Article

Predicting Station-Level Short-Term Passenger Flow in a Citywide Metro Network Using Spatiotemporal Graph Convolutional Neural Networks

Yong Han [1,2], Shukang Wang [1,2,*], Yibin Ren [1,2], Cheng Wang [1], Peng Gao [3] and Ge Chen [1,2]

[1] College of Information Science and Engineering, Ocean University of China, No. 238, Songling Road, Qingdao 266100, China; yonghan@ouc.edu.cn (Y.H.); renyibin@stu.ouc.edu.cn (Y.R.); wc8387@stu.ouc.edu.cn (C.W.); gechen@ouc.edu.cn (G.C.)

[2] Pilot National Laboratory for Marine Science and Technology (Qingdao), No. 1, Wenhai Road, Qingdao 266237, China

[3] Qingdao Transportation Public Service Center, No. 163, Shenzhen Road, Qingdao 266061, China; gaopeng@qd.shandong.cn

* Correspondence: wangshukang@stu.ouc.edu.cn; Tel.: +86-0532-6678-6365

check for updates

Abstract: Predicting the passenger flow of metro networks is of great importance for traffic management and public safety. However, such predictions are very challenging, as passenger flow is affected by complex spatial dependencies (nearby and distant) and temporal dependencies (recent and periodic). In this paper, we propose a novel deep-learning-based approach, named STGCNNmetro (spatiotemporal graph convolutional neural networks for metro), to collectively predict two types of passenger flow volumes—inflow and outflow—in each metro station of a city. Specifically, instead of representing metro stations by grids and employing conventional convolutional neural networks (CNNs) to capture spatiotemporal dependencies, STGCNNmetro transforms the city metro network to a graph and makes predictions using graph convolutional neural networks (GCNNs). First, we apply stereogram graph convolution operations to seamlessly capture the irregular spatiotemporal dependencies along the metro network. Second, a deep structure composed of GCNNs is constructed to capture the distant spatiotemporal dependencies at the citywide level. Finally, we integrate three temporal patterns (recent, daily, and weekly) and fuse the spatiotemporal dependencies captured from these patterns to form the final prediction values. The STGCNNmetro model is an end-to-end framework which can accept raw passenger flow-volume data, automatically capture the effective features of the citywide metro network, and output predictions. We test this model by predicting the short-term passenger flow volume in the citywide metro network of Shanghai, China. Experiments show that the STGCNNmetro model outperforms seven well-known baseline models (LSVR, PCA-kNN, NMF-kNN, Bayesian, MLR, M-CNN, and LSTM). We additionally explore the sensitivity of the model to its parameters and discuss the distribution of prediction errors.

Keywords: citywide metro network; graph convolution neural network; spatiotemporal flow-volume prediction; deep learning

1. Introduction

The prediction of short-term passenger flow volume is a vital component of metro systems. In particular, the accurate prediction of the short-term passenger flow volume in a citywide metro network can help urban managers to fine-tune travel behaviors, reduce passenger congestion, and enhance the service quality of metro systems [1]. Thus, constructing an effective model to predict the passenger flow volume in a citywide metro network is essential.

　　　During recent decades, many passenger-flow prediction models have been proposed based on statistical and machine learning (ML) algorithms, such as support vector machine (SVM) [2–4], Bayesian regression [5], principal component analysis (PCA) [6,7], non-negative matrix factorization (NMF) [8], and artificial neural networks (ANNs) [1,9–11]. However, these conventional methods cannot process raw sample data and require a manual feature engineering procedure. In the ML domain, a feature is an individual measurable property of the phenomenon being observed [12]. In a typical ML model, feature engineering transforms raw data into suitable internal features so that the learning subsystem can detect patterns in the input data. Generally, feature engineering is complex as it requires careful engineering and considerable domain expertise. As the key to achieving a high-accuracy prediction system is capturing features as accurately as possible [13], feature engineering is crucial for ML methods. However, for the situation of a citywide metro network, it is difficult to achieve high-quality feature engineering manually. The most important feature for the prediction of the citywide spatiotemporal passenger flow volume is the spatiotemporal dependencies [14,15]. In a metro network, the spatial dependencies refer to the interactions between the passenger inflow volume and outflow volume in near and distant stations. Specifically, the passenger inflow volume in one station directly affects the outflow volume in another (near or distant) station. Meanwhile, the temporal dependencies refer to the impact of historical observational time-series. For example, the passenger flow volume at 7:00 a.m. may have a strong correlation with that at 6:00 a.m. Additionally, it may also be very similar to the flow volume at 7:00 a.m. on the previous day or that at 7:00 a.m. on the same day of the previous week, since human activities have daily and weekly periodicities. Some metropolises have hundreds of metro stations (e.g., Beijing, China, has 391; Shanghai, China, has 289; and Tokyo, Japan, has 290). The spatial and temporal dependencies of these citywide metro systems are complex and highly integrated. It is difficult to characterize the spatiotemporal dependencies of a metro network by manual feature engineering. Therefore, conventional ML methods are not suitable for the prediction of passenger flow volume in a citywide metro network.

　　　Recently, deep learning (DL) has solved the problem of manually extracting feature engineering [16]. Unlike traditional ML algorithms, DL models can accept input data in raw format and automatically discover the required features level-by-level, a technique that is called "end-to-end" learning. DL enormously simplifies the feature engineering procedure, and also improves feature quality [13]. The performance of DL methods has been confirmed in many domains, such as image recognition, video processing, and natural language processing [13,17–21]. Long short-term memory (LSTM) is a special kind of deep recurrent neural network (RNN) which is suitable for modeling dynamic temporal dependency occurring in a time-series [22,23]. Hence, several LSTM-based models have been proposed to predict the passenger flow volumes in isolated stations or traffic lines, which obtained better accuracy than traditional prediction methods [24–26]. However, LSTM is not good at capturing spatial dependency at the citywide level [27]. Convolutional neural networks (CNNs) have been applied for the prediction of citywide crowd flow [28]. CNNs are specifically designed for data domains with regular grids, such as images. They can straightforwardly identify spatial dependencies among grids with various localized filters or kernels, and these shift-invariant kernels are learned automatically from data. To apply a CNN to the prediction of large-scale spatiotemporal crowd flow volume, Zhang et al. [28] divided a city extent into grids with a predefined size and calculated the flow volume for each grid. Several historical spatiotemporal flow-volume grids were then fed into a deep CNN model, and the spatiotemporal dependencies among grids were learned through stacked localized kernels. However, it should be noted that this approach is not applicable for the prediction of station-level flow volume, since if the grid size is set too large, multiple stations will be covered by the same grid and fail to satisfy the required granularity. Conversely, if the grid size is as small as one station, the resulting huge image matrix with redundant zero elements will strongly increase the computational burden. Ma et al. [29] proposed another method to transform road-network-based data into a two-dimensional image, with the horizontal axis representing time tags and the vertical axis representing road segments. This image was then input into a CNN framework to model the spatiotemporal dependency using

a convolution operator. Although the image of Ma et al. represented each road segment separately, maintaining the spatial precision of each road, the image transformation procedure did not consider the topological dependencies of the road network which is the same for the metro network. Importantly, as citywide metro networks are irregular network structures, the regular-grid-based convolution operation could not accurately capture the spatiotemporal dependencies along the metro network.

For data in irregular or non-Euclidean domains, such as user data in social networks and genetic data in biological regulatory networks, graphs have been applied as main structures to encode heterogeneous pairwise dependencies and complex geometric structures in data [30]. In order to conduct classification and prediction tasks using these graph datasets, a novel DL model, named graph CNN (GCNN), was recently proposed [30,31]. A citywide metro network can be represented as a graph with metro stations as vertices and metro lines as edges. Each station vertex has a feature vector consisting of historical values of passenger flow, and an adjacency matrix can be defined to encode the pairwise dependencies between stations. Therefore, instead of representing metro stations by grids and capturing features using a CNN, a metro network can be characterized as a general graph, and a GCNN can be employed to effectively capture the irregular spatiotemporal dependencies at the metro network level, rather than the grid level.

This paper introduces GCNNs to predict the station-level short-term passenger flow volume in a citywide metro network, proposing a novel DL method named STGCNNmetro (spatiotemporal graph convolutional neural networks for metro). First, we apply stereogram graph convolution kernels to historical metro passenger flow-volume time-series to seamlessly capture the irregular spatiotemporal dependencies along the metro network. Second, a deep structure composed of GCNNs is constructed to capture the distant spatiotemporal dependencies at the citywide level. Finally, to consider the influences of multiple temporal patterns, we fuse the spatiotemporal dependencies captured from recent, daily, and weekly patterns to produce the final prediction values. The model is trained by back-propagation. Compared with existing models, the main contribution of this paper is that it proposes a novel "end-to-end" DL-based model which is able to automatically capture the irregular spatiotemporal dependencies of a metro network from raw passenger flow-volume data and achieve accurate predictions of passenger flow volume for the citywide metro system.

2. Methodology

2.1. Representing Time-Series of Metro Network Passenger Flow Volume by Graphs

As the prediction of the passenger flow volume of a citywide metro network is a typical spatiotemporal series prediction problem, we can predict the passenger flow volume at a subsequent time step given the previous M observations. In this work, we define a citywide metro network on a graph and focus on structured time-series of passenger flow volume. As shown in Figure 1, $V_t \in R^n$ is an observation vector of n metro stations at time step t, each element of which records the historical passenger flow volume for a single metro station. The observation V_t is not isolated in space but is rather linked by pairwise connection in the graph. Therefore, the passenger flow volume V_t of the citywide metro network can be regarded as a graph signal that is defined on an undirected graph G with weights w_{ij}. At time step the t, in the graph $G_t = (V_t, E, W)$, V_t is a finite set of vertices corresponding to the observations from n monitoring stations in a metro network; E is a set of edges, indicating the connectedness between stations; while $W \in R^{n \times n}$ denotes the weighted adjacency matrix of G_t.
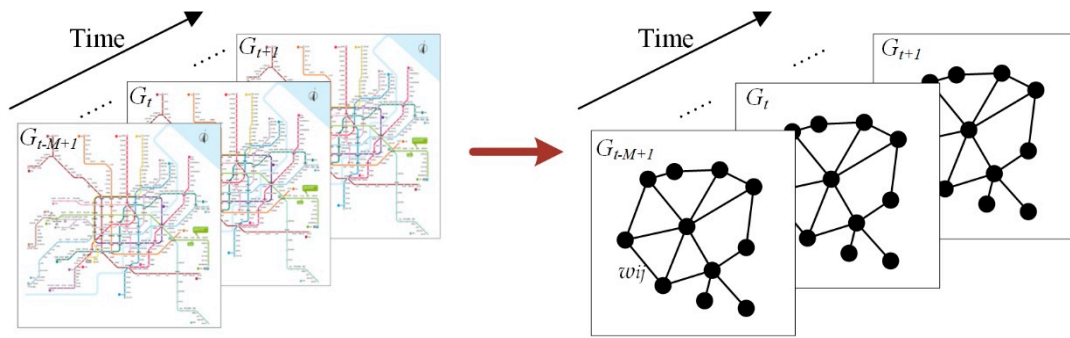
Figure 1. Graph-structured metro passenger flow-volume data. At the time step t, in the graph $G_t = (V_t, E, W)$, V_t is an observation vector of n metro stations at time step t, each element of which records the historical passenger flow volume for a single metro station; E is a set of edges, indicating the connectedness between stations; while $W \in R^{n \times n}$ denotes the weighted adjacency matrix of G_t.

2.2. Capturing Spatiotemporal Dependencies Using Graph Convolutional Neural Networks (GCNNs)

2.2.1. Graph Convolution Operation

A standard convolution for regular grids is clearly not applicable to general graphs. A basic approach currently used to explore how to generalize CNNs to structured data forms is to manipulate in the spectral domain with graph Fourier transforms [32]. It introduces the spectral framework to apply convolutions in spectral domains, which is often called the spectral graph convolution. Several follow-up studies have made the graph convolution more effective by reducing the computational complexity from $O(n^2)$ to linear cost [30,31]. In this study, we adopted the GCNN structure proposed by Defferrard et al. [30] to model passenger flow volume data from a citywide metro network. The notion of the graph convolution operator "$* g$", which is based on the concept of spectral graph convolutions, as the multiplication of a signal $x \in R^n$ with a kernel Θ, is given as follows:

$$y = \Theta * g\, x = \sigma\left(\sum_{j=0}^{K} \alpha_j L^j x\right), \tag{1}$$

where K is the kernel size of the graph convolution, which determines the maximum radius of the convolution from central nodes, α_j is the convolution kernel parameter, L is the normalized graph Laplacian, and σ is an activation function (i.e., rectified linear unit (ReLU)).

2.2.2. Using Stereogram Graph Convolution to Capture Irregular Spatiotemporal Dependencies

After constructing the graph-structured metro network data as described in the previous section, we extract the local irregular spatiotemporal dependencies of the metro network using the GCNN. Taking vertex i as an example, Figure 2 shows the conventional graph convolution with a kernel size of one ($K = 1$). Actually, this is the convolution calculation performed with the first-order adjacencies. Obviously, the spatial dependencies between vertex i and its five first-order adjacent vertices (marked as 1, 2, 3, 4, and 5 in Figure 2) are captured. The illustrated operation is executed for all vertices. For the metro network passenger flow volume time-series data of the previous M intervals, we construct a stereogram convolution kernel to seamlessly extract the spatiotemporal dependencies. The M historical flow-volume graphs are stacked in the temporal dimension. In the stereogram graph convolution, convolutions are seamlessly performed in both the spatial and temporal dimensions, as shown in Figure 3. We define the notion of the spatiotemporal graph convolution operator "$* g'$" as the multiplication of a signal $x' \in R^{M \times n}$ with a stereogram kernel Θ':

$$y' = \Theta' * g'x' = \sigma\left(\sum_{j=0}^{K} \sum_{t=1}^{M} \alpha_{jt} L^j x'_t\right) \tag{2}$$

where α_{jt} is the convolution kernel parameter at the time step t, and x_t' corresponds to the observations from n monitoring stations in a metro network at the time step t. For vertex i, after the stereogram graph convolution, the irregular spatiotemporal dependencies contained in its first-order spatial adjacent vertices and its M-order temporal adjacencies are captured.
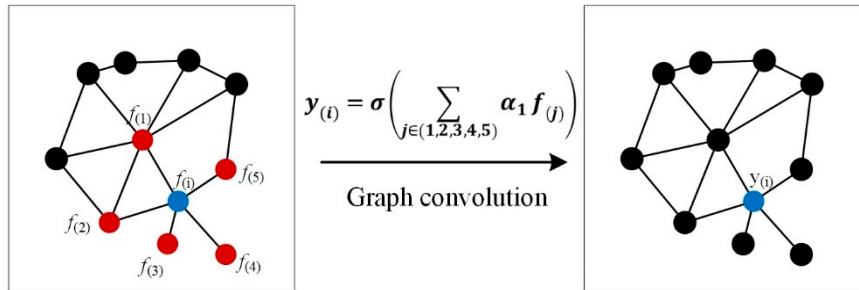

Figure 2. Procedure for capturing spatial dependency by graph convolution for $K = 1$ (using only one vertex as an example). K is the kernel size of the graph convolution, which determines the maximum radius of the convolution from central nodes, α_1 is the convolution kernel parameter, σ is an activation function (i.e., rectified linear unit (ReLU)), $y_{(i)}$ indicates the passenger flow volume of vertex i to be predicted, and each $f_{(j)}$ indicates the historical passenger flow volume of vertex j.
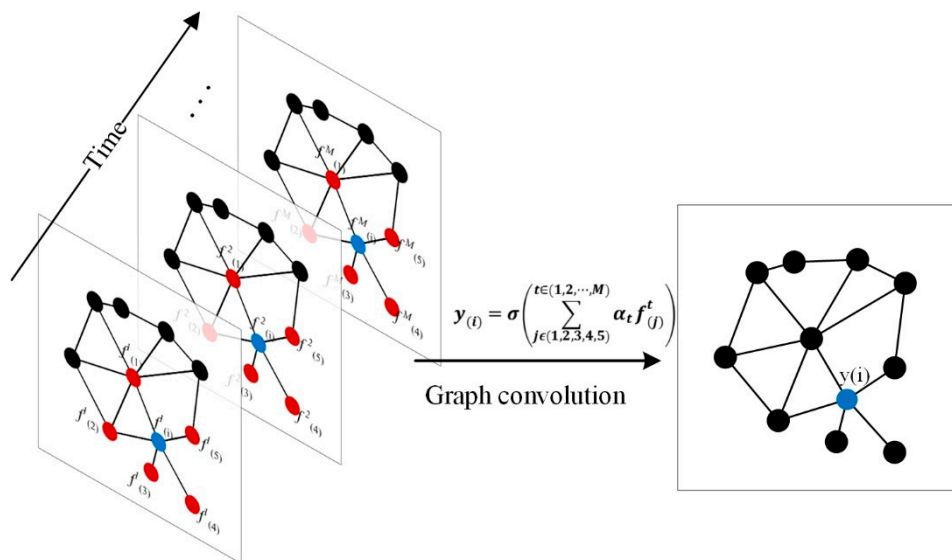

Figure 3. Procedure for capturing spatiotemporal dependencies by graph convolution for $K = 1$ (using only one vertex as an example). α_t is the convolution kernel parameter at the time step t, and $f_{(j)}^t$ indicates the historical passenger flow volume of vertex j at the time step t.

2.2.3. Using Deep GCNNs to Capture Distant Spatiotemporal Dependencies in a Citywide Metro Network

In a metro network structure, high-order adjacencies can be accumulated by low-order adjacencies. For example, second-order adjacencies can be obtained by accumulating two first-order adjacencies. Usually, the spatial scale of a citywide metro network is large and contains many stations from the city center to the suburbs. Intuitively, the passenger flow volumes between nearby stations may affect each other. This can be effectively handled by one GCNN layer which has shown a powerful ability to hierarchically capture spatial structural information [33,34]. Additionally, since metro systems connect two locations separated by a large distance, this leads to spatiotemporal dependencies between distant stations. As one convolution layer accounts for dependencies only between stations that are situated close to each other, in order to capture the spatiotemporal dependencies between distant stations, it is necessary to stack multiple GCNN layers to form a deep GCNN structure.

First, the Laplacian matrix is constructed based on the adjacency matrix of the metro network data, and the local spatiotemporal dependencies (*LSTD*) of the metro network structure are captured, as shown in Equation (3):

$$LSTD = F_{GCNN}(\theta, L_f(W), X), \tag{3}$$

where F_{GCNN} is one *GCNN* layer that performs Equation (2) on all metro stations, θ represents all the parameters learned in the model, L_f is the Laplacian matrix, W is the adjacency matrix of the metro network graph, and $X \in R^{M \times n}$ corresponds to the graphs of passenger flow volume of n metro stations in the past M intervals. Then, multiple *GCNN* layers are stacked to form a deep structure to capture the high-order (distant) spatiotemporal dependencies (HSTD) from the LSTDs, as shown in Equation (4):

$$HSTD = LSTD_N = F_{GCNN}(LSTD_{N-1}) = F_{GCNN}(\cdots F_{GCNN}(LSTD_1)), \tag{4}$$

where $LSTD_N$ refers to the *N*-order spatial dependencies and *M*-order temporal dependencies captured by stacking *N GCNN* layers. Assuming the size of the graph convolution kernel is 1, the specific deep GCNNs procedure is shown in Figure 4. By stacking multiple *GCNN* layers, we can capture the distant spatiotemporal dependencies of a citywide metro network.
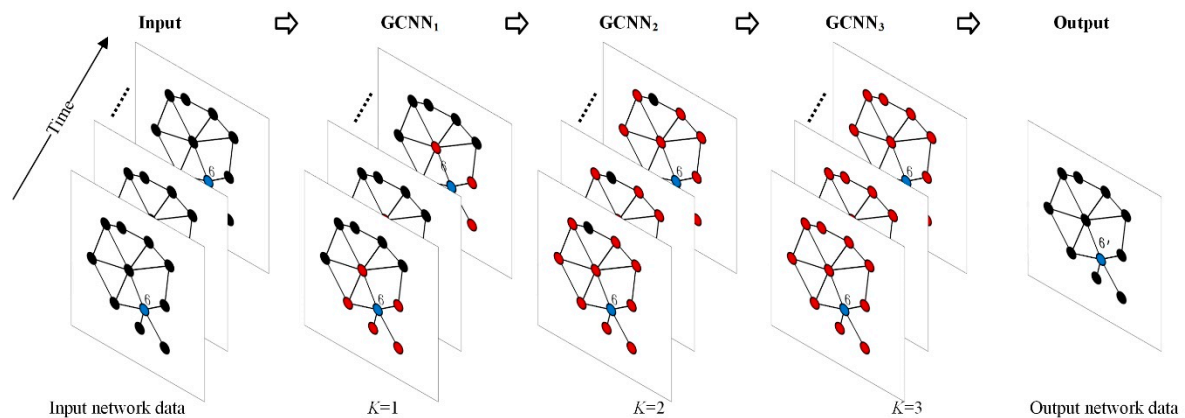

Figure 4. Procedure for capturing the spatiotemporal dependencies of metro network data using graph convolutional neural networks (GCNNs) (taking Station 6 as an example).

2.3. Using Spatiotemporal GCNNs for the Prediction of Station-Level Short-Term Passenger Flow Volume in a Citywide Metro Network

The general framework of the proposed novel DL-based model, STGCNNmetro, is shown in Figure 5. Clearly, the passenger flow of a metro system contains inflow and outflow. In this study inflow volume is the total passenger flow volume entering a station during a given time interval and outflow volume denotes the total passenger flow volume leaving a station. We first transform the historical citywide metro inflow volume graph and outflow volume graph at each time interval into structured spatiotemporal time-series. As the spatiotemporal dependencies have different patterns [15,26,35], the historical flow-volume time-series are divided into recent, daily, and weekly patterns. Then, three sub-models based on GCNNs are constructed to capture the features of these three patterns. Three outputs are fused based on parameter matrices, which assign different weights to the results of the analysis of different patterns in different stations. Finally, the ReLU function is selected to activate the fusion result to give the final prediction values.
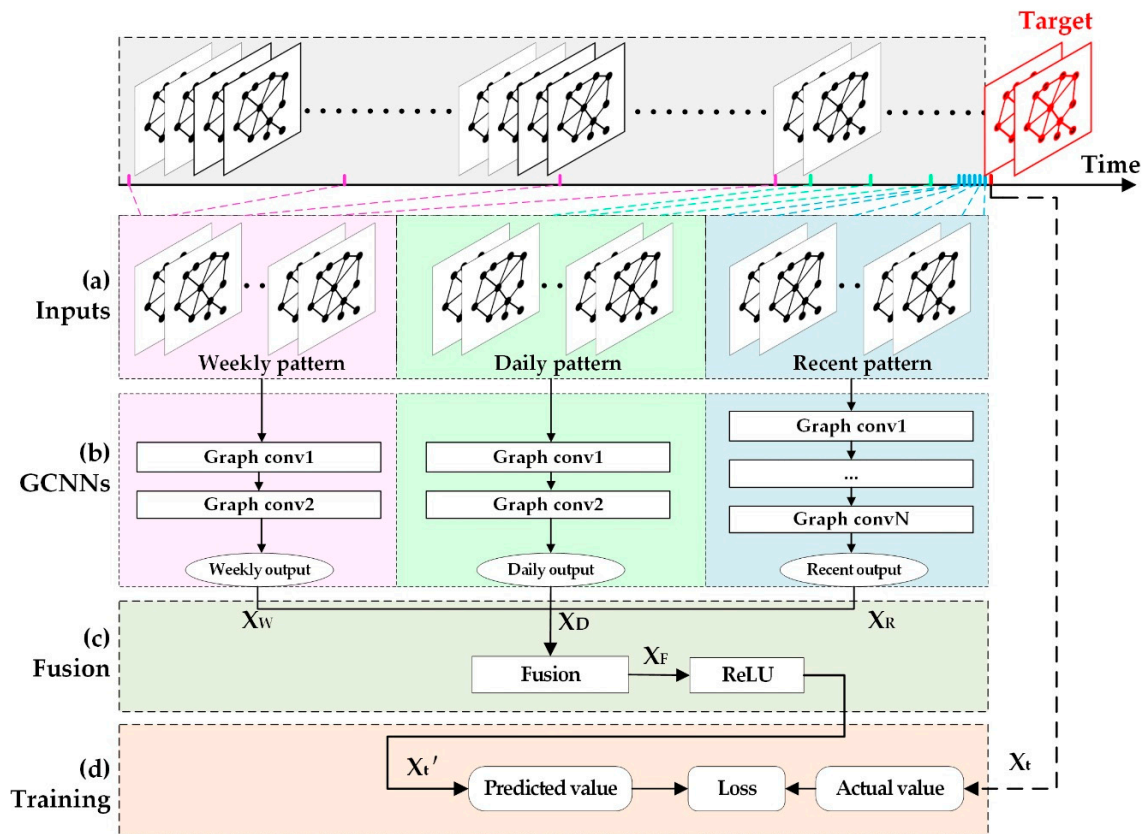
Figure 5. Architecture of the STGCNNmetro model. Conv: Convolution; X_t and X_t' represent actual and predicted values in the tth time interval, respectively; X_R, X_D, and X_W represent the outputs of the recent, daily, and weekly patterns, respectively; X_F is the fusion result of X_R, X_D, and X_W using a parametric-matrix-based method.

2.3.1. Input Datasets

The input datasets used in this study consist of spatiotemporal passenger flow-volume data from recent, daily, and weekly time scales. Recent inputs refer to the historical observations which are close to the target time interval, and daily and weekly inputs refer to the historical observations in the same time interval as the prediction target, but in daily or weekly periodic cycles. For example, assuming that the prediction target is the spatiotemporal flow volume between 9:00 and 9:10 a.m. on a Thursday, then the recent inputs are the historical observations between 8:00 and 9:00 a.m. on the same day, the daily inputs are the observations between 9:00 and 9:10 a.m. on the previous days, and the weekly inputs are the observations between 9:00 and 9:10 a.m. on the previous Thursdays.

Let X_t denotes the passenger flow volume data of the citywide metro network for the tth time interval and let $X_R^{(0)}$, $X_D^{(0)}$, and $X_W^{(0)}$ represent the recent, daily, and weekly historical observation data, respectively. Assume that the number of time intervals in $X_R^{(0)}$, $X_D^{(0)}$, and $X_W^{(0)}$ is r, d, and w, respectively. The time interval to be predicted is t, and the total number of time intervals in one day is m. So, $X_R^{(0)}$, $X_D^{(0)}$, and $X_W^{(0)}$ are defined in Equation (5). In the tth time interval, the inflow and outflow volumes in all stations can be denoted as a matrix $X_t \in R^{2 \times n}$. Three input datasets $X_R^{(0)}$, $X_D^{(0)}$, and $X_W^{(0)}$ can be represented by matrices with dimensions of $2r \times n$, $2d \times n$ and $2w \times n$, respectively, where n is the number of stations.

$$
\begin{aligned}
X_R^{(0)} &= (X_{t-r}, \dots, X_{t-2}, X_{t-1}), \\
X_D^{(0)} &= (X_{t-d \cdot m}, \dots, X_{t-2 \cdot m}, X_{t-1 \cdot m}), \\
X_W^{(0)} &= (X_{t-w \cdot 7 \cdot m}, \dots, X_{t-2 \cdot 7 \cdot m}, X_{t-1 \cdot 7 \cdot m})
\end{aligned}
\tag{5}
$$

2.3.2. Integrally Capturing Spatiotemporal Dependencies

In our task, the final output size should be same as the size of the input (i.e., $2 \times n$). The same problem was found in a video-sequence generating task where the input and output have the same resolution [36]. For this goal, we employ a special type of convolution, i.e., "same convolution." Several methods have been introduced to avoid the loss of resolution brought about by subsampling while preserving distant dependencies. As our approach is different from the classical CNN, we do not use subsampling, but only convolutions [37].

The recent, daily, and weekly patterns use a network structure composed of multiple GCNN layers, as shown in Figure 5. Taking the recent pattern as an example, the recent input $X_R^{(0)}$ is fed into a stacking stereogram graph convolution structure to integrally capture spatiotemporal dependencies. With several convolutions, the output of the recent pattern denoted as X_R is defined as follows:

$$X_R = HSTD\left(X_R^{(0)}\right). \tag{6}$$

Likewise, using the above operations, we can construct the daily and weekly components of Figure 5. The outputs of the daily and weekly components, denoted as X_D and X_W respectively, are as follows:

$$X_D = LSTD_2\left(X_D^{(0)}\right), X_W = LSTD_2(X_W^{(0)}) \tag{7}$$

2.3.3. Feature Fusion

A previous study demonstrated that crowd inflows and outflows in one region are both affected by recent, daily, and weekly dependencies. However, the degrees of influence may be very different [28]. Therefore, we adopted a parametric-matrix-based method to fuse the three components of Figure 5 (recent, daily, and weekly passenger flow volume patterns) as follows:

$$X_t{}' = \sigma(W_R {\cdot} X_R + W_D {\cdot} X_D + W_W {\cdot} X_W), \tag{8}$$

where X_t' is the prediction target in the tth time interval, ∘ is the Hadamard product (i.e., element-wise multiplication), and W_R, W_D, and W_W are the learnable parameters that adjust the degrees affected by recent, daily, and weekly dependencies, respectively. The fusion result is activated by σ (i.e., ReLU) to obtain the prediction target.

2.3.4. Model Training

To train the STGCNNmetro model, the mean-square error (MSE) is used as the loss function. As shown in Equation (9), y_i is the actual value, $\tilde{y}_i$ is the predicted value, and n is the total number of values to be predicted. All samples are divided into three sub-datasets: A training set, a validating set, and a testing set. The training set is fed into the model in batches. For each batch, the value of the loss function is calculated after forward propagation. Then, the loss is back-propagated layer-by-layer and an optimizer updates all trainable parameters according to the loss. The Adam optimizer [38] is a widely adopted optimizer in ML, as it generates adaptive learning rates for different parameters. Here, Adam is applied as the optimizer. By minimizing the loss, all trainable parameters are trained.

$$Loss = MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \tilde{y}_i\right)^2, \tag{9}$$

3. Experiments

3.1. Experimental Data

A metro passenger flow dataset was generated from card-swiping data of the metro system of Shanghai, China, from 1 April to 30 April 2015. During this period, there were about nine million

card records per day, covering 289 unique metro stations from all 14 lines of the Shanghai metro. The distribution of all metro lines and stations are shown in Figure 6.

Figure 6. Map of the Shanghai metro [39].

Corresponding to human activities, the number of metro passengers is significantly less before 5:30 a.m. and after 11:00 p.m. Therefore, we selected the time range of between 5:30 a.m. and 11:00 p.m. As our study period. The time interval was set to 10 min, so that there are 105 time intervals per day. The passenger inflow volume and outflow volume during each time interval was determined based on the card records.

The dataset is divided as follows: The data for April 24 and 25 (including one working day and one non-working day) is used as the verification set, the data before April 25 is used as a training set, and the data for five days after April 25 (including four working days and one non-working day) is used as a test set.

3.2. Comparative Experiments

3.2.1. Evaluation Metrics and Baseline Models

To measure and evaluate the performances of different methods, the root-mean-square error (*RMSE*) and mean absolute percentage error (*MAPE*) were adopted. The corresponding calculations are as follows:

$$MAPE(\tilde{y}, y) = \frac{100}{n} \sum_{i=1}^{n} \frac{|y_i - \tilde{y}_i|}{y_i},$$
$$RMSE(\tilde{y}, y) = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \tilde{y}_i)^2} \tag{10}$$

We compared our STGCNNmetro model with those of seven well-known baselines models. Brief introductions to these are shown in Table 1.

Table 1. Brief introductions to the baselines models used for comparison with the model used in this study.

No.	Name	Description
1	MLR	Multivariable linear regression (MLR) model.
2	LSVR	A version of support vector machine (SVM) for regression, called support vector regression (SVR). Here we used linear SVR (LSVR).
3	Bayesian	Bayesian regression model.
4	PCA-kNN	A mixture of principal component analysis (PCA) and k-nearest neighbor (kNN) regression [6]. PCA is used to select the principal components which are input into kNN for prediction.
5	NMF-kNN	A mixture of non-negative matrix factorization (NMF) and kNN regression which is similar to PCA-kNN above.
6	LSTM	Long short-term memory (LSTM). Here, the LSTM model has multiple LSTM layers and one fully-connected layer.
7	M-CNN	A convolutional neural network (CNN)-based model proposed by Ma et al. [29] which transforms the metro-network-based passenger flow volume into a two-dimensional image whose horizontal axis represents time and whose vertical axis represents the metro station. Prediction is made by performing convolutions on the image.

3.2.2. Experimental Environment and Settings

All experiments were run on a graphics processing unit (GPU) platform with an NVIDIA GeForce GTX 950 graphics card with 8GB of GPU memory, and were implemented in the Python 3 programming language, using Keras [40] and TensorFlow [41] as the DL packages and Sklearn [42] as the general ML package.

The procedure for tuning parameters is an essential part of most DL-based models [15,29,43]. Here, only the settings are listed which obtain the best prediction accuracy, and the tuning procedure will be detailed in Section 3.3. The number of time intervals for recent, daily, and weekly patterns were set as $r = 7$, $d = 1$, and $w = 1$, respectively. Since two passenger flow volumes were predicted, the number of kernels in the last layer was set to 2, while the number of other layers was set to 288. For the recent pattern, the graph convolution kernel size (i.e., K) was set to 3 and the network depth or the number of graph convolution layers was set to 5. The daily pattern and the weekly pattern had the same graph convolutional network structure. Unlike for the recent pattern, the main contribution of the two periodic patterns (daily and weekly) were temporal dependencies. Thus, the K values of the daily and weekly patterns were both set to 1 and their network depths were both set to 2.

We trained the STGCNNmetro model by minimizing the mean-square-error for 100 epochs with a batch size of five. The initial learning rate was 1×10^{-3} with a decay rate of 0.95 after every

20 epochs. For the seven comparison models, we also selected the parameters which gave the best results via tuning.

For inputs, the multivariable linear regression (MLR) model was given only historical observation from three patterns (i.e., recent, daily, and weekly) to predict each passenger inflow or outflow volume, while the other seven models were given inflow volume and outflow volume for all stations from the three patterns, taking into account spatiotemporal correlation.

3.2.3. Experimental Results and Analysis

The test results for all models are shown in Table 2. The RMSEs were calculated from all prediction data. The MAPEs were calculated from the data whose actual value is not 0. Thus, the case where the denominator is 0 is avoided.

The following can be seen from Table 2: (1) Compared with the other seven models, the RMSE of the STGCNNmetro model is the smallest, and the MAPE of the PCA-kNN model is the smallest. The MAPE of the STGCNNmetro is the second smallest and very close to that of the PCA-kNN model, with a difference of 0.14%. However, the RMSE of PCA-kNN is 8.05 larger than that of STGCNNmetro. Thus, STGCNNmetro is better than PCA-kNN overall, which means that STGCNNmetro is better than all other models; (2) the RMSEs of the STGCNNmetro, Bayesian, and LSVR models are similar. However, the training time of the Bayesian model is over 4.5 h, which is nearly nine times that of STGCNNmetro, and the training time of LSVR is close to six hours, which is nearly 11 times that of STGCNNmetro; (3) the RMSE of the STGCNNmetro model is 9.46 less than that of the M-CNN model, which reflects the advantages of STGCNNmetro in capturing the irregular spatio-temporal dependencies; (4) the LSTM model has poor prediction results, which means that it is not good at capturing spatial dependencies of the metro network; and (5) the MLR model directly performs regression operations on historical data, while the passenger flow has spatiotemporal correlation, nonlinearity, and volatility, so that insufficient factors are considered by this model and the result is worse than for the other seven models.

Table 2. Root-mean-square error (RMSE) and mean absolute percentage error (MAPE) values for different models.

No.	Model	RMSE	MAPE	Time (s)
1	STGCNNmetro	32.50	24.35%	1910.13
2	Bayesian	33.15	27.71%	16,415.01
3	LSVR	33.81	28.65%	21,050.68
4	PCA-kNN	40.55	24.21%	7.09
5	M-CNN	41.96	24.75%	303.02
6	NMF-kNN	42.94	25.16%	41.96
7	LSTM	45.40	30.69%	3410.59
8	MLR	49.41	34.12%	1190.64

We first choose two models—one whose MAPE is the closest to that of the STGCNNmetro model and whose RMSE is relatively small (PCA-kNN) and another whose RMSE is closest to that of STGCNNmetro and whose MAPE is relatively small (Bayesian). Then, we take the Shenzhuang metro station, the most crowded station in Shanghai [44], as an example to explore more details of the models. The results of the passenger flow predictions from the three models for working days and non-working days are compared and analyzed as follows:

1. Prediction Results for Passenger Inflow Volume

From the results of the prediction of inflow volume using the test set, shown in Figure 7, it can be seen that the values predicted by the STGCNNmetro, Bayesian, and PCA-kNN models correspond well to the actual values. In some time intervals (i.e., morning and evening, see the red circle in Figure 7), the values predicted by the STGCNNmetro model are superior to those predicted by the Bayesian

and PCA-kNN models. The following is a detailed analysis of the prediction of inflow volume for a Sunday (non-working day) and a Monday (working day):
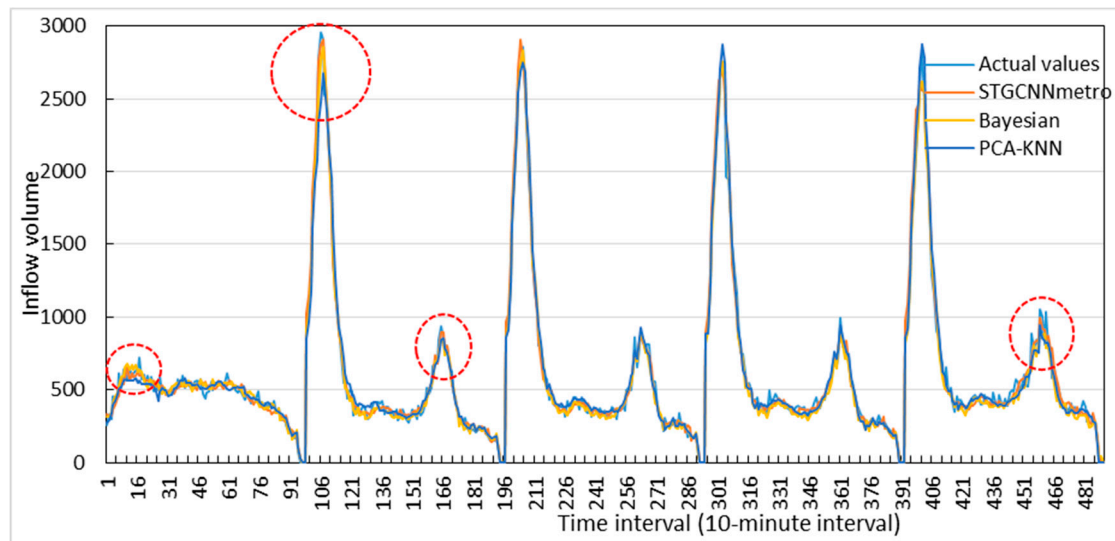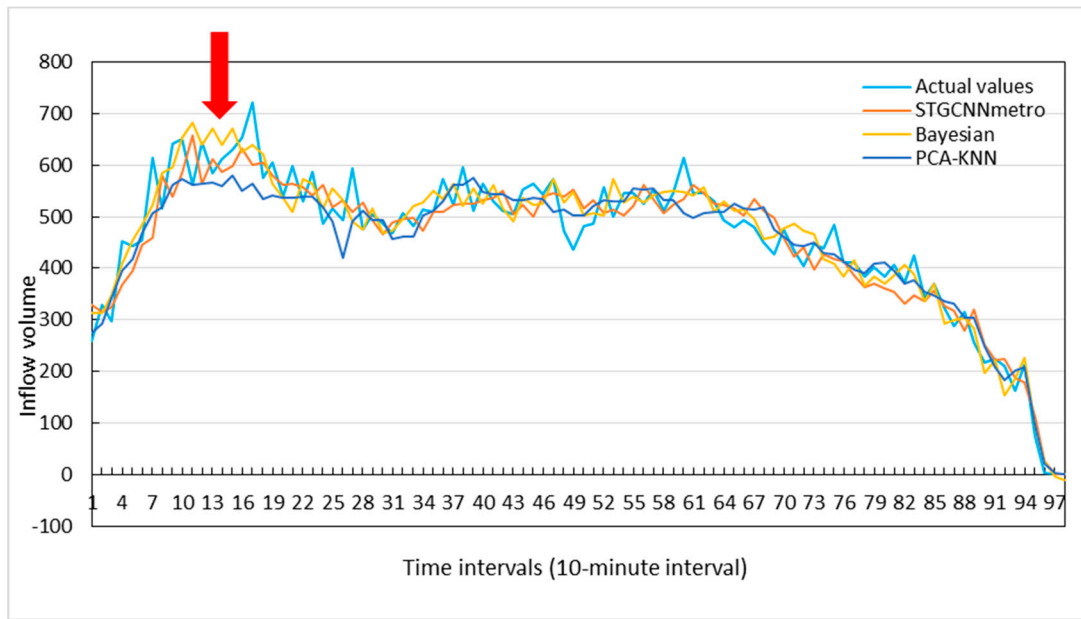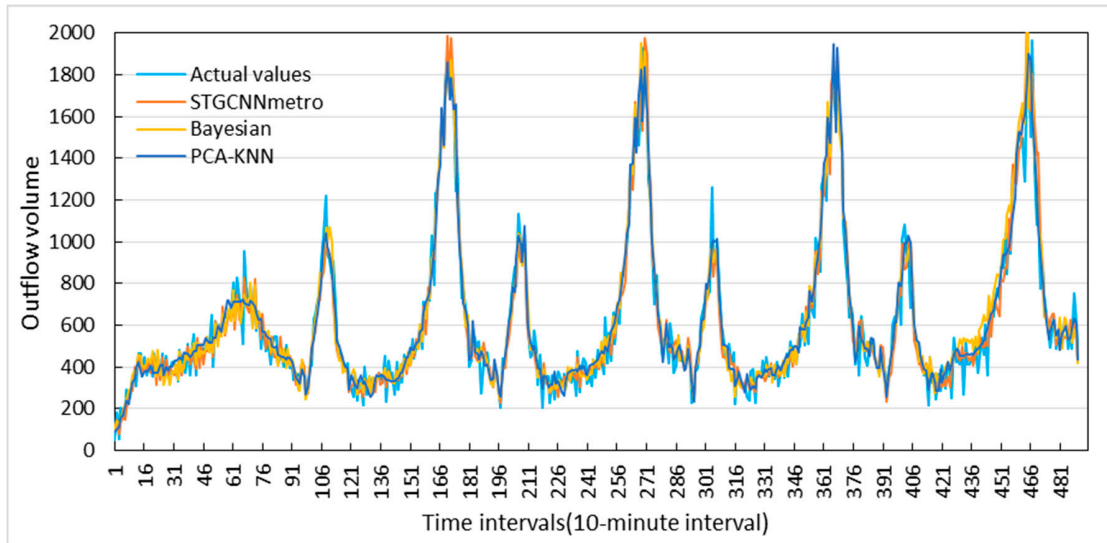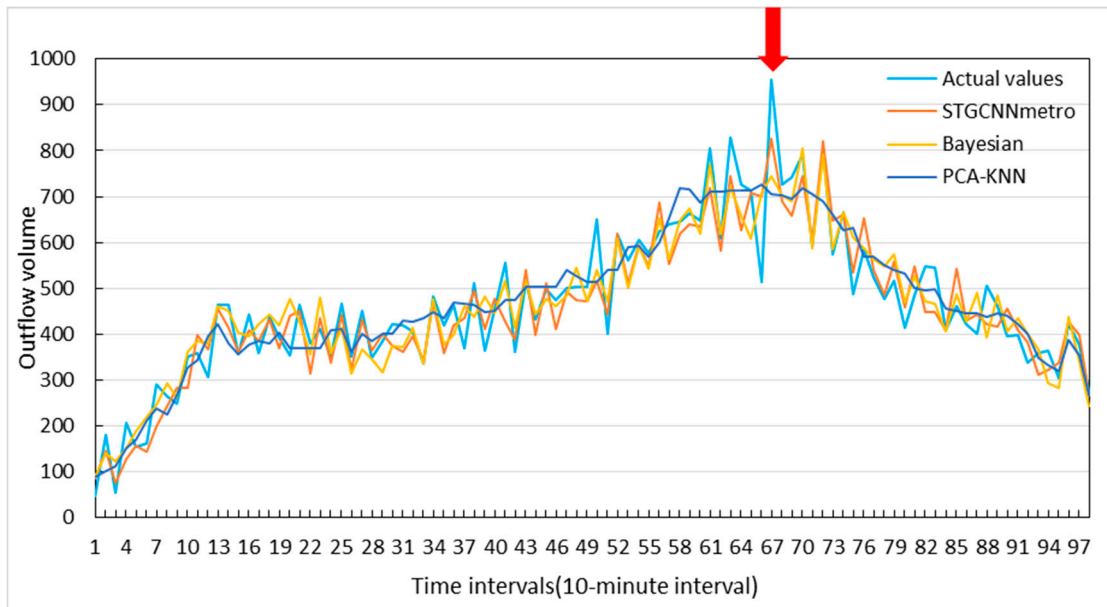

Figure 7. A comparison of the predicted and actual values of passenger inflow volume per 10 min using the test data from the Shenzhuang metro station, Shanghai. Data cover five consecutive days (from Sunday to Thursday). STGCNNmetro: spatiotemporal graph convolutional neural networks model for metro; Bayesian: Bayesian regression model; PCA-kNN: A mixture of principal component analysis (PCA) and k-nearest neighbor (kNN) regression [6].

As shown in Figure 8a, the STGCNNmetro model better predicts the actual values of passenger inflow volume for the non-working day than the Bayesian and PCA-kNN models. Between 8:30 and 9:30 a.m. (see the red arrow in Figure 8a), the predictions of the STGCNNmetro model are obviously superior to those of the Bayesian and PCA-kNN models. The predictions of the Bayesian model are too large, while those of the PCA-kNN are too small. As shown in Figure 8b, all three models effectively predict passenger inflow volume for the working day. However, in the morning and evening peaks, the predictions of the STGCNNmetro model are superior (see the red arrow in Figure 8b).

2.   Prediction Results for Passenger Outflow Volume

As shown in Figure 9, the passenger outflow volume is more variable than the inflow volume, and the prediction results are therefore slightly worse. Nevertheless, the overall predictive abilities of the three models are good.

Based on the results of the prediction of outflow volume on a non-working day shown in Figure 10a, the STGCNNmetro and Bayesian models perform significantly better than the PCA-kNN model, and the prediction results of the STGCNNmetro model during peak periods are better than those of both the Bayesian and PCA-kNN models (see the red arrow in Figure 10a). Furthermore, based on the results of the prediction of outflow volume on a working day, as shown in Figure 10b, the STGCNNmetro and Bayesian models generally perform better than the PCA-KNN model. The ability of the STGCNNmetro model to capture peaks in the evening is better than that of the Bayesian model (see the red arrow in Figure 10b). However, its ability to capture peaks in the morning is slightly worse than that of the Bayesian model.
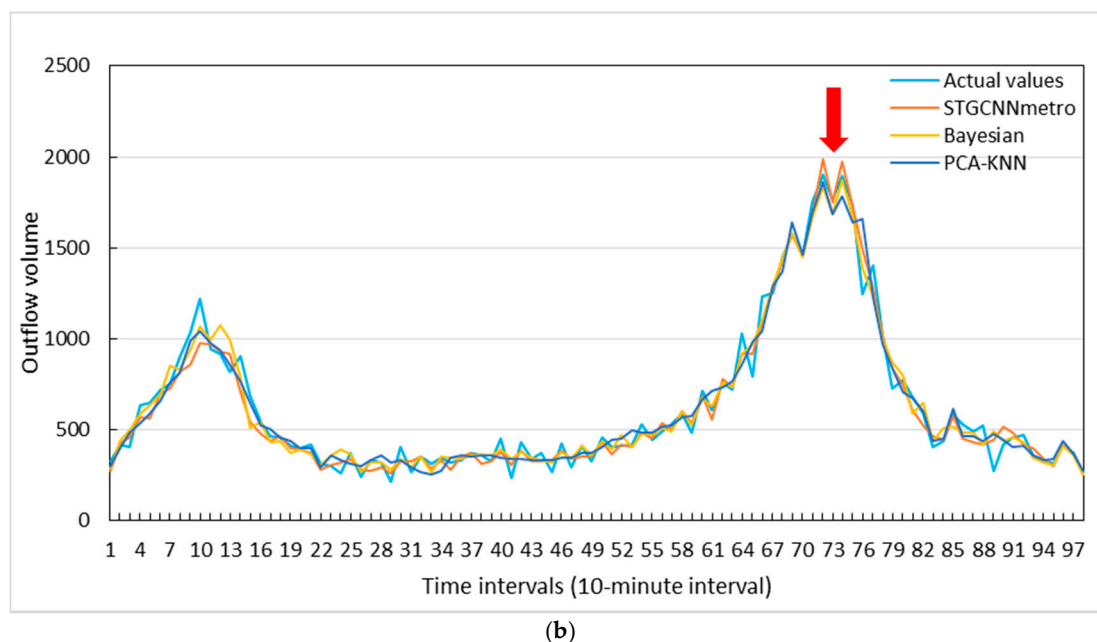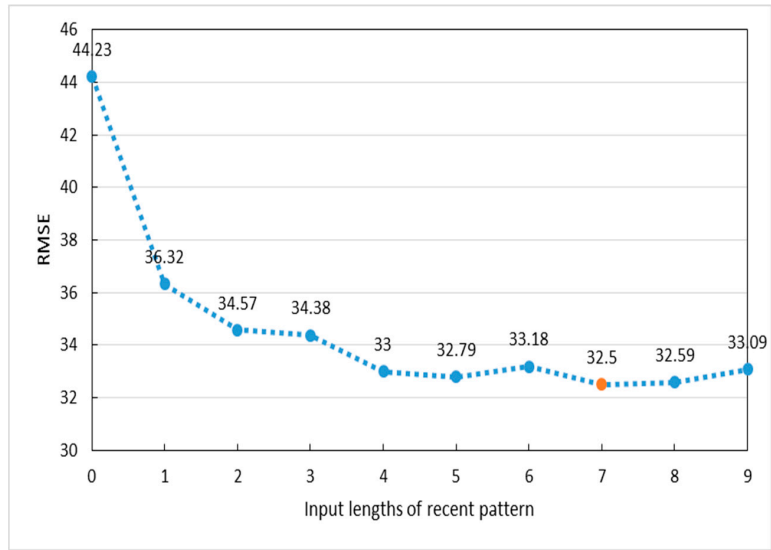
(**a**)

(**b**)

Figure 8. A comparison of the predicted and actual values of passenger inflow volume between 6:40 a.m. and 11:00 p.m. at the Shenzhuang metro station for (**a**) a Sunday (non-working day) and (**b**) a Monday (working day).

In summary, the predictive ability of the STGCNNmetro model is generally better than those of the Bayesian and PCA-KNN models. The STGCNNmetro model's prediction of peaks is usually more accurate than those of the other two models, although its prediction of peak values of outflow volume on working day mornings is slightly worse than those of the Bayesian model. This may be related to the improvement of computational efficiency achieved through convolution sharing parameters. Unlike the STGCNNmetro model, the Bayesian model has specific parameters for each station, and its training times are almost nine times that of the STGCNNmetro model. Compared with the Bayesian model, the STGCNNmetro model shows more advantages for practical application.

Figure 9. A comparison of the predicted and actual values of passenger outflow volume per 10 min using the test data from the Shenzhuang metro station. Data cover five consecutive days (from Sunday to Thursday).

3.3. Tuning Parameters

Hyperparametric tuning is an indispensable process for DL models. The number of graph convolution kernels and graph convolution layers represent spatial attributes. The lengths of the input sequences of the three patterns (i.e., recent, daily, and weekly) represent temporal attributes. This section focuses on the adjustment process of the above three types of parameters (i.e., the number of graph convolution kernels, graph convolution layers, and the lengths of the input sequences of the three patterns).

(**a**)

Figure 10. *Cont*.

(**b**)

Figure 10. A comparison of the predicted and actual values of outflow volume between 6:40 a.m. and 11:00 p.m. for the Shenzhuang metro station on (**a**) a non-working day and (**b**) a working day.

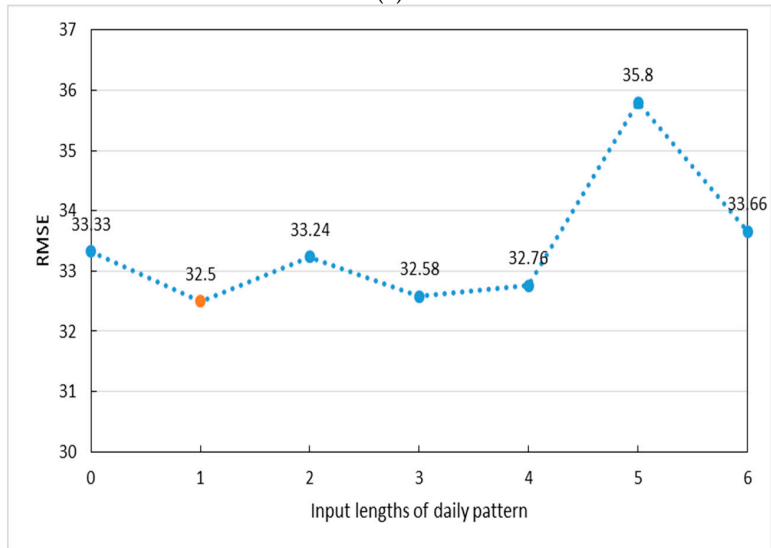### 3.3.1. Different Input Lengths of the Recent, Daily, and Weekly Passenger Flow Volume Patterns

Here, we verify the impact of different input lengths of the three patterns (i.e., recent, daily, and weekly), as shown in Figure 11. Figure 11a shows the effect of temporal closeness when d and w are fixed as 1 but r is changed. For example, $r = 0$ means that we do not employ the recent pattern, and results in a very high RMSE of 44.23. It can be seen that the RMSE first decreases and then increases as the input length increases, with $r = 7$ having the lowest RMSE. Figure 11b depicts the effect of the daily pattern when r is set as 7 and w as 1 but d is changed. It can be seen that $d = 1$ gives the lowest RMSE. The model without the daily pattern (i.e., $d = 0$) has a higher RMSE than the model with $d = 1$, 2, 3, 4, but a lower RMSE than the model with $d = 5, 6$, meaning that short-range periods are always well-modeled and long-range periods may be hard to model or not helpful. Figure 11c presents the effect of the weekly pattern when r and d are fixed as 7 and 1, respectively, while w is changed from 0 to 2. It can be seen that the model with $w = 1$ outperforms the others. For the weekly pattern, the long-range period may be hard to model or useless.

3.3.2. The Effect of Different Numbers of Graph Convolution Layers
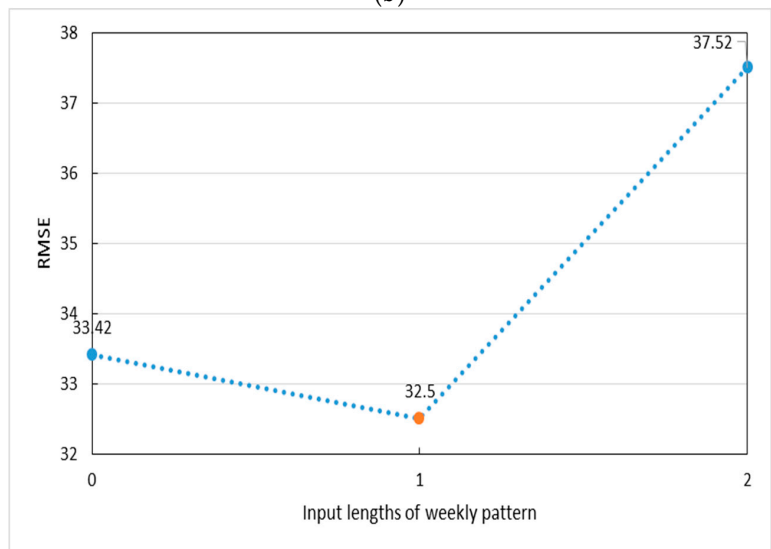
From Figure 12, it can be seen that as the number of graph convolution layers increases, the RMSE of the model first decreases and then increases, demonstrating that more graph convolution layers can give a better result since they can capture not only close spatial dependence, but also distant ones. However, when the network is very large (e.g., number of network layers ≥ 9), the training result becomes worse. The newly included stations are from Shanghai's urban fringe and have little contribution to the passenger flow volume at the core stations. An increasing number of layers leads to more parameters and to greater training difficulty, and as a result, the RMSE increases.

(**a**)

(**b**)

(**c**)

Figure 11. Experimental results for different input lengths of (**a**) the recent pattern, (**b**) the daily pattern, and (**c**) the weekly pattern. RMSE: Root-mean-square error.
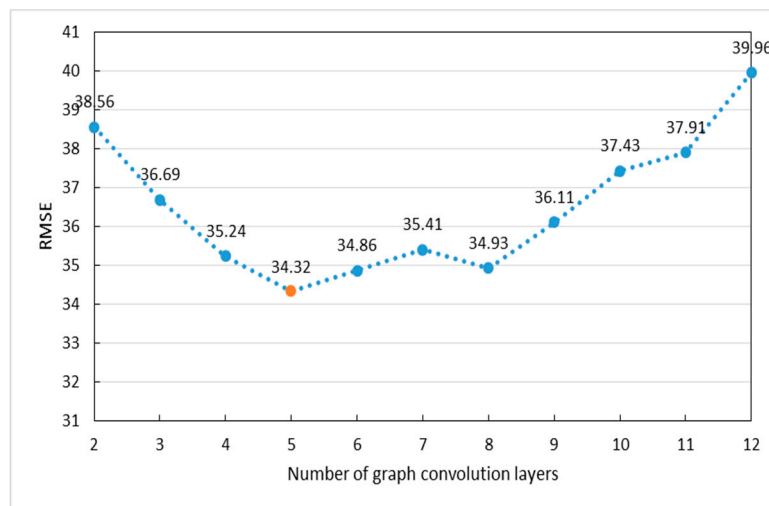
Figure 12. Experimental results with different numbers of graph convolution layers.

3.3.3. The Effect of Different Numbers of Kernels

From the results shown in Figure 13, it can be seen that up to a certain level, more kernels result in a lower RMSE. However, when the number of kernels is very large (e.g., ≥352), the RMSE increases. An increasing number of kernels leads to more parameters and to greater training difficulty, and the RMSE therefore increases.

Figure 13. Experimental results with different numbers of kernels.

4. Discussion

4.1. Spatial Distribution of Error

This section discusses the spatial distribution of the errors of the STGCNNmetro model for working days and non-working days. The daily period of interest is from 6: 40 a.m. To 11:00 p.m. We calculate the RMSE of each station's predicted inflow and outflow volume and calculate the average actual inflow and outflow volume for each station as a reference. The latter is called the station mean actual value (*SMAV*), and its calculation is shown in Equation (11):

$$SMAV = \frac{1}{T} \sum_{i=1}^{T} y_i, \qquad (11)$$

where y_i is the actual value corresponding to the ith time interval and T is the number of predicted time intervals.

From the distribution maps of RMSE and SMAV shown in Figure 14, it can be seen that there is a positive correlation between the model errors and the actual values of passenger flow volume. To further verify the relationship between the two, they are plotted against each other in Figure 15.

Based on the scatter plot of RMSE and SMAV shown in Figure 15, we further carry out regression analysis on these two measures and obtain the linear regression equation and the coefficient of determination (R^2). The results show that, overall, the RMSE is positively correlated with the SMAV. However, several points show some differences. For example, although the SMAV of the People's Square station (see marker "1" in Figure 14) is the largest of all stations, its RMSE is not the largest. Additionally, the RMSEs of the stations of Xu Jidong (see marker "2" in Figure 14), Hongqiao Railway, (see marker "3" in Figure 14), and Shanghai South Railway (see marker "4" in Figure 14) are all abnormally high. The reason for this may be that they are located near the railway station and are affected by external uncertainties. Furthermore, the RMSEs of the stations of Caohejing Development Zone (see marker "5" in Figure 14) and Lu Jiazui (see marker "6" in Figure 14) are also abnormally high. All of these abnormal points should be explored using other types of data in the future.
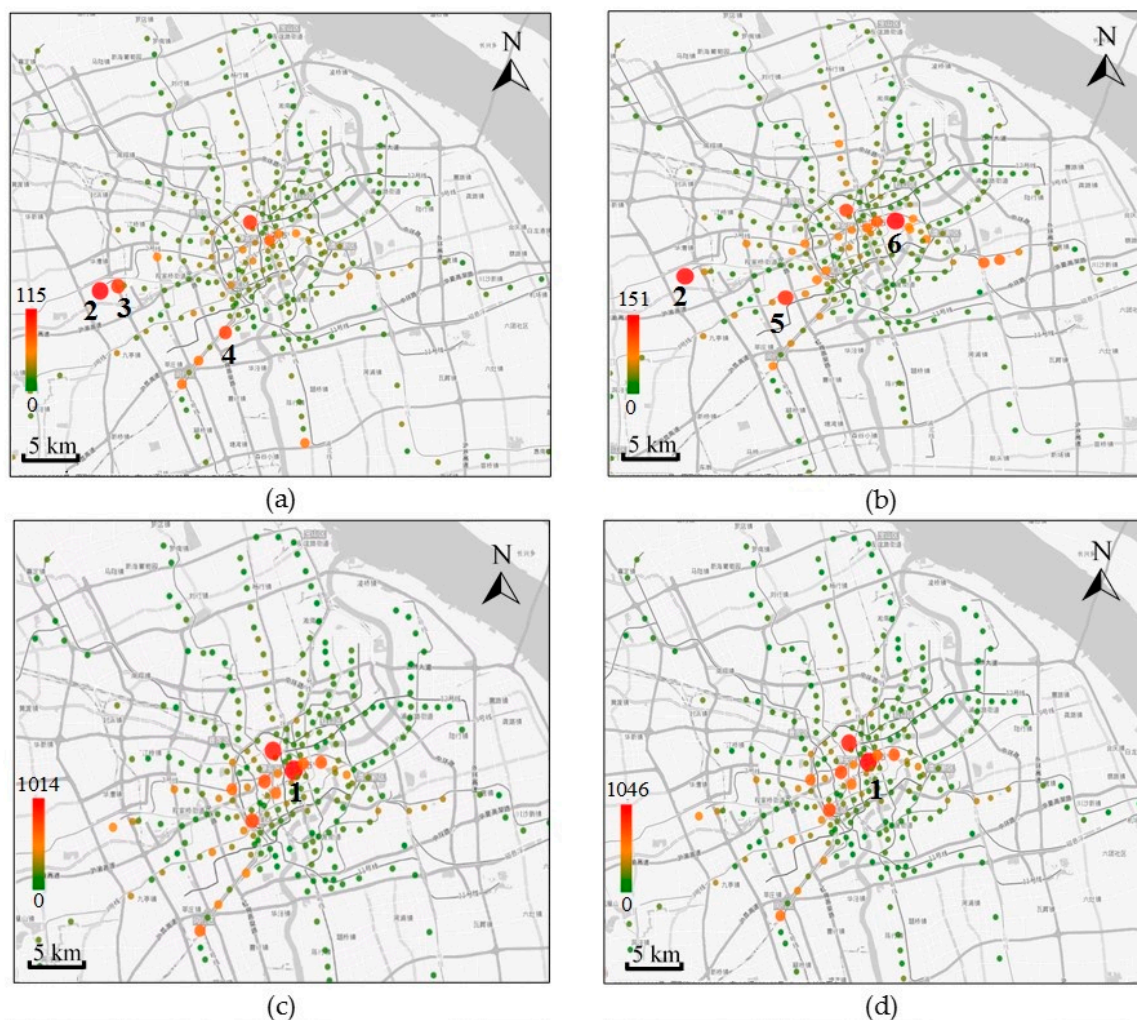

Figure 14. *Cont.*
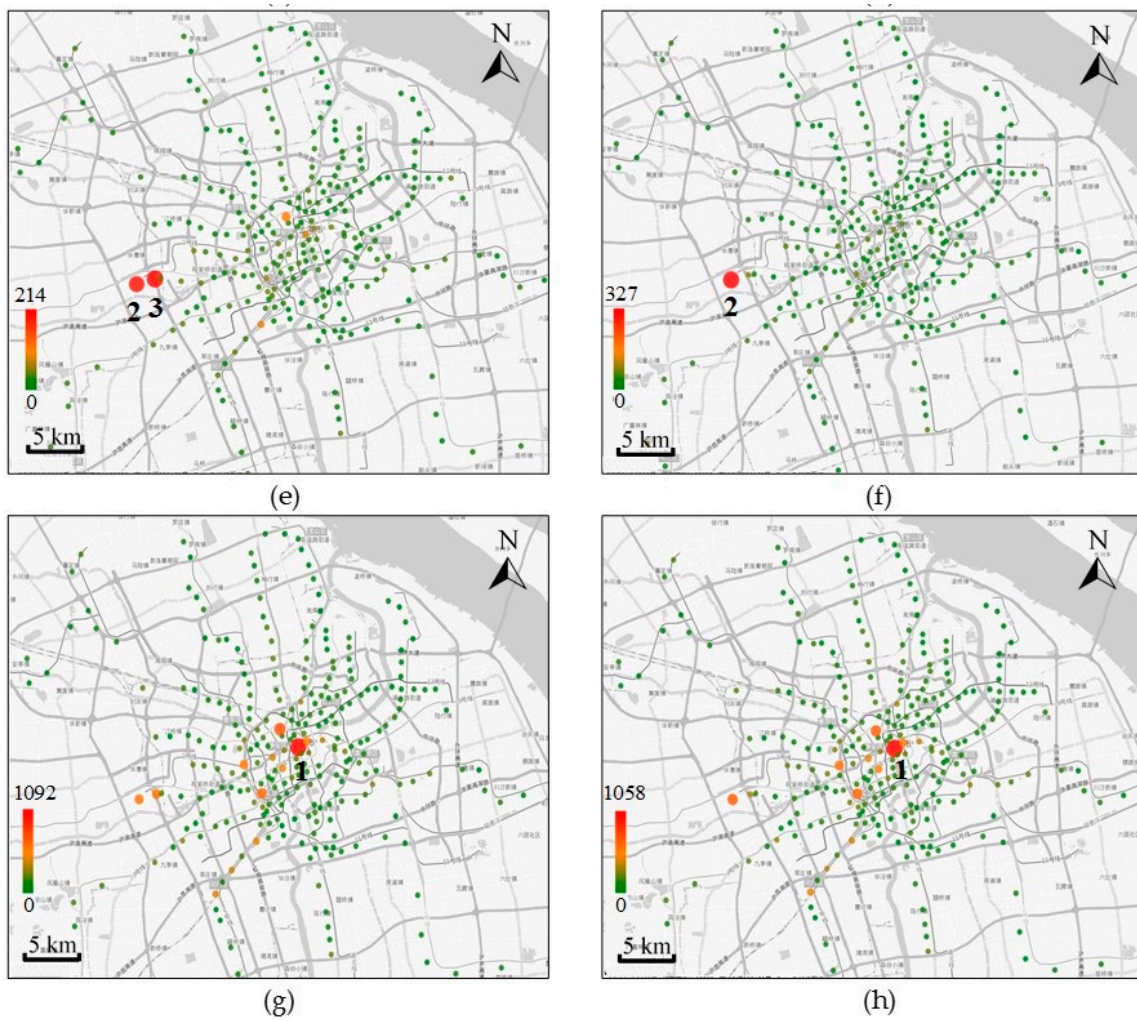
(e)

(f)

(g)

(h)

Figure 14. The spatial distribution of (**a**) the RMSE of passenger inflow volume, (**b**) the RMSE of passenger outflow volume, (**c**) the station mean actual value (SMAV) of passenger inflow volume, and (**d**) the SMAV of passenger outflow volume, at each metro station on working days; and the spatial distribution of (**e**) the RMSE of passenger inflow volume, (**f**) the RMSE of passenger outflow volume, (**g**) the SMAV of passenger inflow volume, and (**h**) the SMAV of passenger outflow volume, at each station on non-working days. The numbers 1–6 indicate the metro stations of People's Square, Xu Jidong, Hongqiao Railway, Shanghai South Railway, Caohejing Development Zone, and Lu Jiazui, respectively.
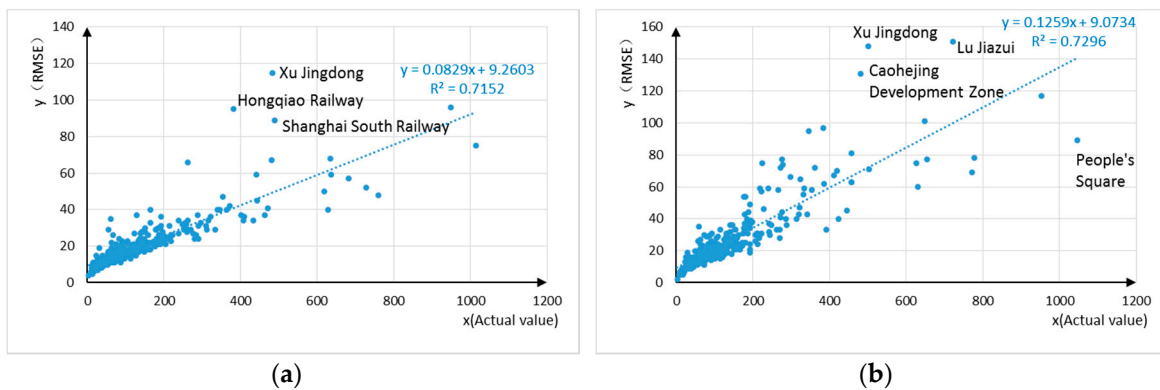

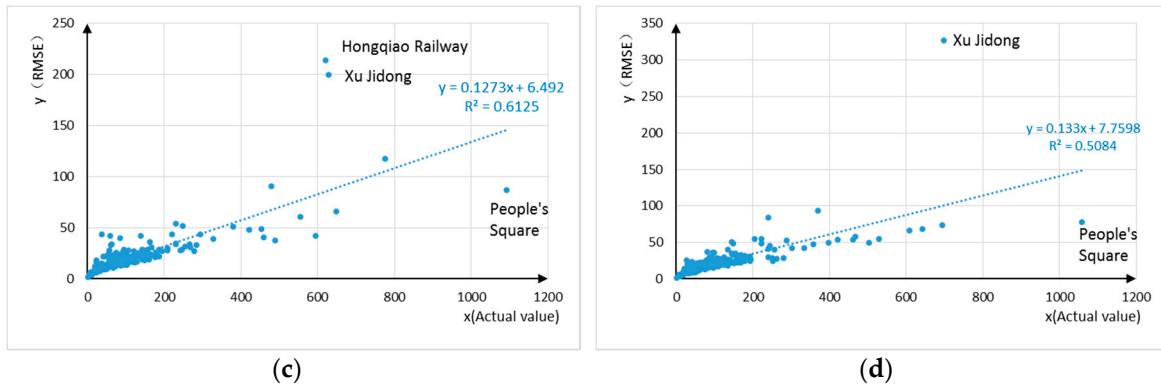(**a**)

(**b**)

Figure 15. *Cont.*

(c) (d)

Figure 15. The relationship between the RMSE and SMAV of (**a**) passenger inflow volume at each station on working days, (**b**) passenger outflow volume at each station on working days, (**c**) passenger inflow volume at each station on non-working days, and (**d**) passenger outflow volume at each station on non-working days.

4.2. Temporal Distribution of Error

This section analyzes the temporal distribution of the errors in the output of the STGCNNmetro model using the data of the test set for five consecutive days (Sunday to Thursday) as an example. The daily period of interest is from 6:40 a.m. To 11:00 p.m. We calculate the RMSE of the predicted passenger inflow and outflow volumes for all stations for every 10 min interval, and also calculate the average of the actual flow volume of all stations at this time interval. The latter is called the time mean actual value (TMAV), and its calculation is shown in Equation (12):

$$TMAV = \frac{1}{S} \sum_{i=1}^{S} y_i, \tag{12}$$

where y_i is the actual value corresponding to the ith station, and S is the total number of stations. The result is shown in Figure 16. In general, the error distribution in the temporal dimension is similar to that in the spatial dimension, and the RMSE is positively correlated with the *TMAV*.
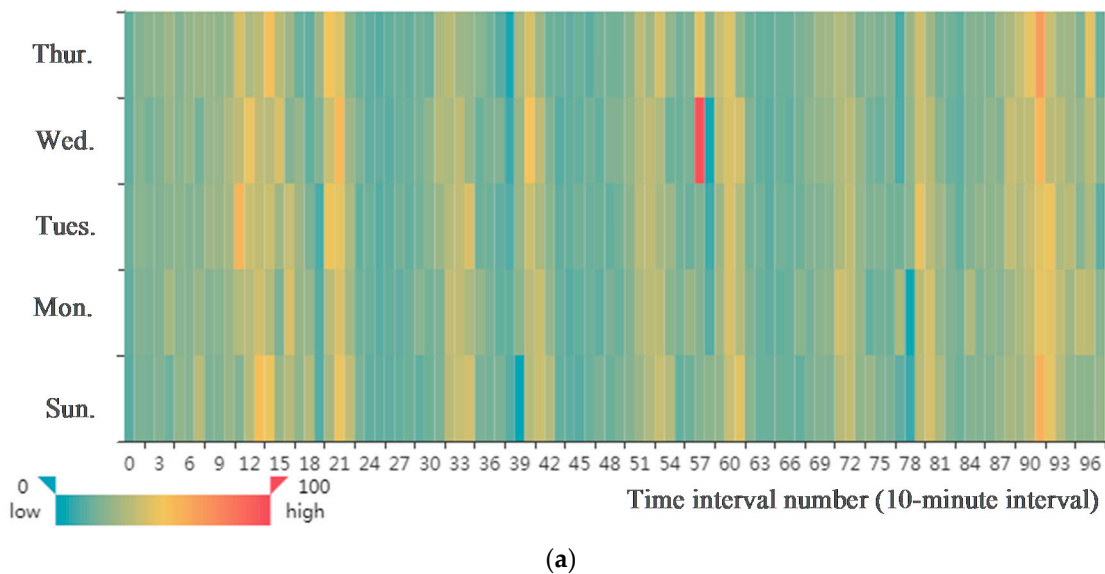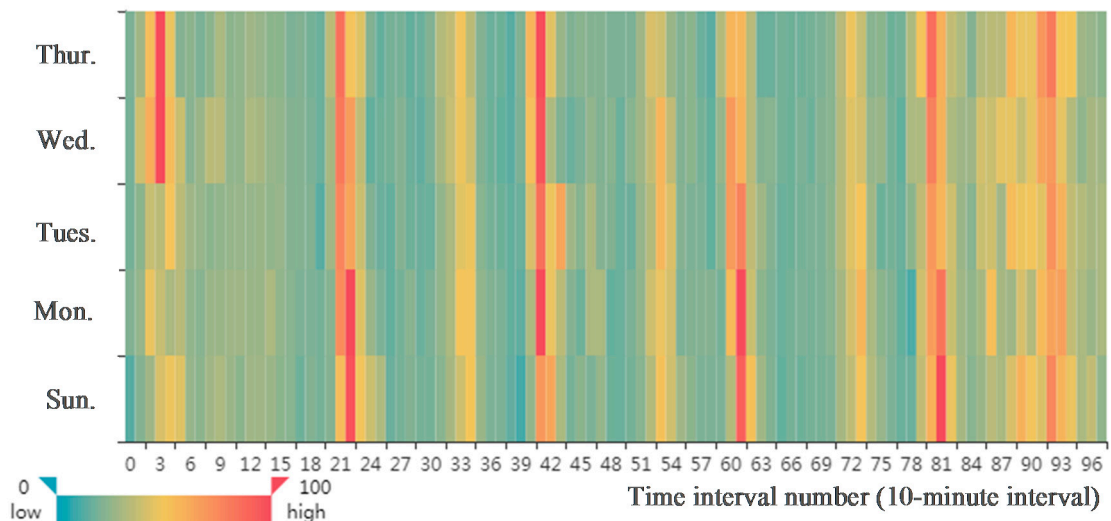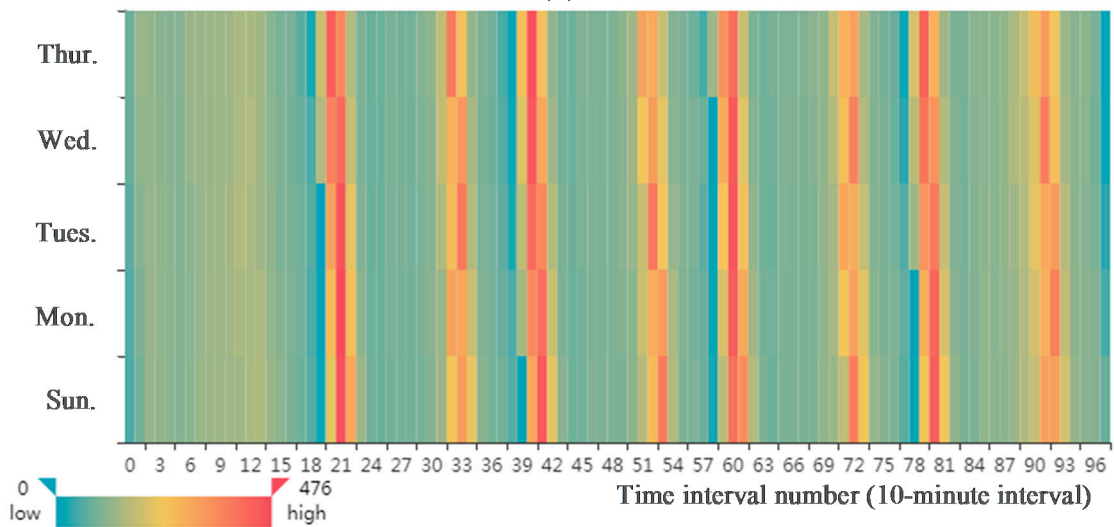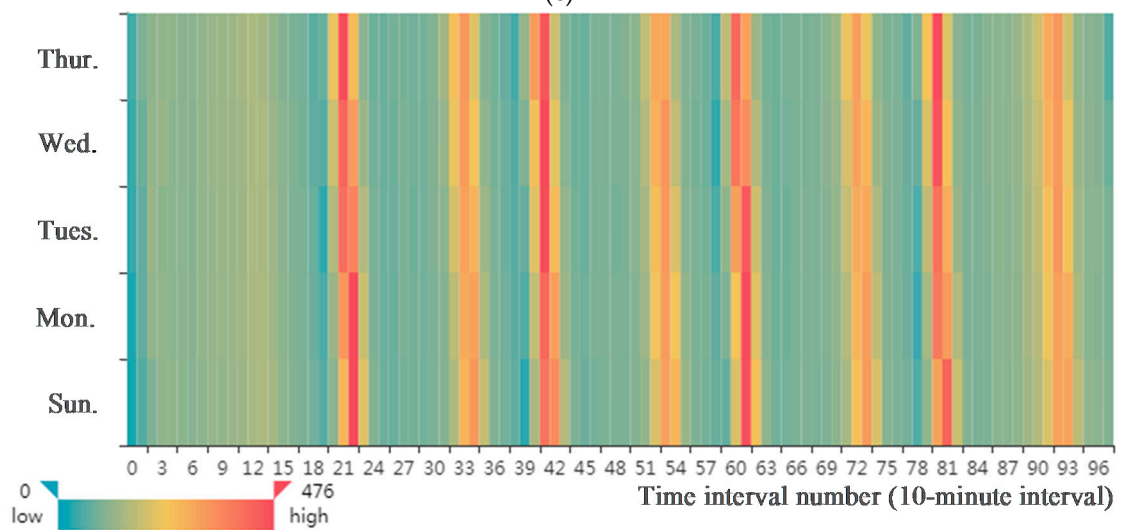

(a)

Figure 16. *Cont.*

(**b**)

(**c**)

(**d**)

Figure 16. The RMSE distribution of (**a**) the passenger inflow volume at each 10-min interval, (**b**) the passenger outflow volume at each 10-min interval; and the time mean actual value (TMAV) distribution of (**c**) the passenger inflow volume at each 10-min interval and (**d**) the passenger outflow volume at each 10-min interval.

Based on the scatter plot of RMSE and TMAV shown in Figure 17, we further carry out regression analysis on these two measures and obtain the linear regression equation and the coefficient of determination. In general, the error distribution in the temporal dimension is similar to that in the spatial dimension, and the RMSE is positively correlated with the TMAV.
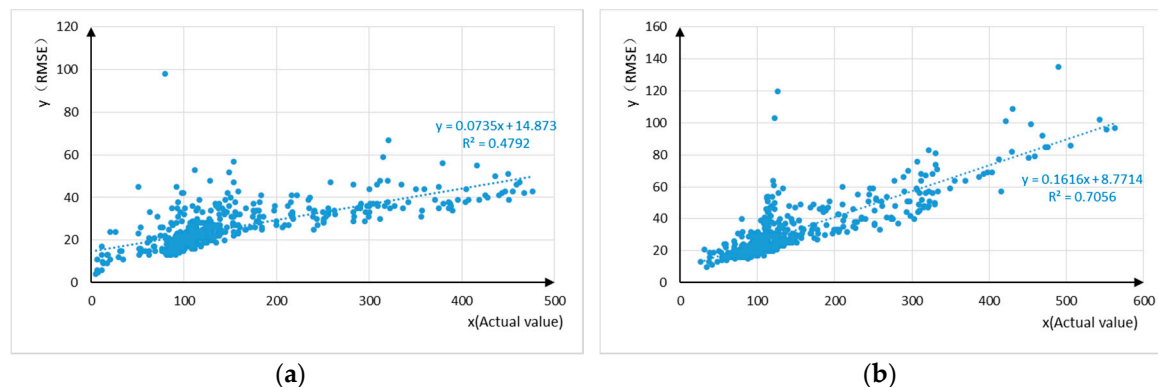

Figure 17. The relationship between the RMSE and TMAV for (**a**) passenger inflow volume and (**b**) passenger outflow volume, at each 10-min interval.

5. Conclusions and Future Work

In this paper, we propose a spatiotemporal graph convolutional neural network model, called STGCNNmetro, to predict the passenger inflow and outflow volumes of a citywide metro network. We apply stereogram graph convolution kernels to historical metro passenger flow-volume time-series to seamlessly capture the irregular spatiotemporal dependencies between stations along the metro network. Furthermore, a deep structure composed of GCNNs is constructed to capture the distant spatiotemporal dependencies at the citywide level. Finally, we fuse the spatiotemporal dependencies captured from recent, daily, and weekly patterns to form final predicted values of passenger inflow and outflow volumes. The effectiveness of the STGCNNmetro model is verified by predicting the citywide short-term passenger flow volume of the metro network of Shanghai, China. Experiments show that our STGCNNmetro model outperforms seven baseline models, namely LSVR, PCA-kNN, NMF-kNN, Bayesian, MLR, M-CNN, and LSTM. It achieves an "end-to-end" prediction that can accept raw metro passenger flow-volume data and automatically capture the effective features of the citywide metro network. By discussing the spatial and temporal error distributions of the results of the STGCNNmetro model, we conclude that the errors are positively correlated with the actual values for most predicted targets. However, there are also several stations with abnormally high RMSEs.

In the future, we will further optimize the model's network structure and parameters to explore the reason for the abnormal prediction points. Moreover, we will continue our work to design an "end-to-end" DL model for the multistep prediction of metro passenger flow volumes.

Author Contributions: Conceptualization, Yong Han; Data curation, Cheng Wang; Formal analysis, Shukang Wang and Yibin Ren; Funding acquisition, Yong Han; Investigation, Shukang Wang; Methodology, Shukang Wang and Yibin Ren; Project administration, Yong Han; Resources, Yong Han; Software, Shukang Wang; Supervision, Peng Gao and Ge Chen; Validation, Shukang Wang; Visualization, Shukang Wang; Writing—original draft, Shukang Wang; Writing—review & editing, Yibin Ren.

References

1. Wei, Y.; Chen, M.-C. Forecasting the short-term metro passenger flow with empirical mode decomposition and neural networks. *Transp. Res. Part C Emerg. Technol.* **2012**, *21*, 148–162. [CrossRef]

2.   Sun, Y.; Leng, B.; Guan, W. A novel wavelet-SVM short-time passenger flow prediction in Beijing subway system. *Neurocomputing* **2015**, *166*, 109–121. [CrossRef]

3.   Chen, Q.; Li, W.; Zhao, J. The use of LS-SVM for short-term passenger flow prediction. *Transport* **2011**, *26*, 5–10. [CrossRef]

4.   Wang, J.; Cheng, T.; Li, X. Nonlinear integration of spatial and temporal forecasting by support vector machines. In Proceedings of the Fuzzy Systems and Knowledge Discovery (FSKD 2007), Haikou, China, 24–27 August 2007; pp. 61–66.

5.   Lin, E.; Park, J.D.; Züfle, A. Real-Time Bayesian Micro-Analysis for Metro Traffic Prediction. In Proceedings of the 3rd ACM SIGSPATIAL Workshop on Smart Cities and Urban Analytics, Redondo Beach, CA, USA, 7–10 November 2017; p. 12.

6.   Truong, R.; Gkountouna, O.; Pfoser, D.; Züfle, A. Towards a Better Understanding of Public Transportation Traffic: A Case Study of the Washington, DC Metro. *Urban. Sci.* **2018**, *2*, 65. [CrossRef]

7.   Dai, X.; Sun, L.; Xu, Y. Short-Term Origin-Destination Based Metro Flow Prediction with Probabilistic Model Selection Approach. *J. Adv. Transp.* **2018**. [CrossRef]

8.   Gong, Y.; Li, Z.; Zhang, J.; Liu, W.; Zheng, Y.; Kirsch, C. Network-wide Crowd Flow Prediction of Sydney Trains via customized Online Non-negative Matrix Factorization. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management, Torino, Italy, 22–26 October 2018; pp. 1243–1252.

9.   Tsai, T.-H.; Lee, C.-K.; Wei, C.-H. Neural network based temporal feature models for short-term railway passenger demand forecasting. *Expert Syst. Appl.* **2009**, *36*, 3728–3736. [CrossRef]

10.   Cheng, T.; Wang, J. Integrated spatio-temporal data mining for forest fire prediction. *Trans. Gis* **2008**, *12*, 591–611. [CrossRef]

11.   Wang, J.; Tsapakis, I.; Zhong, C. A space–time delay neural network model for travel time prediction. *Eng. Appl. Artif. Intell.* **2016**, *52*, 145–160. [CrossRef]

12.   Biship, C.M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*; Springer: Berlin/Heidelberg, Germany, 2007.

13.   LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef]

14.   Chen, J.; Pei, T.; Shaw, S.-L.; Lu, F.; Li, M.; Cheng, S.; Liu, X.; Zhang, H. Fine-grained prediction of urban population using mobile phone location data. *Int. J. Geogr. Inf. Sci.* **2018**, *32*, 1770–1786. [CrossRef]

15.   Zhang, J.; Zheng, Y.; Qi, D.; Li, R.; Yi, X.; Li, T. Predicting citywide crowd flows using deep spatio-temporal residual networks. *Artif. Intell.* **2018**, *259*, 147–166. [CrossRef]

16.   Goodfellow, I.; Bengio, Y.; Courville, A.; Bengio, Y. *Deep Learning*; MIT press Cambridge: Cambridge, MA, USA, 2016; Volume 1.

17.   Vinyals, O.; Toshev, A.; Bengio, S.; Erhan, D. Show and tell: A neural image caption generator. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3156–3164.

18.   Yu, H.; Wang, J.; Huang, Z.; Yang, Y.; Xu, W. Video paragraph captioning using hierarchical recurrent neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 27–30 June 2016; pp. 4584–4593.

19.   Srivastava, N.; Mansimov, E.; Salakhudinov, R. Unsupervised learning of video representations using lstms. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 843–852.

20.   Deng, L.; Yu, D. Deep learning: Methods and applications. *Found. Trends®Signal. Process.* **2014**, *7*, 197–387. [CrossRef]

21.   Collobert, R.; Weston, J. A unified architecture for natural language processing: Deep neural networks with multitask learning. In Proceedings of the 25th International Conference on Machine Learning, Helsinki, Finland, 5–9 July 2008; pp. 160–167.

22.   Lee, D.; Kim, K. Recurrent Neural Network-Based Hourly Prediction of Photovoltaic Power Output Using Meteorological Information. *Energies* **2019**, *12*, 215. [CrossRef]

23.   Zhang, T.; Song, S.; Li, S.; Ma, L.; Pan, S.; Han, L. Research on Gas Concentration Prediction Models Based on LSTM Multidimensional Time Series. *Energies* **2019**, *12*, 161. [CrossRef]

24.   Liu, Y.; Liu, Z.; Jia, R. DeepPF: A deep learning based architecture for metro passenger flow prediction. *Transp. Res. Part C Emerg. Technol.* **2019**, *101*, 18–34. [CrossRef]

25. Ma, X.; Tao, Z.; Wang, Y.; Yu, H.; Wang, Y. Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. *Transp. Res. Part C Emerg. Technol.* **2015**, *54*, 187–197. [CrossRef]

26. Wang, Y.; Currim, F.; Ram, S. *Deep Learning for Bus Passenger Demand Prediction Using Big Data*; Social Science Electronic Publishing: New York, NY, USA, 2017.

27. Ren, Y.; Cheng, T.; Zhang, Y. Deep spatio-temporal residual neural networks for road-network-based data modeling. *Int. J. Geogr. Inf. Sci.* **2019**, 1–19. [CrossRef]

28. Zhang, J.; Zheng, Y.; Qi, D. Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction. In Proceedings of the Thirty-First AAAI Conference on Artificial IntelligenceI, San Francisco, CA, USA, 4–9 February 2017; pp. 1655–1661.

29. Ma, X.; Dai, Z.; He, Z.; Ma, J.; Wang, Y.; Wang, Y. Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction. *Sensors* **2017**, *17*, 818. [CrossRef]

30. Defferrard, M.; Bresson, X.; Vandergheynst, P. In Convolutional neural networks on graphs with fast localized spectral filtering. In Proceedings of the Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016; pp. 3844–3852.

31. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907.

32. Bruna, J.; Zaremba, W.; Szlam, A.; LeCun, Y. Spectral networks and locally connected networks on graphs. *arXiv* **2013**, arXiv:1312.6203.

33. Lin, L.; He, Z.; Peeta, S. Predicting station-level hourly demand in a large-scale bike-sharing network: A graph convolutional neural network approach. *Transp. Res. Part C Emerg. Technol.* **2018**, *97*, 258–276. [CrossRef]

34. Bing, Y.; Yin, H.; Zhu, Z. Spatio-temporal Graph Convolutional Neural Network: A Deep Learning Framework for Traffic Forecasting. *arXiv* **2017**, arXiv:1709.04875.

35. Ma, Z.; Xing, J.; Mesbah, M.; Ferreira, L. Predicting short-term bus passenger demand using a pattern hybrid approach. *Transp. Res. Part C Emerg. Technol.* **2014**, *39*, 148–163. [CrossRef]

36. Mathieu, M.; Couprie, C.; LeCun, Y. Deep multi-scale video prediction beyond mean square error. *arXiv* **2015**, arXiv:1511.05440.

37. Jain, V.; Murray, J.F.; Roth, F.; Turaga, S.; Zhigulin, V.; Briggman, K.L.; Helmstaedter, M.N.; Denk, W.; Seung, H.S. Supervised learning of image restoration with convolutional networks. In Proceedings of the IEEE 11th International Conference on Computer Vision, Rio de Janeiro, Brazil, 14–21 October 2007; pp. 1–8.

38. Kingma, D.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.

39. Shanghai Metro Roadmap. Available online: http://sh.bendibao.com/ditie/linemap.shtml (accessed on 27 May 2019).

40. Chollet, F. Keras: Deep Learning Library for Theano and Tensorflow. 2015. Available online: https://keras.io/ (accessed on 16 January 2019).

41. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv* **2016**, arXiv:1603.04467.

42. Komer, B.; Bergstra, J.; Eliasmith, C. Hyperopt-sklearn: Automatic hyperparameter configuration for scikit-learn. In Proceedings of the 13th Python in Science Conference (SciPy 2014), Austin, TX, USA, 6–12 July 2014.

43. Ke, J.; Zheng, H.; Yang, H.; Chen, X.M. Short-term forecasting of passenger demand under on-demand ride services: A spatio-temporal deep learning approach. *Transp. Res. Part C Emerg. Technol.* **2017**, *85*, 591–608. [CrossRef]

44. These stations of the Shanghai Metro, I Was Crying and Being Squeezed Down. Available online: http://news.fdc.com.cn/mrrd/977358.shtml (accessed on 16 January 2019).