Outdoor Position Recovery From Heterogeneous Telco Cellular Data

Yige Zhang, Weixiong Rao, Member, IEEE, Kun Zhang, and Lei Chen

Abstract—Recent years have witnessed unprecedented amounts of data generated by telecommunication (Telco) cellular networks. For example, measurement records (MRs) are generated to report the connection states between mobile devices and Telco networks, e.g., received signal strength. MR data have been widely used to localize outdoor mobile devices for human mobility analysis, urban planning, and traffic forecasting. Existing works using first-order sequence models such as the Hidden Markov Model (HMM) attempt to capture spatio-temporal locality in underlying mobility patterns for lower localization errors. The HMM approaches typically assume stable mobility patterns of the underlying mobile devices. Yet real MR datasets exhibit heterogeneous mobility patterns due to mixed transportation modes of the underlying mobile devices and uneven distribution of the positions associated with MR samples. Thus, the existing solutions cannot handle these heterogeneous mobility patterns. To this end, we propose a multi-task learning-based deep neural network (DNN) framework, namely PRNet+, to incorporate outdoor position recovery and transportation mode detection. To make sure that PRNet+ can work, we develop a feature extraction module to precisely learn local-, short- and long-term spatio-temporal locality from heterogeneous MR samples. Extensive evaluation on eight datasets collected at three representative areas in Shanghai indicates that PRNet+ greatly outperforms state-of-the-arts by lower localization errors.

Index Terms—Cellular signal measurement, outdoor localization, deep learning

1 INTRODUCTION

Recent years have witnessed unprecedented amounts of data generated by telecommunication (Telco) cellular networks. For example, when mobile devices make phone calls or access data services, measurement records (MRs) are generated to report connection states, e.g., received signal strength of mobile device during each call/session. In a modern urban city, Telco data generated by Telco radio and core equipment account to 2.2TB per day [7]. Massive Telco historical data (e.g., MRs) have been widely used to understand human mobility [4], [42] and facilitate applications such as urban planning and traffic forecasting [5], due to the unique advantages — Telco data can be collected cheaply, frequently, and on a global scale.

The key to model human mobility and the applications above is to capture outdoor positions of mobile users. The position recovery problem, i.e., precisely localizing outdoor mobile devices from Telco historical data (e.g., MRs), has attracted intensive research interests. A simple approach adopted by Google MyLocation [1] is to approximate outdoor locations by historical data (e.g., MRs) have been widely used to localize outdoor mobile devices for human mobility analysis, urban planning, and traffic forecasting. Existing works using first-order sequence models such as the Hidden Markov Model (HMM) attempt to capture spatio-temporal locality in underlying mobility patterns for lower localization errors. The HMM approaches typically assume stable mobility patterns of the underlying mobile devices. Yet real MR datasets exhibit heterogeneous mobility patterns due to mixed transportation modes of the underlying mobile devices and uneven distribution of the positions associated with MR samples. Thus, the existing solutions cannot handle these heterogeneous mobility patterns. To this end, we propose a multi-task learning-based deep neural network (DNN) framework, namely PRNet+, to incorporate outdoor position recovery and transportation mode detection. To make sure that PRNet+ can work, we develop a feature extraction module to precisely learn local-, short- and long-term spatio-temporal locality from heterogeneous MR samples. Extensive evaluation on eight datasets collected at three representative areas in Shanghai indicates that PRNet+ greatly outperforms state-of-the-arts by lower localization errors.

The key to model human mobility and the applications above is to capture outdoor positions of mobile users. The position recovery problem, i.e., precisely localizing outdoor mobile devices from Telco historical data (e.g., MRs), has attracted intensive research interests. A simple approach adopted by Google MyLocation [1] is to approximate outdoor locations by historical data (e.g., MRs) have been widely used to localize outdoor mobile devices for human mobility analysis, urban planning, and traffic forecasting. Existing works using first-order sequence models such as the Hidden Markov Model (HMM) attempt to capture spatio-temporal locality in underlying mobility patterns for lower localization errors. The HMM approaches typically assume stable mobility patterns of the underlying mobile devices. Yet real MR datasets exhibit heterogeneous mobility patterns due to mixed transportation modes of the underlying mobile devices and uneven distribution of the positions associated with MR samples. Thus, the existing solutions cannot handle these heterogeneous mobility patterns. To this end, we propose a multi-task learning-based deep neural network (DNN) framework, namely PRNet+, to incorporate outdoor position recovery and transportation mode detection. To make sure that PRNet+ can work, we develop a feature extraction module to precisely learn local-, short- and long-term spatio-temporal locality from heterogeneous MR samples. Extensive evaluation on eight datasets collected at three representative areas in Shanghai indicates that PRNet+ greatly outperforms state-of-the-arts by lower localization errors.
much smaller than the one of ten minutes. Even with the same time interval (e.g., one minute), the distance between two neighbouring MR samples in the driving mode is significantly greater than the one in the walking mode. If HMM approaches do not take into account the heterogeneity of MR samples, localization performance could degrade greatly.

Recall that Telco data usually cover much higher population and more areas. It is meaningful to detect transportation modes on Telco data for many applications such as urban sensing and planning, transportation management and precision advertisement. Moreover, the transportation modes of underlying mobile devices are helpful to capture moving speed and then estimate the moving distance of mobile devices within a certain time period. If the transportation modes are known, we have chance to improve outdoor position recovery. Unfortunately, existing methods frequently detect transportation modes with help of GPS trajectories [8], [28], [29], [45]. How to detect transportation modes directly from raw MR samples rather than GPS data is non-trivial.

To this end, we propose a multi-task learning-based deep neural network (DNN) framework, namely PRNet+, to incorporate the two tasks (i.e., outdoor position recovery and transportation mode detection). To make sure that PRNet+ can work, we develop a feature extraction module to precisely learn local-, short- and long-term spatio-temporal locality from heterogeneous MR samples, with help of convolutional neural networks (CNNs), long short-term memory (LSTM) cells and attention mechanism. Specifically, PRNet+ 1) allows the various-length sequences of MR samples, such that the two components (CNN and LSTM) are able to capture spatial locality from the samples within each MR sequence, 2) exploits two attention mechanisms for time-intervals between neighbouring MR samples, together with those between neighbouring MR sequences, to capture temporal locality, and 3) incorporates the detected transportation modes and predicted locations of heterogeneous MR data into a joint loss for better performance for both learning tasks. As a summary, we make the following contributions.

1) PRNet+ properly develop a multi-task learning framework to solve the two tasks of outdoor position recovery and transportation mode detection. Since the two tasks share the same features learned from heterogeneous MR data via the developed feature extraction module, we have chance to avoid high overheads to learn features independently for each task.

2) Our work differs from the existing works which typically detect transportation modes by explicitly using GPS location data. Yet, to the best of our knowledge, this is the first attempt to directly learn transportation modes on the fine-grained granularity of individual raw MR samples, which can meanwhile improve position recovery errors.

3) We evaluate PRNet+ against the state-of-the-art counterparts on eight datasets collected in three representative (core, urban and rural) areas in Shanghai, China. Our evaluation indicates that PRNet+ outperforms these counterparts by much smaller localization errors.

The rest of this paper is organized as follows. Section 2 first introduces preliminaries. Section 3 then gives the problem setting and Sections 4, 5, and 6 present the design. Next, Section 7 evaluates PRNet+ and Section 8 reviews related works. Section 9 finally concludes the paper.1

2 PRELIMINARIES

Measurement Record (MR) Data measure the connection states between mobile devices and neighbouring base stations. Table 1 gives an example of 2G GSM MR collected by an Android phone. This MR sample contains a unique number (known as IMSI, International Mobile Subscriber Identity), connection time stamp (MRTIME), up to 7 nearby base stations (RNCD and CELLD), signal measurements such as AsuLevel, SignalLevel, and RSSI (radio signal strength indicator). The base stations are frequently sorted by descending order of signal level and strength. Thus, the base station with the order index 1 (RNCD_1 and CELLD_1) is with the strongest signal and typically selected as the primary serving base station to provide data and communication services for mobile devices. When a mobile device is moving out of the signal coverage range of a primary serving base station, the handoff between base stations occurs to re-select a new primary serving base station for the mobile device.

Telco Location Recovery. Existing works on Telco position recovery are typically divided into three categories. First, measurement-based methods approaches localize mobile devices based on absolute point-to-point distance or angles [21], [31], [33]. Triangulation localization, one of the widely used measurement-based approaches, usually does not work well for 4G LTE MR data where signal strengths of one or at most two cells are available in most cases. Yet triangulation

1. The source code of PRNet+ and partial datasets are available in https://github.com/KaronZ/SeqL.
localization requires signal strength regarding three and ideally four or more cells.

Second, fingerprinting-based methods [16], [18], [20] usually have better performance than measurement-based methods, by dividing an area of interest into small grids and representing each grid by an associated fingerprint [18]. NBL [20], a recently proposed work, assumes that signal strengths of each neighbouring cell tower in the grid follow a Gaussian distribution. The online stage next adopts either Maximum Likelihood Estimation (MLE) or Weighted Average (WA) to localize mobile devices.

Lastly, some recent works [16], [20], [26], [39], [46] adopt machine learning (ML) techniques such as Random Forest to build a localization model. The model maintains the correlations between the features extracted from MR samples and associated locations (e.g., GPS coordinates). The predicted locations could be either spatial regions (grid cells) or numeric GSP coordinates. ML techniques then train the corresponding multi-classifiers [16], [20], [26] or regression models [46]. The recent work DeepLoc [26] develops a data augmentor for more training samples, and then trains a deep learning-based outdoor cellular localization model using the augmented samples for better localization result.

Note that ML techniques differ from measurement- and fingerprinting-based methods in terms of application scenarios. That is, ML techniques usually exhibit better results because they frequently leverage rich information from various Telco data including (geo-tagged) MR data and configuration parameters (e.g., GPS coordinates) of base stations, Web log data, and etc. Instead, when measurement and fingerprinting-based methods work on frontend mobile devices for active localization, they may not exploit the rich data as ML-based localization and cannot achieve comparable accuracy.

3 Problem Setting and Solution Overview

3.1 Problem Setting
Suppose that Telco operators have maintained a historical MR database. MR samples usually do not contain the accurate locations of mobile devices (identified by IMSI), and we expect to annotate MR samples with the accurate locations of these mobile devices. There are various ways to acquire the locations. For example, when mobile users are using vehicle navigation services and switching on GPS receivers in the mobile devices, the GPS coordinates of mobile devices are embedded in the URLs of mobile web logs. By extracting GPS coordinates from such URLs, we establish a linkage between extracted GPS coordinates and MR samples in light of IMSI and timestamp. In this way, MR samples are tagged by the linked GPS coordinates.

Nevertheless, mobile users frequently switch off GPS receivers on mobile devices e.g., for energy saving. Mobile Web logs contain rather sparse GPS coordinates. Thus, the majority of Telco MR data do not be tagged with the associated GPS coordinates. To tackle this issue, we first train a machine learning model by using sparse geo-tagged MR samples as training data and next predict outdoor locations for non-tagged MR samples (as testing data). Recall that single-point-based location recovery [46] may not capture spatio-temporal locality of underlying mobile devices. We thus expect to recover a trajectory of the locations predicted from a sequence of MR samples.

Definition 1. [MR Database D] A database D of MR samples is organized as follows. We first group the MR samples by IMSI and then sort the samples in each group by timestamp. In this way, each IMSI is with a series of sorted MR samples.

Suppose that every MR sample in D is with an associated GPS position and transportation mode. Then for each IMSI, we have a series of sorted MR samples and an associated trajectory of GPS positions. We are now ready to define the trajectory recovery problem.

Problem 1 [Position Trajectory Recovery Problem] Given a database D, by training a multi-task learning-based framework R, the position recovery problem aims to map a testing MR series to a GPS trajectory by R.

3.2 Challenges
A baseline approach to address Problem 1 is to exploit a sequence model such as the popular RNN or LSTM model. For example, we might divide every MR series into fixed-length windows of MR samples and then feed these windows of MR samples into the LSTM network for trajectory recovery. However, the baseline approach does not work well to capture spatio-temporal locality and suffers from high localization errors, due to the following challenges caused by heterogeneous MR samples.

1) Mixed transportation modes: The simple assumption that the mobile device regarding a given MR series always moves by a certain transportation mode (e.g., walking) clearly does not hold. Depending on various transportation modes, the spatial distance between neighbouring MR samples differs significantly.

2) Irregular MR sampling rate: Practically it is highly possible that the time intervals of some neighbouring MR samples within a real world MR series are very short (e.g., one minute) and yet those of other samples are rather long (e.g., one hour). Thus, assuming equal time intervals between neighbouring MR samples is unreasonable.

3) Uneven density of deployed base stations: Base stations deployed in urban areas are frequently very dense and those in rural areas yet sparse. Consequently, the neighbouring MR samples within the MR series located in urban and rural areas, even with exactly the same spatial distance and time interval, could exhibit very different Telco signal handoff behavior between base stations.

3.3 Solution Overview
To address the challenges above, we propose a multi-task learning-based framework, PRNet. As shown in Fig. 1, PRNet consists of three modules: 1) Feature Extraction Module to learn spatio-temporal features for input MR samples, 2) Position Recovery Module to recover a position trajectory for an input MR series, and 3) Mode Detection Module to estimate the transportation mode for each MR sample within the MR series.

The key of PRNet is to train a multi-task learning framework, where the tasks of position recovery and
transportation mode detection tasks share the same learned features. Given the input MR data of PRNet\(^+\), the feature extraction module learns spatio-temporal features, which are then fed into the two learning tasks (position recovery and mode detection) to generate the corresponding output results, respectively. To train the multi-task learning framework, we design a join loss function consisting of three individual loss functions. By incorporating the two learning tasks into the unified multi-task learning framework, PRNet\(^+\) leads to much lower localization errors than the previous work PRNet [40], which is a single-task learning approach of outdoor position recovery. It makes sense because the learned transportation mode in PRNet\(^+\) directly indicates a moving speed constraint for better localization result.

### 3.4 MR (Sub)Sequence

It is not hard to find that the number of MR samples within a MR series varies from the associated mobile devices. To capture the spatial locality from the various-length MR series, we exploit the Telco signal handoff between base stations as follows. Let us assume that the Telco signal coverage range of a certain base station \(bs\) is a circle with the radius around 1000 meters. The MR samples using \(bs\) as the primary serving base station exhibit spatial locality because these samples are located within the circle. Thus, based on the stations in MR samples, we define the following MR (sub)sequences.

**Definition 2.** [MR Sequence \(S\)] Given a series of sorted MR samples regarding a certain IMSI, we divide the MR series into multiple sequences by primary serving base stations (RNCID\(_1\) and CellID\(_1\)). The MR samples in each sequence \(S\), sorted by timestamp, share the same primary serving base station and IMSI.

Fig. 2 illustrates 6 MR sequences \(S_{a1}, S_{a2}, \ldots, S_{c1}, S_{c2}\) generated from 3 MR series identified by the 3 mobile devices \(a, b, c\) across the coverage range of two primary serving base stations \(bs_1\) and \(bs_2\). For a mobile device say \(a\), the handoff point \(h_a\) divides its MR series into two sequences \(S_{a1}\) and \(S_{a2}\). Beyond MR sequence, we further define the following MR subsequences.

**Definition 3.** [MR Subsequence] By the first non-serving base station (RNCID\(_2\) and CellID\(_2\)), we divide a certain MR sequence \(S\) into multiple MR subsequences \(S\). The MR samples in each subsequence \(S\) share exactly the same non-serving base station (RNCID\(_2\) and CellID\(_2\)), primary serving base station and IMSI.

Now even with uneven density of deployed base stations, the division of MR samples into MR (sub)sequences by the approach above could lead to the following benefit: since the MR samples within these subsequences are all within Telco radio coverage of the associated base stations, these MR samples exhibit rather high spatio-temporal locality. In this way, by learning the corresponding locality from MR (sub)sequences, PRNet\(^+\) has chance to achieve better results of outdoor localization and mode detection. Consider that the number of samples within MR sequences varies. We by default set the size of a MR sequence fed to PRNet\(^+\) to be \(\tau = 40\). In case that the actual number of MR samples within a MR sequence is greater than \(\tau\), we then follow the step above to divide the long sequence into shorter sub-sequences until the tail sub-sequence contains at most \(\tau\) samples. For the tail sub-sequence, we add padding items (e.g., \(-1\)) to make sure that this sub-sequence contains \(\tau\) samples.

### 4 Feature Extraction Module

We first introduce the data model of feature extraction module as follows. For convenience, in the rest of this paper, we denote scalars by lowercase letters e.g., \(a\), vectors by bold lowercase letters e.g., \(a\), matrices by italic bold upper-case letters e.g., \(A\), and tensors by bold upper-case letters e.g., \(A\). Specifically, we represent each MR sample by a matrix \(X \in \mathbb{R}^{F \times N}\), where \(N\) is the number of base stations in this MR sample and \(F\) is the number of MR features of each base station. The MR sample in Table 1 involves \(N=7\) stations and \(F=7\) features (i.e., RNCID, CellID, latitude and longitude of the station identified by RNCID/CellID, ASULevel, Signal-Level and RSSI).

We represent a MR sequence \(S\) by a sequence of feature matrices \(X = \{X_{1,1}, \ldots, X_{i,j}, \ldots, X_{q,S}\}\), where \(q\) is the number of MR subsequences within \(S\) and \(|S_i|\) is the amount of MR samples in the subsequence \(S_i\). The MR feature matrix \(X_{i,j} \in \mathbb{R}^{F \times N}\) corresponds to the \(j\)-th MR sample.
in $S_i$ with $1 \leq i \leq q$ and $1 \leq j \leq |S_j|$. Given an input sequence $S$, we will learn a corresponding sequence of feature vectors $V = \{v_{i,1}, \ldots, v_{i,|S_j|}, \ldots, v_{q,|S_j|}\}$.

Fig. 3 gives the overview of Feature Extraction Module, consisting of two main components.

1) **Local Single-Point Predictor** takes each individual matrix $X_{i,j}$ as input and generates a corresponding hidden vector $p_{i,j}$ as the output. The local predictor is composed of a convolution layer and a recurrent layer to capture the local dependencies from the MR features in $X_{i,j}$.

2) **Global Recurrent Predictor** takes a sequence of generated hidden vectors $P = \{p_{i,1}, \ldots, p_{i,j}, \ldots, p_{q,|S_j|}\}$ as input and generates a sequence of learned feature representation vectors $V$. This predictor consists of three layers: a bottom recurrent layer to learn short-term dependencies within each subsequence, an upper recurrent layer to capture the long-term dependencies among subsequences, and an output layer to mix the short- and long-term dependencies in order to generate the shared feature representation vectors of the input MR sequence.

### 4.1 Local Predictor

Each MR sample contains the signal measurements of up to 7 base stations. Thus, in Fig. 4, the convolution layer in the local predictor captures the local dependencies among the $F$ MR features from the input feature matrix $X_{i,j}$ and generates an output feature vector $p_{i,j}$. Intuitively, the MR sample can be alternatively treated as a sequence of the $N$ base stations sorted by the associated signal measurements. Thus, the recurrent layer captures the local dependencies from the sequence.

#### 4.1.1 Local Convolution Layer

Given an input $F \times N$ matrix $X_{i,j}$, the convolution layer captures the local dependencies of $F$ features. Thus, the convolution layer adopts multiple one-dimensional filters with size $F \times 1$ where the height of each filter is equal to the number $F$. Since the size of the convolution filter is consistent with that of the feature vector of a certain base station in $X_{i,j}$, the convolution operation can extract the local dependencies among multiple features of the base station. The output vector of the $k$-th filter throughout the input matrix $X_{i,j}$ is

$$K_{i,j,k}^r = ReLU(W_k^r \circ X_{i,j} + b_k^r),$$

where $\circ$ denotes a convolution operation and $ReLU(\cdot)$ is the activation function. The superscript $c$ indicates the convolution layer. We set the stride of convolution operation to one and ensure that each output vector $K_{i,j,k}$ has the size $N$.

Given $K$ filters, the output of the local convolution layer is a $K \times N$ matrix. For the $n$-th base station $(1 \leq n \leq N)$ within $X_{i,j}$, the $K \times N$ matrix has an associated row vector $x_{i,j,n} \in \mathbb{R}^{K \times 1}$, treated as the latent feature vector of the $n$-th base station.

#### 4.1.2 Local Recurrent Layer

Given a $K \times N$ matrix above, the local recurrent layer treats it as a sequence of $N$ latent feature vectors. The intuition is that the order of these latent feature vectors indicates the inherent correlations among the $N$ base stations in an input MR sample. Thus, this local recurrent layer extracts the local dependencies from the latent feature vectors regarding $N$ base stations. We implement the local recurrent layer by LSTM [12] and compute the hidden state of the LSTM cells for the latent vector $x_{i,j,n}$ as follows:

$$z_{i,j,n}^r = \tanh(W_x^r h_{i,j,n-1}^r \circ x_{i,j,n} + b_x^r),$$

$$f_{i,j,n}^r = \sigma(W_x^f h_{i,j,n-1}^r \circ x_{i,j,n} + b_f^r),$$

$$g_{i,j,n}^r = \sigma(W_x^g h_{i,j,n-1}^r \circ x_{i,j,n} + b_g^r),$$

$$c_{i,j,n}^r = f_{i,j,n}^r * c_{i,j,n-1}^r + g_{i,j,n}^r * z_{i,j,n}^r,$$

$$o_{i,j,n}^r = \sigma(W_x^o h_{i,j,n-1}^r \circ x_{i,j,n} + b_o^r),$$

$$h_{i,j,n}^r = o_{i,j,n}^r * \tanh(c_{i,j,n}^r),$$

where $\ast$ denotes element-wise multiplication and $h_{i,j,n}$ is the hidden state of the $n$-th base station in $X_{i,j}$. The superscript $r$ denotes a local recurrent layer. The output of this layer, $p_{i,j} \in \mathbb{R}^{(N \times d_l) \times 1}$, is the concatenation result of all hidden states, where $d_l$ denotes the number of hidden units in this local recurrent layer.

Until now, the local predictor encodes an individual feature matrix $X_{i,j}$ into a hidden feature vector $p_{i,j}$. When an entire matrix sequence $X$ is processed, the local predictor generates a sequence $P = \{p_{1,1}, \ldots, p_{1,j}, \ldots, p_{q,|S_j|}\}$, which is fed into the global recurrent predictor to generate a sequence of feature vectors.

### 4.2 Global Recurrent Predictor

Fig. 5 gives the network structure of our global recurrent predictor. Inspired by the hierarchical deep neural network for document classification problem [37], the global

---

**Fig. 3.** Network structure of feature extraction module.

**Fig. 4.** Network structure of local predictor.
The recurrent predictor first employs a bottom recurrent layer with time interval attention to extract the short-term latent dependencies among MR samples within a subsequence. An upper recurrent layer is then exploited to learn the long-term dependencies among MR subsequences to generate subsequence attention. The employed attention mechanism is meaningful to overcome the challenge of irregular MR sampling rates. Finally, an output layer merges the hidden state of bottom recurrent layer and generated subsequence attention of upper recurrent layer to produce the final output of this module.

### 4.2.1 Bottom Recurrent Layer

The bottom recurrent layer adopts LSTM cells to capture the short-term dependencies among the hidden vectors \(p_{i,1}, \ldots, p_{i,t} | S_i\) within a certain subsequence \(S_i\). However, a standard LSTM model ignores the difference of time intervals between the neighbouring cells. The length of time interval indicates the relevance from the previous cell to the current one. For instance, if the time interval increases, the contribution of previous cell becomes weak. Thus, we utilize the time interval attention mechanism to address this issue.

**Time-Interval Attention.** Fig. 6 gives the LSTM cell structure with time interval attention. Specifically, the attention acts on the forget gate and input gate, respectively. We construct the time interval attention via a linear perceptron by referring to the time difference between the current state and the previous one with:

\[
a_{i,j} = \tanh(w^a \Delta t_{i,j} + b^a),
\]

where \(w^a\) and \(b^a\) are learnable parameters and \(\Delta t_{i,j}\) is the timestamp difference between the current MR sample w.r.t \(p_{i,j}\) and the previous one. We denote the time interval attention weight by \(a_{i,j}\) to update the forget and input gates of LSTM cell as:

\[
\begin{align*}
z_{i,j,n}^t &= \tanh(W_{ih}^b h_{i,j,n-1}^b + x_{i,j,n} + b^b), \\
f_{i,j,n}^t &= \sigma(W_{if}^b h_{i,j,n-1}^b + x_{i,j,n} + b^b) \cdot a_{i,j}, \\
g_{i,j,n}^t &= \sigma(W_{ig}^b h_{i,j,n-1}^b + x_{i,j,n} + b^b) \cdot (1 - a_{i,j}), \\
c_{i,j,n}^t &= f_{i,j,n}^t \cdot c_{i,j,n-1}^b + g_{i,j,n}^t \cdot x_{i,j,n}, \\
o_{i,j,n}^t &= \sigma(W_{io}^b h_{i,j,n-1}^b + x_{i,j,n} + b^b), \\
h_{i,j,n}^b &= o_{i,j,n}^t \cdot \tanh(c_{i,j,n}^t),
\end{align*}
\]

where superscript \(b\) indicates the bottom recurrent layer. The time attention \(a_{i,j}\) first acts on the forget gate and models a temporal decay to discard the information from the previous cell state. Next, since the latest state mainly determines the output of the current input, the attention \((1 - a_{i,j})\) is thus applied to the input gate.

### 4.2.2 Upper Recurrent Layer

After the bottom recurrent layer has extracted the short-term temporal dependencies in the subsequence \(S_i\), the upper recurrent layer next captures the long-term dependencies between MR subsequences with the following input:

\[
p_i = \sum_j h_{i,j}^b,
\]

where \(h_{i,j}^b\) denotes the \(j\)-th hidden state of the subsequence \(S_i\), acquired from the bottom recurrent layer. Since the upper recurrent layer is to capture the correlations among subsequences, the input of this layer, \(p_i\), needs to consider the latent characteristics of all elements in \(S_i\). Thus, we compute the input \(p_i\) by the sum of the hidden states in \(S_i\).

**Subsequence Attention.** In the upper recurrent layer, we again leverage the LSTM cells to capture the long-term dependency by a subsequence attention mechanism to adaptively select meaningful hidden states (subsequences) across all subsequences \((i = 1, \ldots, q)\). Specifically, the attention weight of the \(i\)-th hidden state \(h_{i}^b\) regarding the subsequence \(S_i\) is computed as:

\[
\begin{align*}
u_i &= (v^T)^T \tanh(W^v h_i^b + b^v), \\
\beta_i &= \exp(u_i) / \sum_{m=1}^q \exp(u_m),
\end{align*}
\]

where \(v^T\), \(W^v\) and \(b^v\) are learnable parameters. The attention weight \(\beta_i\) represents the importance of the \(i\)-th hidden state \(h_i^b\) for the prediction, where superscript \(u\) refers to the upper recurrent layer. Based on the weight \(\beta_i\), the upper recurrent layer generates a weighted context vector \(u_i = \beta_i h_i^b\) as the output.

### 4.2.3 Output Layer

The output layer is a fully connected (FC) layer to merge the outputs of the bottom and upper recurrent layer. The input of this FC layer includes 1) the hidden state \(h_{i,j}^b\) of bottom layer at time step \(j\) of the subsequence \(S_i\) and 2) the context vector \(u_i\) of upper layer of the subsequence \(S_i\). The final output of this module, denoted by \(v_{i,j} \in \mathbb{R}^{d_f}\) where \(d_f\) is the size of the learned feature vector, can be formulated as
where $W^p$, $W^d$, and $b^d$ are learnable parameters of this layer. Until now, the entire Feature Extraction Module can work with the goal to automatically learn the common latent features for the two following tasks.

## 5 DETAILS OF TWO LEARNING TASKS

In this section, we present the details of two learning tasks (outdoor position recovery and transportation mode detection) to process the learned features above.

### 5.1 Outdoor Position Recovery Task

As shown in Fig. 7, the position recovery task contains a Full Connection (FC) layer with two hidden units. Given the learned feature vector $V$, the position recovery task generates a trajectory $Y$ of MR positions, where $y_{i,j} \in Y$ is a predicted GPS position of $v_{i,j} \in V$. The output $y_{i,j}$ can be formulated as

$$y_{i,j} = \sigma(W^p v_{i,j} + b^p)$$

where $W^p$ and $b^p$ are learnable parameters, and the superscript $p$ indicates the task of position recovery. Due to the 2D GPS latitude and longitude coordinates, the output $y_{i,j}$ is a $2 \times 1$ vector $y_{i,j}$ regarding a certain MR sample $X_{i,j}$ (or the learned vector $v_{i,j}$ of $X_{i,j}$).

### 5.2 Transportation Mode Detection Task

In Fig. 8, the transportation mode detection task still takes the learned feature vector $V$ as input and generates a certain transportation mode for an input MR sample via a DNN classifier. The hidden layers of the DNN classifier consist of $T$ stacked FC layers. Slightly different from the position recovery task, the detection task requires a deeper network structure to perform the transportation mode estimation. In this task, the output $h_{i,j,m}$ of the $m$-th hidden FC layer can be written as

$$h_{i,j,m} = \sigma(W^h v_{i,j} + b_m), \quad m = 1$$

$$h_{i,j,m} = \sigma(W^h h_{i,j,m-1} + b^h_m), \quad m = \{2, \ldots, T\}. \quad (9)$$

$$m_{i,j} = \text{Softmax}(h_{i,j,m}), \quad m = T$$

In the equation above, $W^h_m$ and $b^h_m$ are the learned parameters by FC layers, and the superscript $t$ denotes the task of transportation mode detection. $m_{i,j} \in \mathbb{R}^C$ denotes the output of the final hidden layer with a Softmax activation function, where $C$ is a category set containing the possible transportation modes, e.g., walking, cycling, and driving in our datasets. In this way, we classify a transportation mode $m_{i,j}$ per MR sample. This classification differs from the previous works Monitor [3] and MonoSense [2] which classify a certain mode for an entire window of MR samples. In a fine-grained manner, our work can comfortably detect the modes for MR samples even with mixed transportation modes within an input sequence.

## 6 MODEL TRAINING

In this section, we first introduce the individual loss of each learning task, then give the joint loss of the entire learning framework, and finally present the training detail.

### 6.1 Individual Loss

First, since the outdoor position recovery problem is to predict numeric GPS coordinates, we model the problem as a regression task and compute the regression loss $L_{\text{loc}}^W(S)$ with model parameters $W$ on the input sequence $S$ as

$$L_{\text{loc}}^W(S) = \sum_{i=1}^{q} \sum_{j=1}^{|S_i|} \|y_{i,j} - y_{i,j}^{\text{true}}\|_2,$$

where $y_{i,j}^{\text{true}}$ is the ground truth location coordinates of the $j$-th MR sample in the $i$-th subsequence $S_i$.

Second, we model the transportation mode detection problem as a classification task. Thus, to define the loss function $L_{\text{mode}}^W$ of this task, we need to accumulate the mean cross-entropy for each MR sample in a sequence $S$

$$L_{\text{mode}}^W(S) = \sum_{i=1}^{q} \sum_{j=1}^{|S_i|} \sum_{c=1}^{C} \delta_{i,j}^c \log m_{i,j}^c. \quad (11)$$

In the equation above, $m_{i,j}^c$ is the predicted probability of the $c$-th MR sample in the subsequence $S_i$, and $\delta_{i,j}^c$ is a binary indicator (0 or 1) of the MR sample being the $c$-th transportation mode (i.e., ground truth).

Third, we note that the mobility features such as moving speed captured from predicted positions should be consist with the corresponding transportation mode. To this end, we introduce the speed constraint loss $L_{\text{speed}}^W(S)$ to associate the predicted positions with transportation modes

$$L_{\text{speed}}^W(S) = \sum_{i=1}^{q} \sum_{j=2}^{|S_i|} -\log (1 + P_{m_{i,j}}(\hat{v}_{i,j})), \quad (12)$$

In the equation above, the item $P_{m_{i,j}}(\hat{v}_{i,j})$ indicates the likelihood that the speed $\hat{v}_{i,j}$ follows the transportation mode $m_{i,j}$. Here, we estimate the speed $\hat{v}_{i,j}$ via the euclidean distance $||y_{i,j} - y_{i,j-1}||_2$ between two neighbouring positions $y_{i,j}$ and $y_{i,j-1}$. After that, given a speed probability distribution...
of a certain transportation mode $m_{i,j}$, the item $P_{m_{i,j}}(\hat{v}_{i,j})$ indicates the likelihood of the estimated speed $\hat{v}_{i,j}$ following the mode $m_{i,j}$.

Fig. 9 gives an example speed probability distribution of the driving mode in one of our datasets. Given a speed $\hat{v}_{i,j} = 4m/s$, by the speed probability distribution, we then have the probability $P_{m_{i,j}}(\hat{v}_{i,j}) = 0.238$. It indicates that the speed $\hat{v}_{i,j}$ follows the driving mode by the probability 0.238. From the equation above, if the predicted positions are more likely to follow the predicted transportation mode, we have a smaller negative loss $L^w_{\text{speed}}$, which will next minimize the joint loss $L^w$.

### 6.2 Joint Loss

As a simple way, we could define the joint loss by the sum of the three loss functions above. However, this way does not work well due to the significantly difference scale of these loss functions. To tackle this issue, by following the previous work [17], we compute a weighted joint loss function via the homoscedastic uncertainty weight for the prediction.

Note that the two learning tasks in PRNet+ generate numeric GPS coordinates $Y$ (i.e., continuous values) and a transportation mode $M$ (i.e., a discrete value). Following the work [17], we model such continuous and discrete values as a Gaussian likelihood and a Softmax likelihood, respectively, and have the following loss function with the model parameters $W$ on the input sequence $S$

$$L^w(S) = -\log p(Y, M|f^w)$$

$$= -\log N(Y; f^w, \sigma^2) \cdot \text{Softmax}(M; f^w, \sigma^2)$$

$$= \frac{1}{2\sigma_1^2} L^w_{\text{loc}}(S) + \frac{1}{2\sigma_2^2} L^w_{\text{mode}}(S) + \log \sigma_1 + \log \sigma_2,$$  \hspace{1cm} (13)

where $f^w$ denotes the sufficient statistic to derive a joint loss function by maximizing the Gaussian likelihood and $\sigma_1$, $\sigma_2$ are learnable parameters which can be treated as the task weights [17].

Besides the two loss functions $L^w_{\text{loc}}(S)$ and $L^w_{\text{mode}}(S)$, we still need to incorporate the speed-constraint loss $L^w_{\text{speed}}(S)$ into the final joint loss as follows:

$$L^w(S) = \frac{1}{2\sigma_1^2} L^w_{\text{loc}}(S) + \frac{1}{2\sigma_2^2} L^w_{\text{mode}}(S)$$

$$+ \gamma L^w_{\text{speed}}(S) + \log \sigma_1 + \log \sigma_2,$$  \hspace{1cm} (14)

where $\gamma$ is a pre-defined hyper-parameter (e.g., $\gamma = 0.05$ in our evaluation) for $L^w_{\text{speed}}(S)$ in the joint loss. We do not define a homoscedastic uncertainty weight for $L^w_{\text{speed}}(S)$. It is mainly because $L^w_{\text{loc}}(S)$ and $L^w_{\text{mode}}(S)$ are those loss functions regarding the two learning tasks with certain outputs and yet $L^w_{\text{speed}}(S)$ not.

### 6.3 Training Detail

To train the multi-task learning framework PRNet+, we split training data into mini-batches and pad MR sequences (both feature and label sequences) with a certain value e.g., -1. In this way, we make sure that they have the same length (mainly because the length of MR sequence with respect to IMSI differs in our datasets). In order to reduce the negative effect of the padding operation, we define a mask matrix $M$ where the matrix member $M_{i,j}$ is defined as

$$M_{i,j} = \begin{cases} 0, & y^\text{true}_{i,j} \text{ is a padding item,} \\ 1, & \text{otherwise.} \end{cases}$$  \hspace{1cm} (15)

We then have the following updated loss functions

$$L^w_{\text{mode}}(S) = \sum_{i=1}^{q} \sum_{j=1}^{|S|} M_{i,j} \delta_{c_{i,j}} \log m^c_{i,j}$$

$$L^w_{\text{loc}}(S) = \sum_{i=1}^{q} \sum_{j=1}^{|S|} M_{i,j} ||y_{i,j} - y^\text{true}_{i,j}||$$

$$L^w_{\text{cons}}(S) = \sum_{i=1}^{q} \sum_{j=2}^{|S|} -M_{i,j} \log (1 + P_{m_{i,j}}(\hat{v}_{i,j})).$$  \hspace{1cm} (16)

Until now, we give the training steps in Algorithm 1 where an Adam Optimizer is used.

**Algorithm 1. PRNet+ Training Procedure**

**Input** Training set $D$, Position recovery and mode detection ground truth: $Y^\text{true}$ and $M^\text{true}$, Training steps $T$, Batches $B$, Hyper-parameter $\alpha$

**Output** Model parameters $W, \sigma_1, \sigma_2$;

1: Initialize Model params $W, \sigma_1, \sigma_2$;
2: for $t = 1 \rightarrow T$ do
3: \hspace{1cm} for $b = 1 \rightarrow B$ do
4: \hspace{2cm} $V_{b,t} \leftarrow \text{FeatureExtract}(D_{b,t})$;
5: \hspace{2cm} $Y_{b,t} \leftarrow \text{PositionRecoveryTask}(V_{b,t})$;
6: \hspace{2cm} $M_{b,t} \leftarrow \text{ModeDetectionTask}(V_{b,t})$;
7: \hspace{2cm} $L^w \leftarrow \text{JointLoss}(Y_{b,t}, M_{b,t}, Y^\text{true}_{b,t}, M^\text{true}_{b,t}, \alpha)$;
8: \hspace{1cm} $W, \sigma_1, \sigma_2 \leftarrow \text{ADAM}(L^w, W, \sigma_1, \sigma_2)$;
9: end for
10: end for
11: end for

7 EXPERIMENTS

### 7.1 Experimental Setup

1) Data Sets. Our data sets are collected in three representative areas in Shanghai, China: a core business area Xuhui, an urban area Siping and a rural area Jiading. The geographical distances between Jiading and Siping, between Jiading and Xuhui, and between Siping and Xuhui are around 31 km, 37 km and 15 km, respectively. The Xuhui and Jiading-3 datasets were provided by one of the largest Telco operators in China, and the data sets for Jiading-1~2 and Siping were collected by our developed Android mobile app. When mobile users were moving around in outdoor road networks to
collect MR samples, they switched on GPS receivers on Android mobile phones to collect GPS coordinates. Since the collected GPS coordinates may contain noises, we employ a map-matching technique [13], [41] to mitigate the effect of noises. To protect user privacy, all sensitive information such as IMSI has been anonymized.

Table 2 summarizes the used data sets. Note that even for a certain area (e.g., Jiading), MR samples are collected from either 2G or 4G networks at rather different zones and by various collection systems (e.g., front-end Android App or back-end Telco operator base stations). We thus have totally 8 MR datasets collected from the 3 areas. For each dataset, we have collected MR samples and the associated GPS coordinates. Due to the limitations of the Android APIs in the 4G networks, each 4G MR sample in Jiading-1 and Siping contains only one (serving) base station without the information about other stations. Yet each 4G MR sample in Xuhui and Jiading-3, which are collected by Telco operator base stations, still contains up to 7 base stations. The transportation mode ground truth of MR samples in Jiading-1~2, Siping, and Xuhui datasets has been labelled. However, for the Jiading-3 dataset, the Telco operator does not provide transportation mode labels.

Table 3 clearly indicates that the data sets contain heterogeneous MR samples due to mixed transportation modes and uneven timestamp intervals between neighbouring samples. For example, in the Jiading-1 2G data set, the timestamp intervals vary from 1 to 125 seconds. It is mainly due to the uncertain delay of Android threads scheduled by Android OS to collect MR samples and (2) noisy MR samples (e.g., those samples having empty or zero signal measurements or empty base station IDs, and we have to clean the noise). Given these samples, Section 7.4 will study the effect of transportation modes and time intervals between neighboring samples.

Besides, we acquire base station GPS positions from the Telco operator, and use them as additional features of MR samples. However, the parameters such as antenna heights and angles of base stations are still unavailable. We believe that these parameters, if available, could improve PRNet*.

2) Counterparts. We compare PRNet* against 8 counterparts in Table 4 from the following aspects:

(a) Depending upon the location recovery result, the counterparts are either single point- (NBL, RaF [46], CCR [46] and a very recent work DeepLoc [26]), or sequence-based approaches (HMM [22], SeqtoSeq [30], ConvLSTM [25], PRNet [40] and PRNet*). In [46], the location recovery model can be either only a single-layer Random Forest (RaF), or a two-layer Random Forests (CCR) which can be treated as an implicit sequence-based approach due to the contextual features acquired from the predicted result of the 1st layer.

(b) In terms of the used models, the counterparts can be either the fingerprinting-based (NBL), or traditional machine learning-based (RaF, CCR and HMM), or DNN-based models (SeqtoSeq, ConvLSTM, DeepLoc, PRNet and PRNet*).

We follow the previous work [10], [24] to adopt the $k = 10$-fold cross validation by choosing 80% training and 20% testing data from each data set to avoid over-fitting. We compute the prediction error by the Euclidian distance between recovered locations and ground truth (i.e., real GPS coordinates of MR samples), we choose median error, mean error, and top 90% error (by sorting prediction errors in ascending order) as evaluation metrics. Since the
transportation modes of Jiading-3 dataset are unavailable, we exploit the previous transportation mode detection approach on GPS data [45] to predict the transportation modes of Jiading-3 dataset, and then the training MR samples of Jiading-3 dataset tagged by the predicted modes are used to train PRNet+ module for position recovery.

3) **Key (Hyper-) Parameters.** Table 5 lists the (hyper-) parameters used in our experiments. We use default values in the baseline experiment, and vary the values within allowable ranges for multi-task learning and sensitivity study. Due to the low sampling rate and short length of user trajectories in Jiading-3 dataset, we thus treat an entire user trajectory as one MR sequence.

### 7.2 Baseline Study

In Table 6, we compare the prediction errors (median, mean, and 90% errors) of the nine approaches on the eight datasets from three areas (Jiading, Siping, and Xuhui). From Table 6, we have the following findings.

1) **PRNet+** offers the least error among these nine approaches on all MR datasets. For example, in Jiading-1 2G data set, PRNet+ reduces the median error by 56.7% compared with the recent outdoor Telco position recovery approach, DeepLoc. It is mainly because DeepLoc is a simple point-based prediction method, which does not capture contextual dependencies. In addition, PRNet+ outperforms the two DNN-based approaches (i.e., SeqtoSeq and ConvLSTM). Such results indicate that PRNet+ outperforms both the state-of-the-arts and other DNN approaches.

2) Among the top-90% errors of these approaches e.g., in the Jiading-1 2G dataset, PRNet+ outperforms CCR with 58.1% lower top-90% error. Similar results are applicable to other datasets. These numbers indicate that PRNet+ has fully employed the power of hierarchical DNN, sequence model, attention mechanism and joint loss together to mitigate the outliers of predicted locations.

3) We compare the results of sequence-based approaches against single-point-based ones. For example, a simple sequence model, HMM, still outperforms DeepLoc by 31.0% smaller 90% error on Jiading-1 2G data set. In addition, though CCR and RaF are both Random Forest-based approaches, CCR leverages the contextual features such as moving speed and leads to better result than RaF by 10.9% smaller median error. Such results verify that the sequence models, no matter they implicit or explicit use of contextual features, could lead to better results than the original single-point-based approaches.

4) Among the DNN approaches, ConvLSTM suffers from the highest errors. It might be because ConvLSTM is typically used to solve flow prediction (e.g., precipitation nowcasting [25] and traffic accident [38]) for spatio-temporal time series data. However, MR samples do not contain accurate locations, which are just our objective (note that the GPS locations of base stations are used as additional features, but not the locations of MR samples). Thus, ConvLSTM may not properly solve our problem though ConvLSTM also utilizes the power of both CNN and LSTM. In addition, SeqtoSeq, though slightly better than ConvLSTM, still cannot compete with PRNet+.

5) Table 6 is consistent with the results reported by CCR and NBL: In the same area e.g., Xuhui, 4G MR datasets typically lead to better result than 2G datasets; and for the same Telco networks, the results of the Xuhui 2G dataset are better than those of Siping and Jiading-1~2 2G datasets. We have the finding that higher base station deployment density in 4G networks and core areas (see Table 2) can achieve better result than those in 2G and rural areas. Due to only one base station per MR sample in Jiading-1 and Siping 4G datasets, their errors are even higher than 2G datasets. Such result verifies the importance of multiple base stations for location recovery.

6) Among the datasets in Jiading area, the localization errors of Jiading-3 4G data set are higher than the other datasets. It is mainly caused by the lowest sampling rate among all datasets and more sparse base station density within the area covered by Jiading-3 data set. In addition, the localization errors of Jiading-2 2G data set are slightly higher than Jiading-1 2G data set. It is mainly because Jiading-2 data set has a higher percentage of MR samples collected by driving and cycling (see Table 2). We will discuss the effect of transportation modes in Section 7.4.

Due to space limit, we choose Jiading-1 and Jiading-2 2G datasets (the two datasets contain the greatest number of MR samples labelled by transportation modes) to evaluate PRNet+ in the rest of this section.

### 7.3 Evaluation of Multi-Task Learning Framework

In this section, we are interested in how the two individual learning tasks independently work against the multi-task learning framework PRNet+. In Table 7, the first two rows indicate the performance result of individual tasks alone, the third row involves those of the multi-task learning models with man-made uniform weights in the joint loss, and the rest rows give those with task uncertainty weighted loss functions.

Table 7 clearly demonstrates the benefit of the multi-task learning with much better results than an individual task alone. For example, using the multi-task learning framework, PRNet+ in the bottom row improves the classification accuracy from 75.4% to 78.4% in Jiading-1 2G dataset and the position recovery error (median error) from 15.8 meters to 15.2 meters. In addition, we compare

| Parameter | Range | Default Val. |
|-----------|-------|--------------|
| Parameter α in Joint Loss | 0.01, 0.05, 0.1 | 0.05 |
| Timestamp Interval (s) | 3–120 | 3 |
| Base Station Density | 25% - 100% | 100% |
| Learning Rate of PRNet+ | 0.0005, 0.0001, 0.00005, 0.000001 | 0.0001 |
| Module Layer | Hidden Units |
| Local Predictor | Local CNN | 64*1*8 |
| Global Predictor | Local LSTM | 128 |
| Task-1 Output FC Layer | 32*1*8 |
| Task-2 Output FC Layer | 2 |

### TABLE 5

| Parameter | Range | Default Val. |
|-----------|-------|--------------|
| Parameter α in Joint Loss | 0.01, 0.05, 0.1 | 0.05 |
| Timestamp Interval (s) | 3–120 | 3 |
| Base Station Density | 25% - 100% | 100% |
| Learning Rate of PRNet+ | 0.0005, 0.0001, 0.00005, 0.000001 | 0.0001 |
| Module Layer | Hidden Units |
| Local Predictor | Local CNN | 64*1*8 |
| Global Predictor | Local LSTM | 128 |
| Task-1 Output FC Layer | 32*1*8 |
| Task-2 Output FC Layer | 2 |

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
the uncertainty weights against the manual weights in the joint loss. Using equal weights leads to rather poor performance, even worse than the result of an individual task alone. Next, the 4th row (i.e., the loss regarding to the speed constraint is missed in the joint loss) cannot compete the bottom row, i.e., the multi-task uncertainty weights with the pre-defined parameter $g = 0.05$ for the speed constraint loss. It is mainly because using the speed constraint can effectively link the predicted positions with detected modes for better results.

### 7.4 Sensitivity Study

**(1) Transportation Mode.** Due to the mixed transportation modes within MR samples, we are interested in how these modes affect the performance of PRNet$^+$. 

![Fig. 10a evaluates the mode detection accuracy of PRNet$^+$ with two other approaches Monitor [3] and MonoSense [2]. From this figure, we find that PRNet$^+$ performs consistently better than Monitor and MonoSense. The two competitors impractically assume that the entire MR samples within a time window share a certain transportation mode, and do not work well when such MR samples involves mixed transportation modes. In addition, the detection accuracy of either walking or driving mode is significantly higher than the one of cycling. It is mainly because in our datasets, fast (resp. slow) cycling speed is close to the driving (resp. walking) speed, thus blurring the features regarding the cycling mode from those of either driving or walking modes. 

![Fig. 10b studies the benefit of transportation mode detection on position recovery. Specifically, we calculate how much localization error reduction is gained after the speed](image)

### TABLE 6

Baseline Experiment: Localization Errors of Nine Approaches

| Methods     | Jiading-1 (2G) | Jiading-1 (4G) | Jiading-2 (2G) | Jiading-3 (4G) | Siping (2G) | Siping (4G) | Xuhui (2G) | Xuhui (4G) |
|-------------|----------------|----------------|----------------|----------------|-------------|-------------|------------|------------|
| NBL Mean    | 53.4           | 67.2           | 300.9          | 59.7           | 68.2        | 64.7        | 298.3      | 42.8       |
| DeepLoc Mean| 35.1           | 47.6           | 250.3          | 40.2           | 49.3        | 54.2        | 219.9      | 33.5       |
| RaF Median  | 38.3           | 48.3           | 168.9          | 38.5           | 48.9        | 53.6        | 158.9      | 44.7       |
| CCR Mean    | 34.1           | 43.2           | 142.3          | 30.2           | 44.0        | 51.1        | 147.8      | 44.7       |
| HMM Median  | 36.5           | 52.3           | 172.8          | 42.1           | 52.4        | 56.2        | 184.4      | 40.3       |
| SeqtoSeq Mean| 25.4           | 50.6           | 85.3           | 24.2           | 49.5        | 54.1        | 88.4       | 43.7       |
| ConvLSTM Mean| 28.5           | 59.3           | 129.3          | 27.3           | 57.6        | 57.4        | 127.6      | 57.3       |
| PRNet Mean  | 15.8           | 37.8           | 63.2           | 18.4           | 40.6        | 45.5        | 70.4       | 34.2       |
| PRNet$^+$ Mean| 15.2           | 34.3           | 59.6           | 18.1           | 37.4        | 47.8        | 64.7       | 32.3       |

### Table 7

Multi-Task Learning in PRNet$^+$

| Loss | Jiading-1 2G Data | Jiading-2 2G Data |
|------|-------------------|-------------------|
|      | Pos. Recover. | Mode Detect. | 50% Error: meters | Accuracy | 50% Error: meters | Accuracy |
| Task (Loss item) Weights | Speed Const. | 1 | 0 | 15.8 | 16.7 | 16.7 |
| Pos. Recovery Task alone | / | 1 | 0 | 15.8 | 16.7 | 16.7 |
| Mode Detect. Task alone | / | 0 | 1 | / | 0.754 | / | 0.761 |
| Equal weights | 1 | 1 | 1 | 28.7 | 0.682 | 31.4 | 0.679 |
| Multi-task uncertainty weights | / | / | / | 17.1 | 0.775 | 17.9 | 0.752 |
| Multi-task uncertainty weights | 0.01 | / | / | 16.8 | 0.781 | 17.2 | 0.760 |
| Multi-task uncertainty weights | 0.1 | / | / | 17.4 | 0.781 | 18.1 | 0.766 |
| Multi-task uncertainty weights | 0.05 | / | / | 15.4 | 0.784 | 16.2 | 0.773 |

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
constraint loss is incorporated into the joint loss of PRNet+.

We find that the reduction of the 90% errors of PRNet+ is significant, indicating that the transportation mode detection greatly reduces position outliers.

Fig. 10c verifies how transportation modes affect the sequenced-based approaches. We can find that the walking mode leads to the best prediction accuracy. It is mainly because the MR samples in the walking mode exhibit higher spatial locality compared to cycling and driving, i.e., more samples within MR (sub)sequences. Note that the errors of both Seq2Seq and HMM grow significantly on the MR samples with the driving mode. It is mainly because these two models are hard to precisely capture spatio-temporal locality when MR samples are rather sparse in terms of their locations and uneven timestamp intervals,

(2) Timestamp Intervals. In this experiment, we study the effect of timestamp intervals (used by Equation (3) of time-interval attention in Section 4.2.1) on localization errors. From the MR samples of the Jiading-1 and Jiading-2 2G datasets, we randomly select MR samples to make sure that the timestamp difference between neighbouring MR samples in MR sequences is smaller than a certain time interval, and we vary the time intervals from 3 seconds to 120 seconds. From Fig. 11a, we have the following finding. First, a higher time interval (i.e., more sparse data samples) leads to higher prediction errors for all four sequence models. Second, the growth trends of median errors regarding PRNet+ and PRNet are rather smooth. It is mainly because they can capture temporal dependencies even from sparse samples. In addition, HMM suffers from much higher errors than Seq2Seq, indicating that the deep sequence models can better capture long-term dependencies than the first-order HMM model. Then, even with high timestamp intervals from 30 to 120 seconds, PRNet+ and PRNet still perform well, mainly due to the developed attention mechanism.

(3) Base Station Density. We now study how the density of base stations affects PRNet+ by varying the percentage of base stations from 25% to 100%. Specifically, among all base stations in a MR dataset, we randomly remove some base stations from MR samples. If a certain MR sample contains a removed base station, then the signal measurements regarding this base station are dropped from the sample. Fig. 11b indicates that PRNet+ leads to a competitive accuracy even if we drop 50% base stations. Nevertheless, a lower base station density means higher errors. The reason is that sparse base stations incur spatial ambiguity of MR samples. More base stations in MR samples alternatively lead to more discriminative MR features and lower errors.

(4) Ablation Study. We now study how the density of base stations affects PRNet+ by varying the percentage of base stations from 25% to 100%. Specifically, among all base stations in a MR dataset, we randomly remove some base stations from MR samples. If a certain MR sample contains a removed base station, then the signal measurements regarding this base station are dropped from the sample. Fig. 11b indicates that PRNet+ leads to a competitive accuracy even if we drop 50% base stations. Nevertheless, a lower base station density means higher errors. The reason is that sparse base stations incur spatial ambiguity of MR samples. More base stations in MR samples alternatively lead to more discriminative MR features and lower errors.

To study the importance of each component in PRNet+, we design the following variants. 1) PRNet using the local predictor of Feature Extraction Module alone (which can be treated as a single-point localization to process MR samples individually): we need to add an additional dense layer to produce the final outputs; 2) PRNet using the global predictor of Feature Extraction Module alone: we need to reshape the original input matrix \( X_{i,j} \in \mathbb{R}^{F \times N} \) of a MR sequence into a column vector \( (F \times N) \times 1 \), which is directly fed into the global predictor; 3) PRNet without Transportation Mode Detection Module. In Fig. 11c, PRNet+ suffers from the greatest errors, indicating the significant contribution offered by the global predictor to capture the temporal dependencies of an entire sequence. When compared to PRNet, PRNet+ reduces the top 90% errors, mainly because the learned dependencies among neighbouring MR samples are helpful to mitigate prediction outliers. Not surprisingly, PRNet+ performs better than the local and global predictors can lead to lower localization errors than the two variants. With help of transportation modes, PRNet+ outperforms all three variants.

(5) Learning Rate. Table 9 shows the effect of learning rates on PRNet+. Learning rate determines whether the objective (loss) function can converge to a local minimum and how fast it converges. Compared to the SGD optimizer used by PRNet, the Adam optimizer in PRNet+ is quite robust for the selection of model parameters. In this experiment, we vary the learning rate from 0.005 to 0.0001 under the same training epochs. Table 9 shows that low learning rates (e.g., 0.0005) lead to low errors.

(6) LSTM versus GRU. Recall that PRNet+ employs LSTM in the local recurrent layer, global bottom and upper recurrent layers. Beyond LSTM, the global bottom and upper recurrent layers in PRNet+ further adopt two attention mechanisms. To
study the effect of the attention mechanism in LSTM, we directly replace each of the entire recurrent layers by GRU as a variant of PRNet\(^+\). Fig. 11d indicates that PRNet\(^+\) (with LSTM) outperforms the variant with GRU. It is mainly because the attention mechanism in LSTM can better learn temporal locality from the MR samples involving uneven time intervals.

Nevertheless, GRU typically involves a smaller number of parameters and faster convergence time than LSTM. Thus, we meanwhile measure the training and prediction time of PRNet\(^+\) (with LSTM) against its variant with GRU and four deep learning-based counterparts (DeepLoc, SeqtoSeq, ConvLSTM and PRNet). All these approaches are evaluated on a server with AMD EPYC 7302 16-Core Processor, 32 GB memory and a GeForce RTX 3080 GPU. As shown in Table 8, DeepLoc, as a single-point localization approach, leads to the least training time. Instead, the other approaches, as sequential localization approaches, require around 2-hour training time. Compared with the original PRNet\(^+\), the variant PRNet\(^+\) using GRU requires slightly smaller training and prediction time. In addition, we note that the traditional sequential localization such as HMM, though not plotted in this table, suffers from rather high prediction time (e.g., 11.30s and 6.03s in the two datasets), mainly due to the high decoding cost of Viterbi algorithm. Until now, we have the following discussion. Though PRNet\(^+\) by default employs LSTM and two attention mechanism for the recurrent layers, we believe that other powerful recurrent models such as PredRNN [35], [36]) should work within our developed framework, typically involving the trade-off among localization errors, training time and prediction time.

### Table 8

| Methods     | DeepLoc | SeqtoSeq | ConvLSTM | PRNet | PRNet\(^+\) -GRU | PRNet\(^+\) |
|-------------|---------|----------|----------|-------|-----------------|-------------|
| Jiading-1 (2G) |         |          |          |       |                 |             |
| Training    | 1.24h   | 1.95h    | 1.89h    | 1.98h | 1.91h           | 2.08h       |
| Prediction  | 1.01s   | 1.54s    | 1.56s    | 1.82s | 1.82s           | 1.83s       |
| Jiading-2 (2G) |         |          |          |       |                 |             |
| Training    | 0.86h   | 1.49h    | 1.45h    | 1.69h | 1.64h           | 1.75h       |
| Prediction  | 0.88s   | 0.84s    | 0.88s    | 0.95s | 0.94s           | 0.95s       |

7.5 Visualization

Finally, Fig. 12 visualizes the trajectories recovered PRNet\(^+\) and three other approaches (PRNet, HMM, and DeepLoc) on a randomly selected small area from the Jiading-1 2G data set. As shown in this figure, the trajectory by PRNet\(^+\) is the closest to the ground truth in terms of horizontal moving directions along road segments and vertical shift out of road segments. For example, the 4th and 5th points to last in Fig. 12a are diversely distributed along the road and yet those in Fig. 12b are rather close. In particular, DeepLoc leads to the most shift in both horizontal and vertical directions. Though the median localization errors of HMM and DeepLoc are rather close, HMM leads to a smoother trajectory. It is mainly because HMM can better extract contextual dependencies from MR samples.

With help of the road network, we perform the mapmatching technique [44] on the predicted trajectories in order to generate smooth ones (denoted as PRNet\(^+\) \_M, PRNet\(_M\), HMM\(_M\), and DeepLoc\(_M\)). However, as shown in this figure, the map-matching technique cannot fully overcome the horizontal shift along road segments. Until now, we can clearly find that PRNet\(^+\) still leads to acceptable results even without this post-processing step.

### Table 9

| Learning Rate | 0.005 | 0.001 | 0.0005 | 0.0001 |
|---------------|-------|-------|--------|--------|
| Jiading-1 (2G) |       |       |        |        |
| Median Err. (meters) | 16.4  | 15.5  | 15.2   | 15.5   |
| Mean Err. (meters)    | 38.2  | 36.5  | 34.3   | 35.1   |
| 90% Err. (meters)     | 63.7  | 61.2  | 59.6   | 60.8   |
| Jiading-2 (2G) |       |       |        |        |
| Median Err. (meters) | 18.8  | 17.6  | 16.4   | 17.2   |
| Mean Err. (meters)    | 38.2  | 37.8  | 37.1   | 37.9   |
| 90% Err. (meters)     | 66.4  | 65.8  | 64.7   | 65.2   |

8 RELATED WORK

Trajectory Recovery. Unlike single point localization, sequence localization, namely trajectory recovery problem, frequently exhibits lower localization errors. For example, AT&T researchers [22] proposed an outdoor localization approach, which first exploits Hidden Markov Model (HMM) for trajectory recovery and then applies particle filters to generate smooth trajectories. With domain expert knowledge and feature engineering expertise, CCR [46] designed temporal and spatial contextual features such as moving speeds and time gap to greatly improve prediction accuracy. A recent work [47], with rather high localization precision, requires massive third-party GPS trajectories as the prior to precisely estimate the HMM emission probability. By enforcing the properties of spatial and temporal smoothness, the work [32] proposed a regularization framework to reconstruct mobile trajectories from sparse location fingerprints. Similar to [32], the variants of map-matching techniques [13], [41] attempt to recover an entire trajectory from sparse locations, e.g., GPS coordinates. The variants, with help of road

---

Fig. 12. Visualization (Map Scale 1.5:10000). Green: Ground Truth; Orange: Predicted Location; Blue: Map-matching location.
network constraints, project the locations onto road networks to correct outlier locations (e.g., noisy GPS points) and recover an entire trajectory. Unlike all these previous works, our work does not require the availability of either sparse GPS points within input trajectories or road networks.

**Deep Learning for Localization Systems.** Recently, deep neural networks (DNNs) have been used in indoor or outdoor localization. In dynamic indoor environments, Deep Belief Network (DBN) and Gaussian-Bernoulli Restricted Boltzmann Machines have been utilized in fingerprinting-based indoor localization to increase estimation accuracy and reduce generalization error [9]. To process the RSS time-series acquired from wireless local area network (WLAN) access points, the previous work [14] leveraged CNN to learn temporal dependencies from RSS readings for indoor localization. Rather than RSS, DeepFi [34] exploits channel state information (CSI) to develop a DNN indoor fingerprinting system. However, a fingerprinting database typically requires sufficient samples, and does not work well for our problem with heterogeneous samples. Finally, for outdoor Telco localization, DeepLoc [26] develops a data augmentation technique to overcome the issue of insufficient samples. In summary, these DNN-based works above typically perform single-point localization by learning spatio-temporal features [14]. Instead, we target sequence localization.

**Transportation Mode Detection.** Many literature works have studied the problem of transportation mode detection on GPS trajectory data. For example, the previous works [29], [45] first extract the features such as velocity and then train a machine-learning-based detection model. Some works employ deep learning techniques for transportation mode detection, e.g., the CNN-based and LSTM-based detection [8], [28]. Instead of GPS data, some works perform transportation mode detection via the readings from accelerometers and gyroscopes [6], [11].

**Similar to our work, Monitor** [3] and MonoSense [2] detect transportation modes on the data generated by cellular networks. With help of base station IDs and the associated received signal strength (RSSI), the two works extract statistical features such as velocity. However, due to the assumption that the entire MR samples within a time window share a certain transportation mode, they do not work well when MR samples involves mixed transportation modes. Instead, we detect the transportation mode on the granularity of individual MR samples by learning features rather than statistical features. Some works [23], [27] incorporate cellular data and other sensor data collected from mobile devices to detect transportation mode. CellTrans [43] infers that mobile users take either public transportation tools or private car at urban scale by extracting meaningful mobility features from cellular data. Such features are computed by the positions of connected base stations from Mobile flow records (MFRs) at a coarse-grained level (each trip of a user is tagged with a certain mode). Instead, we perform fine-grained transportation mode detection with one mode per sample.

**9 Conclusion**

In this paper, we proposed a multi-task learning framework PRNet. It assembles the power of CNN, LSTM, and two attention mechanisms to properly leverage all of the local, short- and long-term spatial and temporal dependencies to learn MR features. On the learned MR features, PRNet performs two learning tasks (outdoor location recovery and transportation mode detection) with help of a joint loss function. Our extensive evaluation shows that PRNet greatly outperforms state-of-the-art localization approaches and alternative variants of PRNet on the datasets collected in three representative areas in Shanghai. The promising result of PRNet inspires our future studies, such as the use of recovered trajectories from Telco MR data for human mobility analysis and the potentials of applying PRNet to indoor/outdoor localization in 5G networks.

**References**

[1] Google maps for mobile, 2019. [Online]. Available: http://www.google.com/mobile/maps
[2] A. M. AbdelAziz and M. Youssuf, “The diversity and scale matter: Ubiquitous transportation mode detection using single cell tower information,” in Proc. IEEE 81st Veh. Technol. Conf., 2015, pp. 1–5.
[3] A. Al-Husseiny and M. Youssuf, “RF-Based traffic detection and identification,” in Proc. IEEE Symp. Comput. Technol., 2012, pp. 1–5.
[4] R. A. Becker et al., “Human mobility characterization from cellular network data,” Commun. ACM, vol. 56, no. 1, pp. 74–82, 2013.
[5] R. A. Becker et al., “A tale of one city: Using cellular network data for urban planning,” IEEE Pers. Comput., vol. 10, no. 4, pp. 18–26, Apr. 2011.
[6] K. Chen, R. C. Shah, J. Huang, and L. Nachman, “Mago: Mode of transport inference using the hall-effect magnetic sensor and accelerometer,” Proc. ACM Interactive Mobile Wearable Ubiquitous Technol., vol. 1, no. 2, pp. 8:1–8:23, 2017.
[7] C. Costa and D. Zeinalipour-Yazti, “Telco big data: Current state & future directions,” in Proc. IEEE 19th Int. Conf. Mobile Data Manage., 2018, pp. 11–14.
[8] S. Dabiri and K. Heaslip, “Inferring transportation modes from GPS trajectories using a convolutional neural network,” 2018, arXiv:1804.02386.
[9] G. Felix, M. Siller, and E. Navarro-Alvarez, “A fingerprinting indoor localization algorithm based deep learning,” in Proc. IEEE 8th Int. Conf. Ubiquitous Future Netw., 2016, pp. 1006–1011.
[10] J. Hannink et al., “Mobile stride length estimation with deep convolutional neural networks,” IEEE J. Biomed. Health Inform., vol. 22, no. 2, pp. 438–447, Mar. 2018.
[11] L. Hemminki, P. Nurmi, and S. Tarkoma, “Accelerometer-based transportation mode detection on smartphones,” in Proc. 11th ACM Conf. Embedded Netw. Sensor Syst., 2013, pp. 13:1–13:14.
[12] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Comput., vol. 9, no. 8, pp. 1735–1780, 1997.
[13] Y. Huang, W. Rao, Z. Zhang, P. Zhao, M. Yuan, and J. Zeng, “Frequent pattern-based map-matching on low sampling rate trajectories,” in Proc. IEEE 19th Int. Conf. Mobile Data Manage., 2018, pp. 266–273.
[14] M. Ibrahim, M. Torki, and M. ElNainay, “CNN based indoor localization using RSS time-series,” in Proc. IEEE Symp. Comput. Commun., 2018, pp. 1044–1049.
[15] M. Ibrahim and M. Youssuf, “A hidden Markov model for localization using low-end GSM cell phones,” in Proc. IEEE Int. Conf. Commun., 2011, pp. 1–7.
[16] M. Ibrahim and M. Youssuf, “CellSense: An accurate energy-efficient GSM positioning system,” IEEE Trans. Veh. Technol., vol. 61, no. 1, pp. 286–296, Jan. 2012.
[17] A. Kendall, Y. Gal, and R. Cipolla, “Multi-task learning using uncertainty to weight losses for scene geometry and semantics,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 7482–7491.
[18] J. Caffery and G. L. Stubber, “Overview of radiolocation in CDMA cellular systems,” IEEE Commun. Mag., vol. 36, no. 4, pp. 38–45, Apr. 1998.
[19] I. Leontiadis, A. Lima, H. Kwak, R. Stanoeviche, D. Wetherall, and K. Papagiannaki, “From cells to streets: Estimating mobile paths with cellular-side data,” in Proc. 10th ACM Int. Conf. Emerg. Netw. Experiments Technol., 2014, pp. 121–132.
[20] R. Margolis et al., “Can you find me now? Evaluation of network-based localization in a 4G LTE network,” in Proc. IEEE Conf. Comput. Commun., 2017, pp. 1–9.
Yige Zhang received the BSc degree in software engineering from Tongji University, in July 2016. She is currently working toward the PhD degree with the School of Software Engineering, Tongji University, China, since September 2016. Her research interests focus on mobile computing and machine learning.

Weiixong Rao (Member, IEEE) received the PhD degree from the Chinese University of Hong Kong, in 2009. After that, he worked with the Hong Kong University of Science and Technology (2010), University of Helsinki (2011-2012), and the University of Cambridge Computer Laboratory Systems Research Group (2013) as a postdoctoral researcher. He is a full professor with the School of Software Engineering, Tongji University, China, since 2014. His research interests include mobile computing and spatiotemporal data science, and is a member of the CCF and ACM.

Kun Zhang received the PhD degree from the Chinese University of Hong Kong, in 2005. He is an associate professor with the Philosophy Department and an affiliate faculty member with the Machine Learning Department, Carnegie Mellon University. His research interests lie in machine learning and artificial intelligence, especially in causal discovery, causality-based reasoning, and general-purpose artificial intelligence. He coauthored a best student paper at UIAI 2010, received the best benchmark award of the causality challenge 2008, and coauthored a finalist best paper at CVPR 2019. He has served as an area chair or senior program committee member for major conferences in machine learning or artificial intelligence, including NeurIPS, UAI, ICML, AISTATS, AAAI, and UAI, and has organized various academic activities to foster interdisciplinary research in causality.

Lei Chen is a full professor with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology. His research interests include causality-based learning over social media, social media analysis, probabilistic and uncertain databases, and privacy-preserved data publishing. The system developed by his team won the excellent demonstration award in VLDB 2014. He got the SIGMOD Test-of-Time Award in 2015. He is a member of the VLDB Endowment and ACM distinguished scientist.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/doi.