Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 1604268, 10 pages
https://doi.org/10.1155/2021/1604268

Research Article

Human Origin-Destination Flow Prediction Based on Large Scale Mobile Signal Data

Qiuyang Huang,[1] **Yongjian Yang**,[1] **Yuanbo Xu**,[1] **En Wang**,[1] **and Kangning Zhu**[2]

[1]*Department of Computer Science and Technology, Jilin University, Changchun, Jilin 130012, China*
[2]*Department of Software Engineering, Jilin University, Changchun, Jilin 130012, China*

Correspondence should be addressed to Yuanbo Xu; yuanbox15@hotmail.com

Received 29 June 2021; Accepted 30 August 2021; Published 29 September 2021

Academic Editor: Wenzhong Li

The human origin-destination (OD) flow prediction is of great significance for urban safety control, stampede prevention, disease transmission control, urban planning, and many other aspects. Most of the existing methods generally divide the urban area into grids and use vehicle GPS trajectories and metrocard check-in data, combined with machine learning or deep learning models to predict human OD flow. However, these kinds of methods are challenging to capture fine-grained human mobility patterns. Moreover, these methods usually deviate from the actual human OD transfer patterns on a citywide scale due to the particularity of different datasets. To this end, in this paper, we use large-scale mobile phone signal data to achieve human OD flow prediction between the coverage of varying signal base stations. Many signal base stations are distributed in urban geographical space, collecting all the mobile phone user's location information to obtain large-scale fine-grained unbiased human OD flow data. Due to the lack of natural topology structure between base stations, this paper adopts a TGCN model combined with a graph fusion module to pretrain the dynamic population distribution prediction task. The parameters of the graph fusion module are employed to capture the different semantic information in the proposed hybrid machine learning method and finally achieve citywide human OD flow prediction. Extensive experiments on the real-world signal datasets in Changchun, China, demonstrate the effectiveness of our model.

1. Introduction

The O-D (origin-destination) flow prediction of urban residents is beneficial to grasp the dynamic trend of human mobility in urban geospatial space. It can refine and locate the flow in the face of sudden disasters, such as epidemic outbreaks, which helps to improve the vitality and responsiveness of the city. From the perspective of urban management, the OD flow between base stations is predicted and analyzed, achieving the urban population flow monitoring of base station spatial granularity, helps urban managers to study the residents' travel behavior mode, and designs and manages the urban traffic system [1, 2].

For decades, many studies on urban OD flow have been conducted to provide long-term guidance and short-term strategies for urban planning and transportation development. Existing studies usually use subway flow data and vehicle flow data as data sources, and the description of population flow is relatively simple, which cannot capture the OD flow of other travel modes. Besides, the existing studies usually focus on modeling the travel patterns from the systems which have natural topology structure (such as urban subway and road-network systems), while ignoring the spatial correlations of systems without natural topology structure [3, 4]. Figure 1 gives an example to illustrate the user, trajectory, OD pair, and signal data. When mobile phone users move around the city, mobile signal data records the visited base station sequence and time, so that we can obtain the trajectories and OD pairs of all users.

Mobile signal data provides a new solution to the problem of human OD flow prediction. In recent years, with the popularity of smartphones, the mobile signal data

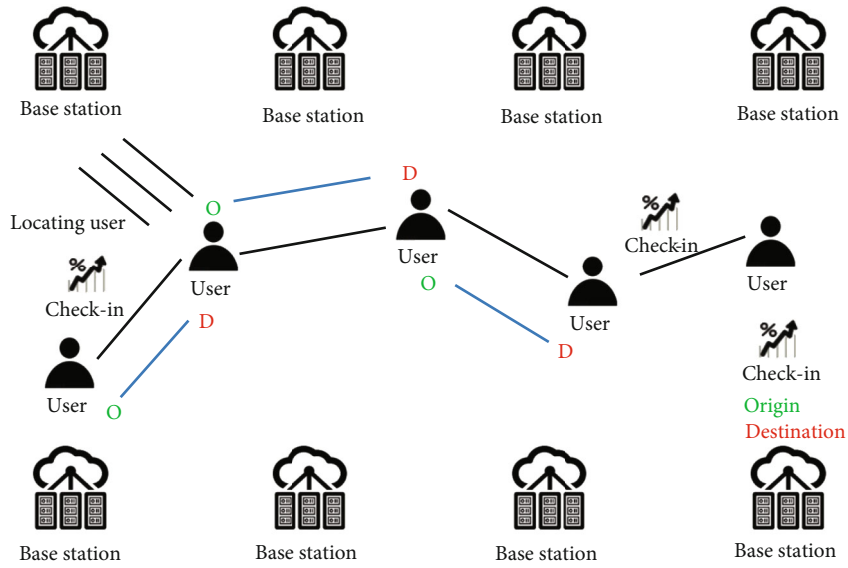

Figure 1: An example to illustrate the user, trajectory, OD pair, and signal data. When a user uses mobile phone to receive or make calls, send or receive messages, surf the Internet, and even move between the coverage of different base stations, the mobile phone will interact with the base station to obtain corresponding communication services, while the base station will passively generate mobile phone signaling data records, so that we can obtain the trajectories and OD pairs of all mobile users.

generated by the interaction between mobile devices and base stations [5] has become a data source with broad coverage, high data accuracy, and easy access. Cell phones are carried by most of the population, enabling mobile signal data to simulate all kinds of traffic modes between different OD areas, which is very suitable for traffic prediction scenarios. Besides, the OD flow between base stations is predicted and analyzed, which can achieve the monitoring of the urban population flow in base station spatial granularity, help urban manages study travel behavior mode of residents, and design and manage the urban traffic systems [6, 7].

However, urban base stations are usually installed in residential buildings and do not have natural topological structures like urban road-network or metrosystem; the OD flow links between different areas are highly dynamic. Therefore, the premise of applying the graph neural network method to extract spatial structure features in the prediction of OD flow between base stations is to find a reasonable graph structure between base stations.

Along with this line, in this paper, we hope to obtain a reasonable graph structure between base stations, a pretraining model with a graph fusion module is used to predict the number of residents, and the resulting base station graph structure is applied to the graph embedding algorithm using the idea of transfer learning to generate the embedding vectors for each base station. The embedding vectors of base stations combined with the manually extracted features (including POI and historical human mobility features) are input into the basic prediction model (such as Lasso, Random Forest, and LightGBM), to predict the OD flow between base stations. This work provides a new way to explore the effectiveness of reasonable graph structure between base stations for OD flow prediction. The contributions of this paper are summarized as follows:

(i) This paper uses large-scale mobile phone signal data to extract human mobility data, then construct training and testing dataset for OD flow prediction task. Compared with the traditional grid division-based methods, mobile signal has the advantages of large number of users, wide coverage, and fine granularity

(ii) In order to obtain the adjacency relationship between base stations in cities, this paper constructs the adjacency matrix of population flow between base stations based on the population movement times between base stations, constructs the distance adjacency matrix based on the longitude and latitude distances between base stations, and then calculates the Jaccard correlation coefficient [8] (which is used to compare the similarity and difference of POI distribution between two base stations, the greater the value is, the higher the similarity) of POI between base stations after matching urban POI data to base stations and generates the Jaccard correlation coefficient matrix. Through the weighted fusion method, the graph structure between the city base stations is finally obtained

(iii) Specifically, the base station graph structure in the pretraining model is used to train the node2vec model using the idea of transfer learning, and each base station is represented as a set of embedding vectors. Combined with the POI distribution characteristics around the base station, historical transfer traffic characteristics, and network structure characteristics, OD flow prediction is carried out, and the effectiveness of the graph embedding method is verified through experiments

2. Related Work

The main problem of human OD flow prediction is to predict the transfer flow between origin and destination areas [9]. In the early days, mathematical models such as the moving autoregressive model ARIMA [10], Kalman filter algorithm [11], and its extended algorithm were mainly used for OD traffic prediction. In recent years, various machine learning and deep learning models have been widely used in traffic OD flow prediction. These models represent the complex nonlinear relationship between different variables and provide a new prediction method [12, 13].

Gong [9] et al. propose a dual-track model OLS-DT, which learns traffic laws from two perspectives. OLS-DT can learn the steady change trend of the human travel flow and show better performance in drastic changes in the OD flow. LP Zapata et al. [14] use Bayesian inference to build a mathematical model; the Monte Carlo algorithm generates a large number of random samples; these samples will be accepted or rejected according to the Metropolis-Hasting criteria, the arithmetic average of all accepted samples as the final results. A model experiment was carried out using the transportation network in the southeastern district of Quito. Duan [15] et al. proposed a convolution-based LSTM model, which added the correlation between travel time and OD traffic and nested the city grid with roads to predict the taxi traffic between different OD areas in the city.

Generally speaking, with the development of big data computing, traffic data is becoming more and more diversified. Machine learning models and deep learning models play an increasingly important role in the current nonlinear dynamic space-time problems, and they have become an effective means to improve OD traffic prediction.

Various end-to-end neural network models have been continuously proposed and applied in appropriate business scenarios [16]. However, it is difficult for traditional deep learning methods to process non-Euclidean structured data, such as structured graph data. In practical problems, graph structure data is ubiquitous, such as bipartite graphs of user-item interactions in the recommended field, social networks, road network data in the urban field. Therefore, graph neural network (GNN) was proposed [17].

Graph neural network is a neural network used to process graph structure data [18]. This article mainly uses the graph embedding (graph embedding) method, including the graph convolutional neural network (GCN) and the graph autoencoder [19]. Graph convolutional neural networks are divided into two categories: spectrum-based methods and space-based methods. The former regards graph convolution as a filtering operation, and the data noise is removed after the convolution operation. The graph embedding method mainly learns the vector representation of the attribute graph and usually uses a set of embedding vectors to represent the nodes in the graph or the entire graph information. This method can well input graph data into machine learning algorithms. Commonly used methods include matrix factorization and random walk [20].

The graph neural network has been applied in various fields. In recommendation systems, Wang et al. [21] used the interaction between users and items to construct a bipartite graph, used graph embedding methods to obtain vector representations between users and items, and more effectively estimate user preferences for items. In the field of urban computing, Guo Shengnan et al. [22] proposed a spatiotemporal cycle convolutional network model, aiming at the short-term law of population flow, including the dependence between daily and weekly flows and unified modeling to predict urban area flow. Zhao et al. [23] extracted a time graph convolutional neural network, combined with GCN and GRU (gate recurrent unit) to capture traffic data's spatial and temporal dependence, thereby predicting traffic flow. In the field of computer vision. Fei-Fei et al. [24] used a graph convolution module to process the input of graph structure on text description generating an image, which enhanced the image generation effect.

In general, because graph neural networks can effectively capture spatial features, the research of graph neural networks has become a hot spot in various fields in recent years.

3. Framework

In this section, we introduce the detailed framework of the proposed OD flow prediction model, as shown in Figure 2. First, we construct different graphs, including geograph, transfer graph, and similarity graph, then input into a softmax network to generate the fusion graph.

After we obtain the fusion graph, we treat this mixture of different graphs as the input of the following two modules: (1) the GCN module: this module input the fusion graph into a GCN layer, then into an RNN framework (we use GRU as the node type. Note that LSTM works as well). Finally, we can achieve the node embedding with an FC layer, which includes the high-level relations (multihop neighbors in the graph) in the fusion graph e^h. (2) The node2vec module is a typical node2vec module, which aims to achieve the low level (1-hop neighbor in the graph) in the fusion graph e^l.

In addition, we extract features from the POI data and the historical human mobility data. The features include the number of different types of POIs within 300 square meters of origin and destination; OD flow in the previous K time slices; the OD flow at the same time slice in the previous day; and the average, maximum, minimum, and variance of OD flow in previous K time slices. We combine these features as e^f.

To predict the OD flow between different OD pairs, we input e^h, e^l, and e^f into the prediction model. We use different SOTA algorithms (such as the LASSO model, Random Forest model, and LightGBM model) as the prediction model to verify the effectiveness of each graph structure.

In the following section, we will give the details about our proposed model.

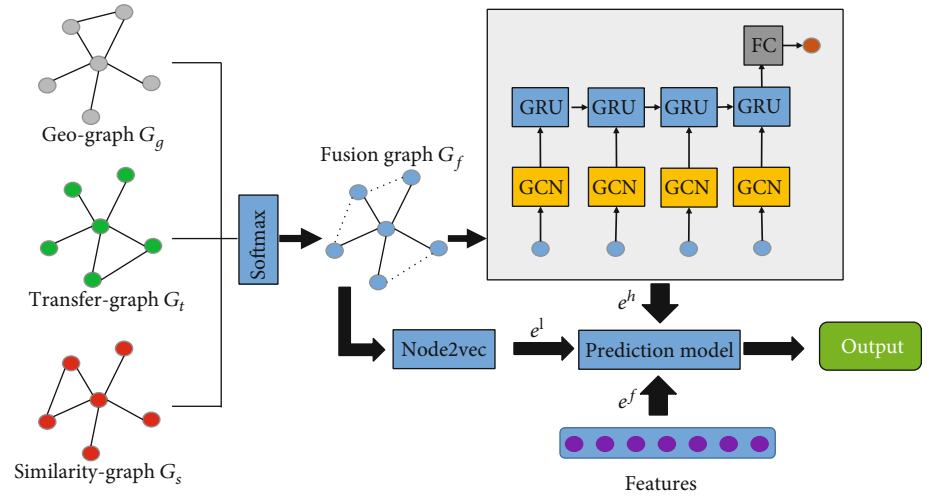

Figure 2: The framework of the proposed model.

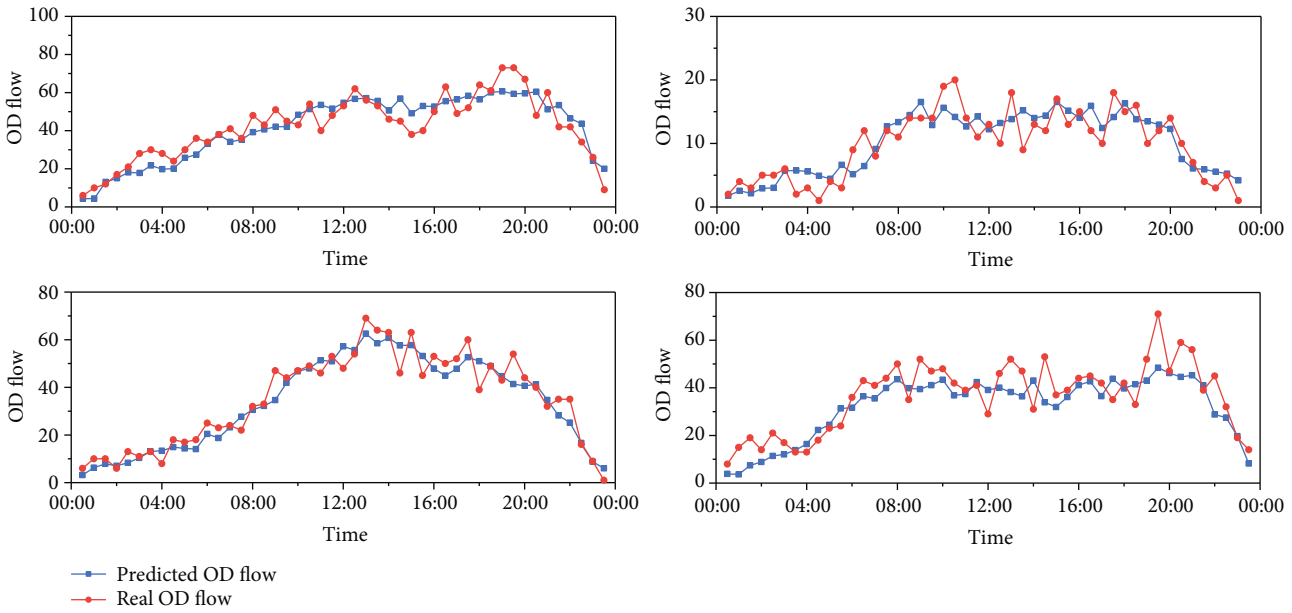

Figure 3: The OD flow time serial between different origin-destination pairs. From left-up to the right-bottom, the time interval is increasing.

4. Methodology

4.1. Definitions. This section gives detailed definitions of traffic flow, OD pairs, and several graph structures (geo-graph, transfer graph, and similarity graph). And in the last of this section, we give the problem definition.

Traffic flow: traffic flow (TF) is defined to measure the congestion level for a section in the city. Specifically, given a time point t, a section's s traffic flow can be defined as the check-in amount N_c during a period t^0. Note that the check-in amount is calculated by the signal data collected by base stations in this paper. Moreover, section s represents the circular coverage of the corresponding base station, which the center point is the longitude and latitude coordinates of the corresponding base station.

OD pairs: OD pairs are proposed to describe the users' trajectory in a fine-grained level. A user's mobile trajectory consists of different check-ins (as shown in Figure 3). An OD pair is a triplet, which is $< O, D,$ and $T >$. O stands for the origin check-in location (x_O, y_O), and D stands for the destination check-in location (x_D, y_D). T stands for the origin, destination check-in time (t_O, t_D).

Basic graph structure: we introduce three basic graph structures: (1) geograph: geograph is denoted as $G_g = \langle V_g, E_g \rangle$, where $v \in V_g$ is a section, and $s, e \in E_g$ is the geodistance between sections. (2) transfer graph: transfer graph is denoted as $G_t = \langle V_t, E_t \rangle$, where $v \in V_t$ is a section s, and $e \in E_t$ is the transfer frequency between sections, which is calculated by the historical check-in data. (3) similarity graph: similarity graph is denoted as $G_s = \langle V_s, E_s \rangle$, where $v \in V_s$ is a section s, and $e \in E_s$ is the Jaccard similarity of POI distribution between sections. With three graphs (geograph G_g, transfer graph G_t, and similarity-= graph G_s) as inputs, we employ a softmax module to build the fusion graph G_f.

Problem definition: given historical OD pairs ODs, city grid sections S, basic graph structure geograph G_g, transfer graph G_t, and section graph G_s, this model is proposed to capture all the features hidden in these data and predict the OD pairs and traffic flows after a short time period $\widehat{t}$: $\mathrm{OD}^{\widehat{t}}, \mathrm{TF}^{\widehat{t}}$.

4.2. Model Design

4.2.1. Graph Building Procedure. This section gives the details of this model, including some motivations, equations, and definitions. We calculate the distance between different sections i and sections j, and the calculation formula is as follows:

$$\mathrm{distance}_{i,j} = 2R \cdot \arcsin\left(\sqrt{\sin^2\left(\frac{\mathrm{lat}_i - \mathrm{lat}_j}{2}\right) + \cos\left(\mathrm{lat}_i\right) \cdot \cos\left(\mathrm{lat}_j\right) \cdot \sin^2\left(\frac{\mathrm{lon}_i - \mathrm{lon}_j}{2}\right)}\right). \tag{1}$$

Then, we calculate POI distributions as the initial embedding of base stations, as follows:

$$P_{si} = \left\{ p_j | \mathrm{distance}\left(s_i, p_j\right) < 300m, p_j \in P_{\mathrm{cc}} \right\}, \tag{2}$$

where P_{cc} is the POI set, and p_j is the specific POI.

If the POI distribution around two base stations is similar, it means that the coverage area of these two base stations has similar urban function expression and traffic mode. In terms of similarity of the urban function expression, this paper regards the Jaccard similarity of POI between base stations as the similarity of the urban function expression between base stations. The following formula calculates the Jaccard similarity. Jaccard similarity is used instead of cosine similarity, Pearson correlation coefficient, and other similarities because the latter two can only one-sided reflect the size of the included angle or the linear correlation between two vectors.

$$J\left(P_{s_i}, P_{s_j}\right) = \frac{\left|P_{s_i} \cap P_{s_j}\right|}{\left|P_{s_i} \cup P_{s_j}\right|} = \frac{\left|P_{s_i} \cap P_{s_j}\right|}{\left|P_{s_i}\right| + \left|P_{s_j}\right| - \left|P_{s_i} \cap P_{s_j}\right|}. \tag{3}$$

We should define the network structure between different base stations. Specifically, we treat each base station as the nodes in the graph, and the relations (recorded as adjacency matrices) are differently defined as the following, which build the several graphs:

(1) Distance adjacency matrix: we define the distance adjacency matrix as follows:

$$G_g = \langle V_g, E_g \rangle, \tag{4}$$

$$\mathrm{sign}\ (e) = \begin{cases} 1, e > 0, \\ 0, e = 0. \end{cases} \tag{5}$$

$$A_g = \begin{pmatrix} 1 & \frac{1}{dist_{0,1}} * \mathrm{sign}\ (d_{0,1}) & \frac{1}{dist_{0,2}} * \mathrm{sign}\ (d_{0,2}) & \cdots & \frac{1}{dist_{0,N-1}} * \mathrm{sign}\ (d_{0,N-1}) \\ \frac{1}{dist_{1,0}} * \mathrm{sign}\ (d_{1,0}) & 1 & \frac{1}{dist_{1,2}} * \mathrm{sign}\ (d_{1,2}) & \cdots & \frac{1}{dist_{1,N-1}} * \mathrm{sign}\ (d_{1,N-1}) \\ \frac{1}{dist_{2,0}} * \mathrm{sign}\ (d_{2,0}) & \frac{1}{dist_{2,1}} * \mathrm{sign}\ (d_{2,1}) & 1 & \cdots & \frac{1}{dist_{2,N-1}} * \mathrm{sign}\ (d_{2,N-1}) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{1}{dist_{N-1,0}} * \mathrm{sign}\ (d_{N-1,0}) & \cdots & \cdots & \cdots & 1 \end{pmatrix}. \tag{6}$$

This formulation defines that the closer base station pairs should achieve a close relationship.

(2) OD flow adjacency matrix: we define the OD flow adjacency matrix as follows:

$$G_t = \langle V_t, E_t \rangle, \tag{7}$$

$$A_t = \begin{pmatrix} 1 & \frac{d_{0,1}}{d_{\max}} & \frac{d_{0,2}}{d_{\max}} & \cdots & \frac{d_{0,N-1}}{d_{\max}} \\ \frac{d_{1,0}}{d_{\max}} & 1 & \frac{d_{1,2}}{d_{\max}} & \cdots & \frac{d_{1,N-1}}{d_{\max}} \\ \frac{d_{2,0}}{d_{\max}} & \frac{d_{2,1}}{d_{\max}} & 1 & \cdots & \frac{d_{2,N-1}}{d_{\max}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{d_{N-1,0}}{d_{\max}} & \cdots & \cdots & \cdots & 1 \end{pmatrix}, \tag{8}$$

where we utilize the maximum $d_{\max}$ as a regularization, and this graph considers the interactions between different base stations. Then, more OD pairs occur between them, the closer relationship they achieve in the graph.

(3) Jaccard adjacency matrix: we use Jaccard similarity to build the initial POI based base station embedding graph as follows:

$$G_s = \langle V_s, E_s \rangle, \tag{9}$$

$$A_s = \begin{pmatrix} 1 & j_{0,1} & j_{0,2} & \cdots & j_{0,N-1} \\ j_{1,0} & 1 & j_{1,2} & \cdots & j_{1,N-1} \\ j_{2,0} & j_{2,1} & 1 & \cdots & j_{2,N-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ j_{N-1,0} & \cdots & \cdots & \cdots & 1 \end{pmatrix}, \tag{10}$$

where this graph denotes the similarity of different sections.

4.2.2. Model Training Procedure. The three base station graph structures are taken as input, and the adjacency matrix used for graph convolution is obtained by weighted fusion of the Softmax function. Then, the historical resident data of several time pieces are used as input to extract the spatial structure features between base stations through the graph convolution layer, and then the time correlation of data is learned through GRU. Finally, the future T-set time slice population's population resident capacity is predicted through the full connection layer. After the population resides, training task will be the figure of embedded features plus history transfer flow between base stations, base station location characteristics, base station network structure based on graph theory to extract features of standard input to the machine learning model, and forecast the OD flow between base stations.

In order to utilize different information hidden in the different graphs, we utilize the weighted fusion method to learn the final, fusion graph, which is formulated as follows:

$$\widehat{\theta}_1, \widehat{\theta}_2, \cdots, \widehat{\theta}_K = soft\max(\theta_1, \theta_2, \cdots, \theta_K), \tag{11}$$

$$\widehat{A}_i = \tilde{D}^{-\frac{1}{2}} \widehat{A}_i \tilde{D}^{-\frac{1}{2}} s.t. (1 \le i \le K), \tag{12}$$

$$\tilde{A}_f = \sum_{i=1}^{K} \widehat{\theta}_i \circ \widehat{A}_i, \tag{13}$$

where θ is the weight, and A_i denotes the ith adjacency matrix. After we achieve the fusion adjacency matrix, we input all the graphs into GCN module and do the graph convolutional action twice:

$$H^{(0)} = \mathrm{RELU}\left(\widehat{A}_f X W^{(0)}\right), \tag{14}$$

$$H^{(1)} = \sigma\left(\tilde{A}_f H^{(0)} W^{(1)}\right) = \sigma\left(\widehat{A}_f \mathrm{RELU}\left(\widehat{A}_f X W^{(0)}\right) W^{(1)}\right), \tag{15}$$

where X is the characteristic matrix of the original input 12 time slice flows, and they are calculated from the parameter matrix to achieve results from graph convolution operation. Then, the result of graph convolution is input into the GRU module to capture the temporal correlation of traffic:

$$R_t = \sigma(H_1 W_{xr} + S_{t-1} W_{sr} + b_r), \tag{16}$$

$$Z_t = \sigma(H_1 W_{xz} + S_{t-1} W_{sz} + b_z), \tag{17}$$

$$\tilde{S}_t = \tanh\left(H_1 W_{xh} + (R_t \otimes S_{t-1}) W_{ss} + b_s\right), \tag{18}$$

$$S_t = Z_t \otimes S_{t-1} + (1 - Z_t) \otimes \tilde{S}_t. \tag{19}$$

Finally, the output of the FC layer is the prediction results. This paper selects the mean square error loss function as the loss function in the pretraining model and adds regularization terms to prevent overfitting. The loss function is shown as follows, where y represents the actual value of population resident, $\hat{y}$ represents the predicted value of population resident, L_{reg} represents the regular term, and λ represents the weight parameter.

$$\mathrm{loss} = ||y - \hat{y}|| + \lambda L_{\mathrm{reg}}. \tag{20}$$

The resulting base station map structure is used for the Node2Vec algorithm to generate the embedding vector of each base station, which is input into the LightGBM model and other features constructed by feature engineering to predict OD traffic between base stations.

In this paper, feature engineering is constructed from three perspectives to make the model learn the flow transfer rules between base stations. The features are as follows: (1) historical transfer flow characteristics between base stations, (2) geographical location characteristics of base stations, and

(3) base station network structure features extracted based on graph theory.

5. Experiments

5.1. Datasets. In this paper, a large-scale signal data from July 3, 2017 to July 7, 2017 in Changchun is obtained to train and verify the performance of the proposed model. The dataset contains 200 million cell phone users and a total of 49,716,815 signal records. The first 4 days is used for training, and the last is used for testing.

5.2. Prediction Model Selection. We select the following three models as the prediction model, respectively, to compare the effects of different graph structures on the performance of the proposed model:

Lasso [25]: a compression estimation method is based on the idea of reducing the variable set (order reduction). By constructing a penalty function, it can compress the coefficients of variables and make some regression coefficients that become 0, so as to achieve the purpose of variable selection.

Random Forest [26]: random forests have been one of the successful ensemble algorithms in machine learning. The basic idea is to construct many random trees individually and make predictions based on an average of their predictions. The great successes have attracted much attention on the consistency of random forests, mostly focusing on regression.

LightGBM [27]: LightGBM excludes a significant proportion of data instances with small gradients and only use the rest to estimate the information gain. It can obtain quite an accurate estimation of the information gain with a much smaller data size and bundle mutually exclusive features (i.e., they rarely take nonzero values simultaneously), to reduce the number of features.

To verify the effectiveness of each graph structure, we test the accuracy of the prediction results by using control experiments, that is using different graph combinations (without graph embedding, with G_g, with G_t, with G_s, with fusion of all graphs) as the input of the prediction model. The results are shown in Table 1.

5.3. Experimental Results

5.3.1. Prediction Accuracy Evaluation. The prediction results are shown in Table 1. To avoid the difference of the prediction results that is too tiny, we filter out the OD flow whose predicted value and the ground truth are both 0, then calculate the loss and RMSE as shown in Table 1. We can see that LightGBM achieves better performance in both of these three selected prediction models, which can better fit the nonlinear relationship in the data. The lasso regression model has poor performance due to its simplicity. By using LightGBM as the prediction model, with fusion of all the graph structures, the performances of loss and RMSE gain 67.72% and 64.03%, respectively, than without embedding. With only one graph structure as embedding feature, transfer graph G_t makes more significant performance improvements than geograph G_g and similarity graph G_s. Without embedding as input achieves the worst performance in both of the prediction models.

In summary, the graph fusion embedding feature can significantly improve the performance of OD flow prediction, and transfer graph G_t plays a more important role than geograph G_g and similarity graph G_s.

Table 1: Experimental result comparison.

	Lasso		Random Forest		LightGBM	
	Loss	RMSE	Loss	RMSE	Loss	RMSE
No embedding	19.74	25.12	18.82	23.45	18.74	23.49
With G_g	14.96	19.98	14.70	18.66	14.01	18.17
With G_t	9.72	12.27	7.80	10.38	7.72	10.06
With G_s	12.77	15.83	12.73	15.75	12.69	15.43
With fusion	7.45	9.54	6.51	9.37	6.05	8.45

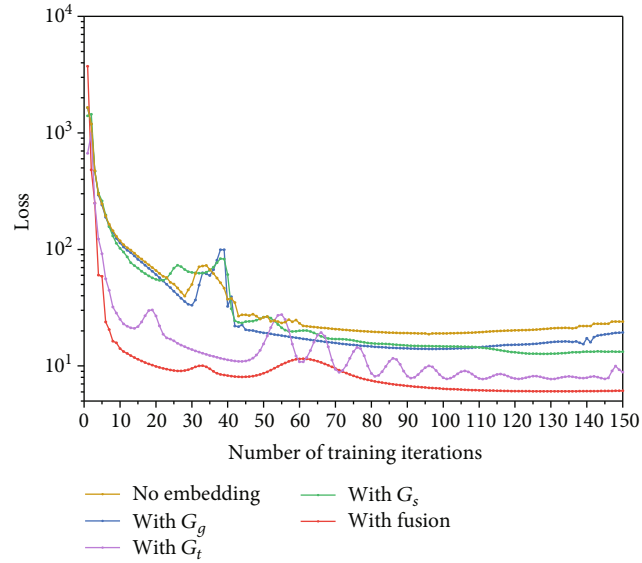

Figure 4: The convergence of the proposed model, with LightGBM as the prediction model, under different graph structure combinations.

Table 2: The training time comparison.

	Using LightGBM as the prediction model	
	Number of iterations with minimum loss	Training time (hours)
No embedding	95	1.97
With G_g	96	2.28
With G_t	111	2.19
With G_s	128	2.37
With fusion	129	3.15

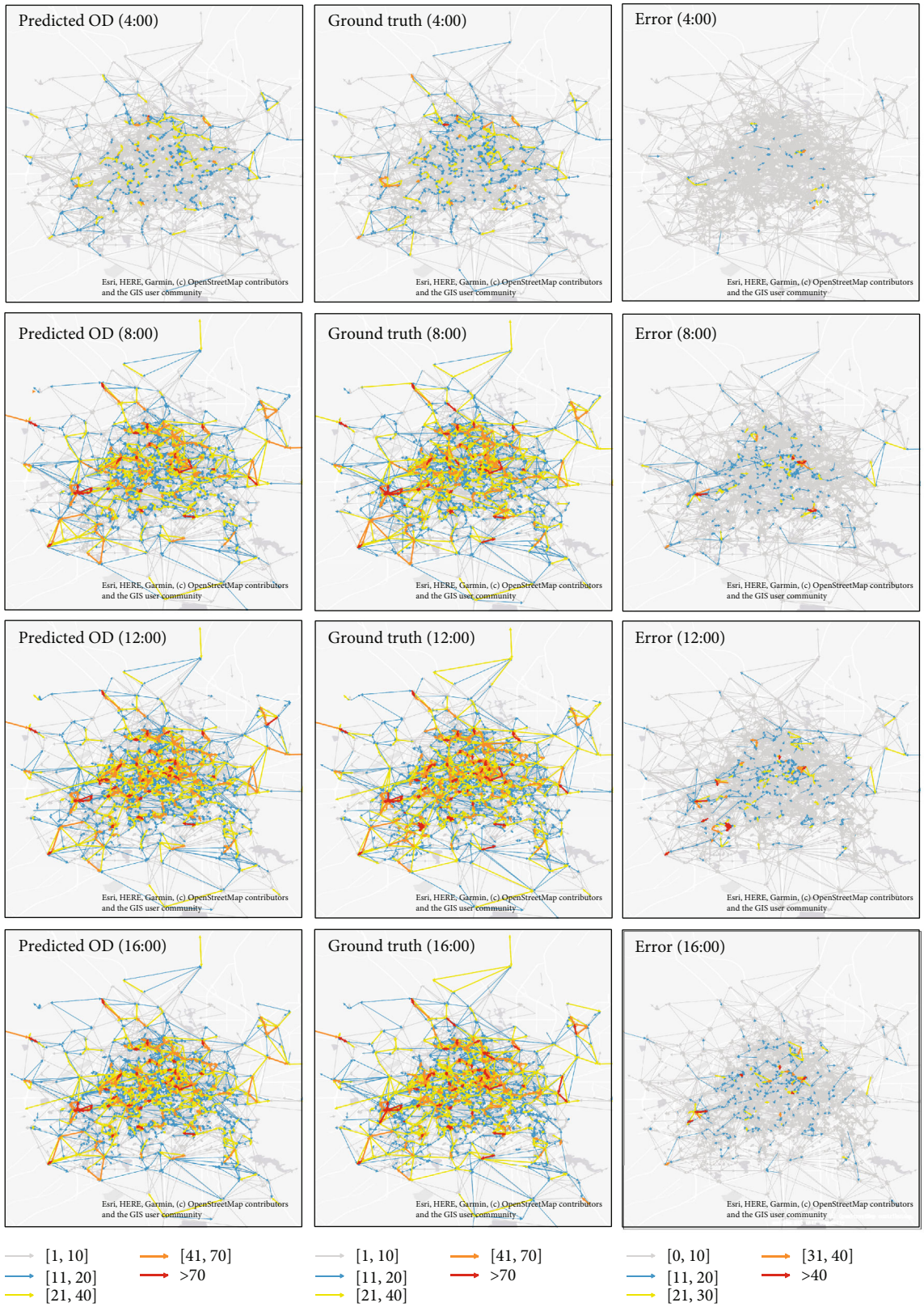

Figure 5: OD flow prediction results in different time periods. Left column predicted OD flow. Middle column is the ground truth. Right column is the absolute error between the prediction results and the ground truth.

5.3.2. Computation Time and Convergence. In this paper, we implement the proposed framework by using python 3.6, tensorflow-1.12.0. The hardware environment is with 16 GB RAM, 3.20 GHz Intel(R) Core (TM) i7-8700 CPU). Under above experimental environment, we select LightGBM as prediction model, to discuss the computation time and convergence under different graph structure combinations, and similar conclusions can be easily extended to the other two prediction models. As shown in Figure 4, the loss of each situation decreased significantly in the first 30 training iterations. Among them, the loss of fusing all graph structures decreases the fastest, followed by using transfer graph G_t, and the loss of without graph embedding decreases the slowest and achieves the worst performance.

We set maximum training iterations as 150, and the number of iterations with minimum loss and the training time (hours) is presented in Table 2. We can find that fusing all graph structures costs more training time (3.15 hours).

5.3.3. Ablation Evaluation. In this section, we separate the time periods into different time pieces. So, we could explore the effect of different OD pairs on our proposed model. Note that the longer OD pair time could lead to a fierce shift in the performance, as shown in Figure 3:

From the results, we can see that with the increasing of the time interval between OD pairs, the performance of our proposed model is decreasing. However, our proposed model still captures OD pairs' tendency, which proves the robustness of our proposed model.

5.3.4. City Scale OD Pair Prediction. In this section, we want to explore the city scale OD pair prediction performance. Specifically, we give the comparison between predict results and ground truth. The results are shown in Figure 5.

The left column is the prediction, the middle column is the ground truth, and the right column is the absolute errors. Each row is a time period (0 : 00-6 : 00, 6 : 00-12 : 00, 12 : 00-18 : 00, 18 : 00-24 : 00). From the result, we can see that our proposed model could capture the city scale OD pair characters and give a proper evaluation of different sections. Specifically, we can see from the right column that our proposed model achieves the best performance in the first time period (0 : 00-6 : 00). The reason is that at this time, the users' action pattern is simple but predictable. Usually, they are works, doctors, and city cleaners. So, the prediction accuracy is quite high. Note that there are several sections where the prediction performance is not satisfying in the following three time periods. Specifically, take deep into this scenario, we can see that the low prediction accuracy sections have some similar characters: first, there sections' traffic flow are always huge and unstable. Some sections are CBDs where locates at the city's center. So, the flow prediction is difficult for our proposed model and all the prediction models because there are too many reasons that could affect these sections' traffic flow.

6. Conclusion

This paper uses large-scale mobile phone signal data to achieve human OD flow prediction between the coverage of varying signal cell towers. Many signal cell towers are distributed in urban geographical space, collecting all the cell phone user's location information to obtain large-scale fine-grained unbiased human OD flow data. Extensive evaluation proves the proposed model's superior performance over some SOTA methods.

In the future, we plan to combine this model with other aspects to build a smart city, such as functional zone division, road traffic congestion monitoring, and other applications. We believe this work could be basic work for other high-level algorithms.

Data Availability

The data used to support the findings of this study have not been made available because the data also forms part of an ongoing study.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work is supported by the National Natural Science Foundations of China under Grant No. 61772230, No. 61976102, No. U19A2065, and No. 61972450, Natural Science Foundation of China for Young Scholars No. 61702215 and No. 62002132, China Postdoctoral Science Foundation No. 2020M681040, Changchun Science and Technology Development Project No. 18DY005, National Defense Science and Technology Key Laboratory Fund Project No. 61421010418, Science Foundation of Jilin Province No. 20190201022JC, and China National Postdoctoral Program for Innovative Talents No. BX20180140.

References

[1] Y. Yang, Y. Xu, J. Han, E. Wang, W. Chen, and L. Yue, "Efficient traffic congestion estimation using multiple spatio-temporal properties," *Neurocomputing*, vol. 267, pp. 344–353, 2017.

[2] H. Hu, Z. Jiang, Y. Zhao, Y. Zhang, H. Wang, and W. Wang, "Network representation learning-enhanced multisource information fusion model for POI recommendation in smart city," *IEEE Internet of Things Journal*, vol. 8, no. 12, pp. 9539–9548, 2021.

[3] Q. Huang, Y. Yang, Y. Xu, F. Yang, Z. Yuan, and Y. Sun, "City-wide road-network traffic monitoring using large-scale mobile signaling data," *Neurocomputing*, vol. 444, pp. 136–146, 2021.

[4] X. Xu, Q. Huang, X. Yin, M. Abbasi, M. R. Khosravi, and L. Qi, "Intelligent offloading for collaborative smart city services in edge computing," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 7919–7927, 2020.

[5] J. Yang, B. Guo, Z. Wang, and Y. Ma, "Hierarchical prediction based on Network-Representation-Learning-Enhanced clustering for bike-sharing system in smart city," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6416–6424, 2021.

[6] Y. Xu, Y. Yang, E. Wang et al., "Neural serendipity recommendation," *ACM Transactions on Knowledge Discovery from Data*, vol. 14, no. 4, pp. 1–25, 2020.

[7] Y. Xu, Y. Yang, J. Han, E. Wang, J. Ming, and H. Xiong, "Slanderous user detection with modified recurrent neural networks in recommender system," *Information Sciences*, vol. 505, pp. 265–281, 2019.

[8] K. Song, J. Min, G. Lee, S. C. Shin, and Y. S. Kim, "An improvement of plagiarized area detection system using jaccard correlation coefficient distance algorithm," *Computer Science and Information Technology*, vol. 3, no. 3, pp. 76–80, 2015.

[9] Y. Gong, Z. Li, J. Zhang, W. Liu, and Y. Zheng, "Online Spatio-Temporal Crowd Flow Distribution Prediction for Complex Metro System," *IEEE Transactions on Knowledge and Data Engineering*, p. 1, 2020.

[10] H. Yang, X. Li, W. Qiang, Y. Zhao, W. Zhang, and C. Tang, "A network traffic forecasting method based on SA optimized ARIMA-BP neural network," *Computer Networks*, vol. 193, article 108102, 2021.

[11] S. Kim, V. M. Deshpande, and R. Bhattacharya, "Robust kalman filtering with probabilistic uncertainty in system parameters," *IEEE Control Systems Letters*, vol. 5, no. 1, pp. 295–300, 2021.

[12] Y. Xu, E. Wang, Y. Yang, and Y. Chang, "A unified collaborative representation learning for neural-network based recommendersystems," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2021.

[13] Y. Xu, Y. Yang, E. Wang, F. Zhuang, and H. Xiong, "Detect professional malicious user with metric learning in recommender systems," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2020.

[14] L. P. Zapata, M. Flores, V. M. Larios-Rosillo, R. Maciel, and E. A. Antunez, "Estimation of people flow in´ public transportation network through the origin-destination problem for the south-eastern corridor of quito city in the smart cities context," in *2019 IEEE International Smart Cities Conference, ISC2 2019*, pp. 181–186, Casablanca, Morocco, 2019.

[15] Z. Duan, K. Zhang, Z. Chen et al., "Prediction of city-scale dynamic taxi origin-destination flows using a hybrid deep neural network combined with travel time," *IEEE Access*, vol. 7, pp. 127816–127832, 2019.

[16] S. Meng, S. Sun, and B. Yang, "Traffic flow prediction with conv-sae," in *Proceedings of the 2020 the 7th International Conference on Automation and Logistics (ICAL)*, pp. 90–93, Beijing, China, 2020.

[17] F. Scarselli, M. Gori, Ah Chung Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2009.

[18] J. Zhou, G. Cui, S. Hu et al., "Graph neural networks: a review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020.

[19] Z. Zhu, R. Li, M. Shan et al., "TDP: personalized taxi demand prediction based on heterogeneous graph embedding," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019*, pp. 1177–1180, Paris, France, 2019.

[20] L. Grad-Gyenge, A. Kiss, and P. Filzmoser, "Graph embedding based recommendation techniques on the knowledge graph," in *Adjunct Publication of the 25th Conference on User Modeling, Adaptation and Personalization, UMAP 2017*, pp. 354–359, Bratislava, Slovakia, 2017.

[21] X. Wang, X. He, M. Wang, F. Feng, and T. Chua, "Neural graph collaborative filtering," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019*, pp. 165–174, Paris, France, 2019.

[22] S. Guo, Y. Lin, S. Li, Z. Chen, and H. Wan, "Deep spatial-temporal 3D convolutional neural networks for traffic data forecasting," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3913–3926, 2019.

[23] L. Zhao, Y. Song, C. Zhang et al., "T-GCN: a temporal graph convolutional network for traffic prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 9, pp. 3848–3858, 2020.

[24] J. Johnson, A. Gupta, and L. Fei-Fei, "Image generation from scene graphs," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018*, pp. 1219–1228, Salt Lake City, UT, USA, 2018.

[25] C. Han, J. Shimamura, T. Takahashi et al., "Real-time detection of malware activities by analyzing darknet traffic using graphical lasso," in *18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering, TrustCom/BigDataSE 2019*, pp. 144–151, Rotorua, New Zealand, 2019.

[26] W. Gao and Z. Zhou, "Towards convergence rate analysis of random forests for classification," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, pp. 9300–9311, Vancouver, Canada, 2020.

[27] G. Ke, Q. Meng, T. Finley et al., "Lightgbm: a highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*, pp. 3146–3154, Long Beach, CA, USA, 2017.