Noname manuscript No.
(will be inserted by the editor)

Dual attentive graph neural network for metro passenger flow prediction

Yuhuan Lu[1], Hongliang Ding[2], Shiqian Ji[3], N.N. Sze[4], Zhaocheng He[5], ✉

Received: date / Accepted: date

✉hezhch@mail.sysu.edu.cn
· 1. School of Intelligent Systems Engineering, Sun Yat-Sen University, Guangzhou, China
· 2. Department of Civil and Environmental Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
· 3. School of Intelligent Systems Engineering, Sun Yat-Sen University, Guangzhou, China
· 4. Department of Civil and Environmental Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
· 5. School of Intelligent Systems Engineering, Sun Yat-Sen University, Guangzhou, China

Abstract Metro system has been increasingly recognized as a backbone of urban transportation system in many cities around the world. To improve the demand management and operation efficiency, it is crucial to have accurate prediction of real-time metro passenger flow. However, the forecast performance is often subject to the complex spatial and temporal distributions of the metro passenger flow data. To this end, we developed a novel Dual Attentive Graph Neural Network (DAGNN) that can effectively predict the distribution of metro traffic flow considering the spatial and temporal influences. Specifically, two directed complete metro graphs (i.e., inbound and outbound graphs) and the weighted matrix of them are proposed to characterize the inbound (entering the system) and outbound (leaving the system) passenger flow respectively. The weighted matrix of inbound graph is estimated based on the historical origin-destination (OD) demand and that of the outbound graph is estimated based on the similarity metrics between every two stations. Moreover, to capture the dependencies between inbound and outbound flows, multi-layer Graph Spatial Attention Networks (GSANs) that incorporate the spatial context are applied to exploit the dynamic inter-station correlations. Then, the acquired dependencies feature integrated with external factors, such as weather conditions, are filtered by temporal attention and fed into a sequence decoder to produce short-term and long-term passenger flow predictions. Finally, a series experiments are conducted based on a comprehensive empirical dataset. Findings indicated that the proposed model does not only well predict the metro passenger flow, but also effectively detect the emergencies and incidents of metro system.

Keywords Metro system · Passenger flow prediction · Graph neural network · Attention mechanism

1 Introduction

To relieve the traffic congestion and improve the air quality, public transport, i.e. bus and metro, has been introduced and developed in many cities. To encourage the public transport use, policy initiatives including park, ride and bike sharing are implemented [1–3]. Specifically, metro system has become the backbone of urban transportation in many cities because of its high capacity and reliability. Also, (underground) metro system does not occupy any road space. However, with the popularity of metro system, the system operation and service quality is often of concern, especially for the high passenger flow in the peak periods [4]. Thus, it is essential to find a method that can accurately predict the metro passenger flow. It is helpful for passenger evacuation, vehicle scheduling, fare policy and demand management.

In the previous studies, numerous approaches have been proposed to predict traffic flow [5–8]. For instance, Ahmed and Cook [9] first introduced autoregressive integrated moving average (ARIMA) model to forecast the short-term traffic flow. Afterwards, ARIMA and its variants [10–12] were widely applied to the freeway traffic flow prediction. ARIMA and its variants outperformed other traditional linear regression models [13]. Moreover, with the advancement of information and communication technologies, comprehensive data is available and provides the opportunity to further investigate the inherent flow dynamics of the transportation systems. Compared with traditional prediction models, machine learning methods are more efficient for the processing of such high-dimensional and nonlinear data. Several machine learning methods approaches including decision tree [14, 15], support vector machine [16, 17], Bayesian network [18–20] and neural network [21–23], have been developed to predict traffic flow. Furthermore, deep learning methods were applied in the prediction models to explore the complex correlation features of the traffic flow data [24–27]. For example, Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) were developed to capture the spatial and temporary correlations of traffic flows. Moreover, some researchers attempted to explore the deep-seated spatial interactions among traffic sensors based on the physical topologies of road networks using some advanced methods, such as Graph Neural Networks (GNNs) [28–32].

Although the above methods are available for the traffic flow prediction, to generalize the prediction methods for metro passenger flow prediction, it is still a challenging task because of the following two considerations:

- **Inbound dependencies**: The current outbound (leaving the system) flow of a station is affected by previous inbound (entering the system) flows of one or several other stations. Notably, such correlations change over time. For instance, in the morning peak hours, the outbound flow of a transfer station is strongly correlated with the inbound flows of one or several stations far away. However, in the evening peak hours, the outbound flow of a transfer station is strongly correlated with the inbound flows of the stations nearby.
- **Outbound dependencies**: The outbound flow of a station is strongly correlated with that of the stations that share the same land use type (e.g. busi-

ness districts) and functional type (e.g. transfer stations).

To capture the inbound and outbound dependencies, a novel Dual Attentive Graph Neural Network (DAGNN) was proposed in this study. Firstly, metro stations are denoted as nodes and two directed complete metro graphs, the weighted matrix of inbound and outbound graph, are proposed to describe the inbound and outbound flow profiles respectively. Specifically, the weighted matrix of inbound graph is estimated based on the historical metro passenger flows and that of the outbound graph is derived by the similarity metrics between two stations. Then the above two graphs are incorporated into the multi-layer Graph Spatial Attention Networks to accommodate the inbound and outbound dependencies, and the dependencies embeddings would be integrated with external factors (e.g. weathers). Moreover, we developed a temporal attention to adaptively select the most relevant embeddings. Finally, the filtered embeddings are fed into a stacked RNN network to forecast the outbound passenger flow of more than one future time intervals.

To sum up, the main contributions of this work are three-fold:

– We develop a novel Graph Spatial Attention Network (GSAN) to capture both inbound and outbound dependencies. Specifically, we integrate spatial attention into a generic GNN to dynamically capture inter-station correlations. Graph spatial attention layer can be stacked like CNNs to facilitate the representation learning of inbound and outbound dependencies.
– We design a temporal attention based decoder to produce both short-term and long-term predictions. At each prediction step, the dependencies embeddings concatenated with external factors are filtered by temporal attention and then the selected embeddings are fed into a stacked LSTM network to forecast the outbound passenger flow at the next time interval.
– Extensive experiments on a real-world benchmark consisting of Guangzhou metro passenger flow and meteorological data justifies that the proposed model outperforms the state-of-the-art methods.

The remainder of this paper is organized as follows. Section 2 describes the formulation of LSTM network, and highlights the key issues to be resolved. The formulation of proposed DAGNN is described in Section 3. Then, the analysis results and illustrative example are presented in Section 4. Lastly, the concluding remarks and future research directions are given in Section 5.

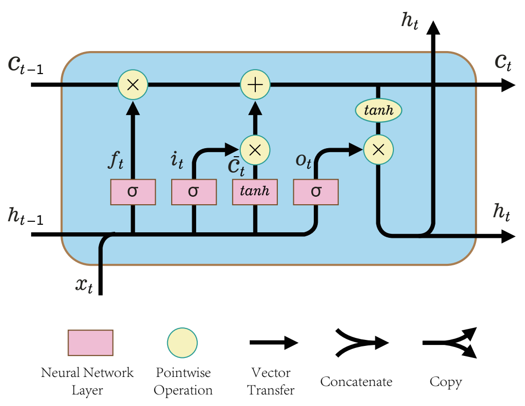


Fig. 1 Internal structure of a LSTM cell.

2 Preliminaries

2.1 Long short term memory network

The Long Short Term Memory (LSTM) network [33] is a variant of traditional (RNN) [34], which has been proved to be capable of alleviating the vanishing gradient problem [35]. As shown in Fig.1, different from basic RNNs, a LSTM cell contains several gates, which can selectively retain information. These small modifications enable the LSTM network to capture long-term dependencies, and thereby eliminate the effect of vanishing gradient. Specifically, after integrating the current input $\mathbf{x}_t$ and previous hidden state $\mathbf{h}_{t-1}$, the complete workflow of a LSTM cell can be represented as follows:

$$f_t = \sigma\left(\mathbf{W}_f\left[\mathbf{h}_{t-1}, \mathbf{x}_t\right] + b_f\right) \quad (1)$$
$$i_t = \sigma\left(\mathbf{W}_i\left[\mathbf{h}_{t-1}, \mathbf{x}_t\right] + b_i\right) \quad (2)$$
$$o_t = \sigma\left(\mathbf{W}_o\left[\mathbf{h}_{t-1}, \mathbf{x}_t\right] + b_o\right) \quad (3)$$
$$\mathbf{c}_t = f_t \odot \mathbf{c}_{t-1} + i_t \odot \tanh\left(\mathbf{W}_c\left[\mathbf{h}_{t-1}, \mathbf{x}_t\right] + b_c\right) \quad (4)$$
$$\mathbf{h}_t = o_t \odot \tanh\left(\mathbf{c}_t\right) \quad (5)$$

where $\{\mathbf{W}_f, \mathbf{W}_i, \mathbf{W}_o\}$ and $\{b_f, b_i, b_o\}$ are trainable parameters. $\{\mathbf{W}_f, \mathbf{W}_i, \mathbf{W}_o\}$ denote the weighted matrices of the forget gate, the input gate and the output gate respectively while $\{b_f, b_i, b_o\}$ denote their bias vectors correspondingly. The operator $\odot$ refers to the Hadamard product. $\sigma(\cdot)$ and $\tanh(\cdot)$ are two widely used activation functions, which can be defined as:

$$\sigma(x) = \frac{1}{1+e^{-x}} \quad (6)$$
$$\tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \quad (7)$$

2.2 Problem statement

A transaction in metro system includes the card id, station name, timestamp, transaction type (either inbound or outbound) and card type. Given a fixed time span (e.g., 10 mins), we aggregate the transactions and denote by $x_{s,i}^t$, $x_{s,o}^t$ the total inbound and outbound passengers at station s during t-th time interval, respectively. Suppose the set of stations is $S = \{s_1, s_2, \ldots, s_N\}$, where N denotes the total number of stations. We use $\mathbf{x}_I^t = \left(x_{s_1,I}^t, x_{s_2,I}^t, \ldots, x_{s_N,I}^t\right)^{\mathrm{T}} \in \mathbb{R}^N$ and $\mathbf{x}_O^t = \left(x_{s_1,O}^t, x_{s_2,O}^t, \ldots, x_{s_N,O}^t\right)^{\mathrm{T}} \in \mathbb{R}^N$ to denote the inbound and outbound series of all stations during t-th interval respectively.

With the aforementioned notations, the objective of this paper is to use the historical passenger flow data $\left\{\left(\mathbf{x}_I^t, \mathbf{x}_O^t\right), \left(\mathbf{x}_I^{t-1}, \mathbf{x}_O^{t-1}\right), \left(\mathbf{x}_I^{t-2}, \mathbf{x}_O^{t-2}\right), \ldots\right\}$ and external factors, to forecast the outbound passenger flow over next τ time intervals $\hat{\mathbf{X}}_O = \left(\hat{\mathbf{x}}_O^{t+1}, \hat{\mathbf{x}}_O^{t+2}, \ldots, \hat{\mathbf{x}}_O^{t+\tau}\right) \in \mathbb{R}^{N\times\tau}$.

3 Methodology

3.1 Model overview

There are two issues in the conventional metro passenger flow prediction models: (1) effective modeling of the non-Euclidean structures of metro systems; and (2) excessive learning of local spatial dependencies. In this study, a novel metro passenger flow prediction model - Dual Attentive Graph Neural Network (DAGNN) - is proposed. In particular, the Graph Neural Network (GNN) method is applied to capture the trends of metro passenger flow pattern. Then, two metro network diagrams are constructed to model the inter-station passenger flow dynamics globally.

Fig.2 presents the framework of Dual Attentive Graph Neural Network (DAGNN). Following the novel graph-to-sequence architecture [36], we employ two graph neural networks to encode inbound and outbound dependencies respectively and then design a sequence decoder to forecast the future outbound passenger flows. More specifically, our DAGNN consists of following two parts: 1) In the encoder, we first organize two metro graphs representing the characteristics of inbound and outbound flows respectively. Then we develop Graph Spatial Attention Networks (GSANs) to capture the dependencies from above metro graphs. Finally, each dependencies embedding is combined with the features of external factors as the input to decoder. 2) In the decoder, an attention mechanism is designed to adaptively select the relevant embeddings and a stacked LSTM network is adopted to produce both short-term and long-term predictions.

3.2 Model input

Before elaborating on the proposed model, we first define the input with regard to inbound and outbound passenger flows. Assume that the current time interval is t and the size of inbound time window is T_I. The inbound flow segment of input can be represented as follows:

$$\mathbf{X}_I^t = \left(\mathbf{x}_I^{t-T_I+1}, \mathbf{x}_I^{t-T_I+2}, \ldots, \mathbf{x}_I^t\right) \in \mathbb{R}^{N\times T_I} \tag{8}$$

Likewise, we can determine the outbound flow segments of input. Notably, the outbound passenger flow at a certain time interval of a day shares a high similarity with those of previous days or weeks. Assume that the time interval to forecast is $t+\epsilon$ $(1 \le \epsilon \le \tau)$. Considering the recent time intervals, daily and weekly periodicity, we intercept three outbound flow segments of length $T_{O,r}$, $T_{O,d}$ and $T_{O,w}$:

$$\mathbf{X}_{O,r}^t = \left(\mathbf{x}_O^{t+\epsilon-T_{O,r}}, \ldots, \mathbf{x}_O^t, \hat{\mathbf{x}}_O^{t+1}, \ldots, \hat{\mathbf{x}}_O^{t+\epsilon-1}\right) \in \mathbb{R}^{N\times T_{O,r}} \tag{9}$$

$$\mathbf{X}_{O,d}^t = \left(\mathbf{x}_O^{t+\epsilon-T_{O,d}\cdot d}, \ldots, \mathbf{x}_O^{t+\epsilon-2d}, \mathbf{x}_O^{t+\epsilon-d}\right) \in \mathbb{R}^{N\times T_{O,d}} \tag{10}$$

$$\mathbf{X}_{O,w}^t = \left(\mathbf{x}_O^{t+\epsilon-T_{O,w}\cdot w}, \ldots, \mathbf{x}_O^{t+\epsilon-2w}, \mathbf{x}_O^{t+\epsilon-w}\right) \in \mathbb{R}^{N\times T_{O,w}} \tag{11}$$

where $\hat{\mathbf{x}}_O^{t+1}$ denotes the predicted value at time interval $t+1$. d and w refer to the one-day and one-week period respectively. Combining these three flow segments, we derive the complete outbound flow segment of input:

$$\mathbf{X}_O^t = \left(\mathbf{X}_{O,r}^t, \mathbf{X}_{O,d}^t, \mathbf{X}_{O,w}^t\right) \in \mathbb{R}^{N\times T_O} \tag{12}$$

where $T_O = T_{O,r} + T_{O,d} + T_{O,w}$.

3.3 Metro graph construction

In Graph Neural Networks (GNNs), the feature representation of a node is determined by its neighbouring nodes. This property makes GNNs naturally suitable for the task of network-wise flow prediction. Thus in previous works concerning traffic estimation, urban road network topology is directly converted into the traffic graph, where nodes refer to the intersections and edges refer to the road segments. However, in the case of the metro system, there are direct interactions (passengers shuttle) between any two stations even though

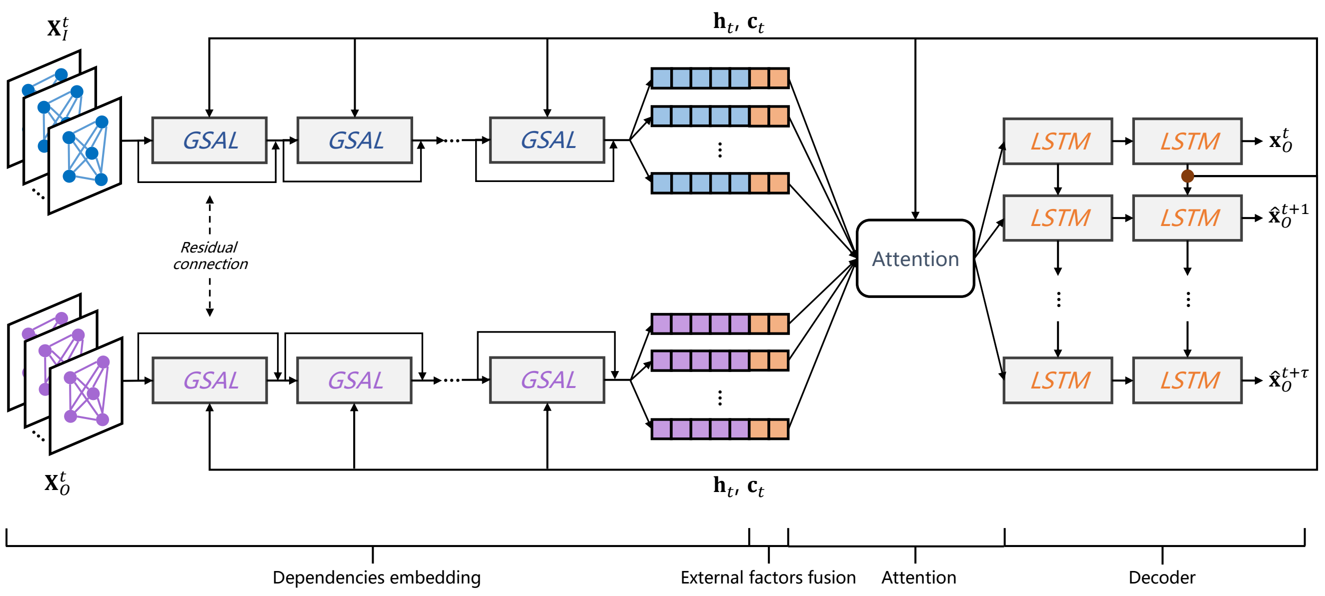


Fig. 2 The framework of our model. GSAL: graph spatial attention layer. $\mathbf{X}_I^t$: inbound passenger flow segment at time interval t. $\mathbf{X}_O^t$: outbound passenger flow segment at time interval t. $\hat{\mathbf{x}}_O^{t+1}$: the predictive outbound flow of all stations at time interval $t+1$. $\mathbf{h}_t$: the hidden state at t-th time interval. $\mathbf{c}_t$: the cell state at t-th time interval.

they are separated by several stations. To fully represent such relationships and expeditiously extract the traffic patterns inherent in the metro system, we construct two directed complete graphs $\mathcal{G}_I(\mathcal{V}, \mathcal{E}, \mathbf{A}_I)$ and $\mathcal{G}_O(\mathcal{V}, \mathcal{E}, \mathbf{A}_O)$ carrying inbound and outbound information respectively. $\mathcal{V} \in \mathbb{R}^N$ denotes the set of nodes and each node corresponds to a metro station. $\mathcal{E} \subseteq |\mathcal{V}| \times |\mathcal{V}|$ denotes the set of edges. $\mathbf{A}_I \in \mathbb{R}^{N\times N}$ and $\mathbf{A}_O \in \mathbb{R}^{N\times N}$ represents the weighted matrices of inbound informative graph $\mathcal{G}_I$ and outbound informative graph $\mathcal{G}_O$, respectively. Given a graph $\mathcal{G}_\beta$ ($\beta = I$ or O), $\mathbf{A}_\beta(i,j)$ is the weight of the edge from node j to node i.

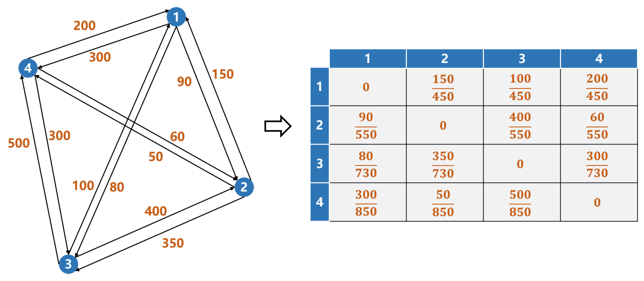
3.3.1 Inbound informative graph

Distinct from the traditional GNNs, the weighted matrix $\mathbf{A}_I \in \mathbb{R}^{N\times N}$ in $\mathcal{G}_I$ is developed based on domain knowledge rather than adjacency and degree of nodes. We first construct an origin-destination matrix $\mathbf{D} \in \mathbb{R}^{N\times N}$, where $\mathbf{D}(i,j)$ denotes the total number of passengers travelling from station s_j to station s_i in the training set. Then we obtain the $\mathbf{A}_I(i,j)$ by employing the row normalization on $\mathbf{D}$ (as shown in Fig. 3):

$$\mathbf{A}_I(i,j) = \frac{\mathbf{D}(i,j)}{\sum_{k=1}^{N}\mathbf{D}(i,k)} \tag{13}$$

3.3.2 Outbound informative graph

Likewise, to derive the weighted matrix $\mathbf{A}_O \in \mathbb{R}^{N\times N}$, we first construct a similarity matrix $\mathbf{Q} \in \mathbb{R}^{N\times N}$. We



	1	2	3	4
1	0	$\frac{150}{450}$	$\frac{100}{450}$	$\frac{200}{450}$
2	$\frac{90}{550}$	0	$\frac{400}{550}$	$\frac{60}{550}$
3	$\frac{80}{730}$	$\frac{350}{730}$	0	$\frac{300}{730}$
4	$\frac{300}{850}$	$\frac{50}{850}$	$\frac{500}{850}$	0

Fig. 3 Illustration of inbound informative graph generation.

denote by $\mathbf{q}_{s_i}$ the historical outbound flow of station s_i extracted from the training set. Thus the similarity score between two stations s_i and s_j can be calculated by:

$$\mathbf{Q}(i,j) = \frac{1}{1 + \exp\left(\mathrm{DDTW}\left(\mathbf{q}_{s_i}, \mathbf{q}_{s_j}\right)\right)} \tag{14}$$

where DDTW$(\cdot)$ denotes the Derivative Dynamic Time Warping [37] algorithm, which is a modification of DTW [38] and is widely used in measuring the distance between two periodic time series. Then we apply the row normalization to $\mathbf{Q}$ to obtain the $\mathbf{A}_O(i,j)$:

$$\mathbf{A}_O(i,j) = \frac{\mathbf{Q}(i,j)}{\sum_{k=1}^{N}\mathbf{Q}(i,k)} \tag{15}$$

3.4 Graph encoder

In this section, we first introduce a generic GNN model used by [39] to represent and propagate passenger flow information. Given the feature matrix $\left(\mathbf{X}_\beta^t\right)^{(l)} \in \mathbb{R}^{N\times T_\beta}$ in the l-th layer ($\beta = I$ or O), the output $\left(\mathbf{X}_\beta^t\right)^{(l+1)} \in \mathbb{R}^{N\times T_\beta}$ can be derived as follows:

$$\left(\mathbf{X}_\beta^t\right)^{(l+1)} = f\left(\mathbf{A}_\beta \left(\mathbf{X}_\beta^t\right)^{(l)} \mathbf{W}^{(l)}\right) \tag{16}$$

where $\mathbf{A}_\beta \in \mathbb{R}^{N\times N}$ is the weighted matrix, $\mathbf{W}^{(l)} \in \mathbb{R}^{T_\beta\times T_\beta}$ refers to the learnable layer-specific parameter matrix and $f(\cdot)$ denotes the activation function.

Obviously, this fundamental GNN model can exploit the spatial and temporal characteristics inherent in metro passenger flow data. Nevertheless, since the weighted matrix $\mathbf{A}_\beta$ is a static constant, it fails to capture the dynamic correlations between different nodes (i.e., stations). To address the above issues, we devise the unique GSAN to reinforce the embedding capability of DAGNN.

3.4.1 Graph spatial attention layer

In the context of metro passenger flow prediction, the interactions between different stations are complex and variable. There are many other factors (e.g. crowding degrees of stations and carriages) can have impact on the passenger flow upon the metro network. Hence, modulating concrete relations between nodes in real time is significant for further exploring the dependencies. Recently, Graph Attention Network (GAT) is prevalent as an efficient paradigm to implement the above function. As contrasted with spectral-domain Graph Convolutional Networks (GCNs), GAT accommodates specifying different importance to neighborhoods within the same hop through self-attention mechanism [39]. Self-attention mechanism has achieved huge successes in natural language processing [40, 41] and computer vision [42–44]. It ensures that GAT has the superiority in modelling the network dynamics compared to GCNs. However, the attention scores calculated in GAT are purely dependent on the input, which induces the learning bias to some extent. Thus we incorporate the hidden states generated by decoder to enhance the relations representation. Specifically, hidden states are used as approximations to the latent features of network status.

To augment the representation power of input features, a linear transformation $\mathbf{U} \in \mathbb{R}^{T_\beta\times T_\beta}$ is applied to each node to obtain the high-dimensional expression. Then we can compute the pair-wise relation score from node j to node i as follows:

$$u_{ij} = g\left(\mathbf{a}^{\mathrm{T}}\left[\mathbf{U}\left(\left(\mathbf{X}_\beta^t\right)_i^{(l)}\right)^{\mathrm{T}} || \mathbf{U}\left(\left(\mathbf{X}_\beta^t\right)_j^{(l)}\right)^{\mathrm{T}} || \mathbf{h}_t || \mathbf{c}_t\right]\right) \tag{17}$$

where $\mathbf{a} \in \mathbb{R}^{2T_\beta+2N}$ is a mapping vector. $\left(\mathbf{X}_\beta^t\right)_i^{(l)}$, $\left(\mathbf{X}_\beta^t\right)_j^{(l)} \in \mathbb{R}^{1\times T_\beta}$ denote the i-th and j-th row of input feature matrix $\left(\mathbf{X}_\beta^t\right)^{(l)}$ respectively. $\mathbf{h}_t$, $\mathbf{c}_t \in \mathbb{R}^N$ are the hidden and cell states at the current time interval t respectively. $(\cdot||\cdot)$ refers to the concatenation operation. $g(\cdot)$ denotes the LeakyReLU function [45]:

$$g(x) = \begin{cases} x & \text{if } x \geq 0, \\ \frac{x}{0.2} & \text{if } x < 0. \end{cases} \tag{18}$$

Afterwards, a softmax function is employed to ensure the attention weights of a node sum to one:

$$e_{ij} = \frac{\exp(u_{ij})}{\sum_{k=1}^{N} \exp(u_{ik})} \tag{19}$$

Finally, these element-wise attention weights can constitute an attention matrix $\mathbf{E}_t^{(l)} \in \mathbb{R}^{N\times N}$ and we impose it on initial weighted matrix $\mathbf{A}_\beta$ to update the propagation rule:

$$\left(\mathbf{X}_\beta^t\right)^{(l+1)} = f\left(\left(\mathbf{E}_t^{(l)} \odot \mathbf{A}_\beta\right)\left(\mathbf{X}_\beta^t\right)^{(l)} \mathbf{W}^{(l)}\right) \tag{20}$$

For convenience, Eq. (20) is abbreviated as:

$$\left(\mathbf{X}_\beta^t\right)^{(l+1)} = \mathrm{GSAL}\left(\mathbf{h}_t, \mathbf{c}_t, \mathbf{A}_\beta, \left(\mathbf{X}_\beta^t\right)^{(l)}\right) \tag{21}$$

3.4.2 Dependencies embedding

On the basis of GSAL, we design two paralleled GSANs to capture the inbound and outbound dependencies respectively. As shown in Fig. 2, we stack the GSALs as building blocks to generate the dependencies embeddings. However, since the range of elements in $\mathbf{E}_t^{(l)}$ is $[0, 1]$, the values will continually decrease, which leads to the degradation of model. To resolve this matter, we introduce the residual connection [46] to promote model training while retaining the original input information. Assume that the number of layers in GSAN is L. The output of l-th layer is:

$$\left(\mathbf{X}_{\beta}^{t}\right)^{(l)} = \begin{cases} \mathrm{GSAL}\left(\mathbf{h}_t, \mathbf{c}_t, \mathbf{A}_{\beta}, \left(\mathbf{X}_{\beta}^{t}\right)^{(l-1)}\right) + \left(\mathbf{X}_{\beta}^{t}\right)^{(l-1)} & , l < L \\ \mathrm{GSAL}\left(\mathbf{h}_t, \mathbf{c}_t, \mathbf{A}_{\beta}, \left(\mathbf{X}_{\beta}^{t}\right)^{(l-1)}\right) \cdot \mathbf{W}_{\beta} & , l = L \end{cases} \tag{22}$$

where $\left(\mathbf{X}_{\beta}^{t}\right)^{(0)} = \mathbf{X}_{\beta}^{t}$ and $\mathbf{W}_{\beta} \in \mathbb{R}^{T_{\beta} \times T_{\beta}}$ is a learnable parameter. To prevent the over-smoothing, we utilize a linear transformation in the last layer to obtain the final embedding $\left(\mathbf{X}_{\beta}^{t}\right)^{(L)}$. Apparently, each column of $\left(\mathbf{X}_{\beta}^{t}\right)^{(L)}$, denoted by $\mathbf{x}_{\beta}^{t'} \in \mathbb{R}^{N}$, characterizes the spatial variation of network-wide inbound or outbound flow at the given time interval.

3.5 External factors fusion

Metro passenger flow is affected by various external factors, especially meteorological conditions [47]. Thus we incorporate the meteorological data to enhance the performance of our model. Inspired by some previous works [25, 48] considering the fusion of external factors in traffic prediction, we construct a Multilayer Perceptron (MLP) to extract the meteorological features. For each meteorological variable (e.g. temperature), Min-Max normalization is applied to map the value into the range of $[0, 1]$:

$$y = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{23}$$

Then the processed data is sent to the input layer, which is followed by three stacked fully-connected layers containing 16, 8 and 4 neurons respectively. To speed up the training, we use ReLU [49] as activation function:

$$\mathrm{ReLU}(x) = \max(0, x) \tag{24}$$

Finally, we can obtain the embedding of meteorological information at each time interval t', which is denoted by $\mathbf{x}_{E}^{t'}$.

3.6 Attention based decoder

After acquiring the dependencies embeddings as well as external factor features, we integrate them as the new input to the decoder:

$$\tilde{\mathbf{x}}_{\beta}^{t'} = \left(\mathbf{x}_{\beta}^{t'} || \mathbf{x}_{E}^{t'}\right) \tag{25}$$

It should be noted that passenger flows at preceding time intervals have unequal contributions to that at next time interval. To capture such dynamic correlations, we employ a temporal attention mechanism to adaptively select the most relevant inputs:

$$u_{\beta}^{t'} = \mathbf{v}_{T}^{\mathrm{T}}\left(\mathbf{W}_{T}^{'}\left[\mathbf{h}_t || \mathbf{c}_t\right] + \mathbf{W}_{T}\tilde{\mathbf{x}}_{\beta}^{t'} + \mathbf{b}_{T}\right) \tag{26}$$

$$\gamma_{\beta}^{t'} = \frac{\exp\left(u_{\beta}^{t'}\right)}{\sum_{k=1}^{T_{\beta}} \exp\left(u_{\beta}^{k}\right)} \tag{27}$$

$$\mathbf{c}_{\beta}^{t} = \sum_{k=1}^{T_{\beta}} \gamma_{\beta}^{k} \tilde{\mathbf{x}}_{\beta}^{k} \tag{28}$$

where $\mathbf{W}_{T} \in \mathbb{R}^{(N+4) \times 2N}$, $\mathbf{W}_{T}^{'} \in \mathbb{R}^{(N+4) \times (N+4)}$ and $\mathbf{v}_{T}$, $\mathbf{b}_{T} \in \mathbb{R}^{N+4}$ are learnable parameters. Subsequently, we concatenate the inbound and outbound context vectors and feed it into the LSTM network to produce the final predictions:

$$\hat{\mathbf{x}}_{O}^{t+\epsilon} = \mathrm{LSTM}\left(\mathbf{c}_{I}^{t} || \mathbf{c}_{O}^{t}, \mathbf{h}_t, \mathbf{c}_t\right) \tag{29}$$

Since our model is smooth, we can use back-propagation algorithm [34] to train the model. During the training stage, parameters in the approach are optimized by Adam optimizer [50] through minimizing the mean squared error (MSE) between the predictive values and ground-truth outbound passenger flows. Specifically, we choose M training samples, each of which contains N stations and τ prediction steps, to optimize our model. The objective function can be defined as follows:

$$\theta = \arg\min_{\theta} \frac{1}{MN\tau} \sum_{i=1}^{M} \sum_{k=1}^{\tau} \left\| \mathbf{x}_{O}^{t_i+k} - \hat{\mathbf{x}}_{O}^{t_i+k} \right\|_2^2 \tag{30}$$

where $\|\cdot\|_2$ refers to L2-norm and θ denotes all learnable parameters.

4 Experiments

4.1 Data description

To verity the proposed architecture, we create a benchmark dataset for the evaluation. It consists of two kinds of data, including metro passenger flow data and meteorological data of Guangzhou in 2017.

4.1.1 Metro passenger flow data

Metro passenger flow is collected from the Guangzhou Metro System, which contains more than 460 million transaction records from June 22th to September 30th in 2017. By 2017, Guangzhou Metro is composed of 10 metro lines and 166 stations, including some stations of Guangfo Line located in Foshan city. The time span to aggregate passenger flow is set to 10 mins.

4.1.2 Meteorological data

We obtained high-frenquency regional weather data (0.5h intervals) from the Wunderground[1], which is an acknowledged meteorological data provider. Since there was only one weather station located near the Guangzhou Baiyun International Airport, the data extracted from it was regarded as that of whole Guangzhou city. To fully depict the meteorological information, five meteorological variables were adopted as external factors, including temperature, precipitation, humidity, wind speed and visibility.

In the experiments, we choose the first 69 days as the training set, the next 16 days as the validation set and the remaining 16 days as the testing set. Our model is trained on the training set with 1000 epochs and early-stopping mechanism is adopted to prevent overfitting.

4.2 Experimental setting

4.2.1 Parameter setting

We set the number of prediction steps $\tau = 3$ to produce both short-term and long-term forecasts. During the training stage, the batch size and learning rate of Adam optimizer are 256 and 0.001 respectively. Totally, our model has 6 hyperparameters. The length of inbound time window T_I is set to 18 (last three hours) and the length of three outbound flow segments $T_{O,r}$, $T_{O,d}$ and $T_{O,w}$ are set to 12 (last two hours), 4 (previous four days) and 2 (previous two weeks) respectively. Moreover, we set the number of layers $q = 2$ for the stacked LSTM network. The hidden size is set to 256.

4.2.2 Evaluation metrics

Two commonly used performance metrics, Mean Absolute Error (MAE) and Root Mean Square Error (RMSE), are applied in the experiments.

[1] `https://www.wunderground.com/`

Table 1 Optional values of input-related hyperparameters

Model	Optional values
T_I	6,12,18,24,30
$T_{O,r}$	3,6,12,18,24
$T_{O,d}$	2,3,4,5,6
$T_{O,w}$	0,1,2,3,4

$$MAE = \frac{1}{zN}\sum_{i=1}^{z}\left\|\mathbf{x}_O^i - \hat{\mathbf{x}}_O^i\right\|_1 \tag{31}$$

$$RMSE = \sqrt{\frac{1}{zN}\sum_{i=1}^{z}\left\|\mathbf{x}_O^i - \hat{\mathbf{x}}_O^i\right\|_2^2} \tag{32}$$

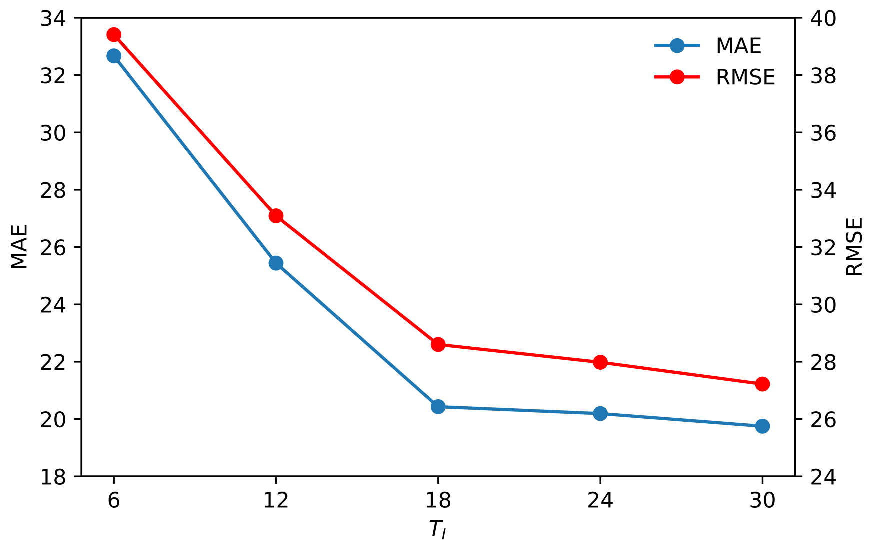
where $\|\cdot\|_1$ refers to L1-norm. z denotes the total number of testing samples and N is the total number of stations. $\mathbf{x}_O^i$ and $\hat{\mathbf{x}}_O^i$ represent the ground-truth and predictive outbound flow of i-th sample respectively.

4.3 Parametric studies

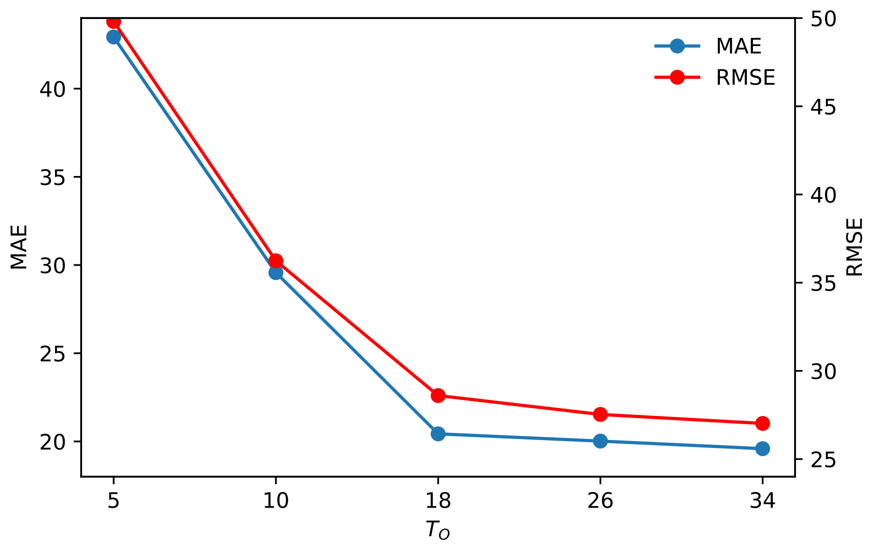
Although increase in the inputs of time span can result in the prediction accuracy of models, the computational cost would be simultaneously increased due to the data-specific parametric learning. Therefore, a trade-off between computational efficiency and forecast precision should be identified. In this section, four input-related hyperparameters T_I, $T_{O,r}$, $T_{O,d}$ and $T_{O,w}$ are investigated on the validation set and the value range of them are listed in Table 1.

To evaluate the performance of DAGNN on the inbound time span, the quantitative analysis is conducted across setting different T_I values (6 to 30), see Fig. 4(a). It can be seen that there is a significant decrease trend at the beginning on the predictive errors, which however level-off after $T_I = 18$. Also, the performance of DAGNN on the outbound time span is examined. To be specific, the impact of overall outbound time span T_O ($T_O = T_{O,r} + T_{O,d} + T_{O,w}$) is adopted for simplifying the calculation process. Fig. 4(b) provides the prediction results over the varying T_O values. Similarly, values of the MAE and RMSE both descend dramatically until $T_O = 18$.

To sum up, with the consideration of the trade-off between the computational efficiency and prediction accuracy, four input-related hyper-parameters $T_I = 18$, $T_{O,r} = 12$, $T_{O,d} = 4$ and $T_{O,w} = 2$ are applied in this study, which is consistent with the parameter setting in Section 4.2.1.



(a) Evaluation on the length of inbound time window.



(b) Evaluation on the length of outbound time window.

Fig. 4 Prediction results of the proposed DAGNN with respect to input-related hyperparameters.

4.4 Model comparison

In addition to the basic methods, such as autoregressive integrated moving average (ARIMA) and Bayesian ridge regression (BRR), the performance of our model was compared with five state-of-the-art deep learning-based metro passenger flow prediction approaches. To comprehensively verify the superiority of proposed model, we carefully tune the parameters for each baseline and present their best results. Furthermore, for convenience, our model is denoted by DAGNN.

– **CNN-BLSTM**: Ma et al. [51] developed a parallel structure by combining CNN and BLSTM. They transformed the metro ridership into an image and utilize CNN to capture the spatial correlations among nearby stations. Simultaneously, BLSTM is employed to explore the temporal dependencies hidden in metro flow sequences. In this study, the convolutional filter and pooling size are set to 3×3 and 2×2 respectively. The hidden size of BLSTM is set to 512 after numerous experiments.

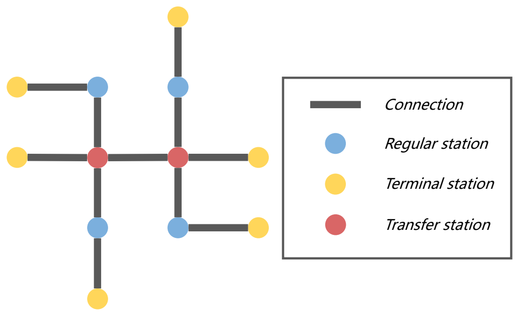


Fig. 5 Example of a metro network.

– **DeepPF**: Liu et al. [47] proposed a LSTM-based ensemble model for metro passenger flow prediction, which considers the meteorological factors and metro operational properties. In this study, when the hidden size of LSTM is set to 512, DeepPF achieves the best results.
– **AS2S**: Hao et al. [52] introduced sequence-to-sequence model with attention mechanism to produce multi-step predictions. Through exploiting the inbound passenger flow at each station in the last few time intervals, this model is able to forecast the outbound passenger flows in the near future. In this study, the hidden size of BLSTM is set to 256. In addition, we test the different combinations of depth of encoder and decoder. The attention Seq2Seq model with 2 layers of encoder and 3 layers of decoder, which is denoted by AS2S_2E3D, outperforms the others.
– **STGCN**: Yu et al. [29] proposed an GCN and CNN combined model for network-wise traffic prediction. They constructed an undirected graph based on the physical topology of a network and then employed GCN to extract spatial features. Simultaneously, a gated CNN along time axis is used to extract temporal features. Here, the kernel size of both GCN and CNN are set to 3.
– **STGAT**: This model is a variant of STGCN, which replaces the GCN with GAT.

In this experiment, we firstly divide the stations into three categories (as illustrated in Fig. 5) and then evaluate the performance of different models on each category. Table 2-4 summarize the comparison results at each time interval. The best results (lowest MAE and RMSE) are highlighted in bold.

While classical models, such as ARIMA and BRR, acquire decent results in this experiment, deep learning-based models are obviously more effective in both short-term and long-term predictions. Moreover, classical models are not capable of handling high-dimensional data, which makes them unsuitable for network-wide forecast. In terms of five deep learning-based baselines, CNN-BLSTM performs worse than the other four models, especially in the long-term prediction task (the next 20-30 mins). The main reason is that CNN is able to extract spatial characteristics from metro passenger flow image to some extent, nevertheless, the rendered image is sparse and can not express the station-level connection. Worse still, the sparsity will become more serious with the increase of prediction steps. AS2S_2E3D and STGAT produce comparatively better results than DeepPF and STGCN respectively, thus the results indicate that attention mechanism is helpful for enhancing the model performance, no matter in short-term or long-term prediction. This is the underlying reason for the development of spatial and temporal attention in our model. In addition, we test DAGNN models of different structures through varying the number of layers in graph encoder. For instance, DAGNN_2L denotes the DAGNN model with 2 GSALs in both inbound and outbound dependencies embedding components. Overall, the DAGNN models have much better performance, especially that have more than 2 GSALs. The best result is yield from DAGNN_3L. Compared to that of DAGNN_3L, the total average RMSE is decreased by 44.26%, 35.53%, 23.25%, 18.62%, 11.52%, 10.68% and 7.34% for ARIMA, BRR, CNN-BLSTM, DeepPF, AS2S_2E3D, STGCN and STGAT, respectively. Apart from the total average error, we also focus on the prediction accuracy of transfer station, since the transfer station has a huge passenger flow and is critical for metro system. Obviously, DAGNN_3L also shows the superiority on the forecast of large ridership. The average RMSE for the transfer station is found to be reduced by 43.52%, 34.47%, 22.14%, 17.20%, 11.18%, 10.98% and 8.84% for ARIMA, BRR, CNN-BLSTM, DeepPF, AS2S_2E3D, STGCN and STGAT, respectively. In addition, as shown in Table 2-4, the proposed model is relatively robust while some of the other baselines suffer from the sharp decline of prediction accuracy as the forecast interval moves forward. This demonstrates that DAGNN indeed has an advantage in capturing the long-term dependencies.

To better evaluate the performance of the proposed DAGNN, Fig. 6 illustrates the prediction accuracy with regard to the passenger flow of each station. As we can see, the prediction accuracy of stations located in downtown area are mostly lower than that of located in suburban area. This can be attributed to the large passenger flow and unstable travel demands of downtown stations. In particular, stations located in bustling commercial area like Tianhe and Haizhu districts, attract enormous non-commuting trips and thus present the poor forecast performance. Also, the traffic mix on the transfer stations is generally complicated, thus it is

Table 2 Prediction results in the next 0-10 mins

Model	Transfer station		Terminal station		Regular station	
	MAE	RMSE	MAE	RMSE	MAE	RMSE
ARIMA	33.24	42.79	23.07	35.46	29.10	39.17
BRR	30.07	36.25	22.18	29.60	25.39	31.82
CNN-BLSTM	24.18	31.07	20.11	25.62	23.08	27.12
DeepPF	22.35	30.88	18.79	24.05	21.03	27.66
AS2S_2E3D	20.93	28.47	17.33	22.80	18.81	26.57
STGCN	21.05	28.39	17.12	22.97	18.68	26.64
STGAT	20.41	27.96	17.04	22.81	18.20	25.97
DAGNN_1L	21.31	29.07	18.39	23.58	20.94	26.84
DAGNN_2L	20.18	27.43	17.28	22.24	19.03	25.79
DAGNN_3L	**18.77**	**25.68**	**15.52**	**20.91**	**17.25**	**24.26**
DAGNN_4L	19.13	26.05	16.74	21.39	18.77	25.53

Table 3 Prediction results in the next 10-20 mins

Model	Transfer station		Terminal station		Regular station	
	MAE	RMSE	MAE	RMSE	MAE	RMSE
ARIMA	37.12	45.64	26.48	39.15	33.17	42.70
BRR	33.58	40.22	25.14	32.31	29.93	36.80
CNN-BLSTM	26.35	32.87	21.57	27.42	24.58	31.15
DeepPF	24.51	32.49	20.86	25.19	23.22	30.68
AS2S_2E3D	21.72	30.58	17.36	23.11	19.34	28.09
STGCN	21.82	30.46	17.19	22.84	19.67	28.15
STGAT	21.48	30.13	16.90	22.71	19.32	27.84
DAGNN_1L	22.43	31.30	18.21	23.97	20.69	29.72
DAGNN_2L	21.56	30.19	17.05	22.83	18.90	26.27
DAGNN_3L	**20.89**	**27.06**	**16.02**	**21.35**	**17.75**	**24.98**
DAGNN_4L	21.32	27.94	16.51	21.60	18.23	25.33

Table 4 Prediction results in the next 20-30 mins

Model	Transfer station		Terminal station		Regular station	
	MAE	RMSE	MAE	RMSE	MAE	RMSE
ARIMA	44.27	55.03	35.17	45.29	40.68	52.37
BRR	38.47	47.19	32.31	41.22	36.97	45.62
CNN-BLSTM	32.71	40.14	29.88	33.40	31.24	38.09
DeepPF	28.03	34.50	24.72	30.93	26.41	33.72
AS2S_2E3D	23.96	32.17	21.26	26.53	21.97	30.34
STGCN	24.05	32.16	21.18	26.39	22.13	30.24
STGAT	23.72	30.81	20.70	25.26	22.08	28.99
DAGNN_1L	24.00	31.82	20.52	26.08	21.15	28.77
DAGNN_2L	22.65	30.61	19.37	24.10	20.81	26.63
DAGNN_3L	**21.33**	**28.29**	**18.49**	**22.96**	**19.82**	**26.20**
DAGNN_4L	21.96	28.40	18.51	23.30	19.96	27.05

not surprising why the prediction performance of the DAGNN on the transfer stations is not as well as other types of stations.

4.5 Variant comparison

To further investigate the effects of different model components on the prediction accuracy, we design the variants of DAGNN_3L as follows and evaluate their performance in this experiment.

– **DAGNN-I$^-$**: We remove the component that captures inbound dependencies.
– **DAGNN-O$^-$**: The component that captures outbound dependencies is excluded from prototype.
– **DAGNN-E$^-$**: This variant does not include the component of external factor fusion.

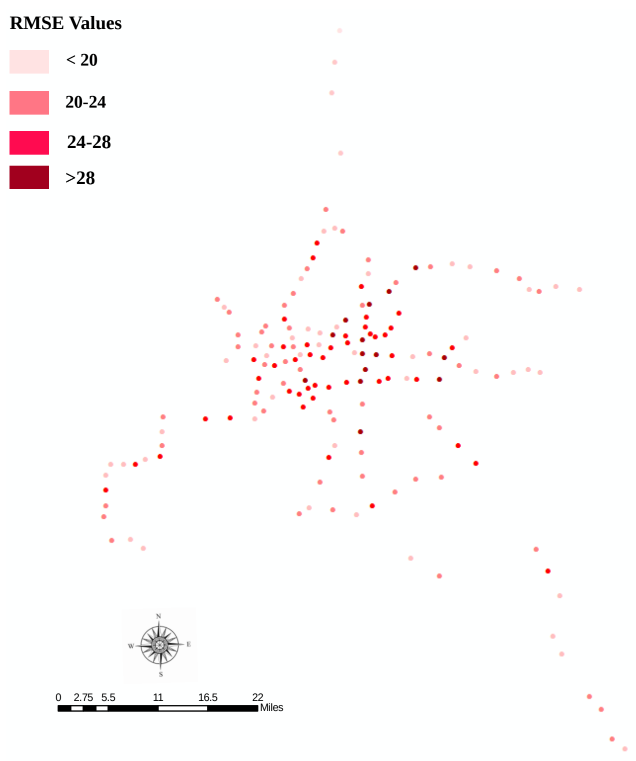


Fig. 6 Spatial distribution of prediction errors.

– **DAGNN-GSAL^{-}**: To validate the effectiveness of GSAL, we replace it with GAT.

The experimental results are shown in Fig. 7. From Fig. 7(a), we observe that the consideration of external factors, meteorological variables, can hardly improve the model performance. The possible reason is that only extreme weather will result in the large fluctuation of metro passenger flow, nevertheless, extreme weather is a rare event in nature. Moreover, DAGNN_3L performs much better than DAGNN-I^{-} and DAGNN-O^{-}, which indicates that it is crucial to capture both inbound and outbound dependencies. Simultaneously, we discover that DAGNN-I^{-} has an advantage over DAGNN-O^{-} in the long-term prediction while DAGNN-O^{-} performs better than DAGNN-I^{-} in the short-term forecast. This illustrates that the capture of inbound and outbound dependencies contribute to the modeling of long-range and short-range correlation respectively, which further proves the superiority of our model. From Fig. 7(b), we find that the prediction accuracy of DAGNN-GSAL^{-} is much lower than that of DAGNN_3L, even worse than that of STGAT with regard to transfer station. Hence, the GSAL in DAGNN do considerably enhance the predictive performance by incorporating the hidden states from decoder.

4.6 Computational efficiency

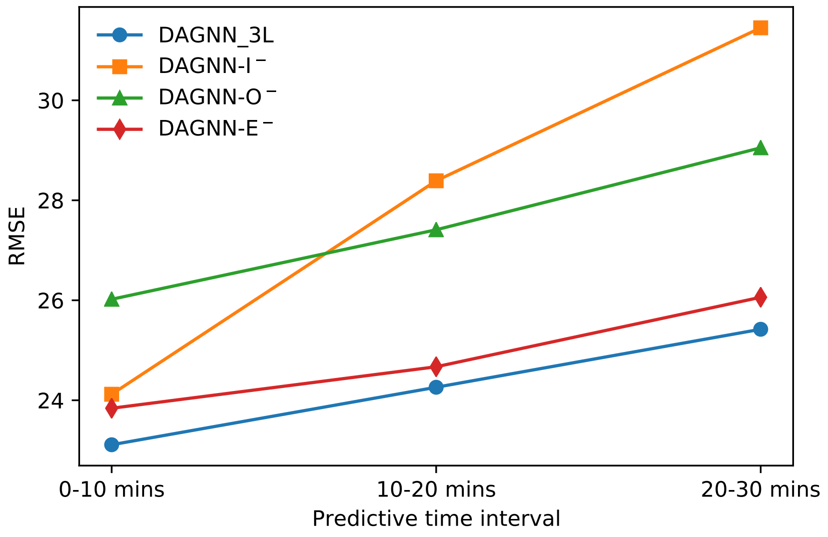
Online detection of metro passenger flow has been receiving more and more attention in recent years. Results of real time detection are useful for the optimal train operation and management of the metro systems. Therefore, it is crucial to enhance the computational efficiency of the passenger flow prediction models. Table 5 presents the average computational time of every step of the proposed model. Results indicate that the proposed DAGNN method is capable for the online detection of metro passenger flow in the order of millisecond. In addition, results also indicate that the average computational times are stable (i.e., five milliseconds) across different prediction steps, time intervals and station types. This justifies the robustness of the proposed DAGNN.

4.7 Case analysis

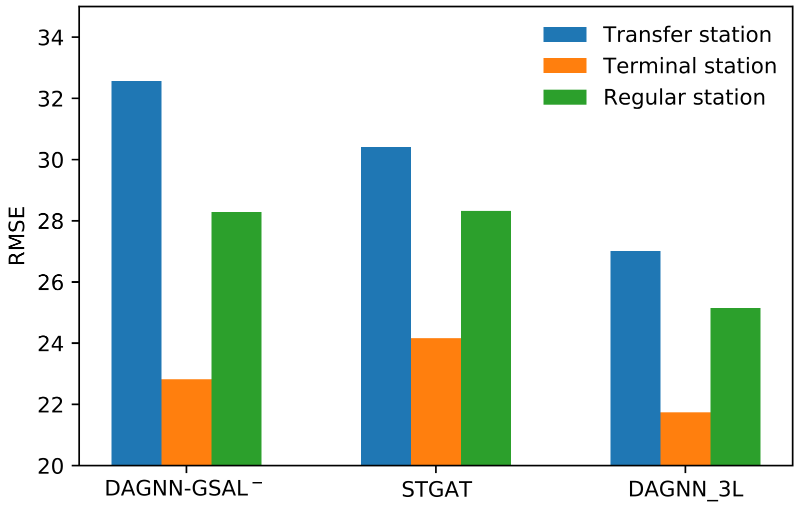
To further investigate the functionality of attention mechanism in DAGNN, we select the most busiest station in Guangzhou (denoted by s_0) which located in the central business district (CBD) and perform a case analysis over the passenger flow of it from 07:30 a.m to 09:00 a.m on Sept. 15, 2017. We plot the spatial attention weights over s_0 obtained from the last layers in inbound and outbound dependencies components in Fig. 8(a) and Fig. 8(b) respectively. For the sake of brevity, we only present a few stations which have high impact on the target station s_0 and mark these stations on the map in Fig. 9.

As illustrated in Fig. 8(a), the outbound passenger flow of s_0 mainly derives from the inbound flows of s_1 and s_2. These two stations are located in rural area (as shown in Fig. 9), which indicates that attention mechanism can well interpret the commuting pattern during the morning rush hours on weekdays. In Fig. 8(b), we observe that outbound flow of s_0 is highly similar to those of adjacent station s_3 and transfer station s_4. Notably, between 08:10 a.m and 08:20 a.m, there was a breakdown happening to Metro Line 3 (news from the official microblog of Guangzhou Metro[2]), which causes the rapid drop in outbound flow of s_0. It worth noting that attention mechanism acutely catches this signal and adjusts the attention weights to make our model pay more attention to the outbound flow of s_3 which is on the same metro line with s_0. This not only highlights the importance of attention mechanism in DAGNN, but also provides a new perspective for anomaly detection in metro system.

[2] `https://www.weibo.com/gzmtr`



(a) Evaluation on the components of inbound dependencies, outbound dependencies and external factor fusion.

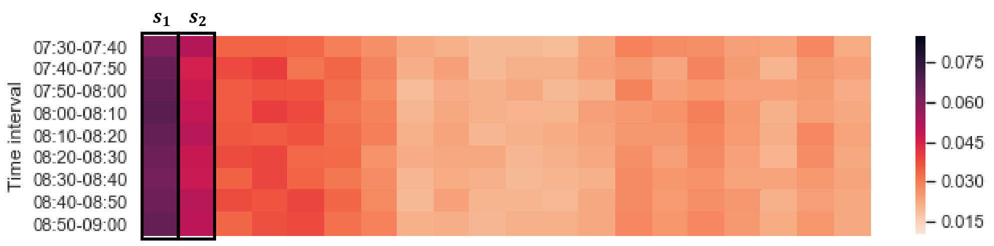


(b) Evaluation on the GSAL.

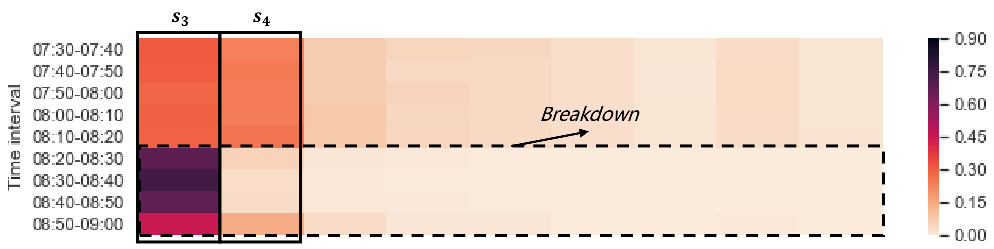
Fig. 7 Comparison of forecast performance among different variants.

Table 5 Time cost per prediction step in testing set (ms)

Time interval	Transfer station	Terminal station	Regular station
0-10 mins	5.164	5.127	5.021
10-20 mins	4.987	4.962	5.007
20-30 mins	5.135	4.990	4.973



(a) Spatial attention weights from inbound dependencies component.



(b) Spatial attention weights from outbound dependencies component.

Fig. 8 Spatial attention matrices obtained from the last layer in inbound and outbound dependencies components, where each row represents the attention weights over inputs.

5 Conclusion

In this study, we revisit the fundamental problem of metro passenger flow prediction. A Graph Spatial Attention Network is proposed to capture both inbound and outbound dependencies. The strength of GSAN is that it introduces spatial attention mechanism to capture the dynamic interactions of (inbound and outbound) passenger flows among stations and adjust the weighted matrices over time. Then, a temporal attention based decoder is constructed to produce both short-term and long-term predictions by incorporating the dependencies embeddings and external factors. Findings of this study justify that the proposed model outperforms five state-of-the-art deep learning-based methods and is effective and efficient for long-term prediction. Moreover, we visualize the attention weights and reveal that DAGNN is useful for illustrating the urban mobility patterns and detecting abnormal events in the metro system.

In the future studies, it is worth exploring the factor interpretability and capability of online detection of metro passenger flow of DAGNN. For instance, the attention weights can be incorporated into the DAGNN for real-time anomaly detection. On the other hand, it is worth investigating the capability of proposed DAGNN for the prediction of metro passenger flow in special circumstances, such as festival events, that have recurrent and non-recurrent fluctuations of the supply and demand of metro services.

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. De Borger B, Kerstens K, Costa A (2002) Public transit performance: what does one learn from frontier studies? Transp Rev 22(1):1–38
2. Li H, Ding H, Ren G, Xu C (2018) Effects of the london cycle superhighways on the usage of the london cycle hire. Transp Res Pt A-Policy Pract 111:304–315
3. Ding H, Sze N, Li H, Guo Y (2020) Roles of infrastructure and land use in bicycle crash exposure and frequency: a case study using greater london bike sharing data. Accid Anal Prev 144:105652
4. Zhong C, Batty M, Manley E, Wang J, Wang Z, Chen F, Schmitt G (2016) Variability in regularity: Mining temporal mobility patterns in london, singapore and beijing using smart-card data. PLoS One 11(2):e0149222
5. Li Y, Wang X, Sun S, Ma X, Lu G (2017) Forecasting short-term subway passenger flow under special events scenarios using multiscale radial basis function networks. Transp Res Pt C-Emerg Technol 77:306–328
6. Tang L, Zhao Y, Cabrera J, Ma J, Tsui KL (2018) Forecasting short-term passenger flow: An empirical study on shenzhen metro. IEEE Trans Intell Transp Syst 20(10):3613–3622

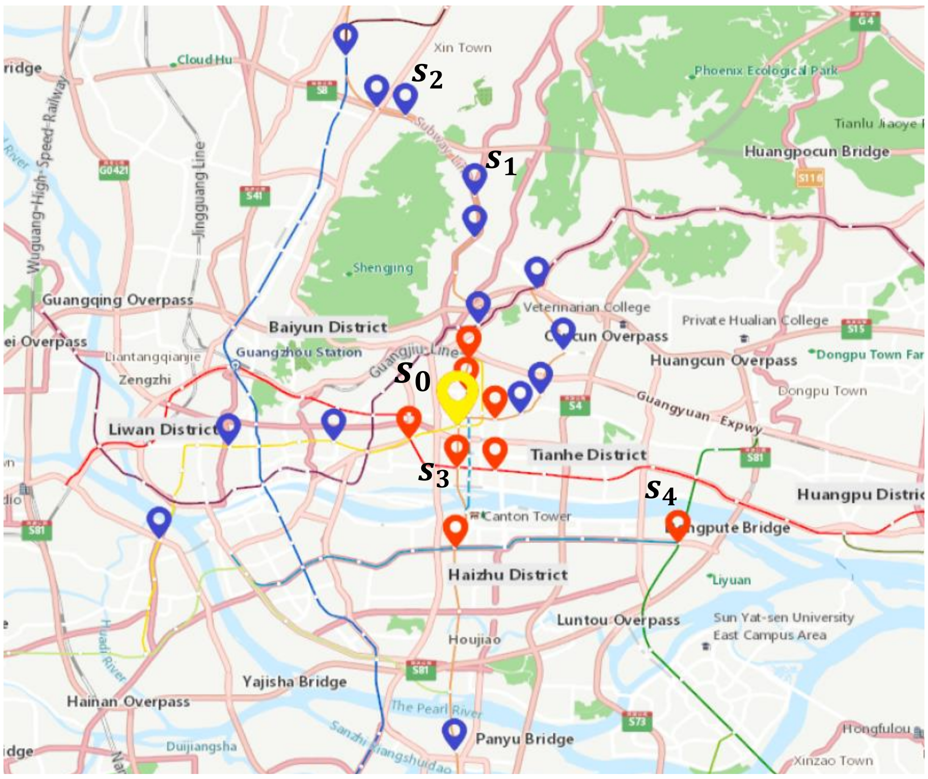


Fig. 9 Distribution of metro stations. Yellow mark denotes the target station s_0. Blue and red marks correspond to the stations in Fig. 8(a) and Fig. 8(b) respectively.

7. Chen E, Ye Z, Wang C, Xu M (2019) Subway passenger flow prediction for special events using smart card data. IEEE Trans Intell Transp Syst
8. Gong Y, Li Z, Zhang J, Liu W, Zheng Y, Kirsch C (2018) Network-wide crowd flow prediction of sydney trains via customized online non-negative matrix factorization. In: CIKM, pp 1243–1252
9. Ahmed MS, Cook AR (1979) Analysis of freeway traffic time-series data by using box-jenkins techniques. Transp Res Record 722:1–9
10. Williams BM, Durvasula PK, Brown DE (1998) Urban freeway traffic flow prediction: application of seasonal autoregressive integrated moving average and exponential smoothing models. Transp Res Record 1644(1):132–141
11. Williams BM, Hoel LA (2003) Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results. J Transp Eng 129(6):664–672
12. Van Der Voort M, Dougherty M, Watson S (1996) Combining kohonen maps with arima time series models to forecast traffic flow. Transp Res Pt C-Emerg Technol 4(5):307–318
13. Smith BL, Demetsky MJ (1997) Traffic flow forecasting: comparison of modeling approaches. J Transp Eng 123(4):261–266
14. Leshem G, Ritov Y (2007) Traffic flow prediction using adaboost algorithm with random forests as a weak learner. In: WASET, Citeseer, vol 19, pp 193–198
15. Crosby H, Davis P, Jarvis SA (2016) Spatially-intensive decision tree prediction of traffic flow across the entire uk road network. In: DS-RT, IEEE, pp 116–119
16. Wu CH, Ho JM, Lee DT (2004) Travel-time prediction with support vector regression. IEEE Trans Intell Transp Syst 5(4):276–281
17. Xu H, Jiang C (2019) Deep belief network-based support vector regression method for traffic flow forecasting. Neural Comput Appl pp 1–10
18. Sun S, Zhang C, Yu G (2006) A bayesian network approach to traffic flow forecasting. IEEE Trans Intell Transp Syst 7(1):124–132
19. Pascale A, Nicoli M (2011) Adaptive bayesian network for traffic flow prediction. In: SSP, IEEE, pp 177–180
20. Lu Y, He Z, Luo L (2019) Learning trajectories as words: a probabilistic generative model for destination prediction. In: MobiQuitous, pp 464–472
21. Zhao SZ, Ni TH, Wang Y, Gao XT (2011) A new approach to the prediction of passenger flow in a transit system. Comput Math Appl 61(8):1968–

1974
22. Hodge VJ, Krishnan R, Austin J, Polak J, Jackson T (2014) Short-term prediction of traffic flow using a binary neural network. Neural Comput Appl 25(7-8):1639–1655
23. Chen Q, Song Y, Zhao J (2020) Short-term traffic flow prediction based on improved wavelet neural network. Neural Comput Appl
24. Wu Y, Tan H, Qin L, Ran B, Jiang Z (2018) A hybrid deep learning based traffic flow prediction method and its understanding. Transp Res Pt C-Emerg Technol 90:166–180
25. Zhang J, Zheng Y, Qi D, Li R, Yi X, Li T (2018) Predicting citywide crowd flows using deep spatio-temporal residual networks. Artif Intell 259:147–166
26. Zhang J, Zheng Y, Sun J, Qi D (2019) Flow prediction in spatio-temporal networks based on multitask deep learning. IEEE Trans Knowl Data Eng 32(3):468–478
27. Zhang Y, Zhou Y, Lu H, Fujita H (2020) Traffic network flow prediction using parallel training for deep convolutional neural networks on spark cloud. IEEE Trans Ind Inform
28. Li Y, Yu R, Shahabi C, Liu Y (2018) Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In: ICLR
29. Yu B, Yin H, Zhu Z (2018) Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. In: IJCAI, pp 3634–3640
30. Chen C, Li K, Teo SG, Zou X, Wang K, Wang J, Zeng Z (2019) Gated residual recurrent graph neural networks for traffic prediction. In: AAAI, vol 33, pp 485–492
31. Wang X, Ma Y, Wang Y, Jin W, Wang X, Tang J, Jia C, Yu J (2020) Traffic flow prediction via spatial temporal graph neural network. In: WWW, pp 1082–1092
32. Zheng C, Fan X, Wang C, Qi J (2020) Gman: A graph multi-attention network for traffic prediction. In: AAAI, vol 34, pp 1234–1241
33. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
34. Rumelhart DE, Hinton GE, Williams RJ (1986) Learning representations by back-propagating errors. Nature 323(6088):533–536
35. Hochreiter S (1998) The vanishing gradient problem during learning recurrent neural nets and problem solutions. Int J Uncertainty Fuzziness Knowl-Based Syst 6(02):107–116
36. Beck D, Haffari G, Cohn T (2018) Graph-to-sequence learning using gated graph neural networks. In: ACL, pp 273–283
37. Keogh EJ, Pazzani MJ (2001) Derivative dynamic time warping. In: SDM, SIAM, pp 1–11
38. Berndt DJ, Clifford J (1994) Using dynamic time warping to find patterns in time series. In: KDD workshop, Seattle, WA, vol 10, pp 359–370
39. Veličković P, Cucurull G, Casanova A, Romero A, Lio P, Bengio Y (2017) Graph attention networks. arXiv preprint arXiv:171010903
40. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention is all you need. In: NIPS, vol 31
41. Devlin J, Chang MW, Lee K, Toutanova K (2018) Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:181004805
42. Fu J, Liu J, Tian H, Li Y, Bao Y, Fang Z, Lu H (2019) Dual attention network for scene segmentation. In: CVPR, pp 3146–3154
43. Ji Y, Zhang H, Wu QJ (2018) Salient object detection via multi-scale attention cnn. Neurocomputing 322:130–140
44. Ji Y, Zhang H, Jie Z, Ma L, Wu QJ (2020) Casnet: a cross-attention siamese network for video salient object detection. IEEE Trans Neural Netw Learn Syst
45. Maas AL, Hannun AY, Ng AY (2013) Rectifier nonlinearities improve neural network acoustic models. In: ICML, vol 30, p 3
46. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: CVPR, pp 770–778
47. Liu Y, Liu Z, Jia R (2019) Deeppf: A deep learning based architecture for metro passenger flow prediction. Transp Res Pt C-Emerg Technol 101:18–34
48. Liang Y, Ke S, Zhang J, Yi X, Zheng Y (2018) Geoman: Multi-level attention networks for geo-sensory time series prediction. In: IJCAI, pp 3428–3434
49. Nair V, Hinton GE (2010) Rectified linear units improve restricted boltzmann machines. In: ICML, pp 807–814
50. Kingma DP, Ba J (2014) Adam: A method for stochastic optimization. arXiv preprint arXiv:14126980
51. Ma X, Zhang J, Du B, Ding C, Sun L (2018) Parallel architecture of convolutional bi-directional lstm neural networks for network-wide metro ridership prediction. IEEE Trans Intell Transp Syst 20(6):2278–2288
52. Hao S, Lee DH, Zhao D (2019) Sequence to sequence learning with attention mechanism for short-term passenger flow prediction in large-scale metro system. Transp Res Pt C-Emerg Technol 107:287–300