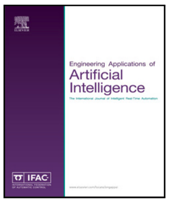
Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



Research paper

TOEformer: Temporal order enhanced Transformer for time series forecasting

Qiang Li [a,b], Jiwei Qin [a,b,*], Dacheng Wang [c], Xizhong Qin [a,b], Daishun Cui [a,b], Jiachen Xie [a,b], Dezhi Sun [a,b]

[a] *School of Computer Science and Technology, Xinjiang University, Urumqi, China*
[b] *Xinjiang Key Laboratory of Signal Detection and Processing, Xinjiang University, Urumqi, China*
[c] *Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing, China*

ARTICLE INFO

Keywords:
Time series
Forecasting
Transformer
Loss function

ABSTRACT

Transformers have shown excellent performance in time series forecasting, but a recent study points out that existing Transformer-based time series forecasting models cannot preserve temporal order well. To address this problem, we preserve temporal order information by integrating linear and Transformer structures and predicting trend and seasonal data, respectively, which compensates for the inadequacy of the Transformer in this regard by leveraging linear models to preserve the temporal order information of the original sequence. Besides, directly combining the seasonal and trend part prediction results increases the overall prediction error. For this problem, this paper proposes a new loss function, the relative value exponential loss, which reduces the impact of the problem on predictions by constraining the network to generate more seasonal-trend pairs that can offset each other's errors. Experimental results on six benchmark public datasets show that our temporal order enhanced Transformer (**TOEformer**) can preserve temporal order information and produce better forecasting performance in more than 85% of multivariate prediction setups compared to state-of-the-art time series forecasting models. The proposed relative value exponential loss effectively mitigates the overall error growth problem.

1. Introduction

Time series forecasting is widely used in many real-world applications, such as finance, transportation, and weather forecasting. The earlier time series forecasting methods mainly rely on traditional statistical models (Lotfalipour et al., 2013; Ariyo et al., 2014) and machine learning-based algorithms (Dai et al., 2018; AlKheder and Almusalam, 2022), which are difficult to effectively capture the nonlinear dynamic features in complex systems. In recent years, deep learning has developed rapidly and demonstrated excellent performance with its powerful automatic feature extraction capabilities.TCN (Temporal Convolutional Network) (Bai et al.; Liu et al., 2022b; Anonymous, 2024) uses convolutional kernels to model the data, while RNN (Recurrent Neural Network) (Qin et al., 2017; Lai et al., 2018) utilizes recurrent structures to capture data interdependencies. In particular, the emergence of the attention mechanism effectively solves the gradient decay problem of previous models in long-range dependency modeling, which not only shines in the field of Computer Vision (CV) (Dosovitskiy et al., 2021; Rao et al., 2021; Khan et al., 2022) and Natural Language Processing (NLP) (Brown et al., 2020; Kalyan et al., 2022), but also attracts more and more attention in the field of time series forecasting.

Transformer-based time series forecasting models are well suited for sequence modeling tasks as they automatically learn the dependencies between elements in a sequence through an attention mechanism. Recent research on Transformer-based work such as Informer (Zhou et al., 2021), Autoformer (Wu et al., 2021), FEDformer (Zhou et al., 2022), Pyraformer (Liu et al., 2021), and PatchTST (Nie et al., 2023) have achieved excellent results in time-series prediction tasks. However, the existing Transformer-based time series forecasting models do not preserve the temporal order well (Zeng et al., 2023; Chen et al., 2023). Chen et al. (2023) believe that the reason for this problem is that the attention-based model is a "data-dependent" model, meaning that the weights of the input sequences strongly depend on the input data. When the input data passes through the attention mechanism, the weights of the input sequences will change. Preserving temporal order information is crucial in time series modeling because it ensures continuity between predicted moments and reveals the dynamics changes of

* Correspondence to: Boda Campus of Xinjiang University, No. 777 Huarui Road, Shuimogou District, Urumqi, Xinjiang, China.
E-mail addresses: 107552201379@stu.xju.edu.cn (Q. Li), jwqin@xju.edu.cn (J. Qin).

https://doi.org/10.1016/j.engappai.2025.110965
Received 25 April 2024; Received in revised form 5 March 2025; Accepted 21 April 2025
Available online 12 May 2025

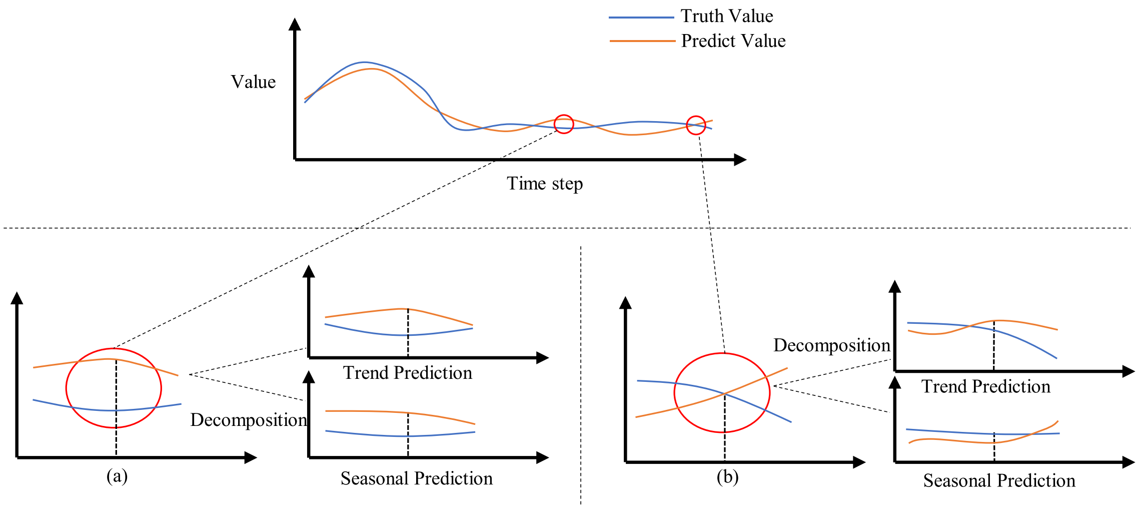


Fig. 1. The influence of subsequences prediction results on final prediction results. (a) Both the seasonal and trend prediction results are greater than the truth value, and errors accumulate. (b) One of the prediction results of the seasonal and trend part is greater than the corresponding truth value, and the other is smaller than the corresponding truth value, and the errors offset each other.

the data. Preserving temporal order information can make predictions more accurate. In order to solve this problem, this paper proposes a new Transformer-based time series forecasting model (TOEformer) that incorporates temporal order information by integrating Linear and Transformer structures. The reason for choosing Linear is the unique "time-step-dependent" feature of the linear model, in which the weights of the mappings are fixed for each time-step in the input sequence, which is in sharp contrast to the "data-dependent" feature of the Transformer. Ensemble learning integrates multiple individual models to obtain better generalization performance. Inspired by deep ensemble learning, TOEformer uses different neural networks to model the seasonal and trend data obtained from the decomposition of the input sequence separately. A Transformer architecture is used better to learn the seasonal features for the seasonal part. For the trend part, a Linear model is used to better preserve the temporal order information in the original series. The final prediction results consist of seasonal and trend forecast results.

Directly combining prediction results does not take into account the effect of seasonal and trend errors on the final prediction results (e.g., Fig. 1). When we forecast two subsequences and calculate their prediction losses, even though the prediction loss of each subsequence is small, after combining these two subsequences, the prediction loss of the combined sequence will not necessarily be small. This is because after combining the sequences, the predictions of the two subsequences may interact, resulting in the prediction performance of the combined sequence being different from that of the subsequences predicted separately. After studying Fig. 1, we found that when using a decomposition prediction network, the more combinations of two seasonal-trend pairs that are simultaneously greater or less than the corresponding truth value at each time step (e.g., Fig. 1(a)), the greater the error in the final prediction. On the contrary, if one part of the pair is greater than the corresponding truth value while the other part is less (e.g., Fig. 1(b)), the better the final prediction result is. Based on this finding, this paper proposes a relative value exponential loss(L_{RVE}) to alleviate this problem by generating more seasonal-trend pairs that can offset each other's errors.

The contributions of this paper are as follows:

- We propose a new Transformer-based time series forecasting model (TOEformer) that effectively utilizes temporal order information, using a sliding average kernel to decompose the original sequence, a Transformer network to learn the seasonal features, and a linear network to preserve the temporal order information in the sequence. The ability of Transformers to preserve temporal order is compensated by integrating the two parts of the network, and the final prediction results are obtained by combining the prediction results of the two-part networks.
- We design a new loss function, the relative value exponential loss, the core idea of which is to encourage the model to generate more seasonal-trend pairs that can offset part of the error in order to mitigate the cumulative effect of the error. The usefulness of the function is proved by experiments.
- We conducted experiments on six publicly available datasets, and TOEformer produced state-of-the-art predictions in over 85% of multivariate prediction settings.

2. Related work

Transformer-based Time Series Forecasting Models. The high computational complexity and large memory requirements associated with the self-attention mechanism make Transformer challenging for modeling long sequences (Li et al., 2019; Kitaev et al., 2020; Zhou et al., 2021). Therefore, much of the previous work has focused on reducing computational costs. Li et al. (2019) used causal convolution in the self-attention layer to reduce the space complexity to $\mathcal{O}(L(logL)^2)$, where L is the length of the input sequence. The locally-sensitive hash-attention mechanism proposed by Reformer (Kitaev et al., 2020) reduces the time complexity from $\mathcal{O}(L^2)$ to $\mathcal{O}(L\log L)$. Reformer (Kitaev et al., 2020) achieves faster and more efficient modeling of long sequences by using reversible residual networks in multi-layer neural networks. Informer (Zhou et al., 2021) proposes the ProbSparse attention mechanism to replace the original attention mechanism, achieving $\mathcal{O}(L\log L)$ time complexity and using distillation behind each attention layer to reduce the space complexity significantly.

While most previous research has focused on reducing computational cost, recent research has increasingly focused on extracting sequence features using Transformer, as its ability to model sequences has been progressively exploited. Autoformer (Wu et al., 2021) devised a new architecture to achieve progressive decomposition of input sequences and proposed an autocorrelation mechanism to extract periodical features in time series. Pyraformer (Liu et al., 2021) proposes a pyramid model architecture to simultaneously learn temporal features at different time scales in a compact multi-resolution manner, acquiring more extensive time scale features with less time space spent. FEDformer (Zhou et al., 2022) uses a frequency-enhanced decomposition of the Transformer architecture to capture global attributes of a time series and proposes Fourier-enhanced and Wavelet-enhanced blocks to capture important structures in time series through frequency domain mapping. NSTransformer (Liu et al., 2022a) makes series more predictable through smoothing and avoids over-smoothing by using a de-smoothing attention mechanism to recombine the non-stationarity of the original series. PatchTST (Nie et al., 2023) enhances local information and captures comprehensive semantic information unavailable in dot product attention by aggregating time steps into subsequence-level patches. iTransformer (Liu et al., 2024) reverses the order of information processing, first analyzing the time series information for each individual variable and then fusing the information from all the variables. Basisformer (Ni et al., 2023) regarded the historical and future parts of the sequence as two different views, designed the Coef module, and used comparative learning to measure the similarity coefficient between the time series and the bases in the historical view to select the time series bases in the future view. TimesNet (Wu et al., 2023) converts 1D time series into 2D tensors and uses a 2D kernel to capture intra-periodic and inter-periodic variations. GCformer (Zhao et al., 2023) utilizes a low-complexity global convolutional branch to capture long-term dependencies and effectively captures fine-grained recent information using a Transformer-based local convolutional branch. Ultimately, it models complex sequence relationships by combining global and local convolutions.

MLP (Multi-Layer Perceptron)-based Time Series Forecasting Models. While Transformers has made excellent progress in time-series prediction tasks, DLinear (Zeng et al., 2023) questioned the effectiveness of Transformers, as DLinear achieved better performance than Transformer-based SOTA models with only a simple linear layer. Oreshkin et al. (2020) did some increased interpretable research in N-BEATS, where they proposed a deep neural architecture based on backward and forward residual links and injected suitable inductive biases into the model to make its internal operations more interpretable and generalizable. Ekambaram et al. (2023) uses two intrinsic properties of time series, hierarchical patch aggregation, and cross-channel correlation, to tune and improve forecasts. Chen et al. (2023) uses a linear model to capture temporal patterns and exploits cross-variate information by alternately applying MLP in the time and feature domains. Liu et al. (2023) introduced time-varying dynamics and Koopman theory into time series prediction by combining a modular Fourier filter and a Koopman predictor for time series prediction using time-variant and time-invariant components. DEPTS (Fan et al., 2022) focuses on modeling periodic time series by learning the complex dependence of the time series signals on the periodicity through a deep residual network. In addition, they proposed a periodic module with parameterized periodic functions to capture diverse periodic features in time series. Wang et al. (2024) propose TimeMixer as a fully MLP-based architecture with Past-Decomposable-Mixing and Future-Multipredictor-Mixing blocks to take full advantage of disentangled multiscale series in both past extraction and future prediction phases.

3. Methods

In this section, we first present the overall architecture of TOEformer (see Fig. 2) and the structure and functionality of each subnetwork, then introduce a new loss function, the relative-value exponential loss, to mitigate the error growth problem.

3.1. Model architecture

Fig. 2 shows the overall architecture of TOEformer, which consists of two main parts: seasonal prediction structure and trend prediction structure. Channel independence means that each variable sequence predicts only the data within its original sequence. The C features of the input sequence are processed separately and independently using the channel independence technique, and the sequence $x(i)$ is decomposed by the moving average kernel decomposition module. We utilized normalization and reverse normalization techniques for trend and seasonality prediction networks to effectively capture sequence features and dependencies and reduce the sensitivity of TOEformer to outliers or extreme values for accurate predictions. In seasonal prediction, the encoder uses global convolution and local convolution to learn global and local features, respectively, and uses seasonal tails to obtain the time dependence around the prediction step and to reduce the computational expense. The seasonal tail is the historical sequence of seasonal items taken from the seasonal items near the forecast window. The decoder uses the cross-attention mechanism to obtain an efficient representation of the seasonal features, and the output from the decoder is the output of the entire seasonal prediction network. A simple linear layer is used in trend prediction to preserve temporal order information. The final forecast result is obtained by adding the trend forecast result and the seasonal forecast result.

3.1.1. Sequence decomposition

To learn with the complex temporal patterns in long-term forecasting context, we take the idea of decomposition, which can separate the series into trend-cyclical and seasonal parts. These two parts reflect the long-term progression and the seasonality of the series respectively. For length L input series $X \in \mathbb{R}^{L\times F}$, the process is:

$$X_t = AvgPool(Padding(X)) \tag{1}$$

$$X_S = X - X_t \tag{2}$$

where $X_s, X_t \in \mathbb{R}^{L\times F}$ denote the seasonal and the extracted trend-cyclical part respectively. F means the feature numbers of input series. We adopt the $AvgPool()$ for moving average with the padding operation to keep the series length unchanged.

3.1.2. Trend prediction

Trend prediction is a crucial aspect of complex time series data. The trend part usually represents a significant component of the time series and reflects the underlying direction of the data over time. We use a simple linear layer to extract and learn trend features. Despite the relative simplicity of the linear layer, its usefulness cannot be ignored. The linear layer captures trends in time series and preserves the temporal order of the data.

3.1.3. Seasonal prediction

In time series forecasting, especially seasonal forecasting, it is crucial to capture the dynamic features on different scales of the time series. To achieve this goal, we employ a strategy that combines local and global convolution as the core component of the encoder. This combined approach considers the long-term time dependence and the recent dynamics and provides the model with richer and more diverse information.

The main goal of global convolution is to capture seasonal patterns in the time series as a whole. The global convolution utilizes larger convolutional kernels to capture long-term dependencies in the sequence and extract seasonal patterns across the entire time series. Local convolution, on the other hand, focuses on recent time series dynamics. It uses smaller convolution kernels to extract recent dynamics around the prediction window. For instance, hour-level forecasting may be more concerned with data changes in the previous hours or days. Local convolution can effectively capture these recent dynamics and patterns.

However, extracting features solely through convolution is insufficient. We need a mechanism to integrate the features extracted from global and local convolutions so that the model considers both long-term and short-term dependencies in predictions. We adopt a cross-attention mechanism as the decoder to combine global and local information. The K and V matrices come from local convolution, and the Q matrix comes from global convolution. Formally, the cross attention can be written as:

$$Attention(Q,K,V) = Softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{3}$$

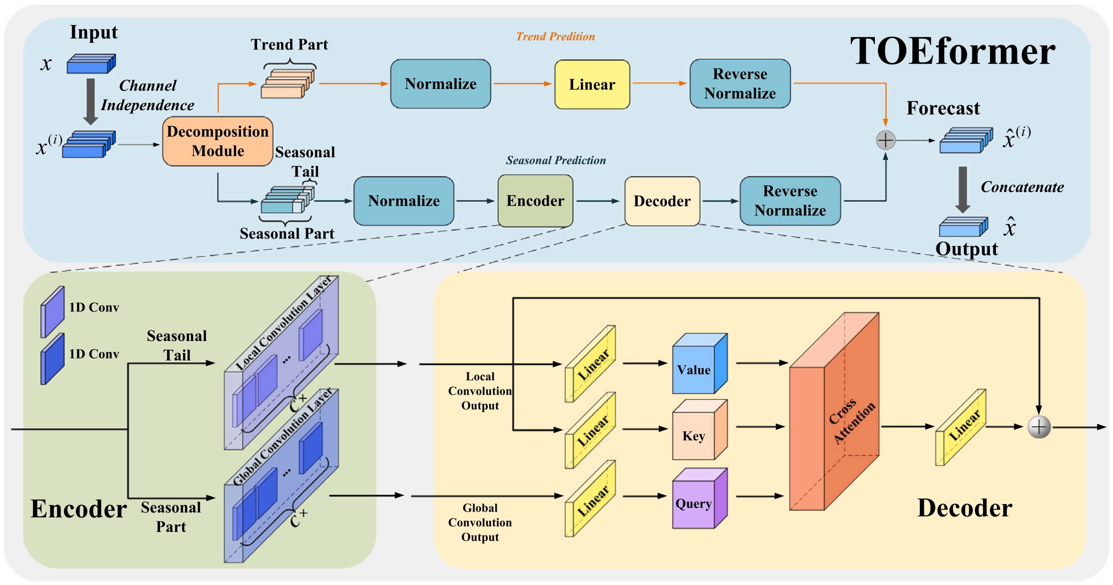


Fig. 2. The overall framework of TOEformer consists of two main parts: seasonal prediction and trend prediction, where $x \in R^{C\times L}, x^{(i)} \in R^{1\times L}, \hat{x}^{(i)} \in R^{1\times T}, \hat{x} \in R^{C\times T}, i = 1, \ldots, C$, C is the number of features, L is the length of the lookback window, and T is the prediction length.

where $Softmax$ is activation function, d_k is the dimension size of the K matrix.

We use cross-attention to fuse local and global information. In this way, the model captures the long-term dependencies and seasonal patterns of the time series and the short-term dynamics around the forecast window. The global–local convolution structure provides an excellent framework for more accurate forecasting of seasonal data.

3.1.4. Time complexity

The input sequence L is decomposed into seasonal and trend parts. The complexity of the process is $\mathcal{O}(L)$. For the trend part, the complexity of its linear transformation is $\mathcal{O}(L \times T)$, where T is the prediction length. For the seasonal part, The time complexity of Encoder is $\mathcal{O}(L \times K \times F^2)$, where K is the convolutional kernel size, F is the number of features. The time complexity of Decoder is $\mathcal{O}(L^2 \times F)$. The complexity of sequence merging is $\mathcal{O}(L)$. So the final time complexity is $\mathcal{O}(L^2 \times F)$.

3.2. The relative value exponential loss

3.2.1. Error growth

We conducted subsequence prediction bias experiments to study the impact of subsequence prediction results on the final result. We used the DLinear model with a decomposition architecture to make predictions on the ETTh2 dataset. We obtained the final prediction results by adding bias to the subsequence prediction results. The experimental results are shown in Fig. 3.

Fig. 3(e) and (h) are the results obtained by adding and subtracting bias β[1] from trend prediction results respectively. In the case of Fig. 3(d), (e), and (f), although the MSE scores of the seasonal and trend prediction results are poor, the final results after combining the seasonal and trend parts are consistent with the results before adding the bias (Fig. 3(c) and Fig. 3(f)). In the case of Fig. 3(g), (h), and (i), not only the MSE scores of the forecast results of the seasonal and trend are poor, but the final forecast results (Fig. 3(i)) also become several times worse. In other words, when using a seasonal-trend prediction network, the more combinations of two seasonal-trend pairs at each time step that are simultaneously greater or less than the corresponding truth value (e.g., Fig. 3(d), (e) and (f)), the more the error grows in the final prediction, and conversely the more combinations that occur in which one part is greater than the corresponding truth value. The other is less than the corresponding truth value (e.g., Fig. 3(g), (h) and (i)), the better the final prediction results.

By analyzing the prediction results, we found that this increase in error occurs mainly in cases where the seasonal and trend prediction values are simultaneously larger or smaller than the corresponding truth values. In these cases, the overall error increases because the prediction errors of the seasonal and trend parts accumulate when combined. Conversely, suppose one of the seasonal and trend forecasts is greater than the corresponding truth values, and the other is less than the corresponding truth values; the errors of the seasonal and trend parts will partially offset each other when combined, thus contributing to a reduction in the overall forecast error. Based on this finding, we realized that to reduce the overall prediction error, the model must generate as many seasonal-trend pairs as possible that can offset each other's errors.

3.2.2. Loss function design

In order to address the problem of error growth, this paper proposes a new loss function: the relative value exponential loss. The design idea of this loss function is to encourage the model to generate more pairs of forecasts that can offset each other by adjusting the relationship between the seasonal and trend forecasts. Specifically, when both the seasonal and trend predictions are either greater than or less than their corresponding truth values, the relative value exponential loss imposes a larger penalty. However, when one is greater while the other is smaller than their corresponding ground truth values, the relative value exponential loss imposes a smaller penalty. Formally, the relative value exponential loss (L_{RVE}) can be described as:

$$L_{RVE} = \frac{1}{L}\sum_{i=1}^{L}\Big(1 - \alpha^{(Seasonal\,Error + Trend\,Error)^2} + Seasonal\,Error^2 + Trend\,Error^2\Big) \tag{4}$$

$$\begin{aligned} Seasonal\,Error &= S_i - \hat{S}_i \\ Trend\,Error &= T_i - \hat{T}_i \end{aligned} \tag{5}$$

where $Seasonal\,Error$ is the relative error in the prediction of the seasonal part, $Trend\,Error$ is the relative error in the prediction of the trend part, L is the length of the sequence, S_i is the true value of the

[1] Adding or subtracting β bias means adding or subtracting β to each value to the original prediction results, β=0.3 in here.

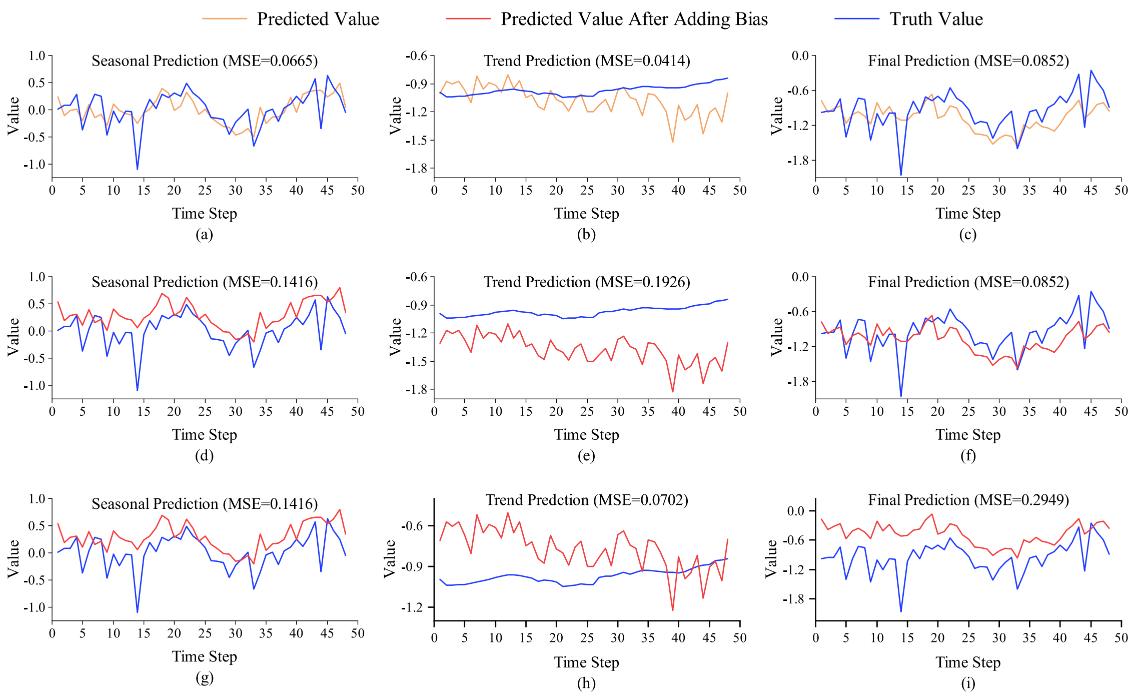


Fig. 3. The difference between the original prediction results of DLinear on a real-world ETTh2 dataset and the results after adding bias. The Original prediction results: (a), (b), (c). The prediction results after adding bias: (d), (e), (f) and (g), (h), (i).

seasonal part, $\hat{S}_i$ is the predicted value of the seasonal part, T_i is the true value of the trend part, $\hat{T}_i$ is the predicted value of the trend part, and α is the hyperparameter. The final loss function used for the whole model is as follows:

$$L = \lambda_1 L_{total} + \lambda_2 L_{RVE} \tag{6}$$

where L_{total} is the final prediction mean square error (MSE), λ_1 and λ_2 are hyperparameters. By constraining the output from the model with L_{RVE}, the model learns how to more accurately predict seasonal and trend parts during training and create a balance between the two to reduce the overall error when combined.

4. Experiments

4.1. Datasets and experimental details

Datasets. We conducted extensive experiments on six real-world datasets, namely ETTh1, ETTh2, ETTm1, ETTm2, Weather, and Exchange_rate. All of the above datasets are multivariate time series, and the detailed properties of the datasets are shown in Table 1. ETT (Electricity Transformer Temperature) contains data collected from a power transformer with seven sensors, including load and oil temperature. It contains two sub-datasets labeled 1 and 2, corresponding to two different power transformers from two different counties in China. Each contains two different resolutions (15 min and 1 h), denoted by m and h. Thus, we have 4 ETT datasets: ETTh1, ETTh2, ETTm1, and ETTm2. Weather contains 21 meteorological indicators, such as humidity and temperature, for the whole year of 2020 in Germany. Exchange contains daily exchange rates for eight different countries from 1990 to 2016. For ETT dataset, we use the ratio of 6:2:2 to divide the original data set into training set, test set and verification set. For both the Weather and Exchange datasets, the ratio of training set: test set: validation set is 7:2:1. The segmentation ratio of these three data sets is also the same one used in most previous work.

Baseline and Experimental Details. We selected popular time series forecasting models as baselines for comparison, including:

Table 1
Detailed properties of the six publicly available datasets used for the experiments, including time steps, number of features, and sampling frequency.

Dataset	Time Steps	Features	Frequency
ETTh1	17 420	7	1 h
ETTh2	17 420	7	1 h
ETTm1	69 680	7	15 min
ETTm2	69 680	7	15 min
Weather	52 696	21	10 min
Exchange	7588	8	1 day

- TimeMixer (Wang et al., 2024) proposes decomposable mixing and future multi-prediction mixing blocks to capture temporal dependencies at multiple time scales.
- MSGNet (Cai et al., 2024) employs frequency domain analysis and adaptive graph convolution to capture cross-variate dependencies at different time scales.
- iTransformer (Liu et al., 2024) reverses the order of information processing, first analyzing the time series information for each individual variable and then fusing the information from all the variables.
- PatchTST (Nie et al., 2023) enhances local information by aggregating time steps into subsequence-level patches.
- DLinear (Zeng et al., 2023) proposed a simple linear model based on decomposition.
- Basisformer (Ni et al., 2023) builds time series forecasting bases through self-supervised comparative learning and utilizes bidirectional attention to dynamically select relevant bases for forecasting across historical and future views.
- TimesNet (Wu et al., 2023) converts 1D time series into 2D tensors and uses a 2D kernel to capture intra-periodic and inter-periodic variations.
- FEDformer (Zhou et al., 2022) uses a frequency-enhanced decomposition of the Transformer architecture to capture global attributes of time series.

Table 2
Multivariate Long-Term prediction results. 96,192,336 and 720 is different prediction length. The smaller the MAE and MSE scores, the better the performance. The best results are in **bold**, the second best are underlined.

Models		TOEformer		TimeMixer		MSGNet		iTransformer		PatchTST		DLinear	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	**0.348**	**0.385**	0.375	0.400	0.390	0.411	0.386	0.405	0.375	0.399	0.378	0.402
	192	**0.364**	**0.402**	0.429	0.421	0.442	0.442	0.442	0.436	0.414	0.421	0.415	0.425
	336	**0.409**	**0.424**	0.484	0.458	0.480	0.468	0.490	0.461	0.425	0.440	0.447	0.448
	720	0.452	**0.469**	0.498	0.482	0.494	0.488	0.508	0.493	**0.449**	0.470	0.481	0.490
ETTh2	96	**0.249**	**0.321**	0.289	0.341	0.328	0.371	0.301	0.350	0.274	0.337	0.290	0.350
	192	**0.299**	**0.360**	0.372	0.392	0.402	0.414	0.380	0.398	0.339	0.379	0.375	0.410
	336	**0.288**	**0.360**	0.386	0.414	0.435	0.443	0.423	0.433	0.331	0.380	0.459	0.462
	720	**0.354**	**0.413**	0.412	0.434	0.417	0.441	0.432	0.447	0.379	0.422	0.733	0.606
ETTm1	96	0.303	**0.346**	0.320	0.357	0.319	0.366	0.341	0.376	**0.295**	0.348	0.307	0.350
	192	**0.330**	**0.370**	0.361	0.381	0.376	0.397	0.383	0.395	0.334	0.373	0.347	0.381
	336	**0.363**	**0.392**	0.390	0.404	0.417	0.422	0.418	0.418	0.369	**0.392**	0.380	0.394
	720	**0.416**	**0.416**	0.454	0.441	0.481	0.458	0.488	0.457	**0.416**	0.419	0.425	0.421
ETTm2	96	**0.164**	**0.250**	0.175	0.258	0.177	0.262	0.186	0.272	0.166	0.256	0.167	0.260
	192	**0.220**	**0.287**	0.237	0.299	0.247	0.307	0.254	0.314	0.223	0.296	0.223	0.304
	336	**0.264**	**0.323**	0.298	0.340	0.312	0.346	0.316	0.351	0.269	0.329	0.291	0.355
	720	**0.336**	**0.374**	0.391	0.396	0.414	0.403	0.414	0.407	0.350	0.380	0.407	0.433
Weather	96	0.167	0.229	0.163	0.209	0.163	0.212	0.178	0.219	**0.147**	**0.198**	0.168	0.226
	192	**0.189**	0.256	0.208	0.250	0.212	0.254	0.226	0.259	0.190	**0.240**	0.211	0.265
	336	**0.236**	0.293	0.251	0.287	0.272	0.299	0.282	0.300	0.242	**0.281**	0.257	0.308
	720	**0.299**	0.339	0.339	0.341	0.350	0.348	0.358	0.350	0.305	**0.329**	0.315	0.354
Exchange	96	**0.075**	**0.192**	0.085	0.203	0.102	0.230	0.086	0.206	0.083	0.200	0.078	0.199
	192	**0.146**	**0.281**	0.177	0.299	0.195	0.317	0.180	0.303	0.171	0.295	0.155	0.290
	336	**0.252**	**0.370**	0.356	0.433	0.359	0.436	0.338	0.422	0.350	0.424	0.307	0.419
	720	**0.594**	**0.582**	1.075	0.765	0.940	0.738	0.858	0.699	0.831	0.685	0.782	0.669

Table 3
This table is a continuation of Table 2.

Models		TimesNet		Basisformer		FEDformer		Autoformer		Informer	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.384	0.402	0.398	0.414	0.375	0.415	0.484	0.467	0.941	0.769
	192	0.436	0.429	0.449	0.439	0.427	0.449	0.491	0.484	1.007	0.786
	336	0.491	0.469	0.479	0.453	0.459	0.465	0.461	0.464	1.038	0.784
	720	0.521	0.500	0.490	0.483	0.485	0.493	0.574	0.554	1.161	0.851
ETTh2	96	0.340	0.374	0.315	0.361	0.342	0.385	0.348	0.399	2.905	1.344
	192	0.402	0.414	0.376	0.397	0.434	0.441	0.459	0.450	6.177	2.078
	336	0.452	0.452	0.423	0.434	0.502	0.494	0.474	0.477	5.328	1.967
	720	0.462	0.468	0.463	0.467	0.478	0.485	0.489	0.492	3.988	1.663
ETTm1	96	0.338	0.375	0.354	0.383	0.364	0.413	0.444	0.452	0.626	0.560
	192	0.374	0.387	0.393	0.401	0.390	0.423	0.619	0.521	0.725	0.619
	336	0.410	0.411	0.421	0.417	0.456	0.463	0.681	0.543	1.045	0.757
	720	0.478	0.450	0.498	0.460	0.504	0.489	0.704	0.571	1.230	0.823
ETTm2	96	0.187	0.267	0.183	0.263	0.189	0.283	0.266	0.324	0.407	0.494
	192	0.249	0.309	0.252	0.308	0.256	0.324	0.274	0.333	0.813	0.706
	336	0.321	0.351	0.310	0.344	0.327	0.364	0.405	0.397	1.446	0.923
	720	0.408	0.403	0.408	0.401	0.436	0.426	0.440	0.433	3.889	1.459
Weather	96	0.172	0.220	0.174	0.214	0.241	0.328	0.266	0.338	0.397	0.442
	192	0.219	0.261	0.219	0.256	0.282	0.344	0.301	0.360	0.498	0.500
	336	0.280	0.306	0.279	0.299	0.363	0.403	0.360	0.393	0.584	0.538
	720	0.365	0.359	0.355	0.347	0.399	0.414	0.409	0.420	1.009	0.744
Exchange	96	0.107	0.234	0.084	0.205	0.138	0.267	0.143	0.273	0.855	0.732
	192	0.226	0.344	0.179	0.301	0.277	0.384	0.283	0.388	1.107	0.838
	336	0.367	0.448	0.329	0.417	0.449	0.493	0.448	0.499	1.624	1.016
	720	0.964	0.746	0.876	0.709	1.154	0.823	1.176	0.845	2.937	1.417

- Autoformer (Wu et al., 2021) uses a seasonal-trend decomposition structure and proposes an autocorrelation mechanism to mine the internal dependencies of sequences.
- Informer (Zhou et al., 2021) proposes the ProbSparse attention mechanism and improves the computational efficiency of Transformer to $\mathcal{O}(L\log L)$.

We mainly use mean square error (MSE) and mean absolute error (MAE) as evaluation metrics to evaluate the models. All the experiments in this paper are performed on a single Nvidia RTX A5000 GPU.

4.2. Main results

4.2.1. Multivariate long-term prediction results

Table 2 and Table 3 shows the multivariate long-term prediction results. Fig. 4 shows the visualized result of the prediction. From the experimental results, it can be seen that TOEformer shows strong competitiveness on all datasets, and by combining the temporal order information of the original input sequences, TOEformer achieves either optimal or sub-optimal results on many prediction settings, with the lowest or the second-lowest MSE or MAE scores for its predictions.

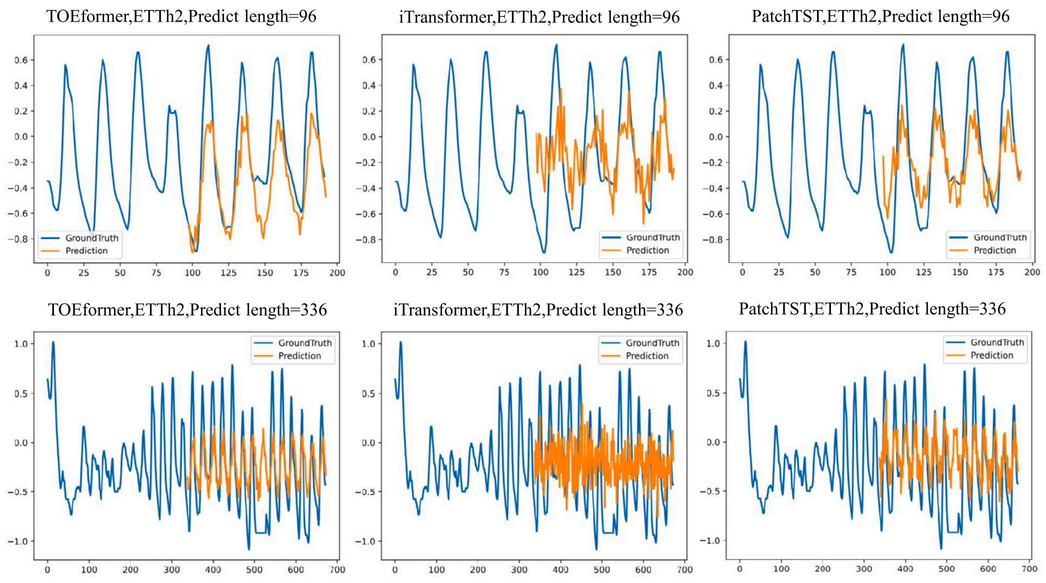


Fig. 4. Visualization of prediction results.

Compared to the baselines, TOEformer reduces the MSE by 1.1% to 24.0%, with an average reduction of 6.9%.

Notably, TOEformer is more effective on the ETT and Exchange datasets. Specifically, the forecasting performance of TOEformer on the ETTh2 and ETTm2 datasets generally outperforms all baselines. On the ETTh1 and ETTm1 datasets TOEformer outperforms the baselines on most prediction settings, and only slightly underperforms PatchTST at a prediction length of 720, but still outperforms most Transformer variants. The experiment results indicate that TOEformer can use Transformer to effectively capture seasonal features in the seasonal-trend framework, and its linear modeling branch can effectively capture the temporal order information in the trend terms compared with the Transformer-based baselines, thus achieving excellent performance.

TOEformer performs well on the Exchange dataset compared to the baseline performance (11.0% reduction on average). This may be due to the fact that the data distribution in the Exchange dataset is characterized by a combination of long-term trends and short-term high-frequency fluctuations, when the seasonal-trend modeling architecture is able to effectively capture these characteristics and achieve the shortest forecasts.

The forecasting results of TOEformer on the Weather dataset did not improve significantly compared to the baselines, which may be due to the fact that the complex correlations of multivariate features on the Weather dataset may not have been adequately modeled, affecting TOEformer's overall performance.

4.2.2. *Efficiency analysis*

To further validate the computational efficiency of our proposed method, we performed training time statistics for all experiments in Table 2, and the results are placed in Table 4. On all datasets and all prediction lengths, the training time of TOEformer is consistently low, generally lower than that of the classical Transformer models (TimeNet, Basisformer, FEDformer, Autoformer, and Informer), and even on the faster than some of the lightweight MLP-based models (TimeMixer, DLinear) for some prediction settings. Compared to the novel Transformer models (iTtransformer and PatchTST), TOEformer is slightly slower overall, with training times slightly higher than these models for longer prediction lengths and slightly lower than these baselines for shorter prediction lengths, due to the fact that we set up the high-dimensional projection layer of Transformer based on the prediction length to better capture features within the sequence. In addition, the training time for TOEformer is much smaller than MSGNet, because MSGNet uses a graph convolution module when modeling sequences, which reduces the training speed.

4.3. *Can TOEformer preserve temporal order information well?*

In order to verify the ability of TOEformer to preserve the temporal order information, we conducted the input shuffling experiments on ETTh2, Weather, and Exchange datasets, and the results are shown in Table 5. The experiment only disrupts the input sequence on the test set and does not change the input sequence on the training and validation sets, so it can verify whether the model is sensitive to the order of the input sequence. When the prediction performance decreases, it indicates that the model is sensitive to the input sequence, and vice versa. In this experiment we refer to the experimental setup in DLinear (Zeng et al., 2023), which are the same as in Table 2, except that the inputs are swapped and shuffled. For the ETTh2 dataset, it can be seen that the four Transformer models, Basisformer, FEDformer, Autoformer, and Informer, have a slight drop in performance, which suggests that the Transformer model is less sensitive to the input sequences with periodic features and is prone to overfitting when carrying out the modeling. In contrast, the Linear model is susceptible to input sequences, with an average drop of 23.08% and 10.42% under *Shuf.* and *Half-Ex.* input settings, which suggests that Linear can model temporal order naturally, avoid overfitting with fewer parameters, and preserving the sequential information in the input sequences. Similarly, TOEformer is also sensitive to input sequences, with average decreases of 25.47% and 15.53% under *Shuf.* and *Half-Ex.* input settings, which suggests that TOEformer can retain the temporal order information in the original input sequences. Besides, compared with Informer, the reason why Basisformer, FEDformer, and Autoformer reach a slight performance drop is that Basisformer modeling data by using some basis subsequences, FEDformer with wavelet, and autoformer with autocorrelation mechanism.

For the other two datasets, due to the absence of clear periodic patterns in their data, Autoformer and Informer overfit during modeling, resulting in little change in performance. FEDformer and Basisformer experience significant performance degradation during prediction because the basis sequences and wavelets used internally are seen as subsequences of the input, and when the input changes, the subsequences also change. DLinear and TOEformer also show noticeable performance decline because both models consider the temporal order

Table 4
Per-epoch training time comparison (seconds).

Models		TOEformer	TimeMixer	MSGNet	iTransformer	PatchTST	DLinear	TimesNet	Basisformer	FEDformer	Autoformer	Informer
ETTh1	96	1.7	3.7	29.5	4.4	2.2	1.9	16.2	9.4	46.1	15.8	10.8
	192	2.4	3.4	25.2	4.4	1.9	1.9	15.5	11.1	48.9	18.7	11.5
	336	3.7	3.7	59.2	4.2	2.0	2.0	16.9	9.8	57.8	24.6	13.7
	720	8.3	3.5	50.8	3.9	2.0	2.2	22.7	10.2	74.1	40.1	18.6
ETTh2	96	2.1	17.7	66.1	4.2	1.9	2.0	15.5	10.3	46.6	15.2	10.8
	192	2.7	17.0	49.8	4.1	1.9	2.0	16.0	10.9	51.3	18.7	11.5
	336	4.7	16.3	49.3	3.8	1.9	2.1	26.1	11.1	56.7	24.6	13.5
	720	8.0	16.2	58.8	5.0	1.9	2.0	42.3	9.8	75.0	39.8	18.5
ETTm1	96	8.6	67.0	69.3	15.2	12.9	17.7	73.8	45.0	177.9	58.8	37.9
	192	14.9	65.4	79.3	13.7	12.8	18.3	85.9	44.3	197.7	73.9	42.3
	336	24.9	62.8	67.5	13.0	13.0	18.0	54.0	40.8	232.1	100.0	53.7
	720	39.7	62.5	76.7	14.8	13.3	18.1	80.8	39.5	321.3	169.5	77.1
ETTm2	96	9.5	9.4	111.9	15.2	12.9	5.8	51.4	44.3	184.2	59.3	39.4
	192	15.7	9.8	105.7	14.4	12.9	5.9	57.9	44.6	199.2	74.1	43.4
	336	51.7	10.2	106.1	13.6	13.1	6.5	73.8	44.1	234.2	99.7	53.9
	720	64.7	10.7	107.3	16.7	13.2	6.7	88.4	41.6	317.3	170.0	76.4
Weather	96	8.4	14.0	205.9	18.7	37.5	12.3	57.9	45.0	195.7	63.8	41.1
	192	17.2	13.8	361.6	18.5	38.1	12.2	77.3	45.9	214.4	79.3	46.6
	336	21.3	14.2	109.0	19.4	38.3	13.0	86.6	44.0	249.8	107.8	58.9
	720	41.6	15.7	202.1	23.2	38.8	13.9	218.5	44.4	343.2	182.8	84.1
Exchange	96	1.1	2.6	43.1	3.2	9.0	3.2	13.0	6.0	28.2	9.7	6.5
	192	1.4	2.6	70.3	4.2	8.7	3.1	13.6	6.5	30.8	11.7	7.2
	336	4.0	2.5	67.9	3.2	6.5	1.4	16.0	5.7	34.4	15.1	8.7
	720	5.4	2.7	62.7	3.6	6.3	1.5	34.9	5.4	43.9	23.4	11.2

Table 5
Comparison of the MSE of the models when shuffling the original input sequence. Ori. indicates that the input sequence is not processed, *Shuf.* indicates that the input sequence is randomly shuffled, and *Half-Ex.* indicates that the first and second halves of the input sequence are randomly swapped. Drop means the performance drop after shuffling the inputs, and the Average Drop is the average value of the performance degradation over all the predicted lengths of the inputs after shuffling them. The performance degradation results are expressed in the form of percentages.

Models	Input and Performance Drop	ETTh2				Weather				Exchange			
		96	192	336	720	96	192	336	720	96	192	336	720
	Ori.	0.290	0.375	0.459	0.733	0.168	0.211	0.257	0.315	0.078	0.155	0.307	0.782
	Shuf.	0.425	0.481	0.515	0.774	0.298	0.315	0.337	0.368	0.141	0.295	0.341	0.794
	Half-Ex.	0.338	0.418	0.501	0.766	0.172	0.213	0.259	0.316	0.080	0.158	0.311	0.783
DLinear	*Shuf.* Drop	46.40%	28.17%	12.17%	5.57%	77.35%	49.30%	31.19%	16.81%	80.75%	90.63%	11.08%	1.54%
	Half-Ex. Drop	16.45%	11.49%	9.21%	4.56%	2.53%	1.06%	0.82%	0.36%	3.14%	1.64%	1.38%	0.08%
	Shuf. Average Drop	23.08%				43.66%				46.00%			
	Half-Ex. Average Drop	10.42%				1.19%				1.56%			
	Ori.	0.249	0.299	0.288	0.354	0.167	0.189	0.236	0.299	0.075	0.146	0.252	0.594
	Shuf.	0.320	0.404	0.355	0.446	0.234	0.240	0.316	0.381	0.133	0.230	0.651	1.053
	Half-Ex.	0.308	0.353	0.341	0.398	0.191	0.197	0.275	0.363	0.119	0.236	0.396	0.974
TOEformer	*Shuf.* Drop	27.15%	32.02%	16.85%	25.86%	40.30%	26.96%	33.96%	27.49%	77.29%	57.72%	158.44%	77.26%
	Half-Ex. Drop	22.17%	15.31%	12.30%	12.33%	14.73%	4.04%	16.62%	21.54%	58.24%	61.44%	57.18%	63.90%
	Shuf. Average Drop	25.47%				32.18%				92.68%			
	Half-Ex. Average Drop	15.53%				14.23%				60.19%			
	Ori.	0.315	0.376	0.423	0.463	0.174	0.219	0.279	0.355	0.084	0.179	0.329	0.876
	Shuf.	0.365	0.440	0.457	0.459	0.219	0.265	0.312	0.378	0.175	0.275	0.424	0.976
	Half-Ex.	0.359	0.434	0.465	0.459	0.200	0.241	0.292	0.363	0.105	0.202	0.365	0.932
Basisformer	*Half-Ex.* Drop	14.05%	15.38%	10.00%	−0.80%	15.31%	9.85%	4.54%	2.13%	24.78%	12.83%	11.24%	6.36%
	Shuf. Drop	16.00%	17.06%	8.01%	−0.89%	26.18%	20.69%	11.75%	6.37%	108.03%	54.12%	29.16%	11.40%
	Shuf. Average Drop	10.04%				16.25%				50.68%			
	H*Half-Ex.* Average Drop	9.66%				7.96%				13.80%			
	Ori.	0.342	0.434	0.502	0.478	0.241	0.282	0.363	0.399	0.138	0.277	0.449	1.154
	Shuf.	0.376	0.453	0.519	0.482	0.269	0.332	0.368	0.408	0.192	0.278	0.450	1.154
	Half-Ex.	0.362	0.445	0.514	0.480	0.237	0.287	0.360	0.396	0.147	0.278	0.449	1.154
FEDformer	*Shuf.* Drop	9.84%	4.29%	3.44%	0.88%	11.42%	17.76%	1.44%	2.16%	39.01%	0.39%	0.15%	0.02%
	Half-Ex. Drop	5.86%	2.64%	2.48%	0.35%	−1.60%	1.64%	−0.85%	−0.87%	6.88%	0.25%	0.11%	0.03%
	Shuf. Average Drop	4.61%				8.19%				9.89%			
	Half-Ex. Average Drop	2.83%				−0.42%				1.82%			
	Ori.	0.348	0.459	0.474	0.489	0.266	0.301	0.360	0.409	0.143	0.283	0.448	1.176
	Shuf.	0.402	0.502	0.493	0.495	0.232	0.309	0.349	0.414	0.158	0.270	0.447	1.176
	Half-Ex.	0.387	0.483	0.492	0.506	0.260	0.313	0.364	0.414	0.144	0.279	0.448	1.176
Autoformer	*Shuf.* Drop	15.53%	9.29%	4.18%	1.19%	−12.62%	2.39%	−2.97%	1.25%	10.88%	−4.62%	−0.24%	0.00%
	Half-Ex. Drop	11.20%	5.10%	3.86%	3.53%	−2.17%	3.96%	1.27%	1.21%	1.21%	−1.24%	−0.02%	0.01%
	Shuf. Average Drop	7.55%				−2.99%				1.51%			
	Half-Ex. Average Drop	5.92%				1.06%				−0.01%			

(*continued on next page*)

Table 5 (*continued*).

Models	Input and Performance Drop	ETTh2				Weather				Exchange			
		96	192	336	720	96	192	336	720	96	192	336	720
	Ori.	2.905	6.177	5.328	3.988	0.397	0.498	0.584	1.009	0.855	1.107	1.624	2.937
	Shuf.	2.920	6.190	5.341	3.989	0.383	0.469	0.559	1.014	0.855	1.107	1.623	2.933
	Half-Ex.	2.906	6.179	5.330	3.988	0.396	0.495	0.580	1.008	0.855	1.108	1.624	2.936
Informer	*Shuf.* Drop	0.48%	0.21%	0.23%	0.04%	−3.47%	−5.88%	−4.26%	0.52%	−0.04%	−0.04%	−0.04%	−0.13%
	Half-Ex. Drop	0.03%	0.05%	0.04%	0.00%	−0.30%	−0.74%	−0.70%	−0.08%	−0.04%	0.07%	−0.01%	−0.04%
	Shuf. Average Drop	0.24%				−3.27%				−0.06%			
	Half-Ex. Average Drop	0.03%				−0.46%				0.00%			

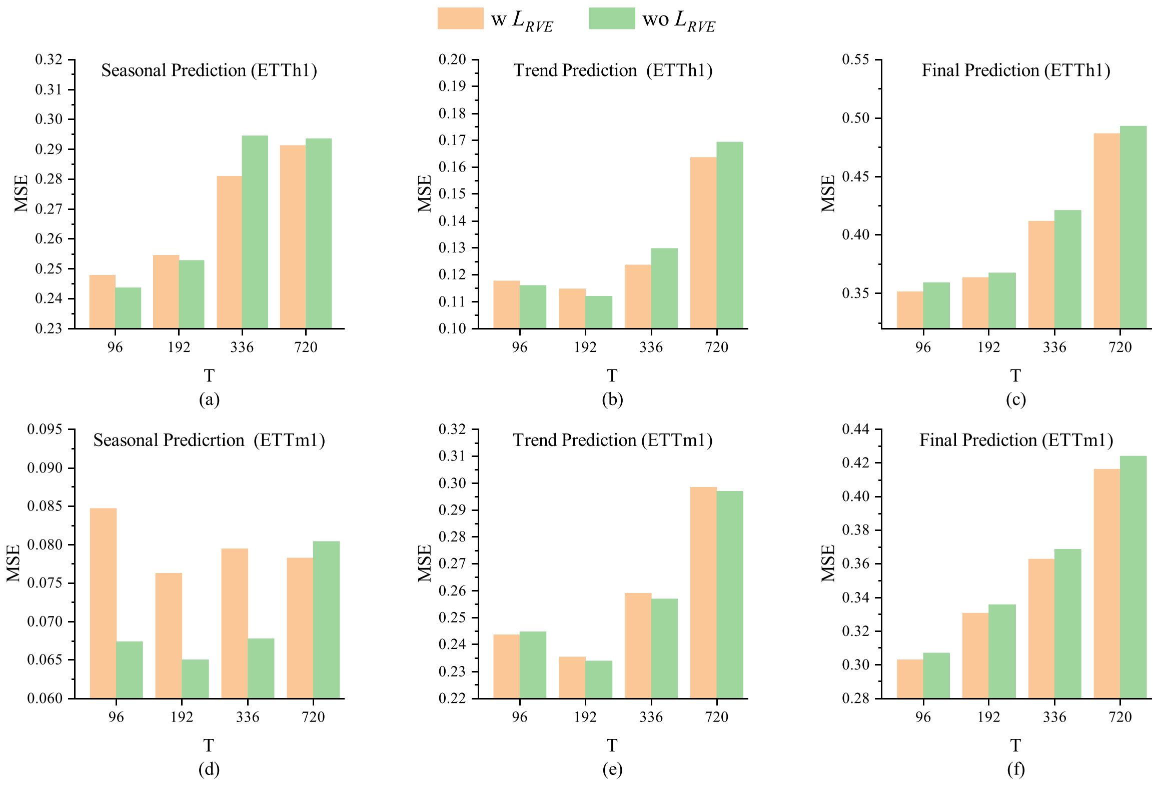


Fig. 5. The MSE scores obtained after ablating the relative value exponential loss function on ETTh1 and ETTm1 datasets. "w L_{RVE}" indicates training with the relative value exponential loss. "wo L_{RVE}" indicates training without the relative value exponential loss.

features in the input sequence, and when the input changes, the corresponding values of the temporal order features change, leading to a decrease in performance.

In addition, as the prediction length increases, the degree of performance degradation decreases for the five models except TOEformer, which indicates that the sensitivity of the other five models to the input sequence decreases over the prediction length. On the other hand, TOEformer shows consistent performance degradation regardless of the prediction length, which suggests that TOEformer effectively utilizes sequence information and maintains sensitivity to the input sequence across different prediction lengths.

4.4. The study of ablation

4.4.1. The ablation study of the relative value exponential loss function

In order to verify the ability of the relative value exponential loss to mitigate the error growth problem, we conducted relative value exponential loss function ablation experiments on the ETTh1 and ETTm1 datasets, where only the loss function was changed in the experimental setup. The rest of the experimental setups were the same as those in Table 2, and the results of the experiments are shown in Fig. 5. In the ETTh1 dataset, when the prediction length is 96 and 192, although the prediction errors of both the seasonal and the trend parts after using L_{RVE} are poorer than without L_{RVE}, the final prediction results are better. Similarly, in the ETTm1 dataset, using L_{RVE} achieves the same effect when the prediction length is 192 and 336. The results provide strong evidence of the ability of L_{RVE} to mitigate the error growth problem. In the remaining four long-time prediction experiments, the overall prediction errors are still better than those without L_{RVE}, although the prediction errors of the seasonal and trend parts are poorer after using L_{RVE}, which suggests that the use of loss L_{RVE} helps the model achieve better prediction results in the framework of seasonal-trend prediction.

In addition, we also applied L_{RVE} loss on DLinear with decomposition architecture, and the experimental results are shown in Table 6. It can be seen that after applying L_{RVE} loss, DLinear improves on all four datasets ETTh1, ETTh2, ETTm1, and ETTm2, which shows that L_{RVE} loss can improve the performance of forecasting models with decomposition architecture.

4.4.2. The ablation study of the network structure

In order to explore the ability of different combinations of network structures to model data, we conducted the ablation study of the network structure on three datasets, ETTh2, Weather, and Exchange, with the same experimental setup as the experiment in Table 2. Five variants were obtained after the ablation of the network structure. Single Linear

Table 6
The experimental results of DLinear after using the L_{RVE} loss. "w L_{RVE} MSE" and "w L_{RVE} MAE" represent the MSE and MAE scores of DLinear with the L_{RVE} applied. "wo L_{RVE} MSE" and "wo L_{RVE} MAE" represent the MSE and MAE scores without L_{RVE}.

Dataset	prediction length	w L_{RVE} MSE	wo L_{RVE} MSE	w L_{RVE} MAE	wo L_{RVE} MAE
ETTh1	96	0.368	0.378	0.389	0.402
	192	0.403	0.415	0.410	0.429
	336	0.436	0.447	0.433	0.469
	720	0.461	0.481	0.479	0.500
ETTh2	96	0.278	0.290	0.336	0.374
	192	0.355	0.375	0.392	0.414
	336	0.430	0.459	0.446	0.452
	720	0.633	0.733	0.558	0.468
ETTm1	96	0.294	0.307	0.336	0.375
	192	0.331	0.347	0.358	0.387
	336	0.368	0.380	0.381	0.411
	720	0.410	0.425	0.408	0.450
ETTm2	96	0.164	0.167	0.253	0.267
	192	0.222	0.223	0.295	0.309
	336	0.282	0.291	0.340	0.351
	720	0.392	0.407	0.414	0.403

removes the seasonal prediction part from our network structure, and the trend prediction network will directly process the data without decomposition. Single Transformer refers to removing the linear trend prediction network from TOEformer and directly performing prediction without sequence decomposition. The Linear trend prediction network replaces the Transformer seasonal prediction network in Double Linear. Double Transformer replaces a Linear trend prediction network with a Transformer seasonal prediction network, and Network_Exchange uses a Transformer network for trend prediction and a Linear network for seasonal prediction. The results of the experiments are shown in Fig. 6. The proposed TOEformer achieves the lowest MSE and MAE scores on four prediction settings of the three datasets, which indicates that TOEformer has the best prediction performance compared to the rest of the structures. The second best is the double linear and the single linear structure due to the inherent modeling capability of linear models for sequences. The worst combination is Network_Exchange. This result is because the Transformer extracts seasonal features efficiently. However, the trend part contains much sequence information in the original sequence, which the Transformer cannot preserve well. Using linear for modeling the seasonal part does not effectively utilize the ability of linear models to preserve the temporal order information.

4.5. The study of hyperparameter

4.5.1. The study of the hyperparameter sensitivity

In order to verify the robustness of the relative value exponential loss to hyperparameters, we selected different α on the four datasets ETTh2, ETTm2, Weather and Exchange for experiments, and the experimental results were shown in Fig. 7. For the ETTh2 dataset, the MSE scores vary greatly with α, indicating that this dataset is more sensitive to α. When α falls between 0.4–0.7, the MSE obtains a smaller value, which indicates that a moderate α helps to optimize the balance between the seasonal and trend errors, while too small or too large a value of α may lead to unstable training. On the ETTm2 dataset, similar to the ETTh2 dataset, the MSE scores obtains smaller values when α takes 0.3–0.7, and reaches the optimization at $\alpha = 0.5$. The MSE scores on the Weather dataset is lower when α takes 0.1–0.3 and then the error increases. The experiment result indicates that on the Weather dataset, the prediction results of the seasonal and trend part have been able to offset each other's errors compared to the ETT datasets, and then a steeper optimization curve can effectively optimize the model's prediction results. For the Exchange dataset, the MSE scores of the model achieves the smaller value when α falls between 0.7–0.9, which indicates that there are more seasonal-trend pairs that cannot offset each other's errors in the prediction results compared with those in the ETT and Weather datasets. And the model needs a larger range of optimization and a more moderate optimization curve in order to optimize the prediction results effectively.

Table 7
The MSE scores are obtained with different weights for the loss function, where α is set to 0.9. The best results are in **bold**.

λ_1		0.1	0.3	0.5	0.7	0.9
λ_2		0.9	0.7	0.5	0.3	0.1
ETTh1	96	0.359	0.354	0.352	0.350	**0.348**
	192	0.374	0.372	**0.364**	0.369	0.369
	336	**0.409**	**0.409**	0.411	0.412	0.417
	720	**0.452**	0.467	0.489	0.527	0.545
ETTh2	96	0.251	**0.249**	0.252	0.249	0.256
	192	0.302	**0.299**	0.306	0.307	0.319
	336	0.288	**0.288**	0.304	0.303	0.323
	720	0.361	0.360	**0.354**	0.365	0.371
ETTm1	96	0.311	0.308	**0.303**	0.311	0.311
	192	0.339	0.337	**0.330**	0.331	0.333
	336	0.372	0.371	**0.363**	0.368	0.366
	720	0.425	0.422	**0.416**	0.416	0.418
ETTm2	96	0.167	0.167	**0.164**	0.166	0.166
	192	0.230	0.229	**0.228**	0.230	0.233
	336	0.283	0.280	**0.264**	0.280	0.277
	720	0.345	0.341	0.339	**0.336**	0.347
Weather	96	0.171	0.169	0.183	0.168	**0.167**
	192	0.198	0.196	0.201	0.192	**0.189**
	336	0.244	0.242	0.240	0.238	**0.236**
	720	0.309	0.307	0.304	0.302	**0.299**
Exchange	96	0.080	0.076	**0.075**	0.081	0.113
	192	0.166	0.155	**0.146**	0.172	0.224
	336	0.292	0.330	**0.252**	0.611	0.782
	720	0.892	0.744	**0.594**	0.714	1.291

4.5.2. The study of the hyperparameter of overall loss

To investigate the impact of different combinations of loss weights (λ_1 and λ_2) on the prediction performance of TOEformer, we conducted experiments with loss weight hyperparameters. The experimental results are shown in Table 7. On the ETTh1 dataset, the model is more sensitive to the choice of hyperparameters, and the optimal results on different prediction Settings are obtained from different combinations of hyperparameters. Secondly, the prediction results obtained by different hyperparameters in each prediction setting also fluctuated greatly. This may be due to the fact that ETTh1 implies more non-stationary components (such as abrupt trends) than the other three ETT datasets, resulting in increased gradient direction conflict between MSE and L_{RVE} during model training. In ETTh2 dataset, $\lambda_1 = 0.3$ and $\lambda_2 = 0.7$ perform best on multiple steps, indicating that the constraint effect of L_{RVE} is more significant on this dataset. For ETTm1, ETTm2, and Exchange datasets, $\lambda_1 = 0.5$ and $\lambda_2 = 0.5$ worked well across all prediction settings, suggesting that appropriate L_{RVE} weights contribute to long-term prediction stability. In addition, Exchange datasets are also sensitive to weight configuration. For example, the MSE (0.252) with $\lambda_1 = 0.5$ was significantly better than the other combinations in the 336 step forecast, but the performance deteriorated sharply at $\lambda_1 = 0.9$ (MSE=0.782), indicating that the binding effect of L_{RVE} loss is indispensable in complex data scenarios. In the Weather dataset, the lowest MSE is obtained when $\lambda_1 = 0.9$ and $\lambda_2 = 0.1$, indicating that the MSE-dominated optimization strategy is suitable for this dataset. In general, the weight adjustment of MSE and L_{RVE} has a significant impact on the prediction effect of different datasets. Reasonable weight allocation can effectively improve model performance.

5. Conclusion and future work

This paper presents a temporal order enhanced Transformer for time series forecasting tasks, effectively utilizing temporal order information

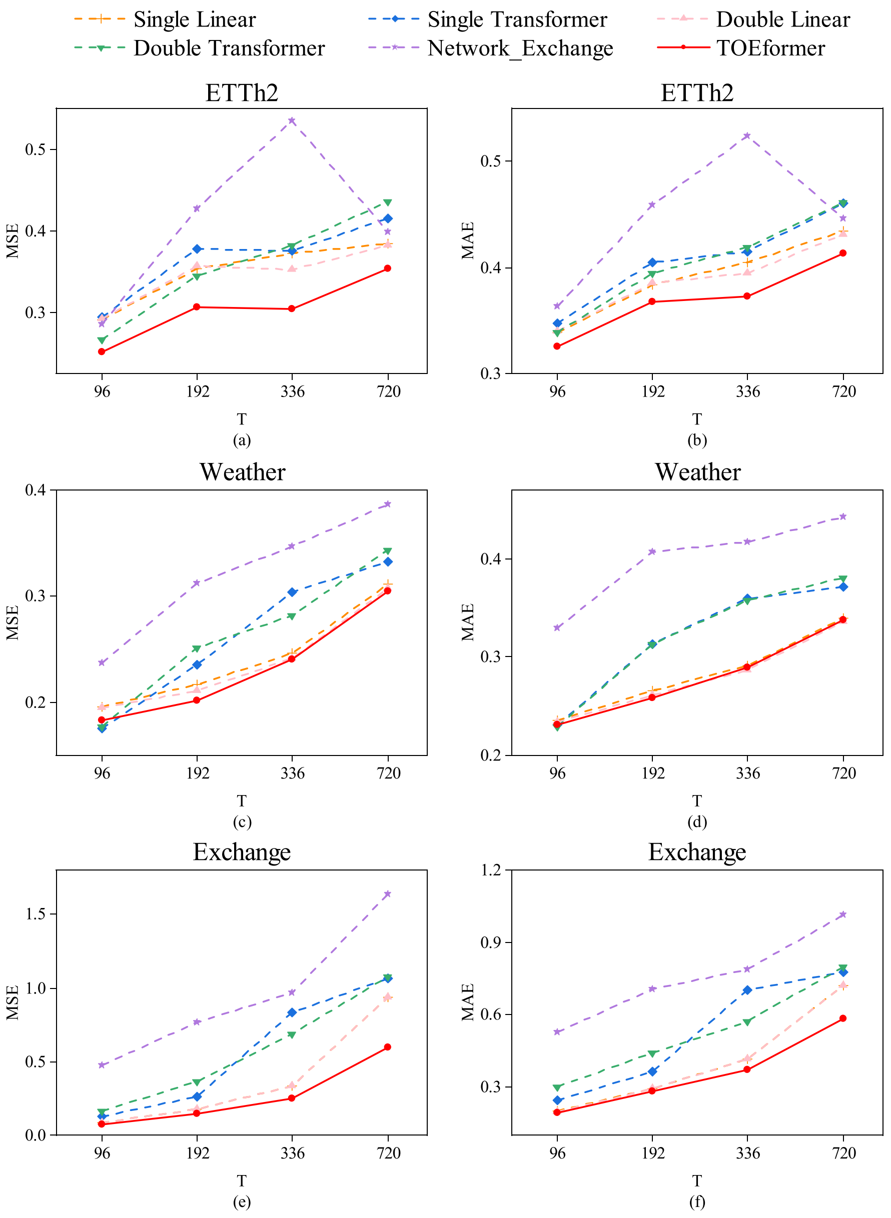


Fig. 6. Network structure comparison experiments. The MSE scores: (a), (c), and (e). The MAE scores: (b), (d), and (f). The prediction length $T \in \{96, 192, 336, 720\}$, and the lookback window length is the same as the prediction length.

and achieving state-of-the-art performance on multiple public datasets. We integrate Linear networks with Transformer networks to preserve temporal order information, and our experiments demonstrate the effectiveness of this approach in maintaining this crucial information. The proposed relative value index loss function encourages the model to generate periodic-trend pairs that can offset errors, thereby mitigating the accumulative effect of errors. Finally, extensive experiments on multiple benchmark datasets show that our proposed model produces the most advanced forecasting results in over 85% of multivariate prediction settings compared to several advanced baselines.

Due to the core part of the model still relies on the attention mechanism, and the computational complexity of the mechanism is usually the square order of the length of the input sequence $\mathcal{O}(L^2)$, which increases the computational cost significantly when dealing with long sequences or large data sets. As a result, TOEformer may face bottlenecks in efficiency and computing resources when dealing with extremely long time series or resource-constrained real-time application scenarios.

In addition, this work also has practical significance: By solving Transformer's shortcomings in temporal order information, TOEformer model can improve accuracy in various forecasting tasks, such as financial data forecasting, climate data forecasting and power transformer data forecasting; This study provides a new idea for how to effectively combine linear model and deep learning model in time series prediction, and promotes the application of hybrid structure in other complex time series problems.

In future work, we will reduce the sensitivity of the model to hyperparameters, thus improving its robustness and generalization ability. In addition, one of the subsequent research directions of this work is to integrate with natural language processing models to enhance causal reasoning and interpretability. With the development of large language models, combining time series data with textual data can enable causal analysis of events in a time series. For example, in finance, models can combine news events with market time series for joint modeling, thus enhancing the accuracy of market volatility prediction. This integration can be done using a multimodal learning approach, where text and time series data are fed into the same model,

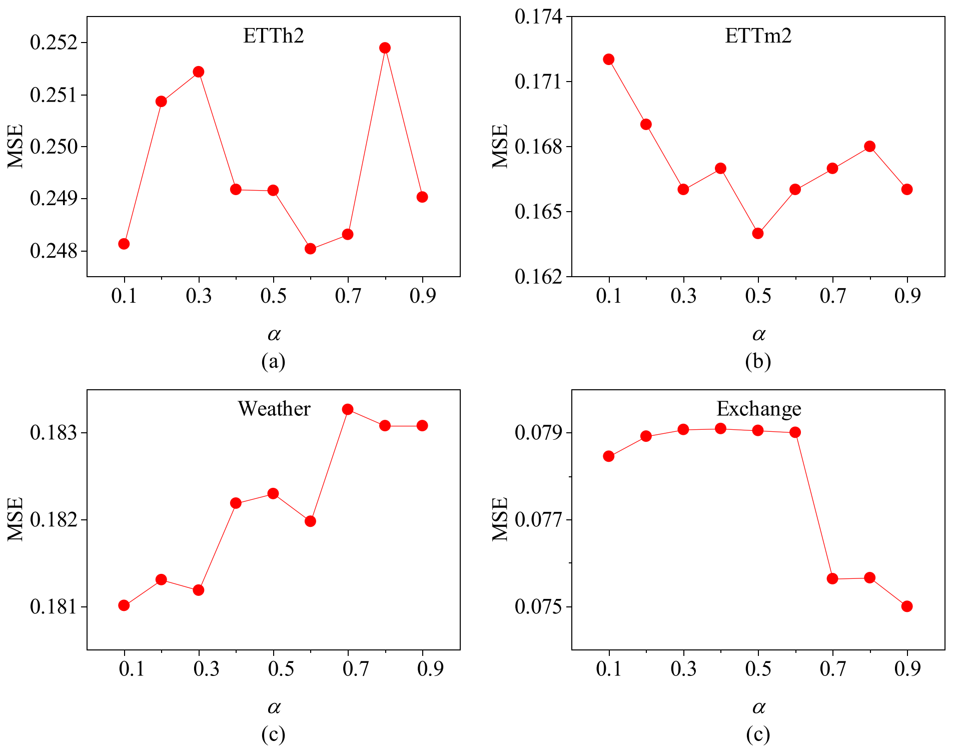


Fig. 7. MSE scores for relative value exponential loss with different bases α, $\alpha \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$, $\lambda_1 = 0.5$, $\lambda_2 = 0.5$,lookback window is 96, and prediction length is 96.

or data from different modalities can be modeled interactively through an attention mechanism to obtain more correlated information.

CRediT authorship contribution statement

Qiang Li: Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Jiwei Qin:** Writing – review & editing, Writing – original draft, Funding acquisition. **Dacheng Wang:** Writing – review & editing. **Xizhong Qin:** Writing – review & editing. **Daishun Cui:** Writing – review & editing, Visualization. **Jiachen Xie:** Visualization, Software. **Dezhi Sun:** Visualization, Software, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by the Key Research and Development Program of Xinjiang Uygur Autonomous Region: Research and Development of Key Technologies for Calculation, Measurement, and Digitized Management of Carbon Emission and Carbon Sink Indicators in the Energy Sector, China (NO. 2022B01010), Outstanding Doctoral Student Innovation Project of Xinjiang University, China (NO. XJU2024BS085).

Data availability

The six publicly available datasets used in this study are all available for self-download online.

References

AlKheder, S., Almusalam, A., 2022. Forecasting of carbon dioxide emissions from power plants in Kuwait using United States Environmental Protection Agency, Intergovernmental panel on climate change, and machine learning methods. Renew. Energy 191, 819–827.

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

Ariyo, A.A., Adewumi, A.O., Ayo, C.K., 2014. Stock price prediction using the ARIMA model. In: 2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation. IEEE, pp. 106–112.

Bai, S., Kolter, J.Z., Koltun, V., An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. Univers. Lang. Model. Fine- Tuning Text Classif.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al., 2020. Language models are few-shot learners. Adv. Neural Inf. Process. Syst. 33, 1877–1901.

Cai, W., Liang, Y., Liu, X., Feng, J., Wu, Y., 2024. Msgnet: Learning multi-scale inter-series correlations for multivariate time series forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, (10), pp. 11141–11149.

Chen, S.A., Li, C.L., Arik, S.O., Yoder, N.C., Pfister, T., 2023. TSMixer: An all-MLP architecture for time series forecast-ing. Trans. Mach. Learn. Res. URL: https://openreview.net/forum?id=wbpxTuXgm0.

Dai, S., Niu, D., Han, Y., 2018. Forecasting of energy-related CO2 emissions in China based on GM(1,1) and least squares support vector machine optimized by modified shuffled frog leaping algorithm for sustainability. Sustainability 10 (4), http://dx.doi.org/10.3390/su10040958, URL: https://www.mdpi.com/2071-1050/10/4/958.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N., 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations. URL: https://openreview.net/forum?id=YicbFdNTTy.

Ekambaram, V., Jati, A., Nguyen, N., Sinthong, P., Kalagnanam, J., 2023. TSMixer: Lightweight MLP-mixer model for multivariate time series forecasting. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. KDD '23, Association for Computing Machinery, New York, NY, USA, pp. 459–469. http://dx.doi.org/10.1145/3580305.3599533.

Fan, W., Zheng, S., Yi, X., Cao, W., Fu, Y., Bian, J., Liu, T.Y., 2022. DEPTS: Deep expansion learning for periodic time series forecasting. In: International Conference on Learning Representations.

Kalyan, K.S., Rajasekharan, A., Sangeetha, S., 2022. AMMU: a survey of transformer-based biomedical pretrained language models. J. Biomed. Informatics 126, 103982.

Khan, S., Naseer, M., Hayat, M., Zamir, S.W., Khan, F.S., Shah, M., 2022. Transformers in vision: A survey. ACM Comput. Surv. 54 (10s), 1–41.

Kitaev, N., Kaiser, L., Levskaya, A., 2020. Reformer: the efficient transformer. In: International Conference on Learning Representations. https://openreview.net/forum?id=rkgNKkHtvB.

Lai, G., Chang, W.C., Yang, Y., Liu, H., 2018. Modeling long-and short-term temporal patterns with deep neural networks. In: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval. pp. 95–104.

Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.X., Yan, X., 2019. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. Adv. Neural Inf. Process. Syst. 32.

Liu, Y., Hu, T., Zhang, H., Wu, H., Wang, S., Ma, L., Long, M., 2024. Itransformer: Inverted transformers are effective for time series forecasting. In: The Twelfth International Conference on Learning Representations. URL: https://openreview.net/forum?id=JePfAI8fah.

Liu, Y., Li, C., Wang, J., Long, M., 2023. Koopa: Learning non-stationary time series dynamics with koopman predictors. In: Thirty-Seventh Conference on Neural Information Processing Systems. URL: https://openreview.net/forum?id=A4zzxu82a7.

Liu, Y., Wu, H., Wang, J., Long, M., 2022a. Non-stationary transformers: Exploring the stationarity in time series forecasting. In: Oh, A.H., Agarwal, A., Belgrave, D., Cho, K. (Eds.), Advances in Neural Information Processing Systems. URL: https://openreview.net/forum?id=ucNDIDRNjjv.

Liu, S., Yu, H., Liao, C., Li, J., Lin, W., Liu, A.X., Dustdar, S., 2021. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In: International Conference on Learning Representations.

Liu, M., Zeng, A., Chen, M., Xu, Z., Lai, Q., Ma, L., Xu, Q., 2022b. SCINet: Time series modeling and forecasting with sample convolution and interaction. In: Oh, A.H., Agarwal, A., Belgrave, D., Cho, K. (Eds.), Advances in Neural Information Processing Systems. URL: https://openreview.net/forum?id=AyajSjTAzmg.

Lotfalipour, M.R., Falahi, M.A., Bastam, M., 2013. Prediction of CO2 emissions in Iran using grey and ARIMA models. Int. J. Energy Econ. Policy 3 (3), 229–237.

Ni, Z., Yu, H., Liu, S., Li, J., Lin, W., 2023. Basisformer: Attention-based time series forecasting with learnable and interpretable basis. In: Advances in Neural Information Processing Systems.

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

Oreshkin, B.N., Carpov, D., Chapados, N., Bengio, Y., 2020. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. In: International Conference on Learning Representations. URL: https://openreview.net/forum?id=r1ecqn4YwB.

Qin, Y., Song, D., Chen, H., Cheng, W., Jiang, G., Cottrell, G.W., 2017. A dual-stage attention-based recurrent neural network for time series prediction. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence. International Joint Conferences on Artificial Intelligence Organization, pp. 2627–2633.

Rao, Y., Zhao, W., Zhu, Z., Lu, J., Zhou, J., 2021. Global filter networks for image classification. Adv. Neural Inf. Process. Syst. 34, 980–993.

Wang, S., Wu, H., Shi, X., Hu, T., Luo, H., Ma, L., Zhang, J.Y., Zhou, J., 2024. TimeMixer: Decomposable multiscale mixing for time series forecasting. In: International Conference on Learning Representations. ICLR.

Wu, H., Hu, T., Liu, Y., Zhou, H., Wang, J., Long, M., 2023. TimesNet: Temporal 2D-variation modeling for general time series analysis. In: International Conference on Learning Representations.

Wu, H., Xu, J., Wang, J., Long, M., 2021. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. Adv. Neural Inf. Process. Syst. 34, 22419–22430.

Zeng, A., Chen, M., Zhang, L., Xu, Q., 2023. Are transformers effective for time series forecasting? In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, (9), pp. 11121–11128.

Zhao, Y., Ma, Z., Zhou, T., Ye, M., Sun, L., Qian, Y., 2023. GCformer: an efficient solution for accurate and scalable long-term multivariate time series forecasting. In: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management. pp. 3464–3473.

Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., Jin, R., 2022. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In: International Conference on Machine Learning. PMLR, pp. 27268–27286.

Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., Zhang, W., 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, (12), pp. 11106–11115.