Electronic Research Archive

http://www.aimspress.com/journal/ERA

ERA, 31 (1): 509–529.
DOI: 10.3934/era.2023025
Received: 24 August 2022
Revised: 08 October 2022
Accepted: 28 October 2022
Published: 14 November 2022

Research article

PM2.5 hourly concentration prediction based on graph capsule networks

Suhua Wang[1], Zhen Huang[2], Hongjie Ji[3], Huinan Zhao[1], Guoyan Zhou[4] and Xiaoxin Sun[3],*

[1] Computer Department, Changchun Humanities and Sciences College, Changchun 130117, China
[2] The University of Science and Technology of China, Department of Computer Science, Hefei 230026, China
[3] School of Information Science and Technology, Northeast Normal University, Changchun 130117, China
[4] School of Environmental Science, Northeast Normal University, Changchun 130117, China

* **Correspondence:** Email: sunxx772@nenu.edu.cn; Tel: +86-431-84536338;
Fax: +86-431-84536338.

Abstract: In this paper, we use a graph capsule network to capture the spatial dependence of air quality data and meteorological data among cities, then use an LSTM network to model the temporal dependence of air pollution levels in specific cities and finally implement PM2.5 concentration prediction. We propose a graph-capsule-LSTM model based on a graph-capsule network and an LSTM network. The model uses a graph capsule network to model the neighboring feature information of the target city and then combines the local data of the target city to form the final feature vector. The feature mapping on the time axis is then used to obtain the temporal feature sequences of the target nodes, which are fed into the LSTM network for learning and prediction. Experiments show that the method achieves better results than the latest baseline model in the PM2.5 prediction task. While demonstrating that the capsule network outperforms the convolutional network, it also shows that this capsule network is very competent for the task of PM2.5 prediction.

Keywords: PM2.5 Hourly Concentration Prediction; Graph Capsule Network; LSTM

1. Introduction

PM2.5 is one of the major types of particulate matter in air pollution, and its higher concentration means poorer air quality. Long-term exposure to high PM2.5 concentrations can cause asthma, bronchitis and other respiratory diseases, and even lung cancer [1]. High PM2.5 air pollution can also lead to reduced visibility and a series of social problems, such as airport and highway closures. Therefore, accurate prediction of PM2.5 concentrations can not only help people to make reasonable arrangements for daily travel and other activities, but also provides a reasonable basis for environmental management departments to target air pollution prevention and formulate relevant policies to improve air quality.

Currently, methods for predicting PM2.5 concentrations can be divided into two main categories: formation mechanism-based prediction methods and time-series-based prediction methods. Formation mechanism-based prediction methods simulate the physical evolution of atmospheric elements and the transport of atmospheric elements across regions, but the prediction accuracy is largely affected by the difficulty of obtaining accurate raw data and the need for sufficient background knowledge of pollution sources, reaction mechanisms, and chemical kinetics [2,3].

Compared with prediction methods based on formation mechanisms, time-series-based prediction methods have become a hot spot for domestic and foreign scholars to study. Specifically, they can be divided into two categories: 1) primitive time-series prediction methods and 2) multivariate time-series prediction methods. The primitive time-series prediction method is mainly based on the PM2.5 historical concentration series, and, for example, entails applying the optimized extreme learning machine model to predict PM2.5 [4,5]. The multivariate time-series prediction method takes into account weather elements, changes in the concentrations of other pollutants, and other factors to construct prediction models; for example, Moisan et al. [6] constructed a dynamic multivariate linear equation including weather and environmental variables; Jiang et al. [7] extracted the optimal subset of features by using a random forest model, which was used as the input data for a multilayer neural network (i.e., a multilayer perceptron (MLP)). It was shown that PM2.5 concentration values are the result of the combined effects of meteorological elements such as temperature, wind speed, precipitation, and pollutants, such as PM10, CO, and NO_2 [8,9]. However, most of the available multivariate time-series forecast only consider the local meteorological conditions and the effect of pollutants on PM2.5 concentrations. In fact, air pollutant particles are small and easily disperse in a certain space along with atmospheric motion, which is typical of geospatial data with spatial heterogeneity and spatial dependence [10]. Shi et al. [11] verified the close correlation between air pollutants at target and neighboring locations and applied a spatial attention mechanism to reveal the spatial interactions between different locations. Therefore, combining relevant factors from the local area and spatial neighboring areas to build a prediction model to predict PM2.5 in the target city is expected to further improve the prediction accuracy.

PM2.5 time-series data have the characteristics of instability and are decomposed at multiple scales, and the decomposed subseries have better scale fluctuation regularity. Zhou et al. [12] decomposed PM2.5 series data based on entire empirical modal decomposition (EEMD); they predicted each subseries using a generalized regression neural network. Weng et al. [13] used EEMD to construct a combined prediction model after decomposing PM2.5 concentration sequences. Most of the existing PM2.5 decomposition integrated prediction models only use a one-dimensional decomposition method to decompose the original PM2.5 time series, and then use the influential factors to predict each subseries separately. On this basis, this study is an attempt to combine the trends of change of PM2.5 time series and multidimensional influential factors, as well as to jointly decompose the PM2.5 and influential factors based on multivariate empirical modal decomposition to further explore the intrinsic correlation characteristics of each relevant factor and PM2.5 series, which can solve the problems of an inconsistent number of intrinsic mode functions and disorderly time s cale matching after decomposition of each series. Considering that the PM2.5 daily average point value series cannot objectively reflect the fluctuation status of pollutant concentration within a day.

Recently, some scholars began to study the problem of predicting the daily interval concentration of PM2.5. Xiong et al. [14] developed a multivariate gray model with center and radius sequences to predict PM2.5 daily degree interval number sequences, that is, to predict the highest and the lowest values. Wang et al. [15] used an MLP to fit the upper and lower bounds of PM2.5 concentration values for each day to obtain the corresponding interval prediction results. Although the PM2.5 daily interval data contain the daily concentration maximum and minimum values, the important information of daily average values is also lost.

In recent years, graph neural networks have started to be applied in air quality prediction. Xu et al. [16] used an encoder-decoder architecture, modeling complex air quality influencing factors such as weather and land use, and proposed a hierarchical graph neural network method called HighAir for air quality prediction. Chen et al. [17] proposed a prediction model based on a convolutional recurrent neural network (RNN) by using information from an air quality monitoring system in Taiwan. Yan et al. [18] constructed a multi-time, a multi-site prediction model for Beijing air quality by using a deep learning network model based on spatiotemporal clustering; they compared it with a back propagation neural network. Dai et al. [19] developed an auto-coder-circulation network for indoor/outdoor PM2.5 concentrations and environmental parameters in apartments to predict future indoor PM2.5 concentrations. Seng et al. [20] constructed an integrated prediction model based on long short-term memory (LSTM) with multiple outputs and supervised learning of multiple indicators by using a combination of particulate matter concentration data, meteorological data, and airborne gaseous pollutant data from a single monitoring station and adjacent monitoring stations for the same period.

2. Preliminaries

2.1. Problem definition

For N different cities, the set $C = \{c_1, c_2, \ldots\ldots, c_N\}$ represents the set of monitoring data for all cities; c_m represents the monitoring data time series for the city numbered m, specifically denoted as $c_m = \{D_1, D_2, \ldots\ldots, D_T\}, (m = 1,2, \ldots\ldots, N)$, with a total of T time steps. The observations of a city at each time step D_t are a d -dimensional vector that contains air quality data and meteorological data after normalization. The problem studied in this paper is defined as follows: at a specific moment t, for the target city m, the corresponding values of c_m at M future moments $(t+1, t+2, \ldots\ldots, t+M)$ are predicted by combining the historical data of the first-order and higher-order neighbors of m on the spatial graph structure in C over the past L time steps $(t, t-1, \ldots\ldots, t-L)$. Therefore, the input of the model is the historical data for all cities over the past L time steps, including air quality data and meteorological data after preprocessing. The output is the predicted hourly PM2.5 concentrations for a specific target city.

In the specific implementation of the experiment, we use the air quality data and meteorological data of the past 48 hours to predict the PM2.5 concentration values of the target cities for the next 6 hours. Due to a large number of cities in the dataset, it is impossible to predict all of the cities as target nodes all over again in each experiment. Therefore, during the experiments, if not specifically stated, for each experiment, 10 cities were picked out and sequentially used as target nodes for the training and prediction of the model; finally, the root means square error (RMSE) and mean absolute error (MAE) values at each future time step were averaged separately as a basis for performance compared with the comparison method. Unless ablation experiments were done for some hyperparameters, it will be specified on which nodes the experiments were conducted.

2.2. Study area and data monitoring

According to the dataset used for the study in this paper (source: China General Environmental Monitoring Station with China Air Quality and Meteorological Historical Data website, https://quotsoft.net/air/), after removing some cities with residual or sparse data, our study area included 189 Chinese cities, and the location distribution of the study area is shown in Figure 1.

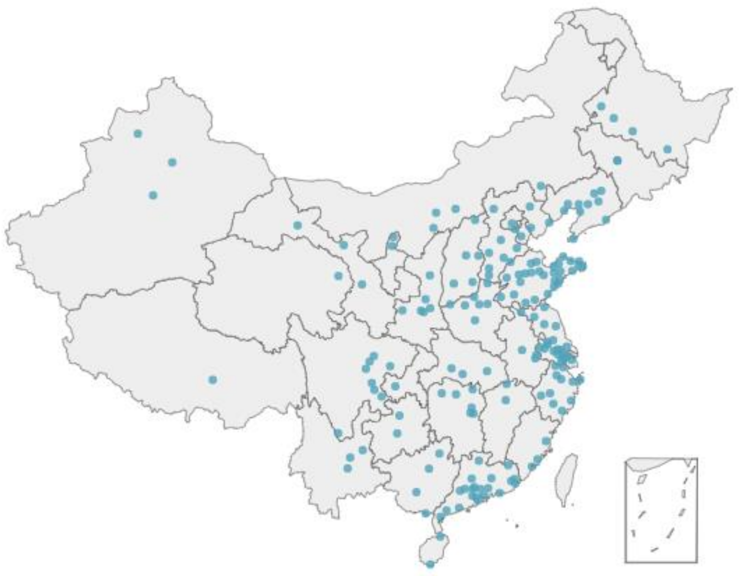

Figure 1. Distribution of the 189 monitored cities.

The datasets we used were obtained from the China General Environmental Monitoring Station with the China Air Quality and Meteorological Historical Data website for the period from May 13, 2014 to January 23, 2021. The air quality data include the daily concentration data of multiple pollutants, such as AQI AQI, PM2.5, PM10, SO_2, NO_2, O_3, CO, etc. In addition, combined with historical Chinese meteorological data, we obtained daily meteorological data, including air temperature, dew point temperature, sea level pressure, wind direction, wind speed, and liquid sedimentation depth dimensions for the same time period. After preprocessing and screening, the following 12 input features were used for all experiments: $PM_{2.5}$, PM_{10}, SO_2, NO_2, CO, O_3, air temperature, dew point temperature, sea level pressure, wind direction, wind speed and total sky condition coverage code.

In order to train and test the experimental model separately, we used the first 80% of the sample sequences within the above time range (May 13, 2014 to January 23, 2021) as the training sample set for model training and learning, and the subsequent 20% as the test sample set for testing the performance of the trained model and evaluating the generalization ability of the model. Specifically, the dataset was divided in the way shown in Table 1.

Table 1. Division of the dataset.

Data Segmentation	Time Frame	Sample Size	Calculation Instructions
Training Set	May 13, 2014 to September 23, 2019	47,001	24 hours (sample observations)/day * 2448 days * 80% = 47,001
Test Set	September 24, 2019 to January 23, 2021	11,750	24 hours (sample observations)/day * 2448 days * 20% = 11,750

2.3. Data preprocessing

2.3.1. Data normalization

Deep learning algorithms, while powerful, have their own specifications and requirements for use. Although, in theory, it is possible to feed raw data directly into the input layer for end-to-end computation, in practice, the result of feeding raw data directly into a neural network is often not satisfactory. The possible reasons are as follows:

1) If the values of data points are too large, it will cause violent oscillations in the computation and transmission in the network, which will eventually lead to violent changes in the weights and the network becoming very unstable.

2) If the data fed to different neurons in the input layer is too different in order of magnitude, it can lead to small input units being almost ignored and the input data becoming invalid. This may defy common sense in application problems.

As an example, the air quality monitoring data of Beijing at 0000 hours on May 13, 2014 are shown in Table 2.

Table 2. Sample air quality data.

AQI	PM2.5	PM2.5_24h	PM10	PM10_24h	SO2	SO2_24h	NO2	NO2_24h	O3	O3_24h	O3_8h	O3_8h_24h	CO	CO_24h
81	49	35	112	73	18	8	56	45	71	145	105	128	0.71	0.68

The meteorological observations are shown in Table 3.

Table 3. Sample meteorological data.

Air temperature	Dew point temperature	Sea level pressure	The direction of wind blowing	Wind speed	Total sky condition coverage code	Liquid settling depth, 1 hour	Liquid settling depth, 6 hours
3	-195	10384	340	30	-9999	-9999	-9999

It shows that both air quality historical data and meteorological historical data are very different in order of magnitude: there are tens of thousands, thousands, hundreds, tens, single digits, decimals and extremely large positive and negative numbers. Such raw values, when fed directly into the neural network, inevitably lead to violent oscillations in the data stream during training and uncontrollable parameter adjustments during backpropagation.

In view of the above reasons, we chose to use a min-max normalization technique to normalize the input data, as shown in Eq (1):

$$Z_i = \frac{x_i - min(x)}{max(x) - min(x)} \tag{1}$$

where Z_i denotes the i-th normative value of the indicator x, x_i denotes the i-th observation (raw value) of x, $min(x)$ denotes the minimum value of the indicator x in the dataset and $max(x)$ denotes the maximum value of the indicator x in the dataset.

After normalization, all indicator values in the original dataset are mapped from the original interval $[min(x), max(x)]$ for that indicator to the interval $[0,1]$. That is, all indicators, no matter how different they were in order of magnitude, are now normalized to the same order of magnitude.

2.3.2. Creation of neighborhood feature matrix

To formally describe the graph structure of the raw data, or rather, to obtain the actual input data structure of the model, we generate three large matrices from the raw data: the adjacency matrix A, the degree matrix D and the feature matrix F_t (t for moments). These three matrices were not provided by the original data, so we constructed them from the data during the preprocessing phase.

The 189 cities involved in the experimental dataset can be obtained as adjacencies within the threshold range by setting different distance thresholds based on their geographical distance from each other. For all pairs of neighboring cities (e.g., [Jiaozhou, Jiaonan], [Handan, Xingtai], etc.), they are connected with an undirected edge on the map, resulting in the following city distribution map showing the neighboring relationships.

Each of the cities represented by a dot in Figure 2 can be called a node or an entity in the topology. Now, there are 189 cities in the dataset, so there will be 189 nodes in the graph. The spatial relationships that exist between different locations can be represented as an undirected graph $G = (V, E)$, where V represents the set of all nodes and E represents all city adjacencies that are within a threshold on spatial distance, represented on the topology as an undirected edge connecting adjacent city nodes. $A \in \boldsymbol{R}^{N \times N}$ is the adjacency matrix based on spatial distance. A is a square matrix that represents whether the 189 cities have adjacency relations with each other. A is also a symmetric square matrix. For example, if $A[3,5] = 1$ indicates that City 3 and City 5 are adjacent cities, there must also be $A[5,3] = 1$; if $A[5,6] = 0$, there must also be $A[6,5] = 0$, indicating that City 5 and City 6 are separated by more than a distance threshold and have no adjacency.

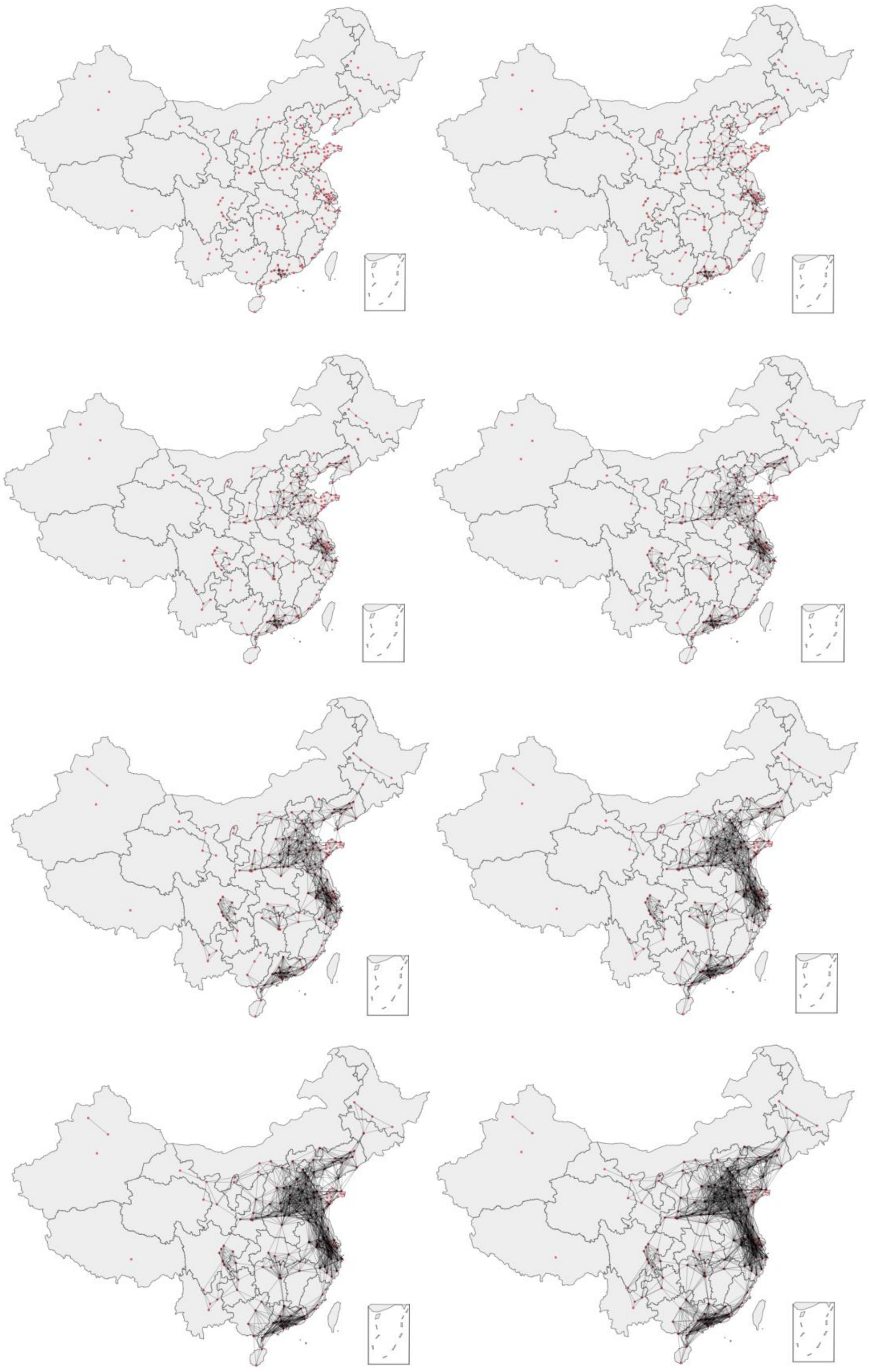

Figure 2. City distribution and adjacency map (distance thresholds of 100, 150, 200, 250, 300, 350, 400, and 450 km for ablation experiments and comparative studies).

To fit the computation of the graph capsule network, for each time step of the target city on the timeline, a neighborhood feature matrix is created for it, i.e., F_t, as shown in Figure 3.

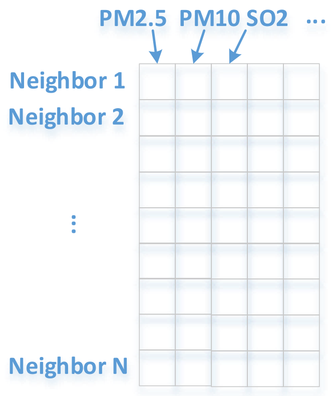

Figure 3. Neighborhood feature matrix F_t at time step t.

3. Proposed method

3.1. Modeling spatial dependencies: Using graph capsule networks

In this section, we propose a graph capsule network-based PM2.5 concentration prediction (graph-capsule-LSTM) framework in order to effectively extract the spatial relationships between urban air quality and meteorological conditions. Next, starting from the first stage, i.e., spatial dependency modeling, we first describe how the graph capsule network fuses the feature information of the target city from the neighboring feature matrix.

The graph capsule network was proposed mainly to address the inherent drawbacks of solving convolutional networks, i.e., the inability of convolutional networks to recognize changes in state, such as displacement or rotation, that occur in a signal (whether it is an image or text); in fact, these changes have a significant impact on PM2.5. Capsule networks solve these problems. Unlike traditional neural networks, the smallest unit of a capsule network is a capsule, which is a set of neurons, or a vector neuron. They will learn to detect a specific target in a given region and output a vector. The vector contains more semantic information compared to individual neurons, and, in addition to identifying features in a certain dimension, it can record changes in air quality data or meteorological data in terms of location and incorporate the effect of this change into the PM2.5 prediction calculation.

Figure 4 is the graph capsule network structure used to learn the neighbor feature matrix. Note that this is modeling spatial neighbor information for a target city at one time step. First, the node neighbor feature matrix is linearly mapped to an embedding space by multiplying a weight matrix, as follows:

$$A^{'} = A \times W_0 \tag{2}$$

$A^{'}$ is called the neighbor embedding matrix of the target node.

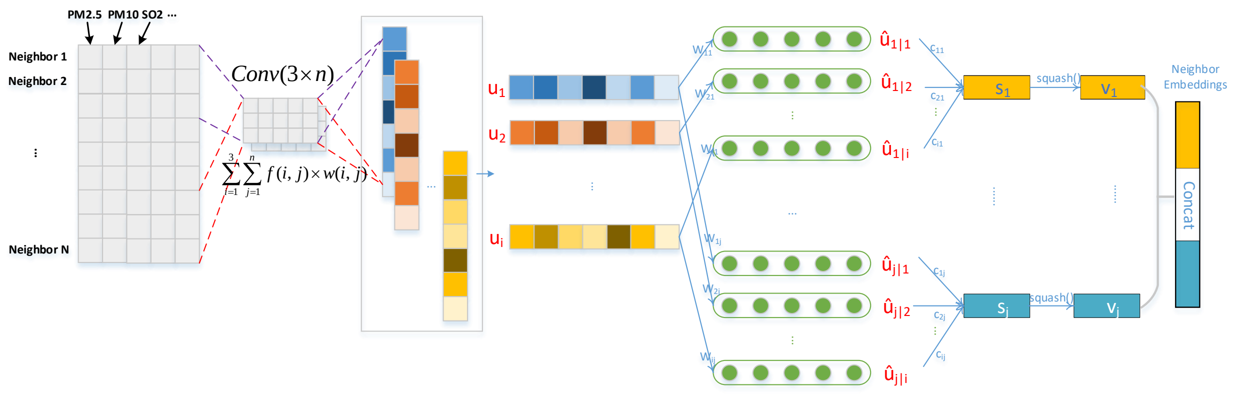

Figure 4. Structure of the graph capsule network for learning the neighbor feature matrix of target nodes.

Then, the neighbor embedding matrix A' is converted into a primary capsule layer by a convolution operation, which is the first layer of the graph capsule network. The convolution is performed as follows: A' is scanned using k $n \times 3$ convolutional kernels. The reason for choosing the convolutional kernel $n \times 3$ instead of 3×3, which is generally used in image processing, is that each row in the neighbor embedding matrix is an "atomic" representation of a neighbor node, i.e., an indivisible whole. In this way, the convolutional kernel scans the entire embedding matrix and generates a vector. This vector has two functions: 1. Local feature detection of the neighbor embedding information, which extracts the underlying features that the convolutional neural network (CNN) is good at extracting but the capsule network is not (the capsule network is used to represent "instances" of an object, so it is more suitable for representing high-level features). 2. Each convolutional kernel operation generates a vector; these vectors constitute the input layer of the capsule network, i.e., the main capsules. Each master capsule is represented as $u_1, u_2, \ldots, u_k$, as shown in Figure 4.

Next, each capsule u_i is multiplied by a series of weight matrices W_{ij} to obtain a series of vectors $\hat{u}_{j|i}$. These matrices and vectors form the second layer of the capsule network: the affine transformation capsules.

Then, after reshaping, aggregation, dynamic routing and nonlinear squeezing functions, the final output capsule is obtained.

The role of the dynamic routing mechanism is to ensure that the output of the capsule is sent to the appropriate parent node in the upper layer. Initially, the output is routed to all possible parents but scaled down by a coupling factor that sums to one. For each possible parent, the capsule calculates a "prediction vector" by multiplying its output by a weight matrix. If this prediction vector has a large scalar product with the output of the possible parents, there is top-down feedback that increases the coupling coefficient of that parent and decreases the coupling coefficient of the other parents. This increases the contribution of the capsule to that parent, which further increases the scalar product of the capsule prediction with the parent's output. This type of "routing-by-agreement" is more efficient than the original form of routing implemented by maximum pooling in traditional convolutional networks. In general, dynamic routing can be thought of as a parallel attention mechanism that allows each capsule in the upper layer to focus on some active capsules in the lower layer and ignore others. Dynamic routing is also similar to a voting mechanism, which will encapsulate the learned spatial location information together with features into a capsule and pass it to higher layers. The importance of the dynamic routing process is also supported by biologically sound models in the visual cortex.

The complete process of the capsule network is as follows:

$$u_i = Conv2D(A') \quad \text{(generate main capsule)} \tag{3}$$

$$\hat{u}_{j|i} = W_{ij} u_i \quad \text{(affine transformation)} \tag{4}$$

$$s_j = \sum_i c_{ij} \hat{u}_{j|i} \quad \text{(reshaping, aggregation, dynamic routing)} \tag{5}$$

$$v_j = squash(s_j) \quad \text{(squeeze)} \tag{6}$$

where the squeeze function $squash()$ is a nonlinear function defined as follows:

$$squash(s) = \frac{\|s\|^2}{1+\|s\|^2} \frac{s}{\|s\|} \tag{7}$$

c_{ij} is the coupling coefficient by the routing process, which is calculated as shown in Algorithm 1.

```
Algorithm 1: Routing process
Input: Primary capsule $u_i$
Output: Coupling coefficients $c$
1    Affine transformation: $\hat{u}_{j|i} = W_{ij} u_i$ ;
2    for all capsules $i$ in the first capsule layer do
3                                 $b_{ij} \leftarrow 0$
4      end for;
5    for iteration=1, 2, ..., n do
6    $c_{ij} \leftarrow \text{softmax}(b_{ij})$;
7    $s_j = \sum_i c_{ij} \hat{u}_{j|i}$;
8    $v_j = squash(s_j)$;
9                             $b_{ij} \leftarrow b_{ij} + \hat{u}_{j|i} \cdot v_j$

11   end for
12    Return $c_{ij}$ for routing operations.
```

Finally, the output vectors of the capsule network v ($v_1, v_2, ...v_j$ ...) are connected and fed into a fully connected network to generate an *m*-dimensional vector, i.e., an embedding vector of spatial relations of the target nodes:

$$F = Relu(W * \text{Concatenation}(v) + b) \tag{8}$$

3.2. Time series relationship modeling: Using LSTM RNNs

The work in the first stage above is to abstract the target node neighbor information into an embedding vector through the graph capsule network, but what is generated in Figure 4 is only the neighbor features of the target node at one time step, and this neighbor feature has not yet combined with the local features of the target city at the current time step. Therefore, two more things need to be done: 1. Combine the neighbor features and local features to form the target features at the current time step. 2. Repeat the operation in Step 1 for each time step to form a sequence of temporal features of the target city and feed it into the LSTM network to predict the PM2.5 values of the target city for the next *N* time steps. This process is summarized in Figure 5.

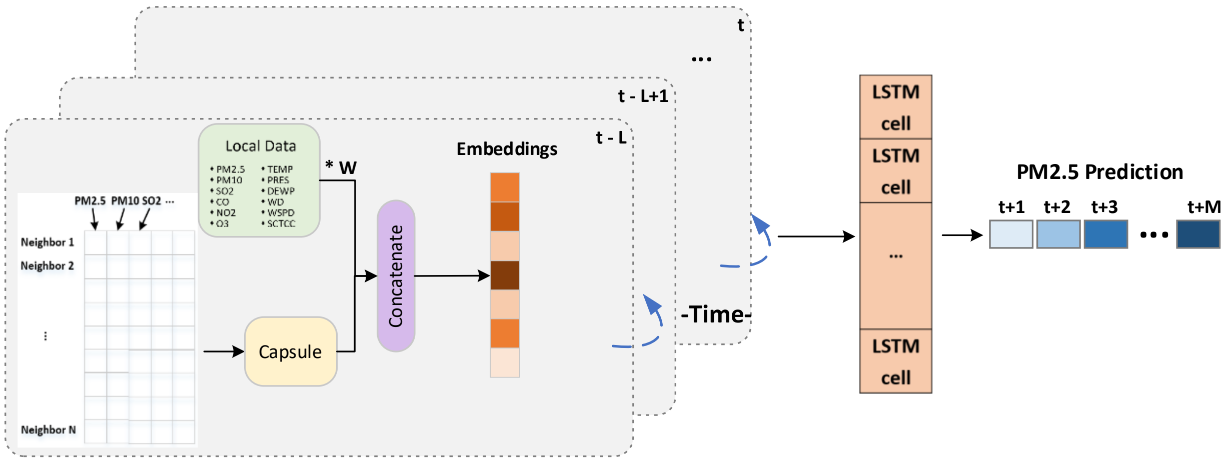

Figure 5. Graph-capsule-LSTM structure diagram, where the structure of the capsule is shown in Figure 4.

As in Figure 5, the input to the model has two components: local feature data and a neighbor feature matrix. The local feature data include the 12 original features of the target city (which have been normalized), which are multiplied by a weight matrix and mapped to a target embedding vector. The neighbor feature matrix is a matrix of all neighboring nodes with their feature vectors superimposed. For example, if the number of neighbors set is 10, then this matrix is a 10 × 12 matrix. Each row represents a feature of a particular neighbor of the target city and consists of pollutant and meteorological data ($PM_{2.5}$, PM_{10}, SO_2, CO, NO_2, O_3, temperature, barometric pressure, dew point, wind direction, wind speed and total sky condition coverage code). According to Eq (2), the kernel slides eight steps ($(10 - 3)/1 + 1 = 8$) in the input matrix with a convolutional kernel (or feature detector) size of 3 and a step size of 1. Experimentally, with a number of convolutional kernels of 60, the convolutional layer produces an 8 × 60 matrix. Each column vector is a main capsule, which is the first layer of the capsule network that follows, and, after a series of calculations from Eqs (4) to (6), the features extracted by the capsule network are obtained, which are the neighboring features of the target node. Then, the target feature vector and the neighbor feature vector are stitched together as a final target vector on a time step F.

Next, for the target node, a target vector is generated at each time step to form a sequence of temporal features of the target node, which is fed into the LSTM network for training and prediction.

$$[PM2.5_{(t+1)}, \ldots, PM2.5_{(t+6)}] = LSTM([F_{t-47}, F_{t-46}, \ldots\ldots, F_t]; G(V,E)) \tag{9}$$

Finally, the output of the LSTM is fed to the fully connected layer and outputs six cells representing the predicted PM2.5 concentrations for the next 6 hours in the target city.

3.3. Optimization methods

For such a regression task, in order to learn the parameters of the model, we used the minimized mean square error function to optimize the model and added the canonical term, so the loss function used is defined as

$$LOSS = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n} + \lambda \|W\|^2 \tag{10}$$

where y_i is the real PM2.5 concentration value and $\hat{y}_i$ is the concentration value predicted by the model. n represents the volume of the training or test set. $\lambda\|W\|^2$ is the L2 regular term, which aims to reduce the complexity of the model and improve the generalization ability of the model. And, W represents the trainable parameters of the model. The Adam optimizer was used in the experiments to minimize the loss function of Eq (10). The Adam optimizer algorithm has the advantages of the AdaGrad and RMSProp algorithms. Adam not only calculates the adaptive parameter learning rate based on the first-order moment mean, like the RMSProp algorithm, but it also takes full advantage of the second-order moment mean of gradients.

4. Experiments

4.1. Evaluation methodology

In order to enable comparison with the previous methods, the evaluation metrics commonly used in regression tasks, i.e., the MAE and RMSE, were used to measure the predictive effectiveness of different models:

$$MAE = \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{n} \tag{11}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}} \tag{12}$$

where y_i is the real $PM_{2.5}$ concentration value and $\hat{y}_i$ is the concentration value predicted by the model. n represents the number of samples in the training set or test set.

4.2. Comparative experiments and parameter settings

4.2.1. Comparison experiments

The proposed method is compared with other methods on the same dataset to prove the effectiveness of the present model. The comparison methods can be divided into four groups, as follows.

Group 1: CNN-RNN, CNN-GRU, CNN-LSTM.

All models in Group 1 only use local data from the target city. The model uses data from the last 48 hours to predict PM2.5 levels for the next 6 hours. Each batch of inputs was fed into the CNN network, which acts as a feature extractor before entering the RNN, gated recurrent unit (GRU) or LSTM network.

Group 2: GCN-GRU.

It is a mixed model. The data of all cities in the dataset were simultaneously entered into the graph convolutional network (GCN) for aggregation, updating and cycling, and the embeddings of all cities were obtained at the same time. In the next stage, the embeddings of the target city in each embedding matrix in the past few hours are successively obtained in the time step for the target city, and a time series is formed and sent to the GRU RNN for training.

Group 3: GAT-LSTM.

It is a mixed model. The data of all the cities in the dataset were simultaneously entered into the graph attention network (GAT) for aggregation, updating and cycling, and the embeddings of all cities were obtained at the same time. In the next stage, the embeddings of the target city in each embedding matrix in the past few hours are successively obtained in the time step for the target city, so as to form a time series and send it to the LSTM RNN for training.

Group 4: Graph-capsule-LSTM.

This is our proposed method, which is a new hybrid model. For the target node, the following operations are performed at each time step: The matrix formed by the air quality and meteorological data of the neighboring nodes is fed into the capsule network. Next, the neighboring feature vectors of the target node are output. Then, the local data of the target node points are combined to generate the final target features. Finally, the target feature sequences at each time step are fed into the LSTM RNN for training and prediction.

As for the networks used in the second and prediction phase, such as the RNN, LSTM network and GRU, the models for all groups consist of a hidden layer for fairness. In the output layer, six neurons are output through the fully connected network to produce the final prediction, representing the predicted PM2.5 concentration for the next 6 hours.

4.2.2. Experimental parameter settings

We implemented the graph-capsule-LSTM model by using the deep learning modeling environment Keras as the development language and TensorFlow as the backend API. Table 4 lists the experimental hyperparameter settings. For the selection of the hidden layer activation function, ReLU was used as the activation function. The advantage of the ReLU is its biological rationality. It is unilateral and more consistent with the characteristics of biological neurons than the sigmoid and tanh functions. ReLU is easier to optimize. Uniformly distributed random initialization of the model parameters was used to ensure that all neurons were treated equally at the beginning of training. For each method, no regularization training was first performed, and if there was overfitting (i.e., training losses kept decreasing while validation losses started increasing), the L2 regularization coefficients were then adjusted in the range of $[10^{-6}, 10^{-5}, \ldots\ldots, 1]$, with the aim being to obtain the best effect of eliminating overfitting during the experiment. The test batch sizes were [64, 128, 256, 512], and the learning rates were [0.1, 0.01, 0.001, 0.0001]. For time-series modeling and final result prediction, the classical LSTM network was used. For this model, capturing spatial relationships using the graph-capsule network is the first stage of the whole algorithm, which is the most complex part of the model when handling big data. For the specific structure of the graph-capsule-LSTM, the following setup was used:

(1) For spatial dependencies, only the first-order (direct) neighbors of the target city node are considered, and no higher-order neighbors are considered. Without specifying, the capsule network uses one layer of convolution and three times of routing. Various other models of the convolutional family use three hidden layers for feature extraction.

(2) In order to capture the cross-influence of air quality and meteorological conditions between spatially adjacent regions, the adjacency between cities is judged according to certain distance thresholds when modeling the map structure. The adjacency distance thresholds were set to 100, 150, 200, 250, 300, 350, 400 and 450 km, respectively. In the analysis of the experimental results, the ablation comparison study was specifically aimed at the effect of thresholds, and the model predicts the RMSE and MAE values under different threshold conditions.

If not otherwise specified, the default settings for this experiment regarding the network structure are shown in Table 4.

Table 4. Network structure settings of graph-capsule-LSTM in the experiment.

Parameters	Default value
Capsule network routing times	3
Capsule network output feature size	150
Determining distance thresholds for neighbors	200 km
LSTM	Number of input cells = 150, activation function = ReLU
Full connection layer	Number of output cells = 6, indicating that PM2.5 values are to be predicted for the next 6 hours

4.3. Experimental results and analysis

4.3.1. Comparison with comparative methods

In this section, the performance of the graph-capsule-LSTM model and other baseline methods are compared. For the embedding-based methods, since the dimension size of the embedding determines the modeling capability, the embedding size was set to 200 for all methods for a fair comparison. Tables 5 and 6 show a comparison of the results of the RMSE and MAE values for the graph-capsule-LSTM model and baseline methods on the same dataset. From the experimental results, the following was found:

(1) Overall, our proposed graph-capsule-LSTM model has the best performance and the smallest prediction error. As can be seen in Table 5, on the RMSE metric, our model reduced the error by 5.18%, 0.57%, 3.43%, 6.28%, 1.93% and 0.29%, respectively, in the future 6 hours, relative to the minimum error in the baseline models, with an average reduction of 2.95%.

(2) As can be seen in Table 6, in the case of the MAE indicator, our model reduced the error by 4.59%, 0.85%, 2.04%, 8.54%, 1.10% and 0.30%, respectively, in the future 6 hours, relative to the minimum error in the baseline models, with an average reduction of 2.90%.

(3) It is worth noting that the graph capsule network uses only one layer of convolution and three times of dynamic routing, and its performance already exceeds that of other methods. In other words, starting from the main capsule layer, the present model requires only three dynamic routes to generate a better capsule feature representation than the previous methods, while the other methods perform a three-layer aggregation process in the feature extraction process. Thus, it can be demonstrated that the capsule network is more suitable for the job of feature extraction than various convolutional networks.

(4) The advantages of our method in the task of predicting PM2.5 can also be proved from Table 7. Table 7 shows the PM2.5 values calculated by all methods and a comparison with the real values. Our chosen time and place are as follows: time: 1:00 to 6:00 AM, December 30, 2014; location: Tianjin, China. In Table 7, |**P-T**| represents |Predicted PM2.5 - True PM2.5|. The smaller the value of |**P-T**|, the closer is Predicted PM2.5 to True PM2.5. As can be seen in Table 7, the prediction accuracy of our model exceeded all baseline methods, with the error between the predicted and the true values in the future 1 to 6 hours being 10, 15, 17, 20, 25 and 29, respectively, which is much lower than the prediction error of other methods.

Table 5. Comparison of RMSE of graph-capsule-LSTM (G-C-LSTM) and baseline methods for future 6-hour PM2.5 prediction. Bold indicates the best results. An underline means second best.

Number	Model	Metric	+1 h	+2 h	+3 h	+4 h	+5 h	+6 h
1	CNN-RNN	RMSE	31.90	37.91	40.19	42.30	45.13	46.20
2	CNN-GRU	RMSE	28.91	33.37	35.91	37.01	40.26	41.31
3	CNN-LSTM	RMSE	29.51	34.22	36.19	37.91	40.11	42.49
4	GCN-GRU	RMSE	25.33	31.05	33.91	34.58	36.18	36.79
5	GAT-LSTM	RMSE	<u>22.76</u>	<u>26.51</u>	<u>31.24</u>	<u>34.06</u>	<u>34.22</u>	<u>34.38</u>
6	G-C-LSTM	RMSE	**21.58**	**26.36**	**30.17**	**31.92**	**33.56**	**34.28**
		Relative Improve	**5.18%**	**0.57%**	**3.43%**	**6.28%**	**1.93%**	**0.29%**
		Average improve			**2.95%**			

Table 6. Comparison of MAE of graph-capsule-LSTM (G-C-LSTM) and baseline methods for future 6-hour PM2.5 prediction. Bold indicates the best results. An underline means second best.

Number	Model	Metric	+1 h	+2 h	+3 h	+4 h	+5 h	+6 h
1	CNN-RNN	MAE	20.23	23.94	26.01	28.30	30.22	28.99
2	CNN-GRU	MAE	18.07	22.39	23.08	25.13	27.01	27.71
3	CNN-LSTM	MAE	18.97	22.83	24.31	26.47	27.12	27.28
4	GCN-GRU	MAE	16.10	19.18	20.49	24.03	23.48	24.52
5	GAT-LSTM	MAE	<u>15.02</u>	<u>17.55</u>	<u>19.14</u>	<u>22.59</u>	<u>22.77</u>	<u>23.01</u>
6	G-C-LSTM	MAE	**14.33**	**17.40**	**18.75**	**20.66**	**22.52**	**22.94**
		Relative Improve	**4.59%**	**0.85%**	**2.04%**	**8.54%**	**1.10%**	**0.30%**
		Average improve			**2.90%**			

Table 7. Predicting PM2.5 values comparison between graph-capsule-LSTM and baseline methods. **|P-T|** represents |Predicted PM2.5 **-** True PM2.5|. The smaller the value of **|P-T|**, the closer is Predicted PM2.5 to True PM2.5. Bold indicates the best predicted PM2.5 values, and an underline means the true PM2.5 values.

Number	Model	+1 h	|P-T|	+2 h	|P-T|	+3 h	|P-T|	+4 h	|P-T|	+5 h	|P-T|	+6 h	|P-T|
1	CNN-RNN	355	74	345	79	310	70	289	66	266	73	239	79
2	CNN-GRU	247	34	225	41	202	38	181	42	143	50	218	58
3	CNN-LSTM	248	33	227	39	203	37	183	40	142	51	217	57
4	GCN-GRU	304	23	290	24	265	25	253	30	237	44	205	45
5	GAT-LSTM	300	19	284	18	258	18	248	25	233	40	206	46
6	G-C-LSTM	**291**	**10**	**281**	**15**	**257**	**17**	**243**	**20**	**218**	**25**	**189**	**29**
	True PM2.5	<u>281</u>		<u>266</u>		<u>240</u>		<u>223</u>		<u>193</u>		<u>160</u>	

4.3.2. Effect of dynamic routing counts on the model for graph capsule networks

By introducing graph capsule networks, the number of dynamic routes is a hyperparameter that affects the performance of the model. The so-called dynamic routing is a cyclic aggregation process, where the similarity between the capsules before and after aggregation is calculated for each route to be used as a coefficient for the next aggregation, which is equivalent to a voting or clustering process. To verify the impact of this hyperparameter, Node 1 (Beijing) is taken as the target node, and dynamic routing was set 1, 2, 3, 4, 5 and 6 times in sequence. Tables 8 and 9 show the impact of the number of dynamic routes on the model prediction results.

Table 8. RMSE values of graph-capsule-LSTM model for 1–6 dynamic routing cases. Bold indicates the best results.

Dynamic Routing	**Metric**	**+1 h**	**+2 h**	**+3 h**	**+4 h**	**+5 h**	**+6 h**	**Predicting Time/s**
1	RMSE	26.87	32.06	37.55	40.18	41.05	41.74	45
2	RMSE	22.11	29.48	32.97	35.89	37.82	37.28	56
3	RMSE	21.58	26.36	30.17	31.92	33.56	34.28	70
4	RMSE	**21.09**	**26.14**	**29.18**	**30.33**	32.73	**33.01**	88
5	RMSE	21.37	26.44	30.03	30.46	**32.16**	33.75	117
6	RMSE	21.28	27.42	31.87	32.45	33.36	34.06	141

Table 9. MAE values of graph-capsule-LSTM model for 1–6 dynamic routing cases. Bold indicates the best results.

Dynamic Routing	Metric	+1 h	+2 h	+3 h	+4 h	+5 h	+6 h	Predicting Time/s
1	MAE	18.71	21.47	22.18	23.59	24.63	24.47	45
2	MAE	15.96	19.30	21.47	22.78	23.17	24.56	56
3	MAE	14.33	17.40	18.75	20.66	22.52	**22.94**	70
4	MAE	14.58	**17.26**	**18.42**	**20.51**	**21.62**	23.14	88
5	MAE	**14.03**	17.34	18.59	20.78	22.53	23.57	117
6	MAE	14.98	17.44	18.85	21.26	22.83	23.92	141

Observing the experimental results in Tables 8 and 9, it can be found that the values of RMSE and MAE were 26.87 and 18.71, respectively, when one dynamic routing was used for PM2.5 prediction in the next hour. The values of both metrics decreased to 22.11 and 15.95 when one dynamic routing was added, which is a larger reduction in error. When three times dynamic routing was applied, the RMSE further decreased to 21.58 and the MAE decreased to 14.33. After that, the RMSE and MAE decreased more slowly. The distribution of the bold data (minimum error values) in the two tables shows that the model achieved essentially the smallest error values and the best performance at four routings. The last column in the table shows the prediction time due to different dynamic routing counts.

4.3.3. Effects of different embedding sizes of the graph capsule network output

In the embedding-based approach, the size of the embedding is theoretically a hyperparameter that can be set arbitrarily, and it is a key factor affecting the model performance. To verify the impact of this hyperparameter, different output embedding dimensions (the final size of the output capsule determined by the weight matrix W_{ij} of Eq (4)) were set for the graph capsule network in each experiment to test the impact of this parameter on the model performance. The capsule lengths were sequentially set to 50, 100, 150, 200, 250 and 300 dimensions by the column dimension of W_{ij} for the comparison experiments. Figures 6 and 7 respectively show the convergence trends of the RMSE and MAE of the model when predicting the future 1-hour PM2.5 values in the target city (Beijing) for different capsule lengths. The horizontal coordinates denote the number of generations of model training, and the models of each length are denoted as Graph-Capsule-LSTM-50, Graph-Capsule-LSTM-100, Graph-Capsule-LSTM-150, Graph-Capsule-LSTM-200, Graph-Capsule- LSTM-250 and Graph-Capsule-LSTM-300, where the last number of each name represents the length of the capsule in the model.

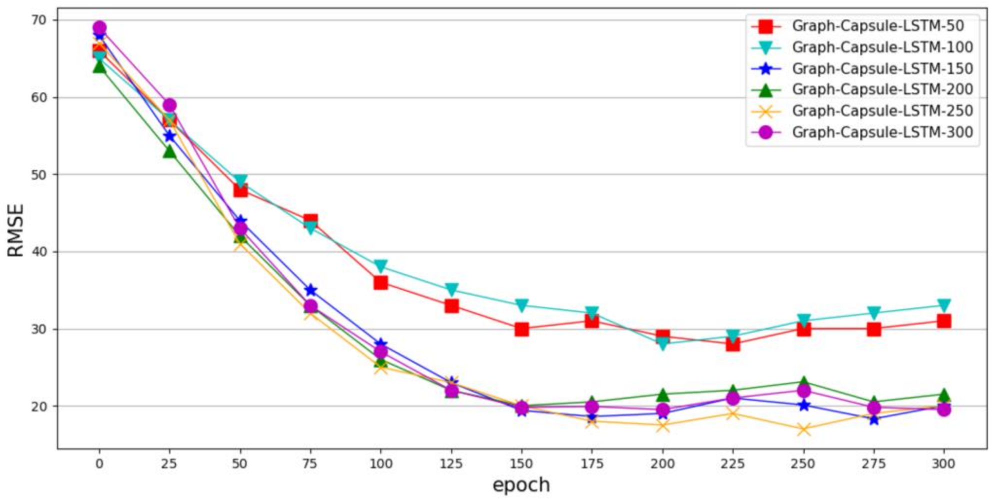

Figure 6. Trends of RMSE during training for graph-capsule-LSTM models with different embedding sizes.

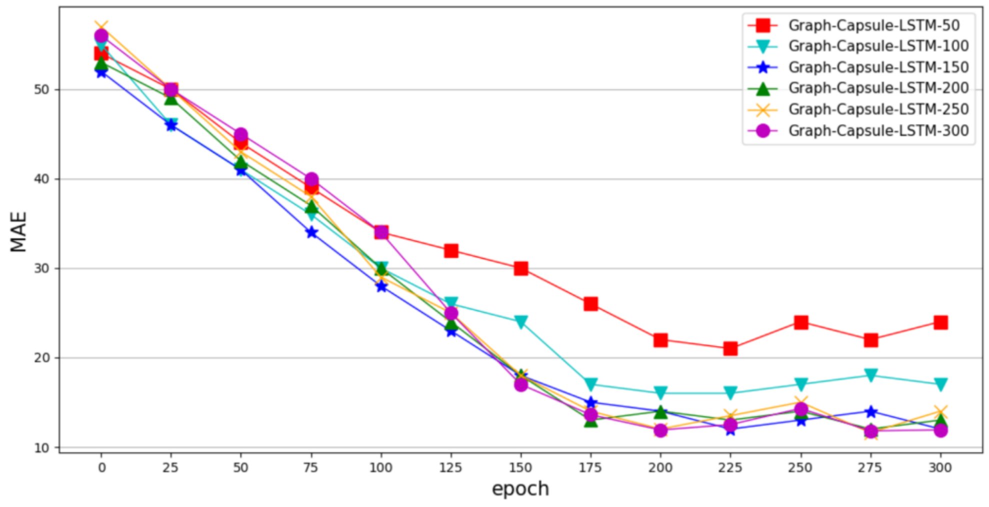

Figure 7. MAE trends of graph-capsule-LSTM models with different embedding sizes during training.

Figure 6 shows that, on the PM2.5 prediction assessment metric RMSE, Graph-Capsule-LSTM-50 and Graph-Capsule-LSTM-100 started to converge at about 200 epochs and the minimum training error for these two models was 28.16 and 28.22, respectively. For the Graph-Capsule-LSTM-150 model, the RMSE converged much faster, having begun to converge by 175 epochs of training and dropping to a minimum of 18.35 by 275 generations. However, when increasing the length of the features extracted by the graph capsule to 200, 250 or 300 dimensions, the model's ability to predict PM2.5 stalled and the RMSE essentially stopped decreasing.

From Figure 7, Graph-Capsule-LSTM-50 and Graph-Capsule-LSTM-100 started to converge when they were trained to about 200 epochs. From the observations, the minimum MAE reached during training was 21.05 for Graph-Capsule-LSTM-50 and 16.72 for Graph-Capsule-LSTM-100 on the training set. When the graph capsule output vector length was 150, the model Graph-Capsule-LSTM-150 started to converge at around epochs 175 with a minimum MAE of 12.24. When the length of the model’s output embedding continued to increase, Graph-Capsule-LSTM-200, Graph-Capsule-LSTM-250 and Graph-Capsule-LSTM-300 performance did not improve significantly.

As can be seen in Figures 6 and 7, in general, the larger the dimensionality of the node feature vector of the capsule network output, the better the prediction of the model. For the dataset of the model in this chapter, it can be seen from the figure that both the RMSE and MAE values reached the optimum when the output size of the capsule was set to 150.

4.3.4. Effects of using different distance thresholds to construct the urban adjacency matrix on the prediction results

Taking the predicted PM2.5 concentration value of Node 1 (Beijing) in the next 1-hour as an example, in order to determine the appropriate neighbor distance, the distance thresholds were set to 100, 150, 200, 250, 300, 350, 400 and 450 km, respectively. Neighborhood information of each node was obtained at each threshold and fed into the capsule network to learn the spatial characteristics of the target node. Figures 8 and 9 respectively show the RMSE and MAE values of the graph-capsule-LSTM model on the test set for different distance thresholds.

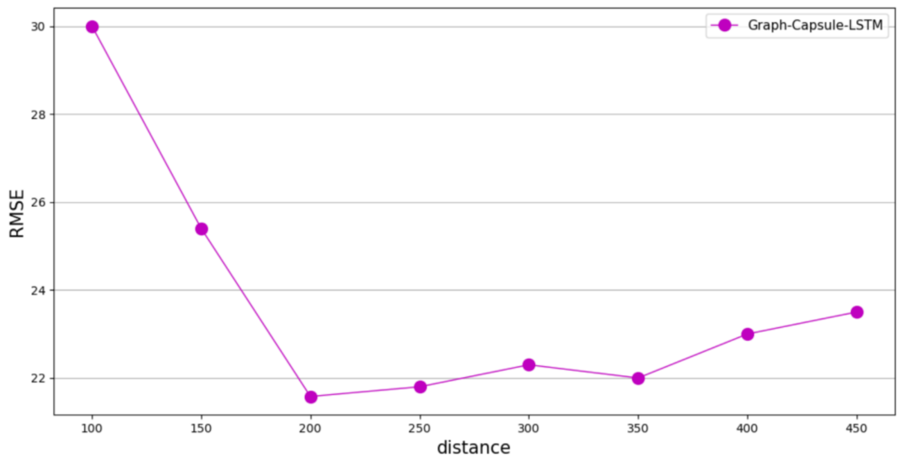

Figure 8. RMSE values of graph-capsule-LSTM model on the test set for different distance thresholds.

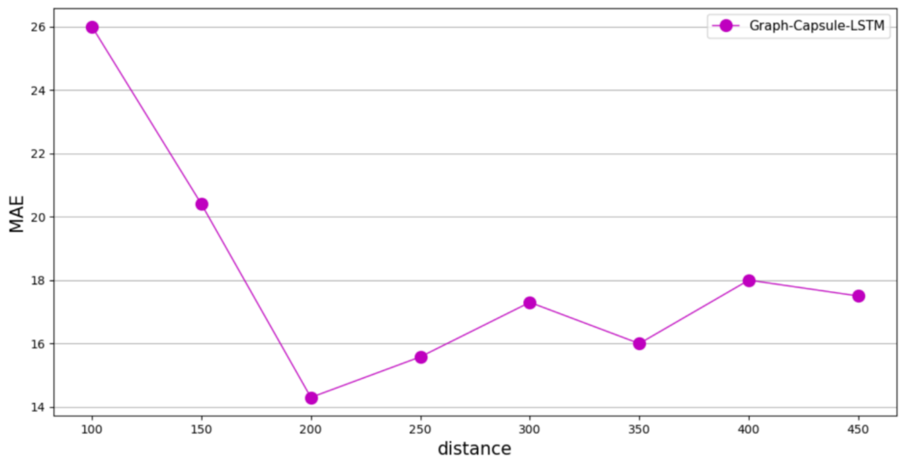

Figure 9. MAE values of the graph-capsule-LSTM model on the test set for different distance thresholds.

The graph-capsule-LSTM model was tested on the test set. From Figures 8 and 9, it can be seen that the model can obtain the best performance when cities within a 200-km distance were selected as neighboring cities; the RMSE and MAE reached minimum values of 21.58 and 14.33, respectively. That is, for the dataset, the graph-capsule-LSTM model should select neighboring cities within 200 km and send their feature data to the capsule network to obtain the neighboring feature embeddings of the target node. Then, this feature and the local feature data of the target node are combined to form a time series, which is sent to the LSTM network for training and prediction.

5. Conclusions

In this paper, a PM2.5 prediction model based on a graph capsule network and an LSTM RNN is proposed to address the "invariance" problem of GCNs. First, we analyzed the "invariance" defect of traditional convolutional networks, which led to the conclusion that GCNs and GATs also cause PM2.5 prediction errors due to "invariance". In order to solve this problem, a "graph capsule network" has been proposed to replace the GCN or GAT. The model is divided into two stages. In the first stage, the first-order neighbor feature matrix of the target city is fed into the capsule network to learn the neighbor feature embedding vector of the target node. In the second stage, for the target city, the neighbor feature vectors of the target city at each time step are obtained on the timeline and then combined with the local features of the target nodes before being fed into the LSTM RNN to learn and predict the PM2.5 values of the target city at future time steps. In the end, the effectiveness of the model was validated on the dataset and analyzed in comparison with baseline models.

Further, we will improve our work in the following directions in the future.

(1) More emphasis should be placed on the trade-off between the complexity of the model architecture and the overall performance of the model. This method tends to consider the tendency of more complex models at the expense of examining the ability to perform modeling tasks with simpler models. Sometimes, the complexity of the model can also undermine the overall performance of the model.

(2) In addition to the effective replicability, predictability and structure of the model, the model proposed in this paper should be evaluated in terms of computational penalties and running time, as these are essential factors for the model to be sufficiently useful in real-world modeling tasks.

(3) Uncertainty analysis of the model results should also be studied more in future work. In order to allow deep learning models to be fully implemented in practical applications, future efforts should be made toward quantifying the uncertainty of the model results.

(4) The training of the model should be analyzed for additional factors that may affect the performance of the model, such as a) the impact of the length of the training data utilized on the overall performance of the deep learning model, b) the impact of the chosen predictors and the model structure utilized on the complexity and overall performance of the deep learning model, c) the connection between the chosen data normalization scheme and the activation function utilized and d) the impact of the adjusted impact of the initialization scheme of weights, biases and other training parameters on the overall performance of the deep learning model.

Acknowledgments

This study was supported by the Jilin Provincial Development and Reform Commission (contract number 2022C046-5), the 2022 Jilin Provincial Education Department Project (contract number JJKH20221172KJ).

Conflict of interest

The authors declare that there is no conflict of interest.

References

1. World Health Organization, WHO. Ambient (Outdoor) Air Quality and Health.Retrieved. 2016. Available from: http://www.who.int/mediacentre/factsheets/fs313/en/ (Accessed date:14 September 2017)
2. Z. Shang, T. Deng, J. He, X. Duan, A novel model for hourly PM2. 5 concentration prediction based on CART and EELM, *Sci. Total Environ.*, **651** (2019), 3043–3052. https://doi.org/10.1016/j.scitotenv.2018.10.193
3. A. Kumar, R. S. Patil, A. K. Dikshit, S. Islam, R. Kumar, Evaluation of control strategies for industrial air pollution sources using American meteorological society/ environmental protection agency regulatory model with simulated meteorology by weather research and forecasting model, *J. Cleaner Prod.*, **116** (2016), 110–117. https://doi.org/10.1016/j.jclepro.2015.12.079
4. F. Jiang, Y. Q. Qiao, PM2.5 concentration prediction based on sample entropy and improved extreme learning machine, *Statistics & Decision*, **37** (2021), 166–171.
5. H. Y. Luo, D. Y. Wang, Y. L. Liu, S. Wei, Y. B. Lin, PM2.5 Concentration forecasting based on two-layer decomposition technique and improved extreme learning machine, *Syst. Eng.-Theo. Practice*, **38** (2018), 1321–1330.
6. S. Moisan, R. Herrera, A. Clements, A dynamic multiple equation approach for forecasting PM2. 5 pollution in Santiago, Chile, *Int. J. Forecast.*, **34** (2018), 566–581. https://doi.org/10.1016/j.ijforecast.2018.03.007
7. H. X. Jiang, J. Tian, C. H. Sun, Ensemble model of recursive random forest and multilayer neural network for PM2.5 prediction, *Syst. Eng.*, **38** (2020), 14–24.

8. W. Sun, J. Y. Sun, Daily PM2. 5 concentration prediction based on principal component analysis and LSSVM optimized by cuckoo search algorithm, *J. Environ. Manag.*, **188** (2017), 144–152. https://doi.org/10.1016/j.jenvman.2016.12.011
9. Q. Sun, Y. M. Zhu, X. M. Chen, A. Xu, X. Peng, A hybrid deep learning model with multi-source data for PM 2.5 concentration forecast, *Air Qual. Atmos. Hlth.*, **14** (2020), 1–11. https://doi.org/10.1007/s11869-020-00954-z
10. W. T. Yang, M. Deng, F. Xu, H. Wang, Prediction of hourly PM2. 5 using a space-time support vector regression model, *Atmos. Environ.*, **181** (2018), 12–19. https://doi.org/10.1016/j.atmosenv.2018.03.015
11. G. Shi, Y. Leung, J. S. Zhang, T. Fung, F. Du, Y. Zhou, A novel method for identifying hotspots and forecasting air quality through an adaptive utilization of spatio-temporal information of multiple factors, *Sci. Total Environ.*, **759** (2021), 143–513. https://doi.org/10.1016/j.scitotenv.2020.143513
12. Q. P. Zhou, H. Y. Jiang, J. Z. Wang, J. L. Zhou, A hybrid model for PM2. 5 forecasting based on ensemble empirical mode decomposition and a general regression neural network, *Sci. Total Environ.*, **496** (2014), 264–274. https://doi.org/10.1016/j.scitotenv.2014.07.051
13. K. R. Weng, M. Liu, Q. Liu, An integrated prediction model of PM2.5 concentration based on TPE-XGBOOST and LassoLars, *Syst. Eng.-Theo. Practice*, **40** (2020), 748–760.
14. P. P. Xiong, S. Huang, M. Peng, X. Wu, Examination and prediction of fog and haze pollution using a multi-variable grey model based on interval number sequences, *Appl. Math. Model.*, **77** (2020), 1531–1544. https://doi.org/10.1016/j.apm.2019.09.027
15. Z. C. Wang, L. R. Chen, J. M. Zhu, H. Chen, H. Yuan, Double decomposition and optimal combination ensemble learning approach for interval-valued AQI forecasting using streaming data, *Environ. Sci. Pollut. Res.*, **27** (2020), 37802–37817. https://doi.org/10.1007/s11356-020-09891-x
16. J. Xu, L. Chen, M. Lv, C. Zhan, S. Chen, J. Chang, HighAir: A Hierarchical Graph Neural Network-Based Air Quality Forecasting Method, *arXiv,* (2021), arXiv:2101.04264. 86.
17. H.C. Chen, K.T. Putra, J. A. Chun-WeiLin, Novel Prediction Approach for Exploring PM2.5 Spatiotemporal Propagation Based on Convolutional Recursive Neural Networks, *arXiv,* (2021) arXiv:2101.06213. 96.
18. R. Yan, J. Liao, J. Yang, W. Sun, M. Nong, F. Li, Multi-hour and multi-site air quality index forecasting in Beijing using CNN, LSTM, CNN-LSTM, and spatiotemporal clustering, *Expert Syst. Appl.*, **169** (2021), 114513. https://doi.org/10.1016/j.eswa.2020.114513
19. X. L. Dai, J. J. Liu, Y. L. Li, A recurrent neural network using historical data to predict time series indoor PM2.5 concentrations for residential buildings, *Indoor Air*, **31** (2021), 1228–1237. https://doi.org/10.1111/ina.12794
20. D. Seng, Q Zhang, X. Zhang, G. Chen, X. Chen, Spatiotemporal prediction of air quality based on LSTM neural network, *Alex. Eng. J.*, **60** (2021), 2021–2032. https://doi.org/10.1016/j.aej.2020.12.009