, the MS degree in information management from DUFE, in 2004, and the PhD degree in information technology and management from the School of Management Science and Engineering (SMSE), DUFE, in 2013. He is currently a professor with SMSE, DUFE. His research interests include business process management, deep learning, cloud computing, and big-data applications.

Ying Jin received a bachelor’s degree in management from the School of Management Science and Engineering, Dongbei University of Finance and Economics (DUFE), Dalian, China, in 2022. She is currently pursuing a master’s degree in management from the School of Management, Huazhong University of Science and Technology (HUST), Wuhan, China. Her research interests include deep learning, data mining, and big-data applications.

Jixiang Yu received his B.S. degree in Computer Science and Technology from the Dongbei University of Finance and Economics, Dalian, China in 2022. He is now pursuing the Ph.D. degree with the Department of Computer Science, City University of Hong Kong, Hong Kong SAR. His research interests mainly lie in the intersection of applied machine learning and bioinformatics.

Weiyue Li received the B.S. degree from the School of Management, Hefei University of Technology, in 2021. He is currently pursuing the doctor’s degree in electronic commerce. His research interests include recommendation systems and social network analysis.

Meiling He received her B.S. degree in management information system in 2018 from the School of Economics and Management, Changchun University of Science and Technology. She is currently pursuing the M.S. degree in Management Science and Engineering from Dongbei University of Finance and Economics. Her research interests include deep learning and their industrial applications, graph neural networks, and cloud computing.

Jiafu Tang received the PhD degree from Northeastern University, in 1999, and from 1998 to 2002 he successively conducted academic cooperative research and visits with the City University of Hong Kong and the Hong Kong Polytechnic University, Akita Prefecture University in Japan, and Pohang University of Technology in South Korea. He is currently the dean with the School of Management Science and Engineering, Dongbei University of Finance and Economics, and a distinguished professor with the “Yangtze River Scholar” of the Ministry of Education. Research fields: manufacturing system production and logistics operation management, quality system engineering, data mining, and Business intelligence research.

Zhiguo Zhu presently serves as a professor and doctor supervisor of Management Science and Engineering School at Dongbei University of Finance and Economics in China. He got his bachelor, master and doctor degree in information management system at Dalian University of Technology in 1999, 2002 and 2010, respectively. His research interests include business intelligence, user interest modeling and intelligent recommendation. He has published over 20 research papers in international journals and conferences, such as Electronic Commerce Research and Applications, Expert Systems and Applications, Computers in Human Behavior, etc.