Information Sciences 607 (2022) 477–492

Contents lists available at ScienceDirect

Information Sciences

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

Predicting high-dimensional time series data with spatial, temporal and global information

Jining Wang [a], Chuan Chen [a,*], Zibin Zheng [a], Luonan Chen [b,c,d], Yuren Zhou [a]

[a] School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou 510275, China
[b] Key Laboratory of Systems Biology, Shanghai Institute of Biochemistry and CellBiology, Center for Excellence in Molecular Cell Science, Chinese Academy of Sciences,Shanghai 200031, China
[c] Key Laboratory of Systems Health Science of Zhejiang Province, HangzhouInstitute for Advanced Study, University of Chinese Academy of Sciences, ChineseAcademy of Sciences, Hangzhou 310024, China
[d] Peng Cheng Laboratory, Shenzhen, Guangdong 518000, China

ARTICLE INFO

Article history:
Received 9 February 2021
Received in revised form 20 January 2022
Accepted 5 June 2022
Available online 8 June 2022

Keywords:
Time series forecasting
Dynamics framework
Attractor
STSM

ABSTRACT

In the field of time series forecasting, deep learning and dynamics-based methods are two main research directions. The former focuses on the temporal information of the data while the latter emphasizes on the spatial information of the data, and rare methods combine the two information properly. In order to make better use of the information in the data, we propose the STSM (spatiotemporal skip-connection model) based on the dynamics framework, which contains a temporal module composed of CNN and a spatial module composed of fully connected layers, as well as a skip connection to the original input to fuse temporal, spatial, and global information in the data. To predict the future value of the target variable, STSM is required to learn a mapping from original attractors to delay attractors in an end-to-end framework. The results of ablation and contrast experiments on one simulated dataset and seven real-world datasets show that STSM not only performs better than a separate temporal or spatial module, but also predicts more accurately than other traditional methods. Besides, we verify the robustness of the model in different scenarios through several experiments.

1. Introduction

In today's highly information-based era, a huge amount of time series data has been generated in various disciplines of scientific research [1]. For example, in the field of biology, driven by gene sequencing technology, high-throughput biological data has grown rapidly [2], such situation also appears in atmospheric science[3] and intelligent transportation[4]. In this case, many researchers are facing huge analysis needs for high-dimensional time series data.

Historically, many theories have been proposed to solve the problem of time series prediction. Early methods are often based on parametric models, like autoregressive models and moving average models as well as their variants [5–7]. The disadvantage of such models is that they cannot model the relationship between high-dimensional data and have strong constraints on the stability of time series data. With the development of machine learning and deep learning, related methods have been widly applied in the field of time series prediction, such as support vector regression [8–10], gaussian process [11],

* Corresponding author.
E-mail address: chenchuan@mail.sysu.edu.cn (C. Chen).

https://doi.org/10.1016/j.ins.2022.06.021
0020-0255/

feedforward neural network [12–14], convolutional neural network [15], recursive neural network [16–18] and so on. Compared with previous methods, they have fewer constraints on the data and enjoy a higher degree of approximation. However, the interpretability is not enough due to the inherent property of black box models, and the information from time and space cannot be combined well. Some methods under the dynamics framework [19–21] have better interpretability. But the shortcomings are obvious, too much attention is paid to spatial information while ignoring temporal information, and it also has high requirements for the chaos of the system.

In order to properly combine the spatial and temporal information on high-dimensional data to make better predictions, we propose a new model called Spatiotemporal skip-connection model (STSM for short) in this paper, which can simultaneously make temporal transformation, make spatial transformation and keep global information with temporal module, spatial module and skip connection respectively, and combine the spatial, temporal, and global information to make predictions. Before describing STSM, we need to introduce the concept of attractor: under the dynamics framework, high-dimensional variables and their changing process over time constitute a dynamic system, and all variables in the system constitute a phase space, the development of phase space tends to a relatively stable state called attractor. And in this study, we consider the local sampling of high-dimensional data as original attractors, the continuation of the target variable in time as delay attractors. Based on the Takens theorm [22], STSM tries to reconstruct delay attractors from original attractors by solving a mapping between them, so that the dynamic information in the high-dimensional data can be converted into the temporal information of the target variable. Specifically, we repeatedly sample the training data to get several pairs of original and delay attractors, take them as input and output of STSM respectively to learn the mapping. Although it seems that STSM can only make short-term predictions, we can get long-term prediction through an iterative process, in which the output of the previous prediction is used as the input of the next step. The results of contrast, ablation and robustness experiments on synthetic and real-world datasets show that STSM model can make more accurate predictions with strong robustness than traditional methods. In all, we make following contributions in this paper.

- Rather than utilizing the spatial information from high-dimensional data to make predictions directly like traditional dynamic methods, we want to combine the spatial information and temporal information properly to get better results. To prove the feasibility of our idea, we propose a model called STSM based on Takens theorem to predict target variable from high-dimensional data, which contains a temporal module, a spatial module and a skip connection to the original input to combine spatial, temporal and global information.
- We conduct several experiments on different datasets to test the ability of our model. It can be seen from the figures and indicators of contrast experiments that STSM surpasses other traditional time series prediction models in terms of predictive effects. And ablation experiments show that each module of our model is indispensable. Besides, the robustness experiments indicate the potential of STSM in combating noise, adapting to different initialization parameters as well as hyperparameters, dealing with time variability and making long-term prediction

The rest of this paper is organized as follows. In Section 2, we summarize related work from the perspective of predicting low-dimensional and high-dimensional data, the advantages and disadvantages of different models are compared in a fair way at the same time. Section 3 gives a definition of the prediction problem and explains how to apply Takens theorem to solve it under the dynamics framework, then our STSM model is introduced from the whole to the part. In Section 4, we do several experiments on one simulated Lorentz dataset and seven real-world datasets, the results of these contrast experiments, ablation experiments and robustness experiments are carefully analyzed. Finally, Section 5 draws the conclusion, analyses the advantages and disadvantages of STSM, and points out the direction for improvements in the future.

2. Related work

This chapter aims to summarize and compare the traditional models and algorithms for time series forecasting, show the advantages and disadvantages of different models, so as to highlight the main improvements of this work. The following contents will be divided into two parts to introduce models for predicting low-dimensional (univariate) and high-dimensional (multivariate) time series respectively.

2.1. Low-dimensional data predictors

This field mainly relies on methods based on parametric models, the solving process of such models can be divided into three steps: determine which model should be used according to the characteristics of the data; calculate the model parameters; use the model to make predictions and evaluate effects [23]. The definitions of common models are as follows.

Autoregressive model (AR) is a model with p-step uncorrelated property. Its basic idea is to model the influence of historical data on current data [5]. Suppose that X_t represents the time series value corresponding to time t, c represents the constant term, p represents the magnitude of the order for AR, and ε_t represents noise at time t, $a_1, a_2, \ldots, a_p$ represent the corresponding weight coefficients of AR. Then AR(p) can be expressed by the following formula:

$$X_t = c + \sum_{i=1}^{p} a_i X_{t-i} + \varepsilon_t. \tag{1}$$

Moving average model (MA) is a stationary sequence model with q-step uncorrelated properties. Its basic idea is to focus on the accumulation of error terms, and use the linear combination of the forecast errors at several previous moments to predict the current value, enabling it to effectively eliminate the random error in the prediction [6]. Similar to the assumption above, suppose that μ represents the constant term, q represents the magnitude of the order for MA, $b_1, b_2, \ldots, b_q$ represent the corresponding weight coefficients of MA. Then MA(q) can be expressed by the following formula, note that when AR is high-order (p is large), it can be approximated by a low-order MA (q is small) [7]:

$$X_t = \mu + \sum_{i=1}^{q} b_i \varepsilon_{t-i} + \varepsilon_t. \tag{2}$$

Autoregressive moving average model (ARMA) is a mixture of AR model and MA model [7], suppose that p, q represent the magnitude of the order for AR and MA respectively, then ARMA(p, q) can be expressed by the following formula:

$$X_t = c + \sum_{i=1}^{p} a_i X_{t-i} + \mu + \sum_{i=1}^{q} b_i \varepsilon_{t-i} + \varepsilon_t. \tag{3}$$

The aforementioned models have strict requirements on the stability of the data, namely that the mean value of the time series data is not allowed to change significantly over time. If the data fluctuates greatly, this requirement cannot be met. In this case, the stable data meeting the requirements of these models can be obtained by the difference method. On the basis of ARMA, autoregressive integrated moving average model (ARIMA) makes d-order difference on data to expand its application scenarios.

The above models have been widely applied in the field of time series prediction with the advantages of simplicity, effectiveness and strong interpretability. But they have high requirements for data (the original data or the data after difference must satisfy the stationarity hypothesis), and can only identify linear associations in time, making them face more and more restrictions in the context of continuously enriching time series data types.

2.2. High-dimensional data predictors

Due to the complex interactions between different variables of high-dimensional data, simple linear models cannot be used to describe such systems and make accurate predictions. Machine learning, deep learning, and some dynamics-based methods have better performance on this task.

A lot of machine learning methods have been applied to predict high-dimensional data. Support vector regression (SVR) has advantages in solving nonlinear regression by introducing the kernel methods, it has been widely researched in the field of financial time series forecasting, from single use [24] to combination with evidence framework [25] and random forest [8]. Bayesian Network (BN) use probabilistic networks for uncertainty reasoning [9], after combining domain knowledge and BN-based semantic network, the multivariate forecasting task on meteorological time series can be completed well [10]. Gauss process (GP) can also utilize kernel methods to enhance prediction capability, in order to solve the problem of high computational complexity, some studies have combined it with the KNN and Kalman filter to calculate more efficiently [11].

As some machine learning methods need to manually select kernels, and their computational complexity is unacceptable while processing high-dimensional data, several deep learning methods have become feasible alternatives. Feedforward neural network (FNN) is the simplest artificial neural network, it can be used to analyse time series directly [12] or combined with autoencoders to predict indoor temperature [13] and traffic flow [14]. Convolutional neural network (CNN) specializes in processing data with grid structures, some researchers have used two CNNs to convolve the input data on the rows and columns to predict asynchronous time series [15]. Recursive neural network (RNN) can capture the association of data for a longer time with the use of the saved states, and has been widely used in time series prediction and natural language processing. Further more, long short-term memory (LSTM) with attention can overcome the vanishing gradient problem in RNN, enabling it to predict financial time series on a larger time scale [16]. Besides, the mixed use of multiple models has become a new trend. For example, long- and short-term time-series network tries to use CNN to extract the correlation between data in a short time scale, and use RNN to extract the correlation between data in a long time scale [17]; LSTM networks with temporal pattern attention use CNN to capture the signal pattern in the hidden layer of RNN [18]. The shortcomings of deep learning methods include complex model structure, poor interpretability, and large demand for data. Considering that complex systems composed of multiple variables often have varing degrees of chaos [26], dynamics-based models are expected to perform better in this case.

Under the dynamics framework, high-dimensional variables and their changing process over time constitute a dynamic system [27], so there are many methods based on dynamics to extract the spatial information from complex interactions among variables and predict the changing trend of the system. The common goal of these methods is to obtain the mapping from the original attractor to the delay attractor of the system, one idea is to use compressed sensing algorithms to directly calculate the mapping [19], another way is to decompose the mapping into several sub-mappings, then use GP [20] or FNN [21] to solve them.

The above methods have their own advantages and disadvantages, but they all focus on extracting the temporal or spatial characteristics of the data, and cannot combine the two well. For this reason, we propose STSM to make up for this deficiency.

3. Model

3.1. Problem Formulation

In this work, we focus on inferring the evolution trend of a single variable from the overall dynamics of a high-dimensional system. Suppose one system contains N variables including $x_1, x_2, \ldots, x_N$, when N is large enough, the system is high-dimensional. We make M continuous observations to get a original attractor $\mathscr{O}$. If the observation starts at time t, the interval between observations is τ, then the time point of the i-th observation can be written as $t_i = t + i\tau$, and the value of x_j at t_i is $x_j(t_i)$. The content of $\mathscr{O}$ can be written as a matrix $\mathbf{X} \in R^{N\times M} = [\mathbf{X}(t_1), \mathbf{X}(t_2), \ldots, \mathbf{X}(t_M)]$, where $\mathbf{X}(t_m) = [x_1(t_m),\ x_2(t_m), \ldots,\ x_N(t_m)]^T$.

Suppose x_k is the target variable to be predicted. If our goal is to predict L steps forward for x_k, the model's task is to predict values of $\hat{x}_k^L = [\hat{x}_k(t_{M+1}), \hat{x}_k(t_{M+2}),\ \ldots,\ \hat{x}_k(t_{M+L})]$. But we don't predict $\hat{x}_k^L$ directly, instead, we compute the delay attractor $\mathscr{D}$ for x_k and extract $\hat{x}_k^L$ from it. The results of expanding $\mathbf{X}$ and $\mathscr{D}$ by elements are as follows, if the box counting dimension of the attractor $\mathscr{O}$ is d and $L > 2d - 1$, then the delay embedding theorem shows that, there exists a mapping from $\mathscr{O}$ to $\mathscr{D}$ [22], and we aim to solve the mapping with STSM.

$$\mathbf{X} = \begin{bmatrix} x_1(t_1) & x_1(t_2) & \cdots & x_1(t_M) \\ x_2(t_1) & x_2(t_2) & \cdots & x_2(t_M) \\ x_3(t_1) & x_3(t_2) & \cdots & x_3(t_M) \\ x_4(t_1) & x_4(t_2) & \cdots & x_4(t_M) \\ \vdots & \vdots & \vdots & \vdots \\ x_N(t_1) & x_N(t_2) & \cdots & x_N(t_M) \end{bmatrix}, \tag{4}$$

$$\mathscr{D} = \begin{bmatrix} x_k(t_1) & x_k(t_2) & \cdots & x_k(t_M) \\ x_k(t_2) & x_k(t_3) & \cdots & x_k(t_{M+1}) \\ x_k(t_3) & x_k(t_4) & \cdots & x_k(t_{M+2}) \\ x_k(t_4) & x_k(t_5) & \cdots & x_k(t_{M+3}) \\ \vdots & \vdots & \vdots & \vdots \\ x_k(t_{L+1}) & x_k(t_{L+2}) & \cdots & x_k(t_{M+L}) \end{bmatrix}. \tag{5}$$

Next, we will introduce our model from whole to part, the meaning of notations used in this section can be found from Table 1.

3.2. Model Architecture

As illustrated in Fig. 1, the STSM can be abstracted as a mapping $\mathbf{\Psi}$ totally, which can compute delay attractor $\mathscr{D}$ from original attractor $\mathscr{O}$. Note that to align dimensions and simplify calculations, we take the transposition of attractors as input and output, $\mathbf{\Psi}$ can be defined as follows.

$$\mathbf{\Psi}\left(\mathbf{X}^T \in R^{M\times N}\right) = \mathscr{D}^T \in R^{M\times(L+1)}. \tag{6}$$

The $\mathbf{\Psi}$ here can be regarded as a multi-step predictor, which can predict the L-step values of x_k once. In fact, it can be decomposed into several single-step predictors $\{\mathbf{\Psi}_1, \mathbf{\Psi}_2, \ldots, \mathbf{\Psi}_L, \mathbf{\Psi}_{L+1}\}$, and the meaning of $\mathbf{\Psi}_i$ is:

$$\mathbf{\Psi}_i\left(\mathbf{X}^{\mathbf{T}}(t_m)\right) = x_k(t_{m+i-1}). \tag{7}$$

Then we can write above formula in matrix form, the right half of the equation is the transposition of the required delay attractor $\mathscr{D}$:

$$\mathbf{\Psi}\left(\mathbf{X}^T\right) = \begin{bmatrix} \mathbf{\Psi}_1(\mathbf{X}(t_1)) & \mathbf{\Psi}_2(\mathbf{X}(t_1)) & \cdots & \mathbf{\Psi}_{L+1}(\mathbf{X}(t_1)) \\ \mathbf{\Psi}_1(\mathbf{X}(t_2)) & \mathbf{\Psi}_2(\mathbf{X}(t_2)) & \cdots & \mathbf{\Psi}_{L+1}(\mathbf{X}(t_2)) \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{\Psi}_1(\mathbf{X}(t_M)) & \mathbf{\Psi}_2(\mathbf{X}(t_M)) & \cdots & \mathbf{\Psi}_{L+1}(\mathbf{X}(t_M)) \end{bmatrix} = \begin{bmatrix} x_k(t_1) & x_k(t_2) & \cdots & x_k(t_{L+1}) \\ x_k(t_2) & x_k(t_3) & \cdots & x_k(t_{L+2}) \\ \vdots & \vdots & \vdots & \vdots \\ x_k(t_M) & x_k(t_{M+2}) & \cdots & x_k(t_{M+L}) \end{bmatrix} = \mathscr{D}^T. \tag{8}$$

Table 1
Notations used in the model.

Parameter	Description
N	The number of variables
M	The time length of the known data
$\mathbf{X^T} \in R^{M\times N}$	The input of our model
$\mathbf{X_i^T} \in R^{i\times N}$	The value of $\mathbf{X^T}$ until the i-th time point
$\mathbf{X^T}(t_i) \in R^{1\times N}$	The value of $\mathbf{X^T}$ at the i-th time point
L	The time length to predict
x_k	The target variable to be predicted
$\hat{x}_k^L$	The predicted values of x_k
$x_i(t_j)$	The i-th variable at the j-th time point
ψ	The mapping used to obtain time features
t_f	The dimension of time features
$\mathscr{F} \in R^{M\times t_f}$	The time features extracted by ψ
ψ_i	The i-th sub-mapping for time features
$\mathscr{F}_i \in R^{1\times t_f}$	The time features extracted by ψ_i
C	The kind of kernels in size of temporal module
n_c	The number of the c-th kind of kernels
cn_i	The number of kernel types available for ψ_i
ϕ	The mapping used to obtain space features
s_f	The dimension of space features
$\mathscr{G} \in R^{M\times s_f}$	The space features extracted from $\mathbf{X^T}$ by ϕ
ϕ_i	The i-th sub-mapping for space features
$\mathscr{G}_i \in R^{1\times s_f}$	The space features extracted by ϕ_i
$\mathscr{O}$	The original system attractor
d	The box counting dimension of $\mathscr{O}$
$\mathscr{D}$	The delay attractor for x_k
$\mathbf{\Psi}$	The mapping from $\mathbf{X^T}$ to $\mathscr{D}^{\mathscr{T}}$
$\mathbf{\Psi}_i$	The sub-mapping of $\mathbf{\Psi}$
$\mathscr{H}$	The mixture of $\mathbf{X}^T, \mathscr{G}$ and $\mathscr{F}$
Φ	The mapping from $\mathscr{H}$ to $\mathscr{D}^{\mathscr{T}}$

The detailed structure of STSM is shown in Fig. 1, it combines a temporal module, a spatial module and a skip connection to make predictions. The spatial module uses a mapping ϕ to make spatial transformation and get space features $\mathscr{G}$ from $\mathbf{X}^T$, and the temporal module uses a mapping ψ to make tempotal transformation and get time features $\mathscr{F}$ from $\mathbf{X}^T$. In order to reduce the difficulty of gradient solution caused by too deep network and keep more global information, we create a skip connection connected to $\mathbf{X}^T$. The mixed $\mathscr{H}$ concatenating $\mathscr{G}, \mathscr{F}$ and $\mathbf{X}^T$ will be sent to a mapping Φ implemented by a simple forward neural network to get $\mathscr{D}^T$. the loss function $\mathscr{L}$ is the root mean square error function of $\mathscr{D}^T$ as follows. The structure of temporal module and spatial module will be intruduced in next sections.

$$\mathscr{L}(\mathbf{\Psi}(\mathbf{X}^T), \mathscr{D}^T) = \mathscr{L}(\mathbf{\Phi}(\mathscr{H}), \mathscr{D}^T) = \mathscr{L}(\mathbf{\Phi}([\phi(\mathbf{X}^T), \psi(\mathbf{X}^T), \mathbf{X}^T]), \mathscr{D}^T). \tag{9}$$

3.3. Temporal module

Considering the evolution of time series data in time, the current value of time series is often affected by historical values, so we use temporal module to make temporal transformation and reflect this hypothesis. The temporal module aims to mine temporal information from historical values of system variables. If the dimension of time features is t_f, the module can be defined as the following mapping ψ:

$$\psi(\mathbf{X^T} \in R^{M\times N}) = \mathscr{F} \in R^{M\times t_f}. \tag{10}$$

As the basic assumption of the temporal module is that the time features of the current time point depend on the comprehensive consideration from the characteristics of the past several time points, so when we try to obtain the time features of the i-th time points, the previous data from 1-th to i-th is required. If $\mathbf{X_i} = [\mathbf{X}(t_1), \mathbf{X}(t_2), \ldots, \mathbf{X}(t_i)]$, the sub-mapping ψ_i is defined to get time features of the i-th time point, where $i = 1, 2, \ldots, M$:

$$\psi_i([\mathbf{X}(t_1), \mathbf{X}(t_2), \ldots, \mathbf{X}(t_i)]^T) = \psi_i(\mathbf{X_i^T} \in R^{i\times N}) = \mathscr{F}_i \in R^{1\times t_f}. \tag{11}$$

In order to obtain time features of different scales, the mapping ψ contains several convolution kernels with increasing size. If there are C kinds of kernels in size totally, the c-th kind of kernels' size is $[c, N]$ for c in $1, 2, \ldots, C$, and the number of the c-th kind of kernels ('channels' in other words) can be written as n_c. For one sub-mapping ψ_i, as its input $\mathbf{X_i^T} \in R^{i\times N}$, the used ker-

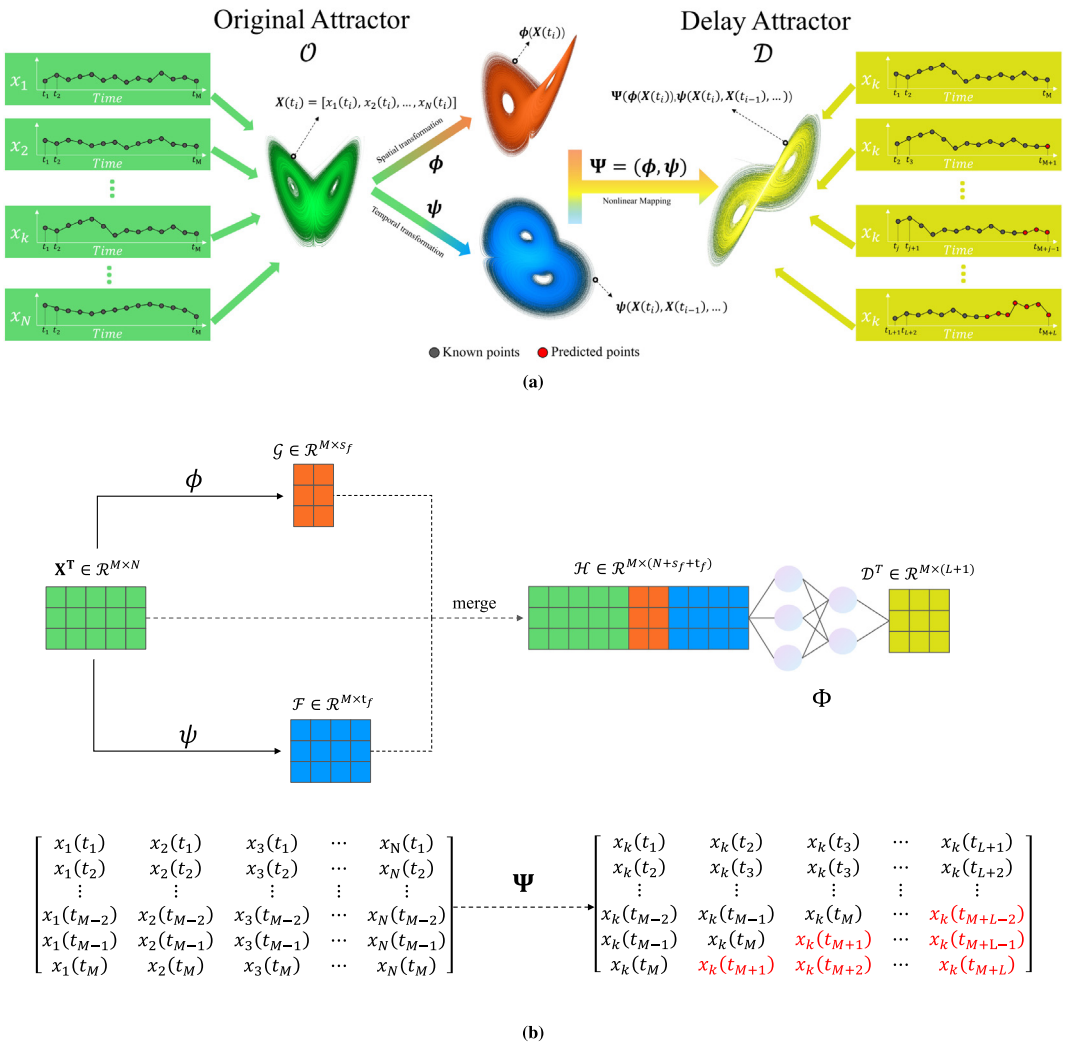

Fig. 1. Neural network calculation process of mapping $\mathbf{\Psi}$. (a) The calculation process at attractor level, the green part is the original attractor composed of several time series on the left side, and the orange part which becomes more distorted in shape is the result of spatial transformation by mapping ϕ, the blue part which becomes more continuous in shape is the result of temporal transformation by mapping ψ. Finally, these parts are used to construct the yellow part, which corresponds to the delay attractor, and on the right is the delayed time series it contains. The circle marked on the figure shows how a point on the original attractor is transformed into the final state by these mappings. (b) The calculation process at matrix level, note that the matrix is one-to-one in color with the attractor above, and the value of the specific elements in the original attractor and the delay attractor is displayed at the bottom, in which the red value is to be predicted.

nels' widths are not allowed to be more than that of input data, so the maximum number of available convolution kernel types for ψ_i is $cn_i = min(C, i)$.

There are three steps to calculate $\mathscr{F}_i$: multiple convolution for multi-scale information; max pooling for most important information; merging and forward for final features. Take ψ_i as example, after first step, we can get cn_i feature maps with size $[n_1, i], [n_2, i-1], \ldots, [n_{cn_i}, i-cn_i+1]$ respectively; then, sizes of these feature maps are reduced to $[1, i], [1, i-1], \ldots, [1, i-cn_i+1]$ respectively by max pooling; finally, we merge these feature maps to get a vector with size $[1, \sum_{j=1}^{cn_i} i-j+1]$, this vector will be fed into fully connected layers to obtain $\mathscr{F}_i$ with size $[1, t_f]$. The details are shown in Fig. 2. After getting each $\mathscr{F}_i$, it's easy to concat them to get $\mathscr{F}$, that's the overall framework of the temporal module.

3.4. Spatial module

Considering that different variables in the system will influence each other, we use spatial module to make the spatial transformation and obtain the space features. If the dimension of space features is s_f, the module can be defined as the following mapping ϕ:

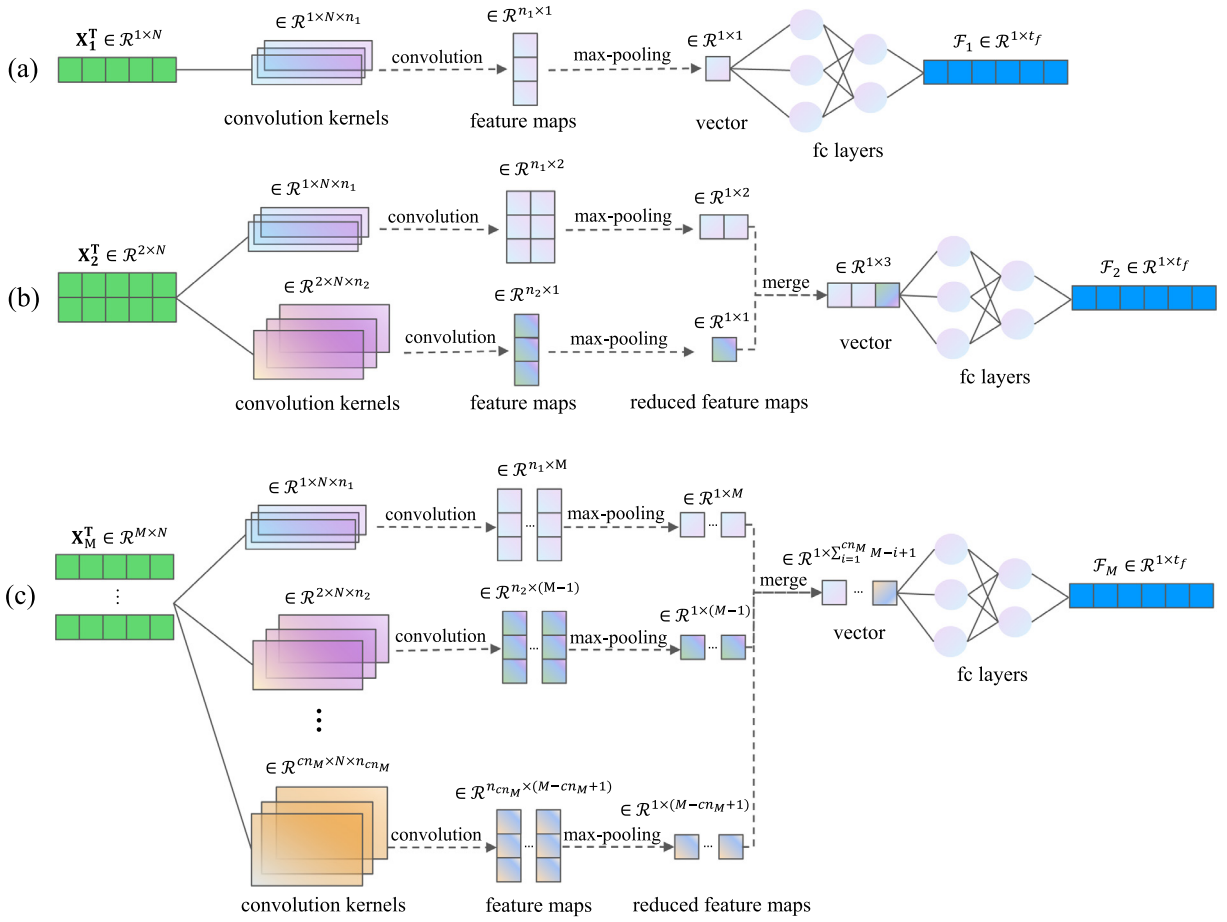

Fig. 2. Neural network calculation process of sub-mapping ψ_i. (a) The schematic illustration of the sub-mapping ψ_1. (b) The schematic illustration of the sub-mapping ψ_2, compared with ψ_1, the size of input data and the number of available kernels have changed a lot, you can see it from blocks of different colours. (c) The schematic illustration of the sub-mapping ψ_M, it can be inferred easily according to the rules shown in the figure.

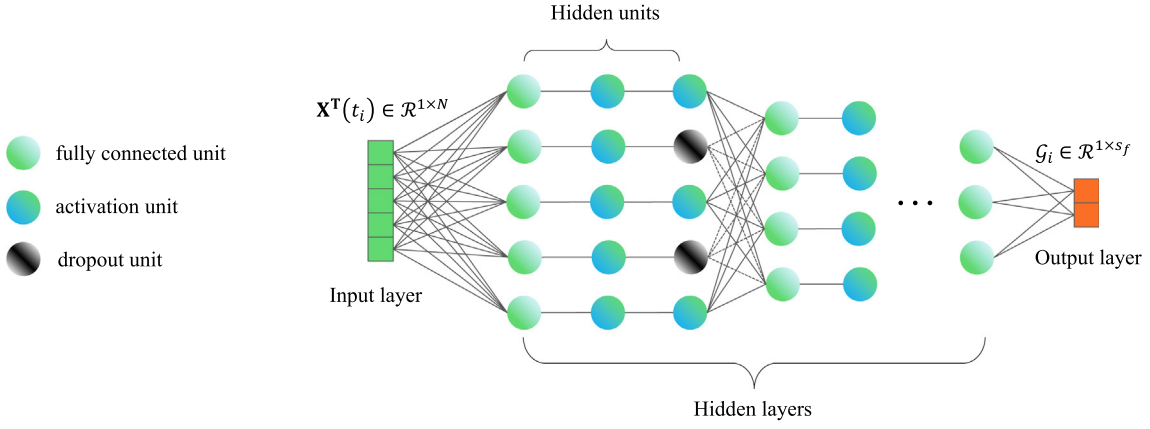

Fig. 3. Neural network calculation process of sub-mapping ϕ_i. The feedforward neural network can be divided into input layer, hidden layer and output layer, note that the hidden layer is composed of several hidden units.

$$\phi(\mathbf{X}^{\mathbf{T}} \in R^{M\times N}) = \mathcal{G} \in R^{M\times s_f}. \tag{12}$$

Similarly, we can define submapping ϕ_i to get space features of the i-th time point, where $i = 1, 2, \ldots, M$:

$$\phi_i(\mathbf{X}^{\mathbf{T}}(t_i) \in R^{1\times N}) = \mathcal{G}_i \in R^{1\times s_f}. \tag{13}$$

As $\mathcal{G}_i$ is only affected by values of system variables at the current moment, so ϕ can be implemented by a feedforward neural network shown in Fig. 3, and the simple structure performs well in the experiment.

4. Experiments

This chapter mainly introduces some details of our experiments, including information of datasets, experimental setup, evaluation indicators, experimental results and corresponding analysis. The contrast experiments and ablation experiments are carried out on synthetic datasets and real-world datasets, and we do several robustness experiments to test the stability of STSM in the end, they will be introduced respectively below.

4.1. Experimental Setting

In the real world, with the development of big data, more and more high-dimensional data has been generated from various scenarios like meteorological forecast, traffic monitoring and gene sequencing. To test the ability of our model to solve real problems, we do experiments on one synthetic dataset based on Lorentz system and seven real-world datasets including gene expression dataset, wind speed dataset, air pollutants and inpatients dataset, traffic flow dataset, Plankton dataset, solar energy dataset and electricity consumption dataset. In order to quantify the prediction effect, we will introduce some evaluation indicators. Assuming that the predicted length of the target variable is L, the estimated value calculated by a certain model is $\hat{y}$, and the actual value is y, then the following indicators are used to measure the effectiveness of the model: mean absolute error (MAE), root mean squard error (RMSE), pearson correlation coefficient (Pearsonr). Note that MAE and RMSE focus on the absolute error of predicted values, while Pearsonr is more sensitive to the trend of predicted values, they are calculated as follows.

$$MAE(y,\hat{y}) = \frac{1}{L}\sum_{i=1}^{L}|y_i - \hat{y}_i|. \tag{14}$$

$$RMSE(y,\hat{y}) = \sqrt{\frac{1}{L}\sum_{i=1}^{L}(y_i - \hat{y}_i)^2}. \tag{15}$$

$$Pearsonr(y,\hat{y}) = \frac{L\sum_{i=1}^{L} y_i\hat{y}_i - \sum_{i=1}^{L} y_i \sum_{i=1}^{L}\hat{y}_i}{\sqrt{L\sum_{i=1}^{L} y_i^2 - \left(\sum_{i=1}^{L} y_i\right)^2}\sqrt{L\sum_{i=1}^{L}\hat{y}_i^2 - \left(\sum_{i=1}^{L}\hat{y}_i\right)^2}}. \tag{16}$$

4.2. Contrast & ablation experiments

To test the prediction effect of STSM, we do contrast experiments to compare STSM with other models including autoregressive model (AR), autoregressive integrated moving average model (ARIMA), and long short-term memory (LSTM). In order to verify that our idea of combining time features and space features is reasonable, we test the performance of STSM, spatiotemporal model as well as spatial model and temporal model (the space features and time features will be sent to a simple feedforward neural network directly to predict for the last two) in ablation experiments, note that spatiotemporal model can be regarded as the combination of the spatial model and temporal model or a simplified STSM without skip connection. These two experiments are carried out on all datasets at the same time, the results are shown below.

4.2.1. Lorentz dataset

To validate our model's abiltity of capturing the dynamics of high-dimensional nonlinear system, we construct a coupled Lorentz system with 90 variables. The i-th ($i = 1, 2, \ldots, 30$) coupled subsystem contains three variable a, b, c, which can be given by

$$\begin{cases} \dot{a}_i = \sigma(t)(b_i - a_i) + Ca_{i-1}, \\ \dot{b}_i = \rho a_i - b_i - a_i c_i, \\ \dot{c}_i = -\beta c_i + a_i b_i \end{cases} \tag{17}$$

In our experiment, we set $\rho = 28,\ \beta = \frac{8}{3},\ \ C = 0.1$ to compute variables in the system. Among all 90 variables, we select three variables called 'x, y, z' randomly to predict. M for each variable is 40 and N is 90, the embedding length is 19 (in other words, L is 18).

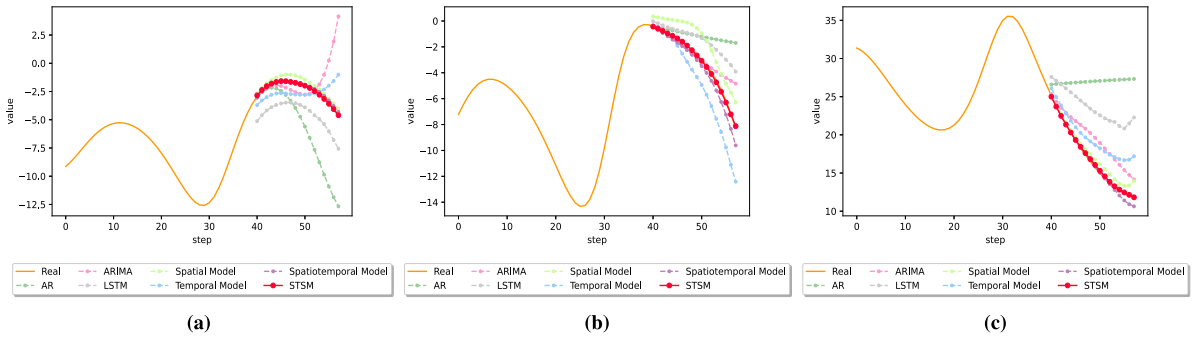

Fig. 4. Performance of different models on the Lorentz dataset. (a) Performance of different models on Lorentz-x. (b) Performance of different models on Lorentz-y. (c) Performance of different models on Lorentz-z.

From Fig. 4 we can see that compared with individual spatial model and temporal model, STSM and spatiotemporal model preform better on the Lorentz system. It's worth noting that the predicted results of spatial model are closer to the true values than those of temporal model, indicting that space features are more important than time features while predicting variables from the Lorentz system, that's becauce the system is time-invariant under our experimental setup. So we make dimension of time features (t_f) smaller than dimension of space features (s_f) in STSM and spatiotemporal model to make full use of this point. Detailed evaluation indicators are shown in the table, note that the best indicators for each line are highlighted in bold.

Although STSM and spatiotemporal model are difficult to distinguish on curves, it can be seen that the former is still better than the latter from Table 2 on almost all indicators of different variables in the Lorentz system, showing the strong competitiveness of the STSM model on synthetic datasets.

4.2.2. Gene expression dataset

This dataset contains the gene expression profiles measured on the laboratory rat cultured cells from SCN, the expression of 31099 genes is recorded on 23 time points [28]. To make the calculation process more efficient, we reduce the dim N to 84, several genes about circadian rhythm are selected for prediction. We use 8 time points as the training data and make predictions on the next 4 time points (M = 8, L = 4).

The results of contrast and ablation experiments are shown in Fig. 5, STSM has a better performance. Detailed evaluation indicators are shown in Table 2, the effect of our model is significantly improved compared to similar models on most indicators.

4.2.3. Wind speed dataset

This dataset contains the wind speed (m/s) time series of 155 stations in Japan [29], which are sampled every 10 min from 2010 to 2012, so N (the dimention of the system) is 155. We resample the dataset with time interval $\delta t = 1$ hour to reduce the influence of noise and accidental factors. We use 100 time points as the training data and make predictions on the next 60 time points (M = 100, L = 60).

We do several experiments introduced above, and the results are shown in Fig. 6(a), as the curve of temporal model is too flat to reflect the trend, we give the spatial model higher weight. Detailed evaluation indicators are shown in Table 2, our model performs better than similar models on all indicators.

4.2.4. Air pollutants and inpatients dataset

This dataset contains the cardiovascular inpatients and common air pollutant indices from 1994 to 1995 in HK [30], the dim N is 22. We use 72 time points as the training data and make predictions on the next 36 time points (M = 72, L = 36).

The results of experiments are shown in Fig. 6(b), it's clear that STSM preforms best on the dataset. Detailed evaluation indicators are shown in Table 2, our model performs better than similar models on all indicators. Note that the curves of both spatial model and temporal model are almost straight, but their combination performs well, indicting the validity of STSM.

4.2.5. Traffic flow dataset

This dataset is obtained from the Caltrans Performance Measurement System (PeMS), containing the traffic flow of several stations within half a year in California, the dim N (the number of sampling points) is 33. The raw data is sampled every 5 min, to get more stable results, we resample the dataset with time interval $\delta t = 15$ minutes. We use 64 time points as the training data and make predictions on the next 31 time points (M = 64, L = 31).

The results of different models are shown in Fig. 6(c), STSM makes the most accurate prediction. Detailed evaluation indicators are shown in Table 2, our model performs better than similar models on all indicators. Note that the curve of spatial

Table 2
Indicitors of different models on target variables from different datasets, variables randomly selected by index are represented by'-', the best indicators are shown in bold.

Dataset	Variable	Indicator	AR	ARIMA	LSTM	Spatial Model	Temporal Model	Spatiotemporal Model	STSM
Lorentz	x	MAE	3.29	1.50	2.14	0.47	1.12	0.13	**0.05**
		RMSE	4.34	2.77	2.16	0.48	1.37	0.17	**0.05**
		Pearsonr	0.8552	-0.8548	**0.9986**	0.9928	-0.6954	0.9936	0.9984
	y	MAE	2.01	0.65	1.64	1.50	1.72	0.54	**0.08**
		RMSE	2.79	1.15	2.02	1.55	2.22	0.65	**0.10**
		Pearsonr	0.9437	0.9491	0.9985	0.9851	0.9961	**0.9999**	0.9993
	z	MAE	10.02	2.70	6.74	0.80	2.82	0.32	**0.17**
		RMSE	10.93	2.95	7.06	0.94	3.11	0.52	**0.20**
		Pearsonr	-0.9919	0.9738	0.9882	0.9985	0.9930	0.9959	**0.9992**
Gene expression	Kif3c	MAE	11.33	10.27	8.39	9.97	11.63	8,85	**6.90**
		RMSE	12.27	11.24	9.88	12.10	14.30	10.63	**8.11**
		Pearsonr	-0.3110	0.0777	0.5573	0.8301	0.9250	0.8682	**0.9693**
	Bmal1	MAE	23.80	16.05	13.37	12.77	14.85	5.70	**4.91**
		RMSE	25.19	16.31	19.64	13.43	18.12	7.59	**5.62**
		Pearsonr	-0.4923	0.5046	0.5779	**0.9925**	0.8771	0.9924	0.9755
	Cry1	MAE	17.95	18.39	35.93	16.24	24.18	14.16	**9.46**
		RMSE	23.15	22.22	47.70	18.05	31.60	16.74	**12.29**
		Pearsonr	0.9090	0.8799	-0.8608	0.9679	0.9358	0.9677	**0.9696**
	Mapk6	MAE	17.40	19.17	16.27	32.34	16.70	15.44	**10.42**
		RMSE	17.90	20.96	17.17	34.86	19.69	15.66	**13.36**
		Pearsonr	-0.5634	-0.7734	0.7789	**0.9876**	0.8756	0.9468	0.9800
Wind speed	-	MAE	1.52	1.49	1.11	0.84	1.54	0.40	**0.39**
		RMSE	1.87	1.86	1.48	1.14	1.79	0.72	**0.57**
		Pearsonr	-0.3409	0.1393	0.8128	0.8286	-0.5145	0.9291	**0.9598**
Air pollutants and inpatients	-	MAE	24.17	23.16	23.78	24.91	25.50	11.40	**10.41**
		RMSE	28.44	28.10	30.51	28.87	29.13	14.00	**12.30**
		Pearsonr	0.1885	0.2397	-0.0652	-0.2527	-0.2626	0.8908	**0.9153**
Traffic flow	-	MAE	49.02	56.41	75.88	44.13	25.10	10.98	**9.88**
		RMSE	54.44	70.05	91.25	49.02	29.98	14.61	**13.74**
		Pearsonr	0.9448	0.9761	-0.9362	0.9684	0.9592	**0.9866**	0.9860
Plankton	oxygen	MAE	0.17	0.32	0.45	0.64	0.48	0.28	**0.03**
		RMSE	0.20	0.33	0.45	0.64	0.48	0.28	**0.03**
		Pearsonr	-0.9355	-0.2515	0.8785	**0.9693**	0.9595	0.9156	0.9025
Solar energy	-	MAE	9.58	8.85	1.79	3.01	9.89	0.43	**0.39**
		RMSE	10.02	9.51	2.66	4.76	11.17	0.93	**0.79**
		Pearsonr	-0.2869	0.1643	0.8606	0.3869	-0.4017	0.9824	**0.9881**
Electricity consumption	-	MAE	473.28	452.89	524.70	526.26	475.67	110.96	**92.68**
		RMSE	562.63	542.51	575.49	576.24	530.89	149.29	**119.36**
		Pearsonr	0.2990	0.3538	-0.3057	-0.0469	0.6455	0.9706	**0.9813**

model is higher than real data but the curve of temporal model is lower than real data, the spatiotemporal model and STSM model give the closest result.

4.2.6. Plankton dataset

This dataset is obtained from the optical plankton counter (OPC), including the changes of several biochemical indexes in seawater [31], the dim N (the number of the indexes) is 58. The raw data is updated every second. We use 12 time points as the training data and make predictions for dissolved oxygen concentration on the next 5 time points (M = 12, L = 5).

The experiment results are shown in Fig. 6(d), indicating that STSM preforms best on the dataset. Detailed evaluation indicators are shown in Table 2, our model performs better than similar models on most indicators. Note that the curve of spatial model is higher than real data and the curve of temporal model is lower than real data although they can reflect the trend of data, spatiotemporal model has reduced the prediction error greatly, while STSM model combines their advantages well and gives a perfect result.

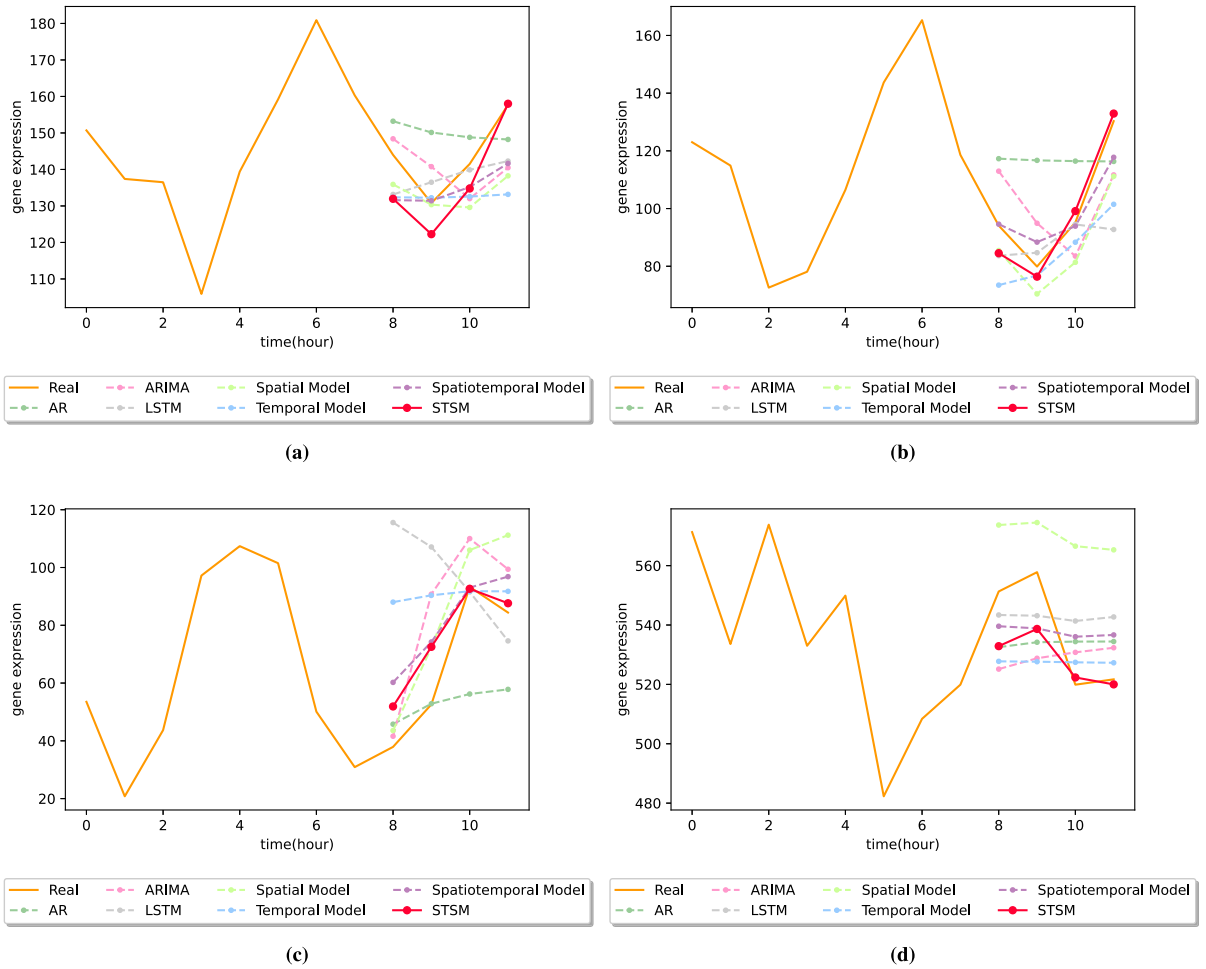

Fig. 5. Performance of different models on the gene expression dataset. (a) Performance of different models on kif3c. (b) Performance of different models on Bmal1. (c) Performance of different models on Cry1. (d) Performance of different models on Mapk6.

4.2.7. Solar energy dataset

This dataset contains the solar power production records in the year of 2006, which is sampled every 10 min from 137 PV plants in Alabama State [17], the dim N is 137. We use 200 time points as the training data and make predictions for solar power production of a random plant on the next 100 time points (M = 200, L = 100).

The results of contrast and ablation experiments are shown in Fig. 6(e), indicating that STSM preforms best on the dataset. Detailed evaluation indicators are shown in Table 2, our model significantly outperforms similar models on all indicators.

4.2.8. Electricity consumption dataset

This dataset contains the electricity consumption in kWh recorded every 15 min from 2011 to 2014 [17], the dim N (the number of the clients) is 321. As some records are lost, the data in 2011 has been eliminated. The data is resampled to reflect hourly consumption. We use 100 time points as the training data and make predictions for dissolved oxygen concentration on the next 70 time points (M = 100, L = 70).

The results of different models are shown in Fig. 6(f), indicating that STSM preforms best on the dataset. Detailed evaluation indicators are shown in Table 2, indicating that our model is superior over other competitors.

4.2.9. Learning issues

In actual practice, the scale of data varies for specific scenes, and the leaning curve of prediction model is expected to converge rapidly under different conditions. To check this out, we draw the loss curves of STSM for different datasets in Fig. 7. Note that for the sake of comparison, the loss values differing greatly from each other have been scaled logarithmically with appropriate bases. Fig. 7a) and Fig. 7(b) show loss curves of different variables in the same dataset, while Fig. 7(c) show loss curves of variables from different datasets. It can be found that the loss curves of variables from the same dataset or

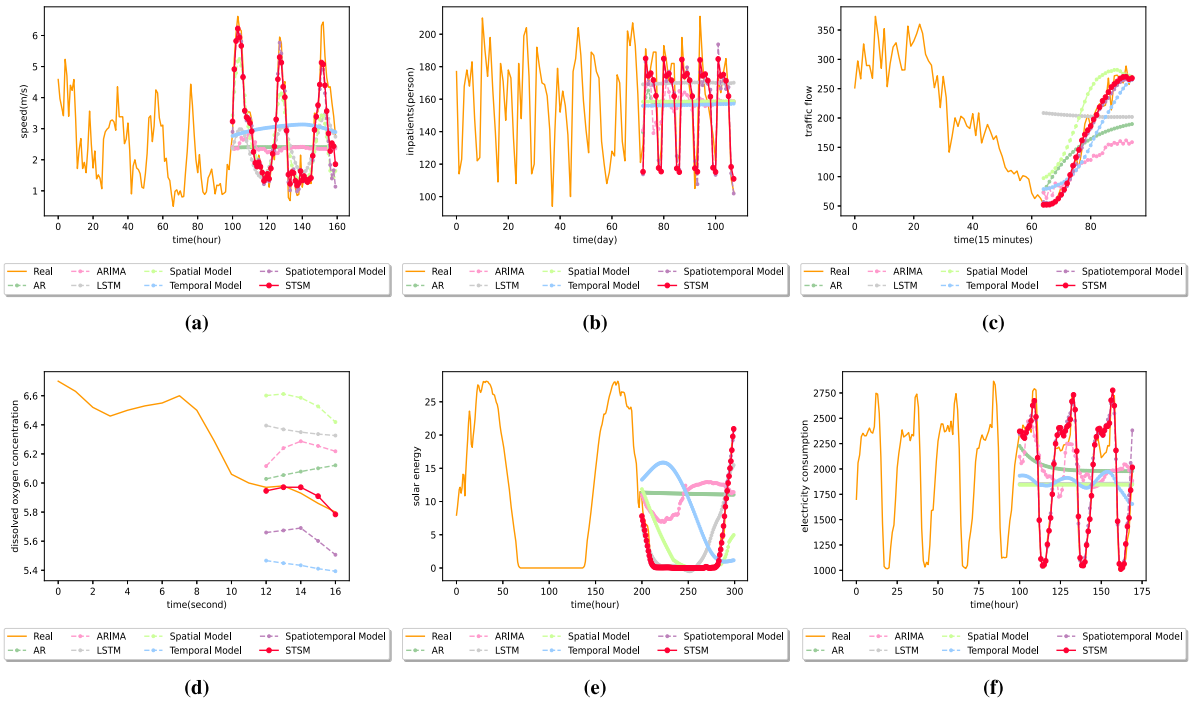

Fig. 6. Performance of different models on several real-world datasets. (a) Performance of different models on the wind speed dataset. (b) Performance of different models on the air pollutants and inpatients dataset. (c) Performance of different models on the traffic flow dataset. (d) Performance of different models on the plankton dataset. (e) Performance of different models on the solar energy dataset. (f) Performance of different models on the electricity consumption dataset.

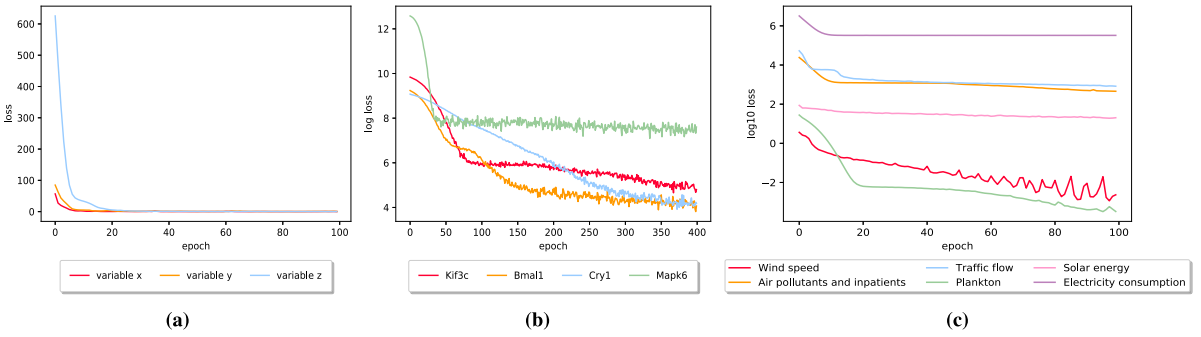

Fig. 7. Loss curves of STSM on different datasets. (a) Original loss curves of different variables on the Lorentz dataset. (b) Loss curves of different variables on the Gene dataset plotted in the base-*e* logarithmic scale. (c) Loss curves of different datasets plotted in the base-10 logarithmic scale.

different datasets all converge within acceptable epochs regardless of the difference in scale, proving the effectiveness of STSM in learning issues.

4.2.10. Quantitative analysis

In this part, we provide a quantitative perspective to analyze above results. To make dimensionless comparison, the difference in MAE and RMSE is measured by percentage, and results in Pearson are simply subtracted to evaluate the improvements considering the existence of negative values. For simplicity, we denote contrast models as the collection of AR, ARIMA, LSTM; and consider ablation models including spatial model, temporal model, spatiotemporal model.

For the synthetic Lorentz dataset, STSM model is 95.68% and 82.36% lower in MAE, 96.23% and 84.89% lower in RMSE, 0.4589 and 0.1924 higher in Pearsonr compared with contrast models and ablation models respectively. For the seven real-world datasets, STSM model is 66.73% and 52.16% lower in MAE, 63.88% and 50.87% lower in RMSE, 0.8287 and 0.2600 higher in Pearsonr compared with contrast models and ablation models respectively.

More specifically, we compare STSM and ablation models on all datasets. Overall, the spatiotemporal model is 56.20% and 50.36% lower in MAE, 59.98% and 58.54% lower in RMSE, 0.2178 and 0.4728 higher in Pearsonr compared with spatial models

and temporal models respectively attributed to the proper combination of spatial information as well as temporal information, the STSM model is 33.19% lower in MAE, 36.23% lower in RMSE, 0.0142 higher in Pearsonr compared with the spatiotemporal model due to the supplement of global information.

4.3. Robustness experiments

When our model is applied in the real world, it may encounter some challenges. In order to test the adaptability of STSM to external interference and its prediction potential, we conduct the following robustness experiments.

4.3.1. Time-varying robustness experiment

The original Lorentz system is time-invariant, to test STSM's ability to predict time-variant data, we modify the constants in the formula for generating Lorentz dataset into variables changing with time, the training length and prediction length are consistent with the previous ones. The experimental results are shown in the Fig. 8 and Table 3. Although the prediction effect of our model has declined, it is still acceptable. Besides, we compare the experimental results under two settings with

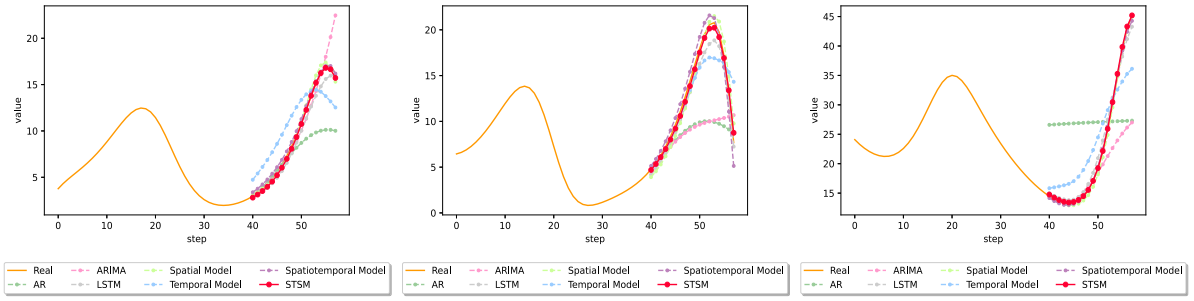

Fig. 8. Performance of different models on the time-variant Lorentz system. (a) Performance of different models on Lorentz-x. (b) Performance of different models on Lorentz-y. (c) Performance of different models on Lorentz-z.

Table 3
Indicitors of different models on the time-variant Lorentz system.

Variable	Indicator	AR	ARIMA	LSTM	Spatial Model	Temporal Model	Spatiotemporal Model	STSM
Lorentz-x	MAE	2.46	0.90	0.75	0.37	2.54	0.36	**0.22**
	RMSE	3.60	1.74	0.84	0.46	2.69	0.40	**0.25**
	Pearsonr	0.9646	0.9639	0.9984	0.9966	0.9059	**0.9999**	0.9996
Lorentz-y	MAE	4.61	4.66	1.14	1.06	1.71	0.97	**0.41**
	RMSE	6.02	5.98	1.24	1.16	2.41	1.12	**0.46**
	Pearsonr	0.8998	0.7906	**0.9994**	0.9841	0.9116	0.9822	0.9979
Lorentz-z	MAE	10.87	4.31	1.03	0.46	4.06	0.33	**0.22**
	RMSE	11.66	7.48	1.17	0.57	4.54	0.40	**0.27**
	Pearsonr	0.8610	0.9792	0.9971	0.9995	0.9623	0.9993	**0.9999**

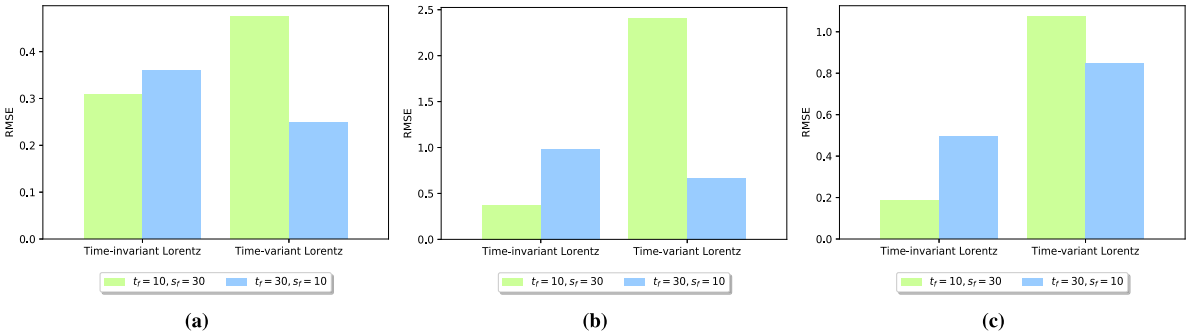

Fig. 9. Performance of STSM on the time-variant Lorentz system and time-invariant Lorentz system with different values for t_f and s_f. (a) Performance on variable x. (b) Performance on variable y. (c) Performance on variable z.

opposite values of time features (t_f) and space features (s_f), which are shown in the Fig. 9. It's worth noting that setting the dimension of t_f larger than dimension of s_f is helpful to get better results under the time-variant setting, while the opposite operation should be taken under the time-invariant setting. That's because the spatial correlation of time-variant systems is weaker than time-invariant systems, and the values of s_f and t_f reflect the preference of the STSM model for spatial and temporal information.

4.3.2. Long-term prediction experiment

In order to show the potential of STSM in long-term prediction, We design an iterative scheme on the Lorentz dataset to get longer result, in which the output of the previous prediction is used as the input of the next step. In this way, We use data from 40 time points to predict 400 time points in the future (M = 40, L = 400) like Fig. 10, detailed evaluation indicators are shown in Table 4.

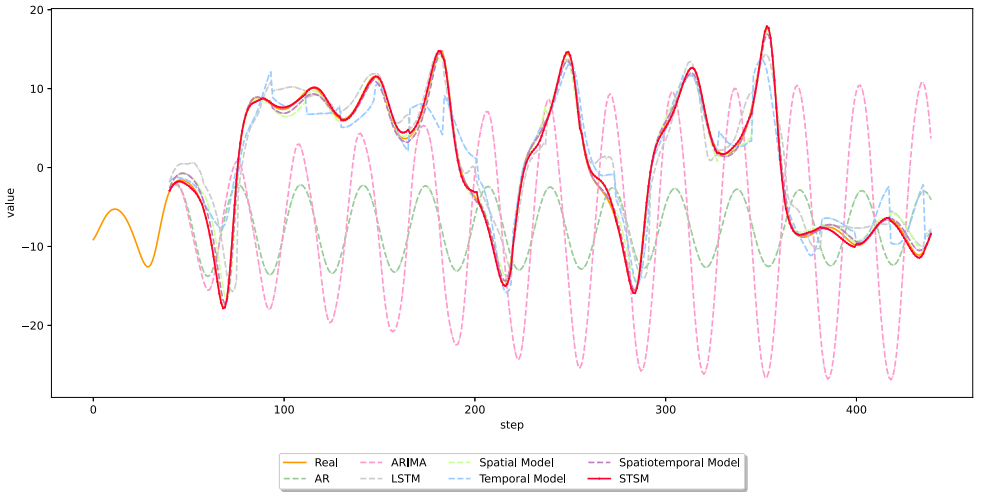

Fig. 10. Performance of different models on the long-term prediction for Lorentz-x..

Table 4
Indicitors of different models on the long-term prediction for Lorentz-x.

Variable	Indicator	AR	ARIMA	LSTM	Spatial Model	Temporal Model	Spatiotemporal Model	STSM
Lorentz-x	MAE	10.50	13.79	2.02	0.74	2.03	0.42	**0.29**
	RMSE	12.54	16.73	2.69	1.00	2.84	0.54	**0.37**
	Pearsonr	-0.0458	-0.1030	0.9531	0.9945	0.9470	0.9986	**0.9991**

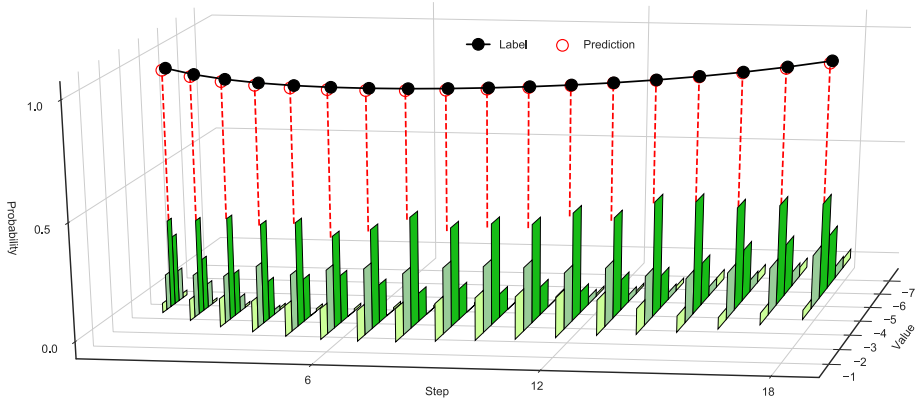

Fig. 11. Performance of STSM under defferent initialized neural networks on Lorentz dataset..

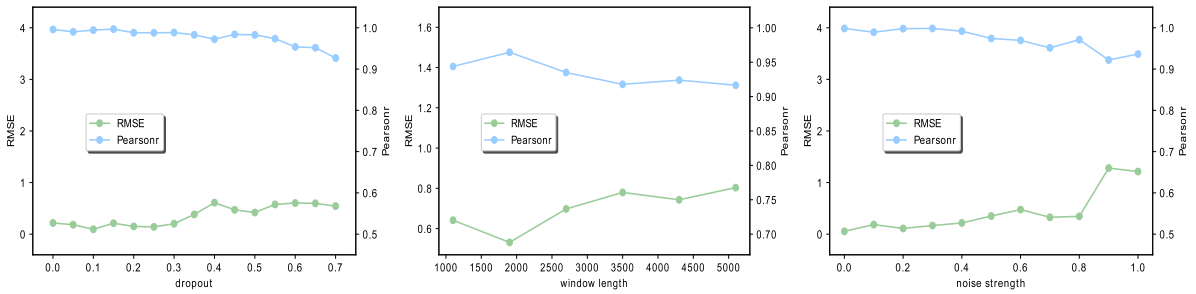

Fig. 12. Performance of STSM under different conditions. (a) Performance of STSM under defferent dropout on Lorentz dataset. (b) Performance of STSM under defferent window length on Wind speed dataset. (c) Performance of STSM under defferent noise strengths on Lorentz dataset.

4.3.3. Parameter robustness experiment

In this part, we initialize the network 100 times randomly to study the influence caused by random initialization of the neural network, for each step, we plot the 100 results into a histogram like Fig. 11. It's clear that the distribution is roughly normal and most results fall into a range near the center, indicating that our model has good adaptability to different initialization parameters.

4.3.4. Hyperparameter robustness experiment

The sensitivity of the model to hyperparameters is an important factor affecting the efficiency of training the model. If one model is too sensitive to hyperparameters, then users have to pay more time and resources to train a suitable version. To test robustness of STSM in this field, we test the performance of STSM on Lorentz and Wind speed dataset with different hyperparameters, the results are shown in Fig. 12(a) and (b). Note that the hyperpatameter 'dropout' is a ratio from zero to one controling the proportion of dropout operations, and the hyperparameter 'window length' is the time span of training data, we can see that the results don't change a lot as the horizontal axis coordinates grow in two figures, indicating the strong robustness for hyperparameters of our model. Besides, we can observe that the prediction effect increases first but then decreases with the increase of window length in Fig. 12(b), it can be explained that when the window length is too small, our model can't learn enough dynamic information from train data; and when the window length is too large, the dynamic knowledge learned by our model is not suitable for current stage.

4.3.5. Noise robustness experiment

Data in the real world is often not clean and contains a lot of noise. To test STSM's adaptation to the noise, we add gaussian white noise with different strength to the original data, and the results are shown in Fig. 12(c). It can be seen that the prediction effect of the model remains at a stable level when the noise strength does not exceed 0.8, considering the small floating range of the data, 0.8 has accounted for a considerable proportion.

5. Conclusion

In this paper, we introduce a new model called STSM to make accurate predictions for high dimensional time series data, which can help to meet some real-life needs such as precise weather forecast, timely traffic warning, stock price prediction and prevention for the rapid spread of infectious diseases like COVID-19. Different from traditional dynamic methods, STSM combines the temporal, spatial, and global information in the data properly through structural innovation. We prove the efficiency and rationality of our model by contrast and ablation experiments on different datasets, and we test the robustness of our model in different scenarios, the results give us positive feedback generally. One defect of our model is that we need to manually set the dimension of time and space features to reflect the inherent characteristics of different datasets. We hope that there will be a calculation formula or learning algorithm for this problem in the future. Generally speaking, our model applies a unique scheme of using data information in dynamic framework, and explores a new way for data-driven time series prediction.

CRediT authorship contribution statement

Jining Wang: Methodology, Software, Writing - original draft, Writing - review & editing. **Chuan Chen:** Conceptualization, Methodology, Writing - original draft, Writing - review & editing. **Zibin Zheng:** Resources, Supervision, Project administration, Funding acquisition. **Luonan Chen:** Conceptualization, Methodology, Resources. **Yuren Zhou:** Supervision, Validation.

Declaration of Competing Interest

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

Acknowledgment

The research is supported by the National Key R&D Program of China (2020YFB1006001), the National Natural Science Foundation of China (62176269), the Guangdong Basic and Applied Basic Research Foundation (2019A1515011043) and the Tencent Wechat Rhino-bird project (2021321).

References

[1] J. Fan, F. Han, H. Liu, Challenges of big data analysis, National science review 1 (2) (2014) 293–314.
[2] D.J. Lockhart, E.A. Winzeler, Genomics, gene expression and dna arrays, Nature 405 (6788) (2000) 827–836.
[3] M. Bosilovich, S. Schubert, G. Kim, R. Gelaro, M. Rienecker, M. Suarez, R. Todling, Nasa's modern era retrospective-analysis for research and applications (merra), in: AGU Spring Meeting Abstracts, Vol. 2007, AGU, 2006, pp. A43D–01..
[4] Y. Lv, Y. Duan, W. Kang, Z. Li, F.-Y. Wang, Traffic flow prediction with big data: a deep learning approach, IEEE Transactions on Intelligent Transportation Systems 16 (2) (2014) 865–873.
[5] H. Akaike, Fitting autoregressive models for prediction, Annals of the institute of Statistical Mathematics 21 (1) (1969) 243–247.
[6] R.J. Hyndman, G. Athanasopoulos, Forecasting: principles and practice, OTexts (2018).
[7] R.S. Tsay, An introduction to analysis of financial data with R, John Wiley & Sons, 2014.
[8] J. Patel, S. Shah, P. Thakkar, K. Kotecha, Predicting stock market index using fusion of machine learning techniques, Expert Systems with Applications 42 (4) (2015) 2162–2172.
[9] J. Pearl, Fusion, propagation, and structuring in belief networks, Artificial intelligence 29 (3) (1986) 241–288.
[10] M. Das, S.K. Ghosh, sembnet: a semantic bayesian network for multivariate prediction of meteorological time series data, Pattern Recognition Letters 93 (2017) 192–201.
[11] Y. Wang, B. Chaib-Draa, A knn based kalman filter gaussian process regression, in: Twenty-Third International Joint Conference on Artificial Intelligence, Citeseer, 2013, pp. 1771–1777.
[12] J.T. Turner, Time series analysis using deep feed forward neural networks, University of Maryland, Baltimore County, 2014.
[13] P. Romeu, F. Zamora-Martínez, P. Botella-Rocamora, J. Pardo, Time-series forecasting of indoor temperature using pre-trained deep neural networks, in: International conference on artificial neural networks, Springer, 2013, pp. 451–458.
[14] Y. Lv, Y. Duan, W. Kang, Z. Li, F.-Y. Wang, Traffic flow prediction with big data: a deep learning approach, IEEE Transactions on Intelligent Transportation Systems 16 (2) (2014) 865–873.
[15] M. Binkowski, G. Marti, P. Donnat, Autoregressive convolutional neural networks for asynchronous time series, in: International Conference on Machine Learning, PMLR, 2018, pp. 580–589.
[16] X. Zhang, X. Liang, A. Zhiyuli, S. Zhang, R. Xu, B. Wu, At-lstm: An attention-based lstm model for financial time series prediction, in: IOP Conference Series: Materials Science and Engineering, IOP Publishing, 2019, 052037.
[17] G. Lai, W.-C. Chang, Y. Yang, H. Liu, Modeling long-and short-term temporal patterns with deep neural networks, in: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, 2018, pp. 95–104.
[18] S.-Y. Shih, F.-K. Sun, H.-Y. Lee, Temporal pattern attention for multivariate time series forecasting, Machine Learning 108 (8) (2019) 1421–1441.
[19] H. Ma, T. Zhou, K. Aihara, L. Chen, Predicting time series from short-term high-dimensional data, International Journal of Bifurcation and Chaos 24 (12) (2014) 1430033.
[20] H. Ma, S. Leng, K. Aihara, W. Lin, L. Chen, Randomly distributed embedding making short-term high-dimensional data predictable, Proceedings of the National Academy of Sciences 115 (43) (2018) E9994–E10002.
[21] C. Chen, R. Li, L. Shu, Z. He, J. Wang, C. Zhang, H. Ma, K. Aihara, L. Chen, Predicting future dynamics from short-term time series using an anticipated learning machine, National Science Review 7 (6) (2020) 1079–1091.
[22] F. Takens, Detecting strange attractors in turbulence, in: Dynamical systems and turbulence, Warwick 1980, Springer, 1981, pp. 366–381..
[23] G.E. Box, G.M. Jenkins, G.C. Reinsel, G.M. Ljung, Time series analysis: forecasting and control, John Wiley & Sons, 2015.
[24] K.-J. Kim, Financial time series forecasting using support vector machines, Neurocomputing 55 (1–2) (2003) 307–319.
[25] T. Van Gestel, J.A. Suykens, D.-E. Baestaens, A. Lambrechts, G. Lanckriet, B. Vandaele, B. De Moor, J. Vandewalle, Financial time series prediction using least squares support vector machines within the evidence framework, IEEE Transactions on neural networks 12 (4) (2001) 809–821.
[26] Z. Chen, C. Chen, Z. Zheng, Y. Zhu, Tensor decomposition for multilayer networks clustering, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2019, pp. 3371–3378.
[27] C. Chen, Y. Li, H. Qian, Z. Zheng, Y. Hu, Multi-view semi-supervised learning for classification on dynamic networks, Knowledge-Based Systems 195 (2020) 105698.
[28] Y. Wang, X.-S. Zhang, L. Chen, A network biology study on circadian rhythm by integrating various omics data, OMICS A Journal of Integrative Biology 13 (4) (2009) 313–324.
[29] Y. Hirata, K. Aihara, Predicting ramps by integrating different sorts of information, The European Physical Journal Special Topics 225 (3) (2016) 513–525.
[30] T.W. Wong, T.S. Lau, T.S. Yu, A. Neller, S.L. Wong, W. Tam, S.W. Pang, Air pollution and hospital admissions for respiratory and cardiovascular diseases in hong kong, Occupational and environmental medicine 56 (10) (1999) 679–683.
[31] D.G. Kimmel, W.C. Boicourt, J.J. Pierson, M.R. Roman, X. Zhang, A comparison of the mesozooplankton response to hypoxia in chesapeake bay and the northern gulf of mexico using the biomass size spectrum, Journal of Experimental Marine Biology and Ecology 381 (2009) S65–S73.