The problem of air pollution threatens public health. Air quality forecasting can provide the air quality index hours or even days later, which can help the public to prevent air pollution in advance. Previous works focus on citywide air quality forecasting and cannot solve nationwide city forecasting problems, whose difficulties lie in capturing the latent dependencies between geographically distant but highly correlated cities. In this article, we propose the group-aware graph neural network (GAGNN), a hierarchical model for nationwide city air quality forecasting. The model constructs a city graph and a city group graph to model the spatial and latent dependencies between cities, respectively. GAGNN introduces a differentiable grouping network to discover the latent dependencies among cities and generate city groups. Based on the generated city groups, a group correlation encoding module is introduced to learn the correlations between them, which can effectively capture the dependencies between city groups. After the graph construction, GAGNN implements message passing mechanism to model the dependencies between cities and city groups. The evaluation experiments on two real-world nationwide city air quality datasets, including the China dataset and the US dataset, indicate that our GAGNN outperforms existing forecasting models.

CCS Concepts: • Information systems → Data mining; Spatial-temporal systems; • Computing methodologies → Neural networks;

Additional Key Words and Phrases: Air quality forecasting, deep learning, graph neural network, urban computing

ACM Reference format:
Ling Chen, Jiahui Xu, Binqing Wu, and Jianlong Huang. 2023. Group-Aware Graph Neural Network for Nationwide City Air Quality Forecasting. ACM Trans. Knowl. Discov. Data. 18, 3, Article 55 (December 2023), 20 pages.
https://doi.org/10.1145/3631713

1 INTRODUCTION

Air pollution causes a range of health problems, particularly harming the cardiopulmonary system. The air quality index (AQI) provides a quantitative description of air quality conditions and is
calculated from the concentration of air pollutants in an area. Forecasting the AQI values of nationwide cities can help the government combat air pollution and the public perceive local air quality trends in advance, especially during major air pollution events, e.g., dust storms and wildfires.

Nationwide city air quality forecasting, as a typical spatial-temporal forecasting problem, involves multiple challenges. First, AQI values have complex dependencies in temporal dimensions with implied periodicity and trendiness. Second, AQI values have complex dependencies in spatial dimensions, with dependencies between both geographically adjacent and distant cities.

Existing air quality forecasting models can be divided into two categories: physical models and machine learning models. Physical models [2, 32, 41] are designed based on the theories of air motion and matter diffusion, which require the integrity of pollution source data and have poor generalization ability. Machine learning models learn the relationship between input features and AQI values from the data and can be further subdivided into time series analysis models, statistical regression models, and deep learning models. Time series analysis models [18] forecast AQI values by finding linear patterns in the historical observation series, but cannot accept the feature inputs other than sequence data. Statistical regression models [27, 37, 47] are capable of supporting multi-source feature inputs, but the performance of models depends on feature engineering. Deep learning models can automate feature learning by stacking multiple neural networks to fit the nonlinear transformation from inputs to outputs. These models generally employ recurrent neural networks (RNNs) [23, 30, 42] and their variants [3, 43, 44, 48] to model complex dependencies in temporal dimensions, and convolutional neural networks (CNNs) [5, 6] or graph neural networks (GNNs) [1, 4, 10, 24, 25, 30, 36, 42] to model complex dependencies in spatial dimensions. However, most of these models focus on citywide air quality forecasting and do not consider the dependencies between cities. A few works [24, 30, 36, 42] model the spatial dependencies between cities, i.e., geographically adjacent cities have similar air quality, but ignore the latent dependencies between geographically distant but highly correlated cities, e.g., the air quality of coastal cities is affected by sea breezes [28]. The receptive field of cities in these works is limited, which is a common problem in spatial-temporal forecasting.

For those models that employ GNNs, a naive strategy that expands the receptive field of entities is to deepen GNNs. However, recent studies [19, 22, 34] discovered that GNNs suffer from the over-smoothing issue when going deeper, i.e., the representations of adjacent entities converge and their local features are lost.

Hierarchical graph neural networks (HGNNs) are a kind of GNNs that construct multi-level graphs [11, 21, 38, 42, 46] and implement interactions among multi-level graphs to model the dependencies between entities. HGNNs expand the receptive field of entities by constructing multi-level graphs rather than deepening GNNs, thus relieving the conflict between expanding receptive fields and preserving local features. However, existing HGNNs rely on predefined rules to construct the coarsened graph, which cannot effectively capture the latent dependencies between entities.

In this article, we propose the group-aware graph neural network (GAGNN), a hierarchical model for nationwide city air quality forecasting. The main contributions of our work are summarized as follows:

(1) We propose GAGNN, which constructs a city graph and a city group graph to model the spatial and latent dependencies between cities, respectively.
(2) We introduce the group correlation encoding module, which end-to-end learns the correlations between city groups to effectively capture the dependencies between city groups.
(3) We evaluate GAGNN on two real-world nationwide city air quality datasets, including the China dataset and the US dataset, and compare it with the SOTA spatial-temporal forecasting models. The experimental results indicate that GAGNN outperforms existing models.
2 RELATED WORK

2.1 Air Quality Forecasting

Existing air quality forecasting models can be divided into two categories: physical models and machine learning models. Physical models, e.g., street canyon model [32, 41] and Gaussian plume model [2], are designed based on the theories of air motion and matter diffusion, which forecast air quality by simulating the emission and diffusion processes of air pollutants. However, these models require the integrity of pollution source data and have poor generalization ability.

Machine learning models learn the relationship between input features and AQI values from data and can be further subdivided into time series analysis models, statistical regression models, and deep learning models. Time series analysis models forecast AQI values by finding linear patterns in the historical observation series. Lee et al. [18] introduced ARIMA, a time-series analysis model, to forecast air quality. Time series analysis models cannot utilize other influencing factors, e.g., locations, as they cannot accept feature inputs other than sequence data. Statistical regression models are capable of supporting multi-source feature inputs. Yu et al. [47] introduced the random forest model to capture the complex nonlinear relationships between multi-source influencing factors and air quality. Wang et al. [37] utilized the RBF network and SVR to forecast air quality, further introducing PCA to reduce the dimensionality of input data. These models are simply used to adapt different feature inputs to different tasks, and their performances depend on feature engineering.

Deep learning models can automate feature learning by stacking multiple neural networks to fit the nonlinear transformation from inputs to outputs. Liang et al. [23] proposed GeoMAN, an encoder-decoder-based spatial-temporal forecasting framework, which uses an attention mechanism to model the correlations between different metrics at the same sensor and the same metric at different sensors. Chen et al. [5] proposed PANDA, a multi-task air quality modeling framework, which implements air quality forecasting and air quality estimation in one model. Qi et al. [31] embedded feature selection and semi-supervised learning methods in deep neural networks, using unlabeled spatial-temporal information to improve the forecasting results. Liu et al. [26] presented a compression and regularized optimization scheme for the modules stacked residual deep fuzzy system to better predict time series.

Recently, GNNs have become practical tools for modeling non-Euclidean distributed entities. Qi et al. [30] proposed GC-LSTM, which introduces the GCN to capture the dependencies among air quality monitoring stations. Lin et al. [25] proposed GC-DCRNN, which constructs the graph based on the geographic context similarity between monitoring stations, and combines the diffusion convolution operation with the GRU gate. Ge et al. [10] constructed several graphs based on different similarity metrics between monitoring stations, modeling the correlations between stations in different semantic spaces. Wang et al. [36] introduced prior knowledge into the graph construction process, enabling the massage passing process in the graph perceiving weather factors. Liang et al. [24] used graph neural networks and multi-head self-attentions in the deterministic stage and a variational autoencoder in the stochastic stage to predict nationwide air quality in China. Ren et al. [33] represented the inter-dependencies between two areas using planar Gaussian diffusion equations to forecast air quality in a city. Han et al. [12] used geographical distance and environmental context to build multi-view graphs connecting air quality and weather monitoring stations, which were utilized for citywide spatial dependency modeling.

Most of the above models focus on citywide air quality forecasting and do not consider the dependencies between cities. Some works [12, 24] model the spatial dependencies between cities, but they ignore the latent dependencies between geographically distant but highly correlated cities.
2.2 GNNs to Expand the Receptive Field

Expanding the receptive field of entities enables GNNs to capture the dependencies between distant entities. To achieve this, a naive strategy is deepening GNNs [39]. However, message aggregation and representation updating in each GNN layer make the representations of adjacent entities more similar. The local features of entities would be lost when GNNs go deeper, and this issue is called over-smoothing [22].

There are two types of methods to expand the receptive field without raising the over-smoothing issue: improving the architectural designs of GNNs [19, 20, 34] and introducing HGNNs. The former methods preserve the local features of nodes by introducing some specific designs of GNNs. Rong et al. [34] randomly removed a certain number of edges at each training epoch to alleviate both over-smoothing and over-fitting issues. Li et al. [19] referred to ResNet [13] and introduced skip connections in GNNs. Built on the previous work, Li et al. [20] further introduced message normalization and proposed a pre-activation version of GNN. The receptive field size, i.e., the layer number of GNNs, of these methods is fixed, which is not adaptive for each entity in the graph.

HGNNs construct multi-level graphs by graph pooling methods [8, 17, 45] to relieve the conflict between expanding receptive fields and preserving local features. Yu et al. [46] proposed ST-UNet, which introduces a heuristic graph pooling method to construct the coarsened graph. Li et al. [21] utilized a geography-based HGNN to model a geographic information system, learning the relationship between socio-economic Census data and election results. Wu et al. [38] proposed HRNR, which constructs multi-level graphs to learn the representations of road segments. Zhang et al. [49] proposed SHARE, which combines HGNNs with semi-supervised learning to forecast citywide parking availability. Xu et al. [42] proposed HighAir, which constructs a city graph and station graphs to consider the city-level and station-level patterns of air quality, respectively. Guo et al. [11] proposed HGCN, which applies spectral clustering to construct the region-level graph based on the road-level distance adjacency matrix. HGCN utilizes multiple GCNs on region and road-level graphs to extract features and instigates the extracted features of both levels by dynamic transfer blocks for traffic forecasting.

However, existing models rely on predefined rules to construct the coarsened graph, i.e., using the geographic distance between nodes in the basic graph to define the correlations between nodes in the coarsened graph, which cannot effectively capture the latent dependencies between entities.

3 METHODOLOGY

3.1 Definitions

Cities and city groups: We define $C = \{c_i\}_{i=1}^{N_{\text{city}}}$ as the set of cities, $L \in \mathbb{R}^{N_{\text{city}} \times 2}$ as the location matrix of cities, i.e., longitude and latitude, and $G = \{g_i\}_{i=1}^{N_{\text{group}}}$ as the set of city groups, where $N_{\text{city}}$ denotes the number of cities and $N_{\text{group}}$ denotes the number of city groups.

City graph and city group graph: We define $G = (V, A, X, E)$ as the city graph, where $V$ denotes the set of city nodes, $A$ denotes the set of edges, $X$ denotes the node attribute matrix, and $E$ denotes the edge attribute matrix. We define $G = (V, \mathcal{A}, Z, R)$ as the city group graph, where $V$ denotes the set of city group nodes, $\mathcal{A}$ denotes the set of edges, $Z$ denotes the node attribute matrix, and $R$ denotes the edge attribute matrix.

We construct the city graph during the pre-processing phase, while the city group graph will be constructed in subsequent processes. The construction rules of the city graph are as follows:

$$d_{i,j} = d_{j,i} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}, \quad (1)$$

$$E_{i,j} = \frac{1}{d_{i,j}}, \quad 0 < d_{i,j} < R_h, \quad (2)$$

ACM Transactions on Knowledge Discovery from Data, Vol. 18, No. 3, Article 55. Pub. date: December 2023.
where \([x_i, y_i]\) and \([x_j, y_j]\) denote the locations of cities \(c_i\) and \(c_j\), respectively. \(d_{i,j}\) and \(d_{j,i}\) denote the Euclidean distance between city \(c_i\) and city \(c_j\), and \(R_h\) is the distance threshold. Only two cities with a distance less than \(R_h\) have connected on the city graph. \(E_{i,j}\) denotes the edge attributes of the edge from city \(c_i\) to city \(c_j\), and \(E_{i,j}\) and \(E_{j,i}\) are symmetrical.

**AQI data:** The AQI of city \(c_i\) at time slot \(t\) is represented as \(I^t_i\).

**Weather data:** The weather data of city \(c_i\) at time slot \(t\) include humidity, rainfall, air pressure, temperature, wind speed, and wind direction. Following [42], we encode the wind direction data as a two-bit vector, and the encoding rules are shown in Table 1.

**Historical observation data:** The historical observation data of city \(c_i\) at time slot \(t\) are denoted as \(H^t_i\), which consists of AQI data and weather data. Thus the historical observation sequence of city \(c_i\) at time slot \(t\) is denoted as \(H^t_i = \{H^t_{i-\tau_{in}+1}, H^t_{i-\tau_{in}+2}, \ldots, H^t_i\}\), where \(\tau_{in}\) is the historical window length.

**Time data:** The month, week, and day information of time slot \(t\) are represented as one-hot encoding vectors, which are embedded end-to-end in the model as month vector, week vector, and hour vector. The time vector of time slot \(t\) is denoted as \(T^t\), concatenating the above three vectors.

**Nationwide city air quality forecasting:** Given the city locations \(L\), the historical observation sequence \(H^t_i\), and the time vector \(T^t\) of time slot \(t\), nationwide city air quality forecasting task aims at forecasting the next \(\tau_{out}\) AQI values for all cities, where \(\tau_{out}\) denotes the forecasting horizon.

### 3.2 Framework

Figure 1 shows the framework of GAGNN, which leverages an encoder-decoder architecture. For the encoder, GAGNN utilizes a self-attention network to extract the features of the historical observation sequence and obtain the city representations. In addition, GAGNN introduces a differentiable grouping network to discover the latent dependencies among cities and groups cities into several city groups by a learning method. Based on the generated city groups, a group correlation encoding module is introduced to learn the correlations between them, which can better capture the dependencies between city groups. GAGNN implements the message passing mechanism in the city graph and the city group graph to model the dependencies between cities and city groups, respectively. The architecture of the decoder is similar to that of the encoder. The decoder forecasts the AQI values of all nationwide cities based on the outputs generated by the encoder, i.e., the updated city representations, the mapping relationships between cities and city groups, and the encoded correlations between city groups.

| Direction     | Vector  |
|---------------|---------|
| North         | [0,1]   |
| Northeast     | [1,1]   |
| East          | [1,0]   |
| Southeast     | [1,−1]  |
| South         | [0,−1]  |
| Southwest     | [−1,−1] |
| West          | [−1,0]  |
| Northwest     | [−1,1]  |
| No sustained direction | [0,0]   |
3.3 Sequence Feature Extraction

The self-attention network accepts the historical observation sequence $H^{t-t_{in}+1:t}$ as input and extracts features to obtain the city representations $X$. As shown in Figure 2, the design of self-attention network refers to the encoder architecture of Transformer [35], where multi-head self-attention module implements the point-wise attention operation with multiple different sets of parameters. The computational process of the self-attention mechanism is defined as follows:

$$\begin{align*}
Q &= f(H, W_{\text{query}}), \\
K &= f(H, W_{\text{key}}), \\
V &= f(H, W_{\text{value}}), \\
\text{Attention}(Q, K, V) &= \text{softmax} \left( \frac{QK^T}{\sqrt{d_{\text{key}}}} \right) V,
\end{align*}$$

where $H$ denotes the historical observation sequence $H^{t-t_{in}+1:t}$ for simplicity, and transformed to query matrices $Q$, key matrices $K$, and value matrices $V$. Here, $W_{\text{query}}$, $W_{\text{key}}$, $W_{\text{value}}$ are learnable parameters and $d_{\text{key}}$ is the dimension of keys.

After the point-wise attention operation implemented by matrix multiplication, a sequence of vectors is obtained. After that, layer normalization and skip connection design are introduced, which can stabilize the output distribution of the network to reduce the difficulty of model training.

3.4 City Grouping

Geographically distant cities may have strong correlations, but introducing multi-layer GNNs to model the latent dependencies between distant cities would cause the over-smoothing issue. Differentiable grouping network utilizes a learning method to capture the mapping relationships between cities and city groups, and generates city groups to discover the latent dependencies between cities. In this way, cities with strong latent dependencies would share the city group representations.
Specifically, a differentiable grouping network utilizes assignment matrix \( S \in \mathbb{R}^{N_{\text{city}} \times N_{\text{group}}} \) to indicate the mapping relationships between cities and city groups, where \( S_{i,j} \) denotes the probability of assigning \( i \)th city to \( j \)th city group. Thus, we have \( \sum_{k=1}^{N_{\text{group}}} S_{i,k} = 1 \). \( S \) is randomly initialized and would be optimized during the training phase. A city can be assigned to multiple city groups with weights representing the relevance between the city and different city groups. Figure 3 further clarifies \( S \) with a case, in which there are 6 cities and 2 city groups. The probability of assigning city \( c_2 \) to city group \( g_1 \) is 0.7 and the probability of assigning city \( c_2 \) to city group \( g_2 \) is 0.3, which indicates that city \( c_2 \) is more correlated with city group \( g_1 \) than city group \( g_2 \).

The transformation from the city representations to the city group representations is achieved by \( S \). In addition, GAGNN introduces the geographic locations of cities in the transformation as their spatial features to capture the latent dependencies between adjacent cities, which is defined as follows:

\[
X'_i = f_v (X_i, L_i),
\]

\[
Z_j = \sum_{i=1}^{N_{\text{city}}} S_{i,j} X'_i,
\]

where \( X_i \) is the output of the self-attention network, \( L_i \) is the geographic location of city \( c_i \), \( X'_i \) is the city representation of city \( c_i \) containing geographic information, \( Z_j \) is the city group representation of city group \( g_j \) obtained by cities assigned to it, and \( f_v \) is a fusion function implemented by a multi-layer perceptron (MLP).

### 3.5 Modeling the Dependencies Between City Groups

GAGNN introduces a group correlation encoding module to construct the city group graph. After that, GAGNN implements a message passing mechanism in the city group graph to model the dependencies between city groups.

Existing models [21, 45, 49] define the correlations between city groups by the correlations between the cities assigned to them, e.g., the geographic distance, which cannot effectively capture the dependencies between city groups.

To address this problem, GAGNN introduces a group correlation encoding module, which encodes the edge attributes between city group nodes by a learning method. Specifically, the group correlation encoding module considers not only the city group representations, but also other factors affecting the correlations between city groups, e.g., time information. We construct the city group graph as a fully connected graph. Considering that \( N_{\text{group}} \) is much smaller than \( N_{\text{city}} \), the computational cost to encode the correlations between pair-wise city groups is acceptable.
process of group correlation encoding is defined as follows:
\[ R_{i,j} = \text{ReLU} \left( \text{enc} \left( Z_i, Z_j, T \right) \right), \] (7)
where \( R_{i,j} \) denotes the edge attributes of the edge between city groups \( g_i, g_j \), and \( Z_i, Z_j \) are their representations, respectively. \( T \) is the time vector, which affects the correlations between city groups, and \( \text{enc} \) is an encoding function implemented by an MLP.

Message passing mechanism is implemented to model the dependencies between city groups, which consists of two major processes: message aggregation and representation updating. The details are shown as follows:
\[ R_i = \left\{ (Z_i, Z_j, R_{j,i}) \right\}_{i \neq j}, \] (8)
\[ r_i \leftarrow \rho_g \left( R_i \right), \] (9)
\[ Z'_i \leftarrow \phi_g \left( r_i, Z_i \right), \] (10)
where \( R_i \) is the set containing all the messages passed to city group \( g_i \), which would be transformed to vector \( r_i \) later, and \( Z'_i \) is the updated city group representation of city group \( g_i \) based on \( r_i \). \( \rho_g \) and \( \phi_g \) are transformation functions implemented by MLPs, and both of them are shared for each city group.

### 3.6 Modeling the Dependencies Between Cities

To model both the spatial and latent dependencies between cities, GAGNN updates the representations of cities based on the city groups they are assigned to and implements the message passing mechanism in the city graph.

Similar to the transformation process in Section 3.4, the group-based representations of cities can be obtained as follows:
\[ H_i = \sum_{j=1}^{N_{\text{group}}} S_{i,j} Z'_j, \] (11)
where \( Z'_j \) is the updated representation of city group \( g_j \) and \( H_i \) is the group-based representation of city \( c_i \).

Slightly different from the calculation process in the city group graph, the message passing mechanism in the city graph firstly fuses the group-based representation of city and the representation obtained from the self-attention network. In this way, the fused city representations contain information of local features and the city groups assigned to.

The message aggregation and representation updating processes are similar to those in the city group graph. The details are shown as follows:
\[ H'_i = \text{cat} \left( X_i, H_i \right), \] (12)
\[ R_{c,i} = \left\{ (H'_i, H'_n, E_{n,i}) \right\}_{n \in N(i)}, \] (13)
\[ r_{c,i} \leftarrow \rho_c \left( R_{c,i} \right), \] (14)
\[ H''_i \leftarrow \phi_c \left( H'_i, r_{c,i} \right), \] (15)
where \( H'_i \) is the fused representations of city \( c_i \) containing the information of local features and city groups, \( R_{c,i} \) is the set containing all the messages passed to city \( c_i \) from its neighbors, which would be transformed to vector \( r_{c,i} \) later, and \( H''_i \) is the updated city representation of city \( c_i \) based on \( r_{c,i} \). \( \text{cat} \) represents a concatenation operation. \( \rho_c, \phi_c \) are transformation functions implemented by MLPs, and both of them are shared for each city.
3.7 Forecasting and Learning

GAGNN leverages an encoder-decoder architecture, and generates AQI forecasting results in the decoder. The decoder accepts the outputs generated by the encoder, i.e., $H_i', S$ (S in the decoder does not require gradient), and R (the correlations would not be re-encoded). The calculation steps in the decoder are similar to the encoder, with the difference that the inputs to the decoder are $H_i''$ rather than the historical observation sequence, thus the self-attention network is omitted.

After the calculation process in the decoder, we get the final city representations $X_{\text{output}}$, and further forecast the AQI values of all cities:

$$\hat{I}_{t+t+\tau}^i = \text{for} \left(X_{t_{\text{output}}}^i\right),$$

where for is a forecasting function implemented by an MLP.

We use MAE to evaluate the error between true AQI values and forecasted AQI values, and the loss function is defined as follows:

$$L(\theta) = \frac{1}{{N_{\text{city}} \times \tau_{\text{out}}}} \sum_{i=1}^{N_{\text{city}}} \sum_{k=1}^{\tau_{\text{out}}}{|I_{t+k}^i - \hat{I}_{t+k}^i|},$$

where $\hat{I}_{t+k}^i$ denotes the forecasted AQI value of city $c_i$ at time slot $t + k$, $I_{t+k}^i$ denotes the true AQI value of city $c_i$ at time slot $t + k$, and $\theta$ denotes the learnable parameters in GAGNN.

4 EXPERIMENTS

4.1 Datasets

We evaluate the performance of GAGNN on two real-world nationwide city air quality datasets, including the China dataset and the US dataset.

The China dataset contains the AQI data and weather data of 209 cities, collected from January 1, 2017 to April 30, 2019. The details of these data are described as follows:

1. AQI data, including AQI values and the locations of cities, are collected from National Urban Air Quality Real-time Release Platform.\(^1\) We collect AQI values at 1-hour granularity.

2. Weather data are collected from Envicloud,\(^2\) a data service provider. We collect weather data at 1-hour granularity.

The geographical locations of all cities of the China dataset are shown in Figure 4, identified by black dots on the map. The China dataset does not include cities with significant AQI or weather data missing. The few remaining missing values in the dataset are filled in by the linear interpolation method. Sliding windows (step = 1 hour) are used to generate samples, and we finally get 20,370 samples from 209 cities.

The US dataset consists of the PM2.5 data and weather data of 175 counties from January 1, 2020 to December 31, 2021. PM2.5 is one of the measures of AQI and can reflect the air quality to a large extent. The details of these data are described as follows:

1. PM2.5 data, including PM2.5 values and the locations of counties, are collected from the United States Environmental Protection Agency (USEPA) website.\(^3\) We collect the PM2.5 values at 1-hour granularity.

---

\(^1\)http://www.cnemc.cn/
\(^2\)http://www.envicloud.cn/
\(^3\)https://aqs.epa.gov/aqswb/airdata/download_files.html
(2) Weather data, consisting of wind speed, wind direction, and temperature, are also collected from the USEPA website at 1-hour granularity.

The geographical locations of all countries of the US dataset are shown in Figure 4, identified by black dots on the map. We ignore the counties with more than 20% missing values on PM2.5 and apply linear interpolation on the missing values. By sliding the window (step = 1 hour), we finally get 17,544 samples from 175 counties.

4.2 Experimental Settings

Besides MAE (Equation (17)), we also use root mean squared error (RMSE) to evaluate the model performance, which is more sensitive to the deviations:

\[
RMSE = \sqrt{\frac{1}{N_{\text{city}}} \sum_{i=1}^{N_{\text{city}}} (T_i^{t+k} - \hat{T}_i^{t+k})^2}.
\]

(18)

In the experiments, we utilize the previous 24-hour observations of all cities to forecast the next 6-hour AQI values, i.e., \(\tau_{\text{in}} = 24\) and \(\tau_{\text{out}} = 6\). We chronologically split all the samples into training data, validation data, and test data by the ratio of 0.7:0.1:0.2. Through the hyperparameter evaluation (Section 4.3), the number of city groups \(N_{\text{group}}\) is set to 15 on the China dataset and 11 on the US dataset. In addition, the batch size is set to 64, the epoch number is set to 300, the hidden size of GNNs is set to 32, the layer number of GNNs is set to 2, the edge attribute dimension of the city group graph is set to 12, and the distance threshold \(R_h\) is set to 250 KM. We choose Adam [16] as the optimization method. The learn rates for the parameters of \(S\) and other parameters are 0.05 and 0.001, respectively.

We implement our model with PyTorch [29] and construct GNNs with PyTorch Geometric library [9]. The source code and datasets are released on GitHub.\(^4\) We implement training, validation, and test tasks on a server with one GPU (NVIDIA RTX 2080Ti). All experiments were repeated five times under the same settings to avoid contingency and take the average of five experimental results to evaluate the performance.

4.3 Hyperparameter Evaluation

The number of city groups \(N_{\text{group}}\) is a hyperparameter that needs to be set in advance. Based on the process introduced in Section 3.4, a differentiable grouping network would assign all cities to \(N_{\text{group}}\) city groups.

\(^4\)https://github.com/Friger/GAGNN
We evaluate the effects of $N_{\text{group}}$ with the average MAE on validation data on the China dataset and the US dataset. Figure 5 shows the evaluation results of different $N_{\text{group}}$ values varying from 10 to 18 (CN), and 8 to 16 (US).

The evaluation results of the China dataset indicate that GAGNN performs optimally when $N_{\text{group}}$ equals 15. As $N_{\text{group}}$ increases, MAE decreases first and then increases. When $N_{\text{group}}$ is too small, the dependencies among cities cannot be fully exploited, while when $N_{\text{group}}$ is too large, the distribution of city grouping results will be more dispersed, increasing the difficulty of model training. The situation of GAGNN on the US dataset is similar to that on the China dataset, and the optimal value of $N_{\text{group}}$ is 11.

K-means algorithm can be considered as a graph pooling method based on the geographical distribution of cities. In Figure 6, we give the grouping visualization on the China dataset ($N_{\text{group}} = 15$ for GAGNN and $k = 15$ for K-means), where different city groups are distinguished by different colors, and there is no correspondence between the city groups with the same color. Specifically, we mark cities by the city group with the highest probability in $S$ for GAGNN.

It can be seen that differentiable grouping network can discover some latent dependencies among cities that K-means algorithm is unable to detect. For example, due to the sea breeze effect [28], cities distributed in a strip distribution along the southern coast have strong correlations,

![Fig. 5. Evaluation results of different $N_{\text{group}}$ on the China dataset (left) and the US dataset (right).](image)

![Fig. 6. Results of city grouping on the China dataset.](image)
Table 2. Results of Model Component Evaluation on the China Dataset

| Model                          | Metric | 1h    | 2h    | 3h    | 4h    | 5h    | 6h    |
|-------------------------------|--------|-------|-------|-------|-------|-------|-------|
| GAGNN                         | MAE    | 5.56  | 8.59  | 10.80 | 12.52 | 13.91 | 15.10 |
|                               | RMSE   | 10.81 | 16.17 | 19.84 | 22.51 | 24.63 | 26.37 |
| G with LSTM                   | MAE    | 5.86  | 9.01  | 11.34 | 13.12 | 14.57 | 15.81 |
|                               | RMSE   | 11.30 | 16.83 | 20.74 | 23.53 | 25.70 | 27.43 |
| G with K-means                | MAE    | 5.91  | 9.15  | 11.53 | 13.55 | 15.02 | 16.10 |
|                               | RMSE   | 11.67 | 17.15 | 20.94 | 23.94 | 26.36 | 27.94 |
| G with DG                     | MAE    | 5.76  | 8.80  | 11.04 | 12.77 | 14.16 | 15.33 |
|                               | RMSE   | 11.11 | 16.44 | 20.13 | 22.79 | 24.86 | 26.53 |
| G w/o CE-i                    | MAE    | 5.74  | 8.78  | 11.05 | 12.84 | 14.29 | 15.54 |
|                               | RMSE   | 11.16 | 16.52 | 20.29 | 23.07 | 25.29 | 27.13 |
| G w/o CE-ii                   | MAE    | 5.82  | 8.90  | 11.29 | 13.18 | 14.71 | 16.05 |
|                               | RMSE   | 11.09 | 16.58 | 20.61 | 23.69 | 26.15 | 28.18 |
| G w/o loc                     | MAE    | 5.69  | 8.72  | 10.96 | 12.70 | 14.12 | 15.37 |
|                               | RMSE   | 11.05 | 16.35 | 20.06 | 22.79 | 24.98 | 27.10 |

while K-means algorithm cannot capture this effect, and southern coastal cities are assigned to different city groups.

### 4.4 Model Component Evaluation

To verify the effectiveness of the components introduced in GAGNN, one of these components was removed or modified at a time in model component evaluation. To ensure fairness, all variant models follow the same experimental settings as GAGNN. We compare GAGNN with the following variants:

- **GAGNN with LSTM (G with LSTM):** G with LSTM replaces the self-attention network with a LSTM to extract the features of the historical observation sequence.
- **GAGNN with K-means (G with K-means):** G with K-means uses K-means algorithm instead of a learning method to obtain the mapping relationships between cities and city groups.
- **GAGNN with dynamic grouping (G with DG):** G with DG employs a GNN to learn an assignment matrix for dynamic grouping, which is conditioned on inputs. Inspired by DiffPool [45], the assignment matrix is obtained by \( S = GNN(X, A) = \sigma(AX\theta + b) \), where \( X \) is the output of the self-attention network and \( A \) is the adjacent matrix constructed based on geometric distances between cities, \( \theta \) and \( b \) are the parameters of GNN, and \( \sigma \) is the activation function. Note that the number of hidden dimensions of the output of GNN is regarded as the number of city groups.
- **GAGNN without the group correlation encoding module (G w/o CE):** (i) G w/o CE-i removes the group correlation encoding module in the city group graph. We construct the city group graph as a fully connected graph without edge attributes. (ii) G w/o CE-ii replaces the attributes of the edges between city groups with learnable parameters.
- **GAGNN without location (G w/o loc):** G w/o loc removes the spatial features, i.e., the geographic locations of cities, introduced in the calculation of city group representations.

The performances of GAGNN and its variants on the China dataset are given in Table 2, and the following tendencies can be discovered:

1. **GAGNN outperforms G with LSTM on all metrics.** The result indicates that self-attention network is a better practice to extract sequence features, as self-attention network can model the point-wise correlations of elements in the sequence.
(2) GAGNN outperforms G with K-means on all metrics. The result indicates that the grouping results obtained by K-means cannot fully discover the latent dependencies among cities. GAGNN introduces a learning method to group cities to discover the latent dependencies among cities.

(3) GAGNN outperforms G with DG on all metrics. The result indicates that the dynamic grouping results conditioned on inputs cannot group cities effectively. One of the possible reasons is that the presence of noise in air quality data makes grouping based on such data vulnerable to unstable hierarchical structures, which adversely affects the forecasting results.

(4) GAGNN outperforms G w/o CE-i and G w/o CE-ii on all metrics. The result indicates that the group correlation encoding module introduced in GAGNN can better capture the dependencies between city groups.

(5) GAGNN outperforms G w/o loc on all metrics. The result indicates that introducing the geographic locations of cities as spatial features in city grouping can capture the latent dependencies between adjacent cities.

4.5 Comparison with other Forecasting Models

To further verify the effectiveness of our model, we compare GAGNN with existing forecasting models. We selected the following types of models for comparison: classical regression models (LSTM, XGBoost), flatten structure forecasting models (FGA, GC-LSTM, Graph WaveNet, MegaCRN, AirFormer), an architecture-enhanced GNN model (DeeperGCN), and hierarchical structure forecasting models (ST-UNet, SHARE, HGCN).

**Classical regression models:**

LSTM: LSTM \cite{14} introduces the gating mechanism based on RNN to relieve the gradient problem. We use the historical observation sequences as the LSTM input and forecast AQI values. LSTMs for all cities share the parameters. After optimization, the hidden unit of LSTM is set to 32.

XGBoost: XGBoost \cite{7} is an engineered implementation of GBRT, which introduces some strategies to support parallel computation and outlier handling. A separate XGBoost model is built to forecast each time slot for each city. We use the grid search method to optimize the hyperparameters.

**Flatten structure forecasting models:**

Flatten GAGNN (FGA): FGA removes the hierarchical structure in GAGNN. Other experiment settings follow the original model.

GC-LSTM: GC-LSTM \cite{30} constructs a flatten city graph based on the geographical distribution of cities and introduces GCN and LSTM to capture spatial and temporal dependencies, respectively. After optimization, the hidden unit of LSTM is set to 64, and the output dimension of GCN is set to 32.

Graph WaveNet (GWNet): GWNet \cite{40} constructs two flatten city graphs, one based on the geographical distribution of cities and the other based on the node embedding dictionaries of cities. GWNet utilizes GCN and CNN to capture spatial and temporal dependencies, respectively. After optimization, the hidden dimension of GWNet is set to 32, and the number of stacked GWNet is set to 2.

MegaCRN: MegaCRN \cite{15} constructs one flatten city graph based on the node embedding dictionaries of cities. MegaCRN introduces GCN and GRU to capture spatial and temporal dependencies, respectively. After optimization, the hidden dimension of MegaCRN is set to 64, the number of node embedding is set to 20, and the number of GRU layers is set to 1.

AirFormer: AirFormer \cite{24} constructs a flatten nationwide city graph based on the domain knowledge of air pollution dispersion and dartboard projection. It models spatial relations and temporal dependencies through GCN and Attention at a deterministic stage and captures the
### Table 3. Results of Different Forecasting Models

| Model       | Metric | China dataset | US dataset |
|-------------|--------|---------------|------------|
|             |        | 1h 2h 3h 4h 5h 6h | 1h 2h 3h 4h 5h 6h |
| GAGNN       | MAE    | 5.56 10.80 12.52 13.91 15.10 | 1.67 2.14 2.43 2.69 2.85 3.03 |
|             | RMSE   | 10.81 16.17 22.51 24.63 26.37 | 3.44 4.49 5.15 5.61 5.98 6.28 |
| LSTM        | MAE    | 6.50 10.26 13.18 15.52 17.40 18.91 | 1.74 2.22 2.52 2.76 2.94 3.10 |
|             | RMSE   | 13.85 19.26 23.52 26.83 29.46 31.55 | 3.71 4.60 5.22 5.68 6.05 6.34 |
| XGBoost     | MAE    | 6.85 10.89 13.99 16.27 18.14 19.56 | 1.76 2.32 2.72 3.04 3.32 3.55 |
|             | RMSE   | 14.25 19.80 24.72 28.14 30.63 33.44 | 3.90 4.89 5.62 6.19 6.67 7.05 |
| FGA         | MAE    | 5.87 9.14 11.71 13.75 15.42 16.80 | 1.84 2.27 2.58 2.85 3.07 3.26 |
|             | RMSE   | 11.91 17.01 21.05 24.09 26.55 28.52 | 3.74 4.67 5.32 5.84 6.25 6.58 |
| GC-LSTM     | MAE    | 5.95 9.16 11.58 13.46 15.00 16.31 | 1.70 2.19 2.50 2.74 2.92 3.08 |
|             | RMSE   | 11.36 17.01 21.05 24.09 26.55 28.52 | 3.79 4.66 5.32 5.84 6.25 6.58 |
| GWNet       | MAE    | 5.76 9.64 12.79 15.30 17.28 18.81 | 1.76 2.34 2.72 3.03 3.31 3.53 |
|             | RMSE   | 11.27 17.57 22.31 25.75 28.52 30.48 | 3.59 4.71 5.48 6.08 6.56 6.93 |
| MegaCRN     | MAE    | **5.38** 8.76 10.80 12.73 14.46 16.03 | **1.66** 2.20 2.55 2.81 3.03 3.21 |
|             | RMSE   | **10.64** 16.46 19.92 22.82 25.45 27.60 | **3.41** 4.52 5.24 5.77 6.18 6.50 |
| AirFormer   | MAE    | 5.95 9.31 11.87 13.87 15.52 16.95 | 1.74 2.25 2.57 2.82 3.00 3.16 |
|             | RMSE   | 11.49 17.23 21.32 24.31 26.72 28.71 | 3.54 4.57 5.23 5.71 6.07 6.35 |
| DeeperGCN   | MAE    | 6.54 9.74 11.77 13.40 15.29 16.41 | 1.78 2.24 2.53 2.76 2.94 3.10 |
|             | RMSE   | 13.67 18.93 21.14 23.83 26.25 28.02 | 4.74 5.24 5.67 6.02 6.31 6.55 |
| ST-UNet     | MAE    | 5.95 9.30 11.58 13.38 14.82 16.02 | 1.73 2.21 2.53 2.77 2.95 3.12 |
|             | RMSE   | 11.74 18.01 21.34 23.90 25.94 27.64 | 3.97 4.78 5.27 6.09 6.43 6.76 |
| SHARE       | MAE    | 5.84 9.07 11.49 13.35 14.74 15.79 | 1.81 2.32 2.65 2.90 3.10 3.26 |
|             | RMSE   | 11.27 16.84 20.77 23.60 25.80 27.38 | 4.29 4.98 5.58 6.04 6.40 6.69 |
| HGCN        | MAE    | 5.70 9.09 11.73 13.84 15.55 16.95 | 1.70 2.20 2.52 2.71 2.90 3.06 |
|             | RMSE   | 11.18 17.09 21.33 24.52 26.99 28.94 | 3.43 4.48 5.17 5.66 6.01 6.33 |

uncertainty of the input through VAE at a stochastic stage. After optimization, the hidden dimension of AirFormer is set to 32, and the number of stacked AirFormer blocks is set to 4.

**Architecture enhanced GNN model:**

DeeperGCN: DeeperGCN [20] introduces a pre-activation architecture and a message normalization method to relieve the over-smoothing issue when GNNs go deeper. After optimization, the layer number of DeeperGCN is set to 8.

**Hierarchical structure forecasting models:**

ST-UNet: ST-UNet [46] adopts a heuristic method to construct coarsened graphs and adopts pooling and unpooling strategies to implement inter-level interactions. In addition, ST-UNet introduces dilated GRUs to capture the multilevel temporal dependencies of sequences. After optimization, the hidden unit of GRU is set to 32, and the output dimension of GCN is set to 32.

SHARE: SHARE [49] is a semi-supervised spatial-temporal forecasting model, which introduces a soft pooling method to construct a coarsened graph and concatenates the entity representations in multi-level graphs to obtain forecast results. The correlations between coarsened nodes are defined by the correlations between the nodes assigned to them. We remove the semi-supervised learning part of SHARE and remain the rest settings.

HGCN: HGCN [11] is a hierarchical graph convolution network that considers both the road segment and region features of the traffic system for traffic forecasting. It utilizes multiple GCNs on road-level and region-level graphs and instigates the extracted features on both levels by dynamic transfer blocks. After optimization, the number of ST-block is set to 2.

The performances of GAGNN and other forecasting models are given in Table 3 (bold indicates the best and underline indicates the second best), and the following tendencies can be discovered from Table 3:

ACM Transactions on Knowledge Discovery from Data, Vol. 18, No. 3, Article 55. Pub. date: December 2023.
Table 4. Results of Long-term Forecasting on the China Dataset

| Horizon | Metric | SHARE | AirFormer | MegaCRN | HGCN | GAGNN |
|---------|--------|-------|-----------|---------|------|-------|
| 9h      | MAE    | 21.72 | 19.83     | 18.47   | 18.88| **17.95** |
|         | RMSE   | 35.50 | 32.96     | 31.16   | 31.95| **30.56** |
| 12h     | MAE    | 23.69 | 21.82     | 21.03   | 20.84| **20.03** |
|         | RMSE   | 38.57 | 35.72     | 34.39   | 34.32| **33.20** |

(1) Classical regression models, e.g., LSTM, show competitive performance on the US dataset due to the stability of the observed data over time. LSTM possesses the capacity to effectively encode and retain historical information, thereby enabling accurate predictions.

(2) Forecasting models that consider the structures between cities, including flatten structures and hierarchical structures, outperform classical regression models. Classical regression models tend to model the relationship between single city features and air quality, neglecting the reality that AQI values of cities can impact each other due to atmospheric motion.

(3) Flatten structure forecasting models, e.g., GC-LSTM and MegaCRN, have competitive performance in short-term forecasting, specifically for 1-hour predictions. One possible explanation for this is that in the short term, AQI values of cities are influenced more by those of neighboring cities. Since flatten structure models focus more on local features, they are better equipped to handle messages passed from nearby cities, leading to more effective short-term forecasting.

(4) Hierarchical structure forecasting models, e.g., SHARE, HGCN, and GAGNN, outperform other models in long-term forecasting, specifically for 6-hour predictions. This may be due to the fact that in the long-term, AQI values of cities are influenced not only by those of nearby cities but also those of distant cities, as a result of atmospheric motion. Hierarchical structures can expand the receptive fields of cities without raising the issue of over-smoothing, enabling them to effectively handle messages passed from distant cities in long-term forecasting. More experiments and discussions regarding long-term forecasting are presented in Section 4.6.

(5) GAGNN outperforms other hierarchical structure forecasting baselines in 22 out of 24 cases. Specifically, GAGNN outperforms ST-Unet and HGCN in 24 and 22 cases, respectively. The major difference between GAGNN and the two baselines is that GAGNN constructs city groups by learning an assignment matrix end-to-end, while ST-Unet and HGCN construct city groups by pre-defined methods, i.e., a heuristic grouping method and spectral clustering, respectively. The results indicate that a learning grouping method can effectively discover the latent dependencies between cities. In addition, GAGNN outperforms SHARE in all cases. The major difference between GAGNN and SHARE is that GAGNN introduces a group correlation encoding module to learn the correlations between city groups, while in SHARE they are defined by the geographical distribution of cities assigned to. The group correlation encoding module encodes the edge attributes between city group nodes based on the city group representations and time information, which can effectively capture the dependencies between city groups.

4.6 Long-term Forecasting Evaluation

To further investigate the long-term forecasting ability of GAGNN, we set the forecasting horizons to 9-hour and 12-hour, and evaluate GAGNN along with competitive baselines from Table 3, including SHARE, HGCN, AirFormer, and MegaCRN. The performance results are presented in Table 4, and the following observations can be made:
Table 5. Efficiency Study on the China Dataset

| Model    | # parameter | Training Time (s/epoch) | Inference Time (s) | MAE   |
|----------|-------------|-------------------------|--------------------|-------|
| SHARE    | 24168       | 31.51                   | 3.26               | 11.71 |
| AirFormer| 169371      | 79.90                   | 4.02               | 12.25 |
| MegaCRN  | 386537      | 32.20                   | 0.57               | 11.32 |
| HGCN     | 937129      | 16.71                   | 0.76               | 12.14 |
| GAGNN    | 48713       | 17.59                   | 1.43               | 11.08 |

(1) Hierarchical structure forecasting models, i.e., HGCN and GAGNN, exhibit an advantage over flatten structure forecasting models, i.e., SHARE, AirFormer, and MegaCRN. This superiority is particularly evident when the horizon is set to 12-hour. One possible explanation is that hierarchical structures might capture latent dependencies between distant cities, which have a more significant impact on long-term forecasting due to the dispersion of pollutants.

(2) GAGNN outperforms HGCN for 9-hour and 12-hour predictions. This phenomenon might be attributed to the fact that the learnable assignment matrix adopted by GAGNN effectively captures latent dependencies between cities, while the pre-defined assignment matrix adopted by HGCN cannot.

4.7 Efficiency Study

To study the efficiency of GAGNN, we compare its number of parameters, training and inference time, and average MAE on 6-hour predictions with those of the baselines. The performance results are presented in Table 5, and we observe that GAGNN is capable of efficiently forecasting nationwide city air quality. Specifically, GAGNN achieves the best average MAE with the second-smallest number of parameters. The training and inference time of GAGNN is also very competitive with the baselines, ranking second-best and third-best, respectively. Such efficiency of GAGNN may be attributed to the shared parameter design in the two message passing mechanisms in the city graph and city group graph.

4.8 Case Study

In case study, we give two examples to illustrate the superiority of GAGNN over existing models when regional pollution occurs and the correlations among cities are complex.

The first case is to forecast the AQI values of Beijing city from 18:00 November 24 to 23:00 November 24, 2018. Figure 7(a) shows the AQI values of Beijing city in the historical window (1–24 hours) and the forecasting horizon (25–30 hours). In this case, an air pollution event occurred in Beijing city and the AQI values in Beijing city increased sharply from 18:00 November 24, making...
the task of forecasting difficult. Table 6 gives the forecasting results of GAGNN and other models in this case. We can find that the forecasting errors of all models become larger than usual, as the dependencies in spatial and temporal dimensions are complicated when regional pollution occurs. Compared with other models, GAGNN achieves high accuracy in forecasting. One possible reason is that GAGNN utilizes weather features not only as node attributes, but also as edge attributes through its group correlation encoding module. As a result, GAGNN can effectively capture the spatial and latent dependencies between cities and provide better responses to air pollution events.

The second case is to forecast the AQI values of Shanghai city from 4:00 January 12 to 9:00 January 12, 2019. Figure 7(b) shows the AQI values of Shanghai city in the historical window (1–24 hours) and the forecasting horizon (25–30 hours). In this case, Shanghai city does not experience high levels of local pollution, and its air quality is impacted by imported pollution from cities in northern China during winter. Modeling the complicated spatial and latent dependencies between Shanghai and distant cities makes the task of forecasting difficult. Table 7 gives the forecasting results of GAGNN and other models for Shanghai city. Compared with other models, GAGNN achieves the highest accuracy in forecasting, and this superiority becomes particularly noticeable for long-term forecasting, e.g., 5-hour and 6-hour. These results suggest that GAGNN can capture both spatial and latent dependencies between cities, verifying the effectiveness of the constructed two graphs, i.e., a city graph and a city group graph, in GAGNN.

5 CONCLUSIONS AND FUTURE WORK

We propose GAGNN, a hierarchical model for nationwide city air quality forecasting. The model constructs a city graph and a city-group graph to model the spatial and latent dependencies between cities, respectively. We evaluate GAGNN on two real-world nationwide city air quality datasets, i.e., the China dataset and the US dataset, and obtain the following conclusions from the experiment results: (1) HGGNNs can effectively relieve the conflict between expanding receptive fields and preserving local features; (2) differentiable grouping network can effectively discover the latent dependencies between cities; (3) the group correlation encoding module can effectively capture the dependencies between city groups.

In the future, we will extend our model in the following aspects. On the one hand, the current model considers only two levels of hierarchy, i.e., cities and city groups. We will introduce a

| Model   | 1h  | 2h  | 3h  | 4h  | 5h  | 6h  |
|---------|-----|-----|-----|-----|-----|-----|
| GAGNN   | 1.79| 15.95| 35.06| 37.00| 29.08| 27.18|
| MegaCRN | 7.06| 26.96| 51.41| 48.32| 40.78| 46.21|
| AirFormer | 11.93| 32.47| 55.76| 54.46| 45.16| 50.85|
| SHARE | 10.23| 23.32| 48.93| 50.78| 49.68| 44.59|
| HGCN | 11.94| 35.41| 57.98| 56.14| 47.02| 50.89|

Table 7. MAEs of Different Models in Shanghai City

| Model   | 1h  | 2h  | 3h  | 4h  | 5h  | 6h  |
|---------|-----|-----|-----|-----|-----|-----|
| GAGNN   | 1.48| 0.60| 3.94| 5.24| 7.40| 4.08|
| MegaCRN | 1.74| 2.04| 6.30| 8.49| 11.45| 8.99|
| AirFormer | 2.47| 3.40| 7.76| 9.17| 12.80| 10.60|
| SHARE | 1.63| 2.97| 7.14| 8.92| 11.87| 10.41|
| HGCN | 3.33| 4.24| 8.24| 9.62| 11.31| 7.20|
multi-level hierarchical structure in the model to capture more complex correlations between entities. On the other hand, the current correlations between city groups obtained by a learning method do not have clear semantic information. We will introduce some constraints that allow the model to consider more semantic correlations between entities, e.g., causality.

REFERENCES

[1] 2022. Auto-STGCN: Autonomous spatial-temporal graph convolutional network search. ACM Transactions on Knowledge Discovery from Data 17, 5 (2022), 1–21.

[2] N. Kh Arystanbekova. 2004. Application of gaussian plume models for air pollution simulation at instantaneous emissions. Mathematics and Computers in Simulation 67, 4-5 (2004), 451–458.

[3] Donghui Chen, Ling Chen, Youdong Zhang, Bo Wen, and Chenghu Yang. 2022. A multiscale interactive recurrent network for time-series forecasting. IEEE Transactions on Cybernetics 52, 9 (2022), 8795–8803.

[4] Ling Chen, Donghui Chen, Zongjiang Shang, Binqing Wu, Cen Zheng, Bo Wen, and Wei Zhang. 2023. Multi-scale adaptive graph neural network for multivariate time series forecasting. IEEE Transactions on Knowledge and Data Engineering 35, 10 (2023), 10748–10761.

[5] Ling Chen, Yifang Ding, Dandan Lyu, Xiaoze Liu, and Hanyu Long. 2019. Deep multi-task learning based urban air quality index modelling. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 3, 1 (2019), 1–17.

[6] Ling Chen, Hanyu Long, Jiahui Xu, Binqing Wu, Hang Zhou, Xing Tang, and Liangying Peng. 2023. Deep citywide multisource data fusion-based air quality estimation. IEEE Transactions on Cybernetics (2023), 1–12.

[7] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785–794.

[8] Frederik Diehl, Thomas Brunner, Michael Truong Le, and Alois Knoll. 2019. Towards graph pooling by edge contraction. In Proceedings of the ICML Workshop on Learning and Reasoning with Graph-Structured Data.

[9] Matthias Fey and Jan Eric Lenssen. 2019. Fast graph representation learning with PyTorch geometric. arXiv: 1903.02428. Retrieved from http://arxiv.org/abs/1903.02428.

[10] Liang Ge, Kunyan Wu, Yi Zeng, Feng Chang, Yaqian Wang, and Siyu Li. 2021. Multi-scale spatiotemporal graph convolution network for air quality prediction. Applied Intelligence 51, 6 (2021), 3491–3505.

[11] Kan Guo, Yongli Hu, Yanfeng Sun, Sean Qian, Junbin Gao, and Baoai Yin. 2021. Hierarchical graph convolution network for traffic forecasting. In Proceedings of the 35th AAAI Conference on Artificial Intelligence, 151–159.

[12] Jindong Han, Hao Liu, Hengshu Zhu, and Hui Xiong. 2023. Kill two birds with one stone: A multi-view multi-adversarial learning approach for joint air quality and weather prediction. IEEE Transactions on Knowledge and Data Engineering 35, 11 (2023), 11515–11528.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In Proceedings of the 29th IEEE Conference on Computer Vision and Pattern Recognition, 770–778.

[14] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. Neural Computation 9, 8 (1997), 1735–1780.

[15] Renhe Jiang, Zhaonan Wang, Jiawei Yong, Puneet Jeph, Quanjun Chen, Yasumasa Kobayashi, Xuan Song, Shintaro Fukushima, and Toyotaro Suzumura. 2023. Spatio-temporal meta-graph learning for traffic forecasting. In Proceedings of the 37th AAAI Conference on Artificial Intelligence. 8078–8086.

[16] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In Proceedings of the 32nd International Conference on Learning Representations.

[17] Junhyun Lee, Inyeop Lee, and Jaewoo Kang. 2019. Self-attention graph pooling. In Proceedings of the 36th International Conference on Machine Learning, 3734–3743.

[18] Muhammad Hisyam Lee, Nur Haizum Abd Rahman, Suhartono, Mohd Talib Latif, M. Nor, and Nur Arina Bazilah Kamisan. 2012. Seasonal ARIMA for forecasting air pollution index: A case study. American Journal of Applied Sciences 9 (2012), 570–578.

[19] Guohao Li, Matthias Muller, Ali Thabet, and Bernard Ghanem. 2019. DeepGCNs: Can GCNs go as deep as CNNs?. In Proceedings of the 17th IEEE/CVF International Conference on Computer Vision, 9267–9276.

[20] Guohao Li, Chenzx Xiong, Ali K. Thabet, and Bernard Ghanem. 2020. DeeperGCNs: All you need to train deeper GCNs. arXiv:2006.07739. Retrieved from https://arxiv.org/abs/2006.07739.

[21] Mike Li, Elia Perrier, and Chang Xu. 2019. Deep hierarchical graph convolution for election prediction from geospatial census data. In Proceedings of the 33rd AAAI Conference on Artificial Intelligence. 647–654.

[22] Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper insights into graph convolutional networks for semi-supervised learning. In Proceedings of the 32nd AAAI Conference on Artificial Intelligence. 3538–3545.
[23] Yuxuan Liang, Songyu Ke, Junbo Zhang, Xiwen Yi, and Yu Zheng. 2018. GeoMAN: Multi-level attention networks for geo-sensory time series prediction. In Proceedings of the 27th International Joint Conference on Artificial Intelligence. 3428–3434.

[24] Yuxuan Liang, Yutong Xia, Songyu Ke, Yiwei Wang, Qingsong Wen, Junbo Zhang, Yu Zheng, and Roger Zimmermann. 2023. AirFormer: Predicting nationwide air quality in china with transformers. In Proceedings of the 37th AAAI Conference on Artificial Intelligence. 14329–14337.

[25] Yijun Lin, Nikhit Magi, Yu Gao, Taguang Li, Yao-Yi Chiang, Cyrus Shahabi, and José Luis Ambite. 2018. Exploiting spatiotemporal patterns for accurate air quality forecasting using deep learning. In Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems. 359–368.

[26] Yunxia Liu, Xiao Lu, Wei Peng, Chengdong Li, and Haixia Wang. 2022. Compression and regularized optimization of modules stacked residual deep fuzzy system with application to time series prediction. Information Sciences 608, C (2022), 551–577.

[27] Mingqi Lv, Yifan Li, Ling Chen, and Tieming Chen. 2019. Air quality estimation by exploiting terrain features and multi-view transfer semi-supervised regression. Information Sciences 483, C (2019), 82–95.

[28] D. K. Papanastasiou and D. Melas. 2009. Climatology and impact on air quality of sea breeze in an urban coastal environment. International Journal of Climatology: A Journal of the Royal Meteorological Society 29, 2 (2009), 305–315.

[29] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An imperative style, high-performance deep learning library. In Proceedings of the 33rd International Conference on Neural Information Processing Systems. 8026–8037.

[30] Yanlin Qi, Qi Li, Hamed Karimian, and Di Liu. 2019. A hybrid model for spatiotemporal forecasting of PM2.5 based on graph convolutional neural network and long short-term memory. Science of the Total Environment 664 (2019), 1–10.

[31] Zhongang Qi, Tianchun Wang, Guojie Song, Zhen Huang, and Jia-song Wang. 2005. Impact of building configuration on air quality in urban street canyon. Atmospheric Environment 98 (2014), 260–270.

[32] Siyuan Ren, Bin Guo, Ke Li, Qianru Wang, Qinfen Wang, and Zhiwen Yu. 2023. CoupledGT: Coupled geospatial-temporal data modeling for air quality prediction. ACM Transactions on Knowledge Discovery from Data 17, 9 (2023), 1–21.

[33] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. 2019. The truly deep graph convolutional networks for node classification. arXiv:1907.10903. Retrieved from http://arxiv.org/abs/1907.10903

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems. 5998–6008.

[35] Shuo Wang, Yanran Li, Jin Zhang, Qingye Meng, Lingwei Meng, and Fei Gao. 2020. PM2.5-GNN: A domain knowledge enhanced graph neural network for PM2.5 forecasting. In Proceedings of the 28th International Conference on Advances in Geographic Information Systems. 163–166.

[36] Wenjian Wang, Zongben Xu, and Jane Weizhen Lu. 2003. Three improved neural network models for air quality forecasting. Engineering Computations 20, 2 (2003), 192–210.

[37] Ning Wu, Xin Wayne Zhao, Jingyuan Wang, and Dayan Fan. 2020. Learning effective road network representation with hierarchical graph neural networks. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 6–14.

[38] Z. Wu, S. Fan, F. Chen, G. Long, C. C. Zhang, and P. S. Yu. 2020. A comprehensive survey on graph neural networks. IEEE Transactions on Neural Networks and Learning Systems 32, 1 (2020), 4–24.

[39] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Graph WaveNet for deep spatial-temporal graph modeling. In Proceedings of the 28th International Joint Conference on Artificial Intelligence. 1907–1913.

[40] Xiaomin Xie, Zhen Huang, and Jia-song Wang. 2005. Impact of building configuration on air quality in street canyon. Atmospheric Environment 39, 25 (2005), 4519–4530.

[41] Jiahui Xu, Ling Chen, Mingqi Lv, Chaoqun Zhan, Sanjian Chen, and Jian Chang. 2021. HighAir: A hierarchical graph neural network-based air quality forecasting method. arXiv:2101.04264. Retrieved from https://arxiv.org/abs/2101.04264

[42] Meiling Xu, Min Han, C. L. Philip Chen, and Tie Qiu. 2020. Recurrent broad learning systems for time series prediction. IEEE Transactions on Cybernetics 50, 4 (2020), 1405–1417.
55:20

[44] Xinghan Xu and Minoru Yoneda. 2021. Multitask air-quality prediction based on LSTM-autoencoder model. IEEE Transactions on Cybernetics 51, 5 (2021), 2577–2586.

[45] Rex Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L. Hamilton, and Jure Leskovec. 2018. Hierarchical graph representation learning with differentiable pooling. In Proceedings of the 32nd International Conference on Neural Information Processing Systems. 4805–4815.

[46] Bing Yu, Haoteng Yin, and Zhanxing Zhu. 2019. ST-UNet: A spatio-temporal u-network for graph-structured time series modeling. arXiv:1903.05631. Retrieved from http://arxiv.org/abs/1903.05631

[47] Ruiyun Yu, Yu Yang, Leyou Yang, Guangjie Han, and Ogutu Ann Move. 2016. RAQ—a random forest approach for predicting air quality in urban sensing systems. Sensors 16, 1 (2016), 86.

[48] Xiang Yu, Dongmei Zhang, Tianqing Zhu, and Xinwei Jiang. 2022. Novel hybrid multi-head self-attention and multi-fractal algorithm for non-stationary time series prediction. Information Sciences 613 (2022), 541–555.

[49] Weijia Zhang, Hao Liu, Yanchi Liu, Jingbo Zhou, and Hui Xiong. 2020. Semi-supervised hierarchical recurrent graph neural network for city-wide parking availability prediction. In Proceedings of the 34th AAAI Conference on Artificial Intelligence. 1186–1193.

Received 27 February 2023; revised 22 August 2023; accepted 22 October 2023