IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 27, 2020, accepted March 14, 2020, date of publication March 17, 2020, date of current version April 14, 2020.

*Digital Object Identifier 10.1109/ACCESS.2020.2981506*

# A Hybrid Method With Adaptive Sub-Series Clustering and Attention-Based Stacked Residual LSTMs for Multivariate Time Series Forecasting

**FAGUI LIU, YUNSHENG LU, AND MUQING CAI**

School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China

Corresponding author: Yunsheng Lu (csyunshenglu@mail.scut.edu.cn)

This work was supported in part by the Engineering and Technology Research Center of Guangdong Province for Logistics Supply Chain and Internet of Things under Project GDDST[2016]176, in part by the Key Laboratory of Cloud Computing for Super-integration Cloud Computing in Guangdong Province under Project 610245048129, in part by the Engineering and Technology Research Center of Guangdong Province for Big Data Intelligent Processing under Project GDDST[2013]1513-1-11, in part by the "Made in China 2025" Industrial Development Fund Project of Guangzhou under Project x2jsD8183470, and in part by the Major Program of Guangdong Basic and Applied Research under Project 2019B030302002.

**ABSTRACT** Multivariate Time Series Forecasting (MTSF) has recently emerged its growing importance in many industries. However, how to reduce the influence of the noise components existing in time series on prediction and extract features effectively are still two challenges in MTSF. This paper focuses on those two challenges and proposes a new prediction method based on decomposition-ensemble framework called adaptive sub-series clustering-stacked residual LSTMs-multi-level attention mechanism (ASC-SRLSTMs-MLAttn). The method consists of three stages: decomposition, prediction, and ensemble. In the decomposition stage, the target series is decomposed by Ensemble Empirical Mode Decomposition into multiple sub-series, which will be clustered and reconstructed by the ASC algorithm to reduce the complexity and the time consumption of prediction. In the prediction stage, the sub-series and correlation series will be fed into SRLSTMs-MLAttn for sub-series prediction. The model is based on the encoder-decoder architecture with stacked residual LSTMs as the encoder, which can effectively capture the dependencies among multi variables and the temporal features from multivariate time series. Besides, a multi-level attention mechanism (MLAttn), which makes full use of the encoding information of the encoder, has been introduced to further improve the prediction performance of the model. In the ensemble stage, the predicted values of each sub-series will be summed to obtain the final prediction of the target time series. We also demonstrate the superiority and effectiveness of our proposed method on four public datasets via the conducted comparison experiment and ablation study.

**INDEX TERMS** Multivariate time series forecasting, decomposition-ensemble framework, adaptive sub-series clustering algorithm, stacked residual long short-term memory, multi-level attention mechanism.

## I. INTRODUCTION

The recent increasing attention attracted towards multivariate time series forecasting (MTSF) roots from its potential application into many fields. For example, in the field of energy, wind speed forecasting can help wind power plants to effectively dispatch power resources [1]. In the field of traffic engineering, accurate road traffic flow forecasting as part of intelligent transportation system can solve the traffic problems caused by high load [2]. In the field of environment, research on air quality forecasting is also an important means to guide the scientific decision-making of severe air pollution warning and air pollution control [3]. However, MTSF is not a simple matter. How to deal with the noise components and extract features from multivariate time series are the two major challenges to obtain accurate prediction.

As multivariate time series is non-stationary and often contain noise components, data preprocessing is necessary to reduce the impact of noise on prediction [4]. Existing methods mainly include two types: elimination and separation, which are both based on the idea that noise are mainly concentrated in the high frequency part of data to locate

The associate editor coordinating the review of this manuscript and approving it for publication was Li He.

and process the noise components. As for the elimination method [5], [6], the general approach is to decompose the time series into several sub-series with frequency from high to low, then eliminate the noise in the high frequency sub-series according to a certain strategy (such as threshold denoising), and finally reconstruct the sub-series to get the de-noised time series for prediction. As for the separation method [7], [8], also called as decomposition-ensemble framework based prediction method, it consists of three stages: decomposition, prediction and ensemble. In the decomposition stage, the target series is first decomposed into several sub-series by means of time series decomposition methods. Then, in the prediction stage, each sub-series is modeled and predicted by a prediction model individually. Finally, in the ensemble stage, prediction results of all sub-series are reconstructed to obtain the final prediction results. Among all time series decomposition methods, Ensemble Empirical Mode Decomposition (EEMD) [9] is widely used because of its remarkable self-adaptability and time-frequency resolution. In spite of the fact that noise can be effectively eliminated by the elimination method, it may also lead to the loss of useful information in time series. Especially in MTSF problems, components seem to be noise in a single variable time series may be driven by other variables. By contrast, the separation method can not only avoid the information loss, but also simplify the difficulty of time series modeling by dividing a single complex forecasting task into several relatively simple sub-tasks, and reduce the impact of noise on the overall prediction effect [10]. However, as it needs to model several sub-series obtained by decomposition, separation type method is suffering from problem of time-consuming. To address this problem, we propose the Adaptive Sub-series Clustering algorithm (ASC) based on decomposition-ensemble prediction framework. Sample Entropy [11] is introduced to describe the complexity of sub-series and K-means is used to adaptively cluster and reconstruct sub-series with approximate complexity into new sub-series for further analysis in the prediction stage, which can ensure the effective separation of noise components and reduce the total time consumption of prediction.

How to effectively extract temporal features and capture dependencies among multiple variables is also an important issue for MTSF. As a data-driven method with strong nonlinear fitting capability, Artificial Neural Networks (ANNs) [12] have shown great performance in many fields and have been regarded as a good alternative to process-based and statistical approaches for MTSF. Among various types of ANNs, Recurrent Neural Networks (RNNs) have been proved to be very suitable for time series modeling because of their excellent ability for sequence processing. Encoder-decoder architecture based models [13], by introducing two RNN as the encoder for feature extraction and the decoder for feature analysis, which shows better prediction performance than a single RNN. However, they are still lack of capacity to capture complex dependencies among multiple variables. Although it can be improved by adding CNN [14], [15] or auto-encoder (AE) [6], [16] to the model, the parameter settings of CNN such as the size of convolution kernel and the number of convolution layers are closely related to the number of variables, making CNN difficult to be applied in different datasets. Besides, the training process of AE and the predictor is separated, which also easily leads to unsatisfactory prediction results. It is widely known that deeper neural network have higher performance [17]. Therefore, by stacking several LSTMs in the vertical direction to form a deep structure similar to feedforward neural network, it can obtain more complex feature learning ability. However, as the activation functions in LSTM are saturated, stacked LSTMs is difficult to train as the number of layers increases due to the degradation problem. To solve the above problem, we introduce residual connection mechanism [18] for stacked LSTMs (SLSTMs), which improves the efficiency of gradient propagation between layers and eases the network optimization. In this paper, we proposed a prediction model based on the encoder-decoder architecture. The model uses stacked residual LSTMs (SRLSTMs) as the encoder to extract the temporal features and capture the dependencies among multiple variables effectively.

Although encoder-decoder architecture based models achieve good performance, as the encoder can only encode information into fixed-length vectors, the performance of the model decreases rapidly with the increase of the length of input sequence or output sequence. To tackle this problem, the attention mechanism [19] is proposed. It enables the decoder to adaptively select the hidden states of the past and extract the useful information. In this paper, to further improve the prediction performance of the model, we propose a new attention mechanism called multi-level attention mechanism (MLAttn). Based on the traditional temporal attention mechanism, it also fully considers the semantic information extracted by SRLSTMs.

The main contributions of this paper are as follows:

1) A prediction method based on decomposition-ensemble prediction framework named ASC-SRLSTMs-MLAttn is proposed for MTSF. It has achieved superior performance in experiments on multiple datasets.

2) To alleviate the efficiency problem of decomposition-ensemble framework, the Adaptive Sub-series Clustering algorithm (ASC) is proposed to ensure the separation of noise and reduces the time consumed by the whole prediction.

3) To capture the dependencies among multiple variables and temporal features effectively, a prediction model based on the encoder-decoder architecture is proposed. Stacked LSTMs is used as the encoder, and the introduction of residual connection solves the degradation problem of SLSTMs.

4) To further improve the performance of the prediction model, the multi-level attention mechanism (MLAttn) is proposed. It can alleviate the information loss and fully considers semantic features across all levels of SRLSTMs.

The remainder of this paper is organized as follows: Section II will introduce the related work of time series forecasting and the related methods using the

decomposition-ensemble prediction framework. Section III gives the definition of the multivariable time series forecasting problem to be solved in this paper. Section IV introduces the proposed method ASC-SRLSTMs-MLAttn in detail. Experiment results and discussion are provided in Section V. Section VI will give conclusions and our future research work.

### A. RELATED WORK

In the past, most of the time series prediction methods are based on mathematical statistics models, such as ARIMA [20], ARIMAX [21], VAR [22] and GARCH [23]. However, since most of these models are based on fixed mathematical formula, they are not sufficient to fully represent the characteristics of time series. Moreover, almost all time series are complex and volatile in the real world, making these models difficult to obtain ideal prediction results.

By treating time series forecasting as a generalized regression problem with time-varying parameters, machine learning methods such as Support Vector Regression [24], Kalman filter [25], Random forest [26], Ridge Regression [27] and LASSO regression [28] can also be applied. Although these methods perform well on simple datasets, they still difficult to capture the complex non-linear relationship between multi-variables on large datasets, resulting in poor performance.

Recent years, more and more studies focused on the use of Artificial Neural Networks (ANNs) to solve complex time series forecasting problems because of their powerful non-linear feature extraction ability. Feedforward Neural Networks (FNNs) are the first neural network structure used for time series forecasting, and many studies have shown their performance. Paoli *et al*. [29] constructed a multi-layer perceptron (MLP) for solar radiation intensity prediction, and demonstrated its superiority over statistical models. Chen *et al*. [30] employed an extreme learning machine (ELM) for electricity price forecasting and achieved better results than traditional machine learning approaches. However, due to their simple structure, FNNs are lack of temporal feature extraction ability. Therefore, many studies start to turn to Recurrent Neural Networks (RNNs).

RNN is a kind of neural network for sequence data processing and has outstanding long and short-term dependencies capture capacity. RNN and its variants, including LSTM [31] and GRU [32], have been widely applied for time series forecasting. Coulibaly *et al*. [33] used vanilla RNN for multivariate reservoir inflow forecasting and obtained better performance than MLP. Kong *et al*. [34] employed LSTM as the predictor for electric load forecasting and achieved the best results comparing to BPNN and ELM. Che *et al*. [35] developed a model based on GRU for MTSF with missing values and obtained better prediction results. The models mentioned above are all based on a single RNN, which only have limited temporal feature extraction ability. As a result, some researches introduced encoder-decoder architecture based models for MTSF, which consists of two RNNs for feature encoding and decoding respectively. For example, Mukhoty *et al*. [36] used two LSTM as the encoder and the decoder respectively to build a seq2seq model for multi-step solar irradiation forecasting and got excellent prediction accuracy. Liang *et al*. [37] demonstrated the superiority of the encoder-decoder architecture over single RNN by comparison experiments on two environment quality datasets.

Despite the fact that RNNs based models do well in temporal feature extraction, they are still difficult to capture dependencies among multiple variables. Thus, many researches are devoted to solve this problem and further improve performance of these models. For instance, Sagheer and Kotb [38] proposed a prediction model named DLSTM, which enables LSTM at different levels to capture different levels of information by stacking LSTMs vertically, proving the effectiveness of stacked LSTMs. Tank *et al*. [39] combined RNNs with sparsity penalties on weights to identify non-linear casual variables and enhance their ability to capture long-range dependencies between series. Bao *et al*. [16] utilized stacked auto-encoders (SAEs) for unsupervised multivariate relationship learning to extract high level features and help to train the predictor based on LSTM more efficiently. Lai *et al*. [40] proposed LSTNet, which uses CNN and RNN to extract short-term local dependencies between variables and find long-term patterns of time series trends, respectively. Qin *et al*. [41] proposed a dual-stage attention-based recurrent neural network (DA-RNN) for MTSF. The model is based on the encoder-decoder architecture, and uses input attention mechanism and temporal attention mechanism to adaptively select input features and hidden states of the encoder. Guo and Lin [42] proposed MV-LSTM with tensorized hidden states to give rise to mixture temporal and variable attention. Chang *et al*. [43] proposed the MTNet model, which includes a large memory component composed of multiple LSTMs, three independent coders and attention mechanism to capture spatial and temporal features. Munkhdalai *et al*. [44] proposed AIS-RNN framework, which combines RNNs with an adaptive input selection mechanism to improve prediction performance.

In addition, many works are based on the decomposition-ensemble framework to simplify the complex data, provide satisfactory results for capturing inner factors and improve the prediction accuracy. For example, Conejo *et al*. [45] applied Wavelet Transform (WT) for time series decomposition and ARIMA for individual prediction of subseries, which outperformed the direct use of ARIMA. Prasad *et al*. [46] introduced Ensemble Empirical Mode Decomposition (EEMD) to decompose original time series data of monthly soil moisture, and used ELM as predictors to produced better prediction results. Zhang *et al*. [47] proposed a hybrid approach which combines EEMD and LSTM for daily land surface temperature forecasting to reduce the difficulty of modeling and to improve prediction accuracy. Zhu *et al*. [48] proposed VMD-BiGRU for rubber futures time series forecasting, in which Variational Mode Decomposition (VMD) is utilized to capture the tendency and mutability

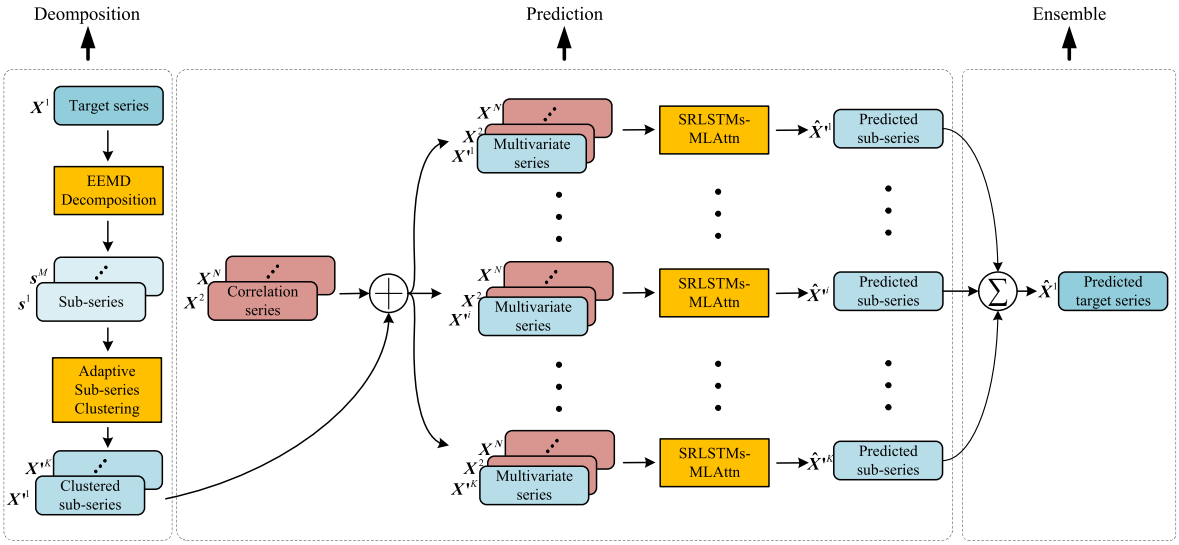



**FIGURE 1.** The prediction flowchart of our proposed method.

information of time series, and BiGRU is to make one-day-ahead prediction. All of these works have proved that hybrid models applying decomposition technology can achieve better performance.

## II. NOTATIONS AND PROBLEM STATEMENTS

### A. NOTATIONS

Univariate time series is a sequence of observations with continuous timestamps for a variable over a period of time. Multivariate time series is a set of multiple univariate time series and can be expressed as $\boldsymbol{X} = \{\boldsymbol{X}^1, \boldsymbol{X}^2, \ldots, \boldsymbol{X}^N\} \in \mathbb{R}^{N\times T}$, where $\boldsymbol{X}^i = \{x_1^i, x_2^i, \ldots, x_T^i\} \in \mathbb{R}^T$ is the time series of the $i$-th variable, $x_t^i \in \mathbb{R}$ is its observation value at time $t$, $N$ is the number of variables, and $T$ is the size of the time window. In this study, we choose $\boldsymbol{X}^1$ as the target series to predict, and other series as correlation series which provide relevant features to assist the prediction.

### B. PROBLEM STATEMENTS

Given a multivariate time series $\boldsymbol{X} = \{\boldsymbol{X}^1, \boldsymbol{X}^2, \ldots, \boldsymbol{X}^N\} \in \mathbb{R}^{N\times T}$, multivariate time series forecasting is aimming at predicting the values of target series at the next $P$ time stamps according to $\boldsymbol{X}$:

$$\hat{\boldsymbol{X}} = F(\boldsymbol{X}) \tag{1}$$

where $\hat{\boldsymbol{X}} = \{\hat{x}_1^1, \hat{x}_2^1, \ldots, \hat{x}_P^1\} \in \mathbb{R}^P$ is the sequence of predicted values of the target series, $T$ is the look back size of the prediction, $P$ is the horizon of prediction and $F(\cdot)$ is the nonlinear function we aim to find.

## III. PROPOSED METHOD

Figure 1 shows the prediction flowchart of our method proposed in this paper, which consists of three stages: decomposition, prediction and ensemble. In the decomposition stage, EEMD is utilized to decompose the target series into multiple sub-series, and ASC algorithm is applied to cluster and reconstruct sub-series into several new sub-series to reduce the total time consumption of prediction. In the prediction stage, sub-series and correlation series are fed into SRLSTMs-MLAttn to get predictions for sub-series. In the ensemble stage, all the predicted values of sub-series are accumulated to obtain the final prediction of the target series.

### A. EEMD DECOMPOSITION

We first introduce Ensemble Empirical Mode Decomposition (EEMD) to decompose the complex target series into $M$ simple sub-series to simplify the modeling complexity. EEMD is a data-driven and adaptive tool for non-linear and non-stationary signals analysis, which is proposed by Wu and Huang [9]. It can adaptively decompose time series into multiple sub-series called Intrinsic Mode Functions (IMFs) and Residue (R) with frequency from high to low. As noise mainly concentrates in the high-frequency domain, EEMD can effectively separate the noise from useful components. Assume that the original time series is $\boldsymbol{X} = \{x_1, x_2, \ldots, x_T\} \in \mathbb{R}^T$ and $\boldsymbol{\omega}^i \in \mathbb{R}^T$ $(i = 1, \ldots, I)$ is the white Gaussian noise added at $i$-th time. The detailed procedures of EEMD are as follows:

*Step 1*: Add white Gaussian noise $\boldsymbol{\omega}^i$ to $\boldsymbol{X}$ and obtain the noisy series $\boldsymbol{X}^i$:

$$\boldsymbol{X}^i = \boldsymbol{X} + \boldsymbol{\omega}^i \tag{2}$$

*Step 2*: Use EMD to decompose $\boldsymbol{X}^i$, and then obtain $M-1$ IMFs $\boldsymbol{c}^{i,j} \in \mathbb{R}^T (j = 1, \ldots, M-1)$ and one Residue $\boldsymbol{r}^i \in \mathbb{R}^T$:

$$\boldsymbol{X}^i = \sum_{j=1}^{M-1} \boldsymbol{c}^{i,j} + \boldsymbol{r}^i \tag{3}$$

*Step 3*: Repeat *Step 1* and *Step 2*, and add different white noise each time until the number of iterations reaches the

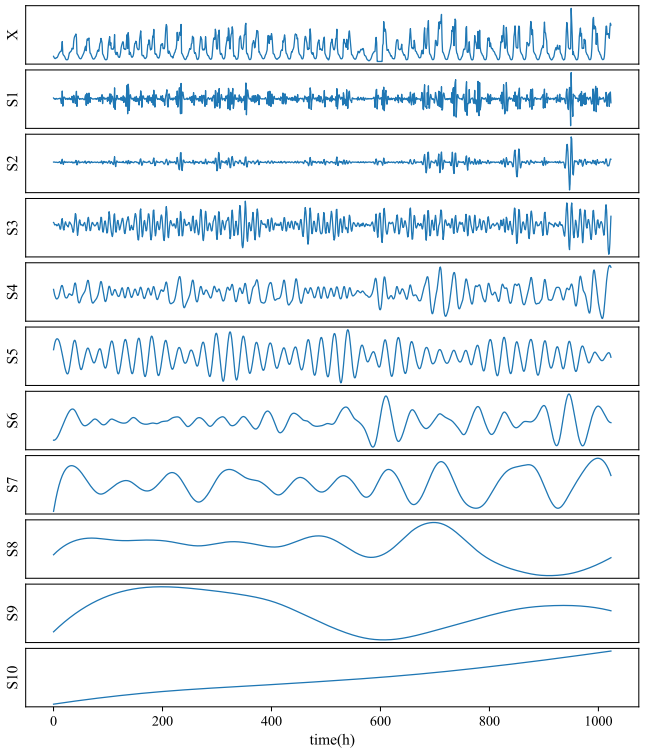



**FIGURE 2.** Results of the EEMD. $X$ is the original time series and $S^1 - S^{10}$ are obtained sub-series.

maximum $I$. In this study, we set $I$ to 200 considering the tradeoff between effect and time consumption.

*Step 4*: Calculate the mean of the ensemble IMFs and R, and obtain the final decomposition results:

$$\mathbf{IMF}^j = \frac{1}{I}\sum_{i=1}^{I} \boldsymbol{c}^{i,j} \quad (j = 1, \ldots, M-1) \tag{4}$$

$$\mathbf{R} = \frac{1}{I}\sum_{i=1}^{I} \boldsymbol{r}^{i} \tag{5}$$

where $\mathbf{IMF}^j \in \mathbb{R}^T (j = 1, \ldots, M-1)$ are the final $M-1$ IMFs and $\mathbf{R} \in \mathbb{R}^T$ is the final Residue. $M$ is related to the length of the original time series $X$.

For simplicity, we denote IMFs and Residue as $\boldsymbol{s}^m \in \mathbb{R}^T (1 < m < M)$, and the set of sub-series can be expressed as:

$$\boldsymbol{S} = \left\{\boldsymbol{s}^1, \boldsymbol{s}^2, \ldots, \boldsymbol{s}^M\right\} \in \mathbb{R}^{M \times T} \tag{6}$$

Figure 2 shows an example of EEMD decomposition applied to a time series with length of 1024 taken from the Traffic dataset. It can be seen that the original complex time series is simplified after being decomposed into 10 sub-series (9 IMFs and one Residue). However, the overall computation time will be greatly increased if each sub-series is modeled directly. Therefore, clustering and reconstruction are necessary to reduce the number of sub-series and improve the prediction efficiency.

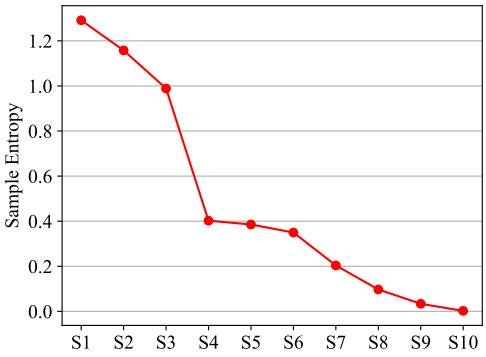



**FIGURE 3.** Sample entropy of all sub-series.

### B. ADAPTIVE SUB-SERIES CLUSTERING

#### 1) SUB-SERIES COMPLEXITY CACULATION

We adopt Sample Entropy (SE) [11] to measure the complexity of each sub-series. The larger the sample entropy is, the higher the complexity of the time series is, and time series with approximate sample entropy contain similar amount of information. In general, for time series $\boldsymbol{X} = \{x_1, x_2, \ldots, x_T\} \in \mathbb{R}^T$, the principle of sample entropy is below:

*Step 1*: Form $\boldsymbol{X}$ into a group of $m$-dimension vectors as follows:

$$\boldsymbol{X}^i = \{x_i, x_{i+1}, \ldots, x_{i+m-1}\} \in \mathbb{R}^T \quad (i = 1, \ldots, T-m+1) \tag{7}$$

*Step 2*: Define the $d(\boldsymbol{X}^i, \boldsymbol{X}^j)$ as the maximum difference of corresponding elements between $\boldsymbol{X}^i$ and $\boldsymbol{X}^j$:

$$d(\boldsymbol{X}^i, \boldsymbol{X}^j) = \max_{k=0,\ldots,m-1} (|x_{i+k} - x_{j+k}|) \tag{8}$$

*Step 3*: Count the total number of $d(\boldsymbol{X}^i, \boldsymbol{X}^j) < r$ of every i value and denote it as $B_i^m$:

$$B_i^m = \frac{1}{T-m+1} \text{num}_{j \neq i} \left\{d(\boldsymbol{X}^i, \boldsymbol{X}^j) < r\right\} \tag{9}$$

and get the mean value of $B_i^m$:

$$B^m = \frac{1}{T-m}\sum_{i=1}^{T-m} B_i^m \tag{10}$$

*Step 4*: Replace the $m$ as $m+1$, repeat *Step 1-3* and get the mean value of $B^{m+1}$:

$$B^{m+1} = \frac{1}{T-m-1}\sum_{i=1}^{T-m-1} B_i^{m+1} \tag{11}$$

The sample entropy of $\boldsymbol{X}$ can be expressed as follows:

$$SE(m, r, \boldsymbol{X}) = -\ln \frac{B^{m+1}}{B^m} \tag{12}$$

According to the general recommendation, the value of the initial dimension $m$ is set to 2 and the similarity tolerance $r$ is set at the $0.1 - 0.25 * std(\boldsymbol{X})$ range, where $std(\boldsymbol{X})$

**Algorithm 1** The Adaptive Sub-Series Clustering Algorithm

**Input:**
The set of sub-series $\boldsymbol{S} = \{\boldsymbol{s}^1, \boldsymbol{s}^2, \ldots, \boldsymbol{s}^M\}$;
Sample Entropy of all sub-series $\boldsymbol{E} = \{e_1, e_2, \ldots, e_M\}$;
The number of cluster $K$;

Select $K$ values from $\boldsymbol{E}$ at a fixed interval $d = \lfloor \frac{M}{K} \rfloor$ as initial mean values $\{\mu_1, \mu_2, \ldots, \mu_K\}$;
**repeat**
  Let $\boldsymbol{C}_i = \varnothing (1 \leqslant i \leqslant K)$ and $\boldsymbol{C}'_i = \varnothing (1 \leqslant i \leqslant K)$;
  **for** $j = 1, 2, \ldots, M$ **do**
    Calculate the distance between $e_j$ and each mean value $\mu_i$: $dis_{ji} = |e_j - \mu_i|$;
    Determine the cluster index of $\boldsymbol{s}^j$ according to the nearest mean value: $\lambda_j = \arg\min_{i \in \{1,2,\ldots,K\}} dis_{ji}$;
    Add $\boldsymbol{s}^j$ into the corresponding cluster: $\boldsymbol{C}_{\lambda_j} = \boldsymbol{C}_{\lambda_j} \cup \boldsymbol{s}^j$;
    Add $e_j$ into the corresponding cluster: $\boldsymbol{C}'_{\lambda_j} = \boldsymbol{C}'_{\lambda_j} \cup e_j$;
  **end for**
  **for** $j = 1, 2, \ldots, K$ **do**
    Calculate the new mean value: $\mu'_i = \frac{1}{|\boldsymbol{C}'_i|} \sum_{e \in \boldsymbol{C}'_i} e$;
    **if** $\mu'_i \neq \mu_i$ **then**
      Update the current mean value $\mu_i$ to $\mu'_i$;
    **else**
      Keep the current mean value unchanged;
    **end if**
  **end for**
**until** Current mean values are not changed

**Output:**
The set of sub-series clusters: $\boldsymbol{C} = \{\boldsymbol{C}_1, \boldsymbol{C}_2, \ldots, \boldsymbol{C}_K\}$;

is the standard deviation of $\boldsymbol{X}$. In this study, we set the $m$ to 2 and the $r$ to $0.12 * std(\boldsymbol{X})$ after applying a grid search.

Sample entropy of $M$ sub-series can be caculated according to Eq(7)-(12):

$$\boldsymbol{E} = \{e_1, e_2, \ldots, e_M\} \in \mathbb{R}^M \tag{13}$$

where $e_i = SE(m, r, \boldsymbol{s}^i)(1 \leqslant i \leqslant M)$ is the sample entropy of the $i$-th sub-series.

Figure 3 shows the sample entropy of each sub-series shown in Figure 2. It can be seen that its value decreases with the frequency of the sub-series from high to low, so as the complexity.

#### 2) SUB-SERIES CLUSTERING

K-means is an unsupervised clustering algorithm widely used in data mining. It can divide $n$ measurements into $K$ clusters iteratively. In this stage, K-means is utilized to cluster sub-series into $K$ clusters adaptively according to the values of their sample entropy. The procedures of the ASC algorithm are summarized in Alogirhtm 1 and Table 1 shows the result obtained by applying ASC to sub-series shown in Figure 2 when $K = 4$.

**TABLE 1.** Result of sub-series clustering with ASC when $K = 4$.

| Cluster | $C_1$ | $C_2$ | $C_3$ | $C_4$ |
|---|---|---|---|---|
| Sub-series | $s^1, s^2$ | $s^3$ | $s^4, s^5, s^6$ | $s^7, s^8, s^9, s^{10}$ |

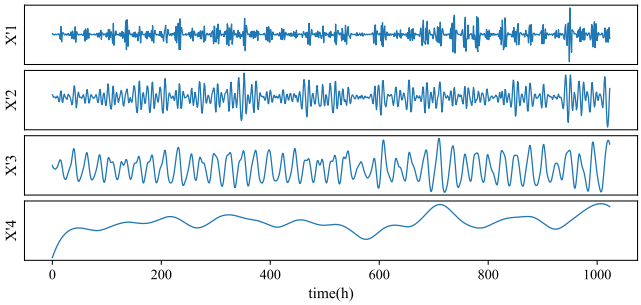



**FIGURE 4.** New sub-series after recontruction when $K = 4$.

#### 3) SUB-SERIES RECONSTRUCTION

By accumulating sub-series in each cluster obtained by ASC, the corresponding new sub-series can be reconstructed:

$$\boldsymbol{X}'^i = \sum_{\boldsymbol{s} \in \boldsymbol{C}_i} \boldsymbol{s} \quad (i = 1, 2, \ldots, K) \tag{14}$$

The obtained new sub-series after reconstruction according to the result show in Table 1 are shown in Figure 4.

### C. PREDICTION MODEL

We proposed the prediction model named SRLSTMs-MLAttn to fully extract features from multivariate time series and make prediction. As shown in Figure 5, the model is based on the encoder-decoder architecture, which uses stacked residual LSTMs (SRLSTMs) as the encoder and single-layer LSTM as the decoder. When predicting, the encoder extracts features from input multivariate time series with a time window of size $T$ and compresses them into context vectors with a fixed size. Then the decoder decodes information from context vectors step by step to get $P$ prediction values. In this section, we will introduce our prediction model in detail.

#### 1) ENCODER

##### a: LONG SHORT-TERM MEMORY

In order to extract temporal features from multivariate time series, we use LSTM as the basic unit of the encoder. LSTM is a variant of Recurrent Neural Network (RNN) with outstanding sequence modeling ability. Like traditional RNN, LSTM updates its hidden state according to the current input to capture long and short-term dependencies of sequence. The difference between LSTM and traditional RNN is the introduction of gated structure, which enables LSTM adaptively memorize or forget historical information so that it can avoid the problem of gradient vanishing or gradient exploding when processing long sequence data. As shown in Figure 6, an LSTM unit contains a memory cell with the state $\boldsymbol{C}_t$ and three gates: forget gate $\boldsymbol{f}_t$, input gate $\boldsymbol{i}_t$ and output gate $\boldsymbol{o}_t$. The update of an LSTM unit with hidden layer size of $H$ at

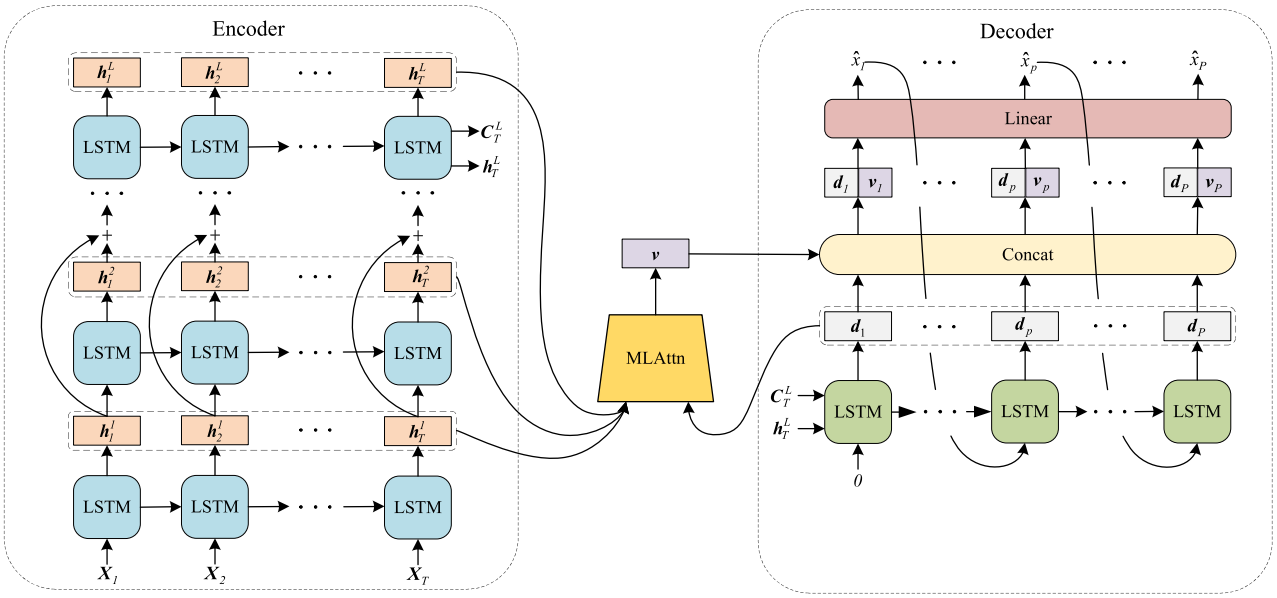



**FIGURE 5.** The proposed prediction model (SRLSTMs-MLAttn).

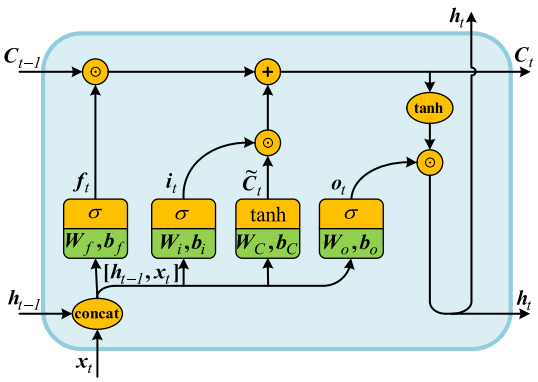



**FIGURE 6.** Structure of an LSTM unit.

time $t$ is summarized as follows:

$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}_f \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_f) \tag{15}$$

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_i \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_i) \tag{16}$$

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_o \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_o) \tag{17}$$

$$\tilde{\boldsymbol{C}}_t = tanh(\boldsymbol{W}_C \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_C) \tag{18}$$

$$\boldsymbol{C}_t = \boldsymbol{C}_{t-1} \odot \boldsymbol{f}_t + \tilde{\boldsymbol{C}}_t \odot \boldsymbol{i}_t \tag{19}$$

$$\boldsymbol{h}_t = tanh(\boldsymbol{C}_t \odot \boldsymbol{o}_t) \tag{20}$$

where $[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] \in \mathbb{R}^{H+N}$ is the concatenation of the previous hidden state $\boldsymbol{h}_{t-1} \in \mathbb{R}^H$ and the current input $\boldsymbol{x}_t \in \mathbb{R}^N$. $\tilde{\boldsymbol{C}}_t \in \mathbb{R}^H$ is the candidate cell state. $\boldsymbol{C}_t \in \mathbb{R}^H$ and $\boldsymbol{h}_t \in \mathbb{R}^H$ are the cell state and the hidden state at time $t$ respectively. $\boldsymbol{W}_f, \boldsymbol{W}_i, \boldsymbol{W}_o, \boldsymbol{W}_C \in \mathbb{R}^{H \times (H+N)}$ and $\boldsymbol{b}_f, \boldsymbol{b}_i, \boldsymbol{b}_o, \boldsymbol{b}_C \in \mathbb{R}^H$ are parameters to learn. $\sigma$ and $tanh$ are sigmoid function and hyperbolic tangent function respectively. $\odot$ represents the element-wise multiplication.

### *b: STACKED LSTMs*

Although LSTM is good at temporal feature extraction, it is still difficult to capture dependencies among multiple variables in case of large complex data due to its simple structure. Inspired by the idea that increasing the depth of the neural network can effectively improve the network performance, we stack multiple LSTMs with the same hidden layer size vertically to construct the deep structure called stacked LSTMs (SLSTMs) as shown in Figure 7. In the vertical direction, SLSTMs is similar to a deep feedforward neural network with measurements of multiple variables at time $t$ as input, and hidden state output by each layer is the input of the next layer so that each layer can extract features at a different level to form a hierarchical representation. At the same time, in the horizontal direction, LSTM of each layer can extract temporal features from the information of the corresponding semantic level. This stacking structure ensures the excellent temporal feature extraction capacity of SLSTMs, and enhances its ability to capture complex relationship of multiple variables.

### *c: RESIDUAL CONNECTION*

Even though the performance of neural network can be improved to a certain extent by stacking multiple layers, it may also cause degradation problem when it goes deeper. Unlike gradient vanishing or exploding problem, it makes the network difficult to converge and even leads to the rise of learning error. For SLSTMs, this is more obvious due to its saturated non-linear activation functions. In our experience with time series forecasting tasks, SLSTMs works poor when the number of layers is more than three. Motivated by [18], we introduce residual connection into SRLSTMs to solve the above problem. As shown in Figure 8, the residual connection

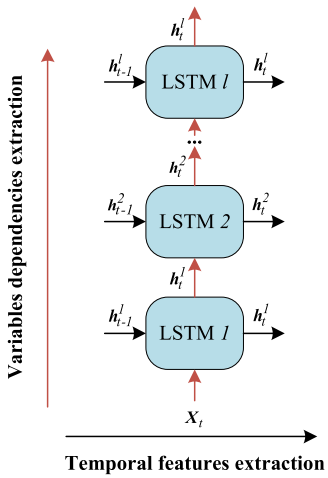



**FIGURE 7.** Stacked LSTMs.

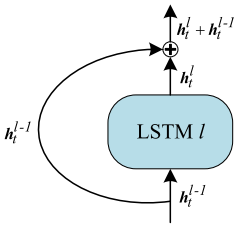



**FIGURE 8.** Residual connection.

is kind of short-cut connection. It shortens the connection between the input and output of the neural network so that when the back-propagation algorithm is applied to calculate the gradient of parameters, the gradient flow can pass to shallower layers, which guarantees the trainability of deep networks. The principle of SRLSTMs with $L$ layers is as follows:

$$\boldsymbol{C}_t^l, \boldsymbol{h}_t^l = LSTM_{\text{encoder}}^l(\boldsymbol{C}_{t-1}^l, \boldsymbol{h}_{t-1}^l, \boldsymbol{x}_t^l) \quad (1 \leqslant t \leqslant T) \tag{21}$$

where:

$$\boldsymbol{x}_t^l = \begin{cases} \boldsymbol{h}_t^{l-1} + \boldsymbol{h}_t^{l-2}, & (3 \leqslant l \leqslant L) \\ \boldsymbol{h}_t^{l-1}, & (l = 2) \\ \boldsymbol{X}_t, & (l = 1) \end{cases} \tag{22}$$

represents the input of the $l$th-layer LSTM at time $t$, and $LSTM_{\text{encoder}}^l$ is the $l$th-layer LSTM unit, which uses the cell state $\boldsymbol{C}_{t-1}^l$, hidden state $\boldsymbol{h}_{t-1}^l$ generated at time $t-1$ and $\boldsymbol{x}_t^l$ as the input, and outputs the cell state $\boldsymbol{C}_t^l$ and hidden state $\boldsymbol{h}_t^l$ at time $t$.

The cell state $\boldsymbol{C}_T^L$ and the hidden state $\boldsymbol{h}_T^L$ of the last layer LSTM at the last time step are the context vectors extracted by the encoder. They contain interdependencies and temporal information of the input multivariate time series and will be fed into the decoder to decode and predict.

### 2) DECODER

The decoder consists of a single-layer LSTM, a multi-level attention mechanism (MLAttn), and a linear layer. Firstly, LSTM decodes the context vectors extracted by the encoder and output the hidden state $\boldsymbol{d}_p \in \mathbb{R}^H$ step by step. Then, the MLAttn computes the weights of hidden states at different semantic levels obtained by the encoder across all time steps, and outputs the attention vector $\boldsymbol{v}_p \in \mathbb{R}^H$. Finally, the linear layer outputs the prediction value by applying a linear transformation to the concatenation of $\boldsymbol{d}_p$ and $\boldsymbol{v}_p$.

#### a: SINGLE-LAYER LSTM

The single-layer LSTM decodes the context vectors $\boldsymbol{C}_T^L$ and $\boldsymbol{h}_T^L$, and outputs the hidden state $\boldsymbol{d}_p \in \mathbb{R}^H$ and the cell state $\boldsymbol{D}_p \in \mathbb{R}^H$ as follows:

$$\boldsymbol{d}_p, \boldsymbol{D}_p = \begin{cases} LSTM_{\text{decoder}}(\boldsymbol{d}_{p-1}, \boldsymbol{D}_{p-1}, \hat{x}_{p-1}), & (2 \leqslant p \leqslant P) \\ LSTM_{\text{decoder}}(\boldsymbol{h}_T^L, \boldsymbol{C}_T^L, 0), & (p = 1) \end{cases} \tag{23}$$

where $LSTM_{\text{decoder}}$ is the LSTM unit of the decoder, $\boldsymbol{d}_p, \boldsymbol{D}_p \in \mathbb{R}^H$ are its hidden state and cell state at time $p$ respectively, and $\hat{x}_{p-1} \in \mathbb{R}$ represents the predicted value of the previous time, which is set to 0 when $p = 1$.

#### b: MULTI-LEVEL ATTENTION MECHANISM

To further improve the performance of the prediction model, as shown in Figure 9, we proposed the multi-level attention mechanism (MLAttn). It consists of two kinds of attention mechanisms: one is temporal attention mechanism and the other is semantic attention mechanism. Temporal attention mechanism can adaptively select the relevant hidden states of each layer of SRLSTMs for temporal attention vectors computing, which overcomes the information loss caused by the encoder compressing the extracted features to a fixed size vector. And semantic attention mechanism can determine the importance of information of every layer of SRLSTMs for prediction, which can make full use of the semantic features of each layer obtained by the encoder.

As for the temporal attention mechanism, the attention weights are computed as follows:

$$\alpha_{p,t}^l = \frac{e_{p,t}^l}{\sum_{j=1}^T e_{p,j}^l} \quad (1 \leqslant t \leqslant T) \tag{24}$$

and

$$e_{p,t}^l = \boldsymbol{m}^l \cdot tanh(\boldsymbol{U}^l \cdot \boldsymbol{d}_p + \boldsymbol{O}^l \cdot \boldsymbol{h}_t^l + \boldsymbol{b}^l) \tag{25}$$

where $\boldsymbol{m}^l \in \mathbb{R}^H$, $\boldsymbol{U}^l, \boldsymbol{O}^l \in \mathbb{R}^{H \times H}$ and $\boldsymbol{b}^l \in \mathbb{R}^H$ are parameters to learn.

Then, the temporal attention vector of each layer is calculated by weighted summation of hidden states across all time steps:

$$z_p^l = \sum_{i=1}^T \alpha_{p,i}^l \cdot \boldsymbol{h}_i^l \quad (1 \leqslant l \leqslant L) \tag{26}$$

As for the semantic attention mechanism, the attention weights of the temporal attention vectors obtained by Eq(26)

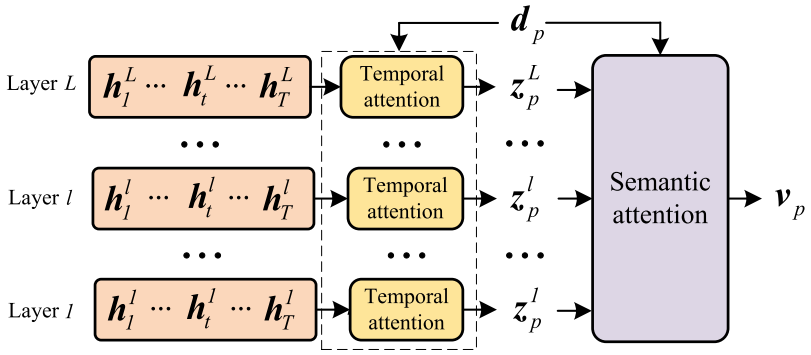



**FIGURE 9.** The proposed multi-level attention mechanism.

can be calculated as follows:

$$\beta_{p,l} = \frac{f_{p,l}}{\sum_{j=1}^{L} f_{p,j}} \quad (1 \leqslant j \leqslant L) \tag{27}$$

and

$$f_{p,l} = \boldsymbol{m}' \cdot \tanh(\boldsymbol{U}' \cdot \boldsymbol{d}_p + \boldsymbol{O}' \cdot \boldsymbol{z}_p^j + \boldsymbol{b}') \tag{28}$$

where $\boldsymbol{m}' \in \mathbb{R}^H$, $\boldsymbol{U}', \boldsymbol{O}' \in \mathbb{R}^{H \times H}$ and $\boldsymbol{b}' \in \mathbb{R}^H$ are parameters to learn.

Finally, all temporal attention vectors summed with weight to get the final attention vector $\boldsymbol{v}_p$:

$$\boldsymbol{v}_p = \sum_{l=1}^{L} \beta_{p,l} \cdot \boldsymbol{z}_p^l \tag{29}$$

#### c: LINEAR LAYER

The prediction value at time $p$ can be obtained by feeding the concatenation of $\boldsymbol{d}_p$ and $\boldsymbol{v}_p$ into linear layer:

$$\hat{x}_p = \boldsymbol{W} \cdot [\boldsymbol{d}_p, \boldsymbol{v}_p] + b \tag{30}$$

where $\hat{x}_p \in \mathbb{R}$ is the predicted value at time $p$, $\boldsymbol{W} \in \mathbb{R}^{2H \times 1}$ and $b \in \mathbb{R}$ are parameters to learn.

### 3) TRAINING PROCEDURE

We use the Stochastic Gradient Descent (SGD) algorithm and Adam optimization algorithm to train our prediction model. Since the proposed prediction model is smooth and differentiable, it can be learned by the standard back-propagation algorithm. We use the Mean Square Error (MSE) as the objective function. The overall loss function is as follows:

$$Loss = \frac{1}{n} \sum_{i=1}^{n} (x_i - \hat{x}_i)^2 \tag{31}$$

where $n$ is the number of prediction values. $x_i$ and $\hat{x}_i$ are the $i$-th actual measurement and the prediction value respectively.

**TABLE 2.** Dataset details.

| Datasets | Total size | No of variables | Sample rate |
|---|---|---|---|
| Beijing Air Quality | 8761 | 236 | 1 hour |
| Traffic | 17544 | 862 | 1 hour |
| Electricity | 26304 | 321 | 1 hour |
| Solar-Energy | 52560 | 137 | 10 minutes |

### D. ENSEMBLE

The final prediction of the target series $\hat{\boldsymbol{X}} \in \mathbb{R}^P$ can be obtained by accumulating all the prediction of sub-series:

$$\hat{\boldsymbol{X}} = \sum_{i=1}^{K} \hat{\boldsymbol{X}}'^i \tag{32}$$

where $\hat{\boldsymbol{X}}'^i \in \mathbb{R}^P (i = 1, 2, .., K)$ are the prediction of $K$ new sub-series obtained by ASC and $P$ is the prediction horizon.

## IV. EXPERIMENTS AND DISCUSSIONS

In this section, we compared our method with other 13 methods on four public datasets to show its superiority for multivariate time series forecasting. Besides, an ablation study is conducted to verify the effectiveness of each part of the proposed method. Finally, we explored the influence of different parameter settings, including the number of clusters $K$, the number of layers of SRLSTMs $L$ and the length of input data $T$.

### A. DATA DESCRIPTION

Four publicly available multivariate time series datasets with a large number of variables are used to validate the performance of prediction models. The details of each dataset are summarized in Table 2.

- Beijing Air Quality dataset contains the hourly air quality data, including PM2.5, PM10, AQI (Air Quality Index), $SO_2$, $NO_2$, $O_3$ and CO data of 34 regions in Beijing in 2018. In this study, we choose the PM2.5 data of the East Fourth Ring Road as the target series and the rest as correlation series. In general, the dataset size is 8761, and the total number of variables is 236.

- Traffic dataset contains two years (2015-2016) hourly data provided by the California Department of Transportation, which describes the road occupancy rate (between 0 and 1) on the San Francisco Bay area freeways. We choose the occupancy rate of one of the roads as the target series and the remaining variables as the correlation series. Overall, the size of the dataset is 17544 and the total number of variables is 862.
- Electricity dataset contains three years (2012-2014) hourly electricity consumption data recorded from 321 clients. We choose the the data from one of the clients as the target series and the rest as correlation series. Overall, the size of the dataset is 26304 and the total number of variables is 321.
- Solar-Energy dataset contains the solar power production recorded in the year of 2006, which is sampled every 10 minutes from 137 PV plants in Alabama State. We choose the the data from one of the PV plants as the target series and the rest as correlation series. Overall, the size of the dataset is 52560 and the total number of variables is 137.

### B. EXPERIMENT DESIGN

As shown in Figure 10, each dataset is divided into several continuous small datasets by a rolling window with a step size of $P$. Each small dataset contains training data and testing data, and the lengths are denoted as $D$ and $P$ respectively. Furthermore, the training data is divided into an 80% training set and a 20% validation set. In particular, when EEMD is used for data preprocessing, the partition method above is applied to each sub-series obtained by decomposition. Then, a rolling window with a step size of 1 is applied to divide the training set into multiple training samples. Each sample contains $T$ input data points and $P$ data points to predict. For the validation set, a rolling window with a step size of $P$ is used to divide the validation set into multiple validation samples to avoid the overlap of prediction data points. Finally, when the model finishes training on the training data, the test data is used to evaluate the prediction results of the model. In this study, we uniformly set $D$ to 1024, and $T$ and $P$ are varied according to the corresponding experimental settings.

We perform all experiments in this study by using Python programming language with version 3.7, on a PC with Intel I7 8700K CPU, GeForce RTX 2080 GPU and Microsoft Windows 10 operating system. 'NumPy' and 'PyEEMD' libraries are used for data processing. Besides, all the deep learning models were implemented by the 'PyTorch' machine learning library with version 1.2.

### C. DATA NORMALIZATION

To ensure the consistency of data distribution and prevent gradient explosions, we use min-max normalization to process the time series of each variable by scaling the data to a fixed range (0-1). The normalization formula is as follows:

$$\boldsymbol{X}_{\text{scaled}} = \frac{\boldsymbol{X} - \min(\boldsymbol{X})}{\max(\boldsymbol{X}) - \min(\boldsymbol{X})} \tag{33}$$

where $\boldsymbol{X}_{\text{scaled}}$ is the normalized time series, $\boldsymbol{X}$ is the original time series, $\max(\boldsymbol{X})$ and $\min(\boldsymbol{X})$ represent the maximum and minimum value of $\boldsymbol{X}$ respectively.

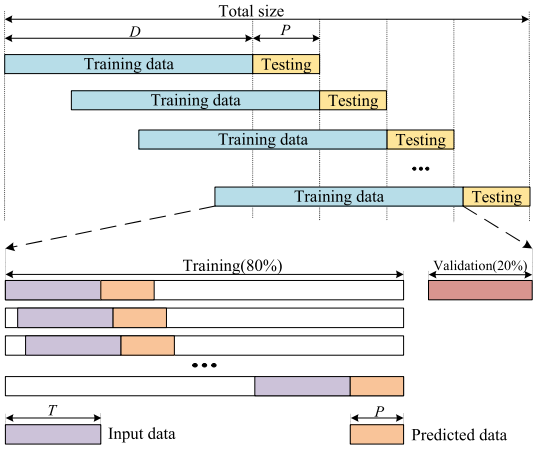



**FIGURE 10.** Overview of data division.

### D. EXPERIMENT METRICS

RMSE and MAE are two indicators commonly used as metrics to evaluate the performance of the prediction model. The definitions of these two metrics are as follows:

- Root Mean Squared Error(RMSE)

$$\text{RMSE}(\boldsymbol{X}, \hat{\boldsymbol{X}}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{x}_i)^2} \tag{34}$$

- Mean Absolute Error(MAE)

$$\text{MAE}(\boldsymbol{X}, \hat{\boldsymbol{X}}) = \frac{1}{N}\sum_{i=1}^{N}|x_i - \hat{x}_i| \tag{35}$$

where $x_i$ and $\hat{x}_i$ are the $i$-th true value and predicted value respectively, and $N$ is the total number of test data points.

### E. EXPERIMENT 1: MODEL COMPARISON

In order to show the superiority of the proposed method ASC-SRLSTMs-MLAttn, we compared ASC-SRLSTMs-MLAttn against other 13 methods by a short- and long-term prediction experiment. In this experiment, the prediction horizon $P$ is set from {3,6,9,12} and the corresponding input data length $T$ is set from {3,6,9,12}. In addition, only 720 data points were predicted taking into account the time consumption. The comparison models are as follows:

- ARIMAX [21]: Autoregressive Integrated Moving Average with Exogenous variables, which extends ARIMA for multi variables input.
- SVR [24]: Support Vector Regression.

**TABLE 3.** Time consumption of each phase of ASC-SRLSTMs-MLAttn.

| Phase | Paramters | $D$ | $K$ | $T$ | $P$ | $L$ | epochs |
|---|---|---|---|---|---|---|---|
| | | 1024 | 4 | 12 | 12 | 6 | 60 |
| EEMD decomposition | | 2.032±0.184s | | | | | |
| Adaptive Sub-series Clustering | | 12.532±0.926s | | | | | |
| Prediction model training | | 313.263±12.240s | | | | | |

**TABLE 4.** Prediction results of all models on Beijing Air Quality dataset.

| Models | Metrics | RMSE | | | | MAE | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Horizon(hour) | 3 | 6 | 9 | 12 | 3 | 6 | 9 | 12h |
| ARIMAX | | 23.05 | 29.44 | 38.81 | 48.12 | 15.27 | 19.57 | 26.92 | 33.16 |
| SVR | | 21.90 | 29.78 | 36.20 | 45.43 | 15.08 | 19.95 | 25.32 | 31.46 |
| MLP | | 20.94 | 28.80 | 35.95 | 43.10 | 14.06 | 19.28 | 25.13 | 30.88 |
| RNN | | 19.78 | 29.53 | 39.24 | 52.31 | 13.37 | 19.72 | 28.36 | 36.87 |
| GRU | | 19.49 | 27.53 | 35.46 | 42.01 | 12.87 | 17.71 | 25.02 | 30.58 |
| LSTM | | 19.60 | 27.48 | 35.33 | 42.03 | 12.98 | 17.64 | 24.92 | 30.63 |
| cLSTM | | 19.52 | 27.10 | 34.84 | 41.69 | 12.61 | 17.46 | 23.93 | 28.52 |
| CNN-LSTM | | 18.98 | 26.78 | 34.66 | 40.55 | 12.75 | 17.39 | 23.67 | 25.72 |
| EEMD-ELM | | 18.80 | 26.02 | 33.42 | 38.72 | 12.36 | 16.98 | 21.40 | 24.90 |
| EEMD-LSTM | | 18.44 | 25.37 | 31.65 | 37.03 | 11.96 | 17.02 | 20.51 | 24.17 |
| DA-RNN | | 18.07 | 25.26 | 32.44 | 37.82 | 11.70 | 16.06 | 20.87 | 24.68 |
| MV-LSTM | | 17.91 | 24.93 | 31.89 | 37.43 | 11.61 | 15.54 | 20.76 | 24.46 |
| AIS-RNN | | 17.76 | 24.74 | 32.04 | 37.50 | 11.52 | 15.27 | 20.82 | 24.59 |
| **ASC-SRLSTMs-MLAttn** | | **16.85** | **23.16** | **29.73** | **33.47** | **10.87** | **14.57** | **19.12** | **21.76** |

- MLP: Multi-Layer Perceptron.
- RNN: Simple Recurrent Neural Network without any gated structure.
- GRU [32]: Single-layer Gated Recurrent Unit.
- LSTM [31]: Single-layer Long-short Term Memory.
- cLSTM [39]: LSTM with sparse regularization on the weights.
- CNN-LSTM [15]: A model combining CNN for multivariate dependency capturing and LSTM for temporal features extraction.
- EEMD-ELM [46]: A model based on decomposition-ensemble prediction framework which uses EEMD for decomposition and ELM as the prediction model.
- EEMD-LSTM [47]: A model based on decomposition-ensemble prediction framework which uses EEMD for decomposition and LSTM as the prediction model.
- DA-RNN [41]: The Dual Stage Attention based Recurrent Neural Network. A prediction model with input and temporal dual attention mechanism based on the encoder-decoder architecture.
- MV-LSTM [42]: Multi-variable LSTM equipped with tensorized hidden states for individual variables hidden states learning.
- AIS-RNN [44]: A model combines RNNs with an adaptive input selection mechanism.

We use the grid search method to adjust hyparameters of each model. For ARIMAX $(p, d, q)$, $p$ and $q$ are determined by the minimum of the Bayes Information Criterion (BIC), and $d$ is determined by the Augmented Dickey-Fuller (ADF) test. SVR uses the linear function as kernel, the penalty parameter is set from $\{1^{-10}, 1^{-8}, \ldots, 1^{8}, 1^{10}\}$, and the tolerance for stopping criteria is $1e$-3. For the size of recurrent and dense layers in the baselines including MLP, RNN, GRU, LSTM, cLSTM, CNN-LSTM, EEMD-ELM, EEMD-LSTM, DA-RNN, MV-LSTM and AIS-RNN, grid search is conducted over {16,32,64,128,256}. The 1D-convolution layers of CNN-LSTM is set to 3 layers with the kernel size of 3 and max-pooling size of 2 in Beijing Air Quality dataset, which is set to 5 layers, 5 and 2 in Traffic dataset, 4 layers, 3 and 2 in Electricity dataset, 3 layers, 3 and 2 in Solar Energy dataset respectively. For AIS-RNN, we choose LSTM as the recurrent unit. For ASC-SRLSTMs-MLAttn, the number of clusters $K$ is set from {2, 3, 4, 5, 6, 7, 8, 9, 10} considering both efficiency and performance, the number of LSTM layers $L$ is set from {3, 4, 5, 6, 7, 8}, and the hidden layer size of LSTM is set from {16, 32, 64, 128, 256}.

Table 3 shows the actual time consumption of each phase of ASC-SRLSTMs-MLAttn with a given parameter settings.

Table 4-7 respectively summarize the prediction results of all the methods on four datasets, in which the best results in each horizon are highlighted in bold. It can be seen that with the increase of the prediction horizon, the prediction error of each method becomes larger, but the proposed ASC-SRLSTMs-MLAttn still outperforms the others in both metrics and has the minimum error growth. Particularly, when the prediction horizon is 12 hours, ASC-SRLSTMs-MLAttn is 10.6% and 11.0% superior than the most comparative method (MV-LSTM) on RMSE and MAE repectively in terms of Beijing Air Quality prediction, and shows 3.8% and 4.6% improvements in Traffic prediction, 2.9% and 3.4% improvements in Electricity prediction and 4.4% and 4.8% on RMSE and MAE respectively in Solar Energy prediction. Figure 11 visualize the prediction results of ASC-SRLSTMs-MLAttn on four dataset respectively. The ground truth is indicated by blue lines, while the prediction is indicated by yellow lines.

**TABLE 5.** Prediction results of all models on Traffic dataset.

| Models | Metrics | RMSE($\times 10^{-3}$) | | | | MAE($\times 10^{-3}$) | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Horizon(hour) | 3 | 6 | 9 | 12 | 3 | 6 | 9 | 12 |
| ARIMAX | | 48.54 | 54.69 | 57.01 | 60.97 | 37.11 | 43.79 | 43.09 | 45.09 |
| SVR | | 49.21 | 56.03 | 62.30 | 65.40 | 37.72 | 45.16 | 47.99 | 50.18 |
| MLP | | 47.85 | 53.97 | 56.12 | 58.39 | 36.52 | 41.40 | 43.05 | 44.72 |
| RNN | | 46.48 | 55.82 | 64.18 | 73.56 | 35.61 | 44.63 | 50.74 | 56.37 |
| GRU | | 46.32 | 48.21 | 51.24 | 52.71 | 34.63 | 37.47 | 38.42 | 40.42 |
| LSTM | | 46.13 | 48.68 | 51.09 | 52.62 | 34.54 | 37.54 | 38.16 | 40.34 |
| cLSTM | | 45.37 | 46.93 | 49.99 | 51.67 | 33.46 | 35.03 | 36.77 | 38.95 |
| CNN-LSTM | | 44.04 | 46.64 | 49.12 | 50.90 | 31.68 | 34.25 | 36.05 | 38.17 |
| EEMD-ELM | | 43.86 | 46.01 | 48.92 | 49.21 | 31.56 | 33.65 | 36.05 | 37.72 |
| EEMD-LSTM | | 42.05 | 44.57 | 46.66 | 47.35 | 28.90 | 31.52 | 32.20 | 34.22 |
| DA-RNN | | 41.87 | 43.66 | 44.52 | 45.97 | 28.74 | 31.27 | 31.46 | 33.51 |
| MV-LSTM | | 41.02 | 42.07 | 42.99 | 44.23 | 27.83 | 30.84 | 30.98 | 31.19 |
| AIS-RNN | | 40.51 | 41.83 | 43.12 | 44.71 | 27.31 | 30.67 | 31.09 | 31.35 |
| **ASC-SRLSTMs-MLAttn** | | **38.53** | **40.43** | **41.64** | **42.54** | **26.64** | **28.47** | **28.70** | **29.74** |

**TABLE 6.** Prediction results of all models on Electricity dataset.

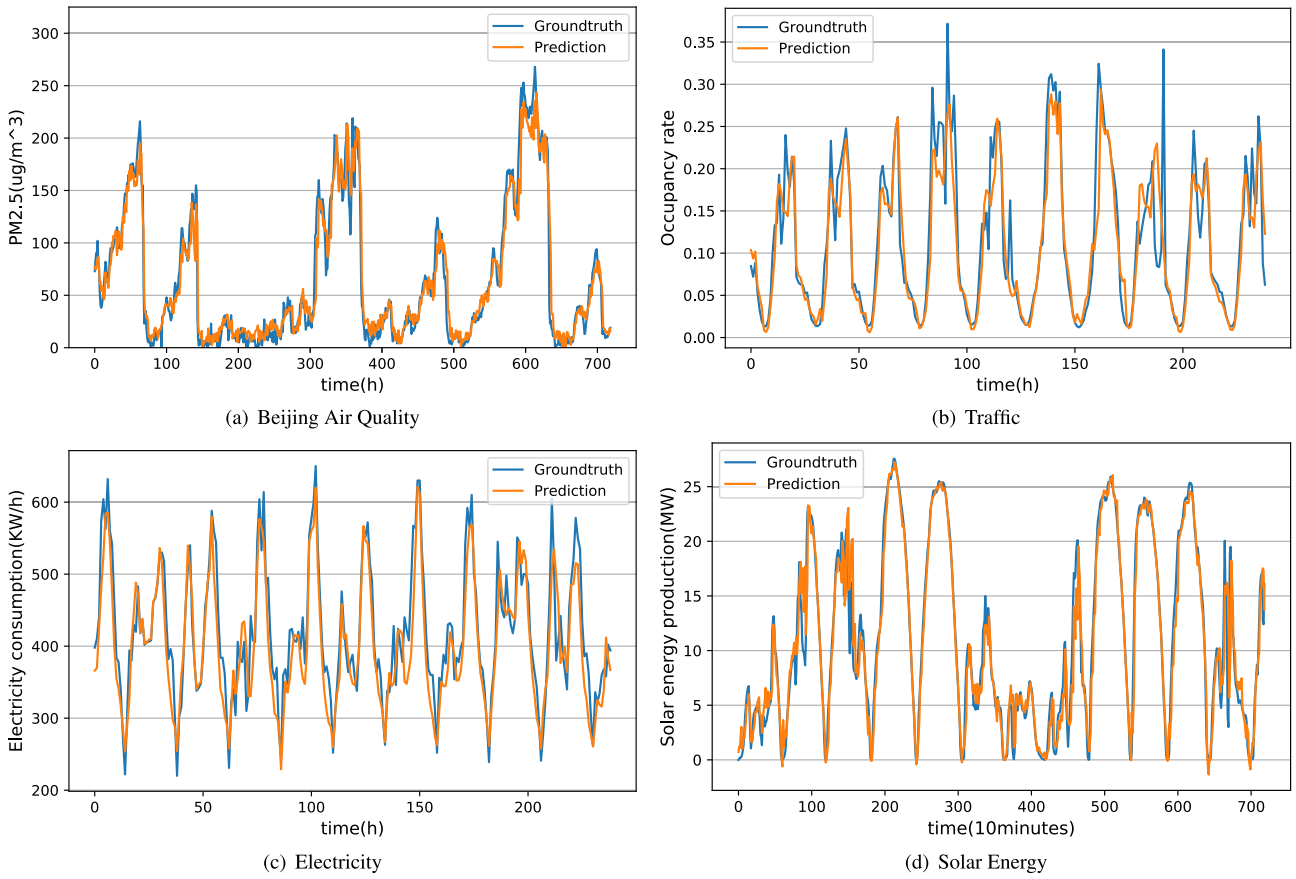
| Models | Metrics | RMSE | | | | MAE | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Horizon(hour) | 3 | 6 | 9 | 12 | 3 | 6 | 9 | 12 |
| ARIMAX | | 95.65 | 106.74 | 111.83 | 121.44 | 72.40 | 83.13 | 86.41 | 94.2 |
| SVR | | 93.27 | 102.92 | 108.35 | 116.63 | 72.36 | 79.54 | 84.06 | 90.38 |
| MLP | | 88.53 | 96.29 | 103.34 | 110.92 | 68.74 | 73.69 | 80.16 | 86.31 |
| RNN | | 87.02 | 100.64 | 118.32 | 132.43 | 66.12 | 76.55 | 90.87 | 103.28 |
| GRU | | 85.93 | 92.04 | 97.63 | 103.52 | 65.19 | 68.45 | 75.5 | 80.36 |
| LSTM | | 86.12 | 91.97 | 97.96 | 103.82 | 66.42 | 71.29 | 75.69 | 80.56 |
| cLSTM | | 84.53 | 90.04 | 96.23 | 100.21 | 65.72 | 69.86 | 74.82 | 77.74 |
| CNN-LSTM | | 83.81 | 88.52 | 94.22 | 98.86 | 65.26 | 68.77 | 73.15 | 76.20 |
| EEMD-ELM | | 82.13 | 86.69 | 92.41 | 95.12 | 63.64 | 67.31 | 71.76 | 73.53 |
| EEMD-LSTM | | 80.08 | 84.69 | 88.67 | 93.04 | 62.13 | 65.61 | 68.52 | 72.11 |
| DA-RNN | | 79.24 | 82.78 | 84.82 | 88.93 | 61.37 | 64.25 | 66.21 | 69.20 |
| MV-LSTM | | 78.16 | 81.22 | 83.11 | 85.83 | 60.66 | 63.09 | 64.39 | 66.93 |
| AIS-RNN | | 76.10 | 79.88 | 83.67 | 86.26 | 59.24 | 62.03 | 64.93 | 67.12 |
| **ASC-SRLSTMs-MLAttn** | | **73.42** | **77.61** | **80.23** | **83.34** | **56.91** | **59.52** | **61.97** | **64.66** |

**TABLE 7.** Prediction results of all models on Solar-Energy dataset.

| Models | Metrics | RMSE($\times 10^{-2}$) | | | | MAE($\times 10^{-2}$) | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Horizon(minutes) | 30 | 60 | 90 | 120 | 30 | 60 | 90 | 120 |
| ARIMAX | | 80.40 | 105.27 | 135.17 | 148.24 | 45.05 | 56.01 | 75.51 | 83.75 |
| SVR | | 76.51 | 92.57 | 116.82 | 127.63 | 43.97 | 53.20 | 63.84 | 70.91 |
| MLP | | 73.64 | 88.62 | 112.40 | 120.11 | 40.59 | 49.44 | 62.79 | 66 |
| RNN | | 71.90 | 95.63 | 127.61 | 159.14 | 39.43 | 54.33 | 70.89 | 86.96 |
| GRU | | 71.13 | 86.52 | 107.14 | 118.84 | 38.66 | 48.06 | 56.39 | 64.24 |
| LSTM | | 71.65 | 84.37 | 106.79 | 118.16 | 38.32 | 46.10 | 60.68 | 65.64 |
| cLSTM | | 69.26 | 81.98 | 104.06 | 115.67 | 37.53 | 44.93 | 58.66 | 65.22 |
| CNN-LSTM | | 68.52 | 80.54 | 102.35 | 113.66 | 36.22 | 44.25 | 55.63 | 64.58 |
| EEMD-ELM | | 65.61 | 76.38 | 96.54 | 105.89 | 33.73 | 43.40 | 52.18 | 58.83 |
| EEMD-LSTM | | 58.37 | 72.97 | 93.29 | 102.23 | 33.35 | 41.45 | 51.54 | 56.48 |
| DA-RNN | | 55.10 | 71.67 | 94.16 | 104.71 | 31.13 | 40.04 | 53.20 | 57.53 |
| MV-LSTM | | 54.21 | 68.74 | 92.61 | 101.15 | 29.62 | 38.19 | 52.62 | 55.27 |
| AIS-RNN | | 53.19 | 69.31 | 93.88 | 103.36 | 28.75 | 39.15 | 50.48 | 56.70 |
| **ASC-SRLSTMs-MLAttn** | | **49.63** | **64.38** | **87.85** | **96.69** | **27.32** | **34.80** | **47.23** | **52.61** |

We can see that ASC-SRLSTMs-MLAttn fits the ground truth well. The results strongly demonstrate the superiority of ASC-SRLSTMs-MLAttn for short- and long-term multivariate time series forecasting.

Meanwhile, we can observe that prediction methods based on deep learning are better than traditional statistics methods (ARIMAX) and methods based on machine learning (SVR). This is mainly because neural networks have strong feature extraction ability and can handle complex nonlinear data, while the traditional regression models always tend to be over-fitting when the data is large. RNNs are superior than MLP. However, the performance of simple RNN decreases when the prediction horizon is large, indicating the existence of gradient vanishing. It can also be easily seen that CNN-LSTM shows better performance than single-layer RNNs (GRU,LSTM,cLSTM). This is because the addition of CNN enhanced the feature extraction ability of LSTM, which reveals the importance of multivariable dependencies capturing for multivariate time series forecasting. Another notable observation is that both EEMD-ELM and EEMD-LSTM




(a) Beijing Air Quality (b) Traffic (c) Electricity (d) Solar Energy

**FIGURE 11. The prediction results of ASC-SRLSTMs-MLAttn on Beijing Air Quality(a), Traffic(b), Electricity(c) and Solar Energy(d) dataset with horizon = 12.**

achieve preferable results since the data preprocessing with time series decomposition can reduce the influences of noise components on prediction. Although DA-RNN, MV-LSTM and AIS-RNN work well in Traffic and Electricity prediction, it shows deficient long-term prediction capability on the Beijing Air Quality and Solar Energy dataset which contains more complex features. It may because of the lack of the noise processing. ASC-SRLSTMs-MLAttn has the best prediction performance because of the combination of the noise separation process and the better feature extraction ability.

### *F. EXPERIMENT 2: ABLATION STUDY*

To demonstrate the effectiveness and explore the influences of components of the proposed method on prediction performance, we conducted an ablation study on Beijing Air Quality dataset. By removing or replacing one component in ASC-SRLSTMs-MLAttn at a time, we can get 5 variants:

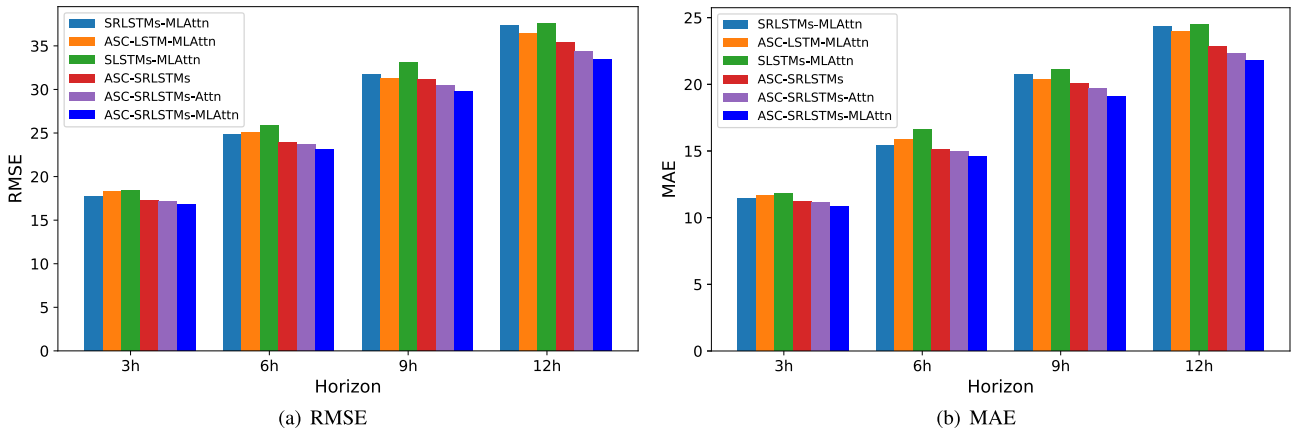
- SRLSTMs-MLAttn: ASC-SRLSTMs-MLAttn without time series decomposition preprocessing.
- ASC-LSTM-MLAttn: ASC-SRLSTMs-MLAttn with single-layer LSTM as the encoder.
- ASC-SLSTMs-MLAttn: ASC-SRLSTMs-MLAttn without residual connection.
- ASC-SRLSTMs: ASC-SRLSTMs-MLAttn without attention mechanism.
- ASC-SRLSTMs-Attn: ASC-SRLSTMs-MLAttn with normal attention mechanism which calculates the attention vectors only according to the hidden states of the last layer of SRLSTMs.

We fix $K$ to 4 and $L$ to 6, and change $P$ and $T$ from {3, 6, 9, 12} in turn. The experiment results are shown in Table 8 and Figure 12, from which we can observe that:

1) ASC-SRLSTMs-MLAttn achieved the best results in all horizons.
2) The removal of ASC greatly reduces the prediction accuracy of the model, showing the importance of data preprocessing.
3) Both ASC-SRLSTMs-Attn and ASC-SRLSTMs-MLAttn outperform ASC-SRLSTMs, and it is more obvious as the horizon increases, proving that attention mechanism can effectively improve the prediction performance.
4) ASC-SRLSTMs-MLAttn obtaines better results than ASC-SRLSTMs-Attn, which demonstrates the effectiveness of the proposed multi-level attention mechanism.
5) ASC-SLSTMs-MLAttn does not perform better than the ASC-LSTM-MLAttn, which indicates that stacking LSTM simply may lead to performance degradation.

**TABLE 8.** Results of Ablation study.

| Dataset | | Beijing Air Quality | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Models | Metrics | RMSE | | | | MAE | | | |
| | Horizon($P$) | 3h | 6h | 9h | 12h | 3h | 6h | 9h | 12h |
| SRLSTMs-MLAttn | | 17.72 | 24.84 | 31.78 | 37.36 | 11.46 | 15.39 | 20.73 | 24.31 |
| ASC-LSTM-MLAttn | | 18.26 | 25.13 | 31.23 | 36.49 | 11.67 | 15.88 | 20.37 | 23.96 |
| ASC-SLSTMs-MLAttn | | 18.38 | 25.86 | 33.15 | 37.62 | 11.85 | 16.64 | 21.15 | 24.52 |
| ASC-SRLSTMs | | 17.31 | 23.95 | 31.13 | 35.43 | 11.23 | 15.14 | 20.09 | 22.83 |
| ASC-SRLSTMs-Attn | | 17.20 | 23.67 | 30.46 | 34.40 | 11.12 | 14.99 | 19.68 | 22.35 |
| **ASC-SRLSTMs-MLAttn** | | **16.85** | **23.16** | **29.73** | **33.47** | **10.87** | **14.57** | **19.12** | **21.76** |




(a) RMSE (b) MAE

**FIGURE 12.** Results of ablation study, RMSE(a) and MAE(b).

6) Compared with the ASC-SLSTMs-MLAttn, the ASC-SRLSTMs-MLAttn shows significant improvements, which reveals the important role of residual connection in SRLSTMs.

The overall conclusion is that each component of the ASC-SRLSTMs-MLAttn model effectively improves the prediction performance. This is because ASC decomposes the difficult single time series prediction task into several simple sub-tasks and separates the noise from useful components, which simplifies the modeling difficulty and improves the temporal feature extraction ability of the model. In addition, the introduction of residual connection ensures the performance of the SLSTMs with deep layers and avoids degradation problems, which provides better optimization capability for the model and enhances the capacity of the model to capture dependencies among multiple variables. Moreover, the MLAttn alleviates the loss of information and fully considers the semantic information of different levels of SLSTMs, which makes the model still perform well when the prediction horizon becomes larger.

### G. EXPERIMENT 3: PARAMETER SETTING

In this section, three experiments are conducted to explore the impacts of difference values of the number of sub-series clusters $K$, the number of layers of the SRLSTMs $L$ and the length of input data $T$ on the prediction performance respectively. In particular, we choose the Beijing Air Quality as the dataset and fix the prediction horizon $P$ to 12.

#### 1) SETTING OF K

$K$ is the number of sub-series clusters after ASC algorithm. It affects the prediction performance of ASC-SRLSTMs-MLAttn, and the total time consumption. In this experiment, we set $K$ from {1, 2, 4, 6, 8, 10} since the total number of sub-series is 10 when $D = 1024$. And $T$ is set to 12. In particular, when $K = 1$, it means that the target time series is not been decomposed. Besides, ASC is not applied when $K = 10$. The experiment results are summarized in Table 9.

From Table 9, we can see that the prediction error of the model decreases with the increase of $K$. This is because the larger the $K$ is, the more number of the sub-series is and the separation of the non-stationary factors is more sufficient, making the impact of the noise components on the overall prediction lower. Besides, it is obvious that the time consumption of prediction is proportional to $K$ and increases linearly. However, it can also be observed that the improvement of prediction accuracy becomes smaller as $K$ increases, which is 6.24%, 4.45%, 0.39%, 0.18% and 0.06% respectively when $K$ is equal to 2, 4, 6, 8 and 10. Especially, when $K$ is larger than 4, the improvement is not significant. In conclusion, for the case that both prediction accuracy and time efficiency need to be considered, the appropriate value of $K$ is required to achieve the best balance, which is 4 in this study.

#### 2) SETTING OF L

$L$ is the number of layers of SRLSTMs, which affects the training effect and the overall prediction result of the model. We set $L$ from {1, 2, 4, 6, 8} and $T$ to 12 in this experiment.

**TABLE 9. Results of ASC-SRLSTMs-MLAttn with different $K$ values on Beijing Air Quality dataset.**

| Dataset | | Beijing Air Quality | | |
|---|---|---|---|---|
| Horizon($P$) | | 12h | | |
| $K$ | Metrics | RMSE | MAE | Consumed time(×%) |
| 1 | | 37.36 | 24.31 | 100 |
| 2 | | 35.03 | 22.44 | 240 |
| 4 | | 33.47 | 21.76 | 415 |
| 6 | | 33.34 | 21.64 | 640 |
| 8 | | 33.28 | 21.57 | 810 |
| 10 | | 33.26 | 21.52 | 1030 |

**TABLE 10. Prediction results of ASC-SRLSTMs-MLAttn with different $L$ values on Beijing Air Quality dataset.**

| Dataset | | Beijing Air Quality | |
|---|---|---|---|
| Horizon($P$) | | 12h | |
| $L$ | Metrics | RMSE | MAE |
| 1 | | 36.49 | 23.96 |
| 2 | | 35.18 | 22.84 |
| 4 | | 33.69 | 22.01 |
| **6** | | **33.47** | **21.76** |
| 8 | | 33.99 | 22.13 |

**TABLE 11. Prediction results of ASC-SRLSTMs-MLAttn with different $T$ values on Beijing Air Quality dataset.**

| Dataset | | Beijing Air Quality | |
|---|---|---|---|
| Horizon($P$) | | 6h | |
| $T$ | Metrics | RMSE | MAE |
| 3 | | 25.64 | 16.43 |
| **6** | | **23.16** | **14.57** |
| 9 | | 24.31 | 14.93 |
| 12 | | 25.28 | 15.73 |
| 24 | | 28.16 | 18.54 |

In particular, when $L = 1$, the encoder of the prediction model is a single-layer LSTM. The experiment results are shown in Table 10, where the best result is shown in bold.

From Table 10, we can observe that with the increase of $L$, the prediction accuracy of the model improves, since the stacking structure deepens the network and can effectively improve the feature extraction ability of the model. Compared with single-layer LSTM, the prediction performance of SRLSTMs with 2, 4, 6 and 8 layers improve by 3.59%, 7.67%, 8.28% and 6.85% respectively, demonstrating the effectiveness of SRLSTMs. However, it can also be seen that when $L$ reaches a certain limit, which is 6 in this study, stacking layers continuously will cause performance degradation. This may be because the parameter space of the model is too large, leading to over-fitting.

### 3) SETTING OF T

The length of input data $T$ also affects the performance of ASC-SRLSTMs-MLAttn. In this experiment, we set $T$ from $\{3, 6, 9, 12, 24\}$ and prediction horizon $P$ to 6. Table 11 shows the prediction results of ASC-SRLSTMS-MLAttn.

When $T = 6$, ASC-SRLSTMs-MLAttn has the best performance. In contrast, the prediction performance of ASC-SRLSTMs-MLAttn when $T = 3$ is poor, which indicates that adequate historical data is essential to make an accurate prediction. In addition, when $T$ is larger than 6, the prediction error is also increasing. This may be because the amount of data is too excessive for ASC-SRLSTMs-MLAttn to extract useful features to make prediction. In conclusion, a appropriate value of $T$ is necessary for ASC-SRLSTMs-MLAttn to achieve the best prediction accuracy.

## V. CONCLUSION AND FUTURE WORK

In this study, we proposed a novel method for multivariate time series forecasting called ASC-SRLSTMs-MLAttn, which is based on decomposition-ensemble prediction framework. The Adaptive Sub-series Clustering algorithm (ASC) is used to cluster the sub-series obtained via time series decomposition to ensure the prediction accuracy and reduce the total time consumption. To achieve better prediction performance, stacked residual LSTM (SRLSTMs) is used to capture the time series features and dependencies among variables. At the same time, MLAttn is also used to further improve the model performance. Sufficient long- and short-term prediction experiments on four publicly available dataset have demonstrated the superiority and the effectiveness of the proposed method.

At present, our approach is only applied to the prediction of a single variable, that is, only one target series is predicted. However, as multiple variables of time series are related, it is possible to predict multiple target series at the same time. Besides, since the decomposition and prediction stage is separate, the result of parameter selection may not be optimal. In the future work, we will further study the feasible methods of multi-target prediction and find the best way to implement our method in end-to-end framework.

## REFERENCES


[1] 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, 2018.

[2] Y. Hou, P. Edara, and C. Sun, "Traffic flow forecasting for urban work zones," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 4, pp. 1761–1770, Aug. 2014.

[3] A. Chelani and S. Devotta, "Air quality forecasting using a hybrid autoregressive and nonlinear model," *Atmos. Environ.*, vol. 40, no. 10, pp. 1774–1780, Mar. 2006.

[4] H. Hassani, A. Dionisio, and M. Ghodsi, "The effect of noise reduction in measuring the linear and nonlinear dependency of financial markets," *Nonlinear Anal., Real World Appl.*, vol. 11, no. 1, pp. 492–502, Feb. 2010.

[5] M. Xu, M. Han, and H. Lin, "Wavelet-denoising multiple echo state networks for multivariate time series prediction," *Inf. Sci.*, vol. 465, pp. 439–458, Oct. 2018.

[6] F. Liu, M. Cai, L. Wang, and Y. Lu, "An ensemble model based on adaptive noise reducer and over-fitting prevention LSTM for multivariate time series forecasting," *IEEE Access*, vol. 7, pp. 26102–26115, 2019.

[7] J. Bedi and D. Toshniwal, "Empirical mode decomposition based deep learning for electricity demand forecasting," *IEEE Access*, vol. 6, pp. 49144–49156, 2018.

[8] J. Cao, Z. Li, and J. Li, "Financial time series forecasting model based on CEEMDAN and LSTM," *Phys. A, Stat. Mech. Appl.*, vol. 519, pp. 127–139, Apr. 2018.

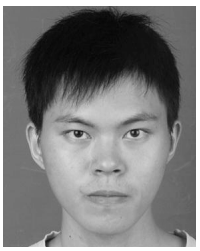
[9] Z. Wu and N. E. Huang, "Ensemble empirical mode decomposition: A noise-assisted data analysis method," *Adv. Adapt. Data Anal.*, vol. 1, no. 1, pp. 1–41, Jan. 2009.

[10] L. Tang, S. Wang, K. He, and S. Wang, "A novel mode-characteristic-based decomposition ensemble model for nuclear energy consumption forecasting," *Ann. Oper. Res.*, vol. 234, no. 1, pp. 111–132, Nov. 2015.

[11] J. S. Richman and J. R. Moorman, "Physiological time-series analysis using approximate entropy and sample entropy," *Amer. J. Physiology-Heart Circulatory Physiol.*, vol. 278, no. 6, pp. H2039–H2049, Jun. 2000.

[12] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.
[13] I. Sutskever, O. Vinyals, and Q. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3104–3112.
[14] Y. Wu and H. Tan, "Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework," 2016, *arXiv:1612.01022*. [Online]. Available: http://arxiv.org/abs/1612.01022
[15] T.-Y. Kim and S.-B. Cho, "Predicting residential energy consumption using CNN-LSTM neural networks," *Energy*, vol. 182, pp. 72–81, Sep. 2019.
[16] W. Bao, J. Yue, and Y. Rao, "A deep learning framework for financial time series using stacked autoencoders and long-short term memory," *PLoS ONE*, vol. 12, no. 7, 2017, Art. no. e0180944.
[17] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.
[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015, *arXiv:1512.03385*. [Online]. Available: http://arxiv.org/abs/1512.03385
[19] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, *arXiv:1409.0473*. [Online]. Available: http://arxiv.org/abs/1409.0473
[20] J. Contreras, R. Espinola, F. J. Nogales, and A. J. Conejo, "ARIMA models to predict next-day electricity prices," *IEEE Trans. Power Syst.*, vol. 18, no. 3, pp. 1014–1020, Jul. 2003.
[21] B. M. Williams, "Multivariate vehicular traffic flow prediction: Evaluation of ARIMAX modeling," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1776, no. 1, pp. 194–200, Jan. 2001.
[22] S. Liu and P. C. M. Molenaar, "IVAR: A program for imputing missing data in multivariate time series using vector autoregressive models," *Behav. Res. Methods*, vol. 46, no. 4, pp. 1138–1148, Dec. 2014.
[23] R. C. Garcia, J. Contreras, M. vanAkkeren, and J. B. C. Garcia, "A GARCH forecasting model to predict day-ahead electricity prices," *IEEE Trans. Power Syst.*, vol. 20, no. 2, pp. 867–874, May 2005.
[24] N. Sapankevych and R. Sankar, "Time series prediction using support vector machines: A survey," *IEEE Comput. Intell. Mag.*, vol. 4, no. 2, pp. 24–38, May 2009.
[25] P. Louka, G. Galanis, N. Siebert, G. Kariniotakis, P. Katsafados, I. Pytharoulis, and G. Kallos, "Improvements in wind speed forecasts for wind power prediction purposes using Kalman filtering," *J. Wind Eng. Ind. Aerodynamics*, vol. 96, no. 12, pp. 2348–2362, Dec. 2008.
[26] H. Tyralis and G. Papacharalampous, "Variable selection in time series forecasting using random forests," *Algorithms*, vol. 10, no. 4, p. 114, 2017.
[27] F. Douak, F. Melgani, and N. Benoudjit, "Kernel ridge regression with active learning for wind speed prediction," *Appl. Energy*, vol. 103, pp. 328–340, Mar. 2013.
[28] D. Yang, Z. Ye, L. H. I. Lim, and Z. Dong, "Very short term irradiance forecasting using the lasso," *Sol. Energy*, vol. 114, pp. 314–326, Apr. 2015.
[29] C. Paoli, C. Voyant, M. Muselli, and M.-L. Nivet, "Forecasting of preprocessed daily solar radiation time series using neural networks," *Sol. Energy*, vol. 84, no. 12, pp. 2146–2160, Dec. 2010.
[30] X. Chen, Z. Y. Dong, K. Meng, Y. Xu, K. P. Wong, and H. W. Ngan, "Electricity price forecasting with extreme learning machine and bootstrapping," *IEEE Trans. Power Syst.*, vol. 27, no. 4, pp. 2055–2062, Nov. 2012.
[31] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[32] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," 2014, *arXiv:1406.1078*. [Online]. Available: http://arxiv.org/abs/1406.1078
[33] P. Coulibaly, F. Anctil, and B. Bobée, "Multivariate reservoir inflow forecasting using temporal neural networks," *J. Hydrol. Eng.*, vol. 6, no. 5, pp. 367–376, Oct. 2001.
[34] W. Kong, Z. Y. Dong, Y. Jia, D. J. Hill, Y. Xu, and Y. Zhang, "Short-term residential load forecasting based on LSTM recurrent neural network," *IEEE Trans. Smart Grid*, vol. 10, no. 1, pp. 841–851, Jan. 2017.
[35] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, "Recurrent neural networks for multivariate time series with missing values," *Sci. Rep.*, vol. 8, no. 1, Dec. 2018, Art. no. 6085.
[36] B. Pratim Mukhoty, V. Maurya, and S. Kumar Shukla, "Sequence to sequence deep learning models for solar irradiation forecasting," 2019, *arXiv:1904.13081*. [Online]. Available: http://arxiv.org/abs/1904.13081
[37] Y. Liang, S. Ke, J. Zhang, X. Yi, and Y. Zheng, "GeoMAN: Multi-level attention networks for geo-sensory time series prediction," in *Proc. IJCAI*, Jul. 2018, pp. 3428–3434.
[38] A. Sagheer and M. Kotb, "Time series forecasting of petroleum production using deep LSTM recurrent networks," *Neurocomputing*, vol. 323, pp. 203–213, Jan. 2019.
[39] A. Tank, I. Covert, N. Foti, A. Shojaie, and E. Fox, "Neural Granger causality for nonlinear time series," 2018, *arXiv:1802.05842*. [Online]. Available: http://arxiv.org/abs/1802.05842
[40] G. Lai, W.-C. Chang, Y. Yang, and H. Liu, "Modeling long-and short-term temporal patterns with deep neural networks," in *Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2018, pp. 95–104.
[41] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, and G. Cottrell, "A dual-stage attention-based recurrent neural network for time series prediction," 2017, *arXiv:1704.02971*. [Online]. Available: http://arxiv.org/abs/1704.02971
[42] T. Guo and T. Lin, "Multi-variable LSTM neural network for autoregressive exogenous model," 2018, *arXiv:1806.06384*. [Online]. Available: http://arxiv.org/abs/1806.06384
[43] Y.-Y. Chang, F.-Y. Sun, Y.-H. Wu, and S.-D. Lin, "A memory-network based solution for multivariate time-series forecasting," 2018, *arXiv:1809.02105*. [Online]. Available: http://arxiv.org/abs/1809.02105
[44] L. Munkhdalai, T. Munkhdalai, K. H. Park, T. Amarbayasgalan, E. Erdenebaatar, H. W. Park, and K. H. Ryu, "An end-to-end adaptive input selection with dynamic weights for forecasting multivariate time series," *IEEE Access*, vol. 7, pp. 99099–99114, 2019.
[45] A. J. Conejo, M. A. Plazas, R. Espinola, and A. B. Molina, "Day-ahead electricity price forecasting using the wavelet transform and ARIMA models," *IEEE Trans. Power Syst.*, vol. 20, no. 2, pp. 1035–1042, May 2005.
[46] R. Prasad, R. C. Deo, Y. Li, and T. Maraseni, "Soil moisture forecasting by a hybrid machine learning technique: ELM integrated with ensemble empirical mode decomposition," *Geoderma*, vol. 330, pp. 136–161, Nov. 2018.
[47] X. Zhang, Q. Zhang, G. Zhang, Z. Nie, Z. Gui, and H. Que, "A novel hybrid data-driven model for daily Land Surface Temperature forecasting using long short-term memory neural network based on ensemble empirical mode decomposition," *Int. J. Environ. Res. Public Health*, vol. 15, no. 5, p. 1032, 2018.
[48] Q. Zhu, F. Zhang, S. Liu, Y. Wu, and L. Wang, "A hybrid VMD–BiGRU model for rubber futures time series forecasting," *Appl. Soft Comput.*, vol. 84, Nov. 2019, Art. no. 105739.

**FAGUI LIU** received the M.S. degree from Beihang University, in 1991, and the Ph.D. degree from the South China University of Technology, Guangzhou, China, in 2006. She is currently a Professor with the School of Computer Science and Engineering, South China University of Technology. Her research interests include service computing, the Internet of Things, cloud computing, and big data.



**YUNSHENG LU** received the B.S. degree from the South China University of Technology, Guangzhou, China, in 2018, where he is currently pursuing the M.S. degree with the School of Computer Science and Engineering. His research interests include machine learning, time series analysis, and the Internet of Things.

**MUQING CAI** received the B.S. degree from the South China University of Technology, Guangzhou, China, in 2017, where he is currently pursuing the M.S. degree with the School of Computer Science and Engineering. His research interests include machine learning, time series analysis, and the Internet of Things.