A Utility-Aware General Framework With Quantifiable Privacy Preservation for Destination Prediction in LBSs

Hongbo Jiang, *Senior Member, IEEE*, Mengyuan Wang, Ping Zhao, *Member, IEEE*, Zhu Xiao, *Senior Member, IEEE*, and Schahram Dustdar, *Fellow, IEEE*

***Abstract*— Destination prediction plays an important role as the basis for a variety of location-based services (LBSs). However, it poses many threats to users' location privacy. Most related work ignores privacy preservation in destination prediction. Few studies focus on specific kinds of privacy-preserving destination prediction algorithms and thus are not applicable to other prediction methods. Furthermore, the third party involved in these studies is a potential privacy threat. Additionally, another line of related work regarding LBSs neither guarantees the utility of the predicted results nor provides quantifiable privacy preservation. To this end, in this paper, we propose a general framework that can provide quantifiable privacy preservation and obtain a trade-off between the privacy and the utility of the predicted results by utilizing differential privacy and a neural network model. Specifically, it first adopts a specially designed differential privacy to construct a data-driven privacy-preserving model that formulates the relationship between injected noise and privacy preservation. Then, it combines a Recurrent Neural Network and Multi-hill Climbing to add fine-grained noise to obtain the trade-off between the privacy preservation and the utility of the predicted results. Our extensive experiments on real-world datasets validate that the proposed framework can be applied to different prediction methods, provide quantifiable location privacy preservation, and guarantee the utility of the predicted results simultaneously.**

***Index Terms*— Destination prediction, privacy preservation, utility awareness, differential privacy, neural network.**

I. Introduction

WITH the popularity of embedded GPS devices and the rapid development of positioning technology, we benefit increasingly from various location-based services (LBSs) that function in all aspects of our lives, e.g., business, healthcare, and work [1], [2]. In recent years, a technology called destination prediction, which is a method used to predict the destinations of mobile users based on their given subtrajectories, has gradually matured and now significantly facilitates LBSs [3]–[6], as shown in Fig. 1. However, it entails location privacy disclosure at the same time.

Destination prediction provides LBSs with various application scenarios (cf. Fig. 1), e.g., automatically setting destinations before manual operations occur, sending targeted advertisements based on these destinations, and dispatching taxi orders based on passengers' destinations. Specifically, in 2016, Google Maps was embedded with destination prediction, assisting drivers in setting their destinations without manual operations [7]. Moreover, in 2017, "Offline Trajectories", an application announced by Facebook, was equipped with destination prediction technology, aiming at targeted advertisements and site recommendations [8]. In addition, to optimize the online car-hailing system, Uber and DiDi[1] apply destination prediction in their dispatching systems to raise the order acceptance rates of drivers and reduce the waiting time of passengers [9], [10].

Destination prediction has achieved high accuracy [10], and therefore, it poses many privacy threats to mobile users. Specifically, such destination prediction contributes to new crime scenes. For example, with background knowledge, i.e., the partial trajectory of a victim, an adversary can infer the victim's destination based on destination prediction frameworks, and this threatens the victim's privacy and security [3], [8]. Currently, the prediction accuracy is approximately 75% with regard to successfully predicting the victim's exact destination [10]. Furthermore, in 2016, Google faced a backlash for an implementation of Google Maps that collected large amounts of mobile users' trajectory information to infer users' exact destinations [8], [11].

Most related work has focused on predicted destinations (i.e., predicted results) with high accuracy, and these can be largely classified into three categories: Markov-based algorithms [3], [4], [12], neural network-based algorithms [6], [13], [14], and other framework-based algorithms [5], [10], [15]. However, these work ignored privacy preservation in destination prediction. In addition, only a few studies [3], [16], [17] concentrated on protecting users'

Manuscript received August 3, 2020; revised February 15, 2021 and May 12, 2021; accepted May 23, 2021; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor K. Ren. This work was supported in part by the National Natural Science Foundation of China under Grant U20A20181, Grant 61732017, and Grant 61902060; and in part by the Open Fund of Zhejiang Lab under Grant 2021LC0AB05. *(Corresponding authors: Ping Zhao; Zhu Xiao; Hongbo Jiang.)*

Hongbo Jiang, Mengyuan Wang, and Zhu Xiao are with the College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China (e-mail: hongbojiang2004@gmail.com; wmywang1997@foxmail.com; zhxiao@hnu.edu.cn).

Ping Zhao is with the College of Information Science and Technology, Donghua University, Shanghai 201620, China (e-mail: pingzhao2014ph@gmail.com).

Schahram Dustdar is with TU Wien, 1040 Vienna, Austria (e-mail: dustdar@infosys.tuwien.ac.at).

This article has supplementary downloadable material available at https://doi.org/10.1109/TNET.2021.3084251, provided by the authors.

Digital Object Identifier 10.1109/TNET.2021.3084251

[1]DiDi Chuxing ("DiDi") is a mobile transportation platform that offers a full range of app-based transportation services, including taxis, buses, and designated drivers.

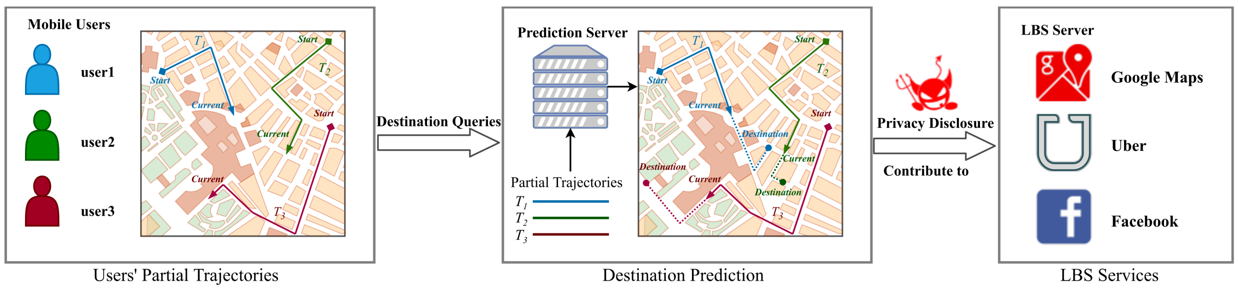


Fig. 1. Illustration of destination prediction services.

location privacy in destination prediction. However, these methods were designed for specific kinds of destination prediction algorithms and thus are not applicable to other prediction methods. Furthermore, the third party adopted in these studies is a potential privacy threat. Additionally, another line of related work mainly consists of three categories: cloaking-based algorithms [2], [18]–[21], dummy-based algorithms [1], [22]–[25] and differential privacy-based algorithms [26]–[29]. However, a fundamental limitation of these three kinds of techniques is that they definitively reduce the utility of the predicted results and cannot provide theoretically quantifiable privacy preservation.

To address the problem mentioned above, we propose a utility-aware general framework that provides quantifiable privacy preservation and obtains a trade-off between the privacy and the utility of the predicted results by utilizing *geo-indistinguishability* [26] (a specially designed type of differential privacy) and a neural network model. Our key observation is that the existing work concerning privacy preservation in destination prediction is prediction method-driven framework and thus is not applicable to other destination prediction methods. Therefore, we investigate a data-driven framework that protects users' location privacy by processing the users' trajectories, i.e., the inputs of destination prediction methods, and so it can be applied to multiple kinds of destination prediction methods. The main idea is that it first adopts a specially designed type of differential privacy to construct a data-driven privacy-preserving framework, in which Multiple Linear Regression is employed to formulate the relationship between the injected noise and privacy preservation, with the aim of providing quantifiable privacy preservation. Then, on this basis, we propose an optimized framework that combines Recurrent Neural Network and Multi-hill Climbing for adding fine-grained noise to obtain the trade-off between the privacy preservation and the utility of the predicted results.

Specifically, we first sample via adding noise of various scales into the trajectories and generate multiple samples from the trajectory dataset. Then, the raw samples and noisy samples are fed into the prediction model as inputs to obtain the multi-group prediction results, which are used to construct a data-driven model. Thereafter, Mltiple Linear Regression is used to fit the relationship between the noise scale and privacy protection. For the sake of the utility of the predicted results, we optimize the noise scales at each location along the sub-trajectories to obtain the trade-off between privacy preservation and the utility of the predicted results. Then, to solve such an optimization, we use Neural Arithmetic Logic Units (NALUs) [30] to formulate a neural network model and utilize Multi-hill Climbing to find a sub-optimal setting because of the huge overhead of fine-density traversal. Finally, we validate the performances of the proposed framework on two real-world datasets, i.e., the dataset from the *T-Drive* project [31] and a dataset from Kaggle [32], and four different prediction methods. The extensive results validate that our framework can be applied to different prediction methods, provide quantifiable location privacy preservation, and guarantee the utility of the predicted results at the same time.

In summary, we make the following *contributions* in this paper:

- To the best of our knowledge, this is the first work with proven guarantees for both users' location privacy and the utility of the predicted results, and it can be applied to several destination prediction methods.
- To obtain the trade-off between privacy preservation and the utility of the predicted results, we construct an optimization framework by employing NALUs [30] and Multi-hill Climbing.
- We conduct extensive experiments on the real trajectory dataset from the *T-Drive* project [31] and a dataset from Kaggle [32] with several of the latest prediction methods to investigate the effectiveness and generalizability of the proposed framework.

The remainder of the paper is organized as follows. Section II presents the preliminaries. Section III introduces the proposed framework for destination prediction. Thereafter, Section IV evaluates the performance of the proposed framework. Finally, we conclude the paper in Section V.

II. Preliminaries

In this section, we first present the employed trajectory data model. Then, we describe the attack model. Thereafter, we explain privacy preservation in destination prediction. Finally, we demonstrate the Laplacian mechanism.

A. Trajectory Data Model

In destination prediction, to process the given trajectory data, most related work [3], [4], [6] employs map matching to divide a trajectory into discrete cell regions in geographic space, and this is shown in Fig. 2. Motivated by these existing approaches, we adopt several definitions as follows:

Definition 1 (Movement): A user's *movement* is a dense position sequence composed of latitude and longitude coordinates $l_1, l_2, \ldots, l_c$, recorded by GPS, where l_c denotes the current location.

Definition 2 (Trajectory): Given a user's movement, we map the movement onto the grid space to obtain multiple

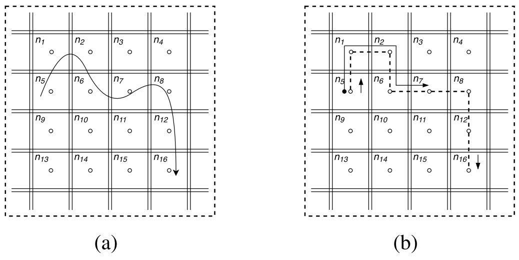


Fig. 2. Representing a trajectory with discrete grids. (a) The user's movement. (b) The trajectory is denoted by '- -' and a sub-trajectory is denoted by '-'.

adjacent grid cells $n_s,\ n_2, \ldots,\ n_d$, which form the *trajectory*, where n_s and n_d are the first and last grid cells, respectively.

As shown in Fig. 2(a), the movement starts from grid cell n_5 to cell n_{16} and goes through several cells n_1, n_2, n_6, n_7, n_8, n_{12}. Then, the movement is mapped to the trajectory $T = \{n_5,$ n_1, n_2, n_6, n_7, n_8, n_{12}, $n_{16}\}$, as shown in Fig. 2(b).

Definition 3 (Sub-Trajectory): We define a *sub-trajectory* as a partial trajectory in the grid space $T^P = \{n_1, n_2, \ldots, n_c\}$, where n_s and n_c denote the first and current grid cells, respectively.

For instance, in Fig. 2(b), the sub-trajectory $T^P = \{n_5, n_1,$ n_2, n_6, $n_7\}$ is a part of the trajectory $T = \{n_5$, n_1, n_2, n_6, n_7, n_8, n_{12}, $n_{16}\}$, where n_7 denotes the current cell.

In addition, we divide the trajectory data into three datasets D_P, D_{train}, and D_{test}, where D_P is used to train the destination prediction model. The other two datasets are devoted to training the proposed model and testing its corresponding performance.

B. Attack Model

The process of destination prediction consists of three steps. Step 1 involves users sending query information to the prediction server[2]; Step 2 includes the prediction of destinations in the prediction server; Step 3 involves returning several of the most likely destinations with their corresponding probabilities to the users. In general, a predicted destination with a larger probability is more likely to be the exact destination of the user.

The prediction server is considered untrusted, as in the existing work [33], [34]. Specifically, it receives users' query information and predicts users' destinations thereafter. However, at the same time, it monitors where users go over time, and it may disclose users' destinations to advertisers, illegal organizations, and so on, for commercial benefit. Moreover, we assume that the untrusted prediction server has the background knowledge that the predicted destination with the largest probability among the predicted destinations is most likely to be the exact destination of the user (hereafter victim). Note that for ease of presentation, we call the untrusted prediction server, advertisers, illegal organizations, etc. the "adversaries" hereafter.

Due to the leakage of destinations, the victim may be vulnerable to various threats. For instance, in the event of destination disclosure, sensitive personal information such as one's lifestyle, social relationships, and political beliefs can be easily revealed, thereby exposing the victim to spam or even blackmails and physical violence. Therefore, it is essential to preserve location privacy in destination prediction.

[2]The prediction server is an LBS server that can predict mobile users' destinations and provide location services based on users' destinations.

C. Privacy Preservation in Destination Prediction

Given N_t query trajectories (sub-trajectories) and a geographic space consisting of $g\times g$ regions, the destination prediction model generates N_t predicted results, each of which contains $g\times g$ predicted probabilities corresponding to the $g\times g$ regions. Each predicted result generates a ranking of the regions based on their predicted probabilities, and these can be intuitively obtained by users and adversaries. There are $g\times g$ ranks (i.e., 1st rank$, \ldots, g\times g$th rank) in a ranking; if the region corresponding to the ith rank is the exact destination, we consider it a successful prediction for the ith rank. To calculate prediction accuracy, we perform statistical computations. First, we accumulate the number of successful predictions at each of the $g\times g$ ranks. Then, we calculate $g\times g$ accuracy by dividing each number of successful predictions by the total number of predictions N_t. The $g\times g$ accuracies $P_1,\ P_2, \ldots,\ P_{g\times g}$ correspond to the 1st rank to the $g\times g$th rank. The accuracy at the ith rank is as follows:

$$P_i = \frac{Na_i}{N_t}, \tag{1}$$

where Na_i denotes the number of successful predictions for the ith rank.

Based on the accuracy for the ith rank P_i, we define the degree of privacy preservation as follows:

Definition 4 (Degree of Privacy Preservation): Based on the different privacy threats in the rankings, we define the degree of privacy preservation by employing the Weighted Absolute Error (WAE) as follows:

$$WAE = \sum_{i=1}^{N} \frac{1}{i} \mid P_i - P_i' \mid, \tag{2}$$

where P_i denotes the accuracy of the ith rank without preservation, P_i' represents the accuracy of the ith rank with preservation, N denotes the number of regions, and $\frac{1}{i}$ denotes the effect of the change in accuracy at the ith rank.

To preserve location privacy in destination prediction, our straightforward conception is to counter the adversary's background knowledge by reducing the accuracy rates at the top rankings. More specifically, with a reduction in the accuracy, the adversary cannot infer the victim's exact destination even with the help of the background knowledge. The reason for this is that the reduction in the accuracy for the top rankings makes the region with the highest probability no longer necessarily being the most likely exact destination of the victim. To achieve this conception, we propose the construction of a data-driven framework. Our framework first processes the sub-trajectories before the victim's query information is sent to the prediction server. In the prediction server, the processed sub-trajectories skew the predicted results to reduce the accuracy at the top rankings. However, the above processed sub-trajectories included in users' queries definitely affect the utility of the predicted results. Referring to a state-of-art work [3], [6], [14], we consider the total accuracy from the 1st rank to the kth rank (called the top-k accuracy) as the utility of the predicted results, where k generally depends on users' requirements. To this end, we define the utility loss as follows:

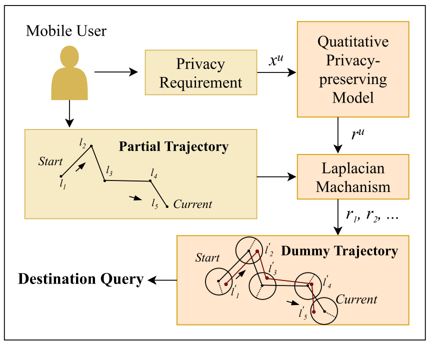


Fig. 3. Overview of the proposed general framework.

Definition 5 (Utility Loss): The utility loss (UL) is the decline of the predicted destinations' utility based on processed query information (i.e., sub-trajectories):

$$UL = 1 - \frac{P'_{top-k}}{P_{top-k}}, \tag{3}$$

where P_{top-k} and P'_{top-k} denote the sum of the accuracies from the 1th rank to the kth rank for the raw sub-trajectories and the processed sub-trajectories, respectively.

In Eq. (3), the closer the top-k accuracy of the processed sub-trajectories P'_{top-k} is to the top-k accuracy of the raw sub-trajectories P_{top-k}, the less utility loss. Note that we extend our framework to restrain the utility loss by maintaining the accuracy P'_{top-k} of the top-k rankings. Considering the enhancement of privacy and the restraint of the utility loss, our framework achieves a trade-off between the privacy and the utility of the predicted destinations.

III. The Proposed Framework for Destination Prediction

In this section, we first present the details of the proposed general framework with quantifiable privacy preservation. Then, we introduce the optimization of the proposed framework.

A. General Framework With Quantifiable Privacy Preservation

Overview. Considering a specific user's privacy requirements (i.e., the WAE) as the input, we obtain the output (i.e., the scale of the noise) from our model. To provide quantifiable privacy preservation, our model injects proper noise into the user's sub-trajectory with the Laplacian mechanism (introduced in Appendix A) based on the output. More specifically, we assume that the user's privacy requirement is x^u and that the sub-trajectory is $l_1^u, l_2^u, \ldots, l_c^u$ (query information). As demonstrated in Fig 3, our privacy preservation framework consists of several steps. First, our model absorbs x^u to generate the scale of the noise r^u. Then, we set multiple circular regions at each location in the sub-trajectory, where the centers of the circles are locations $l_1^u, l_2^u, \ldots, l_c^u$ and all radii are $r_1, r_2, \ldots, r_c$. Hereby, based on the Laplacian mechanism, r_i is generated as follows:

$$r_i = -\frac{1}{\epsilon}(W_{-1}(\frac{p-1}{e}) + 1) + r^u, \tag{4}$$

Thereafter, multiple dummy locations are randomly generated in each circular region instead of in the original locations to construct a dummy sub-trajectory. This well-designed dummy sub-trajectory can guarantee that the degree of privacy preservation is consistent with the user's privacy requirements.

To train our model, it is essential to prepare the processed training dataset (i.e., the WAE and the scale of the noise). In this section, we first introduce the generation method for the processed data. Then, we employ Multiple Linear Regression and propose an algorithm for selecting a subset of coefficients to build our model.

1) Preparation of Processed Data: As shown in Fig. 4, for the generation of processed data, there are three procedures: 1) Noise injection; 2) Multiple accuracy generation; and 3) Processed data generation.

Noise Injection. First, we make N copies of the training dataset D_{train}, thereby generating $N+1$ identical training datasets D_{train}. Then, we set a basic radius for the noise r_{base} (approximately 500 meters). Thus, different scales of noise can be controlled by $i \times r_{base}$, $i \in [0,\ 1, \ldots,\ N]$.

As demonstrated in the Overview, we inject different scales of noise into the N training datasets. In D^i_{train}, $T^i_j \Leftarrow i \times r_{base}$, $j \in [1,\ 2, \ldots,\ L_D]$, where D^i_{train} denotes the ith dataset, T^i_j denotes the jth movement in D^i_{train}, '$\Leftarrow$' represents the Laplacian noise injection and L_D is the size of D_{train}.

According to Eq. (4), with $i \in [0,\ 1, \ldots,\ N]$, we can obtain $N+1$ sets of training datasets with different noise injections corresponding to $N+1$ scales of noise, where $i = 0$ means that no noise is injected.

Multiple Accuracy Generation. We need to generate $N+1$ accuracy sets corresponding to the $N+1$ training datasets. Therefore, we put these $N+1$ noisy datasets into the destination prediction model to obtain $N+1$ sets of predicted results. As shown in Section II-C, each set of results contributes to a set of accuracies $P_1, P_2, \ldots, P_{g \times g}$, where $g \times g$ denotes the number of regions in the grid map. Hence, we obtain $N+1$ accuracy sets $P_0^{g \times g}$, $P_1^{g \times g}, \ldots, P_N^{g \times g}$, where $P_i^{g \times g} = \{P_{i,1},\ P_{i,2}, \ldots, P_{i,g \times g}\}$.

Processed Data Generation. Then, we generate the processed data. We know that the input of our model is the WAE mentioned in the Overview and that a set of accuracies can generate a value of the WAE according to Section II-C. The WAE is calculated as follows:

$$WAE_i = \sum_{j=1}^{g \times g} \frac{1}{j} \mid P_{0,j} - P_{i,j} \mid. \tag{5}$$

With $N+1$ accuracy sets, we generate $N+1$ WAE values $WAE_0, WAE_1, \ldots, WAE_N$, while the output of our model is the scale of the noise. Here, $N+1$ WAE values correspond to $N+1$ scales of noise $0, 1, \ldots, N$. Therefore, $N+1$ WAE values and $N+1$ scales of noise are combined to form the processed data for training our model.

2) Privacy-Preserving Model Training: We employ Multiple Linear Regression to fit the WAEs (i.e., the input variable x) and the scales of noise (i.e., the output variable y) included in the processed data and then propose an algorithm for building the model.

Multiple Linear Regression. In fact, x (WAE) and y (scale of noise) do not form a simple linear relationship. Thus, we consider the response variables of x (e.g., x^2 and $\log x$) and form the variable and its response variables into an input set of variables $\boldsymbol{x}_1,\ \boldsymbol{x}_2, \ldots,\ \boldsymbol{x}_q$. To fix the

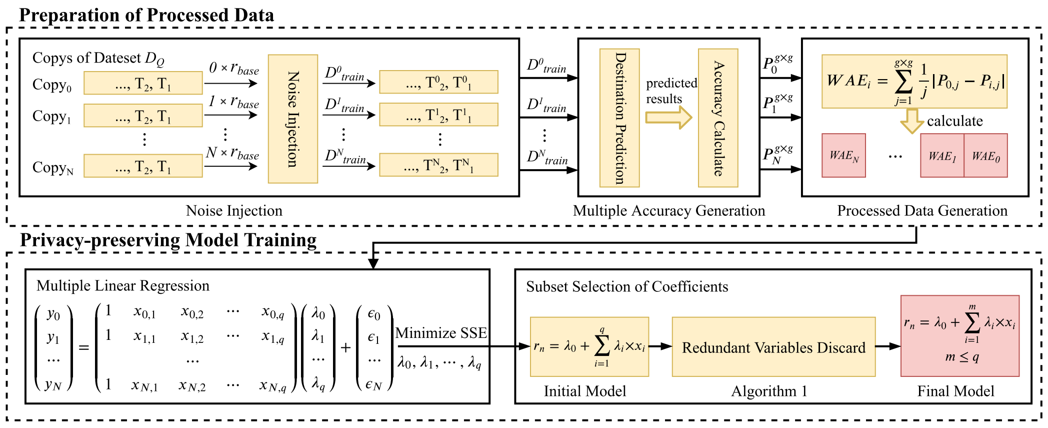


Fig. 4. The procedures for training the privacy-preserving model.

regression model, we assume there is a sequence of coefficients $\lambda_0,\ \lambda_1,\ldots,\ \lambda_q,$, and then, we derive multiple equations:

$$\begin{aligned} y_0 &= \lambda_0 + \lambda_1 x_{0,1} + \lambda_2 x_{0,2} + \ldots + \lambda_q x_{0,q} + \epsilon_0, \\ y_1 &= \lambda_0 + \lambda_1 x_{1,1} + \lambda_2 x_{1,2} + \ldots + \lambda_q x_{1,q} + \epsilon_1, \\ &\ldots \\ y_N &= \lambda_0 + \lambda_1 x_{N,1} + \lambda_2 x_{N,2} + \ldots + \lambda_q x_{N,q} + \epsilon_N, \end{aligned}$$

where ϵ_i denotes the random error of the ith dataset and $x_{i,j}$ represents the jth variable of the ith dataset.

$\boldsymbol{\lambda}$ would be perfectly fitted under several assumptions: *1)* $E(\epsilon_i) = 0$ for $i = 1, 2, \ldots,$n; *2)* var$(\epsilon_i) = \sigma^2$ for $i = 1, 2, \ldots,$n; and *3)* cov$(\epsilon_i, \epsilon_j) = 0$ for $i{\neq}j$. The first assumption shows that our regression model is linear and that there is no need for any additional terms or $\boldsymbol{\lambda}$. Then, the second assumption contributes to var$(y_i) = \sigma^2$, since $x_{i,1}$, $x_{i,2}$,..., $x_{i,q}$ are fixed in the first assumption. The last assumption ensures that there is no correlation between each pair of error terms ϵ_i and ϵ_j. Based on these assumptions, we adopt the least squares estimates of $\lambda_0,\ \lambda_1,\ldots,\ \lambda_q$, aiming to minimize the sum of squares of the deviations of the N observed $\boldsymbol{y}$:

$$SSE = \sum_{i=1}^{N}(\hat{y}_i - \hat{\lambda}_0 - \hat{\lambda}_1 x_{i,1} - \hat{\lambda}_2 x_{i,2} - \ldots - \hat{\lambda}_q x_{i,q})^2, \quad (6)$$

where $\hat{y}_i$ denotes the predicted y_i and $\hat{\lambda}_i$ denotes the ith coefficient among the optimal coefficients. In our method, we use the *scikit-learn* machine learning suite[3] to accomplish multiple linear regression (MLR) with two outputs (the SSE and the coefficients $\boldsymbol{\lambda}$).

Selection of a Subset of Coefficients. In fact, some response variables of x may be redundant and can be discarded. Hence, to simplify our privacy-preserving model, it is necessary to delete redundant response variables from the fitted model. The traditional method for doing so is to examine all possible subsets of x_1, x_2,..., x_q. However, it may be impractical to examine all possible subsets for a large number of permutations. Inspired by Hilton [35], we propose Algorithm 1 to select the subset of coefficients. The key idea of this algorithm is to operate the response variable $\boldsymbol{x}_i$ based on the correlation coefficient between $\boldsymbol{x}_i$ and $\boldsymbol{y}$.

[3]http://scikit-learn.org/stable/

In Algorithm 1, we set all the variables $\boldsymbol{x}_1,\ \boldsymbol{x}_2,\ldots,\boldsymbol{x}_q$, including $\boldsymbol{x}$ and the response variables, as inputs. In line 1, an empty array is defined to store the output coefficients, and A is an array of size q for storing all variables $\boldsymbol{x}_i,\ i \in [1, q]$. Then, we need to obtain the correlation coefficients between all variables and $\boldsymbol{y}$ (cf. lines 2-5):

$$R_j = \frac{\sum_{i=1}^{N}(x_{i,j} - \bar{x}_j)(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{N}(x_{i,j} - \bar{x}_j)^2 \sum_{i=1}^{N}(y_i - \bar{y})^2}}, \quad (7)$$

where $x_{i,j}$ denotes the value of the jth variable in the ith dataset and y_i denotes the value of the ith dataset.

To find the least relevant variables, a list R is set up to store the pairs of correlation coefficients and the orders of variables $\{R_1,\ 1\}$, $\{R_2,\ 2\},\ldots,$ $\{R_q,\ q\}$. Then, we sort R based on the correlation coefficient degrees.

Before subset selection is performed, Algorithm 1 obtains the SSE and initial $\boldsymbol{\lambda} = \lambda_0,\ \lambda_1,\ldots,\ \lambda_q$ based on MLR. Then, the deletion occurs on Lines 8-17 in Algorithm 1. First, the algorithm judges whether the correlation coefficient is greater than 0.5. If so, it means that the correlation between the current variable and $\boldsymbol{y}$ is strong, and the current variable and the rest of the variables are not deleted. Then, a new array is used to store partial variables, where one variable has been deleted. Then, $\boldsymbol{y}$ and the new array with partial variables are put into the MLR model to obtain the new SSE and the new coefficients. Thereafter, it is necessary to judge the sizes of the last SSE and the new SSE. If the size of the last SSE is larger than the new one, A is updated to A_t and the last set of coefficients $\boldsymbol{\lambda}$ is updated to the new set $\boldsymbol{\lambda}_t$, which means that the current variable is deleted. After q iterations, the final coefficients and subset of variables are obtained.

With the final coefficients $(\lambda_0,\ \lambda_1,\ldots,\ \lambda_m,\ m \leq q)$ and the subset of variables, we construct our model as follows:

$$r_n = \lambda_0 + \sum_{i=1}^{m}\lambda_i \times x_i, \quad (8)$$

where x_i is the response variable of the user's privacy requirement and r_n is the scale of the noise.

B. Optimization of the Proposed Framework

In the framework proposed above, the preservation of the user's location privacy will inevitably cause utility loss with

Algorithm 1 Selection of a Subset of Coefficients

```
Input  : Variables $\boldsymbol{x}_1,\ \boldsymbol{x}_2, \ldots,\ \boldsymbol{x}_q$
Output : Coefficients $\boldsymbol{\lambda}$ and a subset of variables
1  Initialize the sets $\boldsymbol{\lambda} = \emptyset$ and $A = \{\boldsymbol{x}_1, \ldots,\ \boldsymbol{x}_q\}$ ;
2  for $j = 1$ to $q$ do
3      $R_j = \frac{\sum_{i=1}^{N}(x_{i,j}-\bar{x}_j)(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{N}(x_{i,j}-\bar{x}_j)^2 \sum_{i=1}^{N}(y_i-\bar{y})^2}}$ ;
4      $R(j) = (R_j$, j) ;
5  end
6  Sort $R$ ;
7  $SSE,\ \boldsymbol{\lambda} =$ MLR($\boldsymbol{y},\ A$) ;
8  for $i = 1$ to $q$ do
9      if $R(i) > 0.5$ then
10         break;
11     end
12     $A_t = A - \boldsymbol{x}_{R(i,2)}$ ;
13     $SSE_t,\ \boldsymbol{\lambda}_t =$ MLR($\boldsymbol{y}, A_t$) ;
14     if $SSE_t < SSE$ then
15         $A = A_t$ ;
16         $\boldsymbol{\lambda} = \boldsymbol{\lambda}_t$ ;
17     end
18 end
19 Return $\boldsymbol{\lambda}$ and $A$ ;
```

respect to the predicted results. Hence, it is crucial to optimize the proposed privacy-preserving framework, with the aim of protecting location privacy and guarantee the utility of the predicted results at the same time.

In Section III-A, we inject the noise of the same scale into each location in a sub-trajectory. However, we observe that if we inject noise of different scales to different locations in each sub-trajectory, the utility of the predicted results can be guaranteed.

Therefore, the key idea of optimization is to inject fine-grained noise based on the comprehensive features of the training dataset. More specifically, we employ a neural network combining NALU [30], Multi-layer Perceptron (MLP) and Gated Recurrent Unit (GRUs) to extract features from the training dataset. Then, based on the built neural network, we search for the optimal model of noise injection.

In summary, this process includes three steps: 1) Pre-generating the training data; 2) Building the architecture of the neural network; and 3) Optimizing the noise injection model.

1) Pre-Generation of Training Data: To build the neural network, we need to obtain the training dataset. We first consider the input as the model of fine-grained noise and the output as the area under the curve constructed by the WAE and UL (called AUC$_{WU}$). Then, we generate the training data.

Model of Fine-grained Noise. We first introduce multiple coefficients μ_1, μ_2,..., μ_c in the method of noise injection, where μ_c denotes the coefficient of the current location. Therefore, the new method of noise injection is $\boldsymbol{r}\times r_{base} = \{\mu_i \times r_{base} \mid i \in [1,\ 2, \ldots,\ L_c]\ \}$, where $\boldsymbol{r}$ denotes the model of fine-grained noise and L_c denotes the length of the sub-trajectory.

However, sub-trajectories with different lengths require models with different sizes, and these cannot be absorbed by the neural network for an input of indefinite length.

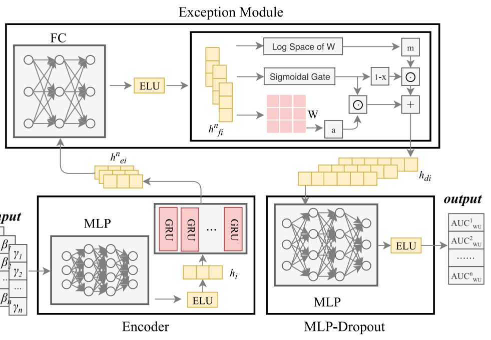


Fig. 5. The training of the area prediction model.

We observe that different locations in a sub-trajectory contribute to different influences on destination prediction. For instance, in destination prediction based on *Markov matrix*, the posterior probability is computed as $P(T^p|d{\in}n_j) = \frac{p_{12}\cdot p_{23}\cdots p_{(c-1)c}\cdot p_{c\to j}}{p_{s\to j}}$, where $p_{i(i+1)}$ denotes the transition probability between two adjacent nodes, and $p_{c\to j}$ and $p_{s\to j}$ represent total transition probabilities from l_c to l_j and from l_s to l_j, respectively. We observe that the current location l_c and the start location l_s have the largest effect on destination prediction. Thus, based on this observation, we adopt a restricted quadratic function to represent the model, where α, β, and γ are employed to denote the parameters of the function. That contributes to the input of our neural network.

The Area under WAE-UL Curve. In Section III-A.1, $N+1$ accuracies generate a set of WAE values WAE_0, WAE_1,..., WAE_N. In addition, with Eq. (3) in Section II-C, we can adopt the obtained $N+1$ accuracies to generate a set of UL values UL_0, UL_1,..., UL_N corresponding to WAE_0, WAE_1,..., WAE_N. Then, with the WAE as the horizontal axis and the UL as the vertical axis, we map the two sets of data in the Cartesian coordinate system, where we can obtain an area under the WAE-UL curve, i.e., the output of the neural network. Within a certain range of WAEs, the smaller AUC$_{WU}$ is, the lower the UL is.

With the input and output above, we first randomly generate M sets of models (α_1, β_1, γ_1),..., (α_M, β_M, γ_M). Then, from Section III-A.1, we know that each set of parameter models can generate $N+1$ sets of accuracies to compute $N+1$ values of WAEs and ULs, thereby generating a value of AUC$_{WU}$. Therefore, M sets of modes contribute to M values of AUC$_{WU}$, which constitutes the training data.

2) Building the Architecture of the Neural Network: As shown in Fig. 5, the architecture of the network consists of three integral components: *Encoder* module, *Exception* module, and *MLP-dropout* module.

Encoder Module. To extract the features from the data, Multi-layer Perceptron (MLP) is used to embed the input α_i, β_i, γ_i and obtain the hidden vector h^n_i with a fixed size, which contains exhaustive information about the training data. Then, Exponential Linear Units (ELUs) are chosen as the activation function to implement gradient descent and back propagation. Moreover, we operate Gated Recurrent Unit (GRU) cells for coefficient distribution changes in the hidden space. The hidden vector and the output of the *Encoder* module are:

$$h_i = \phi(\alpha_i, \beta_i, \gamma_i : W_{ee}), \tag{9}$$

$$h^n_{ei} = GRU(h^{n-1}_{ei}, h_i : W_{Encoder}), \tag{10}$$

where $\phi(\cdot)$ represents the embedding function with ELU nonlinearity, and W_{ee} and $W_{Encoder}$ are the embedding weight and the GRU cell weight, respectively.

Exception Module. In addition to the nonlinear relationship, there may also be a linear relationship between the model and AUC$_{WU}$. However, common neural networks seldom perform well outside of the range of numerical values encountered during training (e.g., linearity) [30]. Thus, to overcome this problem, we employ NALU to handle the linear relationships among features. In this module, a fully connected layer extracts features from the input hidden vector h_i^n. Thereafter, the NALU layer extracts the linearity by reconstructing basic arithmetic relationships:

$$h_{fi}^n = \omega(h_{ei}^n), \tag{11}$$
$$m = exp\boldsymbol{W}(\log | h_{fi}^n | + \epsilon), \tag{12}$$
$$g = \sigma(\boldsymbol{G}h_{fi}^n), \tag{13}$$
$$h_{di} = g \odot a + (1-g) \odot m, \tag{14}$$

where $\omega(\cdot)$ denotes the fully connected layer, $\boldsymbol{W} \in [-1,\ 1]$ is a weight factor, ϵ prevents the calculation of $\log 0$, and g is the learned sigmoidal gate in the NALU layer.

MLP-dropout Module. After the nonlinear and linear features are extracted, the *MLP-dropout* module decodes the hidden state h_{di}. Then, the network outputs the predicted AUC$_{WU}$ $\hat{y}_i$:

$$\hat{y}_i = \psi(h_{di} : W_{Decoder}), \tag{15}$$

where $\psi(\cdot)$ denotes the function of the MLP with ELU nonlinearity, and $W_{Decoder}$ is the weight of the decoder.

Algorithm 2 Optimal Model Searching

```
   Input  : The model $\alpha$, $\beta$, $\gamma$
   Output: Locally optimal model $\alpha_m$, $\beta_m$, $\gamma_m$
1  Initialize $G = \emptyset$ ;
2  Generate approximate gradients $G_\alpha,\ G_\beta,\ G_\gamma$ ;
3  $G = \{G_\alpha,\ G_\beta,\ G_\gamma\}$ ;
4  if $|G(1)| \le G_{min}$ and $|G(2)| \le G_{min}$ and $|G(3)| \le G_{min}$ then
5      $\alpha_m,\ \beta_m,\ \gamma_m = \alpha,\ \beta,\ \gamma$ ;
6  end
7  else
8      $\alpha^t = \alpha + G(1)$ ;
9      $\beta^t = \beta + G(2)$ ;
10     $\gamma^t = \gamma + G(3)$ ;
11     $\alpha_m,\ \beta_m,\ \gamma_m$ = OMS($\alpha^t$, $\beta^t$, $\gamma^t$) ;
12     return $\alpha_m,\ \beta_m,\ \gamma_m$ ;
13 end
14 return $\alpha_m,\ \beta_m,\ \gamma_m$ ;
```

3) Optimizing the Model of Noise Injection: To search for an optimal model of noise injection, we propose an algorithm employing Hill Climbing on the basis of the neural network built above. In Algorithm 2, we set a random model α, β, γ and the optimal model α_m, β_m, γ_m as the input and output, respectively. In Line 1, an empty array is defined to store the approximate gradients G_α, G_β, and G_γ of the model α, β, γ in the neural network. An approximate gradient is generated as follows:

$$G_i = \frac{\hat{y}_{i+\epsilon} - \hat{y}_{i-\epsilon}}{2\epsilon}, \tag{16}$$

where ϵ denotes a small but non-zero positive real number, and $\hat{y}_{i+\epsilon}$ and $\hat{y}_{i-\epsilon}$ denote the results output by the neural network based on the two processed models. These two models are obtained by adding and subtracting ϵ from the model α, β, γ for the ith parameter.

Then, it is necessary to determine whether the absolute values of all gradients in G are less than the threshold G_{min}. If so, the algorithm assigns the input model to the output model α_m, β_m, γ_m. Otherwise, α^t, β^t, and γ^t are used to represent the updated model (cf. Lines 6-8). Thereafter, the updated model is adopted as the input to perform function recursion until all gradients are below the threshold G_{min}. Finally, the algorithm returns the optimal model α_m, β_m, γ_m. However, the optimal model generated by this algorithm may be a local maximum. Therefore, we randomly generate multiple models as inputs and execute multiple algorithms to obtain multiple local maximum values. Then, we select the best model from these local maximum values.

IV. Performance Evaluation

In this section, we conduct extensive experiments on two real-world trajectory datasets [31], [32] to evaluate the performance of the proposed framework against the four attack models [3]–[6].

A. Experimental Setup

1) Datasets: We use two real-world taxi trajectory datasets from Kaggle [32] and *T-drive* project [31] in our experiments.

- *Dataset from Porto*. The dataset was collected from July 1st, 2013, to June 30th, 2014, and published by Kaggle [32]. This dataset is composed of more than 1.7 million trajectories of taxis in the center of Porto, which covers an area of 22 km × 22 km.
- *Dataset from Beijing*. This dataset is a common trajectory dataset provided by *T-drive* project [31] with a sparse number of trajectories collected from September 1st, 2013, to October 25th, 2013. We mainly extract 253,020 taxi trajectories from within the sixth ring road of Beijing. It covers an area of 52.28 km × 41.07 km.

The two datasets consist of different scales of trajectories, where the dataset from Porto is an extensive dataset and the dataset from Beijing is a sparse dataset. These datasets are used to demonstrate the robustness of our framework. Specifically, we randomly divide each dataset into three partial datasets: D_P (50% of the dataset), D_{train} (35% of the dataset), and D_{test} (15% of the dataset). D_P is used to train the attack models (i.e., destination prediction models). D_{train} and D_{test} are used to build our privacy-preserving model and then to demonstrate the privacy preservation ability of the proposed framework.

2) Attack Models: We explain the attack model in Section II-B, where we consider the destination prediction model as the attack model. Here, we select four attack models from different types of destination prediction models to show the performances of our framework against different attack models.

- *Sub-Trajectory Synthesis (SubSyn)* [3]. *SubSyn* is a classic destination prediction method that combines the Markov model and Bayesian inference to improve the coverage of trajectories in sparse datasets.

- *Trajectory Destination Prediction (T-DesP)* [4]. Based on the Markov model, *T-DesP* considers the temporal sensitivities of the trajectories and constructs a transition tensor to enhance the accuracy of the predicted results.
- *Trajectory Distribution-based Model* [5]. This model first obtains clusters of trajectories and then models the main traffic flow patterns by a mixture of 2-D Gaussian distributions to generate possible destinations and probabilities. For convenience, we call this model *"Distribution"* hereafter.
- *Hierarchical Trajectory-based Attentional LSTM Learning model (H-TALL)* [6]. *H-TALL* adopts a bidirectional LSTM network to capture spatial-temporal relations and employs Attention mechanism to increase the prediction accuracy.

3) Metrics: Our experimental results consist of two aspects: privacy preservation and utility loss. To evaluate the performance of our framework in terms of privacy preservation, we employ several metrics, namely, the degree of privacy preservation (WAE), Mean Absolute Error (MAE), Root Mean Square Error (RMSE) and Effective Rate (ER). Specifically, the WAE is defined in Section II-C as implying the degree of privacy preservation. The MAE and RMSE can well reflect the error of the value and measure the deviation, respectively, between the required WAE and the achieved WAE, which are formalized as follows:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|\hat{y_i} - y_i|, \tag{17}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y_i} - y_i)^2}, \tag{18}$$

where $\hat{y_i}$ and y_i denote the required WAE (i.e., privacy requirement) and the achieved WAE, respectively. The closer the MAE is to 0, the more precise the achieved WAE. The closer the RMSE is to 0, the smaller the deviation of the WAE.

Additionally, to quantify the privacy preservation of our framework more intuitively, we define the Effective Rate (ER) metric. First, we set a threshold ω to indicate the degree of tolerance of the user for the privacy error (i.e., the MAE). If MAE of the achieved WAE is less than ω, we call it "satisfactory privacy protection". Thus, the ER is defined as follows:

Definition 6 (Effective Rate (ER)): The Effective Rate is the proportion of "satisfactory privacy protection" instances in multiple experiments:

$$ER = \frac{N_s}{N}, \tag{19}$$

where N_s and N denote the number of "satisfactory privacy protection" instances and the total number of experiments, respectively.

Therefore, the higher the ER is, the higher the proportion of satisfactory predicted results is, and the more precise our quantitative privacy protection is.

To evaluate the utility loss incurred in our experiments, we adopt the definition UL (defined in Section II-C) to imply the utility loss of predicted results.

4) Parameter Settings: To build the proposed model, we randomly sample 450,000 trajectories in D_{train} from Porto (15,000 trajectories from Beijing). Then, to handle the preparation of the processed data obtained in Section III-A.1, we divide the data into 900 (50 for Beijing) sets of 300 trajectories, where each set is injected with noise of N=100 different scales to generate 100 values of the WAE corresponding to the 100 different scales of noise. Therefore, we obtain 900 (50 for Beijing) $\times$ 100 sets of processed data to build our general model. In addition, referring to work [3], [4], [6], we choose a sufficient grid granularity g=30 in the grid space, as this is a proper value in practice.

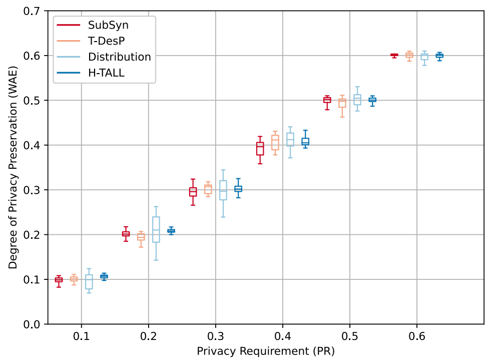
In our experiments, we consider the impacts of three parameters: the privacy requirement (PR), the privacy budget ϵ, and the trajectory completion percentage (tcp). The PR is the WAE of users' requirements. The goal of the proposed framework is to make the privacy preservation of the predicted result (i.e., the WAE) as close to the PR as possible. In addition, with the analysis of experimental results, we observe that ϵ and tcp have obvious impacts on the performance of the privacy preservation. Hence, we sample parameters of multiple scales to repeat the experiment for choosing proper values of the parameters. The default values of ϵ and tcp are 0.5 and 50%, which will be explained in Section IV-B.3.

B. Experimental Results

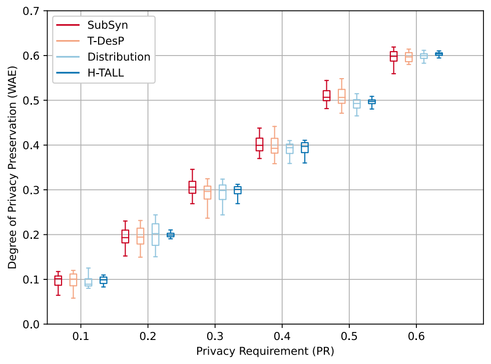
1) Privacy Preservation: We consider 300 predictions as one group to generate a WAE value and arrange 1000 experiments to demonstrate the performance of our privacy-preserving framework with different degrees of PRs. Here, we set the PR to be 0.1, 0.2, 0.3, 0.4, 0.5, and 0.6, respectively.

For the general framework (cf. Fig. 6), the degrees of privacy preservation (i.e., the WAEs) are very consistent with the different degrees of PRs. All WAEs are distributed near the targeted PRs, and this indicates that our framework can guarantee users' privacy requirements. Especially when PR $= 0.1$ and 0.6, the boxplots of the experimental results are particularly narrow. That is, we achieve highly precise privacy protection. In the boxplots, it is obvious that against different attack models, there are significant differences in the performances of our framework. For instance, we can clearly see that for the two datasets (cf. Figs. 6(a) and 6(b)), the box of *H-TALL* is narrower than those of the other attack models, which means that when against *H-TALL*, our framework performs best. In addition, by comparing Figs. 6(a) and 6(b), we can find that the boxes based on the dataset from Porto are narrower than those based on the dataset from Beijing against most of the examined attack models (i.e., *SubSyn*, *T-DesP* and *H-TALL*). This indicates that our framework performs much better on extensive datasets (e.g., the dataset from Porto) than on sparse datasets (e.g., the dataset from Beijing) in most cases. Despite the sparseness of the dataset, the WAEs obtained based on the dataset from Beijing are also precise, thereby proving that our framework can be effectively applied to datasets of different sizes. This proves the robustness of our privacy-preserving framework.

For the optimized framework (i.e., the utility-aware general framework), from Fig. 7, we can intuitively observe the similarities and differences between the performances of the optimized framework and the general framework. For instance, in the boxplots, the box at PR $= 0.2{\sim}0.5$ is wider than that at

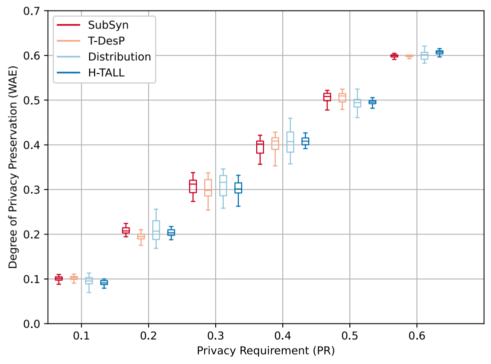


(a) The dataset from Porto.

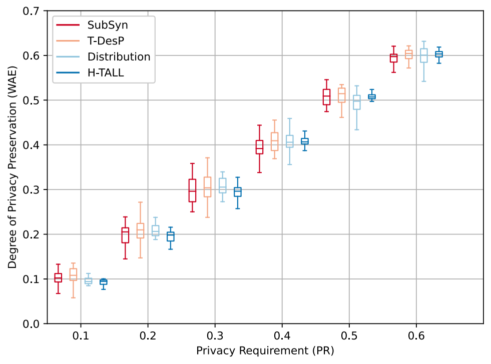


(b) The dataset from Beijing.

Fig. 6. Boxplots of the performances with multiple degrees of PR at $\epsilon = 0.5$ and tcp = 50% in the general privacy-preserving framework.



(a) The dataset from Porto.



(b) The dataset from Beijing.

Fig. 7. Boxplots of the performances with multiple degrees of PR at $\epsilon = 0.5$ and tcp = 50% in the optimized privacy-preserving framework.

TABLE I
MAE OF THE GENERAL FRAMEWORK

WAE	SubSyn		T-DesP		Distribution		H-TALL	
	Porto	Beijing	Porto	Beijing	Porto	Beijing	Porto	Beijing
0.1	0.005278	**0.010383**	0.005084	0.013401	0.014599	0.013494	0.006736	0.006915
0.2	0.005487	0.016718	0.008703	0.018637	0.031275	0.023555	0.008306	**0.004093**
0.3	0.010830	0.015467	0.009413	0.017327	0.023728	0.017285	0.007792	0.008895
0.4	0.014462	0.014038	0.016604	0.020219	0.018451	0.011768	0.009926	0.011137
0.5	0.006611	0.014190	0.010872	0.018848	0.012056	0.011488	0.004861	0.005072
0.6	**0.002049**	0.011831	**0.004822**	**0.009878**	**0.007090**	**0.005668**	**0.003449**	0.004138

PR = 0.1 or 0.6. The performance on the dataset from Porto is better than the performance on the dataset from Beijing. In addition, we can clearly observe that the boxes in Fig. 7 are slightly wider than the boxes in Fig. 6, which indicates that privacy preservation is only slightly decreased in the optimized framework.

More specifically, to demonstrate the performance in detail, we calculate the MAE and RMSE, as shown in Tables I and II and Tables III and IV, respectively. We enlarge the minimum MAE and RMSE in each setting. Table I demonstrates that with different datasets and attack models, our framework can achieve very low MAEs in terms of quantitative privacy preservation, indicating that the WAE provided by our framework can precisely meet the targeted PR. Especially on the dataset from Porto, most MAEs are less than 0.01, which implies that the general framework provides extremely precise quantitative privacy protection on this dataset. On the Beijing dataset, most MAEs are less than 0.02, and the smallest MAEs do not exceed 0.01, implying that the general framework achieves users' PRs with relative precision (especially when PR=0.1 or 0.6). In Table III, it is obvious that the MAE of the optimized framework only increases slightly, whereas the minimum MAE on the dataset from Porto against SubSyn is 0.002049 in Table I, but it increases to 0.003392 in Table III. By calculating the average MAE, we obtain that the average MAE of the general framework is approximately

TABLE II
RMSE OF THE GENERAL FRAMEWORK

WAE	SubSyn		T-DesP		Distribution		H-TALL	
	Porto	Beijing	Porto	Beijing	Porto	Beijing	Porto	Beijing
0.1	0.006627	**0.012938**	0.006030	0.016335	0.016617	0.016111	0.007753	0.008072
0.2	0.006923	0.020026	0.010938	0.022376	0.036246	0.027669	0.009276	**0.004814**
0.3	0.013534	0.018744	0.010211	0.022984	0.027470	0.021844	0.009834	0.011547
0.4	0.017779	0.016485	0.017985	0.023644	0.021497	0.015748	0.013719	0.014572
0.5	0.008189	0.018044	0.014985	0.022774	0.014004	0.014364	0.005974	0.006696
0.6	**0.002405**	0.018044	**0.005649**	**0.011405**	**0.009286**	**0.006745**	**0.004318**	0.004889

TABLE III
MAE OF THE OPTIMIZED FRAMEWORK

WAE	SubSyn		T-DesP		Distribution		H-TALL	
	Porto	Beijing	Porto	Beijing	Porto	Beijing	Porto	Beijing
0.1	0.004588	0.011830	0.004775	**0.015814**	0.008947	**0.007143**	0.008593	0.007690
0.2	0.009069	0.019277	0.008937	0.026604	0.021656	0.012418	0.006837	0.010570
0.3	0.016427	0.025764	0.018152	0.025564	0.023657	0.016242	0.014259	0.012032
0.4	0.012568	0.020328	0.016523	0.021267	0.023889	0.018575	0.009812	0.009473
0.5	0.010456	0.018435	0.011410	0.019196	0.012067	0.020049	**0.006067**	0.007696
0.6	**0.003392**	**0.011700**	**0.001757**	0.016594	**0.008016**	0.019276	0.007272	**0.007083**

TABLE IV
RMSE OF THE OPTIMIZED FRAMEWORK

WAE	SubSyn		T-DesP		Distribution		H-TALL	
	Porto	Beijing	Porto	Beijing	Porto	Beijing	Porto	Beijing
0.1	0.005555	**0.014915**	0.005583	**0.019200**	0.011157	**0.008290**	0.010182	0.009987
0.2	0.011219	0.023768	0.012205	0.034101	0.025670	0.015678	0.008130	0.013072
0.3	0.018525	0.029126	0.021173	0.031923	0.026515	0.019590	0.017048	0.015175
0.4	0.016719	0.025600	0.019392	0.025037	0.028158	0.024580	0.012114	0.012053
0.5	0.012051	0.021799	0.012832	0.021934	0.015794	0.027667	**0.008000**	0.009298
0.6	**0.004556**	0.015513	**0.002623**	0.025377	**0.009497**	0.024326	0.008288	**0.008435**

0.011519, and the average MAE of the optimized framework is approximately 0.013536, where the difference between the two average MAEs is approximately 0.002. Therefore, despite the slight increase in the MAE, our optimized framework can still guarantee users' PRs with precise WAEs.

In Table II, most RMSEs generally have very low degrees. Especially when the attack model is *H-TALL*, the average value of the RMSE does not exceed 0.01, which means that the deviations of our experimental results are very small, and our quantitative privacy protection is robust for guaranteeing users' PRs with minuscule deviation. Moreover, on the dataset from Porto with the *SubSyn* model and PR=0.6, the minimum RMSE is 0.002405, which is negligibly small with respect to users' PRs. Similarly, Table IV demonstrates that the similiar performance is achieved by the optimized framework. Specifically, we calculate two average RMSEs in Tables II and IV, and these are 0.014044 and 0.016654, respectively. That is, both our general framework and our optimized framework can guarantee users' PRs with robust privacy preservation in this case (i.e., the dataset from Porto with the *SubSyn* model and PR=0.6).

With different values of ω (0.01, 0.02, 0.03, 0.04, 0.05), we statistically calculate the ERs, as shown in Tables V. From the statistical results, we find that as ω increases, the ER increases. Specifically, when $\omega = 0.04$ and 0.05, the ER is extremely close to 100%, which means that with tolerant users (i.e., $\omega > 0.03$), the privacy protection provided by our framework can satisfy all users' privacy requirements. Moreover, when the threshold ω is harsh (i.e., $\omega < 0.02$), in Table V, we see that we can also satisfy most users (most ERs exceed 75%), and our optimized framework can satisfy half of the users' PRs. Furthermore, on an extensive dataset (i.e., the dataset from Porto), the ER value is higher than that on a sparse dataset (i.e., the dataset from Beijing), implying that our framework performs better on extensive datasets. In addition, even though the dataset from Beijing is highly sparse, our framework can still achieve an ER of 80% $(0.03 < \omega < 0.05)$.

TABLE V
ERs UNDER DIFFERENT VALUES OF ω FOR THE GENERAL FRAMEWORK AND OPTIMIZED FRAMEWORK

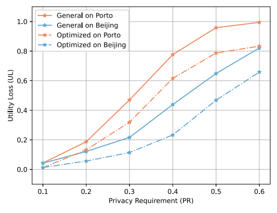
Framework	Threshold	SubSyn		T-DesP		Distribution		H-TALL	
		Porto	Beijing	Porto	Beijing	Porto	Beijing	Porto	Beijing
General Framework	ω=0.01	75.33%	42.67%	64.50%	35.67%	35.17%	47.50%	78.50%	83.17%
	ω=0.02	91.83%	75.33%	88.83%	69.17%	62.33%	75.83%	95.83%	95.17%
	ω=0.03	97.83%	92.00%	98.67%	85.50%	83.00%	87.83%	99.50%	98.33%
	ω=0.04	99.83%	98.50%	100.0%	94.17%	91.33%	95.33%	100.0%	100.0%
	ω=0.05	100.0%	100.0%	100.0%	99.00%	96.00%	99.17%	100.0%	100.0%
Optimized Framework	ω=0.01	64.00%	33.50%	59.50%	31.33%	40.50%	45.67%	62.33%	62.83%
	ω=0.02	88.50%	64.67%	84.67%	54.50%	66.33%	72.17%	92.83%	91.33%
	ω=0.03	96.33%	81.00%	95.00%	78.17%	82.83%	86.00%	99.00%	99.33%
	ω=0.04	99.67%	92.17%	99.17%	87.67%	94.17%	93.00%	100.0%	99.83%
	ω=0.05	100.0%	97.50%	100.0%	92.17%	98.17%	96.00%	100.0%	100.0%

TABLE VI
THE MAEs OF THE DEGREES OF PRIVACY PRESERVATION AND PRIVACY REQUIREMENTS

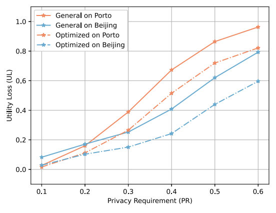
		Porto				Beijing			
		SubSyn	T-DesP	Distribution	H-TALL	SubSyn	T-DesP	Distribution	H-TALL
Equal	General	0.00745	0.00925	0.01787	0.00685	0.01377	0.01639	0.01388	0.00671
	Optimized	0.00942	0.01026	0.01637	0.00881	0.01789	0.02084	0.01562	0.00909
Random	General	0.00460 ∼ 0.01077	0.00645 ∼ 0.01206	0.01323 ∼ 0.02384	0.00551 ∼ 0.00832	0.01208 ∼ 0.01541	0.01304 ∼ 0.01861	0.01004 ∼ 0.01848	0.00503 ∼ 0.00880
	Optimized	0.00621 ∼ 0.01253	0.00605 ∼ 0.01470	0.01190 ∼ 0.02155	0.00710 ∼ 0.01125	0.01427 ∼ 0.02411	0.01783 ∼ 0.02447	0.01147 ∼ 0.01892	0.00779 ∼ 0.01125

Mixture of different PRs: In the above evaluation, we use the same PR value for the entirety of D_{test} in the experiments. However, in practice, we are more likely to encounter situations where different users select different PRs. For this use, we design two experiments: 1) In the first experiment (called *Equal*), we equally divide our D_{test} into 6 parts and set the PRs as 0.1, 0.2, 0.3, 0.4, 0.5, and 0.6, corresponding to the 6 parts of D_{test}. Then, we mix these 6 parts of D_{test} to evaluate the performance of the quantifiable privacy-preserving frameworks. 2) In the second experiment (called *Random*), we randomly generate the PR from 0.1 to 0.6 for each sub-trajectory in D_{test}. We then evaluate the performance achieved with random PRs. We employ the MAEs of the WAEs and PRs to evaluate the performance of our frameworks. Moreover, to demonstrate their effects more rigorously, we repeat the *Random* experiment 100 times and extract the minimum and maximum MAE values from the experimental results.

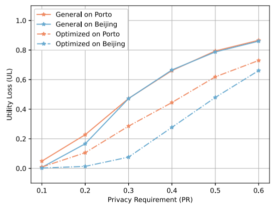
In Table VI, we can intuitively observe that all the MAE values are relatively low and that most of the MAEs do not exceed 0.02, indicating that with different PRs, the privacy preservation provided by our frameworks can still sufficiently satisfy users' privacy requirements. Moreover, in the *Random* experiment, the worst performance of our frameworks is approximately 0.02 (i.e., the maximum MAE). Comparing *Equal* and *Random*, we find that the MAE value in *Equal* is similar to the mean value of the maximum and minimum values in *Random*. For instance, in the condition (Porto, SubSyn, General), the MAE of *Equal* is 0.00745, and the mean MAE of *Random* is (0.00460+0.01077)/2 = 0.007685.



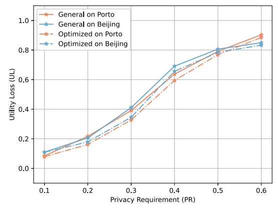
(a) Against SubSyn.



(b) Against T-DesP.

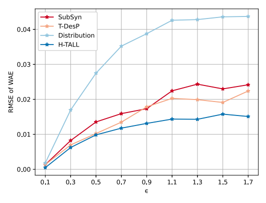


(c) Against Distribution.

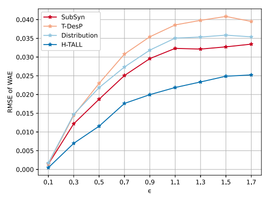


(d) Against H-TALL.

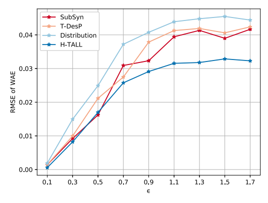
Fig. 8. Utility losses (UL) incurred by the general and optimized privacy-preserving frameworks against four attack models with two real datasets at $\epsilon = 0.5$ and tcp = 50%.



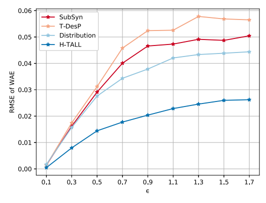
(a)



(b)

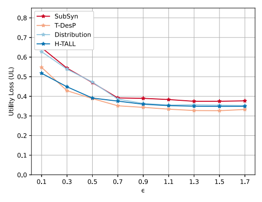


(c)

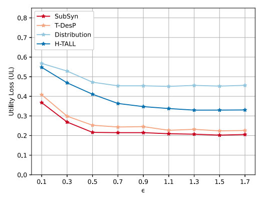


(d)

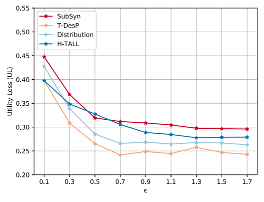
Fig. 9. The impact of ϵ on the RMSE (the deviation of the WAEs) with PR $= 0.3$ and tcp $= 50\%$. a) Experiment with the **general** framework on the dataset from **Porto**. b) Experiment with the **general** framework on the dataset from **Beijing**. c) Experiment with the **optimized** framework on the dataset from **Porto**. d) Experiment with the **optimized** framework on the dataset from **Beijing**.



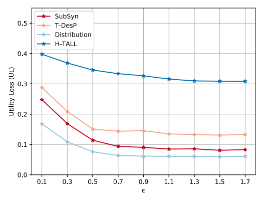
(a)



(b)



(c)



(d)

Fig. 10. The impact of ϵ on the utility loss (UL) with PR $= 0.3$ and tcp $= 50\%$. a) Experiment with the **general** framework on the dataset from **Porto**. b) Experiment with the **general** framework on the dataset from **Beijing**. c) Experiment with the **optimized** framework on the dataset from **Porto**. d) Experiment with the **optimized** framework on the dataset from **Beijing**.

This result prove that our framework performs similarly in *Equal* and *Random*. The reason is that the effects of privacy preservation for each user are independent of each other, and different users select different PRs, which have no influence on the effect of the privacy preservation that they obtained.

Therefore, our privacy-preserving frameworks can provide effective (low MAE) and robust (low RMSE) quantitative protection to satisfy users' privacy requirements. Although optimization does harm the performances of privacy preservation, our optimized framework is still able to achieve a very comparable the effect of privacy preservation compared with that of the general framework. Moreover, the two frameworks perform well on both the dataset from Porto and the dataset from Beijing, thereby proving the robustness of our frameworks.

2) *Utility Loss:* In the general framework, we do not consider the reduction of the utility of the predicted results. Through optimization, the optimized framework significantly improves the utility loss (cf. Fig. 8). In Fig. 8, the orange line and the blue line represent the experimental results based on the dataset from Porto and the dataset from Beijing, respectively, and the solid line and the dashed line represent the experimental results obtained by the general framework and optimized framework, respectively.

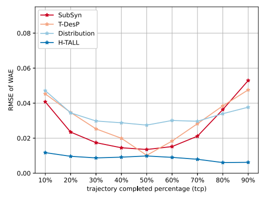
We can observe that as the PR increases, the UL increases. In particular, between PR=0.2 and 0.5, the UL increases more severely than it does in PR=0.1$\sim$0.2 and 0.5$\sim$0.6. For the general framework, at WAE=0.6, the UL is close to 1.0 for the dataset from Porto, which means that the utilities of the predicted results are completely destroyed and the predicted results are worthless for users.

The solid lines demonstrate the experimental results obtained by the optimized framework, where the UL is significantly reduced, especially when PR $= 0.2 \sim 0.6$ (at PR $= 0.1$, the UL is too low to be reduced heavily). The improvement in the UL is very obvious with most attack models (i.e., *SubSyn*, *T-DesP* and *Distribution*), and in *H-TALL*, there is a slight improvement in the UL. Moreover, on the dataset from Beijing and against *Distribution*, the UL declines by approximately 0.4 at PR $= 0.3$ and 0.4. Hence, compared to the general framework, the optimized framework does significantly improve the UL. So it can adequately satisfy users.

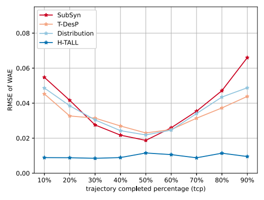
3) *Impacts of the Parameters:* Figs. 9 and 10 show the impact of ϵ on the RMSEs of the WAE and UL, respectively. In Fig. 9, as ϵ increases, the RMSE increases dramatically until $\epsilon = 0.9 \sim 1.7$. Comparing the impacts of ϵ in different figures, we can see that against the *H-TALL* model, as ϵ increases, the RMSE does not increase as severely as it does against other attack models. Moreover, in Figs. 9(c) and 9(d), the RMSE increases more severely with the optimized framework, which indicates that the privacy protection of the optimized framework is more sensitive to ϵ than that of the general framework. In addition, apart from Fig. 9(a), when $\epsilon > 0.7$, most RMSEs are larger than 0.03, which exceeds almost all RMSEs in Table II. Assume that the maximum RMSE of the stable quantitative privacy preservation framework is 0.030. Thus, ϵ must be less than 0.7.

Then, in Fig. 10, we observe that the UL decreases drastically when $\epsilon = 0.1 \sim 0.5$. Furthermore, there are severe utility losses at $\epsilon = 0.1$ (UL > 0.4 in Figs. 10(a) and 10(b)). In addition, it is obvious that the ULs in Figs. 10(c) and 10(d) are less than the ULs in Figs. 10(a) and 10(b). Considering the proper values of ϵ, we find that from $\epsilon = 0.5 \sim 1.7$, the utility of the predicted results is best retained (the lowest UL). Therefore, to choose a proper ϵ, we consider comprehensively that $\epsilon = 0.5$ is the optimal parameter in our experiments for the best RMSE range ($\epsilon = 0.1 \sim 0.5$) and the best UL range ($\epsilon = 0.5 \sim 0.7$).

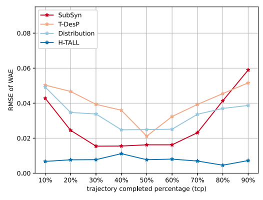
For the tcp, we plot Figs. 11 and 12 to show its impact on the RMSEs of the WAE and UL. In Fig. 11, the relationship between the tcp and RMSE is interesting, where against the most attack models (i.e., *SubSyn*, *T-DesP*, *Distribution*), from tcp $= 10\%$ to 50%, the trend is that as the tcp increases, the RMSE decreases. From tcp $= 50\%$ to 90%, as the tcp



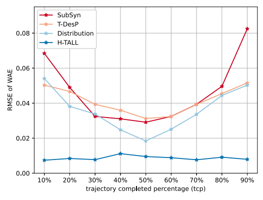
(a)



(b)

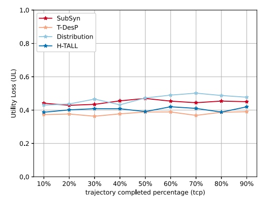


(c)

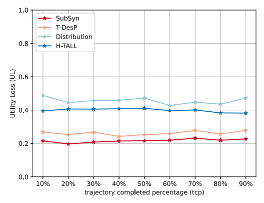


(d)

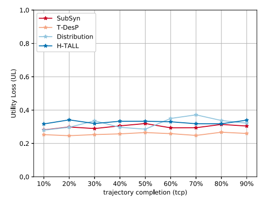
Fig. 11. The impact of the trajectory completion percentage (tcp) on the RMSE (the deviation of the WAEs) with PR $= 0.3$ and $\epsilon = 0.5$. a) Experiment with the **general** framework on the dataset from **Porto**. b) Experiment with the **general** framework on the dataset from **Beijing**. c) Experiment with the **optimized** framework on the dataset from **Porto**. d) Experiment with the **optimized** framework on the dataset from **Beijing**.



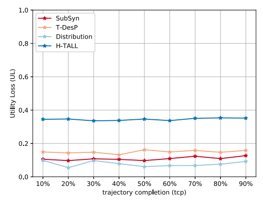
(a)



(b)



(c)



(d)

Fig. 12. The impact of the trajectory completion percentage (tcp) on the utility loss (UL) with PR = 0.3 and $\epsilon = 0.5$. a) Experiment with the **general** framework on the dataset from **Porto**. b) Experiment with the **general** framework on the dataset from **Beijing**. c) Experiment with the **optimized** framework on the dataset from **Porto**. d) Experiment with the **optimized** framework on the dataset from **Beijing**.

decreases, the RMSE increases. Hence, at tcp $= 50\%$, our framework can provide the most stable effect of quantitative privacy preservation. However, against *H-TALL*, there is almost no impact on the RMSE. Regarding the UL, there are distinct decreases in the ULs in Fig. 12(a) and 12(b). However, there is no obvious difference in the impacts of different tcp values on the utility loss. Hence, we set the default value of the tcp to 50%.

In summary, the results of various experiments demonstrate the excellent performance of our general framework and optimized framework. First, our frameworks can provide effective and robust quantitative privacy preservation. Second, compared with the general framework, the optimized framework greatly retains the utility of the predicted results. More specifically, with tolerant users (i.e., $\omega > 0.03$), our frameworks can satisfy almost all users. Even with harsh users (i.e., $\omega < 0.02$), our frameworks can provide sufficient privacy preservation with low deviations of the WAEs to satisfy half of the users. Moreover, due to optimization, there is a significant improvement in the utility loss of the optimized framework, and this contributes to the utility-aware and quantitative privacy preservation.

V. Conclusion

This paper proposes a utility-aware general framework that can provide quantifiable privacy preservation, obtain a trade-off between privacy and the utility of the predicted results, and be applicable to several kinds of destination prediction methods. We first construct a general privacy-preserving framework based on a specially designed type of differential privacy and Multiple Linear Regression. Then, on this basis, an optimized framework utilizing an RNN and Multi-hill Climbing is proposed. Finally, we evaluate the proposed frameworks on the two real-world trajectory datasets and four attack models, and the extensive results validate that the proposed frameworks outperform the methods in existing work.

References

[1] P. Zhao *et al.*, "Synthesizing privacy preserving traces: Enhancing plausibility with social networks," *IEEE/ACM Trans. Netw.*, vol. 27, no. 6, pp. 2391–2404, Dec. 2019.

[2] P. Zhao *et al.*, "P3-LOC: A privacy-preserving paradigm-driven framework for indoor localization," *IEEE/ACM Trans. Netw.*, vol. 26, no. 6, pp. 2856–2869, Dec. 2018.

[3] A. Y. Xue, R. Zhang, Y. Zheng, X. Xie, J. Huang, and Z. Xu, "Destination prediction by sub-trajectory synthesis and privacy protection against such prediction," in *Proc. IEEE ICDE*, Apr. 2013, pp. 254–265.

[4] X. Li, M. Li, Y.-J. Gong, X.-L. Zhang, and J. Yin, "T-Des.: Destination prediction based on big trajectory data," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 8, pp. 2344–2354, Aug. 2016.

[5] P. C. Besse, B. Guillouet, J.-M. Loubes, and F. Royer, "Destination prediction by trajectory distribution-based model," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 8, pp. 2470–2481, Aug. 2018.

[6] J. Zhao, J. Xu, R. Zhou, P. Zhao, C. Liu, and F. Zhu, "On prediction of user destination by sub-trajectory understanding: A deep learning based approach," in *Proc. 27th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2018, pp. 1413–1422.

[7] N. Bogart. *Google Maps Will Soon Predict Where You are Going*. Accessed: Jan. 14, 2016. [Online]. Available: https://globalnews.ca/news/2453460/google-maps-will-soon-predict-where-you-are-going/

[8] N. Nguyen, *Facebook Filed a Patent to Calculate Your Future Location*. Accessed: Dec. 10, 2018. [Online]. Available: https://www.buzzfeednews.com/article/nicolenguyen/facebook-location-data-prediction-patent

[9] L. Zhang, W. Ai, C. Yuan, Y. Zhang, and J. Ye, "Taxi or hitchhiking: Predicting passenger's preferred service on ride sharing platforms," in *Proc. ACM SIGIR*, Jun. 2018, pp. 1041–1044.

[10] L. Zhang *et al.*, "A taxi order dispatch model based on combinatorial optimization," in *Proc. ACM SIGKDD*, Aug. 2017, pp. 2151–2159.

[11] K. Tiffany. *Advertisers Can Easily Track You, Your Kid, Your Doctor, and the President*. Accessed: Dec. 11, 2018. [Online]. Available: https://www.vox.com/the-goods/2018/12/11/18136361/

[12] Z. Yang *et al.*, "An efficient destination prediction approach based on future trajectory prediction and transition matrix optimization," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 2, pp. 203–217, Feb. 2020.

[13] A. de Brébisson, É. Simon, A. Auvolat, P. Vincent, and Y. Bengio, "Artificial neural networks applied to taxi destination prediction," 2015, *arXiv:1508.00021*. [Online]. Available: http://arxiv.org/abs/1508.00021

[14] J. Lv, Q. Sun, Q. Li, and L. Moreira-Matias, "Multi-scale and multi-scope convolutional neural networks for destination prediction of trajectories," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 8, pp. 3184–3195, Aug. 2020.

[15] L. Wang, Z. Yu, B. Guo, T. Ku, and F. Yi, "Moving destination prediction using sparse dataset: A mobility gradient descent approach," *ACM Trans. Knowl. Discovery Data*, vol. 11, no. 3, pp. 1–33, Apr. 2017.

[16] D. Xue, L.-F. Wu, H.-B. Li, Z. Hong, and Z.-J. Zhou, "A novel destination prediction attack and corresponding location privacy protection method in geo-social networks," *Int. J. Distrib. Sensor Netw.*, vol. 13, no. 1, Jan. 2017, Art. no. 155014771668542.

[17] M. Zhang, "Research on travel destination prediction and privacy protection based on trajectory data," M.S. thesis, College Comput. Sci. Technol., Jilin Univ., Changchun, China, 2018.

[18] B. Niu, Q. Li, X. Zhu, G. Cao, and H. Li, "Achieving k-anonymity in privacy-aware location-based services," in *Proc. IEEE INFOCOM*, Apr. 2014, pp. 754–762.

[19] L. Jin, B. Palanisamy, and J. B. D. Joshi, "POSTER: Compromising cloaking-based location privacy preserving mechanisms with location injection attacks," in *Proc. ACM CCS*, Nov. 2014, pp. 1439–1441.

[20] H. Jiang, P. Zhao, and C. Wang, "RobLoP: Towards robust privacy preserving against location dependent attacks in continuous LBS queries," *IEEE/ACM Trans. Netw.*, vol. 26, no. 2, pp. 1018–1032, Apr. 2018.

[21] S. Zhang, X. Mao, K.-K.-R. Choo, T. Peng, and G. Wang, "A trajectory privacy-preserving scheme based on a dual-K mechanism for continuous location-based services," *Inf. Sci.*, vol. 527, pp. 406–419, Jul. 2020.

[22] V. Bindschaedler and R. Shokri, "Synthesizing plausible privacy-preserving location traces," in *Proc. IEEE SP*, May 2016, pp. 546–563.

[23] X. Gong, X. Chen, K. Xing, D.-H. Shin, M. Zhang, and J. Zhang, "From social group utility maximization to personalized location privacy in mobile networks," *IEEE/ACM Trans. Netw.*, vol. 25, no. 3, pp. 1703–1716, Jun. 2017.

[24] H. Liu, X. Li, H. Li, J. Ma, and X. Ma, "Spatiotemporal correlation-aware dummy-based privacy protection scheme for location-based services," in *Proc. IEEE INFOCOM*, May 2017, pp. 1–9.

[25] J. Li *et al.*, "Drive2friends: Inferring social relationships from individual vehicle mobility data," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 5116–5127, Jun. 2020.

[26] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: Differential privacy for location-based systems," in *Proc. ACM CCS*, 2013, pp. 901–914.

[27] Y. Xiao and L. Xiong, "Protecting locations with differential privacy under temporal correlations," in *Proc. ACM CCS*, Oct. 2015, pp. 1298–1309.

[28] L. Wang, D. Yang, X. Han, T. Wang, D. Zhang, and X. Ma, "Location privacy-preserving task allocation for mobile crowdsensing with differential geo-obfuscation," in *Proc. WWW*, Apr. 2017, pp. 627–636.

[29] Z. Ma, T. Zhang, X. Liu, X. Li, and K. Ren, "Real-time privacy-preserving data release over vehicle trajectory," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 8091–8102, Aug. 2019.

[30] A. Trask, F. Hill, S. E. Reed, J. Rae, C. Dyer, and P. Blunsom, "Neural arithmetic logic units," in *Proc. NIPS*, 2018, pp. 8035–8044.

[31] (Apr. 2008). *T-Drive Trajectory Data Sample*. [Online]. Available: https://www.microsoft.com/en-us/research/publication/t-drive-trajectory-data-sample/

[32] (Apr. 2015). *Kaggle Data Set ECML/PKDD 15: Taxi Trajectory Prediction (1)*. [Online]. Available: https://www.kaggle.com/c/pkdd-15-predict-taxi-service-trajectory-i/data

[33] R. Shokri, G. Theodorakopoulos, P. Papadimitratos, E. Kazemi, and J.-P. Hubaux, "Hiding in the mobile crowd: Locationprivacy through collaboration," *IEEE Trans. Depend. Sec. Comput.*, vol. 11, no. 3, pp. 266–279, Jun. 2014.

[34] H. Jiang, J. Li, P. Zhao, F. Zeng, Z. Xiao, and A. Iyengar, "Location privacy-preserving mechanisms in location-based services: A comprehensive survey," *ACM Comput. Surv.*, vol. 54, no. 1, p. 36, 2021.

[35] D. K. Hilton, "A simulation and comparison of three criterion functions used for subset selection in linear regression," Ph.D. dissertation, Dept. Statist., Brigham Young Univ., Provo, UT, USA, 1983.

Hongbo Jiang (Senior Member, IEEE) received the Ph.D. degree from Case Western Reserve University in 2008.

He was a Professor with the Huazhong University of Science and Technology. He is currently a Full Professor with the College of Computer Science and Electronic Engineering, Hunan University. His research concerns computer networking, especially algorithms and protocols for wireless, and mobile networks. He is serving as an Editor for IEEE/ACM TRANSACTIONS ON NETWORKING, an Associate Editor for IEEE TRANSACTIONS ON MOBILE COMPUTING, and an Associate Technical Editor for *IEEE Communications Magazine*.

Mengyuan Wang is currently pursuing the master's degree with the College of Computer Science and Electronic Engineering, Hunan University, China.

His research of interests include information security and mobile computing.

Ping Zhao (Member, IEEE) received the B.E. degree from the Tianjin University of Science and Technology, China, in 2013, and the Ph.D. degree from the School of Electronic Information and Communications, Huazhong University of Science and Technology, in 2018. Thereafter, she joined the faculty of Donghua University, where she is currently an Assistant Professor. Her research of interests are in the area of mobile computing, information security, and the Internet of Things.

Zhu Xiao (Senior Member, IEEE) received the M.S. and Ph.D. degrees in communication and information system from Xidian University, China, in 2007 and 2009, respectively.

From 2010 to 2012, he was a Research Fellow with the Department of Computer Science and Technology, University of Bedfordshire, U.K. He is currently an Associate Professor with the College of Computer Science and Electronic Engineering, Hunan University, China. His research interests include mobile communications, wireless localization, the Internet of Vehicles, and trajectory data mining. He is serving as an Associate Editor for IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS.

Schahram Dustdar (Fellow, IEEE) received the Ph.D. degree in business informatics from the University of Linz, Austria, in 1992.

He is currently a Full Professor of computer science (informatics) with a focus on internet technologies heading the Distributed Systems Group, TU Wien, Wein, Austria. He has been the Chairman of the Informatics Section of the Academia Europaea, since December 2016.

Prof. Dustdar has been a member of the IEEE Conference Activities Committee (CAC), since 2016, the Section Committee of Informatics of the Academia Europaea, since 2015, and the Academia Europaea: The Academy of Europe, Informatics Section, since 2013. He was a recipient of the ACM Distinguished Scientist Award in 2009 and the IBM Faculty Award in 2012. He is an Associate Editor of the IEEE TRANSACTIONS ON SERVICES COMPUTING, *ACM Transactions on the Web*, and *ACM Transactions on Internet Technology*. He is on the Editorial Board of IEEE.