Knowledge-Based Systems xxx (xxxx) xxx

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

CTS-LSTM: LSTM-based neural networks for correlated time series prediction☆,☆☆

Huaiyu Wan [a,b,c,*], Shengnan Guo [a,b], Kang Yin [a,c], Xiaohui Liang [a,c], Youfang Lin [a,b,c]

[a] School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China
[b] Beijing Key Laboratory of Traffic Data Analysis and Mining, Beijing, China
[c] CAAC Key Laboratory of Intelligent Passenger Service of Civil Aviation, Beijing, China

ARTICLE INFO

Article history:
Received 10 April 2019
Received in revised form 15 November 2019
Accepted 16 November 2019
Available online xxxx

Keywords:
Correlated time series prediction
Spatio-temporal correlation

ABSTRACT

Correlated time series refer to multiple time series which are recorded simultaneously to monitor the changing of multiple observations in a whole system. Correlated time series prediction plays a significant role in many real-world applications to help people make reasonable decisions. Yet it is very challenging, because different from single time series, correlated time series show both intra-sequence temporal dependencies and inter-sequence spatial dependencies. In addition, correlated time series are also affected by external factors in actual scenarios. Although RNNs have been proved to be effective on sequential data modeling, existing related works only focus on sequential patterns in a single time series, failing to comprehensively consider the inter-dependencies among multiple time series, which is essential for correlated time series prediction. In this paper, we propose a novel variant of LSTM, named CTS-LSTM, to collectively forecast correlated time series. Specifically, spatial and temporal correlations are explicitly modeled and respectively maintained in cells to capture the complex non-linear patterns in correlated time series. A general interface for handling external factors is further designed to enhance forecasting performance of the model. Experiments are conducted on two types of real-world datasets, viz., civil aviation passenger demand data and air quality data. And our CTS-LSTM achieves at least 9.0%, 16.5% and 21.3% lower RMSE, MAE and MAPE compared to the state-of-the-art baselines.

1. Introduction

In many scenarios, multiple time series are observed and recorded simultaneously to cooperatively reflect and monitor the comprehensive situations in a holistic system. For example, in Fig. 1(a), many air quality monitoring stations in a city timely report pollutant concentrations. Fig. 1(b) presents the passenger demand series of different air routes in the civil aviation domain. Such multiple time series are influenced and depended on each other, so we call them ***correlated time series***. Beyond real-time monitoring and reporting, people desire to know accurate forecasting information about correlated time series to make more sensible plans, e.g., air pollutant forecasts of regions can help local people take on preventive measures in advance to reduce the risk of illness; accurate predictions for passenger demands of air routes can help airlines to allocate transportation resources reasonably.

So simultaneous forecasting the future values of all correlated time series is of great significance, while it has great challenges. The key to the problem of correlated time series prediction is to deeply understand the characteristics of correlated time series and to comprehensively exploit relevant external factors. Correlated time series usually have two crucial characteristics:

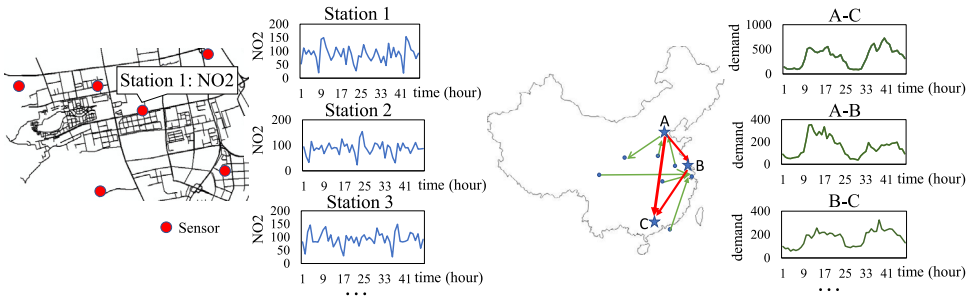
1. **Intra-sequence temporal dependencies.** It refers to the inherent regularities along the temporal dimension in a single time series. Fig. 1(b) shows the real-time passenger demand series of an air route follows a periodic pattern owing to human's daily routine, which has ticketing peaks in the daytime and troughs in the midnight. Similar phenomenon is also observed in Fig. 1(a) that fluctuates and changes of the air quality values in a monitoring station are strongly relevant with historical values in the temporal dimension.
2. **Inter-sequence spatial dependencies.** It means the mutual spatial correlations among the correlated time series.

☆ No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.knosys.2019.105239.

☆☆ This work was supported by the Fundamental Research Funds for the Central Universities, China (Grant No. 2019JBM024).

* Corresponding author at: School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China.
E-mail address: hywan@bjtu.edu.cn (H. Wan).

https://doi.org/10.1016/j.knosys.2019.105239
0950-7051/



(a) Pollutant concentrations monitored by different air quality stations

(b) Passenger demand on different air routes

Fig. 1. Examples of correlated time series.

Fig. 1(b) demonstrates the spatial dependencies between passenger demand series of different air routes due to some flight transfer demands. For example, if there is a excessive demand on the air route A to C but the tickets have been sold out, people will probably choose an airport B to transfer, which will inevitably increase the demands of the air routes A to B and B to C. Likewise, the values of an air quality monitoring station in Fig. 1(a) are also related to those of the surrounding stations.

Additionally, in the real-world scenarios, the values of our studied correlated time series are also influenced by external environment, e.g., a heavy rainstorm may cause a sudden drop in the air pollutant concentrations; a social event such as an international sports competition will lead to a sharp increase on the demand of some air routes. Hence, taking these external factors into considerations can help us to understand some sudden non-linear changes in data.

In recent years, with the breakthroughs in processing sequential data achieved by deep recurrent networks, various types of models based on deep recurrent networks are introduced to time series analysis. The most well-known model is Long Short Term Memory (LSTM) [1], which keeps the history of a sequence in its hidden units to capture the temporal patterns of sequences. LSTMs have achieved good performance on the task of single time series forecasting [2,3]. As for the correlated time series prediction, basic LSTM is usually powerless since it only models the intra-sequence temporal dependencies, but ignores the inter-sequence spatial dependencies among multiple time series that is equally vital for the prediction. The intra-sequence temporal dependencies reveal the changing patterns in a single series, meanwhile the inter-sequence spatial dependencies may give reasonable explanations for the abnormal changes in the series caused by other time series. Therefore, together explicitly modeling the intra-sequence temporal dependencies and the inter-sequence spatial dependencies play a critical role in making accurate correlated time series predictions.

To tackle the challenges mentioned above, in this paper, we propose a novel recurrent network called CTS-LSTM *(Correlated Time Series oriented LSTM)* to collectively predict the future values of all correlated time series. The main contributions of our study are three-fold:

- We propose a novel CTS-LSTM model to explicitly model two types of characteristics in correlated time series, i.e., the intra-sequence temporal correlations and the inter-sequence spatial correlations, in a memory cell;
- In real-world applications, so as to comprehensively consider external factors, we design a general interface to embed external factors into a vector and feed the learned embedding vector into CTS-LSTM to further enhance the prediction performance;
- We conduct extensive experiments on two real-world datasets to evaluate the performance of our proposed CTS-LSTM. The results demonstrate that our model achieves the best forecasting performance against all baselines.

2. Related work

Currently, there are two categories of methods solving time series prediction problem. The first category is the model-driven methods, i.e., the classical statistical methods such as auto-regressive integrated moving average (ARIMA) [4,5], Vector Auto Regression (VAR) [6] and their variants [4]. But it is challenging for these traditional prediction models to capture the complex non-linear dynamic of time series. So some researches [7] further combine technical analysis, genetic algorithms with these classical prediction methods. But for the correlated time series prediction problem studied in this paper, most of the methods mentioned above cannot capture the spatial correlations between the multiple time series. Hence, it is difficult for this kind of methods to make an accurate forecasting result.

The second category is the data-driven methods, such as the traditional machine learning methods and deep learning neural networks. Traditional machine learning methods used for time series prediction are mainly linear regression (LR) [8] and support vector regression (SVR) [9,10]. For example, [11] combined the synthetic minority oversampling technique (SMOTE) and the Adaboost support vector machine to realize dynamic financial series prediction. However, the capacities of these methods are usually not enough to effectively model the complex spatial–temporal features of the high-dimensional nonlinear multiple correlated time series data. Fortunately, due to the development of the modern technology, plenty of multiple correlated time series are recorded and accumulated in real time. A large amount of accumulated historical correlated time series provide a good opportunity for us to employ deep learning to solve the correlated time series forecasting problem. Recurrent Neural Network (RNN) [12] as a typical kind of deep neural network has achieved great success in sequence learning tasks. [1] first proposed LSTM to improve the RNN's ability [13,14] of learning long-term dependencies in sequences. Besides, [15] proposed GRU, another RNN model, for the same purpose. LSTM and GRU have achieved good results in single time series forecast [16]. However, these models can only consider temporal dependencies [17] in time series and cannot capture mutual spatial dependencies among correlated time series [18]. To capture the spatial correlations of the sequential data, some researchers represent the data into grid-based forms, and use convolution operations to capture its local spatial patterns. For example, [19] proposed a novel ConvLSTM network, which changes the multiplication operation in LSTM to the convolution operation in order to consider the spatial features at each time step. The PredRNN model proposed

Please cite this article as: H. Wan, S. Guo, K. Yin et al., CTS-LSTM: LSTM-based neural networks for correlatedtime series prediction, Knowledge-Based Systems (2019) 105239, https://doi.org/10.1016/j.knosys.2019.105239.

by [20] uses a new spatio-temporal memory flow to transfer the high-level spatial features at the current moment to the lower-level layers at the next moment. However, these models are limited by the representation of data. They can only deal with grid-based data and capture the spatial patterns of adjacent grids. Recently, [21] proposed a GeoMAN model based on the encoder–decoder framework combined with multi-level attention mechanisms to capture the spatio-temporal dependencies between multiple sequences. The experiments proved that the GeoMAN achieved the best performance on geo-sensory time series prediction. Although the dependencies among multiple time series are extracted before the first layer in GeoMAN, the spatial information is mixed after being fed into the encoder, so it is still difficult for the higher level layers to further distinguish and utilize these correlations. Besides, this model can only make the prediction for one sequence after inputting a large amount of correlated data, so the amount of computation is large in practical applications.

In summary, the existing models cannot adequately capture the temporal and spatial correlations of correlated time series at the same time. However, the two characteristics are equally crucial for forecasting tasks. Therefore, in this paper, based on the LSTM framework, a novel memory cell is designed to fully extract the spatial–temporal dependencies in correlated time series to improve prediction accuracy.

3. Preliminaries

There are N correlated time series. Among them, $x_t^c \in \mathbb{R}$ is defined as the values of the cth time series at time t, where $c \in \{1, 2, \ldots, N\}$. $\boldsymbol{x}_t = \left(x_t^1, x_t^2, \ldots, x_t^N\right) \in \mathbb{R}^N$ denotes the values of all the time series at time t.

3.1. Problem statement

Given the previous values of all correlated time series $\{\boldsymbol{x}_t | t = 1, 2, \ldots, n\}$, simultaneously predict the values of all the time series at time $n + k$, where k denotes the prediction time interval.

4. Correlated time series oriented LSTM

In this section, we first introduce the conventional LSTM architecture and analyze its deficiencies in predicting correlated time series. Then we describe the architecture of the proposed CTS-LSTM model, which can capture both the intra-sequence temporal dependencies and the inter-sequence spatial dependencies in correlated time series.

4.1. Structure of standard LSTM

Long Short Term Memory (LSTM) network [1] is a kind of Recurrent Neural Network (RNN). Different from the feed-forward neural networks, RNN has recurrent connections between hidden states with a time delay. Ideally, RNN is able to discover temporal correlations between the events that are far away from each other in sequences. However, in practice, it is hard to train RNN because of the vanishing gradient and exploding gradient problems explained in Pascanu et al. [22], so RNN is faced with difficulties in handling long-term dependencies in sequences. Actually, the most effective RNN-based model used in practice is called LSTM. Three extra gating units are introduced in LSTM to control the flow of information, which is beneficial for maintaining important information for a long time. Formally, the LSTM can be formulated as follows:

$$\boldsymbol{i}_t = \text{sigmoid}\left(\boldsymbol{W}_i \left[\boldsymbol{x}_t, \boldsymbol{h}_{t-1}\right] + \boldsymbol{b}_i\right) \tag{1}$$

$$\boldsymbol{f}_t = \text{sigmoid}\left(\boldsymbol{W}_f \left[\boldsymbol{x}_t, \boldsymbol{h}_{t-1}\right] + \boldsymbol{b}_f\right) \tag{2}$$

$$\boldsymbol{o}_t = \text{sigmoid}\left(\boldsymbol{W}_o \left[\boldsymbol{x}_t, \boldsymbol{h}_{t-1}\right] + \boldsymbol{b}_o\right) \tag{3}$$

$$\widetilde{\boldsymbol{c}}_t = \tanh\left(\boldsymbol{W}_c \left[\boldsymbol{x}_t, \boldsymbol{h}_{t-1}\right] + \boldsymbol{b}_c\right) \tag{4}$$

$$\boldsymbol{c}_t = \boldsymbol{i}_t \circ \widetilde{\boldsymbol{c}}_t + \boldsymbol{f}_t \circ \boldsymbol{c}_{t-1} \tag{5}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \circ \tanh(\boldsymbol{c}_t) \tag{6}$$

where $\boldsymbol{W}_*$ are weight matrices, $\boldsymbol{b}_*$ are bias vectors, and $* \in \{i, f, o, c\}$. At each time step t, $\boldsymbol{x}_t$ is an input vector, $\boldsymbol{c}_t$ denotes the memory state vector, and $\boldsymbol{h}_t$ is the hidden state vector calculated based on $\boldsymbol{c}_t$. The activation functions for all the three gating units (i.e., input gate $\boldsymbol{i}_t$, forget gate $\boldsymbol{f}_t$ and output gate $\boldsymbol{o}_t$) are sigmoid functions. Since the sigmoid function outputs a value between 0 and 1, it can either let no flow or complete flow of information throughout the gates. So the three types of gating units control the information entering and leaving a memory cell at each time step. The input modulation $\widetilde{\boldsymbol{c}}_t$ and output $\boldsymbol{h}_t$ usually employ Tanh as the activation functions. Because the internal state vector $\widetilde{\boldsymbol{c}}_t$ should be able to increase or decrease. The output from Tanh can be positive or negative, allowing for increases and decreases in the state. However, the sigmoid output is always non-negative so the values in the state would only increase. Besides, "$\circ$" denotes element-wise multiplication. Because of the gating architecture, LSTM is able to handle both the short and long-term temporal correlations in sequences.

4.2. Structure of CTS-LSTM

4.2.1. Overall structure

Correlated time series have their own regularities in the temporal dimension. For example, due to the daily routines of people, the real-time passenger demand series of air routes often show some patterns such as the ticketing peaks at 8:00 to 9:00 pm every evening during the rest time and the troughs in the midnight when people are sleeping; In the meteorology, the spread of pollutants is gradual, so the air pollutant concentration at the current moment must be related to the previous concentration. Meanwhile, correlated time series also have spatial dependencies among each other. In the civil aviation domain, because of the flight transfer demand, there are competitive and cooperative relations between the air routes. Hence, the changes of demand on an air route may affect those on other air routes. Likewise, in the meteorology, a strong wind may bring the high concentration of air pollutants around a meteorological station to surrounding stations. Therefore, there exists strong correlations between the situations of nearby meteorological stations. Consequently, the spatial correlations are ubiquitous in the correlated time series and are very significant for forecasting.

However, if all the correlated time series are processed separately in different LSTMs, the correlations among them will not be captured. Even though all correlated time series are processed in one LSTM, at each time step t, the information of all the N correlated time series $\boldsymbol{x}_t \in \mathbb{R}^N$ is taken as a N-dimensional vector and pushed into the first LSTM layer. Afterwards, $\boldsymbol{x}_t$ is mapped into a certain dimensional representation, that is to say, all the series' information is fused into one representation. Therefore, based on the fused representation, the respective characteristics of each time series cannot be fully explored and the mutual correlations among them cannot be well mined neither. That is to

Please cite this article as: H. Wan, S. Guo, K. Yin et al., CTS-LSTM: LSTM-based neural networks for correlatedtime series prediction, Knowledge-Based Systems (2019) 105239, https://doi.org/10.1016/j.knosys.2019.105239.

say, right after the first LSTM layer, the unique features of each time series are lost, so the higher layers can hardly distinguish the individual temporal features of each series and the correlated spatial features among all series. Consequently, it is very difficult for the output layer to make accurate forecasts for each of the time series. In a word, conventional LSTM lacks the ability of fully describing the spatial–temporal features in correlated time series, so the prediction results may be far from satisfactory.

To deal with the challenges mentioned above, we propose a novel CTS-LSTM model, which explicitly describes the intra-sequence temporal correlations and the inter-sequence spatial correlations in correlated time series. CTS-LSTM also can take external factors into account in order to comprehensively exploit the factors that may affect the values of time series in the future. The architecture of CTS-LSTM is shown in Fig. 2. It mainly consists of three parts: (1) spatio-temporal cells; (2) spatio-temporal fusion; (3) the external factor module.

4.2.2. Spatio-temporal cell

The structure of the Spatio-Temporal cell (ST-cell) is shown in Fig. 3, which maintains two channels to respectively capture the intra-sequence temporal dependency and the inter-sequence spatial dependency in correlated time series.

Eqs. (7) to (12) illustrate how the ST-cell works in the intra-sequence temporal channel, as shown by the blue line in Fig. 3. When modeling the intra-sequence temporal dependencies (i.e., the similarities in neighboring time steps, the periodicities and trends of series, etc.), different from the conventional LSTM that constructs a single embedding representation for all series, ST-cell respectively constructs a representation for each series. Here, we define the intra-sequence cell state and the intra-sequence hidden state at time t as matrixes $\boldsymbol{C}_{intra\_t}$ and $\boldsymbol{H}_{intra\_t} \in \mathbb{R}^{u\times N}$, where u is the number of dimensions of the embedding representations and N is the number of correlated time series. Like the conventional LSTM, the intra-sequence cell state of ST-cell evolves as a combination of part of the current input and part of the past state. However, unlike the conventional LSTM, the recurrent update of the intra-sequence cell states in ST-cell depends on the individual features of each sequence. Thus, we define three gate matrixes $\boldsymbol{I}_{intra\_t}$, $\boldsymbol{F}_{intra\_t}$, $\boldsymbol{O}_{intra\_t}$ to control how much information in various embedding features of each time series should be kept in the intra cell state at every time step.

The other channel in ST-cell, as shown by the green line in Fig. 3, is specialized for modeling the inter-sequence spatial dependency (i.e., the mutual influence among correlated time series). The input in this channel is $\boldsymbol{X}_{inter\_t} \in \mathbb{R}^{d\times N}$, indicating that there are N correlated series and each of them belongs to a d-dimensional space. $\boldsymbol{X}_{inter\_t}$ is firstly transformed to $\boldsymbol{X}'_{inter\_t}$ by a spatial relation matrix $\boldsymbol{S}$, as shown in Eq. (13), where $\boldsymbol{S} \in \mathbb{R}^{N\times N}$ reflects the pairwise influence between correlated series. Hence, in $\boldsymbol{X}'_{inter\_t}$, the representation of every series incorporates the information about other series. Thereby, through the first step, $\boldsymbol{X}'_{inter\_t}$ reconstructs the representation of each series by fusing all the series' information. The inter-sequence cell states, hidden states and three gates are also matrixes and computed based on $\boldsymbol{X}'_{inter\_t}$. Here, the three gates (i.e., $\boldsymbol{I}_{inter\_t}$, $\boldsymbol{F}_{inter\_t}$, $\boldsymbol{O}_{inter\_t}$) for controlling the information flow and the inter-sequence input modulation $\widetilde{\boldsymbol{C}}_{inter\_t}$ are determined by the $\boldsymbol{X}'_{inter\_t}$ and the previous hidden layer output $\boldsymbol{H}_{inter\_t-1}$ in the inter-sequence spatial channel. Similarly, $\boldsymbol{I}_{inter\_t}$, $\boldsymbol{F}_{inter\_t}$, $\boldsymbol{O}_{inter\_t}$, $\widetilde{\boldsymbol{C}}_{inter\_t} \in \mathbb{R}^{u\times N}$ are also matrixes, that is, individual embedding representation containing inter-sequence spatial information is learnt for each sequence.

Finally, as shown in Fig. 3, $\boldsymbol{H}_{intra\_t}$ and $\boldsymbol{H}_{inter\_t}$ form the final output $\boldsymbol{H}_t$. Likewise, $\boldsymbol{C}_{intra\_t}$ and $\boldsymbol{C}_{inter\_t}$ form $\boldsymbol{C}_t$.

$$\boldsymbol{I}_{intra\_t} = \text{sigmoid}\left(\boldsymbol{W}_i\left[\boldsymbol{X}_{intra\_t}, \boldsymbol{H}_{intra\_t-1}\right] + \boldsymbol{b}_i\right) \tag{7}$$

$$\boldsymbol{F}_{intra\_t} = \text{sigmoid}\left(\boldsymbol{W}_f\left[\boldsymbol{X}_{intra\_t}, \boldsymbol{H}_{intra\_t-1}\right] + \boldsymbol{b}_f\right) \tag{8}$$

$$\boldsymbol{O}_{intra\_t} = \text{sigmoid}\left(\boldsymbol{W}_o\left[\boldsymbol{X}_{intra\_t}, \boldsymbol{H}_{intra\_t-1}\right] + \boldsymbol{b}_o\right) \tag{9}$$

$$\widetilde{\boldsymbol{C}}_{intra\_t} = \text{relu}\left(\boldsymbol{W}_c\left[\boldsymbol{X}_{intra\_t}, \boldsymbol{H}_{intra\_t-1}\right] + \boldsymbol{b}_c\right) \tag{10}$$

$$\boldsymbol{C}_{intra\_t} = \boldsymbol{I}_{intra\_t} \circ \widetilde{\boldsymbol{C}}_{intra\_t} + \boldsymbol{F}_{intra\_t} \circ \boldsymbol{C}_{intra\_t-1} \tag{11}$$

$$\boldsymbol{H}_{intra\_t} = \boldsymbol{O}_{intra\_t} \circ \text{relu}(\boldsymbol{C}_{intra\_t}) \tag{12}$$

$$\boldsymbol{X}'_{inter\_t} = \boldsymbol{X}_{inter\_t} \cdot \boldsymbol{S} \tag{13}$$

$$\boldsymbol{I}_{inter\_t} = \text{sigmoid}\left(\boldsymbol{W}'_i\left[\boldsymbol{X}'_{inter\_t}, \boldsymbol{H}_{inter\_t-1}\right] + \boldsymbol{b}'_i\right) \tag{14}$$

$$\boldsymbol{F}_{inter\_t} = \text{sigmoid}\left(\boldsymbol{W}'_f\left[\boldsymbol{X}'_{inter\_t}, \boldsymbol{H}_{inter\_t-1}\right] + \boldsymbol{b}'_f\right) \tag{15}$$

$$\boldsymbol{O}_{inter\_t} = \text{sigmoid}\left(\boldsymbol{W}'_o\left[\boldsymbol{X}'_{inter\_t}, \boldsymbol{H}_{inter\_t-1}\right] + \boldsymbol{b}'_o\right) \tag{16}$$

$$\widetilde{\boldsymbol{C}}_{inter\_t} = \text{relu}\left(\boldsymbol{W}'_c\left[\boldsymbol{X}'_{inter\_t}, \boldsymbol{H}_{inter\_t-1}\right] + \boldsymbol{b}'_c\right) \tag{17}$$

$$\boldsymbol{C}_{inter\_t} = \boldsymbol{I}_{inter\_t} \circ \widetilde{\boldsymbol{C}}_{inter\_t} + \boldsymbol{F}_{inter\_t} \circ \boldsymbol{C}_{inter\_t-1} \tag{18}$$

$$\boldsymbol{H}_{inter\_t} = \boldsymbol{O}_{inter\_t} \circ \text{relu}(\boldsymbol{C}_{inter\_t}) \tag{19}$$

In the first layer, in order to feed $\boldsymbol{x}_t \in \mathbb{R}^N$ into the ST-cell whose input is a two-channel matrix, we let $\boldsymbol{X}_{intra\_t} = \boldsymbol{X}'_t$, $\boldsymbol{X}_{inter\_t} = \boldsymbol{X}'_t$, here $\boldsymbol{X}'_t \in \mathbb{R}^{1\times N}$ is constructed based on $\boldsymbol{x}_t$ by extending a new dimension. $\boldsymbol{W}_* = \left[\boldsymbol{W}^x_*, \boldsymbol{W}^h_*\right]$, where $\boldsymbol{W}^x_* \in \mathbb{R}^{u\times d}$, $\boldsymbol{W}^h_* \in \mathbb{R}^{u\times u}$ and $* \in \{i, f, o, c\}$. $\boldsymbol{W}_*{}'$ is similar to $\boldsymbol{W}_*$. $\boldsymbol{b}_*$ and $\boldsymbol{b}_*{}' \in \mathbb{R}^u$ are biases. "$\circ$" denotes element-wise multiplication.

In Eqs. (7) to (19), $\boldsymbol{W}_*{}'$, $\boldsymbol{W}_*$, $\boldsymbol{b}_*$ and $\boldsymbol{b}_*{}'$ are all learning parameters. $\boldsymbol{S}$ indicates the mutual influence among correlated series. It can be set to a fixed matrix reflecting a priori knowledge of a certain domain. For instance, if we are studying the geo-spatial correlated time series, $\boldsymbol{S}$ can be a proximity or similarity matrix between different monitoring stations, such as the inverse of the physical distances between the stations. For other applications, it may be the connectivity or other topological properties of a graph structure where different time series are regarded as the signals of different vertices on the graph. $\boldsymbol{S}$ can also be initially set to a priori matrix and then be fine-tuned according to real training data. All the input gates, output gates and forget gates still employ sigmoid as the activation function, while $\widetilde{\boldsymbol{C}}_{intra\_t}$, $\widetilde{\boldsymbol{C}}_{inter\_t}$, $\boldsymbol{H}_{intra\_t}$, and $\boldsymbol{H}_{inter\_t}$ all use the rectified linear unit (ReLU) as the activation function, i.e., $f(x) = \max(x, 0)$.

4.2.3. Spatio-temporal fusion

There is a fusion module on the top of the last ST-cell layer. As shown in Fig. 4, the role of the fusion module is to aggregate the information in the two channels:

$$\boldsymbol{X}_{fusion\_t} = \Theta_1 \circ \boldsymbol{H}_{intra\_t} + \Theta_2 \circ \boldsymbol{H}_{inter\_t} \tag{20}$$

$$\boldsymbol{x}_{seq\_t} = \text{relu}(\boldsymbol{W}_{seq} \cdot \boldsymbol{x}'_{fusion\_t} + \boldsymbol{b}_{seq}) \tag{21}$$

There are two steps in the fusion module. Firstly, a weighted sum $\boldsymbol{X}_{fusion\_t}$ is computed, where Θ_1 and Θ_2 are the learnable parameters that reflect the importance of the intra-sequence features and the inter-sequence features. Then $\boldsymbol{X}_{fusion\_t}$ is flattened to $\boldsymbol{x}'_{fusion\_t}$ and followed by a fully connected layer which can be viewed as an embedding layer for further extracting the spatio-temporal features. This module also chooses ReLU as the activation function, and its output is $\boldsymbol{x}_{seq\_t}$.

Please cite this article as: H. Wan, S. Guo, K. Yin et al., CTS-LSTM: LSTM-based neural networks for correlatedtime series prediction, Knowledge-Based Systems (2019) 105239, https://doi.org/10.1016/j.knosys.2019.105239.

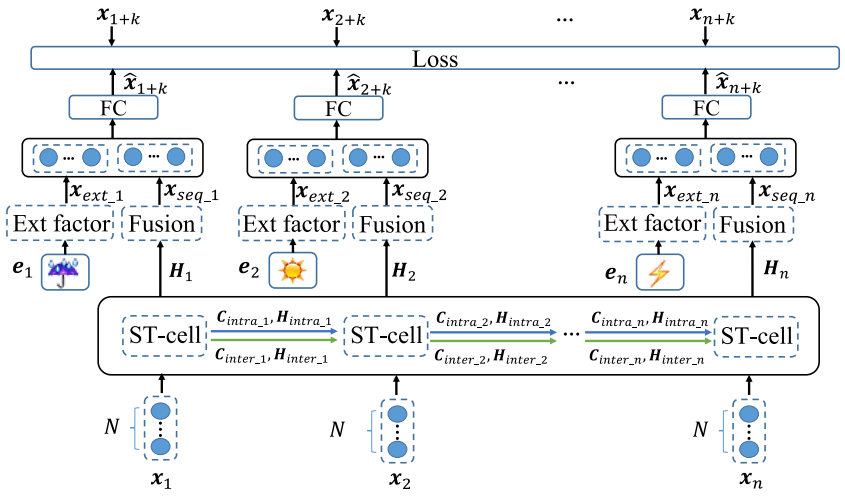


Fig. 2. The framework of CTS-LSTM.

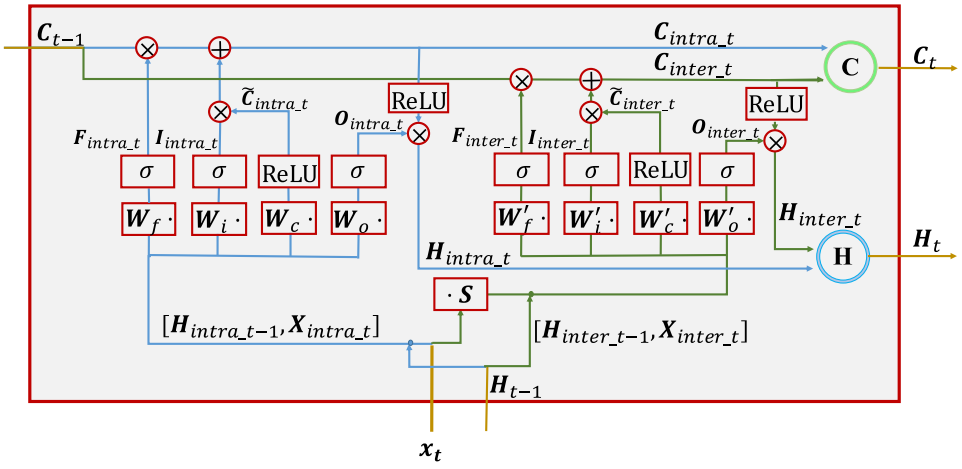


Fig. 3. The structure of ST-cell. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

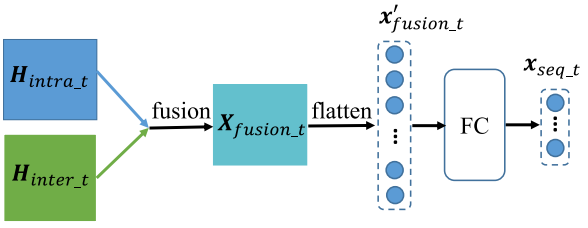


Fig. 4. Structure of the fusion module.

4.2.4. External factor module

In real-world applications, although time series have their own inherent patterns and regularities, they may be also affected by other external factors from different domains. For instance, when we focus on studying air quality, it should be noticed that the concentration of the pollutant is also affected by the meteorological information, e.g., a heavy rain or strong wind may cause a sudden decrease in the air pollution concentration; when we aim to forecast the passenger demand series in the transportation field, it can be easily understood that the demand series are also affected by social events, holidays and so on.

In order to take these external factors into consideration, we propose a universal interface. Firstly, we manually construct a vector $\boldsymbol{e}_t$ to represent all the external factors in our consideration (e.g., the day of week, holiday, weather condition, etc.). Then we stack two fully connected layers with ReLU as the activation function to learn an embedding representation $\boldsymbol{x}_{ext\_t}$ for the external information.

4.2.5. Prediction

At each time step, we concatenate $\boldsymbol{x}_{seq\_t}$ and $\boldsymbol{x}_{ext\_t}$ into $\widehat{\boldsymbol{x}}'_{t+k}$ and feed it to a layer with ReLU as the activation function to further extract the features. Then a second fully-connected layer with Tanh is stacked to ensure the prediction has $\boldsymbol{N}$ dimensions and its values are in $[-1,1]$.

4.2.6. Loss function

CTS-LSTM is trained by back propagation and Adam algorithm [23] aiming at minimizing the mean square error between the true values $\boldsymbol{x}_{t+k}$ and the predicted values $\widehat{\boldsymbol{x}}_{t+k}$. The loss function is as follow:

$$L(\theta) = \|\boldsymbol{x}_{t+k} - \widehat{\boldsymbol{x}}_{t+k}\|_2^2 \tag{22}$$

where θ represents all the learning parameters in the CTS-LSTM model.

5. Experiments

In order to validate our model, we conduct experiments on two real-world prediction tasks.

Please cite this article as: H. Wan, S. Guo, K. Yin et al., CTS-LSTM: LSTM-based neural networks for correlatedtime series prediction, Knowledge-Based Systems (2019) 105239, https://doi.org/10.1016/j.knosys.2019.105239.

Table 1
Statistics of the passenger demand dataset.

Dataset	Description
City list	BJS,SHA,CAN,SZX,CTU, XIY,NKG,TSN,WUH,CGO
#Routes	90
Query date spans	12/17/2014-5/15/2015
Departure date spans	1/17/2015-5/15/2015
Time intervals	1 h
#Training samples	23,520
#Test samples	16,464

5.1. Evaluation metrics and baselines

We use Root Mean Square Error (RMSE), Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE) to evaluate the performance, and compare our model with the following baselines:

- **Pre:** The historical value at the corresponding time on the previous day is directly used as the prediction.
- **ARIMA** [24]**:** ARIMA is a well-known model in time series prediction.
- **VAR** [25]**:** VAR is a multivariate time series prediction model, which can capture the linear interdependencies among multiple time series.
- **LSTM** [1]**:** Long short-term memory network is a deep leaning-based model, good at dealing with sequential data.
- **ConvLSTM** [19]**:** Convolutional LSTM replaces the multiplications in LSTM with the convolutions, so it is able to describe the spatiotemporal patterns.
- **GeoMAN** [21]**:** A deep learning model based on the encoder–decoder framework combined with attention mechanisms. It is specifically designed for air quality prediction and achieved the state-of-the-art performance on that problem. So it is employed as a baseline method in the second experiment.

We run each deep-learning baseline method five times and report mean as well as standard deviation. Besides, we conduct Student's t-test to validate the significance of our proposed model.

5.2. Civil aviation passenger demand prediction

5.2.1. Dataset description

The passenger demand for each air route can be reflected in real time by the number of passenger ticket query requests. So, a real air ticket query dataset provided by an online booking website is used in this paper to represent the civil aviation passengers' demand. The content of the data includes: air routes, departure date, query date, query time and query request count. An air route is determined by a departure city and a destination city. There are 10 cities considered in the experiments, so there are 90 air routes between these cities. The passenger demand series for these air routes are correlated, since they have competitive and cooperative relationships in practice.

Table 1 shows the statistics of the dataset. A passenger demand series refers to the varying passenger demands at different query times on an air route with a certain departure date. The prediction problem studied here is: given the historical values of a series over the last week, predict all the values in the next day, so the prediction time interval $k \in \{1, \ldots, 24\}$. In the experiment, the duration of each series is 18 days. The data on the first 10 days are used as the training set and the remainders are the test set.

Table 2
Hyper-parameter settings of CTS-LSTMs.

	Passenger demand prediction		Air quality prediction	
#history window size	168		24	
#neurons in ST-cell	1st layer 2nd layer	32 16	1st layer 2nd layer	32 16
#neurons in Fusion	64		64	
#neurons in External	1st layer 2nd layer	128 64	1st layer 2nd layer	32 16
#neurons in Prediction	1st layer 2nd layer	128 #routes	1st layer 2nd layer	64 #sensors

Table 3
Hyper-parameter settings of LSTMs.

	Passenger demand prediction		Air quality prediction	
#history window size	168		24	
#neurons	1st layer 2nd layer	128 128	1st layer 2nd layer 3rd layer	64 32 32

5.2.2. Settings

We use the Min–Max normalization to scale the data into the range $[-1, 1]$. In the evaluation, we rescale the predicted values back to the normal values. For external factors, we consider the passenger demands with some other related departure dates and the day attributes of the departure date. Specifically, let fd be the departure date of the underlying passenger demand time series. The demands at the corresponding prediction time window in the time series with the departure dates of $fd-1, fd-2, fd-3, fd-7\times1, fd-7\times2, fd-7\times3$ are used as the external information. Besides, we use one-hot coding to transform the day attributes of the departure date (i.e., day of week, weekday/weekend, holiday) into binary vectors. Finally, the external factors considered in this experiment are represented in a 551-dimension vector. The details about our model including the size of the historical window size and the numbers of neurons in each layer are shown in the second column of Table 2. The spatial relation matrix $\boldsymbol{S}$ in the ST-cell is initialized according to whether two air routes share the same cities. Each element on the diagonal of matrix $\boldsymbol{S}$ represents a relationship of the same air route, so we set it to 1. For any other element (i, j), if there exists at least one same city between the ith and the jth air routes, we set it to 0.5. And if there is no same city, we set it to 0. Afterwards, the spatial relation matrix $\boldsymbol{S}$ is tuned according to the training data. During the training phase, the batch size is 512. A grid search over {0.1, 0.01, 0.005, 0.001, 0.0005, 0.0001} is applied to find a proper learning rate. Considering the forecasting performance and training speed, 0.001 is selected finally.

For the LSTM and ConvLSTM, we also use grid searches to tune the models and the final hyper-parameter settings are shown in Tables 3 and 4.

5.2.3. Performance comparison

In practice, if the passenger demand can be predicted earlier, it will make more sense. Therefore, forecasts one day ahead deserve more attention when $k = 24$. We first analyze the results of all the methods when $k = 24$, as shown in Table 5. Next, we further give the detailed results of $k \in \{1, \ldots, 24\}$ in Fig. 5.

In Table 5, we find our CTS-LSTM achieves the best prediction performance. The simple forecasting strategies like Pre and the time series analysis methods like ARIMA and VAR are difficult to handle such complex and nonlinear correlated time series. The RMSE of CTS-LSTM is at least 23.9% lower than those of the

Please cite this article as: H. Wan, S. Guo, K. Yin et al., CTS-LSTM: LSTM-based neural networks for correlatedtime series prediction, Knowledge-Based Systems (2019) 105239, https://doi.org/10.1016/j.knosys.2019.105239.

Table 4
Hyper-parameter settings of ConvLSTMs.

	Passenger demand prediction	Air quality prediction
#history window size	168	24
#neurons	1st layer 64 2nd layer 32	1st layer 128 2nd layer 64 3rd layer 32 4th layer 128 5th layer 32
Kernel size	3×3	5×1

Table 5
Performance comparison on passenger demand prediction ($k = 24$).

Model	RMSE	MAE	MAPE
Pre	266.56	61.33	9090.42
ARIMA	442.15	105.11	30 441.62
VAR	460.18	103.66	28 755.86
LSTM	268.13 ± 8.44	88.71 ± 3.40	15 146.06 ± 117.22
ConvLSTM	230.59 ± 5.36	66.22 ± 2.14	15 557.83 ± 858.43
CTS-LSTM (ours)	**202.76 ± 3.96***	**53.11 ± 1.31***	**6474.05 ± 639.31***

*Means the result is significant according to Student T-test at level 0.001 compared to the best baseline ConvLSTM.

Table 6
Performance of different configurations.

Network configuration #layers, (#hidden units)	RMSE	MAE
1, (16)	205.68 ± 2.98	53.41 ± 2.01
1, (32)	207.20 ± 3.96	52.29 ± 0.73
2, (16, 16)	208.63 ± 6.4	53.90 ± 2.34
2, (16, 32)	209.36 ± 3.59	53.34 ± 0.36
2, (32, 16)	**202.76 ± 3.96**	**53.11 ± 1.31**

above methods, and the MAE as well as the MAPE are at least 13.4%, 28.8% lower. Besides, the numerical range of civil aviation passenger demand dataset is quite large, which is $[0, 32219]$. Hence, the MAPE of this experiment is relatively large when some forecasting points are far away from the actual points. Deep learning methods usually have lower prediction errors except the basic LSTM. We think the reason for this phenomenon is that the inter-sequence spatial dependencies in the correlated time series are very significant, but the LSTM fails to consider them, which inevitably leads to the loss of valuable information. By comparison, ConvLSTM which models the relationships between correlated time series get better results. However, CTS-LSTM still achieves at least 12.1%, 19.8% and 57.3% improvements in RMSE, MAE and MAPE compared with them.

Effect of Different Network Configurations. To observe how the number of ST-cell layers and the number of neurons in each layer impact the performance of CTS-LSTM, we set the number of ST-cell layers to {1, 2} and the number of neurons in each layer to {16, 32}. Table 6 shows the results of different network configurations. It can be seen that our model is insensitive to the setting of these hyper-parameters.

Effect of Spatial Relation Matrix and External Factor Module. The spatial relation matrix $\boldsymbol{S}$ reflects the spatial correlations among multiple time series. To show the effect of the learnable spatial relation matrix $\boldsymbol{S}$, we conduct comparative studies. CTS-LSTM-fixedS and CTS-LSTM-randomS are two versions of CTS-LSTM. Different from CTS-LSTM whose spatial relation matrix $\boldsymbol{S}$ is tuned after initialized with prior knowledge, the $\boldsymbol{S}$ in CTS-LSTM-randomS is initialized randomly and learned according to training data. While the $\boldsymbol{S}$ in CTS-LSTM-fixedS is fixed all the time after being initialized with prior knowledge. Results in Fig. 6(a) demonstrate that CTS-LSTM whose $\boldsymbol{S}$ is tuned after being initialized with prior knowledge performs better than the other two versions, proving that tuning $\boldsymbol{S}$ according to real data is necessary. To figure out how the external factors influence the prediction performance, we conduct comparative studies on CTS-LSTMs respectively with and without the external factor module. As shown in Fig. 6(b) CTS-LSTM with the external factor module is better than that without the external factor module, which proves that in the real-world applications, the effects of external environment cannot be ignored.

Fig. 5 shows the prediction errors of different methods changing with the prediction time interval k from 1 to 24. It can be seen that CTS-LSTM almost has obvious advantages all the time except the two shortest time intervals. Although ARIMA and VAR get better performance when $k \in \{1, 2\}$, their performance drops significantly when k is larger. As k changes from 1 to 24, the growth speeds of error rates of deep learning methods are relatively slower than those of the time series analysis methods. Specially, the performance of ConvLSTM is more stable than other baselines. This may because it can capture the inter-sequence spatial dependencies in correlated series to some extent, and obviously the spatial correlations are quite important for the long-term prediction. Our CTS-LSTM which explicitly models the intra-sequence temporal dependencies and inter-sequence spatial dependencies has a relatively flat upward trend as k becomes longer. The relative stable forecasting performance is essential in practical applications.

5.3. Air quality prediction

5.3.1. Dataset description

The air quality dataset is scratched from a public website.[1] Its statistics are listed in Table 7. The values of the sensors in a city are correlated in the spatial dimension since pollutants diffuse and aggregate gradually in the space. We select 1st Jan. 2017 to 19th Nov. 2017 as the training set, and the remainders as the test set. For each pollutant, we predict the values of all the sensors in the following day, thus the prediction time interval $k \in \{1, \ldots, 24\}$.

5.3.2. Settings

Likewise, we transform data by the Min–Max normalization. For external factors, we consider the meteorological information, including weather (i.e., sunny, cloudy, etc.), wind direction and speed, temperature, humidity and air pressure. Numerical features are normalized to $[-1, 1]$. Category information is transformed by one-hot coding. The external factors considered here are finally encoded into a 18-dimension vector. The details of our model are shown in the third column of Table 2. The spatial relationship matrix $\boldsymbol{S}$ is initialized by the reciprocal of distance between sensors. Likewise, based on the grid search the learning rate is set to 0.001 and the hyper-parameter settings are described in Tables 3, and 4. The hyper-parameter settings of GeoMAN follow [21], except that the number of neurons in encoder and decoder is to 64.

5.3.3. Performance comparison

Table 8 shows the long-term prediction results of all the approaches, when $k = 24$. Our proposed CTS-LSTM significantly outperforms other baselines in the predictions of all the three pollutants. It has at least 9.0% lower RMSE, 16.5% lower MAE and 21.3% lower MAPE than the baselines.

Effect of Different Network Configurations. A grid search on network configuration is applied and the results are shown in Table 9. Likewise, the experimental results indicate that our

[1] http://zx.bjmemc.com.cn/.

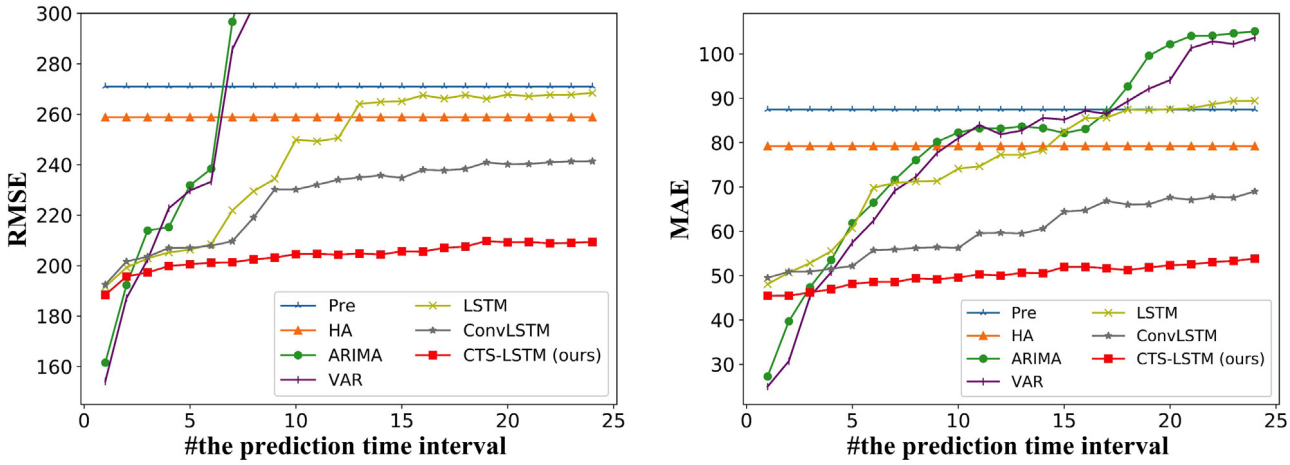


Fig. 5. Performance comparison on passenger demand prediction.

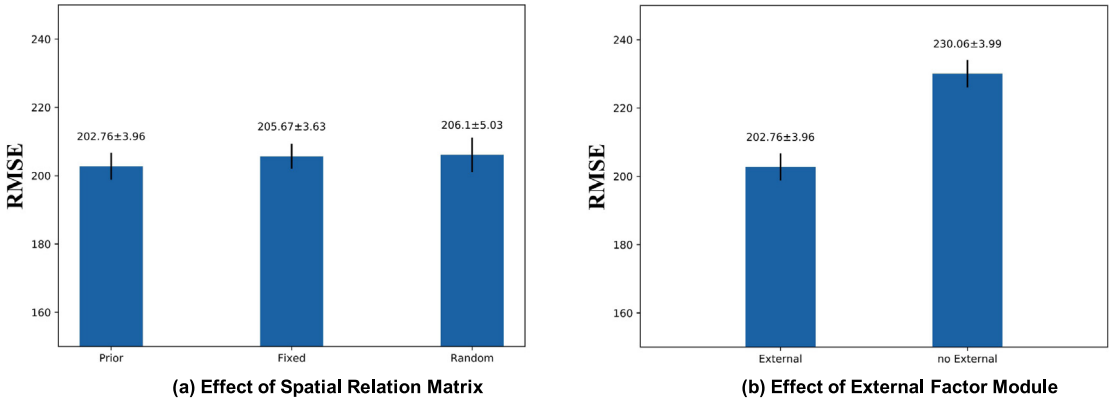


Fig. 6. Effect of spatial relation matrix and external factor module.

Table 7
Statistics of the air quality dataset.

Dataset	Beijing		London
Target series	NO2	O3	NO2
#Sensors	34	34	13
Time spans	1/1/2017-2/21/2018	1/1/2017-2/21/2018	1/1/2017-2/21/2018
Time intervals	1 h	1 h	1 h
#Instances	2400	2250	4080

Table 8
Performance comparison on air quality prediction ($k = 24$).

Dataset	Beijing						London		
Target series	NO2			O3			NO2		
Metrics	RMSE	MAE	MAPE	RMSE	MAE	MAPE	RMSE	MAE	MAPE
Pre	27.91	19.57	51.91	25.22	18.55	119.91	33.53	24.24	84.7
ARIMA	118.75	45.82	377.33	69.46	89.21	235.12	40.99	30.62	145.31
VAR	40.31	25.87	102.48	43.21	32.44	163.98	31.74	26.17	77.43
LSTM	20.46 ± 0.81	15.09 ± 0.85	40.56 ± 1.53	19.61 ± 0.82	14.82 ± 0.44	111.83 ± 3.36	15.19 ± 2.26	11.39 ± 1.00	34.35 ± 0.82
ConvLSTM	19.49 ± 1.40	14.65 ± 1.56	36.50 ± 1.92	19.23 ± 1.01	14.27 ± 0.83	95.43 ± 14.82	22.38 ± 8.88	16.65 ± 0.99	31.15 ± 1.25
GeoMAN	15.73 ± 0.28	12.18 ± 0.29	33.03 ± 1.83	16.71 ± 0.13	13.42 ± 0.03	96.86 ± 2.24	13.42 ± 0.08	10.33 ± 0.21	32.72 ± 1.03
CTS-LSTM (ours)	**14.32 ± 0.19***	**10.17 ± 0.15***	**25.82 ± 0.44***	**14.00 ± 0.22***	**10.35 ± 0.03***	**64.99 ± 1.81***	**10.52 ± 0.03***	**7.72 ± 0.03***	**24.53 ± 0.58***

*Means the result is significant according to Student T-test at level 0.001 compared to the best baseline GeoMAN.

model is insensitive to these hyper-parameter settings and always better than other baselines. Table 10 demonstrates how the history window size impacts the performance of our CTS-LSTM. It shows that the longer window size yields better forecasting results.

We further make a detailed observation for the prediction results at different prediction time intervals. As shown in Fig. 7, as the prediction interval becomes longer, the prediction errors of deep learning models increase more slowly than those of other

Please cite this article as: H. Wan, S. Guo, K. Yin et al., CTS-LSTM: LSTM-based neural networks for correlatedtime series prediction, Knowledge-Based Systems (2019) 105239, https://doi.org/10.1016/j.knosys.2019.105239.

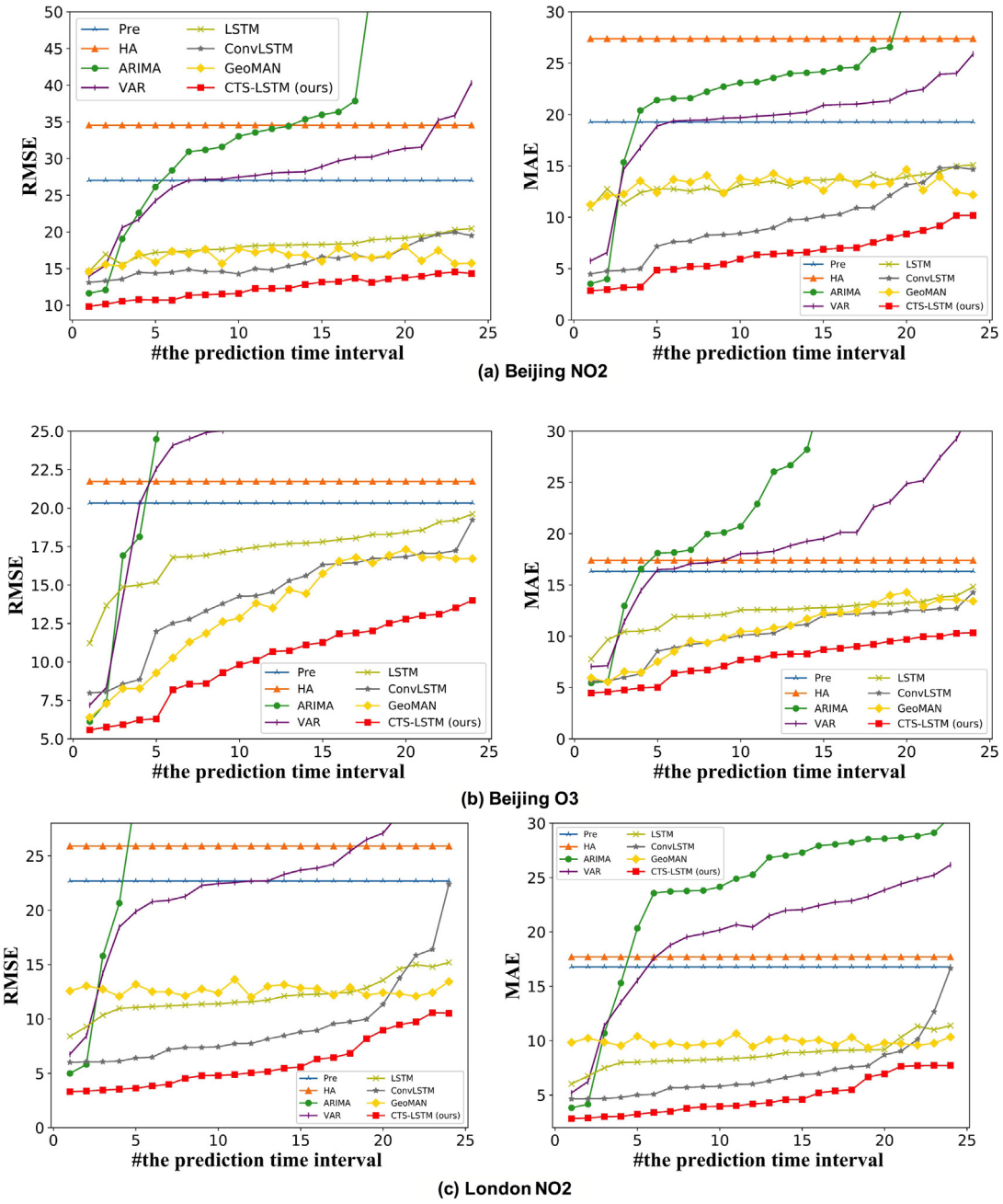


Fig. 7. Performance comparison on air quality prediction.

Table 9
Performance of different configurations.

Dataset	Beijing				London	
Target series	NO2		03		NO2	
#layers, (#hidden units)	RMSE	MAE	RMSE	MAE	RMSE	MAE
1, (16)	14.75 ± 0.29	10.55 ± 0.29	14.93 ± 0.35	11.05 ± 0.25	10.80 ± 0.09	7.92 ± 0.08
1, (32)	**14.32 ± 0.19**	**10.17 ± 0.15**	14.48 ± 0.79	10.67 ± 0.51	10.70 ± 0.05	7.83 ± 0.06
2, (16, 16)	14.69 ± 0.36	10.56 ± 0.33	14.26 ± 0.27	10.52 ± 0.21	10.64 ± 0.02	7.88 ± 0.04
2, (16, 32)	14.77 ± 0.40	10.60 ± 0.33	14.36 ± 0.77	10.58 ± 0.59	**10.52 ± 0.03**	**7.72 ± 0.03**
2, (32, 16)	14.84 ± 0.30	10.63 ± 0.21	**14.00 ± 0.22**	**10.35 ± 0.03**	10.70 ± 0.03	7.93 ± 0.03

traditional methods. Among the deep learning methods, our CTS-LSTM yields the most satisfactory predictions all the time. The results again verify the advantages of our core innovation: keeping two channels in a memory cell to capture the intra-sequence temporal dependencies and the inter-sequence spatial dependencies in the correlated time series. In addition, our model's ability of considering the external factors also improves the robustness of our model and contributes to the slow growth of error rate.

Please cite this article as: H. Wan, S. Guo, K. Yin et al., CTS-LSTM: LSTM-based neural networks for correlatedtime series prediction, Knowledge-Based Systems (2019) 105239, https://doi.org/10.1016/j.knosys.2019.105239.

Table 10
Performance of different history window sizes.

Dataset	Beijing				London	
Target series	NO2		O3		NO2	
#layers, (#hidden units)	RMSE	MAE	RMSE	MAE	RMSE	MAE
6	15.95 ± 0.04	11.83 ± 0.03	15.53 ± 0.04	11.71 ± 0.03	12.00 ± 0.05	8.81 ± 0.02
12	16.13 ± 0.10	11.83 ± 0.15	15.74 ± 0.02	11.85 ± 0.08	10.85 ± 0.08	7.94 ± 0.03
18	15.22 ± 0.41	10.99 ± 0.35	15.79 ± 0.14	11.86 ± 0.12	10.81 ± 0.16	7.90 ± 0.14
24	**14.32 ± 0.19**	**10.17 ± 0.15**	**14.00 ± 0.22**	**10.35 ± 0.03**	**10.52 ± 0.03**	**7.72 ± 0.13**

6. Conclusions

In this paper, we propose a novel deep learning based model named CTS-LSTM for correlated time series prediction. In our model, the intra-sequence temporal dependencies and the inter-sequence spatial dependencies of correlated time series are effectively captured in a modified LSTM cell and are finally aggregated in a weighted way. Considering correlated time series are also affected by external factors in practice, a general embedding module is designed, whose output is combined with the aggregated output to make the final prediction. We evaluate our model on two real-world correlated time series datasets. The experimental results show that our model achieves the best forecasting performance against seven baselines in terms of two metrics RMSE and MAE.

CRediT authorship contribution statement

Huaiyu Wan: Methodology, Writing - review & editing, Funding acquisition. **Shengnan Guo:** Conceptualization, Writing - review & editing. **Kang Yin:** Data curation, Writing - original draft. **Xiaohui Liang:** Data curation, Writing - review & editing. **Youfang Lin:** Funding acquisition, Writing - review & editing.

References

[1] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780, http://dx.doi.org/10.1162/neco.1997.9.8.1735.
[2] R. Yu, Y. Li, C. Shahabi, U. Demiryurek, Y. Liu, Deep learning: A generic approach for extreme condition traffic forecasting, in: SIAM International Conference on Data Mining, 2017, pp. 777–785, http://dx.doi.org/10.1137/1.9781611974973.87.
[3] X. Ma, Z. Tao, Y. Wang, H. Yu, Y. Wang, Long short-term memory neural network for traffic speed prediction using remote microwave sensor data, Transp. Res. C 54 (2015) 187–197, http://dx.doi.org/10.1016/j.trc.2015.03.014.
[4] X. Li, G. Pan, Z. Wu, G. Qi, S. Li, D. Zhang, et al., Prediction of urban human mobility using large-scale taxi traces and its applications, Front. Comput. Sci. 6 (1) (2012) 111–121, http://dx.doi.org/10.1007/s11704-011-1192-6.
[5] A.B. Chelani, S. Devotta, Air quality forecasting using a hybrid autoregressive and nonlinear model, Atmos. Environ. 40 (10) (2006) 1774–1780, http://dx.doi.org/10.1016/j.atmosenv.2005.11.019.
[6] L. Wang, X. Yuan, M. Ting, C. Li, Predicting summer arctic sea ice concentration intraseasonal variability using a vector autoregressive model, J. Clim. 29 (4) (2016) 1529–1543, http://dx.doi.org/10.1175/JCLI-D-15-0313.1.
[7] F. Ye, L. Zhang, D. Zhang, H. Fujita, Z. Gong, A novel forecasting method based on multi-order fuzzy time series and technical analysis, Inform. Sci. 367 (2016) 41–57, http://dx.doi.org/10.1016/j.ins.2016.05.038.
[8] S. Sousa, F. Martins, M. Alvim-Ferraz, M.C. Pereira, Multiple linear regression and artificial neural networks based on principal components to predict ozone concentrations, Environ. Model. Softw. 22 (1) (2007) 97–103, http://dx.doi.org/10.1016/j.envsoft.2005.12.002.
[9] W.Z. Lu, W.J. Wang, Potential assessment of the support vector machine method in forecasting ambient air pollutant trends, Chemosphere 59 (5) (2005) 693–701, http://dx.doi.org/10.1016/j.chemosphere.2004.10.032.
[10] U. Thissen, R. Van Brakel, A. De Weijer, W. Melssen, L. Buydens, Using support vector machines for time series prediction, Chemometr. Intell. Lab. Syst. 69 (1–2) (2003) 35–49, http://dx.doi.org/10.1016/S0169-7439(03)00111-4.
[11] J. Sun, H. Li, H. Fujita, B. Fu, W. Ai, Class-imbalanced dynamic financial distress prediction based on adaboost-svm ensemble combined with smote and time weighting, Inf. Fusion 54 (2020) 128–144, http://dx.doi.org/10.1016/j.inffus.2019.07.006.
[12] T. Mikolov, L. Karafiát M. Burget, S. Černockỳ J. Khudanpur, Recurrent neural network based language model, in: Annual Conference of the International Speech Communication Association, 2010, pp. 1045–1048.
[13] R. Carbonneau, K. Laframboise, R. Vahidov, Application of machine learning techniques for supply chain demand forecasting, European J. Oper. Res. 184 (3) (2008) 1140–1154, http://dx.doi.org/10.1016/j.ejor.2006.12.004.
[14] X. Li, L. Peng, X. Yao, S. Cui, Y. Hu, C. You, et al., Long short-term memory neural network for air pollutant concentration predictions: Method development and evaluation, Environ. Pollut. 231 (2017) 997–1004, http://dx.doi.org/10.1016/j.envpol.2017.08.114.
[15] J. Chung, C. Gulcehre, K. Cho, Y. Bengio, Empirical evaluation of gated recurrent neural networks on sequence modeling, in: NIPS 2014 Workshop on Deep Learning, 2014.
[16] Z. Zhao, W. Chen, X. Wu, P.C. Chen, J. Liu, LSTM Network: a deep learning approach for short-term traffic forecast, IET Intell. Transp. Syst. 11 (2) (2017) 68–75, http://dx.doi.org/10.1049/iet-its.2016.0208.
[17] D. Koller, N. Friedman, Probabilistic Graphical Models: Principles and Techniques, MIT press, 2009.
[18] Y. Zheng, X. Yi, M. Li, R. Li, Z. Shan, E. Chang, et al., Forecasting fine-grained air quality based on big data, in: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015, pp. 2267–2276, http://dx.doi.org/10.1145/2783258.2788573.
[19] X. Shi, Z. Chen, H. Wang, D.Y. Yeung, W.k. Wong, W.c. Woo, Convolutional LSTM network: a machine learning approach for precipitation nowcasting, in: International Conference on Neural Information Processing Systems-Volume 1, 2015, pp. 802–810.
[20] Y. Wang, M. Long, J. Wang, Z. Gao, S.Y. Philip, PredRNN: Recurrent neural networks for predictive learning using spatiotemporal lstms, in: Advances in Neural Information Processing Systems, 2017, pp. 879–888.
[21] Y. Liang, S. Ke, J. Zhang, X. Yi, Y. Zheng, GeoMAN: Multi-level attention networks for geo-sensory time series prediction, in: International Joint Conference on Artificial Intelligence, 2018, pp. 3428–3434, http://dx.doi.org/10.24963/ijcai.2018/476.
[22] R. Pascanu, T. Mikolov, Y. Bengio, On the difficulty of training recurrent neural networks, in: International Conference on Machine Learning, 2013, pp. 1310–1318.
[23] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, arXiv preprint arXiv:14126980.
[24] G.E. Box, D.A. Pierce, Distribution of residual autocorrelations in autoregressive-integrated moving average time series models, J. Amer. Stat. Assoc. 65 (332) (1970) 1509–1526, http://dx.doi.org/10.2307/2284333.
[25] E. Zivot, J. Wang, Vector autoregressive models for multivariate time series, in: Modeling Financial Time Series with S-PLUS®, 2006, pp. 385–429, http://dx.doi.org/10.1007/978-0-387-32348-0_11.

Please cite this article as: H. Wan, S. Guo, K. Yin et al., CTS-LSTM: LSTM-based neural networks for correlatedtime series prediction, Knowledge-Based Systems (2019) 105239, https://doi.org/10.1016/j.knosys.2019.105239.