STEP: A Spatio-Temporal Fine-Granular User Traffic Prediction System for Cellular Networks

Lixing Yu, *Student Member, IEEE,* Ming Li, *Member, IEEE,* Wenqiang Jin, *Student Member, IEEE,* Yifan Guo, *Student Member, IEEE,* Qianlong Wang, *Student Member, IEEE,* Feng Yan, *Member, IEEE,* and Pan Li, *Member, IEEE*

Abstract—While traffic modeling and prediction are at the heart of providing high-quality telecommunication services in cellular networks and attract much attention, they have been approved as an extremely challenging task. Due to the diverse network demand of Internet-based apps, the cellular traffic from an individual user can have a wide dynamic range. Most existing methods, on the other hand, model traffic patterns as probabilistic distributions or stochastic processes and impose stringent assumptions over these models. Such assumptions may be beneficial at providing closed-form formula in evaluating prediction performances, but fall short for practice use. In this paper we propose STEP, a **s**patio-**te**mporal fine-granular user traffic **p**rediction mechanism for cellular networks. A deep graph convolution network, called GCGRN, is constructed. It is a novel combination of the graph convolution network (GCN) and gated recurrent units (GRU), which exploits graph neural network to learn an efficient spatio-temporal model from a user's massive dataset for traffic prediction. The prototype of STEP has been implemented. Extensive experimental results demonstrate that our model outperforms the state-of-the-art time-series based approaches. Besides, STEP merely incurs mild energy consumption, communication overhead and system resource occupancy to mobile devices. Moreover, NS-3 based simulations validate the efficacy of STEP in reducing session dropping ratio in cellular networks.

Index Terms—Fine-granular traffic prediction, deep graph convolution network, cellular networks.

1 Introduction

With billions of mobile devices accessing the Internet via 3G/4G/5G networks, cellular traffic has skyrocketed in the past a few years. It is predicted that over 50% of the global devices and connections will be mobile and the monthly global mobile data traffic will surpass 30.6 exabytes (10^{18}) by 2020 [1]. This trend will continue in the foreseeable future. To accommodate the ever-increasing large volume of data in cellular networks, small cells, including micro-cells, femtocells, and picocells, have been widely deployed because of their capability of growing network capacity especially in populated urban areas that cannot be sustained by conventional macrocells. Despite the nice property, this fashion also raises critical concerns on frequent handoffs a moving user can experience even at low to moderate velocities. To validate this statement, we show in Fig. 1 the distribution of duration that users stay connected to one BS. It is a statistical result from our collected dataset on individual user cellular usage (please refer to Section 3.2 for more details). We observe that 50% of the duration is shorter than 12.7s, which means handoffs take place every 12.7s or even less time. Apparently, the shorter a user stays in a cell, the more frequent it switches between BSs.

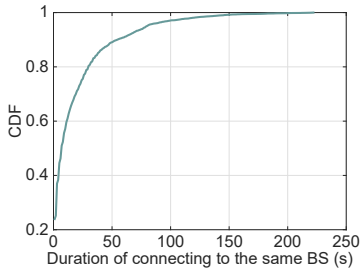


Fig. 1. Cumulative distribution function (CDF) of time duration that users stay connected to one BS.

On the other hand, the current handoff procedure for cellular networks mainly follows the first release of Long-Term Evolution (LTE) standard (Rel-8) where decisions are made according to realtime signal strength measurement from neighboring BSs. As a result, the frequent handoffs will inevitably lead to increased call dropping probability (CDP) due to the latency caused by multi-round interactions between handoff BSs [2], [3], [4], [5]. Moreover, in 5G and beyond, BSs are foreseen to operate over a set of drastically varying frequencies, e.g., from 450 MHz to 50 GHz. Without further assistance, quick switching between shortwave and millimeter wave may be possible in theory but challenging to light-weight radios. Due to the above reasons, the reliability of conventional handoff procedure in handling high-mobility low-latency cellular services, such as communications for self-driving vehicles, in 5G and beyond will be deteriorated.

To address this issue, intelligent schemes, such as proactive cell selection [6], [7], [8], [9] and spectrum reservation

- *Lixing Yu, Yifan Guo, Qianlong Wang, and Pan Li are with the Department of Electrical Engineering and Computer Science, Case Western Reserve University, OH.*
- *Ming Li and Wenqiang Jin are with Computer Science and Engineering Department, The University of Texas at Arlington, TX.*
- *Feng Yan is with Computer Science and Engineering Department, University of Nevada Reno, NV.*

[10], [11], [12], have been proposed to assist handoffs. They facilitate carriers to agilely allocate transmission resources to individual users. For instance, the forecasted accessing BSs can reserve certain amount of resource blocks (RBs) tightly based on users' future traffic demands, a few seconds ahead of their arrivals. In this way, the CDP can be largely reduced by avoiding transmission resource deficiency at target BSs. (Further discussion is provided in Section 6.) Besides, since a user knows which BS it is going to access next and thus its operating frequency range, it can prepare for the proper antenna hardware/software stack ahead of time for later transmission.

The efficacy of these schemes heavily relies on accurate prediction on user traffic and its mobility, which, however, is not an easy task. First, because of the diverse demands of Internet-based apps (e.g., mobile videos, online games, VoIP), a user's cellular traffic can be of a wide dynamic range. Second, in addition to the data rate of each app, fine-granular traffic prediction discussed in this work also aims to associate it with user mobility, i.e., which BSs an online app is going to route through under a time resolution manner. Uncertain human behaviors (e.g., at work, in transit, during sleep) introduces another dimension of uncertainty in prediction. These factors often have a complex correlation and can change over time. Consequently, modeling them is prohibitively difficult, let along exploring them for prediction. While there have been substantial prior works on these topics [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33], [34], they have to impose stringent assumptions on how the above-mentioned factors behave, interact with each other, and affect the prediction. Traffic patterns are modeled as probabilistic distributions or stochastic processes. They may be effective in providing expressive formulas but fall short for practice use. More importantly, existing research mainly focuses on large-scale prediction, e.g., forecasting the future traffic load or the number of subscribers for a city, an urban area, or a cell. Such a prediction is too coarse to be applied to assist handoffs for a particular user.

Under these observations, in this work we propose STEP, a **s**patio-**te**mporal fine-granular user traffic **p**rediction system for cellular networks. Its main objective is to assist carrier to reserve proper amount of RBs for individual users at BSs that they will access soon. The prediction-empowered resource reservation then helps to alleviate the performance degradation caused by frequent handoffs among densely deployed cells. STEP predicts user's time-resolved mobility and traffic for each online app in a realtime manner. For this purpose, we first leverage graphical models to represent user's traffic and mobility patterns in both temporal and spatial domains. To extract latent features in these two domains, a deep graph convolution network, called GCGRN, is developed. It is a novel combination of the graph convolution network (GCN) and gated recurrent units (GRU). The user's historical record is then fed into GCGRN for training, which later forecasts this user's future data rate for each of its online apps and the set of BSs the user is going to access. Instead of coarse large-scale datasets collected by network administrators, our fine-grained prediction relies on data recorded by individual users locally. The contributions of this work are summarized as follows.

- We carefully investigate the characteristics of individual cellular traffic with daily cellular usage collected from 10 volunteers. We demonstrate that there are strong spatio-temporal patterns in individual user cellular traffic, which was not well exploited in previous research.
- We develop a graph convolution network, GCGRN, to exploit features in spatial and temporal domains for fine-granular traffic prediction.
- To assess prediction performances, the prototype of STEP is implemented. Extensive in-field experiments are conducted, examining STEP from the aspects of energy consumption, communication overhead, and system utility.
- NS-3 based simulations are also conducted to investigate the impact of prediction-empowered resource reservation to cellular network performances.

In the rest of the paper, related works are reviewed in Section 2. System architecture and overview are given in Section 3. Spatial-temporal dependency of user traffic and the proposed GCGRN are discussed in Section 4. Extensive experimental results are provided in Section 5, followed by NS-3 based simulation results in Section 6. We conclude the paper in Section 7.

2 Related Works

2.1 Traffic Prediction

Traffic prediction is well-known to facilitate resource allocation and enable intelligent cellular networks. Extensive efforts have been devoted to this topic [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33], [34]. It is generally treated as a time series analysis problem, the performance of which depends on the statistical or stochastic models, such as Lognorm distribution model [34] and Wiener Process [33]. In fact, the pattern of cellular traffic is much more complicated due to various factors, e.g., user mobility and diverse user requirements. Therefore, ideal analytical models can hardly capture complex and nonlinear hidden spatio-temporal dependency in wireless traffic data. Recently, advances in deep learning models have emerged as strong approaches for traffic prediction [35], [36], [37], [38]. For example, Qiu et al. [36] applied recurrent neural network (RNN) to learn and predict cellular traffic. A citywide cellular traffic prediction scheme [37] is proposed based on convolutional neural network (CNN). In the same line of research, Wang et al. [38] resorted to the graph neural network (GNN) to characterize urban cellular traffic. While RNN is good at learning time domain features, it is incompetent in exploring spatial dependencies. Besides, CNN and GNN are originally designed for computer vision or graphical tasks and may have poor performances over sequential data prediction. More importantly, the existing research mainly focuses on large-scale wireless traffic prediction, e.g., forecasting the future traffic load for a city, an urban area, or a cell. Instead, this work studies individual user traffic prediction at a fine granularity. Besides, the dataset that existing works rely on is collected by the network administrator. As we point out later, such a dataset is too vague for fine-granular user-level traffic prediction.

TABLE 1
An illustration of raw data in csv file.

Time	Cell tower ID	GPS coordinate	Overall Tx rate	Overall Rx rate	Youtube Tx rate	Youtube Rx rate	···
March 14, 09:43:50	173113097	37.79419728; -122.26488491	12.75 kbps	365.19 kbps	4.36 kbps	312.33 kbps	···
March 14, 09:43:51	173116680	37.79419157; -122.26482702	25.92 kbps	223.56 kbps	12.3 kbps	201.04 kbps	···
⋮			⋮			⋮	

2.2 Mobility Prediction

Mobility prediction rests on the notion that mobility patterns of mobile users are, to a certain extent, predictable as verified by literature [39], [40]. Some existing studies, such as [13], [14], [15] leverage Discrete Time Markov Chains (DTMC) for mobility modeling and prediction. However, DTMC assumes human mobility is memoryless. Thus, most works apply DTMC only for spatial prediction, i.e., the next cell a user is going to head to. The same limitation also exists in the Continuous Time Markov Chain (CTMC) based mobility prediction [16], [17]. Taking into account the memory property exhibiting in wireless traffic, Semi-Markov model is later adopted, allowing for arbitrarily distributed sojourn times. The works [18], [19], [20], [21] fall into this category. There is also some research [22], [23] applying low-order Markov predictor for mobility prediction. Aside from Markovian models, Nadembega et al. [24] proposed a mobility prediction scheme on estimating the time window that a user stays in one cell along his path. Nonetheless, they assume that the path is known in advance.

Like traffic prediction, most existing works on mobility prediction rely on statistic or stochastic models, which typically impose certain impractical assumptions over user mobility patterns. Besides, none of the above works provides time-resolved mobility prediction as we do. They can only answer questions, such as which cell a user is heading to, or how long a user will stay in a cell. Instead, we endeavor to find out which BS a user is associated with at a given future specific time instance. Therefore, we are targeting a more challenging task.

3 The System Architecture

3.1 Overall Architecture of STEP

STEP consists of two modules: data collection and traffic forecasting. Due to the lack of computing and storage resources at mobile terminals, the traffic forecasting module is implemented at the core network of cellular systems, while the realtime data collection is conducted at devices locally. In particular, data collection is only triggered when a device is using cellular services and is thus off most of the time. It is worth mentioning that existing learning-based large-scale traffic prediction mechanisms [35], [36], [37], [38] rely on traffic statistics recorded by the network administrator. The reasons that we do not utilize the existing dataset but collect our own are as follows. First, these traffic statistics are relatively coarse, e.g., aggregated traffic volume or number of associated devices at each BS. They are of less use for fine-granular user-level traffic prediction. More importantly, network administrators can only have vague visibility over their subscribers' traffic. Previously, an administrator typically looked into HTTP User-Agent fields for app names to identify a user's online app usage. Nowadays, such an approach becomes less effective due to the wide adoption of HTTPs. User's most online data connections have been encrypted and hidden from the administrator.

3.2 Dataset Description

3.2.1 Data Collection

To facilitate data collection, a specialized app is developed and implemented in smartphones. The app runs as a background service and records phone's traffic statistics and geolocations. Table 1 shows the structure of our dataset. The data is sorted by time sequence. Each entry consists of the phone's time instance, the connected BS ID, GPS coordinate, overall cellular Tx/Rx data rate, and cellular Tx/Rx data rate for each online app. This record will serve as both the training and testing set. The fine granularity guarantees the accuracy of traffic prediction. The data collection process spans a period of one month (from 03/01/2019 to 04/01/2019) with a major cellular carrier in the US. Ten volunteers have participated. Half of them live and work in the bay area, while the other half are in a mid-sized city of the US. As for the means of commute, three of the volunteers take the Bay Area Rapid Transit (BART), five of them drive, and the last two walk. In total, we collected more than 10^6 data samples involving 1564 BSs and 12 online apps after deleting erroneous samples.

The sampling rate here is 1 sample per second, i.e., 1Hz. To our best knowledge, 1Hz is the highest sampling rate available in most commercial smartphones. While some GPS chips provide sampling frequency up to 5-10Hz, they are not equipped to current smartphones yet. A higher sampling rate produces more fine-granular real-time readings of user traffic and thus potentially produces a more accurate prediction. On the other hand, while the traffic data is sampled at 1Hz, this module is active only when a phone is transmitting/receiving via cellular connections. Thus, no data is collected when the phone is in sleep mode or connecting with WiFi. Besides, it does not mean that the collected samples should be uploaded at such a high frequency too. As discussed later, the collected data actually can be uploaded in a much lower frequency.

3.2.2 Data Augmentation

Realizing that deep learning usually benefits from large training sets [41], the original dataset is then compensated by the addition of synthetic data that is generated via data augmentation. The commonly used slicing window

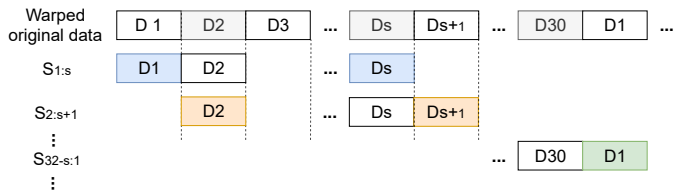


Fig. 2. Illustration of slicing window based data augmentation.

technique is adopted. It was originally introduced for deep CNNs in [42], and then employed to enlarge sample space in computer vision tasks and time-series data processing [43], [44], [45].

Consider a volunteer's data series collected in 30 days $\boldsymbol{Y} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_{30}\}$, where $\boldsymbol{y}_i$ represents the data vector for day i. A slice is a snippet of the original time series, defined as $S_{i:j} = \{\boldsymbol{y}_i, \boldsymbol{y}_{i+1}, \cdots, \boldsymbol{y}_j\}$, $1 \leq i \leq j \leq 30$. Suppose the length of the slice is s. The slicing operation will be performed over a warping data and generate a set of sliced time series: $Slicing(\boldsymbol{Y}, s) = \{S_{1:s}, S_{2:s+1}, \cdots, S_{30-s+1,30}, S_{30-s+2,1}, \cdots S_{30:s-1}\}$. Each element (slice) in $Slicing(\boldsymbol{Y}, s)$ is treated as a data set collected from a virtual user. We illustrate the entire process in Fig. 2. Note that the slicing unit is a single day so as to preserve the day-based temporal dependency in the original dataset. The augmented dataset can be viewed constituting of data from 150 users, including 10 volunteers and 140 virtual users, involving approximate 1.6×10^7 data samples.

As pointed out by [44], both window slicing, the one adopted here, and window warping are widely used data augmentation techniques for time series. While first inspired from computer vision community, with its idea of extracting slices from time series to construct synthetic dataset, window slicing has been investigated for augmenting training dataset for time series for quite some time [42], [44], [46]. As studied in [44], window slicing and window warping data augmentation can both help improve time series classification performance using CNN. Nonetheless, it is hard to tell which one beats the other. On one hand, warping window results in a lower error rate in classification than window slicing. On the other hand, the former causes much higher error rate dispersion than the latter. Besides, the user's traffic data exhibits clear daily-periodic pattern on weekdays, as demonstrated in Fig. 3. Thus, it is convenient to set one-day duration as the size of window for slicing in our case. In regard to window warping, while it does not require any clear time-domain pattern in the original time series for data augmentation, its computation complexity is higher due to, for example, identifying the minimum warping path between two datasets.

As a note, instead of sharing a common model, the proposed GCGRN is customized for each user, i.e., the parameters in the neural network are distinct from each other. Thus, a user's GCGRN is trained solely based on her own historical data and irrelevant to her peers. In this sense, it is more critical to collect a user's data for a long time duration for the training purpose. In the experiment, the data collection lasts for one month, which is deemed sufficient for the proposed GCGRN to extract daily-patterns from the traffic usage. The main reason for collecting data from multiple volunteers, 10 in this paper, and creating 140 more synthetic users is not for training a single model. Instead, it is for validating the effectiveness of GCGRN in traffic prediction across different users with diverse service usage pattern.

3.3 Privacy Considerations

According to our design, telecommunication carriers, e.g., AT&T and Verizon, acquire mobile user's traffic statistics and geolocations for traffic prediction. These data are classified as customer proprietary network information (CPNI) by FCC 16-148 [47]. This regulation recognizes three categories of approval with respect to use of customer CPNI obtained by providing telecommunications services: opt-in approval for the use of sensitive CPNI, opt-out approval for the use of non-sensitive one, and congressionally-recognized exceptions to approval for the use of CPNI that is essential for providing broadband services. Apparently, our case falls into the first category, opt-in approval. In a word, STEP is designed as an add-on function for cellular communication systems. Its installment and launching in a personal mobile device should receive affirmative permission from the user.

4 Spatio-Temporal Modeling for User Traffic Prediction

4.1 Spatio-Temporal Dependency of User Traffic

We first demonstrate the spatio-temporal dependency of user traffic, which lays a foundation for developing spatio-temporal deep learning techniques in traffic forecasting.

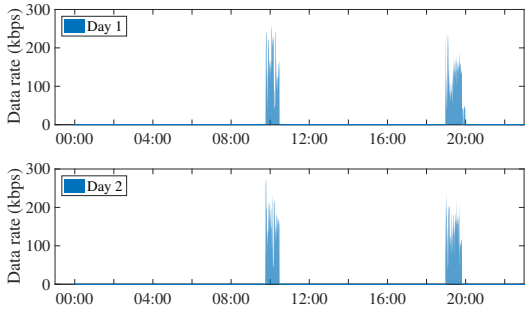
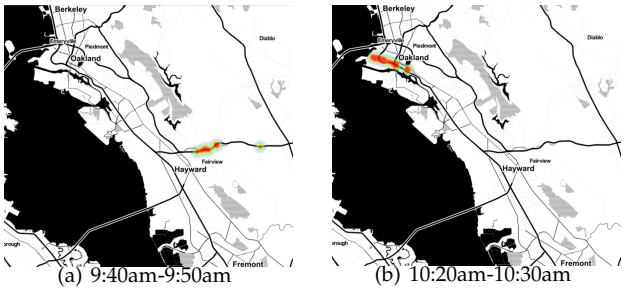


Fig. 3. Time-domain distribution of a randomly selected volunteer's cellular traffic.

Fig. 3 shows the time-domain traffic characteristics from an arbitrary volunteer. We randomly select two working days for comparison. It is observed that the volunteer's traffic on different days exhibit quite similar patterns. Cellular traffic exists in relatively constant time durations throughout a day. Besides, the traffic demonstrates a similar shape in these two days. This is because a user has a relatively stable app usage habit, and each app has similar data rate patterns. STEP should be capable of capturing such properties from the historical record and leveraging them for prediction.

Fig. 4 shows the spatial distribution of a volunteer's traffic at a given fixed time duration. It is a statistic result from his mobility observed for the entire month. This volunteer takes BART from home to work every morning during weekdays. Thus, his cellular traffic presents in the



(a) 9:40am-9:50am (b) 10:20am-10:30am

Fig. 4. Heat map (aggregated over one month) of a randomly selected volunteer's mobility at a fixed time duration of a day.

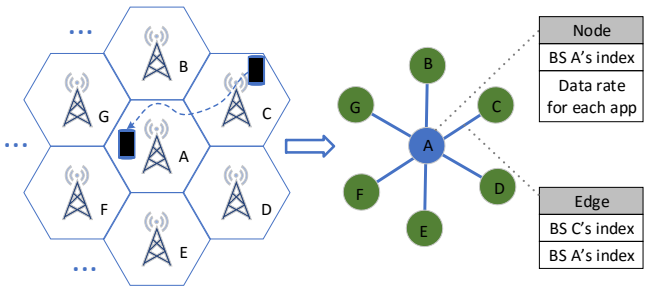


Fig. 5. Part of graph representation of a particular user.

same section of his commute route. As shown in Fig. 4(a), he consistently shows up between Dublin/Pleasanton station and Castro Valley station at 9:40am-9:50am every day. This is because BART is scheduled to arrive at these stations following a fixed timetable daily. We have a similar observation over this volunteer's mobility in other time slots (e.g., Fig. 4(b)), as well as other volunteers' mobility. Clearly, a stronger spatio-temporal correlation in one's traffic will benefit accurate prediction.

4.2 Graph Representation of Spatio-Temporal Dependency

We propose to utilize time-dependent graphs to model the spatio-temporal dependency of user traffic. For a target user, denote by $\mathcal{V}$ the set of all BSs that the user has accessed before. Two BSs, i.e., nodes, are considered as neighbors if the user was observed handed over between them at least once in history. There is a directed edge connecting two neighbors, indicating the direction of the handoff. We then denote by $\mathcal{E}$ the entire set of directed edges. A directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is constructed for each user. Both nodes and edges have associated time-variant features. Given a time instance t, there is only one active node that the user is accessing to. Its features include its index and uplink/downlink traffic for each online app it carries for the user. The features of other nodes at t are all empty sets. If a handoff is observed over an edge at t, the edge features include the source and destination for this handoff. Similarly, features at other edges are empty sets.

Let $\boldsymbol{Y}(t)$ be the information observed over $\mathcal{G}(t) = \{\mathcal{V}(t), \mathcal{E}(t)\}$. The traffic prediction problem aims to learn a function $h(\cdot)$ that maps T' graph information to future T graph information, given a series of time-variant graph $\mathcal{G}(t)$'s

$$[\boldsymbol{Y}(t-T'+1), \cdots, \boldsymbol{Y}(t); \mathcal{G}(t-T'+1), \cdots, \mathcal{G}(t)] \xrightarrow{h(\cdot)} [\boldsymbol{Y}(t+t_0), \cdots, \boldsymbol{Y}(t+t_0+T)]. \quad (1)$$

t_0 is the prediction length, which is defined as the number of time slots ahead of current time instance t. Theoretically, t_0 can range from several seconds to even dozens of hours. For different t_0's, the parameters (fixed through training) in a prediction model will vary as well.

4.3 Overview of GCGRN

To effectively learn spatio-temporal dependency of user traffic, we construct a deep graph network, called graphic convolution gated recurrent unit network (GCGRN). It is a novel combination of graphic convolution network (GCN) [48] and gated recurrent unit (GRU) [49]. Since user cellular traffic is presented as graph-structured data, we want to perform learning and prediction that take graphs as inputs. Feasible approaches include GCN [48], [50], [51], [52], GNN [53], [54], [55] and spectral networks [56]. They are superior to conventional CNN based approaches due to their capability in handling non-Euclidean characteristics of the complex structure of graphs, while CNN was originally designed to process rigid grid-structured pictures. Besides, we choose GCN over GNN, because GNN is more computationally intensive compared to the convolutional approaches, due to the complexities of ensuring convergence or running backpropagation through time. As a contribution of this work, the proposed GCGRN extends the application of GCN to time sequence data, which has rarely been investigated before. A unit of GCGRN is given in Fig. 6. Specifically, GRU is employed to capture temporal patterns exhibited in user traffic. Thus, GCGRN allows to jointly learn user traffic patterns among neighboring cells (in spatial domain) and accumulate such patterns across a long-time periods (in temporal domain).

4.4 Design Details

Convolution Operator. We thus model the spatial dependency by relating traffic flow to a convolution process. Given a graph signal $\boldsymbol{Y} \in \mathbb{R}^{N \times P}$ that consists of features of both nodes and edges, the convolution operation over $\boldsymbol{Y}$ and a filter $f_{\boldsymbol{\theta}}$ is defined as $\boldsymbol{Y}_{:,p} *_{\mathcal{G}} f_{\boldsymbol{\theta}} = \sum_{k=1}^{K} \theta_k L^k \boldsymbol{Y}_{:,p}, p \in \{1, \cdots, P\}$ where P stands for the number of input features, N is the cardinality of the entire set of nodes and edges in $\mathcal{G}$, k is the diffusion step, and $\boldsymbol{\theta} \in \mathbb{R}^{N \times K}$ are parameters for the filter. This expression is the spectral graph convolution utilizing the concept of normalized graph Laplacian $L = D^{-\frac{1}{2}}(D - W)D^{-\frac{1}{2}}$ [55], with $D \in \mathbb{R}^{N \times N}$ as the diagonal degree matrix. By this definition, a graph signal $\boldsymbol{Y}$ is filtered by a kernel $\boldsymbol{\theta}$. In general, computing the convolution can be expensive. However, since $\mathcal{G}$ is sparse, the computation can be executed efficiently using $O(K)$ recursive sparse-dense matrix multiplication [57]. With the convolution operation, we can build a diffusion convolution layer that maps P-dimensional input features into Q-dimensional output features. Denote the parameter tensor as $\boldsymbol{\Theta} \in \mathbb{R}^{Q \times P \times N \times K}$,

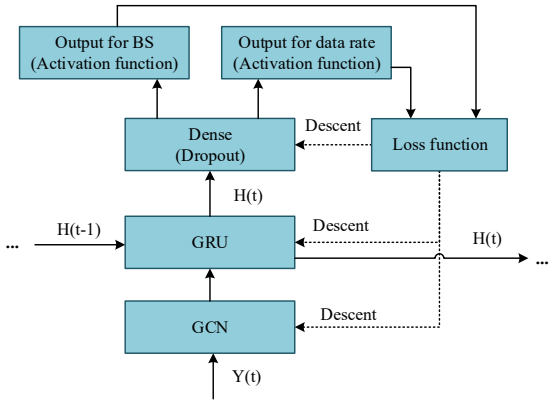


Fig. 6. Architecture for one unit of GCGRN. GCGRN is a concatenation of such units.

where $\mathbf{\Theta}_{q,p,:,:} \in \mathbb{R}^{N\times K}$ parameterizes the convolutional filter for the p-th input and the q-th output. The convolutional layer is thus $\boldsymbol{H}_{:,q} = \sigma(\sum_{p=1}^{P} \boldsymbol{x}_{:,p} * \mathcal{G} f_{\mathbf{\Theta}_{q,p,:,:}}), q \in \{1, \cdots, Q\}$ where $\boldsymbol{H} \in \mathbb{R}^{N\times Q}$ is the output and σ is a non-linear activation function, e.g., ReLU, Sigmoid. The convolutional layer learns the representations for graph-structured data and can be trained using stochastic gradient based methods.

We then integrate GRU into the above graph convolution to further model temporal dependency. In particular, we replace the matrix multiplications in GRU with the convolution operation, which leads to our proposed GCGRN.

$$\begin{aligned}
\boldsymbol{r}(t) &= \sigma(\mathbf{\Theta}_{r} * \mathcal{G}[\boldsymbol{Y}(t), \boldsymbol{H}(t-1)] + \boldsymbol{b}_r) \\
\boldsymbol{u}(t) &= \sigma(\mathbf{\Theta}_{u} * \mathcal{G}[\boldsymbol{Y}(t), \boldsymbol{H}(t-1)] + \boldsymbol{b}_u) \\
\boldsymbol{C}(t) &= \tanh(\mathbf{\Theta}_{C} * \mathcal{G}[\boldsymbol{Y}(t), \boldsymbol{r}(t) \odot \boldsymbol{H}(t-1)] + \boldsymbol{b}_c) \\
\boldsymbol{H}(t) &= \boldsymbol{u}(t) \odot \boldsymbol{H}(t-1) + (1 - u(t)) \odot \boldsymbol{C}(t)
\end{aligned}$$

where $\mathbf{\Theta}_r$, $\mathbf{\Theta}_u$, and $\mathbf{\Theta}_C$ are parameters for corresponding filters. $\boldsymbol{r}(t)$ is the reset gate at time t. Essentially, this gate is used from the model to decide how much of the past information to forget. $\boldsymbol{u}(t)$ is the update gate at time t. It helps the model to determine how much of the past information (from previous time steps) needs to be passed along to the future. $\boldsymbol{C}(t)$ is the current memory content, which determines how exactly the gates will effect the final output. It is realized by the weight matrix and the nonlinear activation function $\tanh$. $\boldsymbol{H}(t)$ is the final memory at the current time step, which holds the information for the current unit and passes it down to the network. It determines what to collect from the current memory content $\boldsymbol{C}(t)$ and what from the previous steps $\boldsymbol{H}(t-1)$.

Output Model. It is where the final prediction takes place. Recall that STEP forecasts two classes of information: which BS to access and uplink/downlink traffic of each online app at a given future time instance. Thus the output model consists of two components, one for each of them.

For the component of traffic prediction, because its output is real-valued, i.e., a vector of predicted uplink/downlink data rate for each app, we simply adopt the linear activation function. For the component of BS prediction, we apply the softmax function, which is typically used in various multiclass classification. Specifically, the input to the function is the result of $|\mathcal{V}|$ distinct linear functions. Recall that $|\mathcal{V}|$ is the number of different BSs in the dataset. Given the output of hidden layers $\boldsymbol{H}'$ and a weighting vector $\boldsymbol{W}'$, the probability that the j-th BS will be the serving BS at a specific future time instance is

$$\Pr[c = j|\boldsymbol{H}'] = \frac{e^{\boldsymbol{H}'^{\top}\boldsymbol{W}'_j}}{\sum_{k=1}^{K} e^{\boldsymbol{H}'^{\top}\boldsymbol{W}'_k}}.$$

Loss Function. Loss function is a critical indicator for whether GCGRN has achieved stable convergence in training. Different from traditional neural networks that typically have homogeneous classification objects, GCGRN handles the prediction over heterogeneous objects, i.e, BS (one-hot vector) and user traffic (real value). We then adopt different weights for each of them during aggregation. These weights, together with weight matrices/vectors in hidden layers and output models, are determined during training. Almeida-Pineda algorithm [58], a gradient-based optimization method, is adopted. It runs the propagation step to convergence and computes gradients based on the converged solution.

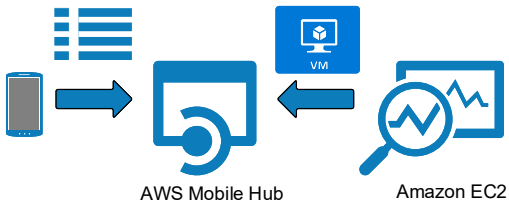


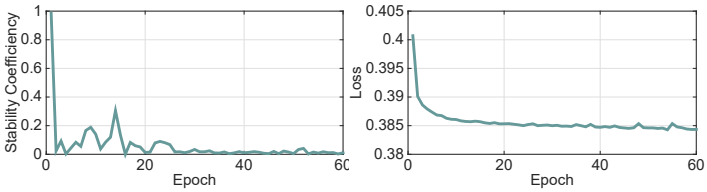
Fig. 7. The prototype of STEP built based on smartphones, AWS Mobile Hub and Amazon EC2.

5 Experimetal Evaluation

The purpose for this section is threefold, to validate the effectiveness of applying GCGRN model to user traffic prediction, to compare its performance with the state-of-the-art time-series based approaches, and to evaluate its system performances.

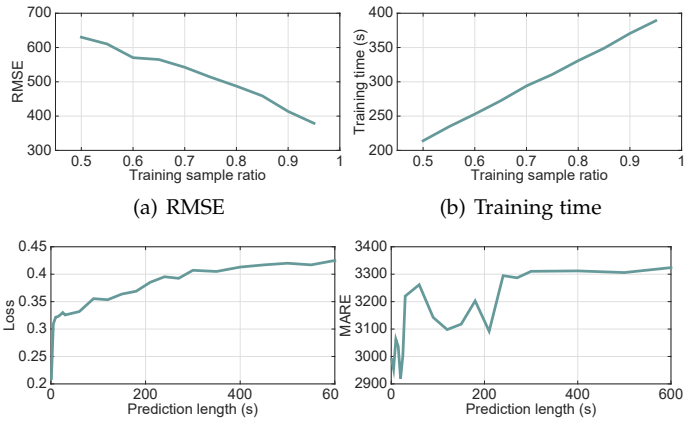
5.1 Experimental Settings

As a proof-of-concept implementation, we prototype STEP based on smartphones, AWS Mobile Hub [59], and Amazon EC2 [60]. Its logic architecture is shown in Fig. 7. All data readings are uploaded to Amazon EC2 with the assistance of AWS Mobile Hub at a predefined frequency. Amazon Mobile Hub is a collection of AWS tools designed to help developers build cloud-based applications for mobile devices. In our implementation, we integrate and configure Amazon EC2 as one service console of AWS Mobile Hub. Thus, the data collected via Mobile Hub can be readily available at the backend EC2 where training and prediction are performed. Amazon EC2 is configured with the central processing unit (CPU) Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz, with 60GB RAM and the Cuda enabled graphics processing unit (GPU) Nvidia Tesla K80. Besides, Huawei Honor 5X mobile phones are used. Each runs Android 5.1, and is equipped with a Qualcomm 8-core processor at 1.5GHz clock speed, 16GB storage and 2GB memory. GCGRN is built using the Keras library [61]. Performances are evaluated based on the



(a) Stability coefficient (b) Loss function value

Fig. 8. Evaluation of offline training process in 60 epochs.



(a) RMSE (b) Training time

(c) Loss (d) MARE

Fig. 9. Prediction performance with respect to training sample size and prediction length.

dataset discussed in Section 3.2. It covers comprehensive cellular data usage traces of 150 users, including both real and synthetic ones. The entire dataset is divided into training and test sets.

5.2 Offline Training Performance

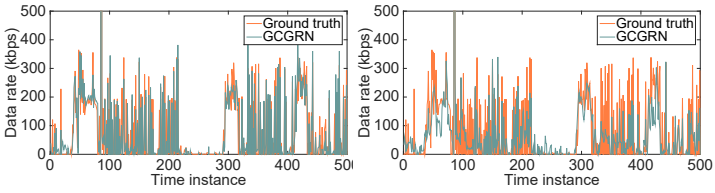
Training plays a critical role to ensure the prediction accuracy. To determine whether the model has been trained properly, we monitor the training process in Fig. 8. Fig. 8(a) shows the stability coefficient. It is calculated by $\sum|\hat{y}(t)-\hat{y}(t-1)|/\sum|\hat{y}(t)|$ that measures the distance between outputs from two successive epochs and tells whether the network converges. Note that one epoch is when an entire training dataset is passed both forward and backward through the model once. The stability coefficient quickly drops to 0 after around 30 epochs. Fig. 8(b) plots the loss value, another indicator whether the model is properly trained. It is considered as the "price" paid for prediction inaccuracy. As shown, loss tends to be stable after 20 epochs. Combining the results above, it is sufficient to set 40 epochs for training. Its corresponding time will be discussed later.

5.3 Prediction Performance

We now examine the prediction performance via two commonly used metrics: Mean Absolute Relative Error (MARE) and Relative Mean Square Error (RMSE). Specifically, MARE $= \frac{1}{M}\sum_i \frac{|y_i-\hat{y}_i|}{y_i}$ and RMSE $= \sqrt{\frac{1}{M}(\sum_i |y_i-\hat{y}_i|^2)}$ where M is the test data size, $i \in [1, M]$ is the index of each test sample, and y_i and $\hat{y}_i$ stand for the ground truth and prediction, respectively. MARE measures the relative error in a set of predictions. RMSE is a quadratic scoring rule that also measures the average magnitude of the error. Both of them are negatively-oriented scores, which means the lower the better. As a note, the prediction output of GCGRN consists of heterogeneous objects, i.e, BS (one-hot vector) and user traffic (real value). In the evaluation of MARE and RMSE, we follow the same idea of unifying them in the *loss function*. A set of weights are adopted. Their values are set to the weights in the loss function derived in the last iteration of training.

Impact of Training Sample Size. We observe in Fig. 9(a) that RMSE decreases almost linearly as the training sample ratio increases. Thus, in order to achieve better prediction accuracy, a straightforward solution is to involve more training samples. On the other hand, as shown in Fig. 9(b), it leads to longer training time. For example, when half of the data is used for training, the corresponding time is 212.5s, which is still acceptable for the offline model.

Impact of Prediction Length. Fig. 9(c) shows that the prediction accuracy deteriorates as the prediction length grows. The prediction length is exactly t_0 in (1), the number of time slots ahead of the current time instance. Loss experiences a sudden surge after $t_0 = 1$s. It increases gradually after that but becomes relatively stable after $t_0 = 300$s. We further plot MARE with respect to t_0 in Fig. 9(d). A similar trend is observed.



(a) $t_0 = 5$s (b) $t_0 = 180$s

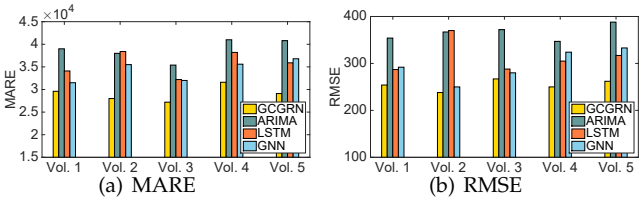
Fig. 10. Virtualized prediction result under different prediction lengths (t_0).

To better quantify prediction accuracy, we visualize forecasting results on traffic data rate under different t_0's in Fig. 10. We find that the forecasting result matches well with the ground truth when $t_0 = 5$s. It not only accurately estimates the mean when small oscillations exist, but also predicts abrupt changes. The performance becomes mediocre under a longer prediction length, which meets our expectations.

5.4 Performance Comparison with Other Prediction Models

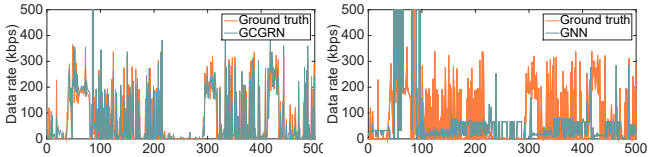
In this section, we compare the performance of GCGRN with some other existing prediction models.

- ARIMA. It is short for Auto-Regressive Integrated Moving Average model [62], which is commonly used for modeling time series behaviors and has been widely adopted in time series prediction.
- LSTM. It is short for Long-Short Term Memory [63], which is one type of recurrent neural network (RNN). In the evaluation, we have 4 layers of LSTM cell. For each layer, there are 8 LSTM cells. ReLU is adopted as the activation function.
- GNN. In [38], GNN is applied to predict large-scale cellular traffic. We use 5 hidden neurons for each



(a) MARE (b) RMSE

Fig. 11. Prediction accuracy comparison among GCGRN, ARIMA, LSTM, and GNN.



(a) GCGRN (b) GNN

Fig. 12. Virtualized prediction result comparison between GCGRN and GNN.

layer of the propagation model with a linear activation function and 6 neurons for each layer of the output model with tanh activation function.

ARIMA model is implemented using the "forecast" R package. It automatically selects the best model parameters based on the given order constraints. LSTM and GNN models are implemented using the "Keras" python library [61]. Fig. 11(a) compares MARE achieved by the four models for five randomly selected volunteers. MARE is short for Mean Absolute Relative Error. It reflects the prediction error of a given algorithm; the higher value is, the more error is exhibited. Take data from the first volunteer as an example, the MARE achieved by GCGRN, ARIMA, LSTM, and GNN is 2.96×10^4, 3.90×10^4, 3.41×10^4, and 3.15×10^4, respectively. Given these values, the prediction error, i.e., MARE, caused by our GCGRN is merely $\frac{2.96\times10^4}{3.90\times10^4} = 75.2\%$, $\frac{2.96\times10^4}{3.41\times10^4} = 84.9\%$ and $\frac{2.96\times10^4}{3.15\times10^4} = 90.7\%$ of ARIMA, LSTM and GNN, respectively. We have a similar observation on RMSE in Fig. 11(b). ARIMA is a model that captures a suite of temporal structures in time series data. For LSTM, since it introduces "memories" (cells), it takes into account the historical data for prediction. When applied in traffic prediction, they simply explore temporal dependencies in user traffic data, but neglect spatial correlations. While GNN is capable of extracting spatial features from graph-structured cellular traffic, it is not designed for handling time series data.

TABLE 2
BS Prediction accuracy comparison.

Volunteers	GCGRN (%)	GNN [38] (%)
Vol. 1	94.4 ± 1.3	81.9 ± 0.7
Vol. 2	93.6 ± 2.9	80.8 ± 1.9
Vol. 3	93.3 ± 4.2	81.1 ± 1.5
Vol. 4	92.8 ± 0.6	79.8 ± 2.5
Vol. 4	94.5 ± 1.8	80.1 ± 3.3

TABLE 3
Performance breakdown of GCGRN and GNN on vol.1.

Training sample	30%	40%	50%	60%	70%
GCGRN	71.2±1.7	81.5±2.4	87.8±0.9	92.9±2.1	94.4±1.3
GNN	62.5±0.4	71.7±1.2	75.2±2.2	80.6±1.6	81.9±0.7

Table 2 compares the BS prediction accuracy over each volunteer's dataset by GCGRN and GNN. This parameter is defined as the ratio of correct prediction on BS access. The prediction length t_0 is set to 5s for both models. We observe that GCGRN is superior to GNN across all apps. As mentioned, this is because GCGRN is empowered by GRU, the memory cells to recognize temporal patterns. Table 3 further breaks down the prediction performance for volunteer 1 as the training data size varies. GCGRN outperforms GNN in all cases. To realize a similar accuracy level, GCGRN acquires for much less training data than GNN. Fig. 12 provides the virtualized prediction result comparison between GCGRN and GNN regarding traffic data rate. Apparently, the former outperforms the latter.

TABLE 4
Comparison of BS Prediction accuracy at handoffs.

Volunteer	GCGRN (%)	GNN [38] (%)
Vol. 1	90.1 ± 0.7	79.9 ± 1.1
Vol. 2	89.3 ± 1.2	78.2 ± 1.2
Vol. 3	91.2 ± 1.3	73.4 ± 2.1
Vol. 4	88.1 ± 0.6	77.1 ± 1.9
Vol. 5	90.7 ± 1.6	75.3 ± 1.7

Table 4 shows BS prediction accuracy during handoff process. Since it is too strict to predict the exact time instance that a handoff is to occur, in the evaluation, a prediction is deemed success if the ground truth is within $\pm$5s of the predicted value. Such a relaxation is not far-fetched. In practice, we can have BSs to reserve resources a short period ahead of the predicted service arrival to remedy the impact caused by inaccurate BS prediction during handoffs. (This is also what we do in the NS-3 simulation.) We observe in the above table that its accuracy is slightly lower than the overall BS prediction accuracy (Table 2). Still, the performance is satisfactory. As shown later, GCGRN achieves similar performance with resource reservation under perfect prediction, in terms of handoff blocking ratio and resource utilization. Moreover, we also compare the prediction accuracy between our proposed GCGRN and GNN. Our approach outperforms GNN over all datasets. For example, the average accuracy for vol. 1 is 90.1%, while that of GNN is only 79.9%.

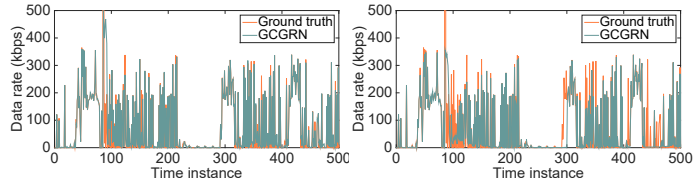
5.5 Online Training vs. Offline Training

We consider both offline and online training for the proposed GCGRN. For offline training, the model, i.e., parameters, is fixed during prediction. The training is conducted based on datasets collected for a long time duration, say one month. For online training, the model continues to update according to newly reported data collected from a user. Fig. 13 compares the prediction performance of GCGRN under these two training frameworks. For the setting $t_0 = 1$min of online training, the user uploads its transmission record for the past one minute to the cloud, which sequentially trains the model and outputs the forecasting. From Fig. 13(a) and Fig. 13(c), we find that online training produces more accurate results. Besides, even when $t_0 = 5$min, the

TABLE 5
Time comsumption comparison.

	Training time	Testing time
Offline	123.22 ± 2.4(s) (30%)	2.72 ± 0.1(s)
	170.79 ± 3.1(s) (40%)	1.65 ± 0.1(s)
	212.09 ± 1.1(s) (50%)	1.37 ± 0.2(s)
Online	45.3 ± 0.1(ms) (1/1min)	1.4 ± 0.1(ms)
	67.9 ± 0.7(ms) (1/3min)	1.9 ± 0.2(ms)
	87.7 ± 0.5(ms) (1/5min)	2.8 ± 0.2(ms)

prediction still matches well with the ground truth via online training. A similar performance can be achieved by offline training, but at a much shorter prediction length (see Fig. 10). Therefore, online training can effectively capture abrupt changes in user traffic even at a relatively longer prediction length.



(a) Online, $t_0 = 1$min (b) Online, $t_0 = 5$min

(c) Offline, $t_0 = 1$min (d) Offline, $t_0 = 5$min

Fig. 13. Prediction performance comparison between offline and online training.

Table 5 gives the time consumption of offline and online training. The number in a pair of parentheses of online and offline stands for training sample ratio and training data uploading frequency, respectively. We observe that testing (i.e., prediction) can be efficiently conducted, within a couple of seconds, in either case. The most time-consuming part is training. Besides, offline training takes longer than online training. This is because the former is performed over a much larger dataset, while the latter is conducted over a data segment for a short time period. For example, only data collected in the past one minute is used to update the current model under the uploading frequency 1/1min, i.e., $t_0 = 1$min.

As a summary, online training produces a more accurate prediction for consistently updating the GCGRN model to better accommodate unforeseen changes in a timely manner. On the other hand, online training also requires more frequent data uploading at the user side. As a result, it inevitably causes extra communication and energy consumption which will be evaluated soon.

5.6 Energy Consumption

Since the cloud is grid powered, its corresponding energy consumption is of less concern. To facilitate online training, mobile devices need to upload collected data to the cloud and is thus energy-consuming. Since mobile devices are battery-powered, it is critical to examine the impact of data transmission on their battery consumption.

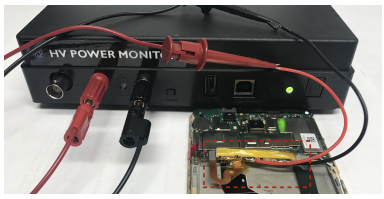


Fig. 14. Power measurement of a Huawei Honor smartphone. A compatible battery interface circuit (as shown in the red box) was carved out from the same smartphone and used as an adapter between the phone and the power monitor.

There have been some existing software-based approaches for measuring energy consumption at smartphones. However, they are mostly manually controlled and thus cannot provide agile readings. To bridge this gap, in this work we measure the precise energy consumption using the dedicated hardware, the Monsoon power monitor [64] as shown in Fig. 14. During the measurement, we kept other components, e.g., WiFi and Bluetooth, offline.

Fig. 15(a) depicts the relation between smartphone's energy consumption and data uploading frequency. The result is an accumulated value for a duration of *one hour*. The energy consumption demonstrates a positive relationship with the uploading frequency. For example, the smartphone consumes a total of 138J in an hour, when data is uploaded every 2min; the value becomes 42J when the frequency decreases to once every 6min. It is worth noting that since STEP focuses on cellular traffic prediction, data collection and uploading are only executed when mobile devices are using cellular connections. Thus, it is appropriate to examine energy consumption over a one-hour period in the experiment. Table 6 provides with power consumption of some common smartphone tasks. We notice that the power consumption of data uploading, 1173.6mW, is only slightly higher than that for screen-on operations, 769.1mW. It is much more energy-efficient than most of the operations, such as map service, sending a text message, etc. We conclude that STEP only imposes mild energy overhead to smartphones.

5.7 Communication Overhead

In addition to energy overhead, data uploading also causes communication overhead, which is measured by the total data amount transmitted in an hour. Communication overhead demonstrates a positive correlation with uploading frequency in Fig. 15(b). While the total transmission payload

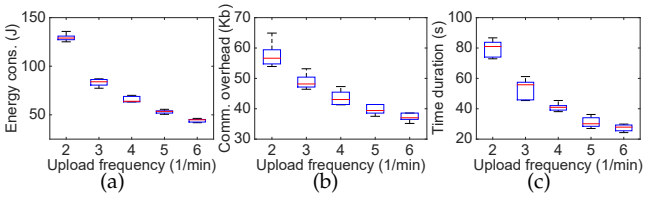


Fig. 15. Energy and communication overhead of data uploading for a one-hour duration.

TABLE 6
Power consumption by different operations at the smartphone.

Operation	Power consumption (mW)
STEP data uploading	1173.6
Screen on	769.1
Map service	2642.8
Sending a text message	1475.0
Sending an email	1632.3
Video call	3351.6
Web browsing	1732.7

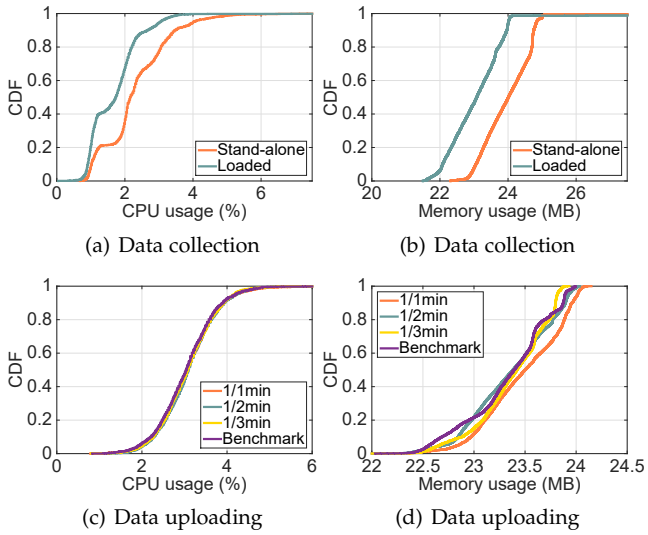


Fig. 16. Mobile device CPU and memory utilization during data collection and data uploading.

(in an hour) is roughly the same under different frequencies, a higher frequency produces extra packet heads and thus leads to larger communication costs. Fig. 15(c) shows the impact of uploading frequency to the total transmission duration in an hour. For example, when the smartphone uploads its data once per 2 minutes, it takes 80.4s as a total (in an hour); it drops to 27.3s when the frequency decreases to once per 6 minutes, which only takes a small portion of a one-hour period, i.e., 0.8%.

5.8 System Usage

To evaluation the CPU and memory utilization during data collection and transmission, we run the prototype application on mobile devices whose system specifications are clarified in Section 5.1. The program is run alone and concurrently with a high load benchmark program provided by AnTuTu [65], separately. Note that AnTuTu is a software commonly used to benchmark phones and devices.

Fig. 16(a) and Fig. 16(a) show the CPU and memory usage during data collection, respectively. We find that the data collection program consumes fewer system resources when the mobile device runs with a high load. More importantly, the data collection program requires low resources. It occupies lower than 6% CPU and 26MB memory 99% of the time. We also provide system usage of data transmission under different uploading frequencies in Fig. 16(c) and Fig. 16(d). Benchmark stands for the measurement obtained when the device is in idle status (without running any program). We observe in Fig. 16(c) that the CDF curves for CPU usage under different uploading frequencies are almost identical to that of the benchmark. It implies that the uploading process of STEP costs negligible CPU from a statistic view. Fig. 16(d) shows that a higher uploading frequency leads to a slightly larger memory occupancy. Still, the extra memory caused by data uploading is mild, compared with the device's total 2GB memory.

Clearly, there is a tradeoff between resource consumption (including energy/communication overhead and system usage) and prediction accuracy. With a lower upload frequency, we can expect a smaller resource consumption overhead. On the other hand, it can only support longer prediction lengths. As shown in Fig. 13(a) and Fig. 13(b), prediction length has a negative correlation with prediction accuracy. Combing the results all above, we find that it is suitable to have mobile devices upload their data records once every 3-6 minutes to achieve both promising prediction accuracy and reasonable resource overhead.

6 NS-3 Based Network Simulation

In order to evaluate the impact of user-level fine-granular traffic prediction to cellular network performances, we further run on the Network Simulator (NS-3) a series of network simulations. The LENA LTE module (Release ns-3.18) is adopted, which provides full LTE stack implementation based on Release 9. Some key simulation parameters are summarized in Table 7. We randomly pick a volunteer's data from the dataset that is gathered during in-field data collection campaign. A twenty-minute record is segmented to generate some critical simulation parameters, including the user's realtime data traffic and its associated cell in each time instance, its trace, and the simulation topology. Thus, all these parameters are extracted from a volunteer's real-world dataset, rather than synthetic values generated by NS-3. As there are totally 29 different cell IDs recorded in this 20-min session, we thus set this value as the total number of BSs in the simulation. A critical task is to decide their coordinates in the simulation. For each BS, we first list the user's coordinates where it is associated with this BS. Then, their centroid is used to approximate the BS' location. In addition to this "real" user, each BS is associated with 30-40 users, randomly distributed within its radius. For these users, each of them is set under `RandomWalk` model. Besides, `A2-A4-RSRQ`[1] is employed as the handoff model.

TABLE 7
Simulation parameters.

Parameter	Value
Center frequency	5.18GHz
Subcarrier bandwidth	180KHz
BS number	29
Cell radius	500m
User distribution	Random uniform
Propagation loss model	Two-ray ground
Antenna height	30m (BS), 1.5m (UE)
BS Tx power	43dB
UE Tx power	20dB
UE traffic type	CBR, 1Mbps
Simulation duration	20min

1. A handoff is triggered when the serving cell's RSRQ (Reference Signal Received Quality) becomes worse than a threshold and, meanwhile, neighbor cell's RSRQ becomes better than another threshold.

The implementation code is modified to channel the prediction results obtained from GCGRN to guide the resource allocation procedure in the MAC layer. Specifically, a certain amount of RBs are reserved at target BSs 10s ahead of the user's arrival. The exact number of RBs is derived from the user's forecasting results. The reservation is released once this user leaves that cell. Simulations are carried out in a system running Ubuntu 16.04 (LTS) 64-bit, with an 8-core Intel Core i7-4810MQ CPU 2.80GHz and 16GB RAM. All results are an average over 100 trials.

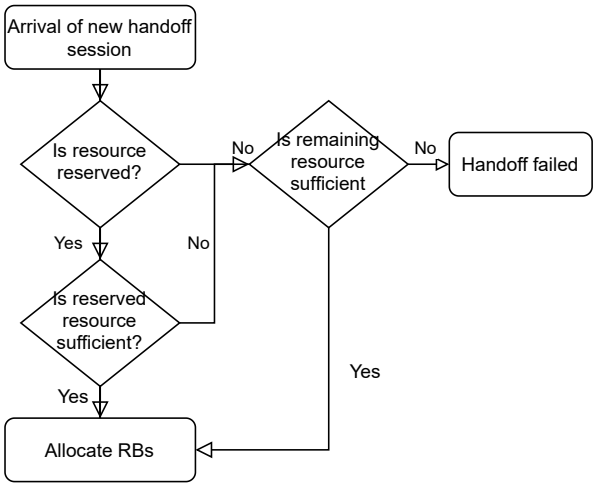
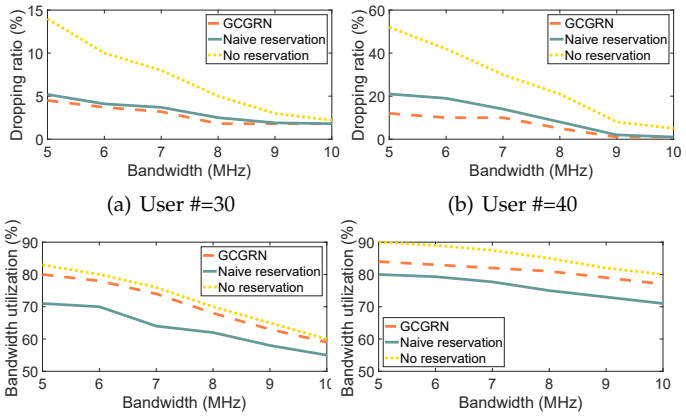


Fig. 17. The flowchart of how BS handles handoff sessions with resource reservation.

Fig. 17 shows the policy BSs follow to handle handoff sessions. It adopts the conventional fashion of prediction guided resource reservation for handoffs in cellular networks with minor adaption. The logic is implemented at the BS' MAC layer in NS-3 simulations. Such a policy follows the idea of *handoff prioritization* [66]. From the users perspective, the termination of an ongoing call is more annoying than the blocking of a new call. Consequently, the handoff blocking and the forced termination probabilities ought to be minimized. The idea of handoff prioritization approaches is to give handoff requests precedence over the new session requests in some way.



(a) User #=30 (b) User #=40

(c) User #=30 (d) User #=40

Fig. 18. Cellular network performances under different bandwidth reservation schemes.

Fig. 18(a) and Fig. 18(b) plot the user's session dropping ratio, which is defined as the percentage of unsuccessful session handoffs among all the tests. It happens when the target BS is short of RBs. We tend to use this ratio to partially reflect the conventional CDP. Besides, we also evaluate the naive reservation and the case without reservation. In the former, every BS reserves one RB for the user throughout the entire simulation duration. We find that the scheme without reservation has the highest dropping ratio, while the other two exhibit similar performances. When the network is lack of bandwidth, say 6MHz for each cell, handoff failures are likely to happen, as high as 28%. However, this value is only 5% when a reservation scheme is in place. Besides, the ratio drops quickly as more bandwidth is available. Thus, bandwidth reservation is effective to bring down session dropping ratios, especially when the network is crowded.

Fig. 18(c) and Fig. 18(d) further illustrate the bandwidth utilization under the three schemes. It is the percentage of the total bandwidth actually being used by connections in a cell and reflects how the network resource is being used. We find that the naive reservation has the lowest bandwidth utilization. Thus, the resource is poorly utilized. For the reservation scheme guided by GCGRN, RBs are only reserved 10s ahead of the user arrival. Moreover, since GCGRN is able to provide accurate traffic prediction, only a proper amount of RBs are reserved tightly according to the forecasted traffic. Thus, GCGRN is intelligent in guiding more flexible and efficient resource reservation.

We further evaluate the impact of inaccurate prediction on network performance. Specifically, we consider perfect prediction as a benchmark by which the traffic demand and arrival time of services are the same as the ground truth. We also apply GNN on traffic prediction. Table 8 compares the session dropping ratio among the three prediction approaches. Recall that the user's session dropping ratio is defined as the percentage of unsuccessful session handoffs among all the tests. We observe that the proposed GCGRN introduces similar dropping ratios as perfect prediction when there are totally 30 users in each cell due to its high prediction accuracy. On the other hand, GNN causes the highest dropping ratio. Besides, the difference is more significant when the system is short of sources, i.e., smaller spectrum bandwidth in a cell. Thus, accurate prediction is more critical to guide resource reservation in these scenarios. As a note, as GNN lacks components to capture time-dependent features in a time series dataset, it cannot provide accurate prediction as the proposed GCGRN.

Table 9 illustrates the spectrum utilization under the three approaches. It is the percentage of the total bandwidth actually being used by connections in a cell and thus reflects how the network resource is being used. We find that perfection prediction leads to the best performance, while GNN results in the worst one. For example, when the total bandwidth is 5MHz, the average utilization achieved by the three approaches are 81.7%, 81.1%, and 76.0%, respectively. Like above, the impact of prediction accuracy becomes negligible when more resources are available. The reason can be briefly explained as follows. When the predicted traffic is larger than the real amount, it would not cause noticeable network resource waste, as there are sufficient RBs to support other incoming services; when the former

TABLE 8
Session dropping ratio comparison under different prediction approaches.

Bandwidth (MHz)	5	6	7	8	9	10
Perfect prediction	3.2±1.5	2.8±1.7	1.8±1.1	1.6±0.5	1.6±0.3	1.4±0.3
GCGRN	4.5±1.4	3.7±1.8	3.2±2.2	1.8±1.5	1.8±0.7	1.8±0.5
GNN	5.0±1.2	4.0±1.3	3.6±2.0	1.9±0.8	1.8±0.4	1.8±0.3

TABLE 9
Bandwidth utilization comparison under different prediction approaches.

Bandwidth (MHz)	5	6	7	8	9	10
Perfect prediction	81.7±5.6	79.3±5.1	74.3±4.2	69.0±3.7	63.5±4.0	59.2±4.2
GCGRN	81.1±6.9	78.6±7.1	75.2±7.4	68.6±6.3	61.6±6.5	59.2±6.1
GNN	76.0±7.8	75.1±7.6	69.7±6.5	62.4±8.2	58.3±5.4	57.5±5.3

is less than the latter, the handoff service will unlikely be rejected too, as the BS can always allocate vacant RBs to it.

As a summary, GCGRN-guided resource reservation effectively reduces session dropping ratio during handoffs in cellular networks, which is a desirable property for densely deployed cells. More importantly, it only occurs mild bandwidth waste due to its accurate prediction.

7 Conclusions

In this paper we propose STEP, a spatio-temporal fine-granular user traffic prediction mechanism for cellular networks. A novel deep graph convolution network, called GCGRN, is developed to learn spatio-temporal dependency from a user's massive dataset for traffic prediction. STEP achieves impressive prediction accuracy, especially under online training model. Even though STEP requires mobile users to periodically record and upload traffic statistics to the (cloud) core network, extensive experimental results show that they merely incur mild energy/communication overhead and negligible system usage. Besides, NS-3 based simulations also show that STEP is effective in bringing down session dropping ratio in cellular networks.

References

[1] "Cisco visual networking index: Forecast and methodology, 2016–2021," https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.pdf, 2017.

[2] X. Ge, H. Cheng, G. Mao, Y. Yang, and S. Tu, "Vehicular communications for 5g cooperative small-cell networks." *IEEE Transactions on Vehicular Technology (TVT)*, vol. 65, no. 10, pp. 7882–7894, 2016.

[3] J. Chen, X. Ge, and Q. Ni, "Coverage and handoff analysis of 5g fractal small cell networks," *IEEE Transactions on Wireless Communications (TWC)*, vol. 18, no. 2, pp. 1263–1276, Feb 2019.

[4] J. Qiao, X. S. Shen, J. W. Mark, Q. Shen, Y. He, and L. Lei, "Enabling device-to-device communications in millimeter-wave 5g cellular networks," *IEEE Communications Magazine*, vol. 53, no. 1, pp. 209–215, 2015.

[5] A. Talukdar, M. Cudak, and A. Ghosh, "Handoff rates for millimeterwave 5g systems," in *Proceedings of the IEEE VTC (Spring)*, 2014.

[6] I. Farris, T. Taleb, M. Bagaa, and H. Flick, "Optimizing service replication for mobile delay-sensitive applications in 5g edge network," in *Proceedings of IEEE ICC*, 2017.

[7] I. Farris, T. Taleb, H. Flinck, and A. Iera, "Providing ultra-short latency to user-centric 5g applications at the mobile network edge," *Transactions on Emerging Telecommunications Technologies*, vol. 29, no. 4, p. e3169, 2018.

[8] G. Ding, J. Wang, Q. Wu, Y.-D. Yao, R. Li, H. Zhang, and Y. Zou, "On the limits of predictability in real-world radio spectrum state dynamics: From entropy theory to 5g spectrum sharing," *IEEE Communications Magazine*, vol. 53, no. 7, pp. 178–183, 2015.

[9] H. Yao, P. Si, R. Yang, and Y. Zhang, "Dynamic spectrum management with movement prediction in vehicular ad hoc networks." *Adhoc & Sensor Wireless Networks*, vol. 32, no. 11, 2016.

[10] M. A. Uusitalo, J.-E. Ekberg, and J.-j. H. Kaaja, "Method and apparatus for providing spectrum reservation," Jan. 3 2013, uS Patent App. 13/171,077.

[11] Y. Gadallah, M. H. Ahmed, and E. Elalamy, "Dynamic lte resource reservation for critical m2m deployments," *Pervasive and Mobile Computing*, vol. 40, pp. 541–555, 2017.

[12] E. Skondras, A. Michalas, and D. D. Vergados, "A survey on medium access control schemes for 5g vehicular cloud computing systems," in *Proceedings of IEEE Global Information Infrastructure and Networking Symposium (GIIS)*, 2018.

[13] A. Mohamed, O. Onireti, S. A. Hoseinitabatabaei, M. Imran, A. Imran, and R. Tafazolli, "Mobility prediction for handover management in cellular networks with control/data separation," in *Proceedings of IEEE International Conference on Communications (ICC)*, 2015.

[14] P. Fazio, M. Tropea, F. De Rango, and M. Voznak, "Pattern prediction and passive bandwidth management for hand-over optimization in qos cellular networks with vehicular mobility," *IEEE Transactions on Mobile Computing*, vol. 15, no. 11, 2016.

[15] A. Nadembega, A. Hafid, and T. Taleb, "A destination and mobility path prediction scheme for mobile networks," *IEEE transactions on vehicular technology*, vol. 64, no. 6, pp. 2577–2590, 2015.

[16] G. Gidófalvi and T. B. Pedersen, "Mining long, sharable patterns in trajectories of moving objects," *Geoinformatica*, vol. 13, no. 1, pp. 27–55, 2009.

[17] G. Gidófalvi and F. Dong, "When and where next: individual mobility prediction," in *Proceedings of the ACM SIGSPATIAL Workshop on Mobile Geographic Information Systems*, 2012.

[18] X. Zhou, Z. Zhao, R. Li, Y. Zhou, J. Palicot, and H. Zhang, "Human mobility patterns in cellular networks," *IEEE communications letters*, vol. 17, no. 10, pp. 1877–1880, 2013.

[19] H. Abu-Ghazaleh and A. S. Alfa, "Application of mobility prediction in wireless networks using markov renewal theory," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 2, pp. 788–802, 2010.

[20] J.-K. Lee and J. C. Hou, "Modeling steady-state and transient behaviors of user mobility: formulation, analysis, and application," in *Proceedings of the ACM international symposium on Mobile ad hoc networking and computing (MobiHoc)*, 2006, pp. 85–96.

[21] H. Farooq and A. Imran, "Spatiotemporal mobility prediction in proactive self-organizing cellular networks," *IEEE Communications Letters*, vol. 21, no. 2, pp. 370–373, 2017.

[22] H. Pang, P. Wang, L. Gao, M. Tang, J. Huang, and L. Sun, "Crowdsourced mobility prediction based on spatio-temporal contexts," in *Proceedings of IEEE International Conference on Communications (ICC)*, 2016.

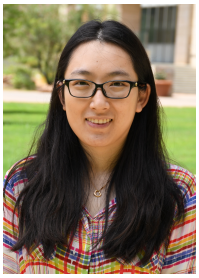
[23] I. F. Akyildiz and W. Wang, "The predictive user mobility profile framework for wireless multimedia networks," *IEEE/ACM Transactions On Networking*, vol. 12, no. 6, pp. 1021–1035, 2004.

[24] A. Nadembega, A. Hafid, and T. Taleb, "Mobility-prediction-aware bandwidth reservation scheme for mobile networks," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 6, pp. 2561–2576, 2015.

[25] J. Hu, W. Heng, G. Zhang, and C. Meng, "Base station sleeping mechanism based on traffic prediction in heterogeneous networks," in *Telecommunication Networks and Applications Conference (ITNAC), 2015 International*. IEEE, 2015, pp. 83–87.

[26] T. D. Lagkas, *Wireless Network Traffic and Quality of Service Support: Trends and Standards: Trends and Standards*. IGI Global, 2010.
[27] X. Chen, Y. Jin, S. Qiang, W. Hu, and K. Jiang, "Analyzing and modeling spatio-temporal dependence of cellular traffic at city scale," in *Proceedings of IEEE International Conference on Communications (ICC)*, 2015.
[28] H. Wang, J. Ding, Y. Li, P. Hui, J. Yuan, and D. Jin, "Characterizing the spatio-temporal inhomogeneity of mobile traffic in large-scale cellular data networks," in *Proceedings of ACM International Workshop on Hot Topics in Planet-scale Mobile computing and online Social neTworking*, 2015.
[29] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," in *International Conference on Machine Learning*, 2015, pp. 2342–2350.
[30] A. Samba, Y. Busnel, A. Blanc, P. Dooze, and G. Simon, "Instantaneous throughput prediction in cellular networks: Which information is needed?" in *Proceedings of IFIP/IEEE Symposium on Integrated Network and Service Management*, 2017.
[31] C.-H. Liu, W. Gabran, and D. Cabric, "Prediction of exponentially distributed primary user traffic for dynamic spectrum access," in *Proceedings of IEEE Global Communications Conference (GLOBECOM)*, 2012.
[32] S. Yin, D. Chen, Q. Zhang, and S. Li, "Prediction-based throughput optimization for dynamic spectrum access," *IEEE transactions on vehicular technology*, vol. 60, no. 3, pp. 1284–1289, 2011.
[33] T. Zhang, E. van den Berg, J. Chennikara, P. Agrawal, J.-C. Chen, and T. Kodama, "Local predictive resource reservation for handoff in multimedia wireless ip networks," *IEEE Journal on selected areas in Communications*, vol. 19, no. 10, pp. 1931–1941, 2001.
[34] B. Ahn, H. Yoon, and J. W. Cho, "A design of macro-micro cdma cellular overlays in the existing big urban areas," *IEEE journal on selected areas in communications*, vol. 19, no. 10, pp. 2094–2104, 2001.
[35] I. Loumiotis, E. Adamopoulou, K. Demestichas, P. Kosmides, and M. Theologou, "Artificial neural networks for traffic prediction in 4g networks," in *International Wireless Internet Conference*. Springer, 2014, pp. 141–146.
[36] C. Qiu, Y. Zhang, Z. Feng, P. Zhang, and S. Cui, "Spatio-temporal wireless traffic prediction with recurrent neural network," *IEEE Wireless Communications Letters (Early Access)*, 2018.
[37] C. Zhang, H. Zhang, D. Yuan, and M. Zhang, "Citywide cellular traffic prediction based on densely connected convolutional neural networks," *IEEE Communications Letters (Early Access)*, 2018.
[38] X. Wang, Z. Zhou, Z. Yang, Y. Liu, and C. Peng, "Spatio-temporal analysis and prediction of cellular traffic in metropolis," in *Proceedings of IEEE International Conference on Network Protocols (ICNP)*, 2017.
[39] K. Dufková, J.-Y. Le Boudec, L. Kencl, and M. Bjelica, "Predicting user-cell association in cellular networks from tracked data," in *Mobile Entity Localization and Tracking in GPS-less Environments*. Springer, 2009, pp. 19–33.
[40] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási, "Limits of predictability in human mobility," *Science*, vol. 327, no. 5968, pp. 1018–1021, 2010.
[41] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," *arXiv preprint arXiv:1611.03530*, 2016.
[42] Z. Cui, W. Chen, and Y. Chen, "Multi-scale convolutional neural networks for time series classification," *arXiv preprint arXiv:1603.06995*, 2016.
[43] M. M. Krell, A. Seeland, and S. K. Kim, "Data augmentation for brain-computer interfaces: Analysis on event-related potentials data," *arXiv preprint arXiv:1801.02730*, 2018.
[44] A. Le Guennec, S. Malinowski, and R. Tavenard, "Data augmentation for time series classification using convolutional neural networks," in *ECML/PKDD workshop on advanced analytics and learning on temporal data*, 2016.
[45] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Data augmentation using synthetic data for time series classification with deep residual networks," *arXiv preprint arXiv:1808.02455*, 2018.
[46] T. T. Um, F. M. Pfister, D. Pichler, S. Endo, M. Lang, S. Hirche, U. Fietzek, and D. Kulić, "Data augmentation of wearable sensor data for parkinson's disease monitoring using convolutional neural networks," in *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, 2017, pp. 216–220.
[47] *FCC 16-148*. FCC, 2016.
[48] A. Fout, J. Byrd, B. Shariat, and A. Ben-Hur, "Protein interface prediction using graph convolutional networks," in *Advances in Neural Information Processing Systems*, 2017, pp. 6533–6542.
[49] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.
[50] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.
[51] M. Henaff, J. Bruna, and Y. LeCun, "Deep convolutional networks on graph-structured data," *arXiv preprint arXiv:1506.05163*, 2015.
[52] R. Li, S. Wang, F. Zhu, and J. Huang, "Adaptive graph convolutional neural networks," *arXiv preprint arXiv:1801.03226*, 2018.
[53] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, "Gated graph sequence neural networks," *arXiv preprint arXiv:1511.05493*, 2015.
[54] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2009.
[55] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in neural information processing systems*, 2016, pp. 3844–3852.
[56] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," *arXiv preprint arXiv:1312.6203*, 2013.
[57] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," *arXiv preprint arXiv:1707.01926*, 2017.
[58] R. C. O'Reilly, "Biologically plausible error-driven learning using local activation differences: The generalized recirculation algorithm," *Neural computation*, vol. 8, no. 5, pp. 895–938, 1996.
[59] Amazon mobile hub. [Online]. Available: https://aws.amazon.com/mobile/
[60] Amazon ec2. [Online]. Available: https://aws.amazon.com/ec2/
[61] F. Chollet *et al.*, "Keras," 2015.
[62] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.
[63] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[64] "Power monitor software," http://msoon.github.io/powermonitor/.
[65] "Antutu," http://www.antutu.com/en/.
[66] A. Sgora and D. D. Vergados, "Handoff prioritization and decision schemes in wireless cellular networks: a survey," *IEEE Communications Surveys & Tutorials*, vol. 11, no. 4, pp. 57–77, 2009.

Lixing Yu received the B.S. degree in Electrical Engineering from Beijing Institute of Technology, China, in 2012, and the M.S. degree in Electrical Engineering from the George Washington University, Washington DC, in 2014, respectively. He is currently working toward the PhD degree in the Department of Electrical Engineering and Computer Science, Case Western Reserve University. His research interests include deep learning, and data mining.



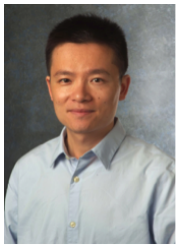
Ming Li received the B.E. degree in Electrical Engineering from Sun Yat-sen University, China, in 2007, the M.E. degree in Electrical Engineering from Beijing University of Posts and Communications, China, in 2010, and the Ph.D. degree in Electrical and Computer Engineering from Mississippi State University, Starkville, in 2014, respectively. She is currently an assistant professor in the Department of Computer Science and Engineering, The University of Texas at Arlington. Her research interests include mobile computing, Internet of things, security, and privacy-preserving computing. Her work won Best Paper Awards in Globecom 2015 and DASC 2017, respectively. She is a member of IEEE.

Wenqiang Jin received the B.E. and M.E. degree in Electrical Engineering from Chongqing University of Posts and Telecommunications, China, in 2011 and 2017, respectively. He is currently an Ph.D. student in the Department of Computer Science and Engineering, The University of Texas at Arlington. His research interests include mobile crowd sensing and IoT security.

Yifan Guo received the B.S. degree in Information and Computing Sciences from Beijing University of Posts and Telecommunications, China, in 2013, and the M.S. degree in Computer Science from Northwestern University, Evanston, in 2015, respectively. He is currently working toward the PhD degree in the Department of Electrical Engineering and Computer Science, Case Western Reserve University. His research interests include artificial intelligence, machine learning and big data.

Qianlong Wang received the B.E. degree in Electrical Engineering from Wuhan University, China, in 2013, and the M.E. degree in Electrical and Computer Engineering from Stevens Institute of Technology, Hoboken, NJ, in 2015, respectively. He is currently working toward the PhD degree in the Department of Electrical Engineering and Computer Science, Case Western Reserve University. His research interests include big data, robust and secure deep learning network, security and privacy in distributed system, e.g., Cyber Physical System (CPS) and Internet of Things (IoT). He is a student member of the IEEE.

Feng Yan Feng Yan is an Assistant Professor in the Department of Computer Science and Engineering at the University of Nevada, Reno. He obtained both M.S. degree and Ph.D. degree in Computer Science from the College of William and Mary, and worked at Microsoft Research and HP Labs. He has a broad interest in big data and system areas. His current research focus includes machine learning, cloud/edge/fog computing, high performance computing, storage, and cross-disciplinary topics among them and others. He is a member of IEEE and ACM.



Pan Li received the B.E. degree in Electrical Engineering from Huazhong University of Science and Technology, Wuhan, China, in 2005, and the Ph.D. degree in Electrical and Computer Engineering from University of Florida, Gainesville, in 2009, respectively. He is currently an Associate Professor with the Department of Electrical Engineering and Computer Science at Case Western Reserve University. He was an Assistant Professor in the Department of Electrical and Computer Engineering at Mississippi State University between August 2009 and August 2015. His research interests include network science and economics, energy systems, security and privacy, and big data. He has been serving as an Editor for IEEE Wireless Communications Letters, IEEE Journal on Selected Areas in Communications – Cognitive Radio Series and IEEE Communications Surveys and Tutorials, a Feature Editor for IEEE Wireless Communications, a General Chair for IEEE SmartData 2017, and a Technical Program Committee (TPC) Co-Chair for Ad-hoc, Mesh, Machine-to-Machine and Sensor Networks Track, IEEE VTC 2014, Physical Layer Track, Wireless Communications Symposium, WTS 2014, and Wireless Networking Symposium, IEEE ICC 2013. He received the NSF CAREER Award in 2012 and is a member of the IEEE and the ACM.