Applied Intelligence (2024) 54:10072–10083
https://doi.org/10.1007/s10489-024-05701-w

Fine-scale deep learning model for time series forecasting

Yuwei Chen[1] · **Wenjing Jia**[1] · **Qiang Wu**[1]

Accepted: 23 July 2024 / Published online: 6 August 2024

Abstract
Time series data, characterized by large volumes and wide-ranging applications, requires accurate predictions of future values based on historical data. Recent advancements in deep learning models, particularly in the field of time series forecasting, have shown promising results by leveraging neural networks to capture complex patterns and dependencies. However, existing models often overlook the influence of short-term cyclical patterns in the time series, leading to a lag in capturing changes and accurately tracking fluctuations in forecast data. To overcome this limitation, this paper introduces a new method that utilizes an interpolation technique to create a fine-scaled representation of the cyclical pattern, thereby alleviating the impact of the irregularity in the cyclical component and hence enhancing prediction accuracy. The proposed method is presented along with evaluation metrics and loss functions suitable for time series forecasting. Experiment results on benchmark datasets demonstrate the effectiveness of the proposed approach in effectively capturing cyclical patterns and improving prediction accuracy.

Keywords Time series prediction · Decomposition · Interpolation · Cyclical pattern

1 Introduction

Due to the large volumes of data generated in the modern world every day, time series data has become one of the most critical forms of data. Time series forecasting involves predicting values at future points based on historical data. It has wide-ranging applications in finance [1], healthcare [2], energy [3] and weather forecasting [4]. In recent years, significant advances have been made in the field of time series forecasting [5], particularly with the advancement of deep learning techniques. These models leverage the power of neural networks to capture complex patterns and dependencies in time series data, and have achieved accurate predictions.

One key aspect of time series forecasting is the decomposition of the time series into its underlying components, such as trend, seasonality, and cyclical patterns. Traditional decomposition methods, widely used in the early 20th century, formed the basis for many subsequent algorithms. These methods assume that the cyclical component remains constant within each cycle and often employ some simple techniques such as moving average to separate the trend-cyclical and seasonal components.

Some recent approaches, such as Autoformer [6], decomposed the time sequence into trend-cyclical and seasonal components, and have significantly improved forecasting accuracy for datasets exhibiting strong seasonality. However, they often overlook the influence of cyclical patterns on the time series due to the lack of handling of the trend-cyclical component after decomposition. As a result, they exclude this factor from the scope of predictions. Such an approach can potentially result in delays in capturing changes and unable to track fluctuations in forecast data accurately. This paper proposes a new method that incorporates interpolation techniques to capture fine-scale cyclical information. Interpolation techniques help fill in missing or irregularly spaced data points in the cyclical sequence, creating a fine-scale representation of the cyclical pattern. It also smooths out the cyclical sequence, reducing the impact of short-term fluctuations and noise, thereby enabling better identification of underlying trends and patterns.

✉ Yuwei Chen
Yuwei.Chen@student.uts.edu.au

Wenjing Jia
Wenjing.Jia@uts.edu.au

Qiang Wu
Qiang.Wu@uts.edu.au

[1] University of Technology Sydney, 2007, NSW, Ultimo, Australia

Moreover, the existing time series prediction models mostly employ some intuitive evaluation metrics and loss functions to assess their performance and to guide their training, including Mean Squared Error (MSE) and Mean Absolute Error (MAE). In practice, we have found that when dealing with data that has high sparsity and strong uncertainty, such intuitive evaluation metrics can significantly reduce the reliability of the model. There may be cases where the model has a very low MSE but performs poorly overall. In scenarios with strong uncertainty, where the ground truth is inherently noisy or uncertain, MSE can not effectively depict the model's performance. The squared error loss function assumes that errors are normally distributed and of equal importance across the data points, which may not be the case when dealing with data with great uncertainty. Consequently, the model may optimize for reducing errors in noisy data points at the expense of overall predictive accuracy. Therefore, in our proposed method, we adopt Dynamic Time Warping (DTW) as a more effective loss function to calculate the differences between the predicted sequence and the ground truth. After comparing the performance of DTW and MSE, we found that the training results using DTW yielded predicted sequences that more closely resembled the original sequence in shape.

In summary, our key contributions are as follows: 1) we proposed a novel approach of adopting a temporal interpolation technique to further improve the precision of time series forecasting by separately predicting the Trend-cyclic sequences; 2) we proposed an enhanced method for time series prediction that leverages the power of DTW as an effective loss function.

The rest of the paper begins by summarizing the existing related work in Section 2, focusing on long-sequence time-series forecasting and decomposition in time series. It then introduces our improved method for time series prediction in Section 3 and presents the proposed temporal interpolation technique, along with evaluation metrics and loss functions that are more suitable for time series forecasting. The experiment design and results are presented in Section 4. Finally, the paper concludes in Section 5.

2 Related work

2.1 Long sequence time-series forecasting

Long sequence time-series forecasting (LSTF) has always been a hot topic in time series analysis. In most studies, long sequence time-series problems typically do not handle a window length larger than 100 [5]. However, in practical applications, time series data is characterized by "massive, high-frequency, and long-term prediction [7]." Combining long sequence prediction to handle these data characteristics is one of the key factors in improving the performance of various time series prediction algorithms.

In the development of time series forecasting, existing solutions can generally be classified into three categories, traditional statistical models, machine learning models, and deep learning models. Most traditional approaches rely on statistical models, such as the commonly used mean regression [8], ARIMA [9] (autoregressive integrated moving average), and exponential smoothing methods [10]. Models in this category, like the ARIMA model, are limited to regression on univariate historical data and cannot handle multivariate scenarios [11]. Although traditional statistical-based methods for time series forecasting have advantages such as low complexity and fast computation speed, they have limitations in handling complex real-world problems. These limitations have prompted the development of machine learning methods that aim to address these challenges and improve the accuracy and flexibility of time series forecasting models.

Machine learning models treat time series forecasting as a regression problem, transforming the temporal problem into supervised learning. The commonly used methods can be classified into two categories: tree-based models and neural network models. Tree-based models, such as LightGBM and XGBoost, are capable of addressing a wide range of complex time series forecasting tasks [12]. They support complex data modeling, handle multivariate regression, and are effective in solving nonlinear problems. However, a major drawback of these models is that feature extraction can be laborious and time-consuming. In recent years, the widespread application of deep learning models in time series has also proven their effectiveness, as shown in works [13–15]. Apart from the earliest CNN models, another category of neural network models incorporated many widely used approaches in natural language processing (NLP). Examples include specialized models like LSTM and GRU, which are designed specifically to address sequence prediction problems [16].

Recently, Transformer models [6, 17–19], originally designed for NLP tasks, have also been extensively employed in time series forecasting problems. Two of the most well-known works based on the vanilla Transformer [17] in the field of time series forecasting are the Informer model [18] and the Autoformer model [6].

The Informer model [18] restructures the Transformer model to address long-term and short-term forecasting challenges, achieving remarkable results and sparking enthusiasm for research in this direction. It employs a novel attention mechanism, a temporal attention module, and a convolutional feedforward network to capture the input sequence's long- and short-term dependencies. Shortly after that, the researchers from Tsinghua University proposed the Autoformer model [6], which builds on the previous work on Informer's design and significantly outperforms previous models on the same forecasting task. Autoformer achieves a

38% relative improvement in performance, demonstrating its superiority in handling long sequence time series data. Autoformer introduces a Decomposition Architecture to extract predictable components from complex time patterns. This separation creates multiple sub-series, each representing a component with simple patterns. Consequently, Autoformer captures crucial features more easily, leading to improved prediction accuracy.

However, along with the mentioned improvements, the above-mentioned models also have some common weaknesses. As highlighted in [20], a common limitation among deep learning models is their inability to directly handle missing values, necessitating the use of imputation or interpolation techniques to facilitate the training process. Although recent advancements in state-of-the-art methodologies such as [21] have begun to overcome this challenge, introducing models capable of directly processing incomplete data, the adoption of such approaches is not yet widespread across all deep learning applications, and a significant number of models continue to rely on traditional methods to address missing values in datasets.

In this paper, we extract the cyclical component from the input time series and then propose a temporal interpolation technique to the fine trend-cyclical component of the input signals, to capture the fine cyclical patterns carried by the signals. By using interpolation, the length of the time series is proportionally extended, resulting in a smoother representation of the trend's variation. It also helps reduce the impact of short-term fluctuations and noise, and enables better identification of underlying trends and patterns. Furthermore, algorithmic improvements are proposed in this article to compare time series of variable size that are robust to shifts or dilation over time.

Next, we will provide a more detailed explanation of the current practices in decomposition, and the decomposition algorithms used in the current models.

2.2 Decomposition in time series

2.2.1 Classical decomposition

The traditional time series decomposition method started in the 1920s and was widely used until the 1950s. The steps of the classic algorithm are relatively simple. It is also the basis for many other decomposition algorithms. In our proposed method, we adopt the additive model as a common basis for decomposition. A time series y_t that follows an additive model can be written as:

$$y_t = S(t) + T(t) + R(t), \tag{1}$$

where $S(t)$, $T(t)$, and $R(t)$ represent the seasonal component, trend-cyclical component, and remainder component, respectively. The classic decomposition method assumes that the cyclical component is the same within each cycle. Most classic algorithms adopt the concept of moving averages, where the prediction $\hat{T}_t$ is computed as:

$$\hat{T}_t = \frac{1}{m} \sum_{j=-k}^{k} y_{t+j} \tag{2}$$

In this equation, $\hat{T}_t$ represents the smoothed or averaged value at time point t in the time series. The parameter k determines the window size, indicating the number of time steps to look back and forward. m represents the total number of values within the window. The equation calculates $\hat{T}_t$ by averaging the values of the time series over times from k steps before t to k steps after t.

The later-developed SEATS decomposition algorithm stands for "Seasonal Extraction in ARIMA Time Series". This method was developed by the Bank of Spain and is now widely used in government departments around the world [22, 23]. However, this algorithm is only for quarterly and monthly data. Therefore, when the designers of Autoformer built the Autoformer framework [6], they followed the Transformer framework design and added a new decomposition block to extract the intrinsic complex time series trends in the model's hidden state. Subsequent variants of the Transformer also followed this design of Autoformer, retaining the decomposition part and adding time embeddings to handle finer time information, such as daily, hourly or per minute. The variants include recent works from [19, 24].

2.2.2 Autoformer decomposition

In Autoformer, the Series Decomp Block utilizes traditional decomposition operations to separate a time series into two components: trend-cyclical and seasonal parts. The trend-cyclical component captures short-term fluctuations and overall trends, while the seasonal component reflects long-term seasonal variations. As shown in Fig. 1, the "Seasonal" component represents the influence of a specific moment within a year on the time series and is highly correlated with time embedding. It denotes known and fixed-frequency variations. On the other hand, the "Trend" component represents long-term trends, which may not necessarily be linear. The "cyclical" component represents short-term, non-fixed-frequency oscillations. These models often treat trend and cyclical components together because it is challenging to algorithmically differentiate between them. Therefore, the decomp block is used to separate the original time series into long-term and short-term variations.

However, as shown in Fig. 2, these approaches do not include short-term variations in the scope of prediction. Instead, during each pass through the series decomp module,

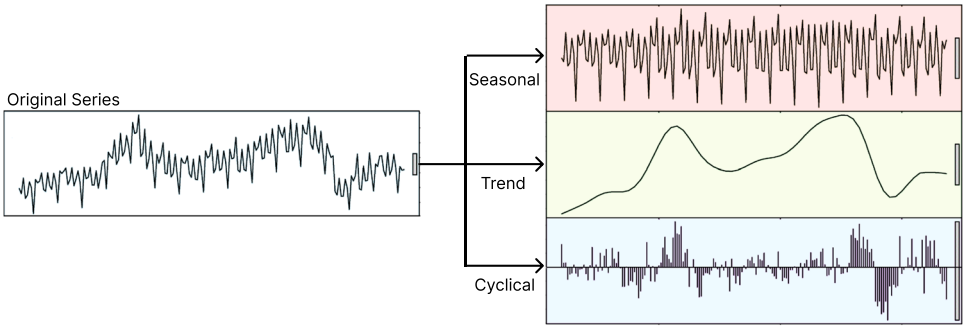

Fig. 1 A time series can be decomposed into a number of components: seasonality, trend, cyclical and error (irregular)

they add the mean of the previous time's trend-cyclical to the predicted seasonal sequence. In other words, these models overlook the influence of cyclical patterns on the time series and exclude this factor from the prediction scope. This approach often results in a lag in the changes in forecast results, failing to accurately track fluctuations in forecast data. To address this issue, we propose a new method that captures more cyclical information by using interpolation and a more effective loss evaluation function.

3 The proposed method

3.1 Motivation

In most time series forecasting or classification tasks, time series are typically classified into four types [25]:

No trend, no seasonality This type of time series does not exhibit any significant trend over the long term and does not have a fixed seasonal pattern. Data points show random fluctuations on the time axis without any noticeable trend or periodic changes. They can be modelled and forecast using simple methods like averaging or moving averages.

Trend, no seasonality This type of time series shows a clear trend but does not have a fixed seasonal pattern. Data points exhibit a gradual increase or decrease trend over time without any fixed periodicity.

Seasonality, no trend This type of time series has a fixed seasonal pattern but does not show any significant trend over time. Data points exhibit repetitive seasonal patterns, such as yearly, monthly, or weekly.

Seasonality and trend This type of time series exhibits both a clear seasonal pattern and a long-term trend. Data points show repetitive seasonal patterns along with a gradual increase or decrease trend.

Specifically, directly predicting cyclical subsequences is challenging. It is difficult to learn its frequency in a straightforward manner when cyclical patterns exhibit significant irregularity. Some cyclical patterns may have very limited samples during short periods, making it challenging to capture their trends effectively.

Similar to the traditional ETS (Error, Trend, Seasonality) models [26, 27], Autoformer [6], and other variants of the Transformer models, primarily focus on considering seasonal subsequences and overlook trend-cyclical subsequences. As mentioned in [28], they directly ignored the trend-cyclical

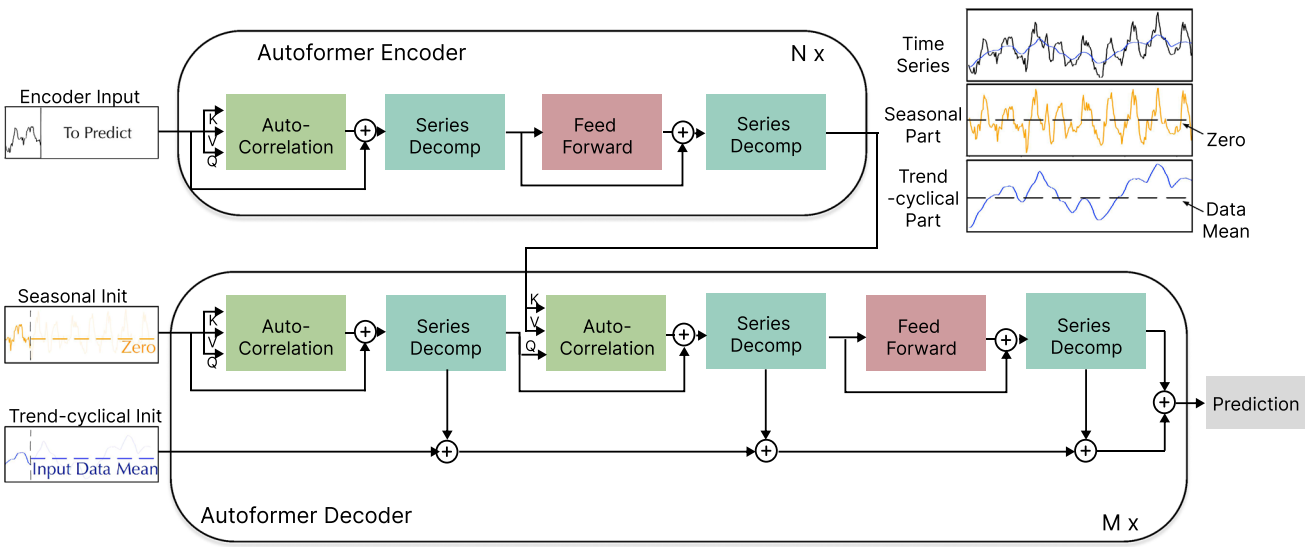

Fig. 2 Autoformer model structure [6]

component because the model they proposed did not perform well in predicting this part of the subsequences. While in reality, although cyclical patterns are distinct from seasonal patterns, they still contain valuable time-related information that is worth predicting [29]. Unlike predicting seasonal patterns, forecasting cyclic type sequences often requires reference to multidimensional features or spatial attributes due to the lack of regular timing information.

In our proposed method, we adopt the decomposition approach of Autoformer [6] and decompose the input time series into two components: trend-cyclical and seasonal parts, and then depict the fine-scale cyclical patterns in the trend-cyclical component. Furthermore, to tackle the shortage of sample points to depict significant random cyclical patterns, we propose augmenting the time information of the cyclical component with a fine-scale interpolation mechanism to smooth the original cyclical sequence. This enables us to capture trends within cyclical patterns better. Additionally, through linear interpolation, we can mitigate the impact of missing or irregularly spaced data points in the time series, thereby improving the effectiveness of predictions.

3.2 Overall model structure

As shown in Fig. 3, our proposed method adopts a Transformer-like encoding-decoding architecture, inherited from the baseline model Autoformer [6]. Each layer consists of three modules: Auto-Correlation, Series Decomp, and Feed Forward, and uses residual structures.

The decoder adopts a dual-path processing mode, where the upper branch handles the seasonal part, and the lower branch handles the trend-cyclical part. We adopt the Series Decomposition Block from Autoformer [6], which utilizes traditional decomposition operations to separate a time series into two components: trend-cyclical and seasonal parts. As shown in Fig. 3, the Seasonal init, serving as the input to the decoder's upper branch, is derived from the output of the Series Decomp block in the encoder. It is composed of the seasonal component obtained after decomposing the original time series. To ensure consistent sequence length, this process involves sampling where the latter half of the seasonal component obtained after decomposition is concatenated with a sequence of zeros, serving as a placeholder.

In the upper branch, the Auto-Correlation block is first used to extract the intrinsic temporal dependencies of future forecasting states. This block analyzes the time series data to identify and capture the relationships between different time steps, providing crucial information for accurate forecasting.

Next, the output of the Auto-Correlation block, along with the encoder's outputs, undergoes another round of series decomposition using a Series Decomp block. This block is responsible for extracting information from historical sequences with high-order temporal dependencies, contributing to a more comprehensive understanding of the time series data. The output of the Series Decomp block is then used as the query input for the subsequent Auto-Correlation block, with the encoder's output serving as the key and value inputs. This step enables the model to refine its understanding of temporal dependencies and capture intricate patterns in the data. Finally, the output of the Auto-Correlation block, along with the processed encoder outputs, passes through the Feed Forward layer for prediction. The Feed Forward layer is responsible for transforming the extracted features into predictions, contributing to the final forecasting outcome.

In contrast, the lower branch utilizes weighted addition to combine the outputs of each sub-layer from the upper branch. The proposed interpolation module is added to the lower branch. That is, after the original time series is decomposed into seasonal and trend-cyclical components, the trend-cyclical sub-sequence will pass through the interpo-

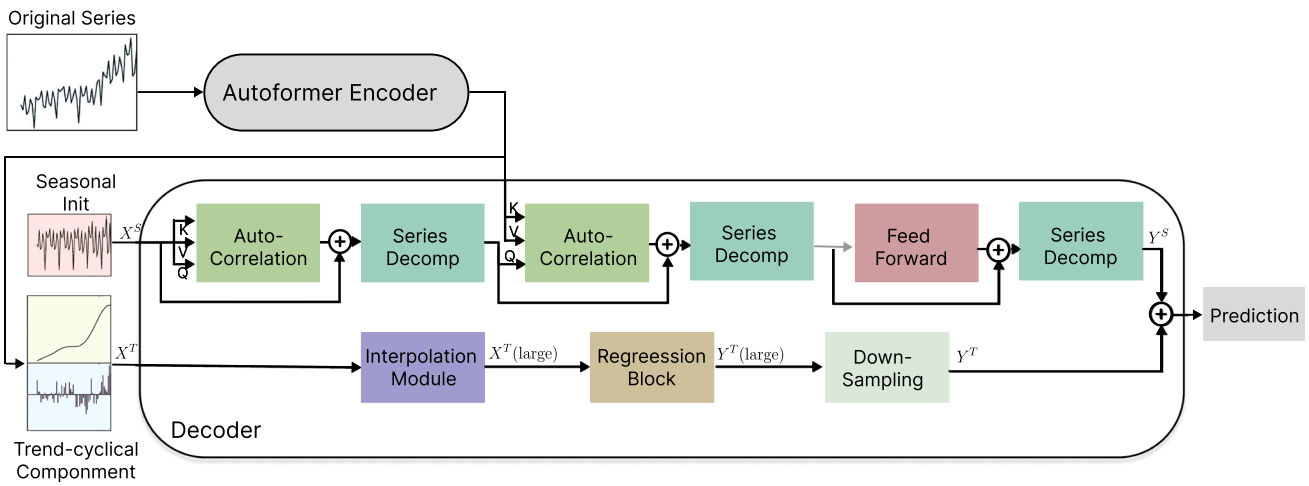

Fig. 3 The structure of the proposed model

lation module to smooth short-term oscillations. Then, after completing the prediction through a linear Regression block, it will undergo downsampling to match the length of the seasonal sub-sequence for the final result summation.

3.3 Interpolation

Denote the input series as X, the trend-cyclical component from decomposition as X^T, and the Seasonal fluctuations of the sequence as X^S. Here, X^S retains the seasonal smoothness of the sequence after subtracting the short-term fluctuations, as:

$$X^S = X - X^T \tag{3}$$

Due to the difficulty in distinguishing between them, X^T may also contain shorter-term cyclical information X^C. X^T can be obtained by storing the average value of each sliding window as:

$$X^T = \text{AvgPool}(\text{Padding}(X)). \tag{4}$$

As shown in Fig. 4, in our approach, X^S still goes through the seasonal prediction block and predicts the corresponding Y^S. Nevertheless, X^T will undergo linear interpolation, where each input sequence is interpolated based on predetermined scales. Due to the lack of seasonal temporal information in the Trend-cyclic sequence, we conduct interpolation before its prediction to smooth out short-term oscillations and achieve finer-scale prediction of the Trend-cyclic sequence. In addition, when there are missing readings in some real-world datasets, our method of filling the missing gaps using interpolation can also improve prediction accuracy. These missing readings can occur due to various factors such as sensor malfunctions or devices going offline. In our experiment, a scale parameter of 5 was chosen, meaning that there will be five interpolated points between every two original points t_1 and t_2. The detailed interpolation process is described below.

Specifically, let t^* represent the target time points for interpolation, t_1 represents the known time point before the target time points, and t_2 represents the known time point after the target point. Let $x^T_{t^*}$ represent the output value of the interpolation, and $x^T_{t_1}$ and $x^T_{t_2}$ represent the values at the first and second known time points, respectively. The interpolation can be expressed as:

$$x^T_{t^*} = x^T_{t_1} + (t^* - t_1) \cdot \left(\frac{x^T_{t_2} - x^T_{t_1}}{t_2 - t_1} \right). \tag{5}$$

After the interpolation is completed, X^T becomes a longer sequence, denoted as $X^T(\text{large})$, which will then be fed into a regression block to complete the prediction for the trend-cyclical sequence.

Following the approach in [24], we employ a linear regression strategy to make predictions about the trend-cyclical sequence, where

$$Y^T(\text{large}) = \text{Regression}(X^T(\text{large})) \tag{6}$$

Up to this point, $Y^T(\text{large})$ does not match the original sequence length. Therefore, it will be downsampled back to the original length using a down-sampling block, resulting in Y^T. For a sequence $Y^T(\text{large}) = \{y_1, y_2, ..., y_N\}$ of length N and another sequence Y^T of length M where $M < N$ indicating the target length, $Y^T(\text{large})$ is downsampled to match the target length M as:

$$y'_i = \frac{1}{D} \sum_{j=(i-1)D+1}^{iD} y_j, \tag{7}$$

where $D = \frac{N}{M}$ ensures the length of the downsampled sequence matches M.

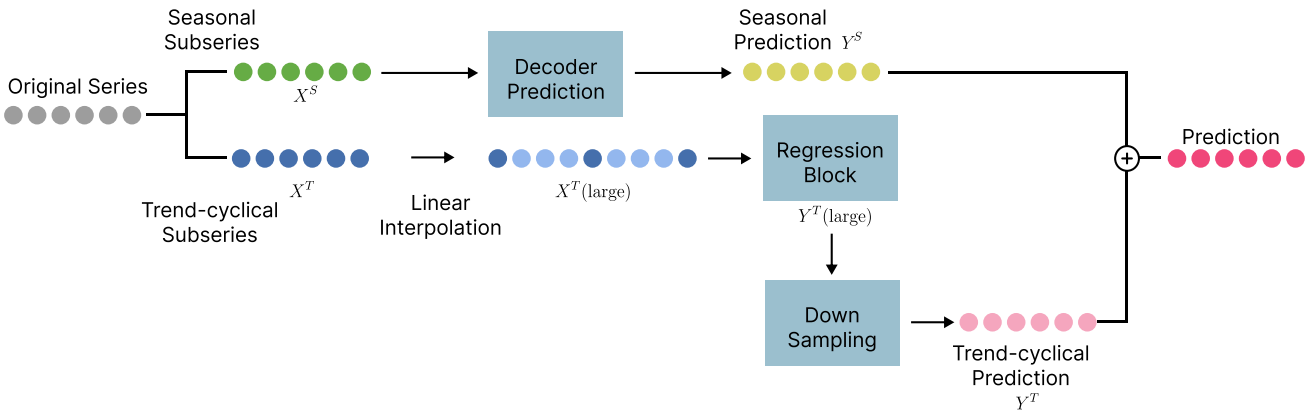

Fig. 4 The proposed interpolation structure

This approach averages every D consecutive points from Y^T (large) to produce the shorter sequence Y^T to ensure that Y^T has the same length as Y^S. Then, we combine the interpolated sequence Y^T with Y^S as the target sequence and compare it to the ground truth sequence Y^{gt}.

3.4 Loss functions

In addition to using interpolation to improve the decomposition module, our method also explores a more suitable loss function for time series which emphasizes the inclusion of cyclical information.

The existing time series prediction models mentioned above employ several intuitive evaluation metrics and loss functions to assess their performance and to guide their training. The most commonly used evaluation metrics and loss functions include Mean Squared Error (MSE) and Mean Absolute Error (MAE), which are calculated as:

$$MSE = \frac{1}{n}\sum_{t=1}^{n}(y_t - \hat{y}_t)^2 \tag{8}$$

and

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|y_t - \hat{y}_t| \tag{9}$$

Here, n represents the number of samples, indicating the quantity of samples used when evaluating the model's performance, y_t denotes the true value, representing the actual observation or label of the sample t, and $\hat{y}_t$ stands for the predicted value, representing the model's predicted output for sample t.

MSE and MAE are both indicators of the error between the predicted and the true results: MSE represents the average of the squared errors and MAE represents the average of the absolute errors. Typically, MSE is more sensitive to capturing cases of large errors, whereas MAE is more sensitive to cases of small errors. In real-time sequence prediction, however, due to noise and other uncertain factors, even if the shape of the input sequence in the first half is similar, the second half sequences to be predicted will not be exactly the same. Instead, they will exhibit fluctuations. Since MSE or MAE loss calculates distances on a one-to-one basis, the final predicted result becomes an average over time. This kind of result is certainly very poor.

In practice, we find that when dealing with data that has high sparsity and strong uncertainty, such intuitive evaluation metrics can significantly reduce the reliability of the model. There may be cases where the model has a very low MSE but performs poorly overall.

Therefore, in our proposed method, we adopted Dynamic Time Warping (DTW) as the loss function to calculate the differences between the predicted sequence and the ground truth. DTW is an algorithm used to compare the similarity between two time series with different lengths and time axis changes. By applying the DTW algorithm to the loss function f, it can effectively avoid the inaccuracies caused by the stretching of the prediction axis and better reflect the relationship between the predicted results and actual data. This method was also employed in work [30, 31] and resulted in improved training outcomes.

However, the DTW algorithm operates discretely and lacks differentiability. Consequently, we cannot directly employ it as a loss function for neural networks. To address this issue, Marco Cuturi *et al* introduced the concept of soft minimum instead of the DTW minimum and devised a differentiable version called Soft-DTW [32]. The objective of Soft-DTW is to find the best path where the elements on the path correspond to the alignment points between the query and reference sequences, taking into account the similarity at each alignment point. Soft-DTW provides a differentiable version of the DTW algorithm. Compared to the Euclidean loss and original DTW, Soft-DTW allows for better alignment between the predicted results and the actual shape of the data.

The Soft-DTW of two series Y and $\hat{Y}$ can be expressed as:

$$DTW_S(Y, \hat{Y}) = S(Y, \hat{Y}) + \min\ [DTW_S(Y_{-1}, \hat{Y}), DTW_S(Y, \hat{Y}_{-1}), DTW_S(Y_{-1}, \hat{Y}_{-1})]. \tag{10}$$

Here, $S(Y, \hat{Y})$ represents a weight matrix used to measure the similarity between the data points in two time series Y and $\hat{Y}$. In our case, Y represents the ground truth sequence of the target, while $\hat{Y}$ represents the prediction of Y, and Y_{-1} refers to the sequence Y with one data point removed. This removal allows for exploring different alignment scenarios, enhancing the flexibility of the alignment process. In Soft-DTW, the objective is to minimize the total alignment cost between the two sequences while considering the similarity between corresponding points. This formulation allows for a differentiable loss function suitable for training neural networks.

In practical applications, the choice of an appropriate distance or similarity measure depends on the specific problem and data type. We follow widely adopted practice and use the Euclidean distance to compute the differences between the corresponding data points, Y_i and $\hat{Y}_i$, of two time series, as:

$$S(Y, \hat{Y}) = \sqrt{\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2} \tag{11}$$

Here, n represents the length of the time series, and Y_i and $\hat{Y}_i$ represent the values of time series Y and $\hat{Y}$ at the i-th time point.

4 Experiments

To evaluate the effectiveness of the proposed approach, we conducted comprehensive experiments and compared our method with the state-of-the-art methods such as iTransformer [33] and Crossformer [34]. In addition, we also conducted ablation experiments to validate the effectiveness of the proposed approach of using Soft-DTW loss and interpolation techniques. We chose Autoformer [6] and Informer [18] as baseline models and compared their performance with and without incorporating Soft-DTW loss and interpolation. Following the existing practice, we utilize MSE as the metric in Table 1 to compare the performance of different models and loss functions.

In our experiments, we followed the same evaluation scheme as SOTA works such as Autoformer [6] and iTransformer [33], where the real-world datasets used for experiments are already split into Train, Validation and Test sets. Section 4.1 presents the details of the datasets. In addition, three groups of experiments were conducted with different prediction lengths to assess the method's performance in both short-term and long-term prediction tasks. All groups had an input window size of 96, while the target prediction sequence length (*a.k.a*, target window size or forecasting horizon) are 24, 48, and 96, respectively.

4.1 Experiment datasets

To evaluate the performance of our proposed method, we conducted experiments on two different datasets: the ETT (Electricity Transformer Temperature) dataset [18] and the weather dataset.

The ETT dataset [18] is a widely used benchmark dataset in time series forecasting tasks. It consists of historical records of transformer temperatures, collected from a power grid monitoring system. The dataset contains multiple transformer temperature time series, each corresponding to a specific transformer unit. The data is sampled at regular intervals and spans a considerable duration, allowing us to capture various temporal patterns and trends.

We selected four sub-datasets: ETT-h1, ETT-h2, ETT-m1, and ETT-m2. These sub-datasets share the same data structure but exhibit different patterns in their time series. The differences include the presence or absence of short-term oscillations, the length of pattern cycles, and the presence of upward or downward trends.

Table 1 Experiment results on ETT and weather datasets. The best performance in each set of experiments is highlighted in bold

Method	ETT				Weather
	h1	h2	m1	m2	
with an input window size of 96, and a target window size of 96					
Informer [18]	0.4973	0.4021	0.5085	0.2919	0.3049
Informer DTW	0.4728	0.3742	0.4791	0.2719	0.2839
AutoFormer [6]	0.4486	0.3462	0.4462	0.2551	0.2663
AutoFormer DTW	0.4260	0.3222	0.4156	0.2373	0.2479
iTransformer [33]	**0.3857**	**0.2973**	**0.3342**	**0.1795**	0.1744
Crossformer [34]	0.4231	0.7445	0.4038	0.2867	**0.1583**
Our method	0.4042	0.3000	0.3905	0.2016	0.2308
with an input window size of 96, and a target window size of 48					
Informer [18]	0.4724	0.3619	0.4827	0.2659	0.2744
Informer DTW	0.4493	0.3388	0.4301	0.2457	0.2588
AutoFormer [6]	0.4264	0.3258	0.4243	0.2423	0.2558
AutoFormer DTW	0.4047	0.2900	0.3948	0.2245	0.2231
iTransformer [33]	**0.3664**	**0.2700**	**0.3178**	**0.1705**	0.1569
Crossformer [34]	0.3808	0.6328	0.3632	0.2438	**0.1345**
Our method	0.3839	0.2726	0.3709	0.1915	0.2077
with an input window size 96, a target window size of 24					
Informer [18]	0.4010	0.3076	0.4223	0.2247	0.2371
Informer DTW	0.4046	0.2925	0.3665	0.2090	0.2146
AutoFormer [6]	0.3657	0.2928	0.3775	0.2126	0.2390
AutoFormer DTW	0.3393	0.2465	0.3455	0.1923	0.1895
iTransformer [33]	**0.3176**	0.2312	**0.2861**	**0.1498**	0.1333
Crossformer [34]	0.3427	0.5373	0.3256	0.2127	**0.1143**
Our method	0.3379	**0.2295**	0.3142	0.1628	0.1764

The ETT-h1 and ETT-h2 sub-datasets consist of transformer temperature time series with short-term oscillations. ETT-h1 has relatively shorter pattern cycles, while ETT-h2 has longer pattern cycles. These sub-datasets challenge the model to capture both short-term and long-term patterns accurately.

On the other hand, the ETT-m1 sub-dataset represents transformer temperature time series without short-term oscillations but still exhibiting varying pattern cycles. This sub-dataset focuses more on capturing seasonal variations or longer-term trends in the data.

Lastly, the ETT-m2 sub-dataset contains transformer temperature time series with a clear upward or downward trend. These time series exhibit a consistent increase or decrease in temperatures over time, allowing us to evaluate the model's ability to capture and forecast long-term trends accurately. By utilizing these two diverse datasets, we aim to validate the effectiveness and robustness of our proposed method across different domains and temporal characteristics. The datasets provide a realistic and challenging testbed for evaluating the performance of our approach and comparing it against baseline models.

The weather dataset on the other hand, contains historical weather observations such as temperature, humidity, precipitation, and wind speed. This dataset provides a diverse range of time series with different temporal characteristics, including daily, monthly, and yearly variations. By incorporating the weather dataset, we aim to evaluate the generalizability of our proposed method across different types of time series data.

Experimental results are based on predictions made on these datasets, with prediction targets of lengths 96, 48, and 24, while all input sequences have a fixed length of 96.

In terms of evaluation, we use Mean Squared Error (MSE) as the evaluation metrics to assess the performance of each model. These metrics provide quantitative measures of the accuracy and quality of the predictions compared to the ground truth values.

Next, we will present the experimental results obtained on these datasets and provide a detailed discussion of the findings.

4.2 Results and discussion

The results obtained from the experiments are summarized in Table 1, which presents the MSE values for each model and prediction target length. This tabulated data provides a clear comparison of the predictive performance of our proposed method against the baseline models and two SOTA models [33, 34]. The results of other methods shown in the table are based on reproducing their outcomes using their publicly available code. All methods were trained and evaluated using consistent hyperparameters, with efforts made to maintain uniformity across models as much as possible. This included ensuring the same number of layers in the encoder and decoder and maintaining the dimension of models. Additionally, the same training and testing datasets were utilized for all experiments.

According to our experimental results, we observed a significant relationship between the model's performance and the target window size. This phenomenon can be attributed to the inherent challenges posed by varying sequence lengths. While our improved model demonstrates enhanced capabilities in handling longer sequences, it is essential to note that shorter sequences present comparatively fewer challenges than longer ones.

Additionally, we provide graphical representations of selected experimental results to illustrate the prediction outcomes visually. These graphs showcase the predicted values alongside the actual ground truth values, allowing for a more intuitive understanding of the model's performance in capturing the underlying patterns and trends of the time series data.

Analyzing the results presented in Table 1, it becomes apparent that the majority of experiments show improved performance with the inclusion of interpolation methods compared to the baseline models. On the other hand, although our method did not outperform the SOTA model iTransformer in most experiments, it has significantly narrowed the gap with SOTA methods compared to Autoformer. Our method even exhibited better performance than iTransformer in short sequence prediction. Our experiment also demonstrates that, in addition to the patch method used in iTransformer, the decomposition of sequences also plays a significant role in predicting sequences. Improvement methods based on decomposition are worth considering in time series prediction tasks. The approach we proposed, which involves interpolating and separately predicting the Trend sequence, effectively improves the performance of Autoformer in prediction tasks.

The application of interpolation methods may lead to a less sensitive response of the model towards capturing the underlying patterns. Conversely, the performance of interpolation tends to be more favourable for short sequence predictions compared to long sequence predictions, suggesting that interpolation techniques contribute more informative features to shorter sequences.

Additionally, comparing the charts in Fig. 5, it is evident that the inclusion of interpolation methods can lead to smoother and better fitting predictions for time series with higher oscillation frequencies. On the other hand, the baseline model lacking smooth interpolation is more susceptible to short-term fluctuations.

Furthermore, models that incorporate DTW, such as AutoFormer and Informer with soft-DTW, outperform the baseline models in terms of predictive accuracy. This finding

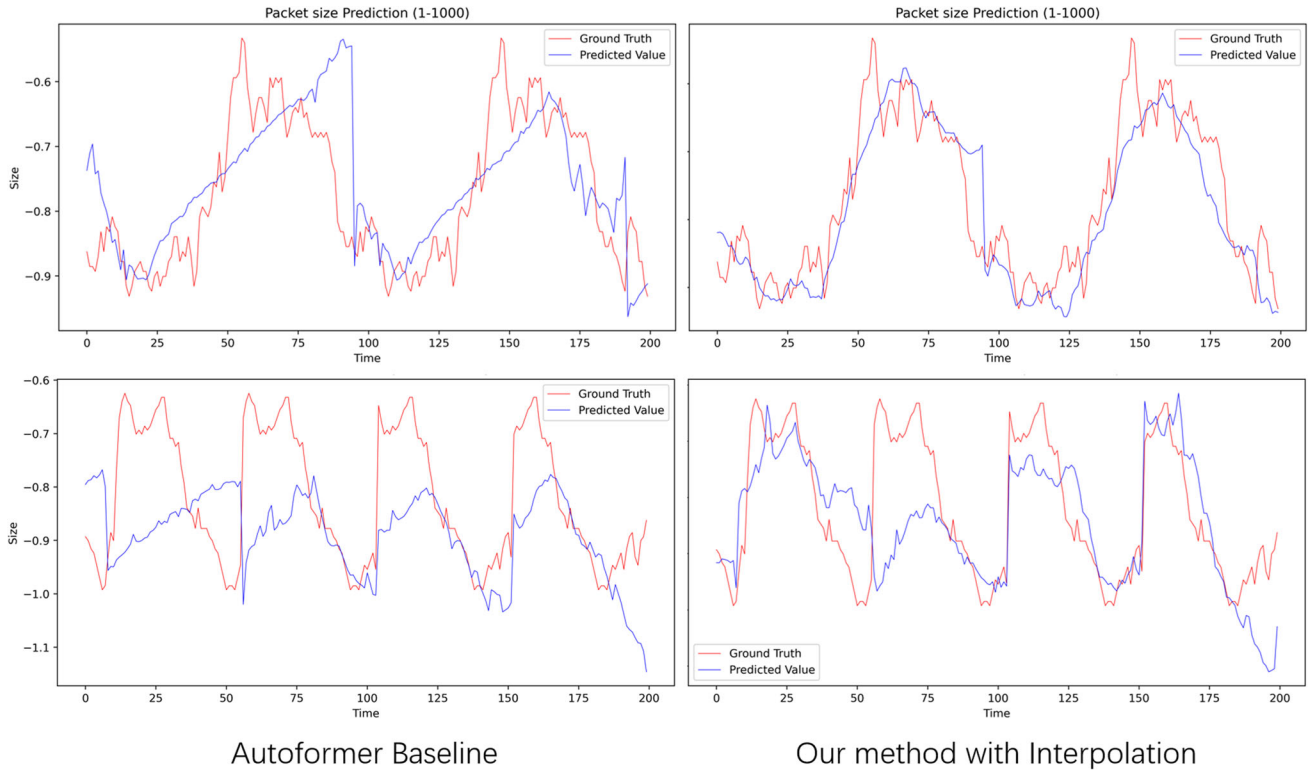

Fig. 5 Visual comparison of the experiment results obtained with the baseline method Autoformer (left column) and our method (right column)

further supports the notion that DTW offers greater value in time series prediction compared to traditional metrics like MSE or MAE. It is important to note the practical implications of these findings. The application of interpolation methods and DTW-based models can potentially improve the accuracy and reliability of time series predictions in various domains.

While this study presents promising results, it is important to acknowledge its limitations. The specific datasets used and the experimental setup may not fully capture the diverse range of time series characteristics encountered in real-world scenarios. Therefore, future research should explore alternative interpolation techniques and investigate the performance of DTW-based models on different types of time series data.

5 Conclusion

This paper highlights the importance of long-sequence time series forecasting in various domains such as finance, healthcare, and weather forecasting. The increasing volume of time series data requires advanced forecasting techniques to make accurate predictions. Additionally, this paper provides a comprehensive overview of long-sequence time series forecasting and emphasizes the significance of models like Informer and Autoformer in addressing the challenges associated with long-term and short-term predictions. Moreover, two methods are proposed to enhance existing models: one focuses on the enhancement of existing trend-cyclical and seasonal decomposition methods, while the other leverages DTW-based evaluation methods during training. We also introduce the concept of decomposition in time series analysis and present a novel method of temporal interpolation to enhance the prediction of cyclical patterns. The findings and contributions of this study are of significant importance for the advancement of time series forecasting techniques and provide valuable insights for future research in this field.

Author Contributions Conceptualization: Yuwei Chen, Wenjing Jia, Qiang Wu; Methodology: Yuwei Chen, Wenjing Jia, Qiang Wu; Formal analysis and investigation: Yuwei Chen, Wenjing Jia, Qiang Wu; Data curation, software and validation: Yuwei Chen; Writing - original draft preparation: Yuwei Chen; Writing - review and editing: Wenjing Jia, Qiang Wu; Supervision: Wenjing Jia, Qiang Wu.

Funding Open Access funding enabled and organized by CAUL and its Member Institutions.

Data Availability The datasets used in this study are publicly accessible.

Declarations

Competing Interests 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.

Ethical Approval This article does not contain studies with human participants or animals.

References

1. Sezer OB, Gudelek MU, Ozbayoglu AM (2020) Financial time series forecasting with deep learning: a systematic literature review: 2005–2019. Appl Soft Comput 90:106181
2. Kaushik S, Choudhury A, Sheron PK, Dasgupta N, Natarajan S, Pickett LA, Dutt V (2020) Ai in healthcare: time-series forecasting using statistical, neural, and ensemble architectures. Frontiers in big data 3, 4
3. Graditi G, Buonanno A, Caliano M, Di Somma M, Valenti M (2023) In: Manshahia MS, Kharchenko V, Weber G-W, Vasant P (eds.) Machine learning applications for renewable-based energy systems, pp 177–198. Springer, Cham
4. Karevan Z, Suykens JA (2020) Transductive lstm for time-series prediction: an application to weather forecasting. Neural Netw 125:1–9
5. Lim B, Zohren S (2021) Time-series forecasting with deep learning: a survey. Phil Trans R Soc A 379(2194):20200209
6. 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
7. Wang X, Kang Y, Hyndman RJ, Li F (2023) Distributed arima models for ultra-long time series. Int J Forecast 39(3):1163–1184
8. Hejase HA, Assi AH (2012) Time-series regression model for prediction of mean daily global solar radiation in Al-Ain, UAE. Int Scholarly Res Notices 2012
9. Zhang GP (2003) Time series forecasting using a hybrid arima and neural network model. Neurocomputing 50:159–175
10. Hansun S (2016) A new approach of brown's double exponential smoothing method in time series analysis. Balkan J Electric Compu Engr 4(2):75–78
11. Petrică A-C, Stancu S, Tindeche A (2016) Limitation of arima models in financial and monetary economics. Theo & Appl Econ 23(4)
12. Zhang D, Gong Y (2020) The comparison of lightgbm and xgboost coupling factor analysis and prediagnosis of acute liver failure. Ieee Access 8:220990–221003
13. Cao J, Li Z, Li J (2019) Financial time series forecasting model based on ceemdan and lstm. Physica A 519:127–139
14. Sagheer A, Kotb M (2019) Time series forecasting of petroleum production using deep lstm recurrent networks. Neurocomputing 323:203–213
15. Zhang X, Shen F, Zhao J, Yang G (2017) Time series forecasting using gru neural network with multi-lag after decomposition. In: Neural information processing: 24th international conference, ICONIP 2017, Guangzhou, China, 14–18, November 2017, Proceedings, Part V 24, pp 523–532 . Springer
16. Yamak PT, Yujian L, Gadosey PK (2019) A comparison between arima, lstm, and gru for time series forecasting. In: Proceedings of the 2019 2nd international conference on algorithms, computing and artificial intelligence, pp 49–55
17. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I (2017) Attention is all you need. Adv neural inf process syst 30
18. 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, pp 11106–11115
19. 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, pp 27268–27286 . PMLR
20. Li D, Li L, Li X, Ke Z, Hu Q (2020) Smoothed lstm-ae: a spatio-temporal deep model for multiple time-series missing imputation. Neurocomputing 411:351–363
21. Buonanno A, Di Gennaro G, Graditi G, Nogarotto A, Palmieri FA, Valenti M (2023) Fusion of energy sensors with missing values. Appl Intell 53(20):23613–23627
22. Hernandez-Matamoros A, Fujita H, Hayashi T, Perez-Meana H (2020) Forecasting of covid19 per regions using arima models and polynomial functions. Appl Soft Comput 96:106610
23. Abonazel MR, Abd-Elftah AI (2019) Forecasting egyptian gdp using arima models. Reports on Economics and Finance 5(1):35–47
24. Wang H, Peng J, Huang F, Wang J, Chen J, Xiao Y (2023) Micn: multi-scale local and global context modeling for long-term series forecasting. In: The eleventh international conference on learning representations
25. Hyndman RJ, Athanasopoulos G (2018) Forecasting: principles and practice. OTexts, Melbourne, Australia
26. Jofipasi CA, (2018) Selection for the best ets (error, trend, seasonal) model to forecast weather in the aceh besar district. In: IOP Conference series: materials science and engineering, vol 352, p 012055. IOP Publishing
27. Khan DM, Ali M, Iqbal N, Khalil U, Aljohani HM, Alharthi AS, Afify AZ (2022) Short-term prediction of covid-19 using novel hybrid ensemble empirical mode decomposition and error trend seasonal model. Front Public Health 10:922795
28. Peng B, Ding Y, Kang W (2023) Metaformer: a transformer that tends to mine metaphorical-level information. Sensors 23(11):5093
29. Banaś J, Kożuch A (2019) The application of time series decomposition for the identification and analysis of fluctuations in timber supply and price: a case study from poland. Forests 10(11):990
30. Lin Y, Koprinska I, Rana M (2020) Springnet: transformer and spring dtw for time series forecasting. In: Neural information processing: 27th international conference, ICONIP 2020, Bangkok, Thailand, 23–27, November 2020, Proceedings, Part III 27, pp 616–628 . Springer
31. Tao Z, Xu Q, Liu X, Liu J (2023) An integrated approach implementing sliding window and dtw distance for time series forecasting tasks. Appl Intell, 1–12
32. Cuturi M, Blondel M (2017) Soft-dtw: a differentiable loss function for time-series. In: International conference on machine learning, pp 894–903 . PMLR
33. Liu Y, Hu T, Zhang H, Wu H, Wang S, Ma L, Long M (2023) itransformer: inverted transformers are effective for time series forecasting. In: The twelfth international conference on learning representations
34. Zhang Y, Yan J (2022) Crossformer: transformer utilizing cross-dimension dependency for multivariate time series forecasting. In: The eleventh international conference on learning representations

Yuwei Chen received the BSc degree in Software Engineering from the University of New South Wales (UNSW) in 2016. He obtained his Master's degree in Information Technology from the University of Technology Sydney (UTS) in 2021. He is currently pursuing the Ph.D. degree with the Faculty of Engineering and Information Technology (FEIT) at UTS. His main research interests include time series forecasting and deep learning.

Wenjing Jia received her Ph.D. degree in Computing Sciences from the University of Technology Sydney in 2007. She is currently an Associate Professor at the Faculty of Engineering and Information Technology (FEIT), University of Technology Sydney (UTS). Her research falls in the fields of image processing and analysis, computer vision, and pattern recognition.

Qiang Wu received the BEng and Meng degrees in electronic engineering from the Harbin Institute of Technology, Harbin, China, in 1996 and 1998, respectively, and the Ph.D. degree in computing science from the University of Technology Sydney, Sydney, Australia, in 2004. He is currently an Associate Professor and a Core Member of the Global Big Data Technologies Centre, University of Technology Sydney. He has published more than 70 refereed papers, including those published in prestigious journals and top international conferences. His major research interests include computer vision, image processing, pattern recognition, machine learning and multimedia processing. He has served as the chair and/or a Programme Committee Member for a number of international conferences.