Cellular Network Traffic Prediction Incorporating Handover: A Graph Convolutional Approach

Shuai Zhao [*] Xiaopeng Jiang [†] Guy Jacobson [‡] Rittwik Jana [‡] Wen-Ling Hsu [‡]
Raif Rustamov [‡] Manoop Talasila [‡] Syed Anwar Aftab [‡] Yi Chen [*] Cristian Borcea [†]
[*] Martin Tuchman School of Management, Leir Research Institute for Business, Technology, and Society
New Jersey Institute of Technology, Newark, NJ, USA
[†] Department of Computer Science, New Jersey Institute of Technology, Newark, NJ, USA
[‡] AT&T Labs, Bedminster, NJ, USA
Email:{sz255,xj8,yi.chen,borcea}@njit.edu, {guy,rjana,hsu,rustamov,talasila,anwar}@research.att.com

***Abstract*—Cellular traffic prediction enables operators to adapt to traffic demand in real-time for improving network resource utilization and user experience. To predict cellular traffic, previous studies either applied Recurrent Neural Networks (RNN) at individual base stations or adapted Convolutional Neural Networks (CNN) to work at grid-cells in a geographically defined grid. These solutions do not consider explicitly the effect of handover on the spatial characteristics of the traffic, which may lead to lower prediction accuracy. Furthermore, RNN solutions are slow to train, and CNN-grid solutions do not work for cells and are difficult to apply to base stations. This paper proposes a new prediction model, STGCN-HO, that uses the transition probability matrix of the handover graph to improve traffic prediction. STGCN-HO builds a stacked residual neural network structure incorporating graph convolutions and gated linear units to capture both spatial and temporal aspects of the traffic. Unlike RNN, STGCN-HO is fast to train and simultaneously predicts traffic demand for all base stations based on the information gathered from the whole graph. Unlike CNN-grid, STGCN-HO can make predictions not only for base stations, but also for cells within base stations. Experiments using data from a large cellular network operator demonstrate that our model outperforms existing solutions in terms of prediction accuracy.**

***Index Terms*—cellular traffic prediction, handover, deep learning, spatio-temporal modeling, LTE, 5G, radio access networks**

I. Introduction

Cellular traffic prediction can facilitate resource allocation and radio access network (RAN) management [24], and therefore improves resource utilization and user experience. For example, if we can correctly predict traffic in an area of interest, the network operator can turn off certain cells to save energy, thus promoting greener cellular networks. Incorporating cellular traffic prediction into the next generation 5G systems is beneficial for advanced RAN features such as beamforming, massive MIMO, and network slicing. For example, by predicting the change in traffic due to user mobility, the weights of antenna elements can be dynamically optimized for best signal coverage. Another example of utilizing traffic prediction is network slicing based on the predicted traffic, as well as other information such as the user's service level agreement (SLA): the operator can create multiple dedicated virtual networks, where each slice can be managed and orchestrated independently guaranteeing the required SLA. Finally, emerging applications including autonomous vehicles, 360-degree panoramic videos, and remote IoT control will account for a significant fraction of future traffic demand. To meet the growing demand for such high bandwidth and low latency applications, the 5G RAN controllers[1] can use traffic prediction for demand-aware allocation of network resources.

Cellular network traffic prediction is challenging due to the high variability of traffic. In addition, the problem is made more difficult by the large heterogeneity among different base stations whose traffic loads differ significantly. Some are heavily used, whereas others are not. Deep learning models have shown promising results for traffic prediction, as they are effective in discovering intricate patterns. Previous studies either applied Recurrent Neural Networks (RNN), specifically Long Short Term Memory (LSTM) models, at base stations [1], [3], [6], [8], [12], [15], [25], [26] or adapted Convolutional Neural Networks (CNN) [22], [23] to work at grid-cells level in a geographically defined grid (i.e., CNN-grid models). These solutions have a number of drawbacks. First, they do not consider explicitly the effect of handover on the spatial characteristics of the traffic, which may lead to low prediction accuracy. Second, RNN solutions are slow to train because the computation is necessarily sequential, and preprocessing [8], [15] further adds computational time. Third, while CNN-grid models are fast to train using parallelization, they do not work for traffic prediction at cell-level and are difficult to apply at base stations.

CNN-grid approaches for traffic prediction split the Euclidean space into grids and output the predictions at each grid-cell. While these models can successfully exploit and capture the shift-invariance and local connectivity when dealing with signals that have an underlying Euclidean structure, they can not be readily applied on non-Euclidean geometric data [18]. This is a problem in our case because the interaction in cellular networks is more naturally represented with non-Euclidean geometric data, such as the one provided by the handover graph.

[1]https://www.o-ran.org/

In cellular networks, location proximity does not necessarily mean frequent spatial interaction, when user mobility [17] and terrain effects [11] are considered. For example, when all users are moving in one direction during rush hour, the interaction of base stations mostly occurs along the direction of user flow, while two close base stations perpendicular to the user flow may rarely interact. Also, large structures or terrain may completely block two close base stations from interacting with each other. In addition, these CNN models do not work for cells, as the traffic of cells sharing the same location can not be separated. Also, they are difficult to apply to base stations, as each grid cell typically contains multiple base stations. While the grid cells could be made small enough to cover just one base station, this would be inefficient as many grid-cells would be empty.

This paper proposes a new prediction model, STGCN-HO, that explicitly incorporates the fine-grained handover information. Our model builds a directed, weighted handover graph for cells or base stations based on their handover frequencies (i.e., the number of handovers over a period of time). STGCN-HO uses the transition probability matrix of the handover graph to capture the spatial characteristics of the traffic. This matrix is used by a stacked residual neural network structure incorporating graph convolutions and gated linear units, which also captures the temporal characteristics of the traffic. The STGCN-HO model can better capture the spatial dependencies and complex non-linear patterns within the cellular network, compared to existing work. In addition, unlike RNN, STGCN-HO is fast to train and simultaneously predicts traffic for all base stations using the information gathered from the whole graph. Furthermore, unlike CNN-grid, STGCN-HO can efficiently separate individual cells or base stations for fine-grain traffic prediction.

The STGCN model is implemented using the Tensorflow-GPU library, and we utilized the standard RMSProp optimizer with exponential-decaying learning rate starting from 0.1 for tuning parameters. We evaluated our model based on real-world 4G LTE traffic data contributed by a major telecom company. In the evaluation, we compared STGCN-HO against Historical Average (HA), ARIMA [7], a vanilla LSTM model, and a state-of-art variation of LSTM, namely multi-task LSTM [12].

The experimental results demonstrate STGCN-HO is better than the comparison models in terms of prediction accuracy both at cell level and base station level. The results are statistically significant, with one-tailed T-test p value < 0.01. The superiority of our model becomes more apparent for longer prediction intervals: at cell-level, the performance of STGCN-HO is 5.53%, 8.03%, and 8.94% better for 15 minutes, 30 minutes, and 45 minutes predictions, respectively when compared with the best comparison model, multi-task LSTM. At base station-level, STGCN-HO is 9.40%, 14.2%, and 16.5% better than the multi-task LSTM. Also, the results show our model works better for cells with high handover frequency with neighbor cells, which further demonstrates the usefulness of incorporating the handover graph information.

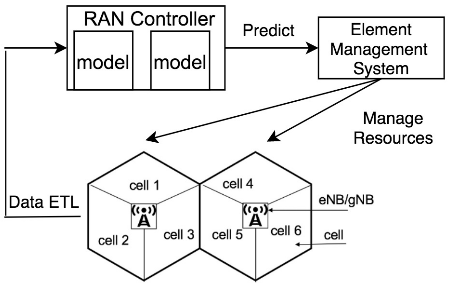

Fig. 1. Example of traffic prediction usage in 4G/5G cellular networks

Furthermore, we show the training time for STGCN-HO is approximately one order of magnitude lower than the training time for vanilla LSTM, which is faster to train than the multi-task LSTM.

The rest of the paper is organized as follows. Section II overviews background information of handover in cellular networks and discusses related work for cellular traffic prediction. After defining the problem in Section III, we present our STGCN-HO model in Section IV. Section V describes the dataset and the data preprocessing. Section VI shows the experimental results and analysis. The paper concludes in Section VII.

II. Background and Related Work

A. Network Handover Overview

Figure 1 depicts how traffic prediction models can be embedded in a RAN controller and resource management system. This paper presents the traffic prediction model, and our future work will integrate the model into the RAN controller. A base station (called eNB in 4G networks and gNB in 5G networks) controls multiple cells, and handovers happen during cell-level communication, either within the same base station or across different base stations. The basic handover between cells follows a simple protocol: if one cell gets overloaded, it will send a request to hand user traffic over to neighboring cells that are less busy. However, the handover may also be driven by user mobility. Sometimes, the cells within the same eNB are very close to each other and share the same location, which indicates that intra-eNB handover is more likely driven by the load balancing policy of the operators. Inter-eNB handover, on the other hand, is more likely driven by user mobility. This difference also motivates us to investigate both cell-level and eNB-level traffic prediction.

B. Related Work

Early exploration of cellular traffic prediction used various types of statistical modeling [2], [7], [16], [19], with the Autoregressive Integrated Moving Average (ARIMA) [7] model still being used as a comparison baseline by newer works. More recently, deep learning has been successfully applied to real cellular traffic data collected in Europe and China. Most efforts focused on RNN, specifically LSTM [3]. LSTM variations include multi-layer LSTM [26], LSTM with data decomposition for single base station traffic prediction [6], and multivariate LSTM with data from a collection of base stations as an input matrix [1]. These models learn the spatial correlation of the traffic implicitly. However, if the spatial correlation can be incorporated explicitly in the models, the prediction accuracy is expected to improve.

Attempts have been made to model the spatial correlation explicitly, as early as the statistical approaches mentioned above [2], [16]. The study of Qiu et al. [12] jointly explored spatio-temporal correlations. The traffic data was fed into both dedicated learning machines for each base station and into shared learning machines which spanned neighboring base stations, where all the learning machines used LSTM models. However, the neighboring base stations considered in the model were limited to those in close proximity. Spatial correlation of expanded regions beyond proximity, helpful for longer prediction intervals when the users are travelling longer distances, was not explored. Also, while the interactions among base stations may be captured by the model, they are not explicitly defined or interpreted.

Instead of feeding cellular traffic data directly into the learning machines, some studies preprocessed the data to extract certain features prior to learning. Wang et al. [15] examined a dataset from a large LTE network in China Mobile and showed non-zero correlation in both temporal and spatial domains, which motivated them to use an autoencoder-based deep model for spatial modeling and LSTM for temporal modeling. Feng et al. [8] proposed Deep Traffic Predictor (DeepTP), consisting of feature extractors for spatial correlation and discrete factors such as point of interest, hour of day, and day of week, as well as layers of LSTM for modeling temporal variations. However, these sophisticated RNN-based networks are widely known to be computationally expensive, and the preprocessing adds even more computation time. More recent successful applications of deep learning models suggest feeding data directly into well-designed parallel learning machines to achieve equivalent or better results with less computation time.

Inspired by the capability of CNNs to capture both regional and global features, a convolutional approach was also applied to cellular traffic prediction. Zhang et al. [22] represented the traffic of each time slot as an image-like two-channel tensor matrix, where periodicity over days and closeness dependence over time were modeled with two deep CNNs and fused together to produce the final output. Zhang and Patras [23] designed a Spatio-Temporal Neural Network (STN) architecture and extended it to Double STN (D-STN) for multi-step mobile traffic prediction. They treated the traffic data over grid areas for each time step as a matrix, ensembled a stack of Convolutional LSTMs (ConvLSTM) and three-dimensional Convolutional Networks (3D-ConvNet) with two fully-connected layers, and concluded that the ensembled model outperformed each individual model. However, these convolutional approaches can only applied to the cellular traffic prediction of grid-based geometry, and cannot predict the traffic of cells or base stations located close to each other. In terms of real-time resource allocation and energy efficiency, when each grid contains different numbers of base stations, the results from these works have less practical value for operators than directly predicting traffic for each base station. Also, location proximity does not necessarily mean frequent interaction, when user mobility and terrain effects are taken into account.

The spatial dependency of the traffic data was further investigated. In the study of Zhang et al. [25], causality was extracted between the main prediction area and the adjacent geographical grids. Based on the extracted causality topological diagram, a multivariate LSTM based on Granger test was used to predict future data. However, models with specific statistical distribution tests are usually not easy to generalize to different data. Wang et al. [17] found that user mobility is tightly correlated with inter-tower traffic. A graph representation of spatio-temporal dependencies was utilized. They separately encoded in-tower traffic and inter-tower traffic into a Graph Neural Network (GNN) that mapped the graphs to the prediction of cellular traffic. The model proposed in our paper is in the same category as this one, but their neural network is much simpler. Unfortunately, since the availability of separate in-tower and inter-tower traffic is challenging (i.e., our data does not include it), their model cannot be compared with ours directly.

Our model, STGCN-HO, addresses the problems of both CNN-grid approaches and RNN approaches. Our main idea is to explicitly incorporate handover in traffic prediction because it generalizes both the neighboring base station interaction and the user mobility. Using fine-grained handover information, STGCN-HO overcomes the problems of CNN-grid approaches and is able to predict traffic both at base station level and cell level. The structure of STGCN-HO, based on a graph CNN to explicitly model the spatial characteristics of the traffic and a gated CNN to capture the temporal traffic characteristics, allows it to make better predictions and be faster to train than RNN approaches.

III. Problem Definition

Consider that we have a finite set of n cells in a cellular network. $\mathbf{V_t} \in \mathbf{R}^n$ represent the traffic observations of the n cells at the t-th timestamp. The handover connections between cells define a directed, weighted graph, with the weights represented by the frequency of handovers between any two cells. The adjacency matrix representing this graph is $\mathbf{A} \in \mathbf{R}^{n\times n}$. Formally, given the traffic at each cell for M historical timestamps, the adjacency matrix $\mathbf{A}$, and some auxiliary features (e.g., day of the week, hour of the day), our goal is to predict the traffic at each cell for the future H timestamps.

$$\mathbf{\hat{V}_{t+1}}, ..., \mathbf{\hat{V}_{t+H}} = F(\mathbf{V_t}, ..., \mathbf{V_{t-M+1}}; \mathbf{A}; \mathbf{X_t^{aux}}) \quad (1)$$

where $\mathbf{X_t^{aux}}$ are the auxiliary features at timestamp t that do not change at every timestamp. The function F is our proposed deep learning model to make traffic predictions.

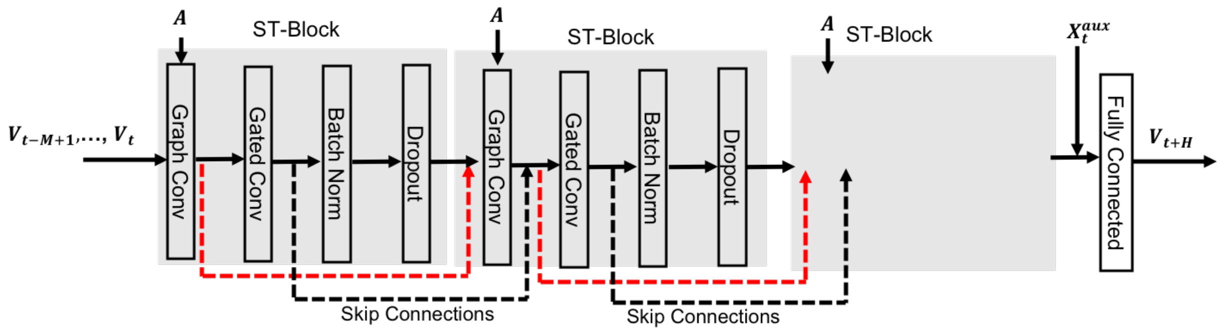

Fig. 2. Architecture of STGCN-HO deep learning prediction model

Although the above problem is defined at cell level, it can easily be generalized to eNB level, which will be considered as well later in this paper.

IV. STGCN-HO Model

This section describes the architecture of STGCN-HO, our deep learning model for cellular traffic prediction, and presents details of each module in the architecture. The section starts with a brief description of the architecture, continues with an overview of convolution on graphs (which is used by our model), and then details each module.

A. *STGCN-HO Architecture*

Our proposed architecture of spatio-temporal graph convolutional networks incorporating handover information (STGCN-HO) can model well spatio-temporal features within one complete neural network structure. As shown in Figure 2, the model architecture comprises of stacked spatio-temporal blocks (ST-Blocks), where each ST-Block has one graph convolution layer and one gated linear convolution layer. Batch normalization and dropout layers are added to avoid overfitting in one block. A densenet-type residual connection is added to avoid the problem of vanishing gradients when training deep stacked ST-Blocks. The auxiliary features are integrated by a fully connected layer.

B. *Overview of Graph Convolution*

Spectral graph convolution based on spectral theory has achieved highly promising results on graph-structured data in recent years, such as 3D skeleton-based action recognition [20] and intelligent transportation [21]. Since our handover data is also graph-structured, we use this method as part of our spatio-temporal prediction model.

Graph convolution is based on graph signal processing, in which graphs are assumed to be undirected. Shuman et al. [14] defined the graph filter on normalized Laplacian matrix $\mathbf{L} = \mathbf{I_n} - \mathbf{D^{-1/2}AD^{-1/2}}$, where $\mathbf{A}$ is the adjacency matrix of the graph, $\mathbf{D}$ is a diagonal matrix of node degrees, and and $\mathbf{I_n}$ is an identity matrix of order n. $\mathbf{L}$ can be factored as $\mathbf{L} = \mathbf{U\Lambda U^{-1}} = \mathbf{U\Lambda U^T}$. $\mathbf{U}$ is the graph Fourier basis, and $\mathbf{\Lambda}$ is identified as the frequencies of the graph. The graph Fourier transform of a signal $\mathbf{x} \in \mathbf{R}^n$ is then defined as $\mathcal{F}(\mathbf{x}) = \mathbf{U^T x}$ and its inverse as $\mathcal{F}^{-1}(\hat{\mathbf{x}}) = \mathbf{U}\hat{\mathbf{x}}$ [14], where $\hat{\mathbf{x}}$ represents the resulting signal from graph Fourier transform.

The graph convolution operator "$*G$" is defined as the multiplication of a signal $\mathbf{x} \in \mathbf{R}^n$ with a kernel $\mathbf{g} \in \mathbf{R}^n$,

$$\mathbf{x} * G\mathbf{g} = \mathcal{F}^{-1}(\mathcal{F}(\mathbf{g}) \odot \mathcal{F}(\mathbf{x})) = \mathbf{U}(\mathbf{U^T g} \odot \mathbf{U^T x}) \quad (2)$$

where $\odot$ denotes the Hadamard product. The graph filter function is defined as $g\theta = diag(\mathbf{U^T g})$, and $g\theta$ is a function of the eigenvalues of $\mathbf{L}$, i.e., $g\theta(\mathbf{\Lambda})$. Therefore:

$$\mathbf{x} * Gg\theta = \mathbf{U}(g\theta \mathbf{U^T x}) = \mathbf{U}g\theta(\mathbf{\Lambda})\mathbf{U^T x} \quad (3)$$

Since the graph convolution needs to calculate the diagonalization of the normalized Laplacian matrix, it is computationally expensive ($\mathcal{O}(n^3)$). Defferard et al. [5] propose a Chebyshev polynomials approximation method (ChebNet) to reduce the computational cost. Further, Kipf et al. [10] introduce the 1st order approximation ChebNet, i.e. 1stChebNet. Based on these works, equation 3 can be simplified as:

$$\mathbf{x} * Gg\theta = \theta(\mathbf{I_n} + \mathbf{D^{-1/2}AD^{-1/2}})\mathbf{x} \quad (4)$$

1stChebNet can be stacked in a deep architecture to recover spatial information in depth, and its time complexity is $\mathcal{O}(|\mathbf{E}|)$, where $|\mathbf{E}|$ is the number of edges in the graph. More importantly, the adjacency matrix $\mathbf{A}$ does not need to be symmetric [18] due to the avoidance of diagonalization of the Laplacian matrix, which provides the opportunity to process directed graphs.

C. *Application of Graph Convolution in STGCN-HO*

Inspired by the vehicular traffic prediction model proposed by Yu et al. [21], we use graph CNN to model the weighted, directed handover graph. In order to expand a single signal $\mathbf{x}$ into a multi-dimensional cellular network traffic signals $\mathbf{X^l} \in \mathbf{R}^{n \times C_i}$ (i.e., a C_i-dimensional feature vector for every node), the 1stChebNet can be rewritten in this matrix format:

$$\mathbf{X^{l+1}} = \tilde{\mathbf{A}}\mathbf{X^l}\mathbf{\Theta} \quad (5)$$

where $\tilde{\mathbf{A}} = \mathbf{I_n} + \mathbf{D^{-1/2}AD^{-1/2}}$, $\mathbf{\Theta} \in \mathbf{R}^{n \times C_o}$ is the set of trainable parameters and $\mathbf{X^{l+1}} \in \mathbf{R}^{n \times C_o}$ is the output of the graph convolution.

In our case, the adjacency matrix of handover information is directed, weighted, and asymmetric. We model this matrix based on the 1stChebNet. The traffic of a cell is influenced not only by its local traffic but also by the incoming handover traffic from neighbor cells. Specifically, for an adjacency matrix $\mathbf{A} \in \mathbf{R}^{n\times n}$ of a directed graph, we can compute a normalized transition probability matrix $\mathbf{P}$ that gives the probability of moving from node i to node j in one step.

$$\mathbf{P}_{i,j} = \frac{e_{i,j}}{\sum\limits_{k\in\mathbf{E},k\neq i} e_{i,k}}(i \neq j) \tag{6}$$

where $e_{i,j}$ is the edge weight between i and j based on their handover frequency. Using the transition probability matrix $\mathbf{P}$, equation 5 is rewritten as:

$$\mathbf{X^{l+1}} = \mathbf{\tilde{P}^{T}X^{l}\Theta} \tag{7}$$

where $\tilde{\mathbf{P}} = \alpha\mathbf{I_n} + \mathbf{P}$. We add a parameter α in front of the identity matrix as a weight to control the impact of the self node. We transpose matrix $\tilde{\mathbf{P}}$ to properly process the direction of transitions, in order to incorporate the incoming traffic handovers instead of outgoing ones to decide the next traffic state of a cell.

D. Gated Linear Units (GLU) for Extracting Temporal Features in STGCN-HO

Recent studies [9], [21] show CNNs are faster to train, have simpler structures, and have no dependency constraints to previous steps, when compared to RNNs, which still suffer from time-consuming iterations and slow response to dynamic changes. We utilize CNNs with a simplified gating mechanism: gated linear units (GLU) [4] to learn the historical traffic sequences. As shown in Figure 3, a width-K_t convolution kernel is collected incrementally from a length-M traffic history sequence, and the convolution output is fed into a gating mechanism. The gating mechanism consists of a sigmoid gate, a point-wise multiplication, and an addition gate. The right part of Figure 3 shows the computation flow of the gating mechanism. Similar to LSTMs, these gates multiply each element of the matrix, and control the information passed on in the hierarchy.

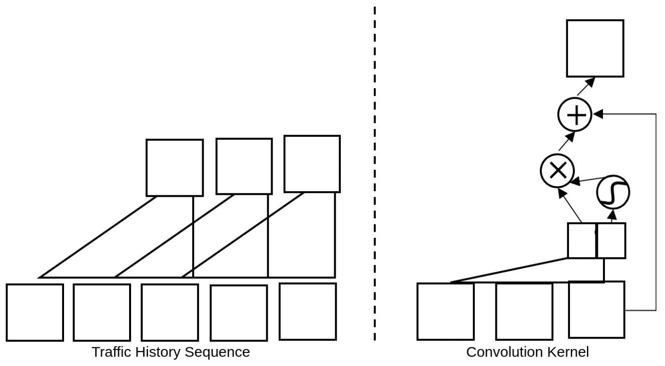

Fig. 3. Structure of Gated Linear Units (kernel size $K_t = 3$ in this example)

E. Spatio-temporal Convolution Block

We fuse features from both spatial and temporal domains by constructing the graph convolution and the gated linear unit as one spatio-temporal convolution block (ST-block). Residual connections are also added for neighbor ST-blocks to resolve the vanishing gradient problem. In order to avoid overfitting and accelerate training, we perform batch normalization and dropout layers within each ST-block. The ST-blocks can easily be stacked or extended based on the scale and complexity of particular cellular network cases, such as the area size and density of the network.

Cellular traffic is influenced by many complex external features, such as big events, week day vs. weekend day, etc. Such auxiliary features do not change at every timestamp. Therefore, it is neither suitable nor necessary to learn these auxiliary features through the deep ST-blocks structure. Instead, a final output layer is added to concatenate previously trained traffic flows with the auxiliary features.

The output layer in our model's architecture (Figure 2) is a fully connected layer, which maps the multiple-channel outputs of the last ST-Block to a single-step prediction. Therefore, the model learns the parameter $\mathbf{w}$ by minimizing the L2 loss measurement, in which $\hat{v}_i$ is the predicted value of cell i.

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \frac{1}{n}\sum_{i=1}^{n} ||\hat{v}_i(\mathbf{v_{t-M+1}}, ..., \mathbf{v_t}, \mathbf{w}) - v_i||^2 \tag{8}$$

V. Dataset and Data Processing

A. Dataset Description

We test our model based on real LTE traffic data contributed by a major telecom company. Cells without traffic for at least 24 hours are considered dead and removed. After removal, our dataset has 89 cells and 26 eNBs within the chosen area. The data was collected during a 6-week period in the fall of 2018. The traffic data is the downlink traffic measured by the number of Physical Resource Blocks (PRBs) allocated per unit of time. PRBs are the best traffic load indicator available in our dataset. Due to the dissimilarity of downlink and uplink traffic, with downlink traffic contributing the most, we decided to investigate only the downlink traffic. The handover data is measured by the number of handovers between each cell/eNB pairs. Let us note that the handover at eNB level is not simply the sum of handovers at cell level because the handover between cells within each eNB is not incorporated in the eNB level handover data.

B. Data Preprocessing

We divide the traffic at each cell into 5-minute intervals. Thus, every cell in the network contains 288 data points per day. Linear interpolation is used to fill in missing values after data cleaning.

Different cells have large variance of their traffic loads. To avoid models over-focusing on optimizing predictions for cells with large traffic loads, it is crucial to normalize the traffic load to a unit size, individually for each cell. Our data are scaled by the min_max scaler (equation 9). We argue the standard scaler (equation 10) is not suitable in our case because the traffic does not follow a Gaussian-like distribution, and it is highly over-dispersed and skewed as the distribution has a "long tail" to larger values. In the min_max scaler, the min and max values are calculated from the training dataset solely in order to avoid any information leaking from the test dataset. Let us also note the model prediction values will be transformed inversely to their original scale when measuring the prediction performance.

$$z = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{9}$$

$$z = \frac{x - u}{\sigma} \tag{10}$$

An illustration of the handover connections between cells is shown in Figure 4. Each heatmap is the handover aggregation for a day, and we randomly select four days to visualize. The heatmaps show that each cell has many handovers with a small number of its neighbor cells. We also observe that the handover connections do not vary significantly over different days, and the traffic pattern is daily cyclical. Therefore, in order to speed up training, the adjacency matrix is based on the aggregation of handover data from only several days chosen randomly. Then, we average the adjacency matrix over days to generate our transition probability matrix $\mathbf{P}$.

In order to control the sparsity of the matrix, we add a threshold ϵ into the transition probability matrix $\mathbf{P}$:

$$\mathbf{P}_{i,j} = \begin{cases} \frac{e_{i,j}}{\sum\limits_{k \in \mathbf{E}, k \neq i} e_{i,k}} (i \neq j), & if \geq \epsilon \\ 0, & otherwise \end{cases} \tag{11}$$

where $e_{i,j}$ is the handover frequency from cell i to j. The time complexity of our graph convolution determined by $|\mathbf{E}|$, the number of edges in the graph. A larger ϵ can speed up

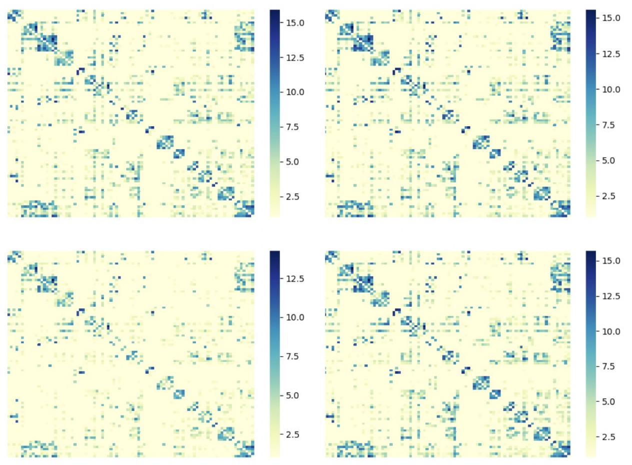

Fig. 4. Heatmap of adjacency matrix for 4 random days; the weight is the handover frequency between cells after log transformation

the convolution process, as it will lead to a sparser adjacency matrix. For auxiliary features, we use one-hot encoding to transform the metadata (i.e., DayOfWeek, HourInDay) into binary dummy vectors.

For eNB experiments, we aggregate the traffic data for all the cells belonging to one eNB. For handover, we consider only inter-eNB handover data. Figure 5 is a plot of a typical cell-level and eNB-level traffic over 3 days. We observe that eNB-level traffic looks smoother than the cell-level traffic. Thus, it may lead to better prediction accuracy.

VI. Evaluation

The evaluation of STGCN-HO has three objectives: (1) assess its sensitivity to different model hyper-parameters in order to select the best ones; (2) compare its prediction accuracy with state-of-art cellular traffic prediction models; and (3) compare its training time with existing models.

A. *Implementation*

The STGCN-HO model is implemented with the Tensorflow-GPU library, and we utilize the standard RMSProp optimizer with exponential-decaying learning rate starting from 0.1 for tuning parameters. The original data is stored in Hadoop HDFS, and we use PySpark to extract and preprocess the data.

B. *Metrics*

To measure the performance of different prediction models, we adopt the following metrics: Mean Absolute Errors (MAE), Root Mean Squared Errors (RMSE), and Relative RMSE (RRMSE) (equation 12). We use the range as the denominator in RRMSE, which is defined as follows.

$$RRMSE = \frac{RMSE}{y_{max} - y_{min}} * 100\% \tag{12}$$

MAE and RMSE are the absolute error terms of our predictions, and RRMSE is the relative percentage of error over the difference between maximum and minimum of the ground truth [13]. RRMSE allows us to compare the performance of cell-level and eNB-level prediction. We calculate the average of the three metrics over all cells or base stations to quantify the overall performance of the models. For all three metrics, the lower the values, the better the performance. We adopt RRMSE instead of the Mean Absolute Percentage Errors (MAPE) because MAPE is problematic when the ground truth is close to zero (i.e., it results in extremely large values).

C. *Comparison models*

Historical Average (HA) and ARIMA [7] are used as simple statistical comparison models. HA is the average of the traffic at the same time period across all days in the training set. We implement, train, and evaluate the ARIMA model using the statsmodels Python package without GPU acceleration. We use grid search to iteratively explore different combinations of the known ARIMA parameters (p, d, q). The learning of ARIMA is based on the exact maximum likelihood via Kalman filter.

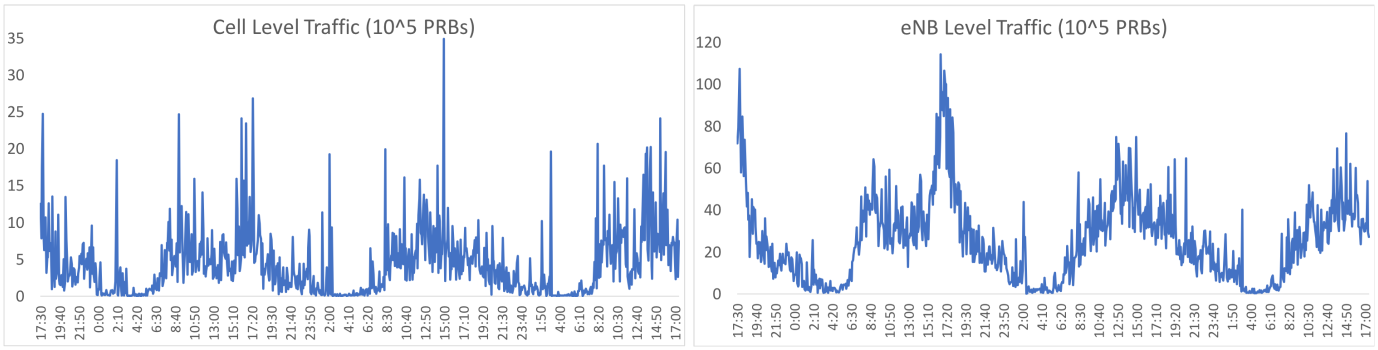

Fig. 5. Typical cell level and eNB level traffic in 10^5 PRBs over 3 days

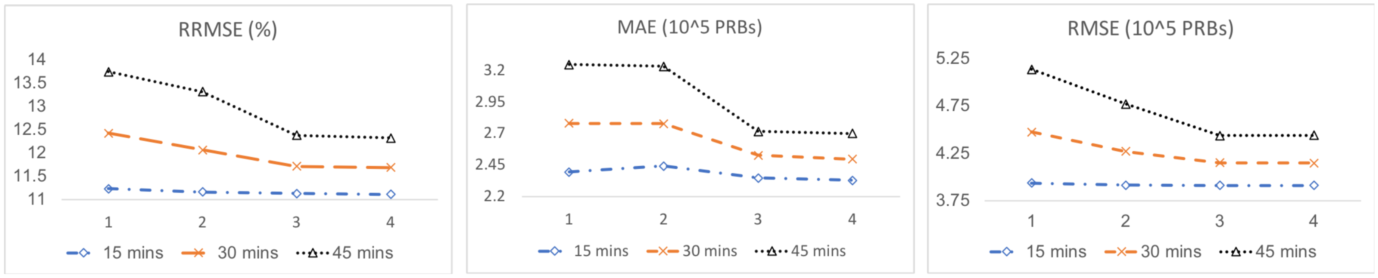

Fig. 6. Performance comparison of STGCN-HO using different numbers of ST-blocks at cell level. The X axes represent the numbers of ST-blocks, and the Y axes represent RRMSE, MAE, and RMSE, respectively.

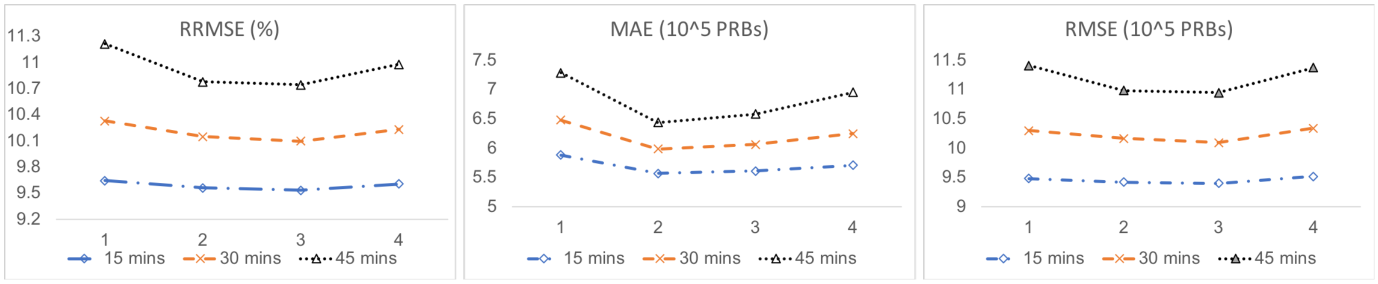

Fig. 7. Performance comparison of STGCN-HO using different kernel sizes at eNB level. The X axes represent the kernel sizes, and the Y axes represent RRMSE, MAE, and RMSE, respectively.

Because of the popularity and capability of LSTM models to learn non-linear patterns and remember information for long time periods, we use two LSTM models for comparison: (i) a vanilla LSTM [3], and (ii) a state-of-art multi-task LSTM [12], which is composed of dedicated LSTMs for each cell of interest and a shared LSTM for the neighboring cells of the current cell. Hence, both the similarity and difference between cells could be generated and exploited to improve the performance. We choose these models for comparison also because they are not limited to grid level, and they can incorporate the handover information easily by feeding the shared LSTM with the traffic of cells having the number of handovers larger than a threshold. In our case, we choose 1,000 as the threshold for both cell level and eNB level; most of cells/eNBs have about six neighbors with the number of handovers over 1,000.

In our dataset, we observe cells that share the exact same location. Since CNN-grid cannot separate these cells, it cannot be applied as a baseline.

To further demonstrate the contribution of the handover graph structure and illustrate the separation of the spatial effects from the temporal effects, we create another baseline, a stacked GLU model, which removes the GCN layers from our STGCN-HO model, and keep only the GLUs to model temporal information.

D. Settings

The experiments are conducted on a Ubuntu Linux cluster (Intel(R) Xeon(R) CPU E5-2680 v4 @ 2.40GHz with 512GB memory, 4 NVIDIA P100-SXM2 GPUs with 64GB total memory). All experiments use 60 minutes as the historical time window, i.e., 12 observed data points, to predict cellular traffic volume in the next 15, 30, and 45 minutes, which are typical time windows necessary to facilitate resource allocation

TABLE I
PERFORMANCE COMPARISON OF TRAFFIC PREDICTION AT CELL LEVEL FOR DIFFERENT MODELS

Model	15 mins / 30 mins / 45mins		
	RMSE(10^5 PRBs)	MAE(10^5 PRBs)	RRMSE (%)
HA	7.06	4.64	17.49
ARIMA	5.71/6.98/8.26	3.31/4.11/4.91	16.38/19.95/23.59
Vanilla LSTM	4.43/4.60/4.81	2.79/2.93/3.05	12.46/12.95/13.36
Multi-task LSTM	4.17/4.55/4.88	2.57/2.85/3.09	11.76/12.71/13.53
Stacked GLU	3.94/4.46/5.22	2.36/2.80/3.40	11.21/12.56/14.58
STGCN-HO	**3.91/4.15/4.44**	**2.33/2.50/2.70**	**11.11/11.69/12.32**

TABLE II
PERFORMANCE COMPARISON OF TRAFFIC PREDICTION AT ENB LEVEL FOR DIFFERENT MODELS

Model	15 mins / 30 mins / 45mins		
	RMSE (10^5 PRBs)	MAE (10^5 PRBs)	RRMSE (%)
HA	21.68	14.20	17.88
ARIMA	13.46/16.61/19.85	7.90/9.87/11.86	13.82/16.87/20.00
Vanilla LSTM	11.36/12.92/13.86	7.23/8.12/8.45	11.28/12.28/12.31
Multi-task LSTM	10.50/11.79/12.99	6.40/7.36/8.21	10.53/11.76/12.85
Stacked GLU	10.00/12.08/14.89	6.07/7.47/9.34	9.95/11.60/13.54
STGCN-HO	**9.40/10.09/10.95**	**5.60/6.05/6.57**	**9.54/10.09/10.73**

and network management. The ratio of training, validation, and testing datasets is 4:1:1. In other words, the first 4 weeks are used for training, the next for validation, and the last for testing. We choose a batch size of 100 for our model, and we notice sufficient convergence within 20 learning epochs.

E. Results

1) Parameter sensitivity of STGCN-HO: To evaluate the parameter sensitivity of our model, we run STGCN-HO with different numbers of ST-blocks and kernel sizes for both cells and eNBs. Since the results are similar, we show one graph for a cell and one graph for an eNB. Figure 6 illustrates the effect of different number of ST-blocks on the performance of STGCN-HO. In general, the performance improves as the number of ST-blocks increases because deeper networks are more capable to learn intricate patterns. When the number ST-blocks reaches 4, we see little further improvement, which indicates 4 ST-blocks is sufficient to capture the complex spatio-temporal correlation of cellular traffic. Figure 7 illustrates the effect of GLU kernel size on the performance of STGCN-HO. In most cases, the performance of STGCN-HO peaks when the kernel size is 3. Therefore, for the rest of the experiments, we will use 4 ST-Blocks and a kernel of size 3.

2) Comparison with other models: Table I shows the prediction accuracy results for STGCN-HO and the comparison models at cell-level. Our model achieves the best performance, and the improvement is statistically significant in all cells/eNBs (one-tailed T-test p value < 0.01). We observe that the superiority of STGCN-HO increases over time. The RRMSE of STGCN-HO is 5.53%, 8.03%, and 8.94% better for 15 mins, 30 mins, and 45 mins, respectively when compared with the best comparison model multi-task LSTM.

Table II shows the prediction accuracy results for STGCN-HO and the comparison models at base station-level, i.e., eNB-level. Similarly, we observe that the RRMSE of STGCN-HO is 9.40%, 14.2%, and 16.5% better for 15 mins, 30 mins, and 45 mins, respectively when compared with multi-task LSTM.

Compared with stacked GLU (i.e., model without GCN layers), STGCN-HO is 0.892%, 6.93%, 15.5% better for 15 mins, 30 mins, and 45 mins, respectively, at cell-level, and 4.12%, 13.0%, 20.8% better at eNB-level. This demonstrates the benefit of adding graph convolutional layers to capture the handover information, which is especially beneficial for longer prediction intervals. For such predictions, deep learning time series models suffer from the fact that the prediction error accumulates over time. While our model has the same artifact,

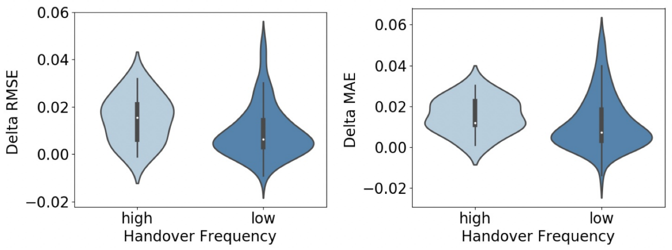

Fig. 8. Performance comparison between STGCN-HO and vanilla LSTM for two cell handover frequency categories, RMSE (left) and MAE (right)

its accumulated error is lower than those of the baseline models. Thus, its prediction is better for longer intervals.

The results in the two tables also show that the prediction performance is better at the base station-level than at the cell-level. The average RRMSE for all base stations is 10.9% lower than the average RRMSE at the cell-level. The main reason is that at an aggregate level eNBs have smoother traffic fluctuation than cells, as shown in Figure 5. ENBs can absorb some sharp changes through aggregation, while the cells cannot. The performance improvement can also be explained by the fact that the handover at eNB-level is driven more by user mobility than by load balancing policy at cell level, which makes the traffic more predictable at eNB-level.

For additional insights into the effect of handover information on prediction accuracy, Figure 8 shows two violin plots for delta RMSE and delta MAE of STGCN-HO and vanilla LSTM. We use vanilla LSTM for comparison because it does not use handover information, and our goal is to understand the effect of using handover information; multi-task LSTM is adapted by us to use handover information. We observe two clear categories of cells below and above 50,000. The x-axis is the binarized handover frequency, with "high" defined as over 50,000, and "low" as under 50,000. Delta RMSE is the difference between RMSE scores of the min_max scaled STGCN-HO and vanilla LSTM to eliminate the effect of traffic volume scale. Delta MAE is defined similarly. The results show STGCN-HO works better for cells with high handover frequency, which further demonstrates that handover information improves cellular traffic prediction.

3) Training Time Comparison: Table III shows the 10 epochs training time of STGCN-HO and vanilla LSTM. Multitask LSTM is much more complex than vanilla LSTM, and its training takes longer. The results show that our model is approximately one order of magnitude faster in training than

TABLE III
TRAINING TIME COMPARISON BETWEEN STGCN-HO AND VANILLA LSTM

Model	10 epochs training time	
	eNB level (seconds)	cell level (seconds)
Vanilla LSTM	662.600	2817.777
STGCN-HO	**97.480**	**239.668**

LSTM. This is because LSTM models are trained and perform prediction on each node individually, and the hidden states of LSTM must be processed sequentially at each node. STGCN-HO, on the other hand, is trained and performs prediction over the entire network at once. The results also show that STGCN-HO scales better than LSTM with the data size, as its improvement in the training time is higher for the cell-level (89 nodes) than for the base station-level (26 nodes).

VII. Conclusion and Future Work

This paper has proposed STGCN-HO, a novel cellular traffic prediction model that utilizes the handover graph within a stacked residual deep learning framework to capture spatio-temporal information as well as auxiliary features. Experiments show our model outperforms state-of-the-art methods on a real-world dataset provided by a major network operator. We also demonstrated that the utilization of the handover graph decreases the prediction error accumulated over time and leads to more accurate predictions over longer time intervals. Furthermore, unlike RNN, STGCN-HO is fast to train because it uses CNN, and is capable to train and predict all cells or base stations at the same time. Unlike CNN-grid, STGCN-HO can make predictions not only for base stations, but also for cells within base stations. While this model has been developed for LTE data, as future work, we plan to leverage its main ideas to build traffic prediction models for 5G networks and integrate the model into a RAN controller to facilitate resource allocation and RAN management.

References

[1] L. Chen, D. Yang, D. Zhang, C. Wang, J. Li, and T.-M.-T. Nguyen. Deep mobile traffic forecast and complementary base station clustering for c-ran optimization. *Journal of Network and Computer Applications*, 121:59–69, 2018.

[2] Xiaming Chen, Yaohui Jin, Siwei Qiang, Weisheng Hu, and Kaida Jiang. Analyzing and modeling spatio-temporal dependence of cellular traffic at city scale. In *2015 IEEE International Conference on Communications (ICC)*, pages 3585–3591. IEEE, 2015.

[3] Anestis Dalgkitsis, Malamati Louta, and George T Karetsos. Traffic forecasting in cellular networks using the lstm rnn. In *Proceedings of the 22nd Pan-Hellenic Conference on Informatics*, pages 28–33. ACM, 2018.

[4] Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. *CoRR*, abs/1612.08083, 2016.

[5] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, pages 3844–3852, 2016.

[6] Q. Duan, X. Wei, Y. Gao, and F. Zhou. Base station traffic prediction based on stl-lstm networks. In *2018 24th Asia-Pacific Conference on Communications, APCC 2018*, pages 407–412, 2019.

[7] Huifang Feng and Yantai Shu. Study on network traffic prediction techniques. In *Proceedings. 2005 International Conference on Wireless Communications, Networking and Mobile Computing, 2005.*, volume 2, pages 1041–1044. IEEE, 2005.

[8] J. Feng, X. Chen, R. Gao, M. Zeng, and Y. Li. Deeptp: An end-to-end neural network for mobile cellular traffic prediction. *IEEE Network*, 32(6):108–115, November 2018.

[9] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1243–1252. JMLR. org, 2017.

[10] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[11] L. Piazzi and H. L. Bertoni. Effect of terrain on path loss in urban environments for wireless applications. *IEEE Transactions on Antennas and Propagation*, 46(8):1138–1147, Aug 1998.

[12] C. Qiu, Y. Zhang, Z. Feng, P. Zhang, and S. Cui. Spatio-temporal wireless traffic prediction with recurrent neural network. *IEEE Wireless Communications Letters*, 7(4):554–557, Aug 2018.

[13] RC Ris, LH Holthuijsen, and N Booij. A third-generation wave model for coastal regions: 2. verification. *Journal of Geophysical Research: Oceans*, 104(C4):7667–7681, 1999.

[14] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *arXiv preprint arXiv:1211.0053*, 2012.

[15] J. Wang, J. Tang, Z. Xu, Y. Wang, G. Xue, X. Zhang, and D. Yang. Spatiotemporal modeling and prediction in cellular networks: A big data enabled deep learning approach. In *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, 2017.

[16] S. Wang, X. Zhang, J. Zhang, J. Feng, W. Wang, and K. Xin. An approach for spatial-temporal traffic modeling in mobile cellular networks. In *2015 27th International Teletraffic Congress*, pages 203–209, Sep. 2015.

[17] Xu Wang, Zimu Zhou, Fu Xiao, Kai Xing, Zheng Yang, Yunhao Liu, and Chunyi Peng. Spatio-temporal analysis and prediction of cellular traffic in metropolis. *IEEE Transactions on Mobile Computing*, 2018.

[18] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. A comprehensive survey on graph neural networks. *arXiv preprint arXiv:1901.00596*, 2019.

[19] F. Xu, Y. Lin, J. Huang, D. Wu, H. Shi, J. Song, and Y. Li. Big data driven mobile traffic understanding and forecasting: A time series approach. *IEEE Transactions on Services Computing*, 9(5):796–805, Sep. 2016.

[20] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[21] B. Yu, H. Yin, and Z. Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In *IJCAI International Joint Conference on Artificial Intelligence*, volume 2018-July, pages 3634–3640, 2018.

[22] C. Zhang, H. Zhang, D. Yuan, and M. Zhang. Citywide cellular traffic prediction based on densely connected convolutional neural networks. *IEEE Communications Letters*, 22(8):1656–1659, Aug 2018.

[23] Chaoyun Zhang and Paul Patras. Long-term mobile traffic forecasting using deep spatio-temporal neural networks. In *Proceedings of the Eighteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 231–240. ACM, 2018.

[24] Chaoyun Zhang, Paul Patras, and Hamed Haddadi. Deep learning in mobile and wireless networking: A survey. *IEEE Communications Surveys Tutorials*, PP, 03 2018.

[25] K. Zhang, G. Chuai, W. Gao, X. Liu, S. Maimaiti, and Z. Si. A new method for traffic forecasting in urban wireless communication network. *Eurasip Journal on Wireless Communications and Networking*, 2019(1), 2019.

[26] X. Zhao, K. Yang, Q. Chen, D. Peng, H. Jiang, X. Xu, and X. Shuang. Deep learning based mobile data offloading in mobile edge computing systems. *Future Generation Computer Systems*, 99:346–355, 2019.