A Graph Convolutional Stacked Bidirectional Unidirectional-LSTM Neural Network for Metro Ridership Prediction

Pengfei Chen, Xuandi Fu, and Xue Wang

***Abstract*— Timely precise metro ridership forecasting is helpful to reveal real-time traffic demand, which is a crucial but challenging task in modern traffic management. Given the complex spatial correlation and temporal variation of riding behaviour in a metro system, deep learning algorithms have been widely applied owing to their superior performance in capturing spatio-temporal features. However, current deep learning models utilize regular convolutional operations, which can barely provide satisfactory accuracy due to either the ignorance of realistic topology of a traffic network or insufficiency in capturing representative spatiotemporal patterns. To further improve the accuracy in metro ridership prediction, this study proposes a parallel-structured deep learning model that consists of a Graph Convolution Network and a stacked Bidirectional unidirectional Long short-term Memory network (GCN-SBULSTM). The GCN module regards a metro network as a structured graph, and a K-hop matrix, which integrates the travel distance, population flow, and adjacency, is introduced to capture the dynamic spatial correlation among metro stations. The SBULSTM module considers both backward and forward states of ridership time series and can learn complex temporal features with stacked recurrent layers. Experiments are conducted on three real-life metro ridership datasets to demonstrate the effectiveness of the proposed model. Compared with state-of-the-art prediction models, GCN-SBULSTM presents better performance in multiple scenarios and largely enhances the efficiencies of training processes.**

***Index Terms*— Deep learning model, traffic prediction, spatio-temporal dependency, origin-destination flow, parallel structure.**

I. Introduction

MULTI-SCALE precise traffic forecasting is one of the most fundamental and crucial tasks for urban transportation control and management, where metro ridership prediction has attracted increasing concerns from both the academic community and authorized departments because of the vital position of the subway in urban public transportation system [1], [2]. It is a challenging task to make collaborative spatial-temporal predictions for metro ridership due to the complicated spatial structure of traffic networks, temporal variations, and uncertainty inherited from human behaviour. Recently, owing to the rapid development of artificial intelligence, computation power and abundant traffic data supported by novel collection and storage techniques, the booming deep learning approaches have flushed current prediction-related research and promoted significant progress in traffic forecasting [3]–[5].

Deep learning methods have been reported to outperform traditional statistical models in many applications, especially in time series forecasting [6]. Typical statistical models, such as auto-regressive integrated moving average (ARIMA) [7] and its variants [8], [9], are commonly adopted for single time series prediction, while they ignore the potential dependency among multiple time series under relatively complex traffic conditions. In contrast, deep learning approaches employ multiple processing layers and allow the models to learn abstracted features and non-linear dependencies from large-scale traffic datasets, which makes deep learning methods as a major solution in current traffic forecasting.

Given the well-acknowledged performance in time series forecasting, Recurrent Neural Networks (RNN) and its variants, such as Long Short-term Memory (LSTM) and Gated Recurrent Unit (GRU), are widely employed in the mainstream studies for traffic forecasting [10]–[12]. However, RNN-based models employ only the temporal features in travel behaviour, while ignoring the underlying spatial dependencies within a traffic network [13]. To capture the spatial dependencies in traffic data, a batch of studies utilizes Convolutional neural network (CNN) to build prediction models, in which a traffic network is commonly transformed into an image based on its geographic locations [4], [14], [15]. However, CNN-based models only consider the absolute distance relationship among stations in 2D Euclidean space, while the non-Euclidean structural features in traffic networks, such as the connectivity, is not fully learned. Also, due to the predefined image size, CNN-based models are prone to generating distorted spatial relationships, which limits their adaption to the varying structure of traffic networks in the real world [16].

Compared with CNN, Graph Convolutional Network (GCN) provides a more feasible way to model spatial dependencies within a traffic network. Given the inherent graph structure of a traffic network, GCN is naturally capable of preserving realistic topology and capture the dependencies between

Manuscript received May 20, 2020; revised November 3, 2020 and January 14, 2021; accepted February 22, 2021. The Associate Editor for this article was A. Y. Lam. *(Pengfei Chen and Xuandi Fu contributed equally to this work.) (Corresponding author: Pengfei Chen.)*

Pengfei Chen and Xue Wang are with the School of Geospatial Engineering and Science, Sun Yat-sen University, Guangzhou 510275, China, and also with the Southern Marine Science and Engineering Guangdong Laboratory (Zhuhai), Zhuhai 519082, China (e-mail: chenpf9@mail.sysu.edu.cn; wangxue25@mail.sysu.edu.cn).

Xuandi Fu is with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213 USA (e-mail: xuandif@andrew.cmu.edu).

Digital Object Identifier 10.1109/TITS.2021.3065404

metro stations by aggregating nodes' information through graph convolution [17]. However, the effective construction of graphs and the integration of GCN with existing neural networks remain as two open problems in current studies. For the first issue, the most relative works directly adapt physical topologies within a network, such as the adjacency, to build graphs [18]–[20]. Nevertheless, given the latent spatio-temporal dependencies implied in traffic data, such as the travel distance and population flow between traffic sites, some virtual graphs can be build up based on prior knowledge to improve the effectiveness of GCN [21], [22]. As for the integration of GCN, many relative studies combine GCN and RNN models to build a joint prediction model, so as to capture both spatial and temporal features for the forecasting problems. For example, Cui *et al.* [23] designed an architecture that uses the output of multiple GCNs as the input of LSTM for traffic speed prediction. Jin et at. [20] fused the output of GCN and variational auto-encoder model and fed the result into a Seq2seq GRU module to predict urban ride-hailing demand. However, the extracted features based on such sequential structure can be distorted when converting the convolution results, which might lead to information loss and uncertain predictions [5].

Based on the aforementioned problems, we propose a parallel GCN and Stacked Bidirectional Unidirectional LSTM model (GCN-SBULSTM) for metro ridership forecasting. In GCN-SBULSTM, both physical topology and virtual graphs, including adjacency, travel distance and population flow among metro stations, are used to construct the GCN module, and a K-hop weight matrix is introduced to adaptively determine the extent of neighbor information to be considered in each graph. The SBULSTM module is used to handle temporal dependencies, which is capable of capturing long-term dependencies by considering both the backward and forward correlations in ridership time series. This architecture is expected to inherit the merits from both GCN in extracting realistic spatial dependencies and SBULSTM in capturing temporal features, while reducing their interference using a parallel instead of a sequential structure.

In summary, the main contributions of this study include:

- Propose a new deep learning architecture composed of two parallel modules considering both spatial and temporal dependencies for metro ridership prediction;
- Design a novel K-hop weight matrix, which integrates adjacency, travel distance, and population flow among metro stations, to represent metro networks, and incorporate the matrix into the GCN module to enhance the extraction of realistic spatial dependency;
- Integrate stacked bidirectional recurrent layers into the model, which improves its ability in capturing long-term context and generating a higher level of representation of sequence data.

II. Related Work

A. Machine Learning for Traffic Forecasting

In early machine learning problems for traffic forecasting, state data form different traffic sites are always organized in terms of their collection timestamps as a batch of time series, and RNN-based models, such as GRU and LSTM, are widely used given their ability in remembering important information about the sequential input with its internal memory. For instance, Yu *et al.* [10] combined deep LSTM and stacked autoencoder to capture both the temporal and static features in traffic data for traffic flow forecasting, and experimental results on real-world data showed that their model can significantly improve the predictive performance especially under extreme conditions, such as peak-hour and post-accident scenarios. Considering the periodicity of metro riding behaviour given the regularity in human's daily activities, Cui *et al.* [11] designed a stacked bidirectional and unidirectional LSTM framework, which concerned both forward and backward dependencies of traffic data, for traffic speed prediction over the whole urban traffic networks. Those studies have demonstrated the superiority of RNN in extracting temporal features for traffic forecasting. However, it is challenging to use solely RNN-based models to maintain the spatial features and topological information in traffic data, which limits their effectiveness in practical applications [13].

Noticing the promising achievement of CNN in computer vision [24], [25], many studies generalize CNN to learn the spatial dependencies in Euclidean space. For example, Zhang *et al.* [4] proposed a deep neural network to predict citywide crowd flow, in which multiple CNN layers were applied on traffic demand heatmaps to extract spatial features. This model was further developed in [15] by being integrated with residual learning to capture large-scale spatial dependencies. By sequentially connecting CNN and LSTM networks, Yu *et al.* [26] proposed a spatio-temporal recurrent convolution network (SRCNs) for traffic speed forecasting. Yao *et al.* [27] developed a Deep Multi-View Spatial-Temporal Network (DMVST-Net) for taxi demand prediction, which jointly concerned the spatial, temporal, and semantic relations using LSTM, CNN and graph embedding, respectively. To reduce the interference of sequentially connected LSTM and CNN modules, Ma *et al.* [5] proposed a parallel CNN-BLSTM framework for metro ridership prediction. The results also proved that the parallel structure could significantly improve prediction accuracy. However, these models inherit the drawbacks of CNN that ignores topology information within a traffic network, which inevitably hampers their performance in given the increasing complexity of traffic patterns [16].

B. Graph Convolution Networks

For the last few years, the emergence of GCN has refreshed the way of modelling traffic data. By treating a traffic network as a graph instead of the predefined image in CNN, GCN can largely preserve the realistic topological information and thus benefiting the extraction of comprehensive spatial features [17], [19]. Also, GCN can greatly preserve the globality of metro networks through conducting convolution on the whole structured graphs, which theoretically outperforms CNN that can only capture neighbouring spatial pattern due to limited kernel window size.

By combining with temporal dynamics, GCN-based models have made significant progress in traffic forecasting problems. For instance, Li *et al.* [18] proposed a diffusion convolutional recurrent neural network (DCRNN), in which the traffic flow was modelled as a diffusion process on a directed graph and spatial dependency was captured using bidirectional random walks on the graph. Wu *et al.* [28] developed a novel architecture named Graph-WaveNet, which adopted an adaptive adjacency matrix through embedding technique to capture hidden spatial dependencies. Yu *et al.* [29] combined graph convolution and gated temporal convolution to capture precise spatio-temporal correlations for traffic speed forecasting, which also enhanced training efficiency with a reduced number of parameters. Lu *et al.* [21] used a dynamic weighted graph to modelled the road relationship and developed an adaptive graph gate convolution network for traffic flow prediction. Rather than the single physical topology in traffic networks, some domain knowledges also included in recent studies to guide graphs construction. For example, Du *et al.* [30] extracted virtual stations using a density-peak based clustering method and developed a dynamic convolution neural network to predict traffic demands. Liu *et al.* [22] established a Physical-Virtual Collaboration Graph Network (PVCGN), which incorporates the connection among metro stations, ridership similarity, and inter-station passenger flow into a Graph Convolution Gated Recurrent Unit for spatio-temporal dependency learning. However, due to the large number of parameters in PVGCN, its efficiency is relatively low compared to other models.

III. Methodology

In this section, we formulize the learning problem of metro ridership forecasting and elaborate on the motivation and detailed steps for the construction and combination of the GCN and SBULSTM module.

A. Metro Ridership Forecasting Problem

Metro ridership forecasting is a fundamental spatio-temporal prediction problem given the spatial correlation and periodicity of people's daily riding behaviour. Ridership data of each metro station are commonly summarized using a specific time interval and thus forming a batch of time series for further operation. In this study, our goal of is to predict the ridership in next m time intervals given the historical data in previous n time intervals. Based on the observations from s metro stations, the input data for our model can be expressed as a matrix X:

$$\begin{aligned} X &= [X_{T-n}, X_{T-n+1}, \cdots, X_{T-1}] \\ &= \begin{bmatrix} x^1_{T-n} & x^1_{T-n+1} & \cdots & x^1_{T-1} \\ x^2_{T-n} & x^2_{T-n+1} & \cdots & x^2_{T-1} \\ \cdots & & \cdots & \cdots \\ x^s_{T-n} & x^s_{T-n+1} & \cdots & x^s_{T-1} \end{bmatrix} \end{aligned} \tag{1}$$

where X_{T-i} encodes the ridership vector measured at the i_{th} time intervals before timestamp T, and x^j corresponds to the ridership data of j_{th} station.

In addition to the raw ridership data, the metro network can be represented by an undirected graph, $\mathcal{G} = (V, E)$ where V, E denotes the set of stations and lines in the network, respectively. V encodes the features of nodes, which in this task refers to the ridership time series of each station. E encloses all edges linking two nodes, of which the values can be different based on raw data. For example, in a modern metro system, the riding behaviour is always recorded using smart card transaction logs, thereby some personal information, such as the card ID and travel path, can be used as valuable supporting information for E. For simple usage, we use I to represent these additional data. Therefore, the forecasting problem can be formulated as learning a function f:

$$[\mathrm{X}^{T-n}, \cdots, \mathrm{X}^{T-1}; \mathcal{G}; I] \xrightarrow{f} [\mathrm{X}^{T}, \cdots, \mathrm{X}^{T+m-1}] \tag{2}$$

The resultant prediction is denoted by $\hat{X} = [X_T, \cdots, X_{T+m-1}]$ in the rest of this article, where each element is a vector of the s stations' ridership at a future time step.

B. Foundation of the Graph Convolution Network Module

Currently, several strategies have been investigated to build effective graphs for traffic forecasting. Commonly employed are adjacency matrix [31] and Laplacian matrix [32], [33]. GCN based on Laplacian matrix incorporates the spectral theory to graph convolution, which is often named as spectral graph convolution. The classic GCN encodes adjacency relationship among nodes to represent arbitrarily structured graphs. It normally utilizes a binary-encoded adjacency matrix A to denote the connectivity among nodes. If node i and j are directly connected in the metro network, $A_{ij} = 1$, otherwise $A_{ij} = 0$.

However, within collaborated spatial-temporal prediction, spatial dependency should be considered dynamically, as it could vary in different scenarios. For example, ridership of distant stations may exhibit low correlation within a short counting period, e.g., 10-minute interval, while the correlation could significantly increase with the length of a target prediction interval due to the city-scale globality of passenger riding behaviour. Therefore, simply applying a binary adjacency matrix or predefined stationary distance is not sufficient to handle complex scenarios.

To tackle aforementioned problems, we initialize the GCN module by intuitively taking metro stations as nodes, and define three graphs, including travel distance graph, population flow graph, adjacency graph, to weight the edges.

- **Adjacency graph**: The connection between metro stations is widely acknowledged to affect the relationship of their ridership [18], [29]. However, traditional adjacency matrix mostly focuses on the directly connected stations, i.e. the 1-order neighboured stations, while the indirect connection among stations has been ignored. Therefore, as shown in the first line of 1, a k-hop adjacency matrix A^k is adopted in this study to encode the direct and indirect adjacency relationship among metro stations. Given a constant K, each element A^k_{ij} should be 1 if station i

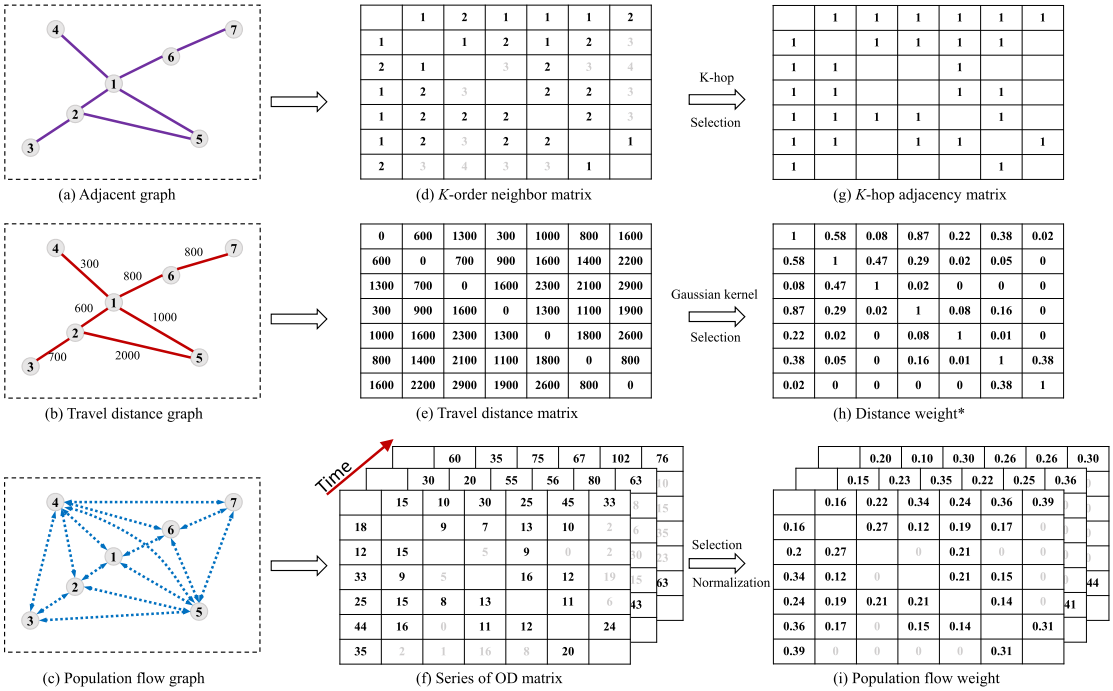

Fig. 1. Illustration of graphs consisting the K-hop weight matrix used in the GCN module (K = 2). *Extremely small value (i.e. smaller than 0.005) is shown as zero.

and j are K-order neighboured; otherwise, the element is set to 0. mathematically:

$$A_{ij}^{k} = \begin{cases} 1, & \partial \leq k \\ 0, & otherwise \end{cases} \tag{3}$$

where ∂ denotes the least number of steps from station i to j.

- **Travel distance graph**: According to the First Law of Geography and results from previous studies, traffic behaviours occurred closely are likely to be related [34]–[36]. For instance, the ridership pattern of neighbouring stations along metro lines can be highly correlated as passengers within a region may have similar daily travel pattern. Therefore, from the view of the geography, we take the travel distance as an important factor during the initialization of GCN module. An example is illustrated in the second line of Figure 1. Specifically, following the definition in [18], we calculate the distance weight matrix D using a Gaussian Kernel [37], where the element D_{ij} is calculated as:

$$D_{ij} = \exp(-\frac{\text{dist}(v_i, v_j)^2}{\sigma^2}) \tag{4}$$

where $\text{dist}(v_i, v_j)$ indicates the shortest travel distance along the metro network between station v_i and v_j, σ is the standard deviation of travel distances.

- **Population flow graph**: Population flow is a virtual connection between metro stations, which reflects their latent dependencies implied by the regularity of passengers' daily activity. A large population flow should indicate a relatively high dependency between metro stations [38], [39]. However, as the population flow temporally varies, a series of population flow matrixes are generated to dynamically represent the dependency. As shown in the third line of 1, we extract the origin-destination (OD) flows between metro station and generate the population flow matrix F through normalization. Each element F_{ij} of matrix F is calculated as:

$$F_{ij} = \frac{1}{2}(\frac{N_{ji}}{\sum_{k}^{n} N_{jk}} + \frac{N_{ij}}{\sum_{k}^{n} N_{ik}}) \tag{5}$$

where N_{ij} is the number of passengers travelling from station i to station j during a specific timespan. In this work, this timespan is defined as the period from the earliest historic frame to the last one. In addition, considering the large number of stations in a common metro system, we set $N_{ij} = 0$ if $A_{ij}^{k} = 0$ to reduce the interference from distant stations and enlarge the weights of nearby stations with a large population exchange.

Finally, we define a k-hop weight matrix, M^k, which integrates the graphs of K-hop adjacency, travel distance, and population flow for dynamically capturing spatial dependency among stations. Mathematically:

$$M^k = F \odot D \odot A^k \tag{6}$$

where $\odot$ stands for element-wise multiplication, threshold k should be treated as a hyperparameter in the model, thereby ensuring the most significant spatial correlation among stations can be learned.

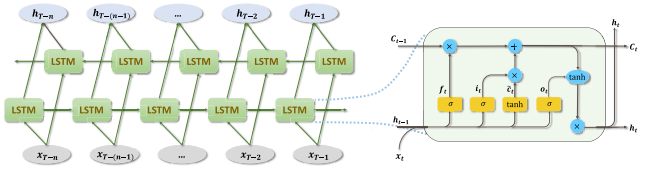

Fig. 2. Architecture of a Bidirectional Long-short Term Memory network and an LSTM Memory Cell.

Based on the proposed K-hop matrix, the graph convolution can be defined as:

$$h_g^{l+1} = g(h_g^l, M^k) \tag{7}$$

$$g(h_g^l, M^k) = \mathrm{ReLU}(M^k h_g^l W_g^l) \tag{8}$$

$$\mathrm{ReLU}(x) = max(0, x) \tag{9}$$

where h_g^l, h_g^{l+1} are the input graph generated by the former layer l and the output graph at layer $l+1$, respectively. W_g is the trainable weight matrix for generating output features at each layer. A non-linear activation function (ReLU) is employed after each convolutional layer before the features are forwarded to the next layer.

C. Foundation of the Stacked Bidirectional Unidirectional LSTM Module

From the temporal perspective, metro ridership variations possess several special characteristics, including non-linearity, periodicity and regularity [40]. Considering those features, the Stacked Bidirectional Unidirectional LSTM (SBULSTM) framework is adopted to learn the complex temporal pattern from the historical ridership inputs and to make sequential predictions [11]. The theoretical foundation and detailed steps for SBULSTM are elaborated in the following content.

1) Long Short-Term Memory: LSTM architecture is the basic unit in SBULSTM for capturing temporal feature of metro ridership data. It has been widely acknowledged that LSTM outperforms other recurrent architectures for handling sequence-based tasks with long-term dependencies. Its sophisticated gated memory mechanism has helped to avoid gradient vanishing or exploding problems exhibiting in traditional RNN [41]. As demonstrated in Figure 2, each LSTM cell contains three gates, including the input gate i_t, forget gate f_t, and output gate o_t. The input gate determines the information to be preserved, forget gate controls the partition to be abandoned, and output gate decides the result to be generated [42]. Detailed procedures for calculating three gates and cell memory in each memory unit is represented as follows:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{10}$$

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{11}$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \tag{12}$$

$$\widetilde{c}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{13}$$

where W_i, W_f, W_o are the weighted matrices and b_i, b_f, b_o and b_c are bias vectors of LSTM to be learned during training. σ is the gate activation function, which normally indicates the sigmoid function. Based on those three gates, the cell output state c_t and the hidden layer output h_t of current cell can be generated as follows:

$$c_t = f_t \odot c_{t-1} + i_t \odot \widetilde{c}_t, \tag{14}$$

$$h_t = o_t \odot \tanh(c_t) \tag{15}$$

where $\odot$ stands for element-wise multiplication, and tanh is the hyperbolic tangent function. Here, when taking the ridership prediction problem as an example, only the last element of the output vector

2) Bidirectional Long Short-Term Memory: Bidirectional LSTM network is utilized for capturing the periodicity and regularity of metro ridership. It is noted that LSTM structure can only make use of forward dependencies and inevitably filter out useful information due to the long-term gated memory chain. The bidirectional LSTM structure can help solve the problem through concatenating forward and backward LSTM layers [43]. It can employ hidden states from both directions, complementing for the information loss along the chain within LSTM. Therefore, bidirectional LSTM has a better capability for capturing long-term contextual dependencies in sequential prediction tasks and making more precise sequential predictions [44], [45]. Apart from that, the periodicity of metro ridership pattern is another consideration for including backward temporal dependency in the model. Unlike traffic incident, wind speed or other randomly organized features, metro traffic possesses strong periodicity and regularity. Utilizing bidirectional information can enhance the ability in modelling periodic pattern of metro ridership and making comprehensive predictions.

The bidirectional LSTM network contains two parallel LSTM layers in both propagation directions, as shown in Figure 2.

$$\overrightarrow{h}_t = \mathrm{LSTM}_{fw}(x_t, \overrightarrow{h}_{t-1}) \tag{16}$$

$$\overleftarrow{h}_t = \mathrm{LSTM}_{bw}(x_t, \overleftarrow{h}_{t+1}) \tag{17}$$

LSTM_{fw} and LSTM_{bw} denote the forward and backward LSTM, respectively. $\overrightarrow{h}_t$ and $\overleftarrow{h}_t$ are the hidden states of the input temporal feature x_t learned from bidirectional LSTM. The bidirectional hidden state h_t for each input x_t is obtained through concatenating the generated forward and backward hidden states:

$$u_t = [\overrightarrow{h}_t, \overleftarrow{h}_t] \tag{18}$$

3) Stacked Bidirectional Unidirectional LSTM: Deep recurrent networks have demonstrated its ability to generate a higher level of representation from sequential input in previous studies [46]–[48]. The prediction power of a neural network can be enhanced through deepening model structure, of which the effectiveness has been proved in many domains, such as speech processing [46], [47], text recognition [48] and so on. Therefore, to break through the limited performance of single LSTM or BLSTM architecture, this study adopts SBULSTM proposed in [11] to learn the temporal dependencies in ridership data. In SBULSTM, The output of BLSTM network

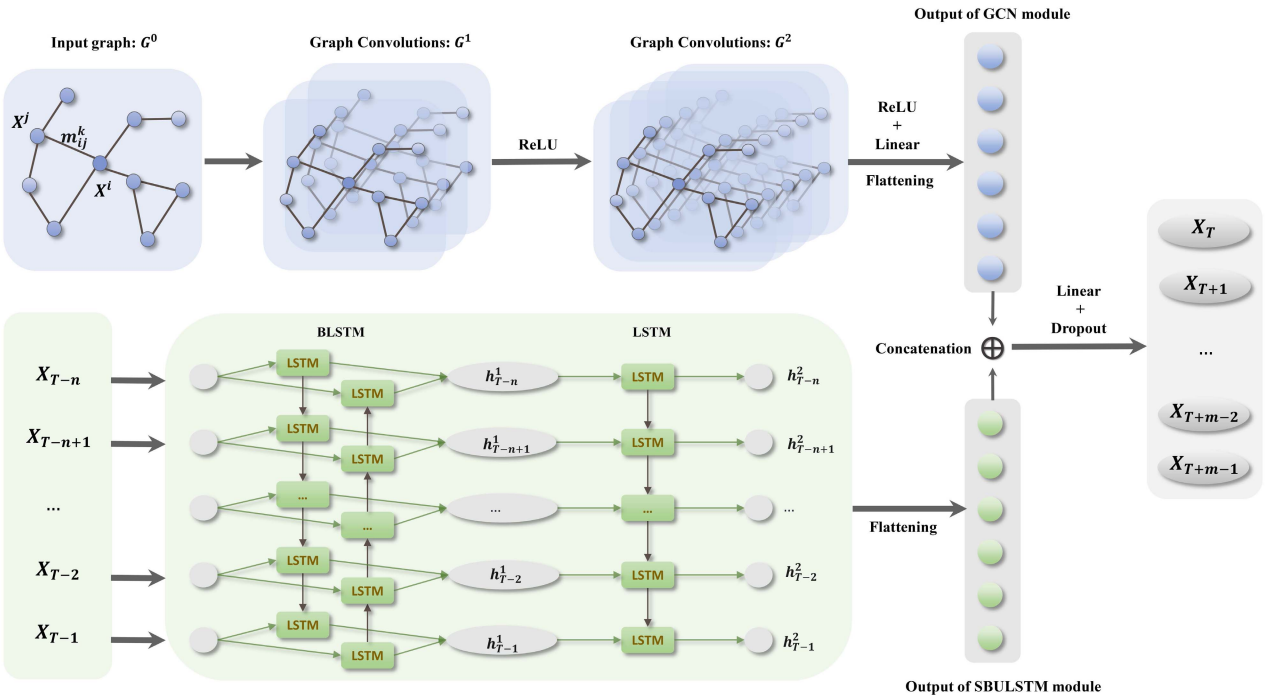

Fig. 3. Architecture of the proposed graph convolutional stacked bidirectional-LSTM neural network (GCN-SBULSTM).

is further fed to LSTM layer to generate higher sequential representations. Theoretically, SBULSTM inherits the merits from both LSTM and BLSTM, which on the one hand can capture both forward and backward temporal dependency, and on the other hand, allow a higher level of representation of the ridership data. Nevertheless, it has not been incorporated with spatial learning module previously, which limits its capability for making a comprehensive spatial-temporal prediction.

D. Spatial-Temporal Prediction With GCN-SBULSTM

Previous studies commonly combined spatial and temporal modules sequentially. For instance, the generated output from CNN is fed into LSTM network in [26], [27]. However, the original flow pattern may be distorted passing through complex spatial operations, i.e. deep convolutions, as the generated output from convolutional layers cannot fully represent the pattern of raw metro ridership data [5].

Therefore, to preserve the effectiveness of spatial and temporal modules as much as possible and integrate their results for a mutual complement, this study establishes a new deep learning architecture, in which a GCN and SBULSTM module are parallelly combined to make predictions for future time frames. The effectiveness of such a parallel structure in ridership prediction has been proved in previous study [5], and this study is an extension by using a dynamic graph learning approach instead of CNN. As shown 3, ridership data are first organized into two forms, including dynamic graphs and time series; then these two forms of data are respectively fed to the GCN and SBULSTM modules to learn spatial and temporal dependencies, the outputs can be represented by $H_G = [h_{g_1}, \cdots, h_{g_k}]$ and $H_T = [h_{t_1}, \cdots, h_{t_p}]$, where k and p is the number of hidden units in the last layer of GCN and SBULSTM module, respectively; finally, the flattened outputs of two modules, $O_G = \text{Flatten}(H_G)$ and $O_T = \text{Flatten}(H_T)$, are concatenated, and a fully connected layer with dropout mechanism are applied to obtain the prediction results, which can be formulized as follows:

$$\hat{X} = W_{st}(O_G \parallel O_T) + b_{st} \tag{19}$$

where W_{st} and b_{st} are the trainable weight and bias parameters for generating final predicted results $\hat{X}$. $\parallel$ is the concatenating operator.

IV. Experiments

A. Data Description

Three metro ridership datasets are used to validate the effectiveness of the proposed GCN-SBULSTM: 1) a real-world ridership dataset named SZMetro, which was collected from the metro system in Shenzhen, China; 2) two public ridership datasets shared in [22], respectively named HZMetro and SHMetro, which are used for benchmark tests. The details of these three datasets are summarized in Table I.

SZMetro: This dataset was collected during Jan. 17 2017 to Feb. 22 2017 based on the transaction records provided by the metro system in Shenzhen, China. At the collection time, there were 8 running metro lines with a total of 166 metro stations for Shenzhen metro system, as shown in Figure 4. Each record contains passengers' inbound information, including the transaction time and name of stations. Station-level ridership is summarized based on a 4-minute time interval. As the service time of Shenzhen metro is from 6 AM to the midnight, 270 ridership samples per day are obtained. It is noticeable that

TABLE I
DATASETS SUMMARY

Dataset	SZMetro	HZMetro	SHMetro
City	Shenzhen	Hangzhou	Shanghai
# Station	166	80	288
Time interval	4 min	15 min	15 min
# Samples per day	270	70	70
Train Timespan	17/01/2017 - 28/01/2017; 10/02/2017 - 17/02/2017	1/01/2019 - 1/18/2019	7/01/2016 - 8/31/2016
Test Timespan	18/02/2017 - 22/02/2017	1/21/2019 - 1/25/2019	9/10/2016 - 9/30/2016

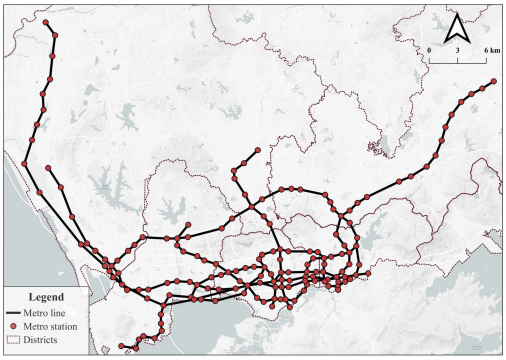

Fig. 4. Shenzhen metro network.

the timespan of training data in SZMetro is discontinuous, this is because the period from Jan. 28 to Feb. 9 corresponds to the Spring Festival holiday in China, leading a significant decline in ridership and very different dynamic patterns compared to other dates. To avoid the influence of these special dates and retain the universality of trained models, data from Jan 28 to Feb 9 are discarded in the experiments on SZMetro. Consequently, ridership data from the first 20 days are used for training, and the data from the last 5 days are used for testing.

HZMetro and SHMetro: These two datasets were built up based on the metro system in Hangzhou and Shanghai, respectively. They were both summarized with a 15-minute time interval, generating 70 samples per day. It is notable that, since no station information is provided for HZMetro and SHMetro, we cannot transform the metro network to an image, so that all state-of-the-art models containing CNN module will not be tested on these two datasets. More information about HZMetro and SHMetro can be found in [22].

B. Experiment Design

Experiments include two main parts:

1) Test the performance of GCN-SBULSTM with respect to different temporal scales (i.e., different input and output length) and validate the effectiveness of different graphs used in our model. This part of experiments is conducted on SZMetro because its raw data are available, so that we can easily reorganize the raw data for different prediction tasks.

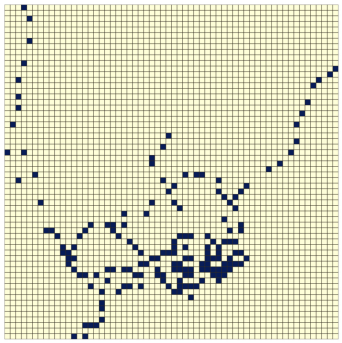

Fig. 5. The input image of CNN.

Specifically, two tasks are designed on SZMetro to test the performance of GCN-SBULSTM:

- **Task 1 (5 to 5)**, to forecast the next 5 samples based on previous 5 samples,
- **Task 2 (10 to 10)**, to predict the next 10 samples using previous 10 samples.

2) Run benchmark tests on open datasets HZMetro and SHMetro using its original input and output length (i.e., 4) to further verify the superiority of GCN-SBULSTM.

To demonstrate the advantages of GCN-SBULSTM, classic deep learning architecture, including LSTM, CNN, GCN, and advanced models, including DMVST-Net [27], CNN-LSTM [5], SRCNs [26], SBULSTM [11], DCRNN [18], STGCN [29], Graph WaveNet [28] and PVCGN [22] are implemented for comparison. In addition, ablation test is performed to analyze the effectiveness of K-hop matrix used in GCN-SBULSTM. Specifically, suffixes are used to distinguish different ablated models: "w/o dist" indicates the model without using distance graph, "w/o OD" denotes the model without using population flow graph; "w/o K-hop" stands for the model using only the traditional adjacency matrix in the GCN module.

C. Computational Environment and Experimental Setup

All experiments are compiled and tested on a desktop equipped with an Intel(R) Core(TM) CPU i9-10940X and an NVIDIA GTX 2070i running Windows 10. The parameters for each prediction model are carefully tuned to obtain the best accuracy on test dataset: most models are tested following the setting in their original article, while minor adjustments are made on tunable parameters, such as the number of hidden units and batch size, to enhance the accuracy as much as possible.

To generate the traffic image for experiments on SZMetro, the metro network map is divided by a 60×60 grid, which follows the setting in [5]. Through the division, 5 pairs of metro stations fall into the same cell, and one of each pair is assigned to the nearest cell to avoid overlapping. The resultant image input for CNN is exemplified in Figure 5. The value of each cell is set to the average ridership during the trained time frames of the corresponding metro station. The number of hidden unit of LSTM, as well as SBULSTM, is set to 1000 for SZMetro and SHMetro, 600 for HZMetro; two stacked GCN

TABLE II
RESULTS OF TASK 1 ON SZMETRO

Model	MAE	RMSE	MAPE	Average time per epoch
LSTM	8.49	15.02	30.31%	2s
CNN	8.69	15.97	31.08%	1s
GCN	8.65	15.83	30.55%	1s
DMVST-Net	8.34	14.85	29.80%	2s
CNN-LSTM	8.35	14.64	30.56%	2s
SRCNs	8.36	15.68	28.15%	3s
SBULSTM	8.27	15.13	27.72%	2s
DCRNN	8.29	14.84	28.32%	23s
STGCN	8.46	14.91	31.95%	4s
Graph-WaveNet	8.35	16.12	31.24%	12s
PVGCN	8.24	14.05	30.01%	264s
GCN-SBULSTM	7.96	14.41	27.94%	3s
GCN-SBULSTM w/o OD	8.02	14.60	27.72%	3s
GCN-SBULSTM w/o dist	8.04	14.48	28.44%	3s
GCN-SBULSTM w/o K-hop	8.06	14.56	27.86%	3s

TABLE III
RESULTS OF TASK 2 ON SZMETRO

Model	MAE	RMSE	MAPE	Average time per epoch
LSTM	9.00	16.39	31.45%	2s
CNN	9.74	18.80	33.33%	2s
GCN	9.20	17.22	30.77%	1s
DMVST-Net	8.89	16.22	31.43%	3s
CNN-LSTM	8.78	15.83	31.34%	3s
SRCNs	8.84	16.78	29.57%	3s
SBULSTM	8.58	16.32	28.84%	3s
DCRNN	8.56	15.47	28.23%	45s
STGCN	8.77	15.97	33.94%	6s
Graph WaveNet	8.75	16.56	29.68%	12s
PVGCN	8.48	14.90	29.39%	530s
GCN-SBULSTM	8.36	15.62	28.38%	4s
GCN-SBULSTM w/o OD	8.42	15.87	28.47%	4s
GCN-SBULSTM w/o dist	8.44	15.86	28.87%	4s
GCN-SBULSTM w/o K-hop	8.46	15.99	28.55%	4s

layers with 60 and 80 channels and a fully connected layer with hidden units of 10 are sequentially connected in the GCN module.

As for optimizing the training process of GCN-SBULSTM, the batch size is set to 32 and 64, respectively, for Task 1 and Task 2 on SZMetro, while the batch size is 8 and 64 for HZMetro and SHMetro. Adam is selected as the optimizer for training considering its good performance in preliminary tests. The initial learning rate is set to 0.001 and the decay ratio is 0.1. Early stopping is applied during training to avoid overfitting.

This study evaluates the performance of each model using three common metrics, including Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), and Root Mean Square Error (RMSE), which are defined as follows:

$$\text{MAE} = \frac{1}{n}\sum_{i}^{n}\left|\widehat{Y}_i - Y_i\right| \tag{20}$$

$$\text{MAPE} = \frac{1}{n}\sum_{i}^{n}\frac{\left|\widehat{Y}_i - Y_i\right|}{Y_i} \tag{21}$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i}^{n}\left|\widehat{Y}_i - Y_i\right|} \tag{22}$$

where n is the length of samples, $\widehat{Y}_i$ is the predicted ridership and Y_i is the actual ridership. MAE is also adopted as the loss function in the training process. Specifically, as MAPE and RMSE are respectively sensitive to small ground truth and large error value, we take MAE, which is more robust to outlier and can reflect actual error [49], as the main metric in our following discussion.

V. RESULT ANALYSIS

A. Task 1 and 2 on SZMetro

The performance of different models on Task 1 and 2 are summarized in Table II and Table III, respectively. Among all tested models, CNN and GCN obtain the worst accuracies with MAE values of 8.69/9.74 and 8.65/9.20 for Task 1 and 2, which indicates the limited effectiveness of adopting only spatial dependency in ridership forecasting. However, the better performance of GCN demonstrates its advantages in capturing realistic spatial dependencies for ridership forecasting. By integrating CNN and LSTM to capture both spatial and temporal dependencies, DMVST-Net, CNN-LSTM, and SRCNs can achieve better and similar performance, which reduces the MSE value to around 8.35 in Task 1 and 8.80 in Task 2. However, these models are easily affected by the

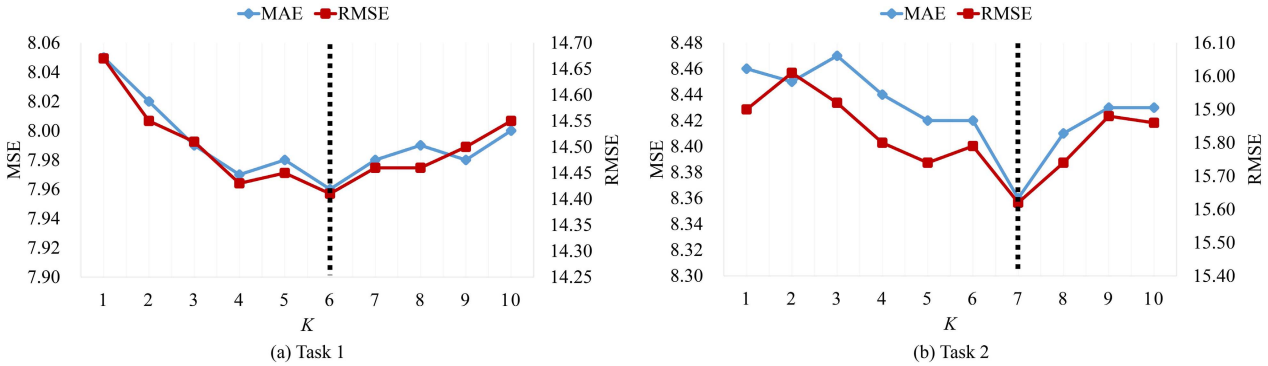

Fig. 6. Results of GCN-SBULSTM with different values of K on SZMetro.

uncertainty of the size of input image for CNN module due to CNN's difficulty in fully representing the topologies of a metro network. Thanks to the advantage of graph learning, DCRNN and PVGCN significantly improve the accuracy with an MAE value lower than 8.30 in Task 1 and 8.60 in Task 2. Notably, even with graph learning module, STGCN and Graph-WaveNet just obtain results with a similar level of CNN-based models, which should be explained as the interference of spatial and temporal dependencies caused by their sequential structure. Surprisingly, SBULSTM achieves competitive performance compared with DCRNN with MAE values of 8.27 in Task 1 and 8.58 in Task 2. Given the limited accuracy of LSTM, the outstanding performance of SBULSTM proves the effectiveness of Bidirectional-LSTM architecture in ridership forecasting.

In comparison with the above-mentioned models, the proposed GCN-SBULSTM is reported to achieve the best prediction accuracy, in terms of the lowest MAE of 7.96 in Task 1 and 8.36 in Task 2. GCN-SBULSTM is also the only model that can obtain an MAE lower than 8.00 in Task 1 and 8.40 in Task 2. The result of ablated models further confirms the effectiveness of each graph used in GCN-SBULSTM: compared with the result of GCN-SBULSTM, GCN-SBULSTM w/o OD obtained a lower accuracy, i.e., MAE of 8.02 and 8.42, which proves the significance of incorporating dynamic spatio-temporal relationship in the GCN module; the accuracy further decreased in GCN-SBULSTM w/o dist, of which the MAE is 8.04 and 8.44, indicating the positive effect of travel distance graph in GCN module; compared to other ablated models, GCN-SBULSTM w/o K-hop has the worst performance in two tasks with MAE values of 8.06 and 8.46, respectively. However, even though accuracy decreased in these ablated models, they still outperform the other tested models, validating the general effectiveness of the model design.

As shown in Table II and III, CNN and GCN are the most efficient models amongst all tested models, while the proposed GCN-SBULSTM achieves competitive training efficiency in terms of its average training time. Specifically, GCN-LSTM only requires 3s and 4s respectively for task 1 and 2, which is slightly higher than all basic architectures, including LSTM, CNN and GCN, and some advanced models, including SBULSTM, CNN-LSTM, and DMVST-Net. In contrast, PVCGN, the second-best model in terms of prediction accuracy in two tasks, is the least efficient model taking 264s and 530s per epoch for task 1 and 2, which are over 10 times longer than DCRNN and 100 times longer than the proposed GCN-SBULSTM. In summary, the GCN-SBULSTM is significantly efficient considering its high accuracy among other advanced models, which benefits the process of parameter tuning and its migration to different tasks.

To illustrate the influence of different values of K on the accuracy, RMSE and MAE values with respect to different K ranging from 1 to 10 are plotted in Figure 6 for task 1 and 2. It can be seen that RMSE and MSE generally start with a high value, then gradually decrease to its minimum at $K = 6$ and $K = 7$ for task 1 and 2, respectively, and finally increase as K becomes larger. Notably that lines in Figure 6 are not ideally smooth, which can be explained as the uncertainty introduced by some random factors during the training process, such as parameter initialization and dropout mechanism. The general tendency shown in Figure 6 prove that: 1) except for adjacent stations, the spatial dependencies among indirectly connected stations to some extent also have positive influences on building up effective prediction model; 2) an overestimated K even lead to negative effects.

B. Benchmark Tests on HZMetro and SHMetro

There are some modifications in the setup of GCN-SBULSTM in this section. Since no station and dynamic OD information is provided in the original article for HZMetro and SHMetro, we make some alternatives to the travel distance graph and population graph for GCN-SBULSTM. For travel distance graph, we compute the number of hops between each pair of stations using the Physical graph (i.e. adjacency graph) and input the result to Equation 4 to generate a "fake" distance weight matrix; as for the population graph, we calculate a single overall population flow graph based on the Correlation graph in [22]. Also, as the information about metro stations is not provided for HZMetro and SHMetro, we cannot generate the traffic images required in CNN-based models. Therefore, CNN, SRCN and DMVST-Net will not be compared during the benchmark tests on HZMetro and SHMetro.

TABLE IV
RESULTS FOR HZMETRO

Time	Metric	LSTM	GCN	SBULSTM	DCRNN	STGCN	Graph-WaveNet	PVCGN	GCN-SBULSTM (K=6)
15min	*MAE*	23.43	23.94	24.31	23.24	23.86	23.50	**22.20**	22.22
	RMSE	40.13	42.89	42.67	41.43	45.03	41.88	38.12	39.83
	MAPE	14.41%	14.74%	14.51%	13.65%	12.48%	13.77%	13.15%	13.16%
30min	*MAE*	24.38	25.99	24.75	25.78	26.07	24.75	23.13	**22.84**
	RMSE	42.33	47.06	43.73	43.23	49.16	43.70	40.00	41.08
	MAPE	15.54%	16.87%	15.04%	15.32%	13.72%	15.68%	13.87%	13.76%
45min	*MAE*	25.33	29.23	25.45	26.23	28.52	25.87	23.95	**23.53**
	RMSE	44.50	53.70	45.49	46.97	52.58	46.50	41.21	42.45
	MAPE	17.18%	20.06%	15.72%	16.01%	15.18%	16.77%	14.89%	14.61%
60min	*MAE*	26.74	33.44	26.46	27.15	31.47	27.85	**24.55**	24.58
	RMSE	47.90	62.39	47.07	48.56	59.74	48.69	42.26	44.48
	MAPE	19.88%	27.62%	17.42%	18.64%	16.95%	20.45%	16.35%	15.73%

TABLE V
RESULTS FOR SHMETRO

Time	Metric	LSTM	GCN	SBULSTM	DCRNN	STGCN	Graph-WaveNet	PVCGN	GCN-SBULSTM (K=8)
15min	*MAE*	23.50	24.21	23.16	23.34	23.84	23.75	22.85	**22.75**
	RMSE	47.08	49.20	45.31	47.24	47.18	47.73	45.47	46.09
	MAPE	20.23%	21.05%	17.40%	18.02%	18.71%	20.23%	16.95%	16.50%
30min	*MAE*	24.50	25.75	24.17	25.33	26.99	27.12	24.16	**23.77**
	RMSE	49.63	52.34	48.39	51.31	57.40	54.15	50.18	49.04
	MAPE	22.64%	24.26%	18.52%	19.12%	19.41%	21.42%	18.83%	17.62%
45min	*MAE*	25.59	28.64	25.38	27.65	30.81	29.23	25.45	**25.02**
	RMSE	53.35	57.48	53.67	57.21	67.61	60.10	54.84	52.89
	MAPE	24.39%	29.36%	20.07%	20.42%	20.46%	22.64%	18.83%	18.95%
60min	*MAE*	26.87	31.60	26.41	29.01	33.82	31.56	26.37	**25.87**
	RMSE	56.53	63.24	59.27	63.32	77.00	68.10	58.49	55.41
	MAPE	26.16%	34.25%	21.45%	21.52%	23.69%	24.92%	19.67%	20.12%

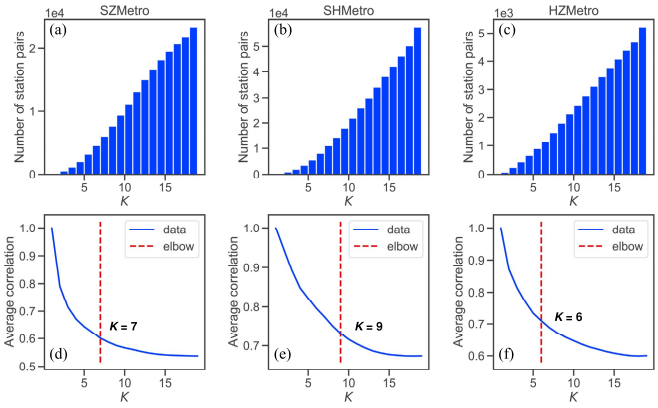

Fig. 7. Influence of K on station pairs and their average correlation.

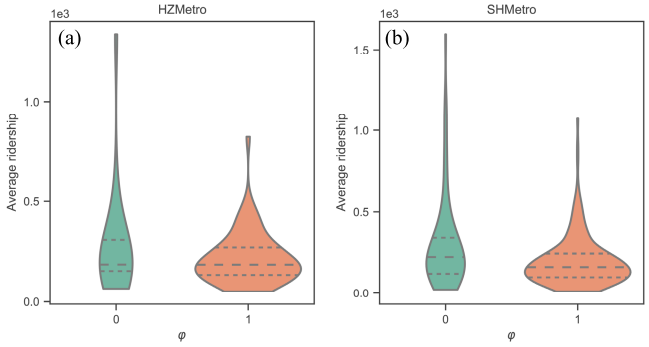

Fig. 8. Violin plots of the relationship between event φ and average ridership.

The performances of different models on HZMetro and SHMetro are summarized in Table IV and V, respectively. The general tendency is similar to the previous experimental results on SZMetro. GCN has the worst performance on these two datasets as it only captures spatial dependency for prediction. Surprisingly, STGCN and Graph-WaveNet become even worse than LSTM, especially for the prediction at 60 min, further indicating the poor performance of parallel structure on ridership data with large time intervals, in which temporal regularity is more significant and dominant than those with short time intervals. DCRNN obtains satisfactory accuracy with MAE of 23.24 for HZMetro and 23.34 for SHMetro at the first time interval. However, with the increment of time, the accuracy of DCRNN dramatically decreases. PVCGN achieves competitive accuracy, especially on HZMetro, and in terms of RMSE, PVCGN is always the best one on HZMetro, indicating its advantage in reducing outlying predictions.

Compared with other models, the proposed GCN-SBULSTM achieves the best accuracy in terms of MAE at 30min and 45min on HZMetro ($K = 6$) and significantly outperforms the other models with a large margin on SHMetro ($K = 8$). RMSE values obtained by GCN-SBULSTM on SHMetro are also improved and surpass those of PVCGN in most cases except for the first interval. Given the lack of station information and the dynamic changes of population flow in this experiment, we believe the

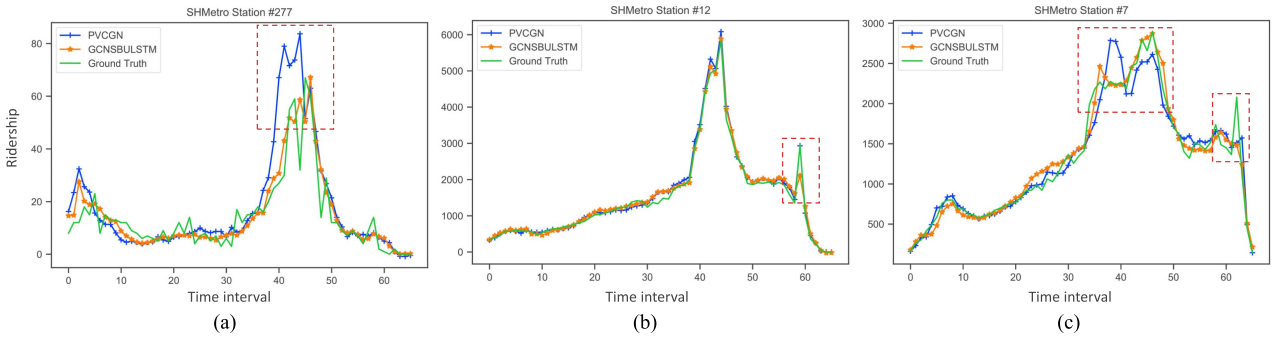

Fig. 9. Snapshot of three prediction instances. Station #277 and #12 are respectively of the lowest and highest average ridership in SHMetro, Station #7 is a more general instance with moderate average ridership.

performance of GCN-SBULSTM on HZMetro and SHMetro can be further improved if necessary data are available.

C. Results Analysis

1) Connotation of Optimal K: According to previous experiments, a K-hop matrix is proved to preserve more comprehensive spatial dependencies than the traditional adjacency matrix and thus promoting the performance of the GCN module. However, an overestimated K is prone to having negative effects on prediction accuracy. To explain such observation and investigate the connotation of optimal K, statistical analyses are performed on each experimental dataset.

As shown in Figure 7 (a) to (c), the number of station pairs rapidly rises as K increases, while the average correlation curve (Figure 7 (d) to (f)), which is computed as the average absolute Pearson correlation coefficient among station pairs, dramatically declines within the first few steps and finally reaches a stable state. This indicates that the correlation of ridership, either positive or negative, becomes less significant between high-order-neighboured stations.

Given the above observation, an appropriate K is expected to balance the number of station pairs and the significance of the correlation between them. In that sense, we compute the elbow point of each curve in Figure 7 (d) to (f). Mathematically, the elbow point is defined as the point with maximum curvature on a curve [50]. In our cases, the elbow point refers to a cutoff K value so that adding higher-order-neighboured stations does not result in better capture of spatial correlation. It is found that the elbow point is 7, 9 and 6 for SZMetro, SHMetro and HZMetro, which is very close to the optimal values in previous experiments, i.e. 6 and 7 for two tasks on SZMetro, 8 for SHMetro, and 6 for HZMetro. This observation largely explains the underlying rationality of optimal K in each prediction task. Also, the elbow point of average correlation curve can be used as an important reference for selecting optimal K when training GCN-SBULSTM.

2) Visulization Analysis: To further illustrate the advantages of GCN-SBULSTM, a visualization analysis is conducted on HZMetro and SHMetro by taking PVCGN as the control method. Since ridership volume is proved to be a critical factor affecting prediction accuracy [22], we focus on exploring the performance of GCN-SBULSTM for different ridership volumes. We first define an event φ for each station, where $\varphi = 0$ when PVCGN obtains a lower MAE than GCN-SBULSTM; otherwise, $\varphi = 1$. A violin plot is used to depict the distribution of φ as well as its relationship to the average ridership. As shown in Figure 8, the violins with $\varphi = 1$ are much 'fatter' than those with $\varphi = 0$, which means that, for most stations of SHMetro and HZMetro, GCN-SBULSTM achieves lower MAE than PVCGN. Moreover, the dash lines inside violins with $\varphi = 1$, which refer to quartiles of the distribution, are generally lower than those with $\varphi = 0$, indicating that GCN-SBULSTM is more suitable for low-ridership stations.

Three instances are further selected from SHMetro to illustrate the performance of GCN-SBULSTM on different ridership volumes. As shown in Figure 9 (a), GCN-SBULSTM performs well in capturing the overall trend as well as narrow fluctuation in low ridership, while PVCGN is likely to overestimate the ridership in many cases. As for the station with high ridership in Figure 9 (b), GCN-SBULSTM produces more accurate predictions in most cases but does not fully capture the marked drastic fluctuation. In comparison, PVCGN seems to be more sensitive to this kind of fluctuation, but overestimation can be still easily observed. Furthermore, such sensitivity of PVCGN might be also invalid and introduce uncertainty, such as the significant bias and miss of fluctuation marked in Figure 9 (c).

In summary, PVCGN seems to provide a radical prediction, which sometimes overreacts to ridership fluctuation and prone to producing overestimated results. In contrast, the proposed GCN-SBULSTM achieves a higher prediction accuracy than PVCGN in most instances and can better handle the fluctuations especially for low-ridership stations.

VI. Conclusion and Discussion

Metro ridership forecasting is a fundamental issue in modern public transportation management. Focusing on improving the accuracy of metro ridership forecasting, this study proposes GCN-SBULSTM, a novel deep learning network with a parallel structure concatenating GCN and SBULSTM modules. In the GCN module, a novel K-hop weight matrix, which integrates adjacency, travel distance, and population flow, is introduced to capture comprehensive spatial correlation within a metro network. GCN-SBULSTM inherits both the

merits of GCN and SBULSTM, and the parallel structure helps preserve most independence of spatial and temporal information, thus reduce the uncertainty caused by their interference.

According to the results on three real-world datasets, the proposed GCN-SBULSTM outperform the state-of-the-art models in terms of its high accuracy and training efficiency. The slightly poorer performance of ablated models verified the effectiveness of using both physical and virtual graphs in improving the overall accuracy. Additionally, in comparison with CNN-based models, the higher accuracy obtained by GCN-based models indicates the superiority of treating traffic network as a graph than a simple 2D image in network-related traffic forecasting tasks. Last but not least, the relatively lower accuracy of STGCN and Graph-WaveNet verifies the hypothesis that the parallel structure can preserve, at the most extent, the integrity of both spatial and temporal dependencies for better prediction.

Improvements can be made in future work. One issue is to incorporate more factors, such as weather condition and holiday events which may correlate with ridership patterns, to enhance the prediction model. Apart from that, it is notable that the proposed model is only applied for inbound ridership prediction. As passengers' outbound preference highly depends on time schedules and the functional zone where a metro station locates, it is not sufficient to accurately forecast outbound ridership using solely the number of previous trips in a time-series form, especially for a short time interval. However, provided accurate time schedules and other auxiliary information that can support the diagnose of passengers' preference, the fundamental idea of GCN-SBULSTM is expected to apply to outbound ridership prediction as well with further improvement.

REFERENCES

[1] C. Ding, D. Wang, X. Ma, and H. Li, "Predicting short-term subway ridership and prioritizing its influential factors using gradient boosting decision trees," *Sustainability*, vol. 8, no. 11, p. 1100, 2016. [Online]. Available: https://www.mdpi.com/2071-1050/8/11/1100

[2] S. Derrible and C. Kennedy, "Evaluating, comparing, and improving metro networks: Application to plans for Toronto, ON, Canada," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2146, no. 1, pp. 43–51, Jan. 2010, doi: 10.3141/2146-06.

[3] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 865–873, Apr. 2015. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6894591/

[4] J. Zhang, Y. Zheng, D. Qi, R. Li, and X. Yi, "DNN-based prediction model for spatio-temporal data," in *Proc. 24th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.* New York, NY, USA: ACM, Oct. 2016, pp. 92:1–92:4, doi: 10.1145/2996913.2997016.

[5] X. Ma, J. Zhang, B. Du, C. Ding, and L. Sun, "Parallel architecture of convolutional bi-directional LSTM neural networks for network-wide metro ridership prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 6, pp. 2278–2288, Jun. 2019.

[6] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015. [Online]. Available: http://www.nature.com/articles/nature14539

[7] S. Shekhar and B. M. Williams, "Adaptive seasonal time series models for forecasting short-term traffic flow," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2024, no. 1, pp. 116–125, Jan. 2007, doi: 10.3141/2024-14.

[8] X. Li *et al.*, "Prediction of urban human mobility using large-scale taxi traces and its applications," *Frontiers Comput. Sci.*, vol. 6, no. 1, pp. 111–121, Feb. 2012.

[9] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas, "Predicting taxi–passenger demand using streaming data," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1393–1402, Sep. 2013.

[10] R. Yu, Y. Li, C. Shahabi, U. Demiryurek, and Y. Liu, "Deep learning: A generic approach for extreme condition traffic forecasting," in *Proc. SIAM Int. Conf. Data Mining*, 2017, pp. 777–785, doi: 10.1137/1.9781611974973.87.

[11] Z. Cui, R. Ke, Z. Pu, and Y. Wang, "Deep bidirectional and unidirectional LSTM recurrent neural network for network-wide traffic speed prediction," 2018, *arXiv:1801.02143*. [Online]. Available: http://arxiv.org/abs/1801.02143

[12] R. Fu, Z. Zhang, and L. Li, "Using LSTM and GRU neural network methods for traffic flow prediction," in *Proc. 31st Youth Acad. Annu. Conf. Chin. Assoc. Autom. (YAC)*, Nov. 2016, pp. 324–328.

[13] X. Cheng, R. Zhang, J. Zhou, and W. Xu, "DeepTransport: Learning spatial-temporal dependency for traffic condition forecasting," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–8.

[14] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang, "Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction," *Sensors*, vol. 17, no. 4, p. 818, Apr. 2017. [Online]. Available: https://www.mdpi.com/1424-8220/17/4/818

[15] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 1655–1661. [Online]. Available: http://dl.acm.org/citation.cfm?id=3298239.3298479

[16] Z. Xie, W. Lv, S. Huang, Z. Lu, B. Du, and R. Huang, "Sequential graph neural network for urban road traffic speed prediction," *IEEE Access*, vol. 8, pp. 63349–63358, 2019.

[17] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," 2019, *arXiv:1901.00596*. [Online]. Available: http://arxiv.org/abs/1901.00596

[18] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," Feb. 2018, *arXiv:1707.01926*. [Online]. Available: http://arxiv.org/abs/1707.01926

[19] L. Zhao *et al.*, "T-GCN: A temporal graph convolutional network for traffic prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 9, pp. 3848–3858, Sep. 2020.

[20] G. Jin, Y. Cui, L. Zeng, H. Tang, Y. Feng, and J. Huang, "Urban ride-hailing demand prediction with multiple spatio-temporal information fusion network," *Transp. Res. C, Emerg. Technol.*, vol. 117, Aug. 2020, Art. no. 102665.

[21] B. Lu, X. Gan, H. Jin, L. Fu, and H. Zhang, "Spatiotemporal adaptive gated graph convolution network for urban traffic flow forecasting," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2020, pp. 1025–1034.

[22] L. Liu, J. Chen, H. Wu, J. Zhen, G. Li, and L. Lin, "Physical-virtual collaboration modeling for intra-and inter-station metro ridership prediction," Jun. 2020, *arXiv:2001.04889*. [Online]. Available: http://arxiv.org/abs/2001.04889

[23] Z. Cui, K. Henrickson, R. Ke, and Y. Wang, "Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 11, pp. 4883–4894, Nov. 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8917706/

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Red Hook, NY, USA: Curran Associates, 2012, pp. 1097–1105.

[25] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face recognition: A convolutional neural-network approach," *IEEE Trans. Neural Netw.*, vol. 8, no. 1, pp. 98–113, Jan. 1997, doi: 10.1109/72.554195.

[26] H. Yu, Z. Wu, S. Wang, Y. Wang, and X. Ma, "Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks," *Sensors*, vol. 17, no. 7, p. 1501, Jun. 2017. [Online]. Available: https://www.mdpi.com/1424-8220/17/7/1501

[27] H. Yao *et al.*, "Deep multi-view spatial-temporal network for taxi demand prediction," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 1–8.

[28] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph WaveNet for deep spatial-temporal graph modeling," 2019, *arXiv:1906.00121*. [Online]. Available: http://arxiv.org/abs/1906.00121

[29] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 3634–3640. [Online]. Available: http://arxiv.org/abs/1709.04875

[30] B. Du, X. Hu, L. Sun, J. Liu, Y. Qiao, and W. Lv, "Traffic demand prediction based on dynamic transition convolutional neural network," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 2, pp. 1237–1247, Feb. 2021.

[31] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*. [Online]. Available: http://arxiv.org/abs/1609.02907

[32] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," 2013, *arXiv:1312.6203*. [Online]. Available: https://arXiv.1312.6203

[33] M. Henaff, J. Bruna, and Y. LeCun, "Deep convolutional networks on graph-structured data," 2015, *arXiv:1506.05163*. [Online]. Available: http://arxiv.org/abs/1506.05163

[34] W. R. Tobler, "A computer movie simulating urban growth in the Detroit region," *Econ. Geography*, vol. 46, pp. 234–240, Jun. 1970.

[35] X. Ma, J. Zhang, C. Ding, and Y. Wang, "A geographically and temporally weighted regression model to explore the spatiotemporal influence of built environment on transit ridership," *Comput., Environ. Urban Syst.*, vol. 70, pp. 113–124, Jul. 2018.

[36] H. Yang, X. Lu, C. Cherry, X. Liu, and Y. Li, "Spatial variations in active mode trip volume at intersections: A local analysis utilizing geographically weighted regression," *J. Transp. Geography*, vol. 64, pp. 184–194, Oct. 2017.

[37] G. Kusano, Y. Hiraoka, and K. Fukumizu, "Persistence weighted Gaussian kernel for topological data analysis," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 2004–2013.

[38] S. Raveau, J. C. Muñoz, and L. de Grange, "A topological route choice model for metro," *Transp. Res. A, Policy Pract.*, vol. 45, no. 2, pp. 138–147, Feb. 2011.

[39] D. An, X. Tong, K. Liu, and E. H. W. Chan, "Understanding the impact of built environment on metro ridership using open source in Shanghai," *Cities*, vol. 93, pp. 177–187, Oct. 2019.

[40] Y. Gong, Y. Lin, and Z. Duan, "Exploring the spatiotemporal structure of dynamic urban space using metro smart card records," *Comput., Environ. Urban Syst.*, vol. 64, pp. 169–183, Jul. 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0198971516301089

[41] K. Greff, R. K. Srivastava, J. Koutnik, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.

[42] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[43] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997.

[44] A. Graves, S. Fernández, and J. Schmidhuber, "Bidirectional LSTM networks for improved phoneme classification and recognition," in *Artificial Neural Networks: Formal Models and Their Applications—ICANN*, W. Duch, J. Kacprzyk, E. Oja, and S. Zadrożny, Eds. Berlin, Germany: Springer, 2005, pp. 799–804.

[45] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Netw.*, vol. 18, nos. 5–6, pp. 602–610, Jul. 2005.

[46] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 6645–6649.

[47] A. Graves, N. Jaitly, and A.-R. Mohamed, "Hybrid speech recognition with deep bidirectional LSTM," in *Proc. IEEE Workshop Autom. Speech Recognit. Understand.*, Dec. 2013, pp. 273–278.

[48] A. Ray, S. Rajeswar, and S. Chaudhury, "Text recognition using deep BLSTM networks," in *Proc. 8th Int. Conf. Adv. Pattern Recognit. (ICAPR)*, Jan. 2015, pp. 1–6.

[49] T. Chai and R. R. Draxler, "Root mean square error (RMSE) or mean absolute error (MAE)?—Arguments against avoiding RMSE in the literature," *Geosci. Model Develop.*, vol. 7, no. 3, pp. 1247–1250, Jun. 2014.

[50] Q. Zhao, V. Hautamaki, and P. Fränti, "Knee point detection in BIC for detecting the number of clusters," in *Proc. Int. Conf. Adv. Concepts Intell. Vis. Syst.* Berlin, Germany: Springer, 2008, pp. 664–673.

Pengfei Chen received the B.S., M.S., and Ph.D. degrees from Wuhan University, in 2012, 2015, and 2019, respectively, and the joint Ph.D. degree from The Hong Kong Polytechnic University, in 2020. He is currently an Assistant Professor with the School of Geospatial Engineering and Science, Sun Yat-Sen University, Guangdong, China. He is also a Key Member of the Polar Oceans and Climate Change Innovation Team, Southern Marine Science and Engineering Guangdong Laboratory, Zhuhai. His research interests include human mobility modeling, geospatial artificial intelligence, and spatial data uncertainty.

Xuandi Fu received the B.S. degree from The Hong Kong Polytechnic University in 2017. She is currently pursuing the M.S. degree with the Department of Electrical and Computer Engineering, Carnegie Mellon University, USA. Her research interests include natural language processing, graph convolutional neural networks, human mobility modeling, and spatial data analytics.

Xue Wang received the B.S. and M.S. degrees from Peking University, in 2012 and 2015, respectively, and the Ph.D. degree from The Chinese University of Hong Kong, in 2019. She currently works as an Assistant Professor with the School of Geospatial Engineering and Science, Sun Yat-Sen University, Guangdong, China. She also works as a Key Member of the Polar Oceans and Climate Change Innovation Team, Southern Marine Science and Engineering Guangdong Laboratory, Zhuhai. Her research interests include urban informatics and change detection.