Received 20 February 2024, accepted 28 April 2024, date of publication 1 May 2024, date of current version 8 May 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3395651

RESEARCH ARTICLE

Online Attention Enhanced Differential and Decomposed LSTM for Time Series Prediction

LINA LI[1], SHENGKUI HUANG[1], GUOXING LIU[2], CHENG LUO[1], QINGHE YU[1], AND NIANFENG LI[1]

[1]College of Computer Science and Technology, Changchun University, Changchun 130022, China
[2]College of Cyber Security, Changchun University, Changchun 130022, China

Corresponding author: Nianfeng Li (linf@ccu.edu.cn)

This work was supported in part by the Natural Science Foundation of Jilin Province, China, under Grant YDZJ202101ZYTS191; in part by the National Natural Science Foundation of China under Grant 61772228; in part by Jilin Scientific and Technological Development Program under Grant 20210201083GX; in part by the Industry Research Innovation Fund of the University, China, under Grant 2020HYB03002; and in part by Jilin Medical and Health Industry Development Special Project 20230401104YY.

ABSTRACT Due to the time variability and bursty of data, accurate and lag-free time series prediction is difficult and challenging. To address these problems, we propose an online attention enhanced differential and decomposed LSTM (Long Short Term Memory) model called OADDL, which can better capture the comprehensive core features and important structures of time series. In this model, the core features of the time series are first generated through differential and decomposition methods to reduce data complexity and remove noisy data. Then, the self-attention module and LSTM capture the full time core features and important structures of time series. Finally, FCN (Fully Connected Network) fuses the omnidirectional features of time series. Meanwhile, we design an online two-stage training mode for this model, in which attention enhanced LSTM and FCN models are sequentially trained, and the training set and model hyper-parameters are continuously updated over time, thus further capturing the time-varying and burst characteristics of time series. We conduct tests on three typical datasets, and the experimental results show that compared with latest typical deep learning models, OADDL can more accurately predict time series data and effectively alleviate the problem of prediction lag.

INDEX TERMS Online prediction, time series, LSTM, self-attention mechanism, difference and decomposition.

I. INTRODUCTION

With the advent of big data era, time series data widely exists in the fields of traffic monitoring [1], [2], electrical system [3], [4], meteorological and environmental measurement [5], [6], [7], financial services [8], [9], bioinformatics [10], [11], [12], image processing [13], [14], [15], [16], etc., and has gradually become an important part of big data. In some applications, such as proactive resource scheduling in the stream processing platforms [17], [18] and exchange management in finance, time series prediction is the premise and key to correct decision-making [19], [20], thus getting more and more attention. However, time series continues to grow over time, exhibiting characteristics such as time-varying and sudden changes [21], making it difficult to make accurate predictions, and the prediction lags severely (i.e. the predicted data trend lags behind the actual trend). Currently, the accurate and lag-free prediction of time series has become an important and challenging issue in data analysis [22], [23].

Currently, time series prediction has become a research hotspot. In many existing works, time series prediction is mainly based on statistical [24], [25], machine learning [26], [27] and deep learning models [28], [29]. Statistical and machine learning models are not suitable for learning complex nonlinear time series features, especially bursts and their trends, resulting in lower predictive performance. In view

The associate editor coordinating the review of this manuscript and approving it for publication was Vivek Kumar Sehgal.

of the advantages of memory, hyper-parameter sharing and context awareness, variations of recursive neural network (RNN), long and short term memory (LSTM) and Transformer have been applied to time series data prediction [30], [31], [32], and the prediction performance is improved to a certain extent. Moreover, online learning mode and data decomposition method [33] have been introduced in some deep learning [23], [31], to adapt to the changes of time series patterns over time, resulting in improved training time and prediction performance of the models. Therefore, the deep learning model has gradually become the mainstream model of time series data prediction.

Although deep learning has gradually become the mainstream technology for time series prediction, there are still challenges and shortcomings. These shortcomings are mainly reflected in three aspects: data feature extraction, model construction, and training mode. In time series prediction, high-quality feature extraction methods will improve the performance of temporal data prediction. However, for complex temporal data, existing feature extraction methods are not comprehensive, resulting in inaccurate feature extraction. Meanwhile, model construction is the core of time series prediction, and there has been relatively little work that balances high predictive performance with low model complexity, with a focus on long-term prediction. In addition, the training mode of time series models is mainly offline, while online training mode is more complex and has a higher spatial cost. Therefore, it is necessary to design an efficient and high-quality time series prediction method based on deep learning, which supports all-round feature extraction, efficient and accurate time series models, and low-cost model training modes.

In order to address the aforementioned problems and challenges, inspired by the existing work, this paper comprehensively considers the data feature extraction method, model selection and training mode, and explores an online time series data prediction solution. In this solution, we use LSTM and FCN as the basic model structure, combined with self attention modules, differential methods, decomposition methods, and online two-stage training modes, to continuously capture the comprehensive core features and important structures of time series, supporting efficient, high-quality, and low-cost time series prediction. Our contributions are as follows.

(1) We propose an attention enhanced differential and decomposed LSTM and FCN model for time series prediction, in which differentiation and decomposition methods generate the core features of time series, self-attention module captures important structures of time series, LSTM learns the full-time core features and important structures of time series, while FCN fuses the full features of time series to improve the accuracy of time series prediction and reduce prediction lag.

(2) We design an online two-stage training mode to further improve the performance of the model. In this mode, the time series is divided into pieces by sliding window, and several pieces form one group for single training. We sequentially train attention enhanced LSTM to predict trends and residuals, and train FCN to predict time series data, and continuously update the hyper-parameters of the prediction model with new data, thereby improving the model's generalization ability.

(3) We conduct extensive experiments over three typical datasets across disease, meteorology and environment fields. Our empirical results show that compared with classic and latest prediction models, our proposed model has higher predictive performance and effectively alleviates prediction lag. Meanwhile, we further discuss the roles of various components of the model, the impact of different hyper-parameter settings, and the complexity of the model.

This paper is organized as follows. Section II reviews existing works on time series prediction. The proposed model is presented, involving the framework of the model, multi-dimensional time series generation, the attention enhanced differential and decomposed LSTM, and the two-stage online training mode in Section III. Section IV shows the experiment results and evaluation. In Section V, the model is discussed. Section VI concludes this work and prospects for future work.

II. RELATED WORKS

The prediction models of time series mainly include traditional models and deep learning models. The former includes the statistical model and the machine learning model. The latter can be further divided into offline learning and online learning according to the timing of model updating. Since our work belongs to online learning, we especially review some major online learning-based approaches related to ours. This section introduces the related research from three aspects and analyzes the existing problems.

A. TRADITIONAL MODELS

For the time series, the statistical models, mainly including AR (Autoregressive), ARMA (Autoregressive Moving Average) and ARIMA (Autoregressive Integrated Moving Average), are used in early prediction work [34]. Typically, Bai et al. [25] present a neuron-based Kalman filter to represent the state-space model, and use the nonlinear autoregressive model to excavate the functions of the neuro units. Wang et al. [24] conduct the ARIMA (Autoregressive Integrated Moving Average) model to predict short-term cloud coverage, in which a difference processing process is added to ARMA [35]. Ding et al. [36] develop an online convex optimization method to predict time series by approximating the evolution of ARIMA processes and using a discounted online Newton method. Although statistical models can fit stationary time series well, they perform poorly in dealing with nonlinear data. Sue et al. [26] use the extreme learning machine model to predict the financial time series, which is based on the 2,1-norm and Random Fourier Mapping. Yang et al. [27] and Liu et al. [37] propose the dynamical regularized echo state network model and the quantum echo state network model to determine the structure size and predict the financial time series, respectively. Combined with ensemble empirical mode decomposition (EEMD) technique, Xu et al. [38] establish a dual-scale deep belief network for predicting the daily demand water volume of the urban. These machine learning models have achieved good performance in nonlinear time series predicting. However, they are still unable to fully learn the characteristics of time series, resulting in not fitting the volatility and bursty of complex time series well.

B. DEEP LEARNING MODELS

Deep neural networks have powerful learning ability of characteristics and have also been applied for time series prediction [39]. Especially, the recurrent neural network (RNN) [28] mines the time and semantic information of data, thus attracting the most attention. Huang et al. [30] propose a bidirectional recurrent neural network (BRNN) model to accurately and real-time predict the traffic flow data. Further, the long short term memory (LSTM) [29], [40], called as gated RNN, introduces input, forgetting and output gates and activation functions to solve short-term and long-term dependence problems, and is promising in modeling irregular time series data [41], [42], [43], [44]. Wang et al. [31] propose an incremental ensemble LSTM model-IncLSTM, which combines ensemble learning with transfer learning to build the hierarchical network structure. Shi et al. [22] combine data decomposition and parallel deep networks to predict data, in which each deep network is separately trained by using each data group. In [45], an LSTM+ESN architecture is presented to combine characteristics of both networks to accurately predict wind generation. Tian et al. [46] integrate the hidden feature of the CNN and LSTM models to improve the forecasting accuracy. Yu et al. [47] propose a bespoke LSTM combining dynamic time warping (DTW) for accurate daily peak load forecasting. Zhou et al. [48], propose a frequency enhanced decomposed Transformer to capture the detailed structure and global profile of time series for long-term series forecasting. In [49], a general multi-scale framework is proposed for transformer-based time series forecasting models, which iteratively refines a forecasted time series at multiple scales with shared weights. Shen et al. [32] propose a novel two-stage Transformer framework to fit different statistical properties between input and prediction sequences. Although these deep learning models have achieved good prediction results by integrating decomposition methods, statistical models or neural networks, they still have not solved the problem of prediction lag of time series, and the accuracy of short-term time series predicting needs to be further enhanced.

C. ONLINE LEARNING METHODS

In order to overcome the time-consuming and static characteristics of offline learning method and better adapt to sudden nonlinear data, some research work focuses on online learning method. The online learning method keeps the memory of the original data and learns new data features through constantly updating model hyper-parameters with new datasets, so as to better adapt to data changes with lower training costs [50], [51], [52]. Liu et al. [5] initiate model updates for online prediction after detecting data distribution changes through concept drift detection and specified rules. Zhang [23] adopts meta-learning to adaptively adjust the learning rate of the stochastic gradient descent (SGD) in recurrent neural networks to continuously fit time series data. These two schemes either introduce additional processing steps or are only applicable to the network structure with SGD, and the serious prediction lag still exists in some cases. In IncLSTM [31], multiple LSTM weak learners are trained in parallel by the sliced time series data to learn the time-varying characteristics of data and obtain better prediction performance, which increases the cost of storage and selecting learners. The above online learning models improve the accuracy of sudden data prediction to a certain extent, but it still does not solve the problem of prediction lag better, and has additional storage and processing costs, and the solution is not universal.

III. ONLINE ATTENTION ENHANCED DIFFERENTIAL AND DECOMPOSED LSTM MODEL

A. FRAMEWORK OF THE PREDICTION MODEL

In this paper, we propose an online attention enhanced differential and decomposed LSTM model (OADDL) for time series prediction. The framework of OADDL is composed of three parts, including the continuous sample generation, the attention enhanced differential and decomposed LSTM and FCN [53], and the online two-stage training mode. The structure and overall flow chart of the framework is shown in Figure 1.

As shown in Figure 1, the time series is first divided into G groups to simulate the data samples of online learning, and the length of each data group is M. Each data group carries out online learning once. After each data group training, use the next data group to train the model online. Each data group is divided into slices every N data in the sample generation phase to support online characteristics learning (the details are in Section III-B). Each slice is taken as a sub-dataset to ensure the integrity and continuity of training data and provide diversified data samples for the model. Then, each sub-dataset is differentiated and decomposed to generate first-order difference, second-order difference, period, trend and remainder series. These derived time series represent the multi-dimensional characteristics of the original time series, which reduces the complexity of data and the difficulty of prediction. Further, these data are fed into the prediction model based on LSTM and FCN, which is composed of three cascaded networks (the details are in Section III-C). For irregular data, the self-attention module (SAM) located in front of LSTM captures the important structure of all time series, then LSTM predicts the trend and remainder of time series, while FCN fits the features of all time series. Finally, the online two-stage training mode is conducted to constantly train and update the prediction model. In the process of model training, two data groups are used to train two cascaded networks in turn, respectively, then different data are predicted after each training, and the prediction result of the first network is used as the training data of the second network, finally, the trained

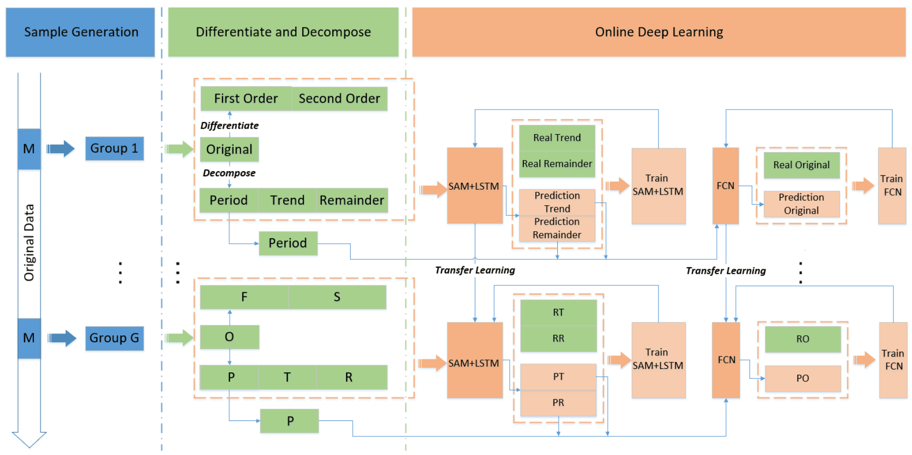

FIGURE 1. **The framework of online attention enhanced differential and decomposed LSTM model.**

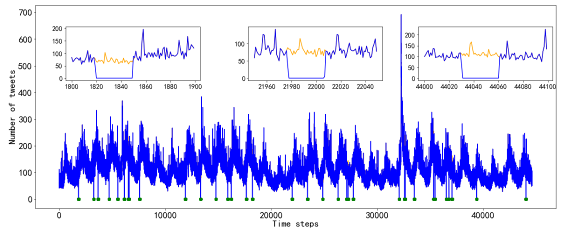

FIGURE 2. **The time series with abnormal data and filled data.**

network is used to predict the original time series data (the details are in Section III-D).

B. SAMPLE GENERATION OF CONTINUOUS DATASETS

In the time series, there may be abnormal data with a value of 0, as shown by the green dot in Figure 2. To ensure the validity of time series data, we first handle these abnormal data. Due to the continuity of missing values, we use periodic interpolation methods to calculate the data where the 0 value is located. Specifically, the classic decomposition method [54] is first used to decompose the time series data to obtain trend, residuals and periods terms. Then, considering the characteristic that the trend will not change significantly, the entire trend component is used as the trend component of the missing value, and a random number between -10 and 10 is used as the remainder. Finally, the trend, period, and remainder are added as the fill value. The partial interpolation results of time series abnormal data are shown in the subgraph of Figure 2.

For the valid time series $\mathscr{D}$, we divide it into continuous groups, and use each group D with a length of M to train the online model. Specifically, D is divided into a series of time slices with overlapping time series data. Each time slice is generated through a sliding window with a size of N, which moves backward one position at a time. One slice is generated in chronological order, called one sub-dataset. The continous n sub-datasets form a dataset O and each sub-dataset O_i is represented as follows.

$$O_i = \{D_i, D_{i+1}, \cdots, D_{i+N-1}\}, 1 \leq i \leq n \tag{1}$$

D_i is the first data in O_i, and i is the number of the sub-dataset, and is also the number of the data D_i in group D. $O \in \mathbb{R}^{n\times N}$ is a two-dimensional matrix with n rows and N columns, representing n sub-datasets and N data in each sub-dataset. O_{ij} represents the j-th data of the i-th sub-dataset.

C. ATTENTION ENHANCED DIFFERENTIAL AND DECOMPOSED LSTM

For the purpose of reducing the fluctuation and complexity of the time series, the difference method and decomposition method are used to transform time series of each dataset O to generate its multi-dimensional time series, so as to extract data features more accurately. The first-order are second-order difference can be represented as

$$F, S = Differentiated(O) \tag{2}$$

$F, S \in \mathbb{R}^{n\times N}$ are the new series obtained by first-order difference and second-order difference, respectively. The first data in the first-order difference and the first two data in the second-order difference are both null values filled with zeros.

The first three subgraphs of Figure 3 show an example of the difference results of 500 COVID-19 related tweets on twitter in August 2021. As shown in Figure 3, the two new time series are generated by differencing the original data. Obviously, the data volatility of these two new time series is smaller and smoother than the original time series, and the effect of the second-order difference time series is more prominent. It can be seen that the difference method can indeed slow down the volatility of data and remove noise data to certain extent.

Further, the decomposition method is also applied to the data for precision feature extraction. There are many time series decomposition methods, such as Classical decomposition [54], X11 decomposition, SEATS decomposition [55], STL decomposition [56] and so on. In this paper, the addition model

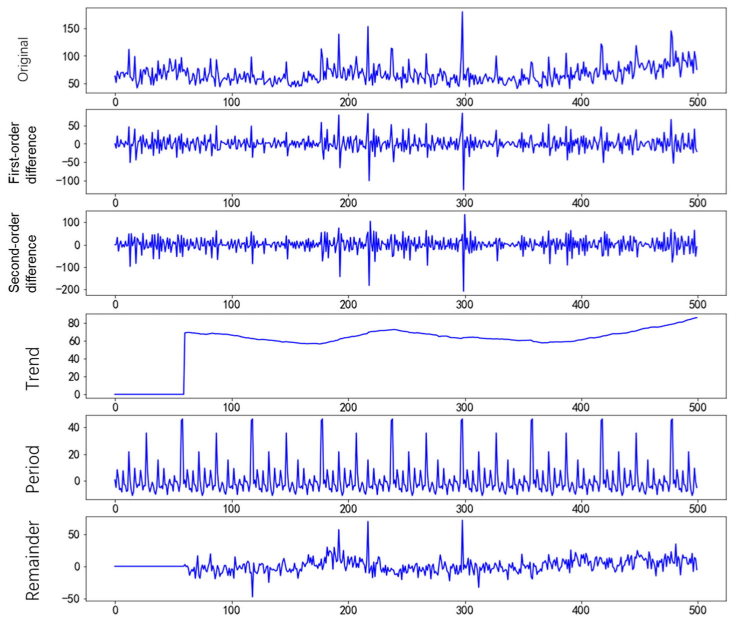

FIGURE 3. An example of the difference and decomposition results of the covid19_twitter time series.

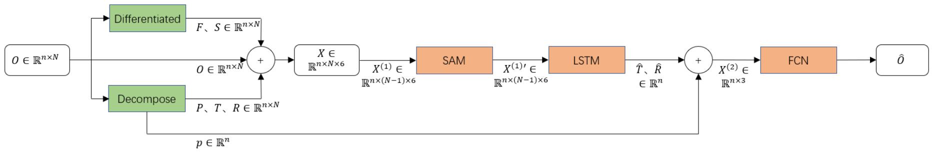
$O \in \mathbb{R}^{n\times N}$
Differentiated
Decompose
F、$S \in \mathbb{R}^{n\times N}$
$O \in \mathbb{R}^{n\times N}$
P、T、$R \in \mathbb{R}^{n\times N}$
$p \in \mathbb{R}^{n}$
$X \in \mathbb{R}^{n\times N\times 6}$
$X^{(1)} \in \mathbb{R}^{n\times(N-1)\times 6}$
SAM
$X^{(1)\prime} \in \mathbb{R}^{n\times(N-1)\times 6}$
LSTM
$\hat{T}$、$\hat{R} \in \mathbb{R}^{n}$
$X^{(2)} \in \mathbb{R}^{n\times 3}$
FCN
$\hat{O}$

FIGURE 4. The attentiion enhanced differential and decomposed LSTM and FCN.

of classical decomposition method is adopted, which is widely used in the decomposition of time series [57], [58]. The trend item represents a trend or state of continuous change in a long period of time, and the seasonal or periodic items represent the regular changes of time series, that is, the frequency of periodic changes. The residual item represents the impact of many accidental factors on the time series, which is the result of the original time series after removing seasons or periods and trends. For the dataset O, the decomposition of O can be described by

$$P, T, R, p = Decompose(O) \tag{3}$$

$P, T, R \in \mathbb{R}^{n\times N}$ represent the periodic component, trend component, and remainder of O after decomposition, respectively. $p \in \mathbb{R}^n$ is an array of length, which represents the last periodic component of each sub-dataset, and p_i is P_{iN}.

The last three subgraphs of Figure 3 show an example of the decomposition results of the same time series. It can be seen that the trend item shows a stable trend with a small fluctuation over time. The period of the seasonal item is 60. While, the residual term fluctuates frequently, and in most cases, it is consistent with the fluctuation amplitude and the trend of the original time series.

By combining differentiation and decomposition processing, we construct a prediction model based on SAM, LSTM and FCN, as shown in Figure 4. SAM considers the relationship between each element in time series and all other elements to better understand the contextual structure information. LSTM is adopted to predict the trend and residuals components of time series, with the input of the original time series and 5 new time series after difference and decomposition. FCN is used for fitting the predicted components, and outputs the predicted result of the time series. The input is the trend and residual components output by LSTM, as well as the original periodic components. The inputs and outputs of three models can be expressed as

$$\begin{aligned} X^{(1)'} &= f_{SAM}(X^{(1)}) \\ \hat{T}, \hat{R} &= f_{LSTM}(X^{(1)'}) \\ \hat{O} &= f_{FCN}(\hat{T}, \hat{R}, p) \end{aligned} \tag{4}$$

$X \in \mathbb{R}^{n\times N\times 6}$ is a set of original time series and its derived 5 time series, which is a three-dimensional array with n rows, N columns, and 6 depth (channels). X is further divided into two training sets, $X^{(1)}$ and $X^{(2)}$, and their corresponding ground-truth sets are $Y^{(1)}$ and $Y^{(2)}$, respectively. $X^{(1)'}$ is the result of $X^{(1)}$ being transformed through the SAM network, namely $X^{(1)}$ is multiplied by the normalized attention weight. $X^{(1)}$ and $Y^{(1)}$ are used for training the SAM and LSTM networks (abbreviated as SAM_LSTM), and $X^{(2)}$ and $Y^{(2)}$ are used for training FCN, which can be expressed as

$$\begin{aligned} X^{(1)} &= \{X_{ij} | 1 \le i \le n, 1 \le j \le N-1\} \\ Y^{(1)} &= \{T_{iN}, R_{iN} | 1 \le i \le n\} \qquad (5) \\ X^{(2)} &= \{\hat{T}_i, \hat{R}_i, p_i | 1 \le i \le n\} \\ Y^{(2)} &= \{O_{iN} | 1 \le i \le n\} \qquad (6) \end{aligned}$$

$\hat{T}$ and $\hat{R}$ are the outputs of the LSTM network, representing the predicted values of T_{iN} and R_{iN}, respectively. Since the periodic component appears repeatedly in each cycle, p is known, and does not need to be predicted. X_{ij} represents 6 data values corresponding to the j-th data in the i-th sub-dataset, namely,

$$X_{ij} = \{O_{ij}, F_{ij}, S_{ij}, P_{ij}, T_{ij}, R_{ij}\}, 1 \le i \le n, 1 \le j \le N \qquad (7)$$

D. ONLINE TWO-STAGE TRAINING MODE

We further design an online two-stage training mode for the OADDL model, which includes two aspects: online transfer learning and two-stage training. Online transfer learning improves the efficiency and accuracy of model training by continuously loading the training model of the old dataset as the initial value of the current model to retain the characteristics of all previous time series data. In two-stage training, the SAM_LSTM and FCN are trained separately, enabling the model to learn the features of the dataset from different dimensions to improve the performance of the prediction. Algorithm 1 describes the above training mode.

In Algorithm 1, for the time series $\mathscr{D}$, obtain one group D each time (lines 1-2). If the group D is successfully obtained, continue to judge whether it is the first group. If it is, initialize the prediction model of OADDL. Otherwise, load the last trained model, that is, execute transfer learning (lines 3-7); Otherwise, return the prediction model of OADDL (lines 8-9). Next, calculate the number of sliding windows and generate samples for the dataset O using formula (1). Then, according to formulas (2) and (3), generate 5 new datasets from O and normalize all datasets (lines 10-13). Furthermore, construct the training set X and generate the training sets $X^{(1)}$ and $Y^{(1)}$ for SAM_LSTM through formula (5), i.e. prepare the training data for the first stage (lines 14-15). Then, perform the first stage of training using formula (4) (lines 16-18), and generate the training sets $X^{(2)}$ and $Y^{(2)}$ for FCN through formulas (4) and (6), i.e. prepare the training data for the second stage (line 19). Finally, perform the second stage of training using formula (4) (lines 21-23), and save the prediction model of OADDL trained with the current group of data (line 24).

Algorithm 1 The Online Two-Stage Training Algorithm

Input: Original dataset $\mathscr{D}$, Length of each group M, Size of sliding window N, Epochs of training E.
Output: the prediction model of OADDL

```
1:  while $\mathscr{D}$ is not null do
2:      Get one group $D$ of $\mathscr{D}$
3:      if $D.Length == M$ then
4:          if $D$ is the first group then
5:              Initialize the OAODL model
6:          else
7:              Load the prediction model of OADDL    ▷ Transfer learning
8:      else
9:          Return the prediction model of OADDL
10:     Calculate the number of sliding windows, $n = M - N + 1$
11:     Generate samples of dataset $O$ using formula (1)
12:     Generate five new datasets $F, S, T, R, P$ from $O$ using formulas (2) and (3)
13:     Normalize $O, F, S, T, R, P$
14:     Construct the training dataset $X$ and generate the training sets of SAM_LSTM $X^{(1)}$ and $Y^{(1)}$ using formula (5)
15:                                               ▷ Preparing training data
16:     for $e = 1 : E$ do
17:         Train SAM_LSTM with $X^{(1)'} = f_{SAM}(X^{(1)})$ and $\hat{T}, \hat{R} = f_{LSTM}(X^{(1)'})$ using formula (4)
18:                                               ▷ Training SAM_LSTM
19:     Generate the training datasets of FCN $X^{(2)}$ and $Y^{(12}$ using formulas (4) and (6)
20:                                               ▷ Preparing training data
21:     for $e = 1 : E$ do
22:         Train FCN with $\hat{O} = f_{FCN}(X^{(2)})$ using formula (4)
23:                                               ▷ Training FCN
24:     Save the prediction model of OADDL
```

IV. EXPERIMENT RESULTS AND EVALUATION

A. EXPERIMENTAL SETUP

The experiments are conducted on a PC with windows 10 system, and the hardware configuration is AMD ryzen 5 4600h (3.0GHz) with radeon graphics (12CPUs) processor, 16GB RAM memory, 300GB HDD Disk, and NVIDIA geforce GTX 1650 graphics card. The experimental programs are written by Python 3.7, and the deep learning framework is tensorflow 1.14.

We set the hyper-parameter setting of OADDL, as shown in Table 1. The optimizer, learning rate, batch size and epochs of SAM_LSTM and FCN in OADDL are the same, which are Adam (RMSProp with classical momentum), 0.001, 64 and 20, respectively. SAM_LSTM has two outputs, and we set the same loss function for them, that is MSLE (mean squared logarithmic error), but their weights are different for each output to modulate their contributions to the total training loss. The loss_weights of SAM_LSTM is [1.0,0.6]. The loss function of FCN is MAE (mean absolute error). The whole dataset is divided into five groups to simulate online learning. The ratio of training set to test set in each group is 3:1.

Six deep learning models are selected to compare with our model. A-LSTM is a significant deep learning model, with a self attention module added before LSTM. PDCI (parallel depth prediction with covariance cross fusion) [22], incLSTM (incremental integration LSTM) [31], FEDformer [48], scaleformer [49] and GBT [32] are the latest deep learning models, and have achieved good performance on data series prediction.

The hyper-parameters of comparative models are shown in Table 2, the epochs of six models are 200, and the Epochs of the strong and weak learners of incLSTM are 200. The batch sizes of these four models are 24, 64, 64, 32, 8 and 16, respectively. The loss of incLSTM is MAE, while the other five are MSE (mean square error). The optimizers of these six models are all Adams. The learning rate of scaleformer, GBT and FEDformer is 0.0001, and the other three ones are 0.001.

TABLE 1. The hyper-parameter setting of OADDL.

Network	hyper-parameter	Value
SAM_LSTM	optimizer	Adam
	loss	MSLE
	learning_rate	0.001
	batch_size	64
	epochs	20
	loss_weights	[1.0,0.6]
FCN	optimizer	Adam
	loss	MAE
	learning_rate	0.001
	batch_size	64
	epochs	20

TABLE 2. The hyper-parameter settings of other models.

Model	hyper-parameter				
	epochs	batch_size	loss	optimizer	learning_rate
PDCI	200	24	MSE	Adam	0.001
incLSTM	200,200	64	MAE	Adam	0.001
A-LSTM	200	64	MSE	Adam	0.001
scaleformer	200	32	MSE	Adam	0.0001
GBT	200	8	MSE	Adam	0.0001
FEDformer	200	16	MSE	Adam	0.0001

B. DATASETS AND EVALUATION CRITERIA

For the validation of the proposed model, the experiments are conducted with three practical datasets. In time series data prediction model training, the ratio of training set to test set is 3:1, and the length of each time slice is set to 61. The first 60 data are used as training sub data sets, and the 61st data is the predicted true value.

(1) COVID-19_twitter dataset. The dataset is related to COVID-19 on Twitter, starting from August 1, 2021, with a time range of 31 days and daily data of about 150000-20000 tweets. We sort the original data according to time, and calculate the number of tweets per minute to obtain the original time series data.

(2) Beijing air quality dataset (http://www.stateair.net/web/historical/1/1.html). The dataset consists of air quality data

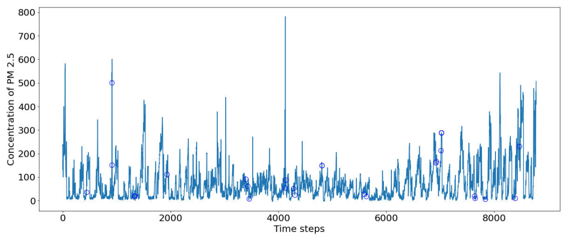

FIGURE 5. The PM2.5 data filled by the nearest interpolation method.

from Beijing in 2016. These data include the location, time, and value of PM2.5. We are interested in the value of PM2.5, with a data interval of one hour and a total of over 8000 pieces of data. As shown in Figure 5, some PM2.5 data is missing, and we use the nearest interpolation method to fill in the missing values.

(3) Sunspot dataset (https://www.sidc.be/silso/INFO/snmtotcsv.php). The dataset is the monthly average total sunspot count from January 1749 to November 2021, with a total of over 3000 pieces of data.

Without losing generality, we use four common evaluation indices to evaluate the performance of time series prediction models: RMSE (Root Mean Square Error), MAE (Mean Absolute Error), MAPE (Mean Absolute Percentage Error)and R^2 (R Squared) [38], [59]. Specifically, RMSE measures the deviation between the predicted value and the true value, which is the square root of the square mean of the error between the predicted value and the true value. MAE is the absolute mean value of the error between the real value and the predicted value. MAPE is a measure of relative error. R^2 is the determination coefficient, which determines the closeness between the real value and the predicted value, expressed as the proportion of the sum of squares caused by the error of the real value and the predicted value in the total sum of squares of the mean value of the real value and the error of the predicted value. For the OADDL being an online learning model, the whole dataset is divided into five groups to simulate online learning. We choose the indicators of the last group as the performance indicators of the OADDL.

C. ONLINE PREDICTION PERFORMANCE

The prediction accuracy of the proposed OADDL was compared with six comparison models on three datasets, with evaluation metrics including RMSE, MAE, MAPE, and R^2, as shown in Table 3. On the COVID-19_twitter dataset, OADDL performs best on R^2. Compared with the optimal PDCI among the six models, R^2 increases from 0.794 to 0.828, an increase of approximately 4.3%, while MAE is the smallest, a decrease of 1.3%, and RMSE and MAPE are slightly higher, with a difference of approximately 0.4% and 2.8%, respectively. Compared with the suboptimal model incLSTM, R^2 increased by approximately 8.7%, while RMSE, MAE, and MAPE decreased by nearly 11%, 9%, and 9%, respectively. The results indicate that OADDL can predict data more accurately. The prediction results of

TABLE 3. The performance comparison of different models.

Model	COVID-19_twitte dataset				Beijing air quality dataset				Sunspot dataset			
	RMSE	MAE	MAPE	R^2	RMSE	MAE	MAPE	R^2	RMSE	MAE	MAPE	R^2
OADDL	12.57	9.68	0.106	0.828	24.60	14.70	0.303	0.943	22.71	16.64	0.259	0.886
PDCI	12.52	9.81	0.103	0.794	28.29	17.75	0.477	0.904	22.44	16.17	0.308	0.864
incLSTM	13.95	10.58	0.117	0.762	33.21	20.71	0.322	0.898	28.80	20.01	0.277	0.823
GBT	18.37	12.34	0.125	0.636	54.45	33.71	0.376	0.709	27.95	21.09	0.333	0.853
FEDformer	18.65	13.01	0.129	0.631	53.18	31.75	0.326	0.723	30.39	21.99	0.263	0.841
scaleformer	14.74	10.89	0.117	0.750	39.44	25.15	0.895	0.863	25.68	20.32	0.432	0.864
A-LSTM	16.28	12.45	0.124	0.713	30.34	19.30	0.298	0.913	25.87	19.85	0.267	0.865

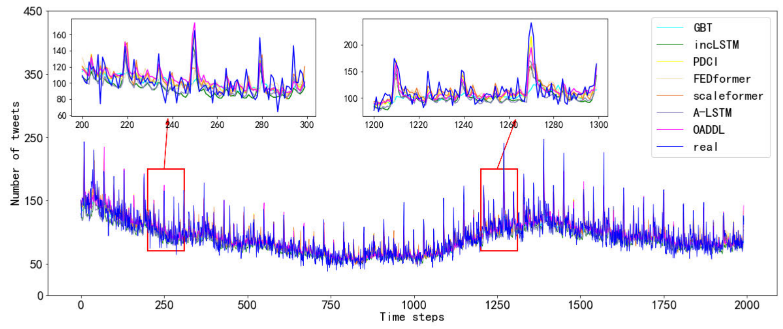

FIGURE 6. Comparison of the predicted results for COVID-19_twitter.

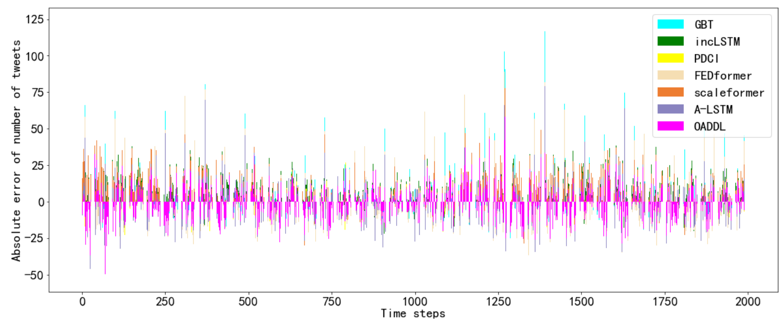

FIGURE 7. Absolute errors of the prediction on COVID-19_twitter by different models.

the fifth dataset of COVID-19_twitter dataset are shown in Figure 6. For an intuitive comparison, the absolute errors of the prediction results are calculated, as shown in Figure 7. In Figures 6 and 7, the curves in different colors correspond to the predicted results of different models, with blue representing the true values. From the partially enlarged drawing in Figure 6, it can be seen that the curve of OADDL fits the true curve best, especially the peak of abrupt changes. Correspondingly, as shown in Figure 7, the absolute error of OADDL is smaller than that of other models. Therefore, the proposed method performs better than other models.

Table 3 also shows the performance comparison on the predicted the air quality dataset. All models performed better on this dataset, while OADDL had the best overall performance. R^2 is the highest, reaching 0.943, while RMSE and MAE are both the lowest, and MAPE is the second smallest. Compared with the optimal performance among the six models, it has improved by about 3.3% (R^2, A-LSTM), decreased by 13% (RMSE, PDCI), decreased by 17.2% (MAE, PDCI), and increased by 1.7% (MAPE, A-LSTM). The results indicate that OADDL has better prediction accuracy on the air quality dataset and still outperforms other models. The predicted results for this dataset are shown in Figure 8. The predicted curve of OADDL is closest to the curve of the real values, which means that OADDL can more accurately predict the trend of data changes and values, thus having good prediction performance.

Similarly, the evaluation metrics for the Sunspot dataset are shown in Table 3. It can be observed that compared with the other six models, OADDL has the largest R^2 (0.866), the smallest MAPE (0.379), the second smallest RMSE (22.71) with a difference of 1.2% (PDCI), and the second smallest MAE (16.64), increased by 2.8% (PDCI). Compared with other models, the OADDL model has the smallest overall error and the predicted values are more in line with the real values. In addition, the predicted results of the Sunspot dataset are shown in Figure 9. The OADDL model still shows the best fit, especially with high matching of predicted trends, effectively reducing errors caused by prediction lag.

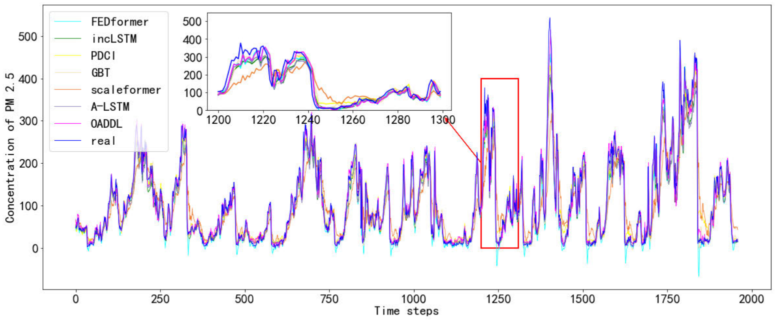

FIGURE 8. Comparison of the predicted results for PM2.5.

In summary, OADDL shows consistent optimal prediction accuracy on three typical datasets with different time-varying and burst characteristics, with the overall smallest prediction error and the most matching prediction trend. Therefore, OADDL has good prediction performance, to some extent alleviates prediction lag, and applicability to different time series datasets. This indicates that through self attention mechanism, differential and decomposition methods, and online two-stage training mode, OADDL can effectively learn the characteristics of data itself and the correlations between data from time series, thus having stronger prediction and generalization abilities.

V. DISCUSSION

A. ABLATION STUDIES

Our proposed OADDL mainly consists of four parts, namely a deep learning module LSTM, a differential and decomposition module, a self-attention module, and a two-stage training mode. In this subsection, the contribution of each module to the entire model is evaluated on the COVID-19_Twitter dataset by removing other modules from the model. (1) LSTM. We refer to the network model with deep learning module LSTM as LSTM. (2) LSTM_DD. We refer to the model as LSTM_DD, which performs the differential and decomposition processing on the original time series data and uses the results as input to LSTM. (3) LSTM_A. Similarly, the model that combines self-attention module in front of LSTM is called LSTM_A. (4) Two-stage. We refer to the network model with the differential module removed as Two-stage.

The results of the ablation experiment are shown in Table 4. It can be clearly seen that LSTM has the worst prediction performance, and LSTM_DD with the addition of differential and decomposition components, the prediction performance has improved with an R^2 increase of approximately 20%. Similarly, R^2 of the LSTM_A model is 0.729, which is about 18% higher than the original LSTM model. After adding a two-stage training mode, the improvement is more significant with R^2 directly increasing from 0.619 to 0.813, an improvement of nearly 31%, close to the performance of OADDL of 0.828. The other three indicators also have similar performance variation characteristics. The performance of LSTM_DD and LSTM_A has been improved but not significantly, with an improvement range of 7.3%-17.7% and 5.7%-15.7%, respectively. While in the two-stage prediction mode, the performance can be improved by up to 30.0% (RMSE) and as low as 12.2% (MAPE). In addition, the performance comparison between Two-stage and OADDL shows that differential and decomposition have played a certain role, and the predictive performance has been improved from about 1.8% to 4.3%. Therefore, the two-stage training mode combining with self-attention module and decomposition is crucial and has the greatest impact on improving the performance of the model. The influence of differential and decomposition combination and self-attention module weakens in turn, and the differential is the weakest.

TABLE 4. The ablation study of OADDL.

	LSTM	LSTM_DD	LSTM_A	Two-stage	OADDL
RMSE	18.75	15.43	15.81	13.13	12.57
MAE	12.66	11.12	11.66	9.93	9.68
MAPE	0.123	0.114	0.116	0.108	0.106
R^2	0.619	0.742	0.729	0.813	0.828

In addition, we also compare the results of offline and online training for OADDL. In the experiment, the dataset is still divided into five groups, each with an independent test set. Offline training reconstructs a model for each group for training and testing, while online training uses the same model for continuous training and testing on five training sets. The epochs for the first group of data training are 20, while for the other four groups, the epochs are 5, and other parameter settings are the same as Table 1. As shown in Table 5, the performance of online training is slightly better than offline training, with an average improvement of 2.62% in R^2, 4.96% in RMSE, 3.48% in MAE, and 0.94% in MAPE, respectively. The reason for the above phenomenon is that online training can not only remember existing features of data that change or mutate over time, but also discover and learn new data features, enabling the model to better adapt to data changes, thereby improving the predictive performance of unpredictable data. Therefore, online training can to some extent improve the model's adaptability to time-varying burst data. Meanwhile, the training time cost is also shown in the Table 5. It can be seen that the training time for each epoch is about 5 seconds, and online training is significantly better than offline training

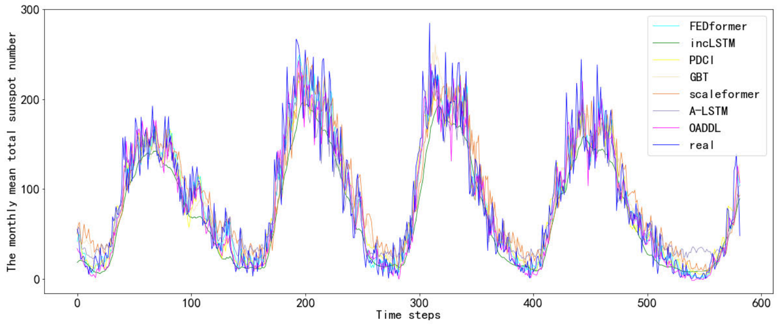

FIGURE 9. Comparison of the predicted results for Sunspot.

in terms of time cost, saving nearly 69% of the time on average. As time goes on, the advantages of online training become increasingly apparent.

TABLE 5. The performance and training cost comparison with online and offline learning.

Groups	mode	RMSE	MAE	MAPE	R^2	cost(s)
Group 1	online	16.23	11.94	0.102	0.792	81.85
	offline	16.23	11.94	0.102	0.792	82.96
Group 2	online	15.88	11.85	0.106	0.797	24.60
	offline	17.40	12.62	0.106	0.756	91.99
Group 3	online	16.43	12.01	0.112	0.828	24.23
	offline	18.29	12.78	0.112	0.788	94.76
Group 4	online	16.52	12.29	0.101	0.835	24.78
	offline	16.75	12.53	0.103	0.830	93.98
Group 5	online	12.57	9.68	0.106	0.828	25.09
	offline	13.16	10.02	0.109	0.812	102.23

B. PARAMETER ANALYSIS

In this subsection, we analyze the setting of hyper-parameters on OADDL, including the loss weight, the optimizer and the loss function, while keeping other hyper-parameter settings the same as Table 1. As shown in Table 6, the OADDL performance under five sets of loss weights is similar, indicating that the loss weight of SAM-LSTM has little impact on model performance. In Table 7, we compared the effects of three optimizers on the model which include RMSProp (root mean square prop), Adam and Nadam (RMSProp with Nesterov momentum). As can be seen from Table 7, when the optimizers of FCN are the same and SAM_LSTM uses different optimizers, the performance of OADDL is similar. On the contrary, OADDL shows different performance. For the FCN, the performance of optimizer RMSProp is the worst, and Adam outperforms Nadam. The reason for the above phenomenon is that the characteristics of the three optimizers are different. RMSProp stabilizes the model's representation of common features and allows it to rapidly catch the representation of rare features. Momentum has the advantage of accelerating gradient descent learning along dimensions and slowing it along turbulent dimensions where the gradient is significantly oscillating. Therefore, Adam and Nadam have more obvious advantages. In Table 8, the similar effects are shown in OADDL under different loss function combinations, including MSE (mean square error), MAE (mean absolute error) and MSLE (mean square logarithmic error). From the above results, the hyper-parameter selection of FCN has more impact on the performance in OADDL. The reason is that FCN makes the final output, which has a certain correction effect on the output of SAM_LSTM, thus affecting the performance of OADDL.

TABLE 6. The performance comparison with different SAM_LSTM loss weights.

loss weight	RMSE	MAE	MAPE	R^2
[1.0,0.4]	12.56	9.67	0.106	0.829
[1.0,0.6]	12.57	9.68	0.106	0.828
[1.0,0.8]	12.88	10.06	0.113	0.820
[1.0,1.0]	12.71	9.85	0.110	0.825
[1.0,1.2]	12.54	9.65	0.106	0.829

TABLE 7. The performance comparison with different optimizers.

SAM_LSTM	FCN	RMSE	MAE	MAPE	R^2
Adam	Adam	12.57	9.68	0.106	0.828
Adam	Nadam	13.42	10.60	0.121	0.805
Adam	RMSprop	14.70	11.81	0.137	0.766
Nadam	Adam	12.59	9.72	0.107	0.828
Nadam	Nadam	13.70	10.84	0.124	0.796
Nadam	RMSprop	15.48	12.49	0.145	0.740
RMSprop	Adam	12.58	9.59	0.103	0.828
RMSprop	Nadam	13.77	10.91	0.125	0.794
RMSprop	RMSprop	15.29	12.91	0.143	0.747

C. SENSITIVITY ANALYSIS

Further, we conduct sensitivity analysis on the hyper-parameters of OADDL, including the loss function, learning rate and batch size, and other hyper-parameter settings are the same as Table 1. We train the OADDL model with 200 epochs, and the loss function curve of OADDL is shown in Figure 10, with the left and right vertical axes corresponding to loss values of SAM_ LSTM and FCN, respectively. Starting from 20 epochs, the loss of SAM_LSTM and FCN show little or only slight fluctuations, which means that two models complete convergence at 20 epochs. Thus, we set the epochs of these two models to 20 in the experiments. There are two optional values for setting the learning rate, which are 0.01 and 0.001, respectively. As shown in Table 9, when these two network learning rates are both set to 0.001, the

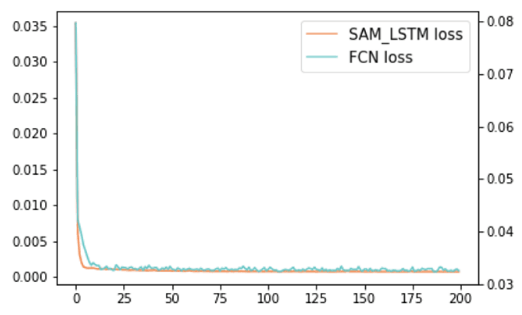

FIGURE 10. The training loss function curves of SAM_LSTM and FCN.

effect is the best. While the learning rate is set too large, the gradient may fluctuate back and forth near the minimum value, and may even fail to converge. Table 10 shows the performance of OADDL using different online batch size b, $b \in \{16, 32, 64, 128, 256\}$. It can be seen that OADDL has comparable or better performance than other competitive sizes at $b = 64$. The reason is that when the batch size is too small, it takes too much time, and the gradient oscillation is serious, which is not conducive to convergence. On the contrary, more training time is needed to converge.

TABLE 8. The performance comparison with different loss functions.

SAM_LSTM	FCN	RMSE	MAE	MAPE	R^2
MSE	MSE	12.52	9.68	0.106	0.830
MSE	MAE	12.50	9.59	0.104	0.830
MES	MSLE	12.58	9.72	0.107	0.828
MAE	MSE	12.93	10.11	0.114	0.819
MAE	MAE	12.58	9.63	0.105	0.828
MAE	MSLE	12.64	9.59	0.102	0.827
MSLE	MSE	12.73	9.93	0.111	0.824
MSLE	MAE	12.57	9.68	0.106	0.828
MSLE	MSLE	12.69	9.89	0.108	0.825

TABLE 9. The performance comparison with different learning rates.

SAM_LSTM	FCN	RMSE	MAE	MAPE	R^2
0.01	0.01	12.88	10.03	0.111	0.820
0.01	0.001	12.58	9.70	0.106	0.828
0.001	0.01	12.92	10.04	0.111	0.819
0.001	0.001	12.57	9.68	0.106	0.828

TABLE 10. The performance comparison with different batch sizes.

SAM_LSTM	FCN	RMSE	MAE	MAPE	R^2
16	16	13.08	9.90	0.103	0.814
32	32	12.68	9.83	0.109	0.825
64	64	12.57	9.68	0.106	0.828
128	128	13.79	10.77	0.121	0.794
256	256	13.41	10.05	0.106	0.805

D. COMPLEXITY ANALYSIS

The OADDL model structure is shown in Figure 11. In the OADDL model, only one attention layer is in the SAM module, which mainly includes four layers, namely Permute, Dense, Permute, and Multiply in turn. The Permute layers are used for dimensional transformation. The Dense layer includes (6,60) neurons. The Multiply layer calculates the product of the original input and the output of the second Permute layer. The Differential and Decomposed LSTM is a 5-layer network, and the first layer is the LSTM layer. LSTM has 100 neurons and each neuron has forgetting gate, input gate, and output gate. The second to four layers are all Dense layers, containing 50, (24, 1), and 1 neurons, respectively. In the third layer, 24 neurons predict the trend component and connect to the fourth layer, while 1 neuron predicts the residual component. The FCN module is a network composed of four layers of Dense modules, each layer containing neurons 100, 50, 24, and 1, respectively. The parameters and FLOPs (floating point operations) of each layer in the OADDL model are shown in Table 11. Obviously, the LSTM layer has the highest number of parameters and FLOPs, with 42800 and 84800, respectively. The total number of parameters in the model is 59509, and the total FLOPs of the model are 130644. In summary, due to the parameter size being less than 10M and FLOPs being less than 10G, the model belongs to small-scale and low computational models. Therefore, the OADDL model has low spatial and temporal complexity.

TABLE 11. Parameters and FLOPs of the OADDL.

Layers	Output	Parameters	FLOPs
Attention	(60,6)	3660	7200
LSTM	100	42800	84800
Dense_1	50	5050	10000
Dense_2	24	1224	2400
Dense_3	1	25	48
Dense_4	1	51	100
Dense_5	100	400	600
Dense_6	50	5050	10000
Dense_7	24	1224	2400
Dense_8	1	25	48

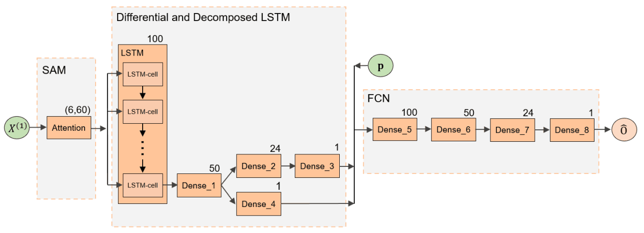

FIGURE 11. The model structure of the OADDL.

VI. CONCLUSION AND FUTURE WORK

In this paper, we propose an online attention enhanced differential and decomposed LSTM model (OADDL) to address the problems of low accuracy and prediction lag in time series. In this model, we combine differentiation, decomposition, self-attention module, LSTM and FCN networks to comprehensively capture the core features and important structures of time series. First, the attention enhanced SAM_LSTM takes the original time series and new time series generated after differentiation and decomposition as inputs in the first training stage, and the trained model outputs the prediction results of trend and remainder. Then, FCN is connected with SAM_LSTM in a cascade manner, taking the predicted result of SAM_LSTM and the original period as its input in the second training stage, and the predicted result of FCN is the predicted result of the time series. The above model adopts online training mode, which can be dynamically updated in real-time and capture the diversity and historical features of time series data. The experiment results show that the proposed model is superior to the latest typical deep learning models in performance on three typical datasets. That is, our OADDL can better fit nonlinear time series and alleviate the lag of prediction. Further, the results of ablation research also demonstrate that differential, decomposition, and two-stage training modes improve the performance of OADDL in different degrees, and none of them are indispensable. In the future work, we will expand and improve the existing model, including the introduction of model parallel and ensemble learning, or deep reinforcement learning technology, and combined with expert domain knowledge, so as to achieve accurate and consistent long-term prediction of non-stationary time series data.

REFERENCES

[1] K. Miyaguchi and H. Kajino, "Cogra: Concept-driftaware stochastic gradient descent for time-series forecasting," in *Proc. 33rd AAAI Conf. Artif. Intell.*, 2019, vol. 33, no. 1, pp. 4594–4601.

[2] H.-F. Yang, T. S. Dillon, and Y. P. Chen, "Optimized structure of the traffic flow forecasting model with a deep learning approach," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2371–2381, Oct. 2017.

[3] M. H. Amini, A. Kargarian, and O. Karabasoglu, "ARIMA-based decoupled time series forecasting of electric vehicle charging demand for stochastic power system operation," *Electric Power Syst. Res.*, vol. 140, pp. 378–390, Nov. 2016.

[4] M. D. Olagoke, A. A. Ayeni, and M. A. Hambali, "Short term electric load forecasting using neural network and genetic algorithm," *Int. J. Appl. Inf. Syst.*, vol. 10, no. 4, pp. 22–28, Jan. 2016.

[5] T. Liu, S. Chen, S. Liang, S. Gan, and C. J. Harris, "Fast adaptive gradient RBF networks for online learning of nonstationary time series," *IEEE Trans. Signal Process.*, vol. 68, pp. 2015–2030, 2020.

[6] X. Jin, N. Yang, X. Wang, Y. Bai, T. Su, and J. Kong, "Integrated predictor based on decomposition mechanism for $PM_{2.5}$ long-term prediction," *Appl. Sci.*, vol. 9, no. 21, p. 4533, Oct. 2019.

[7] L. Xu, Q. Li, J. Yu, L. Wang, J. Xie, and S. Shi, "Spatio-temporal predictions of SST time series in China's offshore waters using a regional convolution long short-term memory (RC-LSTM) network," *Int. J. Remote Sens.*, vol. 41, no. 9, pp. 3368–3389, May 2020.

[8] R. Chandra and S. Chand, "Evaluation of co-evolutionary neural network architectures for time series prediction with mobile application in finance," *Appl. Soft Comput.*, vol. 49, pp. 462–473, Dec. 2016.

[9] H. Hu, L. Tang, S. Zhang, and H. Wang, "Predicting the direction of stock markets using optimized neural networks with Google Trends," *Neurocomputing*, vol. 285, pp. 188–195, Apr. 2018.

[10] W. Li, Y. Guo, B. Wang, and B. Yang, "Learning spatiotemporal embedding with gated convolutional recurrent networks for translation initiation site prediction," *Pattern Recognit.*, vol. 136, Apr. 2023, Art. no. 109234.

[11] Y. Guo, D. Zhou, X. Ruan, and J. Cao, "Variational gated autoencoder-based feature extraction model for inferring disease-miRNA associations based on multiview features," *Neural Netw.*, vol. 165, pp. 491–505, Aug. 2023.

[12] Y. Guo, D. Zhou, P. Li, C. Li, and J. Cao, "Context-aware Poly(A) signal prediction model via deep spatial–temporal neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Dec. 12, 2024, doi: 10.1109/TNNLS.2022.3226301.

[13] S. Yang, D. Zhou, J. Cao, and Y. Guo, "Rethinking low-light enhancement via transformer-GAN," *IEEE Signal Process. Lett.*, vol. 29, pp. 1082–1086, 2022.

[14] S. Yang, D. Zhou, J. Cao, and Y. Guo, "LightingNet: An integrated learning method for low-light image enhancement," *IEEE Trans. Comput. Imag.*, vol. 9, pp. 29–42, 2023.

[15] R. Hou, B. Xu, T. Ren, and G. Wu, "MTNet: Learning modality-aware representation with transformer for RGBT tracking," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2023, pp. 1163–1168.

[16] S. Xie, W. Qian, R. Nie, D. Xu, and J. Cao, "GAGCN: Generative adversarial graph convolutional network for non-homogeneous texture extension synthesis," *IET Image Process.*, vol. 17, no. 5, pp. 1603–1614, Apr. 2023.

[17] F. Lombardi, A. Muti, L. Aniello, R. Baldoni, S. Bonomi, and L. Querzoni, "Pascal: An architecture for proactive auto-scaling of distributed services," *Future Gener. Comput. Syst.*, vol. 98, pp. 342–361, Sep. 2019.

[18] V. Rampérez, J. Soriano, D. Lizcano, and J. A. Lara, "FLAS: A combination of proactive and reactive auto-scaling architecture for distributed services," *Future Gener. Comput. Syst.*, vol. 118, pp. 56–72, May 2021.

[19] H. Hoeltgebaum, N. Adams, and C. Fernandes, "Estimation, forecasting, and anomaly detection for nonstationary streams using adaptive estimation," *IEEE Trans. Cybern.*, vol. 52, no. 8, pp. 7956–7967, Aug. 2022.

[20] Y. Zhou, H. Ren, Z. Li, N. Wu, and A. M. Al-Ahmari, "Anomaly detection via a combination model in time series data," *Int. J. Speech Technol.*, vol. 51, no. 7, pp. 4874–4887, Jul. 2021.

[21] H. T. T. Thuy, D. T. Anh, and V. T. N. Chau, "Efficient segmentation-based methods for anomaly detection in static and streaming time series under dynamic time warping," *J. Intell. Inf. Syst.*, vol. 56, no. 1, pp. 121–146, Feb. 2021.

[22] Z. Shi, Y. Bai, X. Jin, X. Wang, T. Su, and J. Kong, "Parallel deep prediction with covariance intersection fusion on non-stationary time series," *Knowl.-Based Syst.*, vol. 211, Jan. 2021, Art. no. 106523.

[23] W. Zhang, "POLA: Online time series prediction by adaptive learning rates," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 3375–3379.

[24] Y. Wang, C. Wang, C. Shi, and B. Xiao, "Short-term cloud coverage prediction using the ARIMA time series model," *Remote Sens. Lett.*, vol. 9, no. 3, pp. 274–283, Mar. 2018.

[25] Y.-T. Bai, X.-Y. Wang, X.-B. Jin, Z.-Y. Zhao, and B.-H. Zhang, "A neuron-based Kalman filter with nonlinear autoregressive model," *Sensors*, vol. 20, no. 1, p. 299, Jan. 2020.

[26] J. Xue, S. Zhou, Q. Liu, X. Liu, and J. Yin, "Financial time series prediction using $\ell_{2,1}$RF-ELM," *Neurocomputing*, vol. 277, pp. 176–186, Feb. 2018.

[27] C. Yang, J. Qiao, L. Wang, and X. Zhu, "Dynamical regularized echo state network for time series prediction," *Neural Comput. Appl.*, vol. 31, no. 10, pp. 6781–6794, Oct. 2019.

[28] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.

[29] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[30] H. Bohan and B. Yun, "Traffic flow prediction based on BRNN," in *Proc. IEEE 9th Int. Conf. Electron. Inf. Emergency Commun. (ICEIEC)*, Jul. 2019, pp. 320–323.

[31] H. Wang, M. Li, and X. Yue, "IncLSTM: Incremental ensemble LSTM model towards time series data," *Comput. Electr. Eng.*, vol. 92, Jun. 2021, Art. no. 107156.

[32] L. Shen, Y. Wei, and Y. Wang, "GBT: Two-stage transformer framework for non-stationary time series forecasting," *Neural Netw.*, vol. 165, pp. 953–970, Aug. 2023.

[33] D. Ostroski, K. Slovenec, I. Brajdic, and M. Mikuc, "Anomaly correction in time series data for improved forecasting," in *Proc. 16th Int. Conf. Telecommun. (ConTEL)*, Jun. 2021, pp. 85–88.

[34] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "Statistical and machine learning forecasting methods: Concerns and ways forward," *PLoS ONE*, vol. 13, no. 3, Mar. 2018, Art. no. e0194889.

[35] J. Cadzow, "ARMA time series modeling: An effective method," *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-19, no. 1, pp. 49–58, Jan. 1983.

[36] D. Ding, J. Yuan, and M. R. Jovanovic, "Discounted online Newton method for time-varying time series prediction," in *Proc. Amer. Control Conf. (ACC)*, 2021, pp. 1547–1552.

[37] J. Liu, T. Sun, Y. Luo, S. Yang, Y. Cao, and J. Zhai, "An echo state network architecture based on quantum logic gate and its optimization," *Neurocomputing*, vol. 371, pp. 100–107, Jan. 2020.

[38] Y. Xu, J. Zhang, Z. Long, and Y. Chen, "A novel dual-scale deep belief network method for daily urban water demand forecasting," *Energies*, vol. 11, no. 5, p. 1068, Apr. 2018.

[39] J. F. Torres, D. Hadjout, A. Sebaa, F. Martínez-Álvarez, and A. Troncoso, "Deep learning for time series forecasting: A survey," *Big Data*, vol. 9, no. 1, pp. 3–21, Feb. 2021.

[40] S. Hochreiter and J. Schmidhuber, "LSTM can solve hard long time lag problems," in *Proc. Adv. Neural Inf. Process. Syst.*, 1997, pp. 473–479.
[41] P. B. Weerakody, K. W. Wong, G. Wang, and W. Ela, "A review of irregular time series data handling with gated recurrent neural networks," *Neurocomputing*, vol. 441, pp. 161–178, Jun. 2021.
[42] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *Eur. J. Oper. Res.*, vol. 270, no. 2, pp. 654–669, Oct. 2018.
[43] Y. Tian, K. Zhang, J. Li, X. Lin, and B. Yang, "LSTM-based traffic flow prediction with missing data," *Neurocomputing*, vol. 318, pp. 297–305, Nov. 2018.
[44] O. Anava, E. Hazan, S. Mannor, and O. Shamir, "Online learning for time series prediction," in *Proc. Conf. Learn. Theory*, 2013, pp. 172–184.
[45] E. López, C. Valle, H. Allende, E. Gil, and H. Madsen, "Wind power forecasting based on echo state networks and long short-term memory," *Energies*, vol. 11, no. 3, p. 526, Feb. 2018.
[46] C. Tian, J. Ma, C. Zhang, and P. Zhan, "A deep neural network model for short-term load forecast based on long short-term memory network and convolutional neural network," *Energies*, vol. 11, no. 12, p. 3493, Dec. 2018.
[47] Z. Yu, Z. Niu, W. Tang, and Q. Wu, "Deep learning for daily peak load forecasting—A novel gated recurrent neural network combining dynamic time warping," *IEEE Access*, vol. 7, pp. 17184–17194, 2019.
[48] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, "FedFormer: Frequency enhanced decomposed transformer for long-term series forecasting," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 27268–27286.
[49] M. A. Shabani, A. H. Abdi, L. Meng, and T. Sylvain, "Scaleformer: Iterative multi-scale refining transformers for time series forecasting," in *Proc. 11th Int. Conf. Learn. Represent.*, 2023, pp. 1–23.
[50] W. Katz, J. Snell, and M. Merickel, "Artificial neural networks," *Methods Enzymol*, vol. 210, pp. 610–636, Jan. 1992.
[51] H. B. McMahan, "A survey of algorithms and analysis for adaptive online learning," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 3117–3166, Jan. 2017.
[52] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual lifelong learning with neural networks: A review," *Neural Netw.*, vol. 113, pp. 54–71, May 2019.
[53] L. Peng, L. Wang, X.-Y. Ai, and Y.-R. Zeng, "Forecasting tourist arrivals via random forest and long short-term memory," *Cognit. Comput.*, vol. 13, no. 1, pp. 125–138, Jan. 2021.
[54] R. Hyndman and G. Athanasopoulos, *Forecasting: Principles and Practice*. Melbourne, VIC, Australia: OTexts, 2018.
[55] E. Dagum and S. Bianconcini, *Seasonal Adjustment Methods and Real Time Trend-Cycle Estimation*. Berlin, Germany: Springer, 2016.
[56] R. Clevel, "STL: A seasonal-trend decomposition procedure based on loess," *Neural Netw.*, vol. 6, no. 1, pp. 3–33, 1990.
[57] J. Zhou, Z. Liang, Y. Liu, H. Guo, D. He, and L. Zhao, "Six-decade temporal change and seasonal decomposition of climate variables in Lake Dianchi watershed (China): Stable trend or abrupt shift?" *Theor. Appl. Climatol.*, vol. 119, nos. 1–2, pp. 181–191, Jan. 2015.
[58] L. Qin, W. Li, and S. Li, "Effective passenger flow forecasting using STL and ESN based on two improvement strategies," *Neurocomputing*, vol. 356, pp. 244–256, Sep. 2019.
[59] K. Wang, D. Niu, L. Sun, H. Zhen, J. Liu, G. De, and X. Xu, "Wind power short-term forecasting hybrid model based on CEEMD-SE method," *Processes*, vol. 7, no. 11, p. 843, Nov. 2019.

LINA LI received the M.S. degree from the School of Computer Science & Technology, Harbin Institute of Technology, Harbin, in 2006, and the Ph.D. degree in computer system architecture from the College of Computer Science and Technology (CCST), Jilin University, Changchun, in 2019. She is currently an Associate Professor with the College of Computer Science and Technology, Changchun University. Her research interests include blockchain, deep learning, and cloud computing.

SHENGKUI HUANG was born in Xuzhou, Jiangsu, China, in 2001. He is currently pursuing the bachelor's degree with Changchun University, Jilin, China. His research interests include big data mining and deep learning. He won the Second Prize in Jilin Mathematical Modeling Competition, in 2021, and has applied for a patent and published a paper on IEEE International Conferences. He is also participating in a project funded by the Natural Science Foundation of Jilin Province.

GUOXING LIU was born in Jining, Shandong, China, in 1999. He is currently pursuing the master's degree with Changchun University. He has participated in an application for the National Social Science Foundation on the topic of public opinion prediction and has successfully published a paper at an EI conferences on the adaptive propagation model of network hotspot events. His research interests include deep learning, NLP, and public opinion prediction.

CHENG LUO was born in Tongren, Guizhou, China, in 1999. He is currently pursuing the Graduate degree with the College of Computer Science and Technology, Changchun University. His research interest includes deep learning. He has conducted some research on time series prediction. He won the Third Prize of Jilin Provincial Mathematical Modeling Competition and the Third Prize of Jilin Provincial Division of the National Mathematical Modeling Competition, in 2019.

QINGHE YU was born in Fuyang, Anhui, China, in 1997. He is currently pursuing the master's degree with the College of Computer Science and Technology, Changchun University, Jilin, China. He has participated in a provincial scientific research project based on deep reinforcement learning technology and has conducted some research on cloud-edge-end resource scheduling optimization. His research interests include deep learning, cloud computing, and edge computing.

NIANFENG LI received the Ph.D. degree in mechatronics Engineering from the Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences. He is currently a Professor and the Dean of the College of Computer Science and Technology (CCST), Changchun University, China, where he is also the Vice Director of the Biomedical Engineering Research and Development Center. His current major research interests include image processing, somatosensory technology, rehabilitation training systems, and campus safety technology.