Metro Passenger Flow Prediction via Dynamic Hypergraph Convolution Networks

Jingcheng Wang, Yong Zhang, *Member, IEEE*, Yun Wei, Yongli Hu, *Member, IEEE*, Xinglin Piao, and Baocai Yin, *Member, IEEE*

***Abstract*— Metro passenger flow prediction is a strategically necessary demand in an intelligent transportation system to alleviate traffic pressure, coordinate operation schedules, and plan future constructions. Graph-based neural networks have been widely used in traffic flow prediction problems. Graph Convolutional Neural Networks (GCN) captures spatial features according to established connections but ignores the high-order relationships between stations and the travel patterns of passengers. In this paper, we utilize a novel representation to tackle this issue - hypergraph. A dynamic spatio-temporal hypergraph neural network to forecast passenger flow is proposed. In the prediction framework, the primary hypergraph is constructed from metro system topology and then extended with advanced hyperedges discovered from pedestrian travel patterns of multiple time spans. Furthermore, hypergraph convolution and spatio-temporal blocks are proposed to extract spatial and temporal features to achieve node-level prediction. Experiments on historical datasets of Beijing and Hangzhou validate the effectiveness of the proposed method, and superior performance of prediction accuracy is achieved compared with the state-of-the-arts.**

***Index Terms*— Metro flow prediction, hypergraph, graph neural network.**

I. Introduction

RAIL transit, represented by the metro, is a critical component of the transportation system. With the significant progress of urbanization worldwide, underground transit is becoming the leading force in public transportation. By way of example, as of September 2019, 41 cities in China have put subways in use. Beijing has one of the most crowded metro networks at present. Statistically, the volume of the average daily passenger in Beijing has far exceeded 10 million person-time, and the total annual passenger volume has reached 4.5 billion person-time. It can be seen that the current metro system is already a mega engineering project with a massive passenger volume every single day. The cities without metro are planning for construction, and the cities that have already developed subways are building new tracks and stations due to traffic pressure. Rail transit has already been the first choice for urban travel in big cities.

Meanwhile, the problem of congestion is increasingly severe. Taking Beijing as an example again, according to "Beijing Transportation Development Annual Report 2019", 12 of the 15 lines were identified as "Heavy Congestion" on each morning peak period. The minimum departure interval of several lines has been shortened to 120 seconds during morning peak periods, but it still cannot alleviate congestion. Therefore, an accurate passenger flow prediction systematic approach is necessary, which could benefit route scheduling, network design, and crowd regulation. It can help locate and avoid excessive commuting stress on certain stations, reducing the probability of delay or even trampling.

Research on traffic flow prediction has yielded considerable achievements. The majority of previous researches focused on short-term forecasting based on mathematical models [1] and machine learning methods such as Linear Regression [2] and Support Vector Machine (SVM) [3]. Recently, neural networks are playing an increasingly important role in the subject of traffic forecasting. Studies on predicting volume and speed of vehicles on road have made remarkable progress. When it comes to rail transit, related researches are relatively limited.

The specific spatial and temporal characteristics of rail transit flow determine the particularity of this issue. The primary problem is to use a suitable representation to model the spatial structure. Metro tracks compose a graph-based system. Traditional mathematical methods and general machine learning methods either neglect the spatial and temporal information or can only handle the well-arranged Euclidean data [4]. Hence conducting the traditional methods on each station and then averaging the prediction results could not take full advantage of the graph structure. Besides, compared with road transportation, rail transportation owns distinct topologies and operating modes, which could be inappropriate to be represented by the conventional graph. A more advanced data structure that can describe the relationship between subway tracks is needed. Moreover, there is also a

Manuscript received March 15, 2020; revised September 23, 2020 and February 7, 2021; accepted March 25, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 62072015, Grant U19B2039, Grant U1811463, Grant 61632006, Grant 61876012, and Grant 61902053; in part by the Natural Science Foundation of Beijing under Grant 4172003; and in part by the China Scholarship Council under Grant 201806540008. The Associate Editor for this article was Y. Lv. *(Corresponding author: Yong Zhang.)*

Jingcheng Wang, Yong Zhang, Yongli Hu, and Baocai Yin are with the Beijing Key Laboratory of Multimedia and Intelligent Software Technology, Faculty of Information Technology, Beijing Institute of Artificial Intelligence, Beijing University of Technology, Beijing 100124, China (e-mail: wang.jc@emails.bjut.edu.cn; zhangyong2010@bjut.edu.cn; huyongli@bjut.edu.cn; ybc@bjut.edu.cn).

Yun Wei is with Beijing Urban Construction Design and Development Group 53 Company Ltd., Beijing 100029, China (e-mail: weiyun@bjucd.com).

Xinglin Piao is with the Pengcheng Laboratory, Shenzhen 518055, China, and also with the Shenzhen Graduate School, Peking University, Shenzhen 518055, China (e-mail: piaoxl@pcl.ac.cn).

Digital Object Identifier 10.1109/TITS.2021.3072743

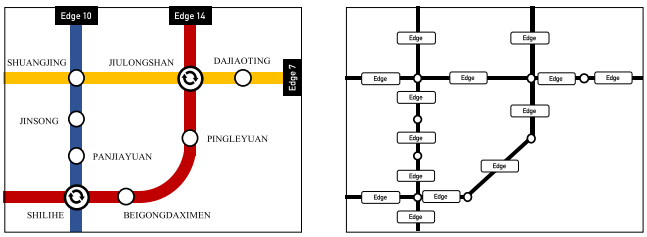


Fig. 1. Comparison of hypergraph (left) and the conventional graph (right) in metro structure representation. Taking a portion of the Beijing subway as an example, hypergraph can model the lines with hyperedges, where the conventional graph can only describe the pairwise relationship between stations.

problem in the temporal pattern. Considering the sustained passenger flow in a whole graph, the connotative inherent correlation of passengers and their OD is varying all the time. Therefore, we also need a dynamic mechanism to extract valuable information from the OD to achieve a more precise prediction.

To solve the problem above, we explore a novel structure of data representation - hypergraph, instead of the existing data modeling method in traffic prediction algorithms. By definition, the hypergraph is a generalization of conventional graphs, representing non-pairwise relationships between vertices with hyperedges. Therefore, hypergraph can model the inherent relationship of data with higher order. For graph-based neural network models of road traffic flow prediction, it is clear each edge connects two nodes. However, the connection between metro stations carries more information than a point-to-point relationship because all the connections belong to the corresponding metro lines. The lines are different from each other in terms of the time interval, operating time, and the number of vehicles. To illustrate, the time between two trains of Airport Express in Beijing is about 30 minutes. While the time gap of Line 10 is about 5 minutes. The edges in conventional graphs are restricted to the pair-wise relationship, which cannot carry such higher-order information. The hypergraph, however, can model it with hyperedges, as shown in Fig. 1. That is the first reason why hypergraph representation is suitable for this problem in the aspect of inherent characteristics. Moreover, we have designed a dynamic mechanism that takes advantage of the scalability of hypergraphs to describe the varying travel patterns of passengers. We construct the primary hypergraph through the subway's topology, and the advanced hypergraphs are extracted from the travel OD of passengers. To tackle the metro flow prediction problem dynamically, we integrate these two kinds of hyperedges jointly. This mechanism could dynamically combine the spatial and temporal features to achieve more accurate flow predictions.

The main contributions of this paper are summarized as:

1) A novel hypergraph representation of rail transit data is proposed. The method takes the peculiar structure of the metro and the varying travel patterns of passengers into account and builds the primary hypergraph and advanced hyperedges accordingly. Not only are the high-order topological connections described by the proposed hypergraph representation, but it also models the dynamic travel patterns of passengers.
2) A multilayer spatio-temporal hypergraph neural network for metro flow prediction is put forward. Spectral convolution on hypergraph is derived from graph convolution operation and hypergraph learning. Then spatio-temporal blocks with hypergraph convolution layers and temporal convolution layers are stacked to establish the flow prediction framework.
3) Extensive experiments are conducted on historical full-scale metro flow datasets of Beijing and Hangzhou. Comparative experiments cover full time, peak period, and abnormal situation. Besides, ablation experiments and time consumption experiments are completed. Comparison with the existing baselines verifies the practicability and effectiveness of the proposed model.

In the following, Section II reviews literature about admitted metro flow prediction methods. Apart from that, relevant graph neural networks and hypergraph learning algorithms are introduced in this section. Section III elaborates the proposed method in detail, covering hypergraph construction and spatio-temporal hypergraph neural networks. An overview of datasets and sufficient experiments are given in Section IV. Finally, Section V concludes this paper.

II. Related Works

The research content and proposed methods in this paper involve passenger flow prediction, graph convolution neural networks, and hypergraph learning theory.

A. Passenger Flow Prediction

Passenger flow prediction is one of the most significant problems in the field of intelligent transportation. Generally, the flow prediction is applied in various scenarios, such as bus stations, subways, shopping malls, and other important POI. Thus the critical literature of traffic flow prediction and metro passenger flow forecasting will be both reviewed.

There are two major categories of prediction methods covered in the flow prediction issue, which are statistical methods and machine learning methods [5]. Among the statistical algorithms, the autoregressive integrated moving average model (ARIMA), also known as the Box-Jenkins model, is one of the most effective approaches, specially designed for time-sequential prediction. After Ahmed and Cook [6] analyzed time-series data of freeway by using the Box-Jenkins model, both Williams *et al.* [7] and Lee and Fambro [8] applied the ARIMA model to realize urban freeway traffic flow prediction. Furthermore, Williams [9] explored the ARIMAX model, Williams and Hoel [10] proposed seasonal ARIMA, and Min *et al.* combined STARIMA and DTRP model to propose DSTARIMA, which is an efficient hybrid spatio-temporal approach of short-term traffic forecasting. Much research in recent years focused on applying deep learning algorithms to transportation applications, which has generated considerable achievement. Huang *et al.* [11] claimed that their research is the earliest application of deep learning to traffic flow prediction research through the proposed DBN networks. Lv *et al.* [12] utilized auto-encoders as deep learning blocks to represent flow features for prediction. Li *et al.* [13]

then predicted traffic flow with GLA and Liu and Chen [14] answered the BRT flow prediction problem with hybrid SAE mode.

Specific to subway passenger flow, Leng *et al.* [1] predicted with a statistical model of probability tree. Ni *et al.* [15] trained Linear Regression model with event occurrences information to tackle the abnormal prediction. Sun *et al.* [3] proposed Wavelet-SVM and discussed the idea to predict different kinds of passenger flows. A combination of empirical mode decomposition and MLP was proposed by Wei and Chen [16] to forecast the short-term metro flow. Later on, a multi-scale radial basis function network [17] was proposed by the same authors to improve their forecasting accuracy. Besides, some researchers focused on transfer stations instead of the whole subway system. For instance, the flow volume of the transfer stations is predicted with a nonparametric regression model especially [2].

By reviewing previous work, the research of metro flow prediction is relatively deficient, and the methods applied are outmoded to some extent. Furthermore, existing machine learning methods either did not consider the unique topology of the subway network or ignore the inherent spatio-temporal characteristics of passenger flow even though they consider physical connections. Thereby the improvement of flow prediction accuracy is limited.

B. Graph Neural Networks

In recent years, methods based on graph representation and graph networks have made great progress in learning spatio-temporal data. Deep learning models such as Convolutional Neural Networks (CNN) and Recurrent Neural Network (RNN) have shown gorgeous ability on Euclidean domains data [4], but it cannot deal with graph represented data directly. Hence the graph-based networks were put forward to tackle this issue. There are two main categories of graph neural networks, one is based on the spatial domain [18], and the other is based on the spectral domain [19]. Spatial domain methods elucidated convolution operation on vertices [20]. Their methods are more similar to traditional CNN [21]. While spectral graph neural networks were first proposed by Bruna *et al.* [19]. Their work defined Fourier Transformation on a graph and then defined graph convolution operation. In order to reduce computation complexity of graph convolution, Defferrard *et al.* and Kipf *et al.* proposed ChebNet [22] and 1stChebNet [23] respectively. For applications to graph neural network, [23] explored semi-supervised node-level classification with GCN. [24] proposed gated GCN, which focus on sequence outputs. [25] combined GCN and RNN to realize spatio-temporal sequence modeling on moving MNIST. [26] applied attention mechanism on GCN and achieve a more accurate citation classification issue.

Graph neural networks demonstrate the excellent learning ability on non-European data. Compared with CNN, GCN requires specialized modeling of the graph structure. Therefore, it is essential to use the most appropriate graph representation method to tackle the specific issue.

C. Hypergraph Learning

However, due to some limitations of graph representation, such as the inability to represent multiple interrelations, researchers have developed hypergraph learning theory.

The hypergraph is initially defined as a mathematical concept. Specifically, a conventional graph is a particular form of a hypergraph with fixed order of two. Dengyong Zhou *et al.* deduced normalized hypergraph cut from normalized graph cut and random walk theory and gave the expression of hypergraph Laplacian [27]. Moreover, a hypergraph based spectral clustering methodology and transductive classification algorithms were proposed in their article.

Then hypergraph learning methods are gradually used in clustering and classification issues. For example, social information classification and network motif clustering are two proper implementations, which are realized by Arya and Worring [28] and Li and Milenkovic [29] with geometric hypergraph learning and inhomogeneous hypergraph clustering, respectively. Furthermore, Yan *et al.* [30] introduced image matching method with discrete hypergraph and Leordeanu and Sminchisescu [31] utilized hypergraph clustering methods to realize affine-invariant matching. Leordeanu *et al.* [32] addressed image matching with a semi-supervised hypergraph learning method, which made matching more robust to changes in scale, deformations, and outliers. For other applications, Su *et al.* [33] took advantage of vertex-weighted hypergraph to classify multi-view 3D object with hypergraph partition algorithms. An *et al.* [34] also implemented hypergraph partition and designed person re-ID algorithms. Their work introduced hypergraph fusion methods, which gave us the inspiration for a new feature fusion method. Hypergraph learning methods have an evident trend of combination with graph neural network methods, which leads to several brand new approaches. Yadati *et al.* [35] and Feng *et al.* [36] proposed the method of training GCN on hypergraphs. As for application, they both improved the accuracy of citation classification on the Cora dataset.

Hypergraph representation learning has achieved satisfactory results on clustering and some classification problems. The literature indicates the high order representation ability of hypergraph learning. However, the current hypergraph network is static, so we intend to capture the temporal and spatial features more effectively through a dynamic mechanism to achieve more accurate metro flow prediction.

III. METODOLOGY

The basic idea of the proposed method is to first model the metro flow prediction problem through hypergraph representation. The hypergraph spectral convolution is derived from the hypergraph learning theory. The hypergraph neural network framework is then designed with the dynamic mechanism, and the node-level passenger flow prediction is effectuated.

A. Overview

As a strict time series problem, metro passenger flow prediction is a sub-problem of traffic flow prediction. The purpose

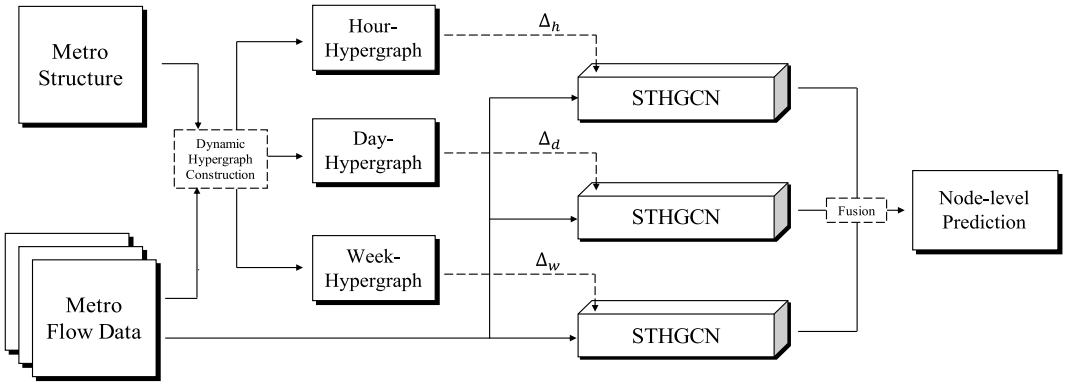


Fig. 2. Overview of the framework of the proposed method. Metro structure and flow data are combined to construct dynamic hypergraphs in three time spans: hour, day, and week. Then hypergraph Laplacian matrices and historical metro flow data are fed into individual spatio-temporal hypergraph graph neural network (STHGCN). The results would fuse and finally output the node-level prediction.

is to estimate the flow count at a future time based on the data collected over previous periods from certain observation locations. Thus, given a basic metro structure and a set of historical data $\{\mathbf{X}^t \in \mathbb{R}^{N\times F}\}_{t=1}^{T}$, where N denotes the size of stations, F denotes the channel of features, and T denotes the dimension of time. Each $\mathbf{X}^t = (x_1^t, x_2^t, \ldots, x_N^t) \in \mathbb{R}^{N\times F}$ to represent the input vector of historical flow on the t-th timestamp of the stations. For the outputs, the node-level prediction results are $\hat{\mathbf{X}}^m = (\hat{x}_1^{T+m}, \hat{x}_2^{T+m}, \ldots, \hat{x}_N^{T+m})^\top \in \mathbb{R}^{F\times N}$, where $\hat{x}_n^{T+m}$ denotes the outputs of the n-th station on a future timestamp m.

Generally, hypergraph representation models the problem with feature-based methods or structure-based methods. The feature-based methods mine the correlation between vertices by clustering features to construct hyperedges. The structure-based methods utilize the inherent topology of the dataset itself. Using only one of these methods may simplify the original structural information, but we combine the two methods to enrich the connotation of hypergraph representation.

Moreover, a dynamic mechanism is inspired by the travel patterns of passengers. Commuters travel back and forth from the workplace and their residence during weekdays. When the quantity of passengers reaches a considerable scale, there will be strong and dynamic correlations between the corresponding stations.

Thus we design the Dynamic Spatio-Temporal Hypergraph Graph Convolution Network (DSTHGCN). Fig. 2 demonstrates the architecture of the proposed model. As illustrated in the framework, the raw input data includes metro flow data and the rail network structure. The dynamic hypergraphs are firstly built according to the raw input data. Afterward, the hypergraph Laplacian matrices and the metro flow data are fed into the STHGCN, which has hypergraph convolution layers and temporal convolution layers in the spatio-temporal blocks to capture both the spectral and temporal feature of each vertex in hypergraphs. Finally, the proposed model fuses the prediction of three time spans and then outputs node level prediction.

B. Hypergraph Construction

Conventional graph, donated by $\mathcal{G} = (\mathcal{V}; \mathcal{E})$, is a common used topological structure. A graph contains a set of vertices $\mathcal{V}$, which are pair-wisely connected by edges $\mathcal{E}$. By contrast, the preliminary definition of the hypergraph is to consider a generalization of graphs where the hyperedges can be denoted as sets of certain vertices. The hypergraph with vertex set $\mathcal{V}$, hyperedge set $\mathcal{E}^h$ and the weights $\mathcal{W}$, denoted by $\mathcal{G}^h = (\mathcal{V}; \mathcal{E}^h = (e_i)_{i\in I}; \mathcal{W})$.

A hypergraph is regular if every vertex has the same degree. Conventional graphs are 2-regular hypergraphs. Therefore, some definitions of graphs and hypergraphs are homogeneous. The incidence matrix of graph theory reveals the relationship between vertices and edges. For an undirected hypergraph without isolated vertex, $\bigcup_{i\in I} e_i = \mathcal{V}$, the incidence matrix $\mathbf{H} \in \mathbb{R}^{N\times I}$ of N vertices and I hyperedges is defined by

$$h_{ij} = \begin{cases} 1 & if\ v_i \in e_j, \\ 0 & otherwise. \end{cases} \tag{1}$$

For a vertex $v \in \mathcal{V}$ in hypergraph, the degree is defined as the summation of all hyperedges weights $\omega(e)$ attached to vertex v:

$$d(v) = \sum_{e\in\mathcal{E}^h} w(e_i)h(v, e_i), \quad i \in I. \tag{2}$$

Here e is a single hyperedge owning a set of vertices, and $\mathcal{E}$ is the set of hyperedges. The degree of a hyperedge is defined as the number of vertices connected by it:

$$\delta(e) = \sum_{v\in\mathcal{V}} h(v, e_i), \quad i \in I. \tag{3}$$

Moreover, according to the introduction of Bretto *et al.* [37], the adjacency matrix $\mathbf{A}$ of a undirected hypergraph is a symmetric matrix with rows and columns indexed by the vertices of $\mathcal{G}^h$. For all $x, y \in \mathcal{V}, x \neq y$ the entry $a_{x,y} = |\{e \in \mathcal{E}^h; x, y \in e\}|$ and $a_{x,x} = 0$. However, Voloshin *et al.* [38] pointed out it is impossible to reconstruct a hypergraph solely by its adjacency matrix, which means there is no one-to-one correspondence between them. This work uses the incidence matrix to describe hypergraphs to avoid this disagreement, even though it is common to use the adjacency matrix in graph neural networks.

Hypergraph construction in our proposal is divided into two procedures: construction of primary hypergraph $\mathcal{G}_p^h$ and advanced hypergraph $\mathcal{G}_a^h$.

Algorithm 1 Hypergraph Construction

Input: Metro structure and passenger flow datasets $\{\mathbf{X}^t \in \mathbb{R}^{N\times F}\}_{t=1}^{T}$

Output: Incidence matrices of dynamic hypergraphs $\mathbf{H}_d \in \mathbb{R}^{N\times(I+J)}$

for $n = 0$ to N, $i = 0$ to I **do**
  Check if $v_n \in \mathcal{E}_i^h$.
  Build primary hypergraph based on Eq.(1).
end for
for $j_h = 0$ to J_h, $j_d = 0$ to J_d, $j_w = 0$ to J_w **do**
  Preprocess OD matrix
  Set η_t, P_{min} and E_{ps}
  Cluster with DBSCAN algorithm
  Build advanced hyperedges $\mathcal{E}_{h,d,w}^h$
end for
Concatenate primary hypergraph and advanced hyperedges $\mathbf{H}_d = [\mathbf{H}_p, \mathbf{H}_a] \in \mathbb{R}^{N\times(I+J)}$

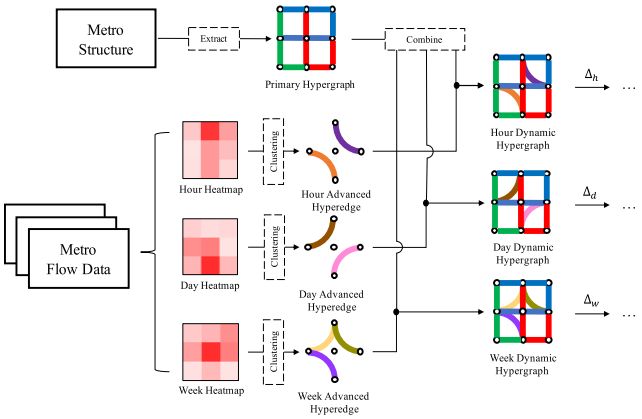


Fig. 3. Schematic diagram of dynamic hypergraph construction. The dynamic hypergraphs are composed of a primary hypergraph and advanced hyperedges. The primary hypergraph is extracted from the metro structure. Vertices represent the stations, and hyperedges in the primary hypergraph represent the lines. The advanced hyperedges are clustered from the flow heatmap of different time spans. Then the primary hypergraph and advanced hyperedges are combined to generate the dynamic hypergraphs.

Primary hypergraph is designed to indicate the fundamental topology of metro network. Vertices $\mathcal{V} = (v_n)_{n\in N}$ in hypergraph are the subway stations with count of N; hyperedges $\mathcal{E}_p^h$ in primary hypergraph are built according to the subway lines. Stations on the same track are supposed to share the common features. According to the metro lines, we have the primary hypergraph $\mathcal{G}_p^h = (\mathcal{V}; \mathcal{E}_p^h = (e_i)_{i\in I})$, where I is a finite set of the indexes of metro lines.

The advanced hypergraph is designed to reveal more spatial information behind the OD pattern of passengers to serve the prediction. OD is short for or origin-destination, which is commonly used in transit to determine travel patterns in an area of interest for a period of time. Since the travel pattern of passengers keeps changing dynamically with time, their OD is varying accordingly. We propose a dynamic mechanism with varying hypergraphs to capture more temporal features, as shown in Fig. 3. Firstly, we analyze passenger flow information of the whole metro network and generate the OD matrices of different time spans. We use the heatmaps to visualize the OD matrices and dig the internal pattern. The region with a high heating value on the map indicates that the passenger flow of the relevant station is huge. The low heating value indicates that the passenger flow there is relatively small. By analyzing the heatmaps, travel OD of different time spans reveals the significant flow of the majority in the corresponding period, that is, what stations the majority of pedestrians will enter and then exit. According to this information, we construct advanced hyperedges $\mathcal{E}_a^h$ through clustering algorithm DBSCAN. The main reason to use DBSCAN is to effectively process noise points and find spatial clusters of arbitrary shapes. Compared with some other clustering methods, such as K-Means, there is no need to preset the number of clusters. The most suitable parameter value of noise threshold η_t, neighborhood points P_{min} and the neighborhood radius E_{ps} [39] are found through corresponding experiments.

Time spans are intercepted into three periods: $\mathcal{T}_h$, $\mathcal{T}_d$ and $\mathcal{T}_w$, which are the hour, day, and week respectively. The hour-scale OD matrices are designed to reveal the flow pattern within one hour; the rush hours and flat periods, for example. The day-scale OD matrices are expected to tell the difference of days, such as weekdays and weekends. The week-scale OD matrices describe the travel patterns of passengers in a longer perspective.

Advanced hyperedges $\mathcal{E}_a^h = (e_j)_{j\in J}$ are therefore built. Here J denotes the index set of advance hyperedges. To denote the hypergraphs $\mathcal{G}^h = (\mathcal{V}; \mathcal{E}^h = (e_k)_{k\in I,J})$, we concat the primary hypergraph and advanced hyperedges. Because of sharing vertices, the incidence matrix of proposed hypergraph can be represented as $\mathbf{H}_d = [\mathbf{H}_p, \mathbf{H}_a] \in \mathbb{R}^{N\times(I+J)}$.

C. Hypergraph Convolution

As introduced, we utilize the graph-based neural network to predict the metro flow. However, graph convolution cannot be applied to hypergraph directly. The hypergraph spectral convolution is derived from the hypergraph learning theory and conventional graph convolution.

Initially, the spectral hypergraph partitioning problem with normalized hypergraph cut is a basic problem of hypergraph theory. In the research of the hypergraph partitioning problem, the hypergraph Laplacian was put forward, which gave us a bridge to hypergraph learning. The optimization of partitioning f could be formulated as [27]:

$$\begin{aligned} &\underset{f\in|V|}{argmin} \ \frac{1}{2}\sum_{e\in\mathcal{E}}\sum_{\{u,v\}\subseteq e}\frac{w(e)}{\delta(e)}\left(\frac{f(u)}{\sqrt{d(u)}}-\frac{f(v)}{\sqrt{d(v)}}\right)^2. \\ &s.t. \quad \sum_{v\in V} f^2(v)=1, \sum_{v\in V} f(v)\sqrt{d(v)}=0 \end{aligned} \tag{4}$$

where u and v are subsets of vertices in a hypergraph, $w(e)$ is the weight of a hyperedge e of a weighted hypergraph, and $\delta(e)$ and $d(v)$ denote the degree of hyperedges and vertices respectively.

With incidence matrix $\mathbf{H}$, weight matrix $\mathbf{W}$, vertex degree matrix $\mathbf{D}_v$, and hyperedge degree matrix $\mathbf{D}_e$, the formulation above can be derived as $f^\top f - f^\top \mathbf{D}_v{}^{-1/2}\mathbf{H}\mathbf{W}\mathbf{D}_e{}^{-1}\mathbf{H}^\top\mathbf{D}_v{}^{-1/2}f$. The formulation can be further expressed as $2f^\top\Delta f$, where

$$\Delta = \mathbf{I} - \mathbf{D}_v^{-1/2}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^\top\mathbf{D}_v^{-1/2}. \tag{5}$$

Δ is a square positive semi-definite matrix with only real eigenvalue. The smallest eigenvalue of Δ is 0. On account of these properties, Δ is regarded as the hypergraph Laplacian, which is the key to spectral hypergraph convolution.

A hypergraph with N vertices has the Laplacian Δ of $\mid N \times N \mid$. By eigen decomposition $\Delta = \Phi\Lambda\Phi^\top$, we could generate a non-negative eigenvalue matrix $\Lambda = diag(\lambda_1, \dots \lambda_n)$ and the orthonormal eigenvector $\Phi = diag(\phi_1, \dots \phi_n)$. Applying the continuous eigenvector Φ as the Fourier bases for a spectral sequence signal $x = (x_1, \dots x_n)$, the Fourier transform is defined as $\hat{x} = \Phi^\top x$, and the inverse Fourier transform is $x = \Phi\hat{x}$. Hence, the spectral convolution of hypergraph signal x and convolution kernel g could be formulated as

$$(x * g)_{HG} = \Phi((\Phi^\top g) \odot (\Phi^\top x)) = \Phi g_\theta(\Lambda)\Phi^\top x, \quad (6)$$

where $\odot$ is the element-wise Hadamard product of two matrices. $\Phi^\top g$ is regarded as a trainable filter of eigenvalue, denoted by $g_\theta(\Lambda)$, which is also the Fourier coefficients. Due to the fact that the time complexity of computing the above equation is $\mathcal{O}(n^2)$ and eigendecomposition of large graphs and hypergraphs is extraordinarily time-consuming, we adopt an efficient calculation method of 1stChebNet proposed by [23]. The Chebyshev polynomial was exploited to approximate the product of filter and hypergraph signal

$$\begin{aligned}(x * g)_{HG} &\approx \sum_{i=0}^{K-1} \Phi\theta_k T_k(\tilde{\Lambda})\Phi^\top x \\ &\approx \sum_{i=0}^{K-1} \theta_k T_k(\tilde{\Delta})x, \end{aligned} \quad (7)$$

where $\tilde{\Delta} = 2\Delta/\Delta_{max} - \mathbf{I}_N$, which is the eigenvector matrix after scaling with range of [-1,1]. The purpose of scaling is to satisfy the condition of truncated expansion of the K_{th} order of Chebyshev polynomial $T_k(x)$. λ_{max} is spectral radius and θ_k is the coefficient of Chebyshev polynomial. The recursive definition of Chebyshev polynomial is $T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$, where $T_0(x) = 1$ and $T_1(x) = x$. As suggested parameters of 1srChebNet, K is set to 1 and λ_{max} is set to 2 because of the scale adaptability. Thus the hypergraph convolution operation could be expressed as

$$(x * g)_{HG} \approx \theta_0\ x - \theta_1 \mathbf{D}_v^{-1/2}\mathbf{H}_d\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}_d^\top\mathbf{D}_v^{-1/2}x. \quad (8)$$

Moreover, in order to restrain the quantity of parameters and prevent overfitting, a parameter θ is defined as $\theta = -2\theta_1 = 2\theta_0\mathbf{D}_v^{1/2}\mathbf{H}_d^{-1}\mathbf{W}^-1\mathbf{D}_e^{-1}(\mathbf{H}_d^\top)^{-1}\mathbf{D}_v^{1/2}$. Then the hypergraph convolution could be further simplified to

$$(x * g)_{HG} \approx \theta\mathbf{D}_v^{-1/2}\mathbf{H}_d\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}_d^\top\mathbf{D}_v^{-1/2}x. \quad (9)$$

The renormalization trick proposed by [23] to deal with the problem of gradient explosion and disappearance is, however, not suitable to be applied to hypergraph convolution directly. According to the renormalization trick $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_N$ and $\tilde{\mathbf{D}}_{ii} = \sum_j \tilde{\mathbf{A}}_{ij}$, which made the vertices in the graph all self-looped. We adapt the renormalization trick to our hypergraph convolution. As mentioned before, we use incidence matrix instead of adjacency matrix, the incidence matrix with self-loop could be denoted as $\tilde{\mathbf{H}}_d = [\mathbf{H}_d, \mathbf{H}_{sl}] \in \mathbb{R}^{N\times(I+N)}$, where

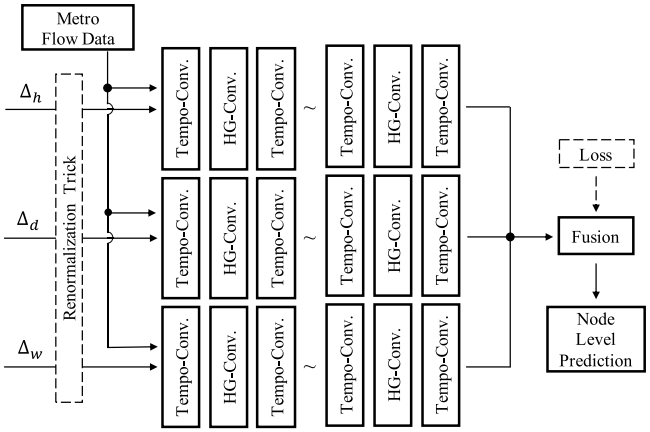


Fig. 4. Structure diagram of spatio-temporal hypergraph convolution neural network. The inputs of the network are hypergraph Laplacian matrices after renormalization trick and the historical metro flow data. Each network contains two blocks. There are two temporal convolution layers and one hypergraph convolution layer inside one block. The output of networks would be fused and finally output the node level prediction.

$\mathbf{H}_{sl}$ is the diagonal identity matrix with the size of vertices. Meanwhile, the degree matrix of vertices is $\tilde{\mathbf{D}}_v = \mathbf{D}_v + \mathbf{I}$. The degree matrix and weight matrix of hyperedges are expanded from $\mathbf{D}_e \in \mathbb{R}^{I\times I}$ to $\tilde{\mathbf{D}}_e \in \mathbb{R}^{(I+N)\times(I+N)}$ and from $\mathbf{W} \in \mathbb{R}^{I\times I}$ to $\tilde{\mathbf{W}} \in \mathbb{R}^{(I+N)\times(I+N)}$ respectively. The hypergraph convolution operation then could be denoted as

$$(x * g)_{HG} \approx \theta\tilde{\mathbf{D}}_v^{-1/2}\tilde{\mathbf{H}}_d\tilde{\mathbf{W}}\tilde{\mathbf{D}}_e^{-1}\tilde{\mathbf{H}}_d^\top\tilde{\mathbf{D}}_v^{-1/2}x. \quad (10)$$

Therefore, the hypergraph convolution layer $f(\mathbf{X}^{(l)}, \Theta^{(l)})$ can be built with

$$\mathbf{X}^{(l+1)} = \sigma_{ReLU}(\tilde{\mathbf{D}}_v^{-1/2}\tilde{\mathbf{H}}_d\tilde{\mathbf{W}}\tilde{\mathbf{D}}_e^{-1}\tilde{\mathbf{H}}_d^\top\tilde{\mathbf{D}}_v^{-1/2}\mathbf{X}^{(l)}\Theta^{(l)}), \quad (11)$$

where $\mathbf{X}^{(l)} \in \mathbb{R}^{N\times F}$ is the input hypergraph signal of l_{th} layer and F is the input channels of feature at each vertices. It is worth mentioning that initial input $\mathbf{X}^{(0)} = \mathbf{X}$. The activation function of ReLU is denoted as σ_{ReLU} to distinguish it from the activation function of S-T blocks of our hypergraph neural network framework.

D. Dynamic Spatio-Temporal Hypergraph Neural Networks

After deriving spectral hypergraph convolution, we explore the temporal convolution to capture the flow feature of the time dimension and compose the spatio-temporal convolution blocks, which is the body part of the whole proposal. The architecture is illustrated in Fig. 4. The network's inputs are renormalized hypergraph Laplacian of three different time spans and the training dataset of metro passenger flow. The major components are spatio-temporal blocks composed of hypergraph convolution layers and temporal convolution layers. The output of spatio-temporal blocks is node-level predictions, which would be fused and then give the prediction results at each station.

The spatio-temporal block has two temporal layers and one spectral hypergraph convolution layer. The spectral hypergraph layer is supposed to extract spatial features, and these features are well described through hypergraph convolution with the representation of the metro topology. The temporal layers are

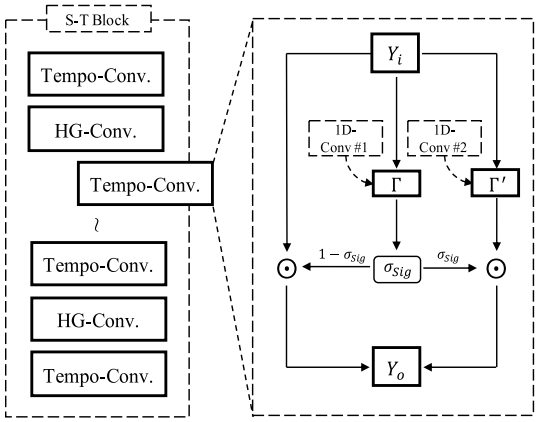


Fig. 5. Detailed structure diagram of temporal convolution layer. Two one dimension dilated convolution operations are applied on the input of the layer. One result would be activated and regarded as the gate. Moreover, the residual structure is integrated into the unit.

designed to extract features to depict passenger flow variation along a time axis. Here the temporal layers employ parallel adapted residual gated linear units (GLU) on each vertex. For a input $\mathbf{Y} = [y_1, y_2, \ldots, y_t]$ with time-series of t, the algorithm first applies two one-dimension convolution operation on the input simultaneously. These two convolution operations are the same in format, but the weights are not sharing. After convolution, one result will be activated with the sigmoid function, which would perform as a gate with a range of (0,1) later, and the other is not activated. The equation could be expressed as

$$\Gamma *_\tau \mathbf{Y} = \Gamma'(\mathbf{Y}) \odot \sigma_{Sig}(\Gamma(\mathbf{Y})), \tag{12}$$

where Γ and Γ' are both one dimension convolution with the same layout and different weights. σ_{Sig} is the Sigmoid activation function. Unlike GLU dealing with NLP problem [40], the input sequence of traffic flow has a much longer range on the time axis than language sentences. Therefore we replace ordinary one dimension convolution with dilated convolution to enable the model to capture features of prolonged distances. The gated structure could handle the problem of vanishing gradients since the first convolution is not activated. Due to the identical dimension of input and output, the residual structure could be integrated into the units as shown in Fig. 5; information then can be transmitted over multiple channels. The temporal convolution is expressed as

$$\begin{aligned}\Gamma *_\tau \mathbf{Y} &= \mathbf{Y} \odot (1 - \sigma_{Sig}(\Gamma(\mathbf{Y}))) + \Gamma'(\mathbf{Y}) \odot \sigma_{Sig}(\Gamma(\mathbf{Y})) \\ &= \mathbf{Y} + (\Gamma'(\mathbf{Y}) - \mathbf{Y}) \odot \sigma_{Sig}(\Gamma(\mathbf{Y})).\end{aligned} \tag{13}$$

It is clear that the temporal information would pass directly with probability of $1 - \sigma_{Sig}(\Gamma'(\mathbf{Y}))$, and pass after abstraction with probability of $\sigma_{Sig}(\Gamma'(\mathbf{Y}))$. Compose the spatial layer and temporal layers, the spatio-temporal convolution blocks can be described as the status update formula:

$$H^{(l+1)} = \Gamma_f^{(l)} *_\tau f((\Gamma_b^{(l)} *_\tau H_d^{(l)}), \Theta^{(l)}), \tag{14}$$

where H is the output tensor of a spatio-temporal convolution block, Γ_f is the front one dimension dilated convolution kernel within the block, and Γ_b is the back one. $\Theta^{(l)}$ is the spectral hypergraph convolution kernel.

The node-level prediction result would output through one full connection layer to map multi-channels into a single one. The results of three different time-spans are then fused with individual weight. The process of training hypergraph neural network is the process of training fusion weights as well:

$$\hat{\mathbf{X}} = \hat{\mathbf{X}}_j \odot \mathbf{W}_j + b, \quad j \in (h, d, w), \tag{15}$$

where $\hat{\mathbf{X}}_j$ is the prediction result of three different time-span, and $\mathbf{W}_j$ is the individual learning weight, which indicates to what extent this span would influence the prediction result. Thus, the final loss function to measure our model could be expressed as

$$\begin{aligned}L(\hat{\mathbf{X}}, \mathbf{W}_j, b) = \sum_t \parallel (\hat{\mathbf{X}}_j^t \odot \mathbf{W}_j + b) - \mathbf{X}^t \parallel^2, \\ \times\, j \in (h, d, w).\end{aligned} \tag{16}$$

Here we use the L_2 loss function to optimize the prediction problem. $\hat{\mathbf{X}}^t$ is the prediction result of timestamp t and the $\mathbf{X}^t$ denotes the ground truth.

IV. Experiments

Substantial experiments have been conducted on historical full-scale metro flow datasets of Beijing and Hangzhou. The details of the dataset would be introduced firstly. Full time experiments and comparison with the existing baselines are first observed to evaluate our model integrally, verifying the practicability and effectiveness of the proposed model. Then we focus on peak-period and flat-period experiments, which are the most instructive and practical applications for the current intelligent transportation system. The ablation experiments are conducted to evaluate each component of the proposed framework. The experiments under abnormal situations and time consumption experiments indicate the practicability of our method.

A. Experiments Setup

1) Datasets: The proposed model is validated on two historical metro passenger flow datasets of Beijing in 2015 (BJMF15) and Hangzhou in 2019 (HZMF19). The raw datasets are used for generating OD matrices, and the preprocessed data are used for flow prediction. Both datasets after data preprocessing would be released on corresponding repositories.[1]

The BJMF15 is the Beijing metro flow dataset collected in 2015. The raw data has attributes of card code, entry station, entry line, entry time, exit station, exit, exit line, and exit time, covering the whole metro net of 327 stations and 22 lines. In July, August, September, November, and December, daily flow data are collected in this dataset. The data after preprocessing is aggregated into every five minutes. Each station has three values every five minutes: exit flow volume, entry flow volume, and flow volume. The flow volume is the summation of exit and entry volume. Therefore, we have three channels of value at the feature dimension of inputs.

[1] https://github.com/JCwww/DSTHGCN

The raw data of HZMF19 is provided in "Ali Global City Computing AI Challenge".[2] The contest offered 25 days of subway card data records of January 2019, involving a total of about 70 million data from 81 subway stations on three lines. The raw data includes nearly the same attributes of BJMF15; hence we preprocessed the data in the same way as BJMF15 and got flow data every five minutes for passenger flow prediction. Similarly, the inputs of the Hangzhou dataset for our model are managed into the same shape.

2) Baselines and Compared Methods: As mentioned in the related works, the comparison methods could be generally divided into two categories of non-graph methods and graph-based methods. For the classical methods, we compared our model with Locally Weighted Linear Regression (LWR), Random Forest (RF) [41] model, Auto-Regression Integrated Moving Average (ARIMA) [10] model, Stacked Autoencoders (SAEs) [42] model, Support Vector Regression (SVR) [43] model, Long Short-Term Memory (LSTM) [44]network and Gated Recurrent Unit (GRU) [45] network. Since these methods are non-graph, we implement these methods on each metro station with one-dimensional vectors of the nodes as the input. And then average the results to evaluate their prediction accuracy.

We compared several graph-based methods as well. Graph Neural Networks (GCN) [18] and Spatial-Temporal Graph Convolution Networks (STGCN) [46] are compared as the baseline of graph-based methods. We also compare the proposed model with a graph convolution based RNN model DCRNN [47]. The DCRNN model also takes spatial and temporal features into account. This method uses bidirectional random walks on the graph to capture spatial features and uses the encoder-decoder framework to model temporal dependence. Moreover, we compared STHGCN and DSTHGCN to evaluate the dynamic mechanism. The dynamic mechanism contains three parallel time spans: hour, day, and week, which are related to the scale of datasets.

3) Experimental Settings: To quantify and evaluate the performance of the proposed model and the other methods, two widely accepted evaluation indicators are adopted: Mean Absolute Errors (MAE) and Mean Absolute Percentage Errors (MAPE). An essential reason for using these two indicators is the fact that there might be zeros in grand truth data of metro flow. All experiments are compiled and executed on OpenI platform with CPU of "Intel(R) Xeon(R) Platinum 8268 CPU @ 2.90GHz" and GPU of "NVIDIA TESLA K80". The model is trained for epochs of 50 and batch size of 64. The learning rate is set to 10^{-3} initially and decreases to 10^{-5} at the final epoch.

B. Full Time Experiments

Full time experiments are first conducted to evaluate our proposed dynamic hypergraph construction model on the whole dataset without segmentation. Advanced hyperedges are composed of hourly, daily and weekly hyperedges, obtained from at least weeks-long datasets. Thus the full time experiments were performed first to observe the effectiveness of the proposed method. Fig. 6 shows a day level heatmap of Beijing, which demonstrates a highly symmetrical pattern. It is also clear that there are many clusters along the diagonal, indicating that many trajectories of passengers are concentrated on short trips.

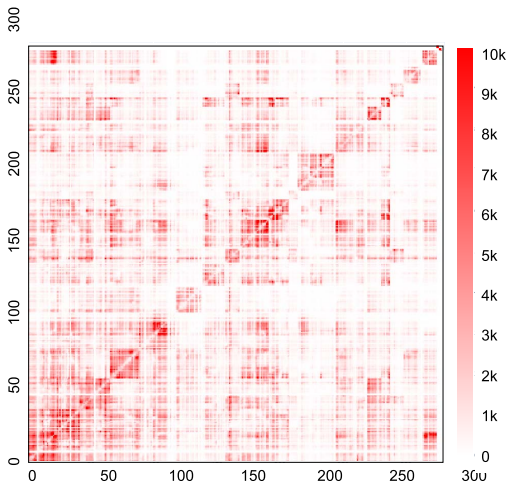


Fig. 6. Day level Heatmap of Beijing on July 1,2015. The x-axis and y-axis denote the code of the station where passengers entered and exited, respectively.

Therefore, we would like to test the specific effect of advanced dynamic hyperedges through full-time experiments. The experimental data is supposed to cover a sufficiently long range to assess the effectiveness of advanced hypergraphs at different time scales. Considering the size of the dataset, for data of each month, we take the ratio of 8: 1: 1 to divide the training set, validation set, and test set. Within one month, we use five-fold cross validation to guarantee a more stable result. Finally, three channels of prediction output are averaged and compared. The experiment results are shown in Table I. The best prediction results are red, the second ones are underlined.

The comparison results of full time experiments on the BJMF15 dataset cover three months: July, September, and December. For the flow data of a month, the quantity of hour-scale, day-scale, and week-scale OD matrices are 24, 7, and 4, respectively. The compared methods are the same ten methods as previous. Generally, the results of the three months follow the same pattern, although the prediction accuracy of September is relatively lower than the other two months. It is clear that graph-based models outperform the methods without the graph. Among these methods, the prediction accuracy of LWR is the lowest, and LSTM and GRU are higher. For the graph-based methods, the proposed STHGCN takes advantage of hypergraph representation and improves accuracy by about 15% than STGCN. The DCRNN model has an outstanding accuracy, which is much better than all the other classical methods. In the category of graph-based methods, the DCRNN can predict more accurately than the basic GCN and STGCN. The utilization of spatial-temporal features of the STGCN model is actually more straightforward than the DCRNN model. Moreover, the proposed DSTHGCN ranks the top, and the performance gain by using dynamic

[2]https://tianchi.aliyun.com/competition/

TABLE I

FULL-TIME EXPERIMENTS OF BEIJING.
(15MIN/30MIN/45MIN)

Methods	July		September		December	
	MAE	RMSE	MAE	RMSE	MAE	RMSE
LWR	22.21/24.32/27.92	35.23/38.54/41.89	23.12/26.01/28.00	36.90/39.35/42.86	23.84/25.03/27.13	36.12/40.67/43.98
ARIMA	18.34/20.12/23.32	29.14/33.37/36.81	19.64/21.31/24.06	30.74/34.30/36.66	18.39/20.52/24.22	30.24/33.81./35.06
SVR	14.73/16.55/18.26	25.24/31.33/32.71	15.89/16.71/17.36	28.30/33.01/34.09	14.12/15.75/16.92	26.56/28.52/28.89
LSTM	10.76/12.27/12.86	21.22/22.33/23.74	11.95/12.56/13.77	23.95/26.43/28.34	11.05/12.33/14.49	21.50/22.79/25.36
GRU	12.98/14.75/17.28	25.49/26.38/28.79	13.95/14.44/16.87	27.45/29.67/31.81	12.81/14.17/16.05	26.98/27.21/29.24
SAEs	11.39/13.41/15.44	23.14/29.09/31.81	12.21/14.84/16.09	24.90/25.72/27.74	11.79/14.31/16.63	22.41/24.84/26.21
GCN	10.88/12.46/12.73	20.93/22.72/24.61	11.00/12.77/13.75	23.98/25.98/28.35	10.99/12.76/13.44	21.48/23.68/25.62
DCRNN	8.41/9.73/11.56	19.43/23.76/25.77	8.72/9.24/12.73	20.94/22.01/25.82	8.83/9.71/11.79	21.52/23.15/26.04
STGCN	9.04/10.29/10.88	19.38/20.49/22.51	10.99/11.95/13.42	21.03/23.03/24.06	9.06/10.65/11.32	20.74/22.22/23.55
STHGCN	7.49/8.82/9.41	17.11/17.36/19.02	9.01/10.92/11.26	17.03/18.74/**19.62**	8.10/9.09/10.19	16.69/17.80/18.84
DSTHGCN	**6.82/7.01/8.24**	**15.24/16.72/18.31**	**7.25/8.19/9.08**	**16.31/17.78**/19.93	**6.93/7.10/9.04**	**15.52/16.15/17.19**

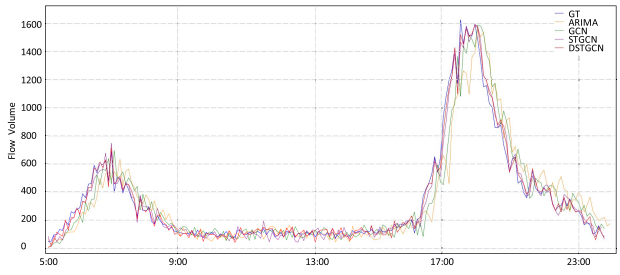


Fig. 7. Prediction value of entry flow on station Guopgongzhuang of Beijing subway Line 9 in a single day.

hypergraph construction for temporal pattern capturing is over 10%.

To illustrate the prediction accuracy of the methods in a single day, we visualize the comparison of the prediction result of entry volume with the ground truth of a typical station: Guogongzuang on Line 9. As shown in Fig. 7, the station has two obvious peak periods during one day. It is clear that our proposed method could predict the trend of flow compared with the other methods, especially on peak periods. Besides, our model responds faster to fluctuation than the other methods, making the model more reliable at solving the prediction problem of time lag.

The HZMF19 dataset has 27 days of flow data for January. Full time experiments are conducted on this dataset as well. The scale of this dataset is smaller than BJMF15, and the data segmentation of the full time experiments is different. Here 24 days of data are used as the training set, two-day data is validation set, and one-day data is test set. Three-fold cross validation is taken to evaluate the model. The results are shown in Table II.

As observed in Table II, the proposed DSTHGCN achieves the best performance among the comparison methods, followed by the DCRNN model. Note that the improvement between hypergraph-based methods and conventional graph-based methods is relatively small compared with the BJMF15 dataset. The reason might be the simpler metro structure of Hangzhou. When the basic topology is not complicated, the difference between the hypergraph and the conventional graph is trivial. Consider an extreme case: when there is only one track in the metro, there will be no difference between the hypergraph and the conventional graph.

TABLE II

FULL TIME EXPERIMENT ON HANGZHOU.
(15MIN/30MIN/45MIN)

Methods	January	
	MAE	RMSE
LWR	29.02/31.33/33.28	52.67/54.94/56.60
ARIMA	26.15/28.07/30.05	46.55/48.85/50.74
SVR	19.00/20.91/21.57	35.01/36.25/37.30
LSTM	16.92/19.13/20.11	30.22/33.83/34.98
GRU	16.99/18.52/20.25	31.96/34.06/36.12
SAEs	14.03/16.19/18.14	31.43/33.75/34.03
GCN	13.93/15.25/17.02	32.27/33.50/35.63
DCRNN	11.82/13.71/15.82	31.41/32.94/36.03
STGCN	12.97/14.65/15.96	28.12/29.02/31.50
STHGCN	11.40/**12.90**/14.63	**26.77**/28.22/29.97
DSTHGCN	**11.08**/13.01/**14.24**	26.99/**27.84/29.53**

C. Peak/Flat Period Experiments

To make further refinements, we made predictions for different periods. Passenger flow prediction of morning and evening rush hours plays a vital role in an intelligent transportation system. For the morning rush hours, we intercepted the data from 7 a.m. to 9 a.m. on weekdays. For the evening rush hours, the period from 6 p.m. to 8 p.m. on weekdays was selected. Besides, experiments on the flat period (2 p.m. - 4 p.m.) are conducted to compare the difference between periods. As mentioned in the dynamic hypergraph construction section, the heatmaps of the morning peak period of Beijing Fig. 8. Comparing the morning peak hours heatmap with day level heatmap Fig. 6, the day level heatmap demonstrates a much more divergent and symmetrical travel pattern. In contrast, the hourly heatmap of the morning is clearly more concentrated.

As shown in the heatmap of the morning period, there are several high flow volume stations of entry, which are presented as crimson banded shape areas. In contrast, the heatmap of a flat period would be much more dispersed and symmetrical.

TABLE III

PEAK EXPERIMENTS OF BEIJING. (15MIN/30MIN/45MIN)

Methods	Morning Peak (7 a.m. - 9 a.m.)		Evening Peak (6 p.m. - 8 p.m.)		Flat Period (2 p.m. - 4 p.m.)	
	MAE	RMSE	MAE	RMSE	MAE	RMSE
LWR	19.98/20.21/21.76	29.01/29.91/30.33	20.89/21.23/22.09	29.91/31.22/32.86	18.40/18.85/19.36	27.29/28.21/29.42
ARIMA	18.79/21.21/23.84	28.03/30.09/31.33	19.45/22.02/23.39	26.91/29.62/30.86	18.21/20.55/22.92	26.62/28.06/30.20
SVR	15.32/17.92/18.23	26.59/26.63/28.09	14.55/16.24/18.70	25.35/26.06/27.42	15.35/17.38/19.55	24.18/25.71/25.78
LSTM	13.26/14.22/16.41	22.95/23.84/24.27	12.80/14.28/16.36	23.42/24.48/25.71	12.05/15.72/17.28	23.12/23.88/25.84
GRU	13.05/14.41/16.10	22.55/23.75/25.75	12.49/15.02/16.71	22.14/23.40/25.65	13.34/15.55/15.81	23.11/24.20/24.14
SAEs	10.81/12.60/13.49	20.92/22.69/24.73	11.10/13.04/15.71	21.22/23.06/24.73	11.04/12.52/14.20	21.01/23.48/25.71
GCN	9.24/8.83/10.29	19.46/20.36/21.04	9.46/9.16/11.40	20.26/20.80/23.57	8.23/9.81/10.93	19.61/20.85/22.40
DCRNN	7.34/8.63/9.35	15.57/16.43/17.34	8.79/9.23/9.52	16.22/17.34/18.46	7.20/7.94/8.85	14.45/15.56/19.47
STGCN	8.21/8.73/9.66	16.22/17.31/19.02	8.13/8.49/9.42	16.07/17.26/18.91	7.80/8.44/9.29	15.84/17.18/18.83
STHGCN	5.91/6.21/7.33	14.49/15.60/16.59	6.24/6.41/8.08	14.61/15.31/16.71	5.49/**5.92**/6.49	**14.04** /15.05/16.42
DSTHGCN	**5.70/6.09/6.76**	**14.30/15.24/16.43**	**5.82/6.14/7.23**	**14.04/15.29/16.57**	**5.25**/6.18/**6.27**	14.83/**14.90/15.70**

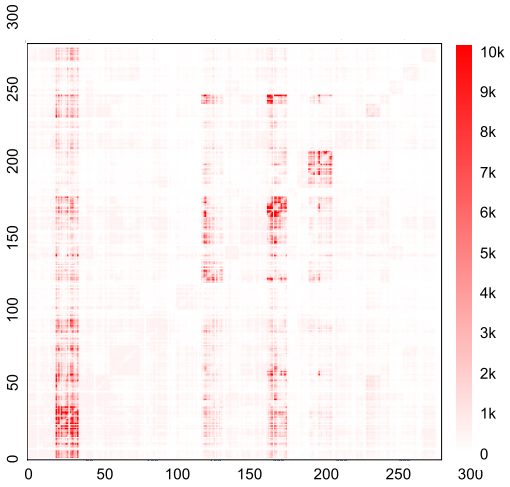


Fig. 8. Morning peak passenger flow Heatmap of Beijing on July 1, 2015. The x-axis denotes the code of subway stations where passengers entered, and the y-axis denotes the code of subway stations where passengers exited. The travel OD of passengers shows a specific aggregation, which supports constructing advanced hyperedges.

The input of the proposed model is in the shape of $\mathbb{R}^{N\times F\times T}$. Where N is the number of metro stations, T is the time slides, and F is the input channel number of features. Input data of BJMF15 and HZMF19 both have three channels at each station: entry, exit, and turnover. To make the experimental results more concise, we represent the average value of these three channels and compare them with each other. Moreover, the results include prediction accuracy of 15-minute, 30-minute, and 45-minute to evaluate the prediction ability of the proposed model under different time spans. The specific experimental results are shown in Table III.

The table compares the subway flow prediction accuracy among ten methods in terms of morning peak, evening peak, and flat period. The comparison methods are divided into two categories of non-graph methods and graph-based methods. Both MAE and RMSE would indicate the more accurate the model predicts with a lower value. It is clear that compared with the figure of peak period integrally, that of the flat period is slightly higher. LWR has the lowest prediction accuracy among the six non-graph methods, followed by ARMIA, SAEs, and SVR successively. LSTM and GRU could predict more accurately on our dataset; both of them are effective solutions to overcome short-term memory problems. However, there is still a gap between the classical methods and graph-based methods. Basic GCN has performed better than all non-graph methods, although it has the worst accuracy among graph methods. STGCN moves forward with the spatial and temporal features. However, DCRNN makes better use of spatio-temporal features and achieves better results than STGCN. DSTHGCN ranks the top arriving MAE at around 6 and RMSE at around 15, followed by STHGCN. The gap between STHGCN and DSTHCN is relatively small compared with the full-time experiment. Due to the design of multiple time-span, the DSTHGCN achieves limited performance in a data set with peak hours.

TABLE IV

EXPERIMENTS ON MORNING PEAK OF HANGZHOU. (15MIN/30MIN/45MIN)

Methods	Morning Peak (7 a.m. - 9 a.m.)		Flat Period (2 p.m. - 4 p.m.)	
	MAE	RMSE	MAE	RMSE
LWR	14.78/15.71/16.90	22.88/24.34/25.30	13.04/14.73/15.06	22.32/23.25/26.06
ARIMA	13.07/13.24/13.32	21.86/21.75/23.78	12.93/12.86/13.28	20.93/21.28/22.14
SVR	11.80/12.85/13.49	19.64/18.67/19.52	12.00/12.66/12.53	19.00/19.33/20.32
LSTM	12.61/12.47/13.38	20.63/21.59/21.84	12.46/12.32/13.39	19.70/21.04/22.76
GRU	12.30/12.56/13.43	20.25/20.49/21.62	12.46/12.61/13.62	19.72/20.76/20.22
SAEs	11.15/12.06/13.74	18.24/19.47/20.48	12.39/12.90/13.72	19.69/20.50/21.14
GCN	12.47/12.50/12.63	19.90/19.34/20.83	12.48/12.47/12.83	19.06/19.88/21.73
DCRNN	11.45/12.10/12.63	18.99/19.34/21.13	11.18/12.44/12.87	18.06/18.88/20.34
STGCN	11.63/11.96/12.34	19.70/19.92/20.06	11.87/12.07/12.18	18.58/19.79/19.69
STHGCN	11.18/11.35/**11.27**	18.44/**17.80**/18.36	11.32/11.30/11.43	16.73/19.25/**17.56**
DSTHGCN	**10.91/11.18**/11.40	**17.32**/18.17/**18.32**	**10.87/10.99/11.19**	**16.58/17.52**/18.06

To evaluate the methods on the different metro systems, we compare the flow prediction results on the Hangzhou dataset. The compared methods are the same as the experiment on the Beijing dataset, and the comparison periods are the morning peak and flat period. As illustrated in Table IV, the graph-based methods are still generally more accurate with the smaller dataset than non-graph methods. Both DCRNN and STGCN take advantage of temporal and spatial features. The accuracy of these two methods is close, but both

TABLE V
PREDICTION OF ABLATION EXPERIMENTS. (15MIN/30MIN/45MIN)

Ablation Combinations		July	
		MAE	RMSE
Temporal layer only	1 Temp-layer	14.34/20.77/32.29	29.22/44.61/72.01
	2 Temp-layers	13.44/17.24/22.42	27.79/36.08.48.09
HG-Conv layer only	1 HGconv-layer	23.17/27.16/31.70	65.99/68.08/71.68
	2 HGconv-layers	24.97/29.02/34.08	69.20/71.07/74.32
Mixed layers	1 Temp + 1 HGconv	12.05/15.35/19.79	26.87/35.23/46.52
	2 Temp + 2 HGconv	10.64/12.49/15.77	21.43/25.96/30.37
Ours	4 Temp + 2 HGconv	**8.25/9.19/10.08**	**18.31/21.78/27.93**

have a significant improvement compared to GCN. Compared with LWR, DSTHCGN has improved the MAE by about 15%. Compared with graph-based methods, the improvement seems less significant especially comparing with STHGCN. The reason might be that hypergraphs are more suitable for representing redundant and complex data. Hangzhou has only three subway lines and eighty stations, which is simpler in structure than the Beijing subway.

D. Ablation Experiments

The design of the ablation experiment is to remove some components or processes in the neural network to test the effect of the removed part on the experimental result. Therefore, we conducted several different combined experiments on the temporal convolution layers and the hypergraph convolution layers of the neural network. The results of the July dataset are shown in Table V.

We conducted experiments of a temporal layer only and a hypergraph convolution layer only first. The accuracy of using the temporal layer only is better than using the hypergraph convolution layer only, which is not surprising because the flow prediction is a time series problem. Two temporal layers combination performs slightly better than one temporal layer. In comparison, two hypergraph convolution layers combination has a slightly lower accuracy than one hypergraph convolution layer, indicating that simply stack layers in a neural network cannot guarantee better performance. When mix the temporal layers and hypergraph convolution layers up, the best result comes from our final combination, which is two blocks of six layers in total. The result verifies the effectiveness of the proposed framework.

E. Prediction Under Abnormal Situation

Passenger flow prediction under abnormal situations is an important experiment to evaluate the robustness of a method. The abnormal value might due to large-scale activities in the surrounding area, subway maintenance, and many other events. We would not classify and analyze the causes of abnormalities and only test the accuracy of the model when the data is under abnormal situations.

We use the Robust Principal Component Analysis (RPCA) method to detect abnormal passenger flow. In the process of abnormal detection, we refer to paper [48]. The RPCA is a common main component extraction algorithm. The algorithm is widely used in abnormal detection of time series data [49]. The essence of RPCA is a matrix factorization algorithm. The goal is to decompose the input data into the low-rank approximation. The essence of low-rank estimation is to project the highly correlated rows of the matrix to a lower-dimensional linear space, which achieves a dimensionality reduction and smoothing function. While reducing the dimensionality, the redundant information is removed, and the matrix features are extracted. After extracting the main component, the remaining sparse matrix and the noise are obtained. When doing anomaly detection here, we could think that the low-rank matrix can restore most of the input sequence. The characteristics of abnormal points should be represented in the sparse matrix, and the noise matrix is obtained. In the RPCA algorithm, our threshold parameter is defined as:

$$\delta = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\mathbf{X}-\mathbf{L})}, \tag{17}$$

where N is the total number of stations, $\mathbf{X}$ is the original input data matrix, and $\mathbf{L}$ is the low-rank matrix, which is the ideal matrix of the passenger flow. Through the RPCA algorithm, we can detect the abnormal value and the time when the abnormality occurs.

We use the historical flow of three months as the experimental data. The experimental results of the average prediction accuracy of the corresponding abnormal time after using RPCA for anomaly detection are given in Table VII.

Among the evaluate index, RMSE is more affected by outliers. It can be illustrated from Table VII that graph-based methods have much better performance than non-graph methods. This might due to the graph-based methods can aggregate the features of neighbor nodes through edges in the process of prediction. The proposed DSTHGCN and DCRNN have similar performance on MAE, and both are better than other methods. The proposal is also superior to other methods to varying degrees on RMSE, which shows that the prediction accuracy of the proposed method under abnormal conditions is still impressive.

F. Time Consumption Experiments

As for a traffic flow prediction method, the time consumption is important, determining whether the method can be applied to rapid real-time prediction in the intelligent transportation system.

For machine learning methods, the computational time involves training time and testing time, i.e., prediction time. Many factors would affect training time. Generally speaking, the training time complexity of machine learning methods is roughly related to epochs, dataset size, batch size, and the detailed framework structure of various methods. Therefore, it is usually hard to compare the methods of various frameworks in an absolutely fair way.

Testing time is more relevant to the practical application of the methods. Similar to the training time, the testing time complexity is also tricky to be defined accurately. It is roughly positively correlated with the quantity of weights in the network. We experimented with passenger flow in the future 15 minutes to compare the performance of these methods in terms of prediction time. The results of each method are

TABLE VI
TESTING TIME CONSUMPTION

Methods	LWR	ARIMA	SVR	LSTM	GRU	SAEs	GCN	STGCN	DCRNN	**STHGCN**	**DSTHGCN**
BJMF15	739.25	874.52	941.63	552.63	461.07	323.93	28.96	**10.28**	53.87	11.32	15.64
HZMF19	175.42	221.43	236.12	118.70	111.74	76.23	22.71	**7.71**	41.46	9.43	12.27

TABLE VII
PREDICTION UNDER ABNORMAL SITUATION

Methods	July		August		September	
	MAE	RMSE	MAE	RMSE	MAE	RMSE
LWR	63.82	106.37	67.31	113.24	59.42	102.74
ARIMA	54.31	97.91	55.19	98.13	48.93	87.55
SVR	53.14	100.17	55.82	105.84	47.29	93.76
LSTM	45.04	76.55	43.76	77.49	40.90	66.46
GRU	49.76	83.21	47.98	86.02	42.03	71.87
SAEs	47.86	79.34	45.70	82.47	44.71	67.93
GCN	28.42	58.31	30.83	53.65	28.94	55.73
STGCN	23.84	52.93	24.90	51.27	21.88	49.01
DCRNN	22.73	41.69	18.81	37.04	16.13	34.92
STHGCN	21.77	39.58	19.08	35.92	17.77	31.56
DSTHGCN	**19.21**	**38.46**	**18.05**	**33.97**	**15.09**	**30.28**

compared in Table VI. The configuration of the experiment platform has a significant influence on the results of the time consumption experiment.

It is supposed to be noted that the non-graph methods need to predict all stations on the entire metro network separately; that is, the prediction results require a loop with the size of the number of stations. In comparison, the graph-based methods can output the prediction results of the entire metro network at once, which is more efficient. The computational time results of Table VI have indicated the considerable time gap between non-graph methods and graph-based methods. The graph topology of the Beijing subway has about 320 vertices, while the Hangzhou subway has 80 stations. For the different scale of graphs, the DSTHGCN has a longer testing time than STHGCN and DCRNN, which is due to the more complex framework. The DCRNN has the shortest time consuming among the graph-based methods. However, this time consumption of DSTHGCN can still meet the real-time requirements of intelligent transportation systems.

V. CONCLUSION

This paper has addressed the metro flow prediction problem via dynamic hypergraph representation and spatio-temporal hypergraph convolution neural network. The proposed method takes advantage of the unique structure of the subway and the spatial correlation between passenger flow and traveling OD to achieve more accurate metro flow prediction. Experiments have been conducted on factual datasets of Beijing and Hangzhou. The experiments cover five major categories and eleven compared methods. Results show that our model outperforms other non-graph and graph-based state-of-the-art methods. In the future, we will optimize the framework of the proposed method. We also would like to explore the hypergraph theory and applications in intelligent transportation system more comprehensively.

REFERENCES

[1] B. Leng, J. Zeng, Z. Xiong, W. Lv, and Y. Wan, "Probability tree based passenger flow prediction and its application to the Beijing subway system," *Frontiers Comput. Sci.*, vol. 7, no. 2, pp. 195–203, Apr. 2013.
[2] Y. Sun, G. Zhang, and H. Yin, "Passenger flow prediction of subway transfer stations based on nonparametric regression model," *Discrete Dyn. Nature Soc.*, vol. 2014, pp. 1–8, Apr. 2014.
[3] Y. Sun, B. Leng, and W. Guan, "A novel wavelet-SVM short-time passenger flow prediction in Beijing subway system," *Neurocomputing*, vol. 166, pp. 109–121, Oct. 2015.
[4] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," 2019, *arXiv:1901.00596*. [Online]. Available: http://arxiv.org/abs/1901.00596
[5] E. I. Vlahogianni, J. C. Golias, and M. G. Karlaftis, "Short-term traffic forecasting: Overview of objectives and methods," *Transp. Rev.*, vol. 24, no. 5, pp. 533–557, Sep. 2004.
[6] M. S. Ahmed and A. R. Cook, "Analysis of freeway traffic time-series data by using Box-Jenkins techniques," *Transp. Res. Rec.*, vol. 722, pp. 1–9, 1979.
[7] B. M. Williams, P. K. Durvasula, and D. E. Brown, "Urban freeway traffic flow prediction: Application of seasonal autoregressive integrated moving average and exponential smoothing models," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1644, no. 1, pp. 132–141, Jan. 1998.
[8] S. Lee and D. B. Fambro, "Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1678, no. 1, pp. 179–188, Jan. 1999.
[9] B. M. Williams, "Multivariate vehicular traffic flow prediction: Evaluation of ARIMAX modeling," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1776, no. 1, pp. 194–200, Jan. 2001.
[10] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results," *J. Transp. Eng.*, vol. 129, no. 6, pp. 664–672, Nov. 2003.
[11] W. Huang, G. Song, H. Hong, and K. Xie, "Deep architecture for traffic flow prediction: Deep belief networks with multitask learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 5, pp. 2191–2201, Oct. 2014.
[12] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 865–873, Apr. 2015.
[13] Z. Li *et al.*, "A hybrid deep learning approach with GCN and LSTM for traffic flow prediction," in *Proc. IEEE Intell. Transp. Syst. Conf. (ITSC)*, Oct. 2019, pp. 1929–1933.
[14] L. Liu and R.-C. Chen, "A novel passenger flow prediction model using deep learning methods," *Transp. Res. C, Emerg. Technol.*, vol. 84, pp. 74–91, Nov. 2017.
[15] M. Ni, Q. He, and J. Gao, "Forecasting the subway passenger flow under event occurrences with social media," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 6, pp. 1623–1632, Jun. 2017.
[16] Y. Wei and M.-C. Chen, "Forecasting the short-term metro passenger flow with empirical mode decomposition and neural networks," *Transp. Res. C, Emerg. Technol.*, vol. 21, no. 1, pp. 148–162, Apr. 2012.
[17] Y. Li, X. Wang, S. Sun, X. Ma, and G. Lu, "Forecasting short-term subway passenger flow under special events scenarios using multiscale radial basis function networks," *Transp. Res. C, Emerg. Technol.*, vol. 77, pp. 306–328, Apr. 2017.
[18] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2009.
[19] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," 2013, *arXiv:1312.6203*. [Online]. Available: http://arxiv.org/abs/1312.6203
[20] J. Atwood and D. Towsley, "Diffusion-convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 1993–2001.
[21] M. Niepert, M. Ahmed, and K. Kutzkov, "Learning convolutional neural networks for graphs," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 2014–2023.
[22] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3844–3852.

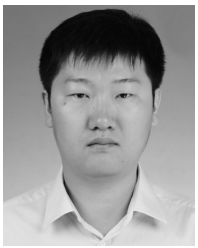
[23] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*. [Online]. Available: http://arxiv.org/abs/1609.02907
[24] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, "Gated graph sequence neural networks," 2015, *arXiv:1511.05493*. [Online]. Available: http://arxiv.org/abs/1511.05493
[25] Y. Seo, M. Defferrard, P. Vandergheynst, and X. Bresson, "Structured sequence modeling with graph convolutional recurrent networks," in *Proc. Int. Conf. Neural Inf. Process.* Cham, Switzerland: Springer, 2018, pp. 362–373.
[26] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*. [Online]. Available: http://arxiv.org/abs/1710.10903
[27] D. Zhou, J. Huang, and B. Schölkopf, "Learning with hypergraphs: Clustering, classification, and embedding," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 1601–1608.
[28] D. Arya and M. Worring, "Exploiting relational information in social networks using geometric deep learning on hypergraphs," in *Proc. ACM Int. Conf. Multimedia Retr.*, Jun. 2018, pp. 117–125.
[29] P. Li and O. Milenkovic, "Inhomogeneous hypergraph clustering with applications," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 2308–2318.
[30] J. Yan, C. Zhang, H. Zha, W. Liu, X. Yang, and S. M. Chu, "Discrete hyper-graph matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1520–1528.
[31] M. Leordeanu and C. Sminchisescu, "Efficient hypergraph clustering," in *Proc. Artif. Intell. Statist.*, 2012, pp. 676–684.
[32] M. Leordeanu, A. Zanfir, and C. Sminchisescu, "Semi-supervised learning and optimization for hypergraph matching," in *Proc. Int. Conf. Comput. Vis.*, 2011, pp. 2274–2281.
[33] L. Su, Y. Gao, X. Zhao, H. Wan, M. Gu, and J. Sun, "Vertex-weighted hypergraph learning for multi-view object classification," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 2779–2785.
[34] L. An, X. Chen, S. Yang, and X. Li, "Person re-identification by multi-hypergraph fusion," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 11, pp. 2763–2774, Nov. 2017.
[35] N. Yadati, M. Nimishakavi, P. Yadav, V. Nitin, A. Louis, and P. Talukdar, "HyperGCN: A new method for training graph convolutional networks on hypergraphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 1509–1520.
[36] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao, "Hypergraph neural networks," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 3558–3565.
[37] A. Bretto, "Hypergraph theory," in *An Introduction: Mathematical Engineering*. Cham, Switzerland: Springer, 2013.
[38] V. I. Voloshin, *Introduction to Graph and Hypergraph Theory*. Hauppauge, NY, USA: Nova Science Publishers, 2009.
[39] J. Sander, M. Ester, H.-P. Kriegel, and X. Xu, "Density-based clustering in spatial databases: The algorithm GDBSCAN and its applications," *Data Mining Knowl. Discovery*, vol. 2, no. 2, pp. 169–194, Jun. 1998.
[40] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 1243–1252.
[41] A. Liaw and M. Wiener, "Classification and regression by randomforest," *R News*, vol. 2, no. 3, pp. 18–22, 2002.
[42] X. Yuan, B. Huang, Y. Wang, C. Yang, and W. Gui, "Deep learning-based feature representation and its application for soft sensor modeling with variable-wise weighted SAE," *IEEE Trans. Ind. Informat.*, vol. 14, no. 7, pp. 3235–3243, Jul. 2018.
[43] H. Drucker, C. J. Burges, L. Kaufman, A. J. Smola, and V. Vapnik, "Support vector regression machines," in *Proc. Adv. Neural Inf. Process. Syst.*, 1997, pp. 155–161.
[44] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[45] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, *arXiv:1412.3555*. [Online]. Available: http://arxiv.org/abs/1412.3555
[46] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," 2017, *arXiv:1709.04875*. [Online]. Available: http://arxiv.org/abs/1709.04875
[47] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," 2017, *arXiv:1707.01926*. [Online]. Available: http://arxiv.org/abs/1707.01926
[48] X. Wang, Y. Zhang, H. Liu, Y. Wang, L. Wang, and B. Yin, "An improved robust principal component analysis model for anomalies detection of subway passenger flow," *J. Adv. Transp.*, vol. 2018, pp. 1–12, Aug. 2018.
[49] R. Kwitt and U. Hofmann, "Unsupervised anomaly detection in network traffic by means of robust PCA," in *Proc. Int. Multi-Conf. Comput. Global Inf. Technol. (ICCGI)*, Mar. 2007, p. 37.

Jingcheng Wang received the bachelor's degree in the Internet of Things engineering from the Beijing University of Technology, China, in 2017, where he is currently pursuing the Ph.D. degree. His research interests include intelligent transportation systems and hypergraph theory.

Yong Zhang (Member, IEEE) received the Ph.D. degree in computer science from the Beijing University of Technology in 2010. He is currently an Associate Professor in computer science with the Beijing University of Technology. His research interests include intelligent transportation systems, big data analysis and visualization, and computer graphics.

Yun Wei received the B.S. degree from the Nanjing University of Aeronautics and Astronautics in 2005 and the M.S. and Ph.D. degrees from Southeast University in 2008 and 2013, respectively. He is currently the Vice Director of the National Engineering Laboratory, Urban Railway Green and Safe Construction. His research interests include intelligent vision analysis and pattern recognition. He is a member of the Comprehensive Intelligent Transportation Systems Technical Committee of the Chinese Association of Automation. He received the Beijing Science and Technology New Star Award and the Exceptional Talent Award of Beijing.

Yongli Hu (Member, IEEE) received the Ph.D. degree in computer science from the Beijing University of Technology, China, in 2005. He is currently a Professor in computer science with the Beijing University of Technology. His research interests include computer graphics, pattern recognition, and multimedia technology.



Xinglin Piao received the Ph.D. degree from the Beijing University of Technology, Beijing, China, in 2017. He is currently a Post-Doctoral Researcher with the Pengcheng Laboratory. His research interests include intelligent traffic, pattern recognition, and multimedia technology.

Baocai Yin (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in computational mathematics from the Dalian University of Technology, Dalian, China, in 1985, 1988, and 1993, respectively. He is currently a Professor with the Beijing Key Laboratory of Multimedia and Intelligent Software Technology, Faculty of Information Technology, Beijing University of Technology. His research interests include multimedia, image processing, computer vision, and pattern recognition.