PMF: A Privacy-preserving Human Mobility Prediction Framework via Federated Learning

JIE FENG, Beijing National Research Center for Information Science and Technology (BNRist), Department of Electronic Engineering, Tsinghua University, China
CAN RONG, School of Software and Microelectronics, Peking University, China
FUNING SUN, Tencent Inc., China
DIANSHENG GUO, Tencent Inc., China
YONG LI, Beijing National Research Center for Information Science and Technology (BNRist), Department of Electronic Engineering, Tsinghua University, China

With the popularity of mobile devices and location-based social network, understanding and modelling the human mobility becomes an important topic in the field of ubiquitous computing. With the model developing from personal models with own information to the joint models with population information, the prediction performance of proposed models become better and better. Meanwhile, the privacy issues of these models come into the view of community and the public: collecting and uploading private data to the centralized server without enough regulation. In this paper, we propose **PMF**, a **p**rivacy-preserving **m**obility prediction framework via **f**ederated learning, to solve this problem without significantly sacrificing the prediction performance. In our framework, based on the deep learning mobility model, no private data is uploaded into the centralized server and the only uploaded thing is the updated model parameters which are difficult to crack and thus more secure. Furthermore, we design a group optimization method for the training on local devices to achieve better trade-off between performance and privacy. Finally, we propose a fine-tuned personal adaptor for personal modelling to further improve the prediction performance. We conduct extensive experiments on three real-life mobility datasets to demonstrate the superiority and effectiveness of our methods in privacy protection settings.

CCS Concepts: • **Security and privacy → Human and societal aspects of security and privacy**; • **Information systems → Location based services**; **Data mining**; • **Human-centered computing → Ubiquitous and mobile computing design and evaluation methods**.

Additional Key Words and Phrases: Mobility prediction; Privacy-preserving system;

ACM Reference Format:
Jie Feng, Can Rong, Funing Sun, Diansheng Guo, and Yong Li. 2020. PMF: A Privacy-preserving Human Mobility Prediction Framework via Federated Learning. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 1, Article 10 (March 2020), 21 pages. https://doi.org/10.1145/3381006

Authors' addresses: Jie Feng, feng-j16@mails.tsinghua.edu.cn, Beijing National Research Center for Information Science and Technology (BNRist), Department of Electronic Engineering, Tsinghua University, Beijing, 100084, China; Can Rong, School of Software and Microelectronics, Peking University, Beijing, China, rongcan@pku.edu.cn; Funing Sun, funingsun@tencent.com, Tencent Inc. Beijing, 100084, China; Diansheng Guo, dguo@tencent.com, Tencent Inc. Beijing, 100084, China; Yong Li, liyong07@tsinghua.edu.cn, Beijing National Research Center for Information Science and Technology (BNRist), Department of Electronic Engineering, Tsinghua University, Beijing, 100084, China.

2474-9567/2020/3-ART10 $15.00
https://doi.org/10.1145/3381006

1 INTRODUCTION

With the popularity of mobile devices and location-based social networks, human mobility prediction becomes an important and emerging topic in ubiquitous computing. Human mobility prediction is of great value in many applications, ranging from traffic engineering, network optimization to urban planning and ride-sharing. For example, with the help of mobility prediction techniques, the government can better understand the demand of the population in transportation system and other related resources to make the reasonable and timely decisions. For ride-sharing platforms like Uber and Didi, accurate mobility prediction will help them to sense the dynamic travel demand in the city and schedule vehicles to meet the requirement with minimum cost. For the LBSN platforms, appropriate mobility prediction can help them to provide a list of locations for users to make it easily find the expected check-in goal, which helps improve the customer satisfaction.

Due to the great value in practical applications and systems, the research community has witnessed a large number of studies in human mobility prediction. In this paper, we will focus on the problem of next place forecasting for the individual in the human mobility prediction while omitting the works of population-level mobility prediction [27] which have no privacy concerns. In the field of individual mobility prediction, existing works can be classified into two categories: 1) personal model with self-information; 2) joint model with population information. Personal model includes Markov model [16], hidden Markov model [31], decision tree [21] and so on. These works try to model the mobility behavior of individuals with various methods by only utilizing their own mobility data records. While these models perform well in mobility prediction with dense data records, the high requirement on the data quality makes them fail in the real and challenging scenario with sparse and limited data records, such as in LBSN [14, 15, 23, 28, 49].

Recently, to overcome the difficulties introduced by the sparsity of mobility records for individual mobility prediction, the research community starts to explore the potential of jointly training the model with the data from the user group. The basic idea of these works is to train the mobility model with not only the individual's mobility data but also with mobility data from other individuals. They assume that individuals can be divided into different user groups, where individuals in the same user group usually share significant movement regularity. Following this direction, enormous methods are proposed and achieve promising performance on LBSN datasets (*e.g.*, Foursquare [14], Twitter [49]) with various joint training mechanism, including FPMC [37], GMove [49], ST-RNN [28], DeepMove [14], HST-LSTM [23], STGN [50] and so on.

While advanced joint models achieve better prediction performance than the personal models, they are faced with significant privacy issues due to the process of data uploading and centralized model training. With the increasing awareness of public and government on user privacy and data security, this privacy issue of existing methods becomes urgent to be solved. To solve the privacy issue, multiple solutions [1, 19, 40, 41, 45] are proposed by introducing different data protection mechanisms like k-anonymity [19] and differential privacy [12]. However, these data-based solutions are not for the scenario we cared about here and also significantly reduce the follow-up modelling performance.

Recently, federated learning [32] with decentralized training is proposed for privacy-sensitive modelling, which provides us another choice to solve the aforementioned issues. In the framework of federated learning, a shared model is first decentralized trained in mobile devices and then aggregated in the cloud without directly accessing to personal data. In this way, the private data is stored and analyzed in the mobile device, which protects the privacy naturally. Inspired by this, we apply the concept of federated learning into our task to achieve the goal of privacy-preserving mobility prediction.

However, direct application of federated learning is not enough for our privacy-preserving mobility prediction task due to the following two challenges. **First**, the uploading and aggregation process of federated learning are not secure in the mobility prediction task, which can leak private information of individuals. On the one hand, the external hackers can attack the uploading process by analyzing the change of model weights to obtain

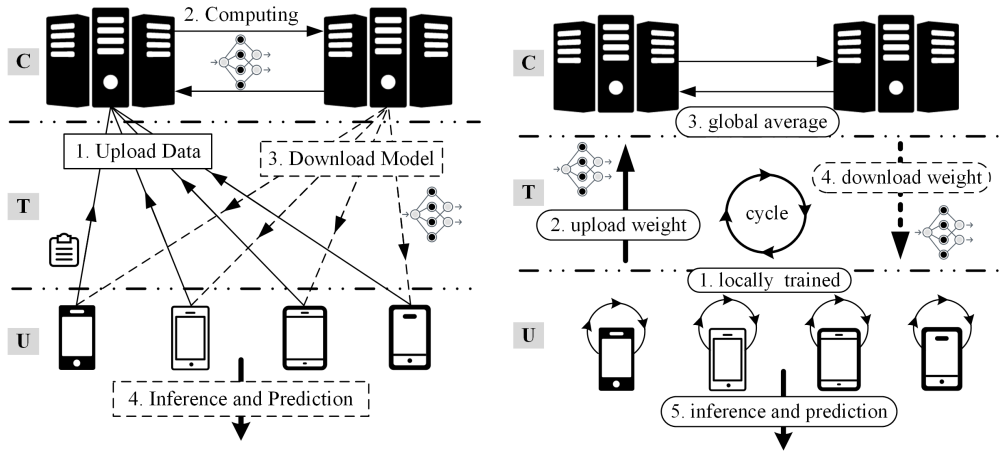
private information of specific individuals. On the other hand, the owner of centralized training sever also has the opportunities to do similar things to obtain personal information. While some works [17, 33] apply differential privacy to enable the secure aggregation, they fail to solve the issue in the uploading process and also cost too much by applying noise in all the model parameters. **Second**, the application of privacy protection will lead to significant performance reduction in the mobility prediction task. As mentioned before, while existing data protection mechanism like l-diversity [30] or differential privacy transfer protocol [39] can protect the personal privacy, they have to destroy the original data and thus greatly harm the final performance.

In this paper, we propose **PMF**, a novel **P**rivacy-preserving **M**obility prediction framework via **F**ederated learning, to protect the user privacy and figure out the aforementioned challenges. ***First***, to avoid private data collection and uploading, we apply the concept of federated learning into the mobility prediction task. In this paradigm, what we upload to the server is not data but only the intermediate result like the gradient of neural network weight, which is more secure and can not be easily decoded by others. ***Second***, we give a practical attack case in the mobility prediction task and design a group optimization algorithm on the mobile devices to avoid this kind of attack with little resource cost and performance sacrifice. In the group optimization procedure, the whole model is divided into the risky group trained with protected data and the secure group trained with normal data. Furthermore, we propose an efficient aggregation strategy for robust convergence and an effective polling schema for fair clients selection in the centralized server. ***Finally***, we strengthen the modelling of the personal pattern on the mobile devices to further improve the prediction performance. We achieve this by adding a personal adaptor as an additional component of the mobility model, which is only appeared in the local device. In this way, the original whole model can still participate in the normal federated learning process and the personal adaptor can be fine-tuned for better performance in the local mobile device.

Our contributions can be summarized as follows:

- To the best of our knowledge, we are the first to propose a privacy-preserving mobility prediction framework for accurate mobility prediction under the constrain of protecting personal privacy. With preserving private data on the user device and conducting joint aggregation in the centralized server via model parameters, our framework succeeds in utilizing the shared knowledge without leaking personal privacy.
- We give a practical attack case in the mobility prediction task and propose a specific group optimization algorithm for the secure and efficient training of mobility model on the local devices. With the help of group optimization strategy, the performance degradation introduced by the privacy protection mechanism can be reduced significantly. We also propose a fine-tuned personal adaptor to further improve the prediction performance.
- Based on a simulated multi-user movement environment, we conducted extensive experiments on two real-world mobility data to demonstrate the effectiveness of the proposed framework for mobility prediction and privacy protection. The proposed framework sheds lights for the privacy-preserving modelling for human mobility and can also be applied in many other user behaviors modelling like recommendation with sensitive item information.

The rest of this paper is organized as follows. We first formulate the problem in Section 2. Then, we introduce details of the proposed privacy-preserving framework and personalized mobility model in Section 3. After the framework description, we apply PMF on three real-world mobility datasets with simulated multi-user environment and conduct extensive analysis on the prediction performance and the effectiveness of the privacy-preserving mechanism in Section 4. After systematically reviewing the related works in Section 5, we conclude our paper in Section 6.



(a) Workflow of the conventional system.

(b) Workflow of the proposed system.

Fig. 1. Workflow comparison between the conventional system and the proposed system. There are three environments in the figure: "C" denotes the cloud servers, "T" denotes transfer environment, and "U" denotes mobile devices.

2 PRELIMINARIES

In this section, we give a brief definition of the mobility prediction problem. With the popularity of mobile devices and the advance of location techniques, location-based service becomes an important component of enormous internet services like LBSN and ride-sharing. Thus, different location data are recorded and processed with different requirements from various services. For the generalizability and simplicity, we give the general definition of human mobility trajectory as follows.

Definition 1.1 (mobility trajectory) A location record q is denoted as a tuple of three basic elements: location identification l, timestamp t, user identity u, i.e., $q =< l, t, u >$. Thus, a mobility trajectory s of user u can be defined as a set of ordered location records $\{q_1, q_2, ..., q_n\}$, i.e., $s_u = \{< l_1, t_1 >, < l_2, t_2 >, ..., < l_n, t_n >\}$.

In the mobility trajectory, the location identification can be longitude and latitude coordinates, spatial grid ID or street block number. Following recent practice [14, 50], we transform various location identification into a new unique ID for simplicity. Further, we quantify the time interval into fixed value for the simplicity of the modelling. We choose 30 minutes as the default time interval by considering the characteristic of human mobility and the general frequency of location recording in various services. It is noted that the spatial and temporal resolution can be easily adapted to the requirement.

Definition 1.2 (mobility prediction problem) Given the past mobility trajectory $s_u = \{q_1, q_2, ..., q_n\}$ of user u, mobility prediction task is to estimate the probability of location l_{n+1} for user u in the next time step t_{n+1}. Furthermore, mobility trajectories from other users can also be utilized in this task as auxiliary information about location and the mobility pattern.

3 SYSTEM DESIGN

In this section, we first overview the whole system and then give a detailed description of core components.

3.1 System Overview

Fig. 1 presents the basic workflow of the conventional system and the proposed system for the mobility prediction task. Here, we only consider the system framework for joint models due to their promising performance. In brief, the personal model can be only trained and launched on mobile device without uploading any data. Besides, the personal model can also be employed as the joint model in Fig. 1(a).

As Fig. 1(a) shows, the standard procedure of conventional system for mobility prediction task can be divided into four steps: 1) collecting data from mobile device and upload them via the transfer environment; 2) training a joint model for the aggregated mobility data; 3) distributing the global optimal model into mobile device via the transfer environment; 4) inferring the future mobility behavior on mobile device. Based on the assumption of the weak computing power and limited storage capacity of local devices, the conventional system takes over all these rights from local devices and work well in the past. However, with the increasing awareness of data security and personal privacy, the conventional system is faced with great privacy challenges. Meanwhile, the past decades witnessed the great progress of mobile devices and communication techniques, which provide us other opportunities to rethink the conventional paradigm. By considering the privacy issue within distributed computing, the concept of federated learning [32] is proposed to utilize the power of the mobile devices to participate more in the whole system for better privacy protection. It aims to train a shared model while leaving the sensitive personal data on each user's mobile device.

Inspired by this, we propose a novel federated learning based privacy-preserving mobility prediction system. As presented in Fig. 1(b), the whole process of proposed system can be divided into five steps: 1) each device trains a mobility model with only self data locally; 2) each selected device uploads the trained mobility model to the cloud server; 3) the cloud server generates a global model by aggregating the received various local models; 4) the cloud server chooses candidate mobile devices to distribute the updated global model to repeat the aforementioned 4 steps until meeting the stop criteria; 5) the mobile device downloads the optimal model to infer and predict the mobility behaviors. In this system, the private data is only stored and accessed on the local device. Thus, the personal right for controlling the data is protected successfully from the source. Furthermore, we develop a specific group optimization algorithm for the secure transfer and global training process to prevent external attackers and server owner to extract personal information from the uploaded local models. Besides, we also propose a global optimization mechanism to complete the joint training task on the centralized server and efficient polling schema for fair and flexible task assignments. Finally, we also propose a personal adaptor as the additional component of the mobility model to enable the local fine-tune in the mobile devices for better prediction performance. Details about the optimization and model design are introduced in the following sections.

3.2 Mobility Prediction Model

The framework of our basic mobility prediction model is presented in Fig. 2. As Fig. 2 shows, the mobility prediction model is consisting of three parts: the input module with multi-modal embedding, the sequential module with LSTM, and output module. Details of these modules are introduced in the following sections.

3.2.1 Input Module with Multi-modal Embedding. Following the definition in Section 2, the original mobility trajectory is converted into a sequence S_u. It is noted that we replace the lowercase notation defined in Section 2 with the capital in Fig. 2 for better visual effects. As Fig. 2 shows, the basic elements of trajectories including location and time are first converted into one-hot vectors. Another choice of location is to directly use the longitude and latitude of the location as the input of neural network. We also try this and find it does not have a significant positive influence on the final results. This operation is similar to the practice in recent works [14, 42, 43].

Due to there are more than thousands of locations in the dataset, the dimension of the one-hot location can be up to thousands. Thus, based on the one-hot input vector of location and time, we design two specific embedding

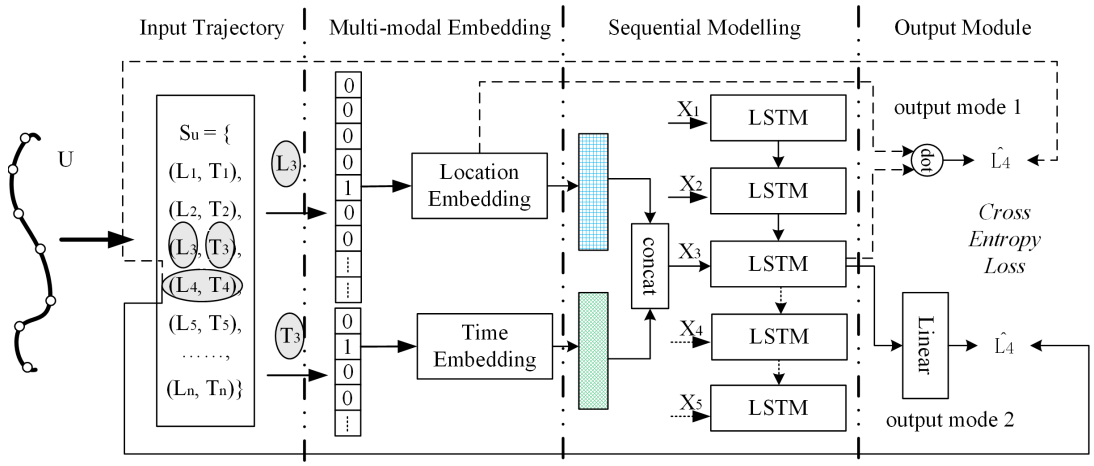


Fig. 2. The framework of the basic mobility prediction model, which includes three parts: multi-modal embedding, sequential modelling, and output module. Here, X_3 is the concatenate vector of location embedding vector $Emb(L_3)$ and time embedding vector $Emb(T_3)$.

tables to reduce the feature dimension and learn the dense representation of discrete location and time. Similar to embedding in word2vec [34], embedding table is just a lookup table which stores the dense representation of different index. The dimension of these dense representations can be lower to one hundred, which is much lower than the original thousands of dimension. For better performance and meaningful semantics, the embedding module is trainable and optimized with the whole network during the training. In this way, the high-dimensional location and time input are converted into a low-dimensional dense representation with meaningful semantics. Finally, we concatenate the location vector and time vector to obtain the dense representation X of a spatial-temporal point. The formula is as follows,

$$X_i = tanh([W_t t_i + b_t; W_l l_i + b_l]), \tag{1}$$

where W and b denote the learnable parameters of embedding layers, $tanh$ denotes the non-linear activation function, $[;;]$ denotes the concatenate function. The fusion operation can also be replaced with the element-wise summation, multiply or other advanced fusion techniques [36]. In the experiment, we find the concatenation operation is parameter-efficient and works well in most cases. For the simplicity of the model, we only use concatenation in the experiment. It is noted that other advanced fusion methods can also be applied into our model to further improve the performance, we do not consider them in our paper and leave them for future work.

3.2.2 Sequential Modelling Unit. With the recent success of recurrent network in sequential modelling, we employ the advanced recurrent neural network to model the sequential transition relation in the mobility trajectory. Recurrent neural network is a class of neural network which combine the last output of the network and the current input to make network learn to capture the relationship between the sequential inputs. Long short-term memory (LSTM) [20], as the most successful variant of recurrent neural network, has been widely used in different sequential modelling task including mobility modelling task. LSTM works by utilizing the gating

function to control what to remember and what to forget. The formulation of LSTM is as follows,

$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + b_i), f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + b_f), \tag{2}$$

$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1} + b_o), g_t = tanh(W_{gx}x_t + W_{gh}h_{t-1} + b_g), \tag{3}$$

$$c_t = f_t \circ c_{t-1} + i_t \circ g_t, h_t = o_t \circ tanh(c_t) \tag{4}$$

where '$\circ$' denotes Hadamard product, x_t, h_t, c_t denote the input, hidden state and cell state, i_t, f_t, o_t denote three types gates, and g_t denotes the useful information from the input. We also consider other variants of LSTM like GRU [9] in our model, which is a simplified version of LSTM and works well in some cases. It is noted that other variants of recurrent neural network can also be directly applied in our sequential unit by replacing LSTM unit without any modification. In our paper, without specific demonstration, we use LSTM as the default sequential unit.

3.2.3 Output Module. Following the sequential module, we design two types of output modules to translate the hidden state into final location prediction result. The first kind of output module is a projection-based method. As the solid line in the bottom right of Fig. 2 shows, we utilize a linear layer to directly project the hidden state into a high-dimensional location vector. Then, we apply the soft-max function on the output of projection to obtain the probability distribution of the predicted location. The formulation of this output module is as follows,

$$P(y_i|h) = \frac{\exp(z_i)}{\sum_1^n \exp(z_i)}, \mathbf{z} = Wh + b, \tag{5}$$

where h is the hidden state from the former sequential unit, W and b denote the learnable parameter of projection linear layer, n is the dimension of candidate locations, $p(y_i|h)$ is the prediction probability of location y_i when h is given.

The second kind of output module is a correlation-based method. As the dash line in the top right of Fig. 2 shows, we do not use any projection layer but directly use the hidden state vector to calculate the correlation between the hidden state and the dense location embedding representation. Based on the correlation results, we use the similar soft-max function to obtain the final prediction probability distribution. Compared with the projection-based method, correlation-based method is parameter-efficient and secure. In our experiment, we also observe that correlation-based output mode is more robust to the dataset characteristics and always performs better. Thus, we choose the correlation-based output mode as the default output mode in our system.

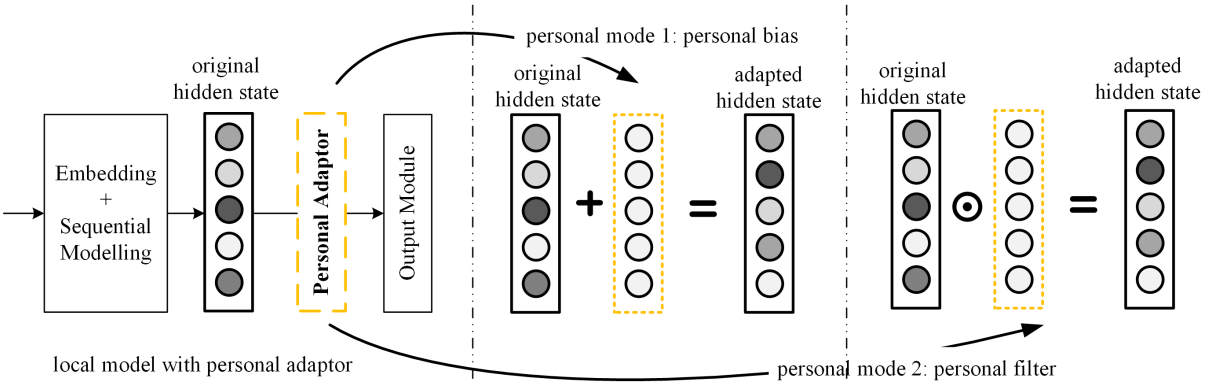


Fig. 3. Two types of personal adaptors on the local device. This personal adaptor only exists on the local device and waits to be fine-tuned in the local device for final prediction.

3.2.4 Personal Adaptor. To further improve the prediction performance while leaving the whole model in the normal federated learning procedure, we propose a personal adaptor to be fine-tuned in the local device with better personal pattern modelling. As presented in Fig. 3, we aim to revise the intermediate hidden state before the output module by fine-tuning a personal adaptor. To achieve this goal, we design two types of personal adaptors. The first design is the personal adaptor with bias: the original hidden state is adapted by a trainable bias vector, which has the same size as the hidden state. Based on the add operation, additional bias vector slightly changes the distribution in the hidden state by a learnable personal bias. The second design is the personal adaptor with filter: the hidden state is filtered by a trainable same-size vector with sigmoid function by the multiply operation. In this way, the hidden state vector can be fine-tuned by the personal preference existed in the personal data. The formulas of two types of personal adaptors are as follows,

$$H_{ap} = H_{og} + b_p, H_{ap} = H_{og} * \sigma(v_p), \tag{6}$$

where H_{ap} denotes the adapted hidden state, H_{og} denotes the original hidden state, b_p is the personal bias, and v_p is the personal vector for filtering. The personal adaptor only exists in the local devices and is fine-tuned with the whole frozen model after the normal federated learning process. It is noted that we do not design complicated structure for the personal adaptor to avoid the potential overfitting on limited personal data.

3.3 Privacy-preserving Optimization Mechanism

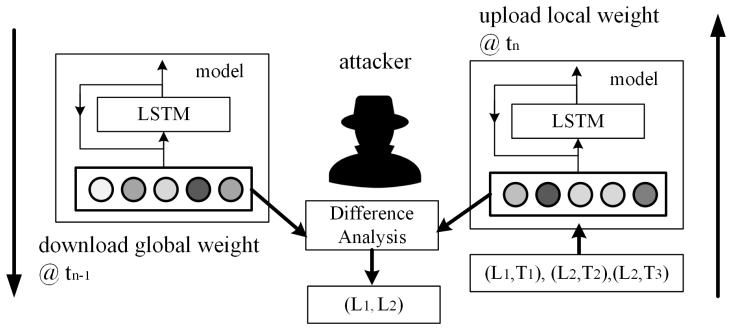


Fig. 4. Attacker gets the private information by analyzing the variation of the uploaded model.

3.3.1 A Simple Attack Case. Based on the 5 steps in Fig. 1(b), we avoid to directly upload any private data into the cloud server. In this way, private information seems to be protected naturally. Without uploading any private data to cloud server, we have to upload trained model weights (e.g., gradient) to participate in the training procedure and benefit from the joint training. Due to the complexity and implicit nature of model weights, it is much harder to extract private information from the model weights than from the data. But, with some prior knowledge, attacking the uploading procedure and extracting private information from the uploaded model weights becomes easy. Fig. 4 presents a simple attack case, where external attacker or server owner can infer the visited location set of specific users by analyzing the difference between the last round global model and the current round uploaded local model. For example, user Alice first accepts a global model $M_g^{t_{n-1}}$ at t_{n-1} from the cloud server. Then, with this global model as the start point, Alice obtains a local model $M_l^{t_n}$ at t_n after training on her private mobility trajectory. Due to the regularity of human mobility [10, 18], the visited locations of one user during his or her daily life are limited to a small sub-set of the whole location set in the city. Furthermore, based on the location embedding design in Section 3.2.1, the limited visited locations in the whole location set mean

that only embeddings of these limited locations will be updated during the training of this local optimization step. Thus, by comparing the difference of location embedding table between the global model $M_g^{t_{n-1}}$ and the local model $M_l^{t_n}$, attackers can easily infer which location has been visited by the target user. Apart from the embedding table, another important part of the mobility prediction model is the sequential unit. Due to the black-box characteristics of the neural network, it is very difficult to extract useful information from the weight itself or the change of the weight of the recurrent neural network. Thus, we do not consider the specific attack for recurrent neural network in this paper.

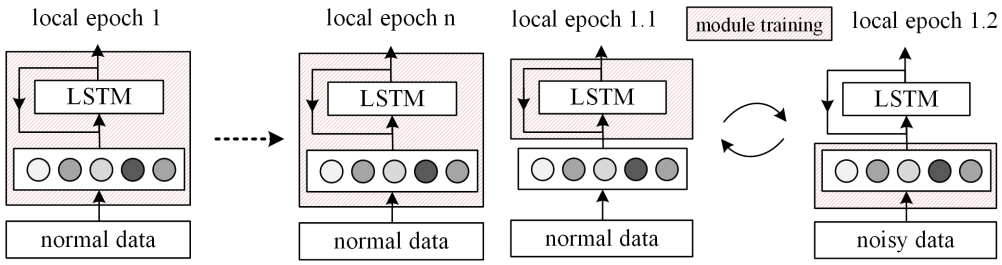
3.3.2 Group Optimization on the Local Devices with Differential Privacy. We propose a privacy-preserving local optimization method to prevent the aforementioned attack risk. We introduce differential privacy into the optimization of the local models to obtain the controlled and privacy-preserving embedding table for secure model sharing. Fig. 5 presents the comparison between the conventional local optimization strategy and the privacy-preserving local optimization strategy. Compared with the general optimization process with normal data, our privacy-preserving local optimization is to train different parts of the whole model with different data. Particularly, based on the flexible combination nature of different components in the neural network, we divide the sub-modules in the neural network into different groups by their privacy leakage risk level. The module group with privacy risk is to be trained with the protected data (*e.g.*, noisy data protected with the differential privacy mechanism) and the module without privacy issue is to be trained with the normal data. Two training procedures are iterated again and again until the stop criteria is met. Fig. 5(b) illustrates the proposed iterated local optimization process. In the local optimization of the mobility model, we first fix the location embedding module and train the left modules with the normal data. Then, we start train the location embedding module with noisy data and fix the left modules simultaneously. Finally, we repeat these two steps again and again. It is noted that the proposed group optimization method is not limited to the basic mobility prediction model introduced before. Once any advanced module is added into the mobility prediction model, we can evaluate the privacy risk of it and select the appropriate training group for it to make the whole new model be trained with enough security guarantee.

The noisy data satisfied the differential privacy requirement is generated based on the planar Laplace mechanism introduced in [4]. We map each true location l to a randomly drawn location point p in the infinite continuous space $\mathcal{P}$ according to the probability density function, which is formulated as follows,

$$D_\epsilon(l,p) = \frac{\epsilon^2}{2\pi} e^{-\epsilon d_S(l,p)}, d_S(l,p) = d_{euclidean}(l,p), \tag{7}$$

where l is the original real location, p is the noisy location, ϵ denotes the privacy parameter of differential privacy. Based on this formulation, we can generate obfuscated location data (noisy data) from the normal data to train the risky sub-module to make it robust to the potential attack.

3.3.3 Global Optimization on the Centralized Server. After the mobile devices uploading the trained local model weights, the centralized server needs to aggregate them to obtain an optimal global model, which is shown as the 3rd step in Fig. 1(b). As the most successful optimization method for deep learning, stochastic gradient descent (SGD) algorithm and its variants are widely used in the training of deep learning models. Thus, we choose SGD algorithm as the default optimization method of deep learning based mobility model in our problem. Based on SGD, a simple global optimization method FedAvg [32] is utilized for federated learning settings. In this paper, we adopt these methods in our settings to aggregate local models into the global updated model. Assumed that we have K mobile devices participate in the local training, the centralized server will obtain K updated local models $\{m_l^k, k = 1, 2, ..., K\}$. Based on these local models, the global model M_g is updated by $M_g^t \leftarrow M_g^{t-1} - \alpha \sum_{k=1}^{K} \beta \nabla m_l^k$, where α denotes the learning rate, β denotes the scale factor. In other words, the updated global model is easily obtained as a weighted average of current local models, $M_g^t \leftarrow \sum_{k=1}^{K} \beta m_l^k$. In our implementation, we replace the



(a) Conventional local optimization. (b) Iteratively group optimization.

Fig. 5. Different local optimization strategies.

Algorithm 1: Training procedure of the proposed system

Global parameters: M^t is the global model at t step; K the number of selected clients; α is the learning rate.

Local parameters: m^k is the local model of client k; p^k is the personal adaptor for client k; *epoch* is the number of local epoch; $\mathcal{D}$ is the private data on the local device; ϵ is the parameter of differential privacy.

Server:

initialize M^0

for *round* $t \in \{1, 2, ...\}$ **do**

- construct/update client candidates pool
- select clients set from candidates pool by polling scheme
- **for** *client* $k \in \{1, 2, ..., K\}$ ***in parallel*** **do**
  - $(m^k, \beta^k) \leftarrow$ **Client**$(M^t, \text{epoch}, \alpha)$
- normalize $\beta^k \in \{1, 2, ..., K\}$
- $M^t \leftarrow \sum_{k=1}^{K} \beta^k m^k$

Client:

// construct noisy data with differential privacy ϵ

$\mathcal{D}_\epsilon \xleftarrow{+noise} \mathcal{D}$

for $i \in \{1, 2, ..., epoch\}$ **do**

- // train the risky module group r of model m with protected data
- $m_r \leftarrow m_r - \alpha \nabla l(m; \mathcal{D}_\epsilon)$
- // train the normal module group n of model m with normal data
- $m_n \leftarrow m_n - \alpha \nabla l(m; \mathcal{D})$

return (m, l)

Fine-tune personal adaptor p on clients before inference.

sample size with the normalized loss of each local model during the training as the scale factor β in the weighted average for better performance. With only knowing the updated local model protected by the former group optimization, the owner of the centralized server is also not able to extract any private information from the local models. These local models are only useful for the optimization of current global model, which means that they can be safely deleted and only leak minimal personal information.

Table 1. Basic statistics of three mobility datasets.

Dataset	City	Duration	Users	Locations	Records	Loc./user	Rec./user
Foursquare	Tokyo	10 months	2293	67124	537703	29	234
Twitter	Los Angeles	4 months	24161	527977	826212	22	34
DenseGPS	Beijing	1 month	5000	31522	15007511	48	3001

Another crucial task for the centralized server is to select a fraction of mobile devices to do the next round optimization. The most simple way is to randomly select mobile devices just like in SGD algorithms. For better fairness and more flexible arrangement, we maintain a candidate pool in each round based on the current status of mobile devices. Then we assign local optimization task by polling scheme. In this way, with each available mobile device is assigned with appropriate task, the performance and fairness of the whole system is guaranteed. With the combination of group optimization in the local devices and global optimization on the centralized server, mobile devices can benefit from the joint training without the risk of privacy issues (compared with the original joint model).

3.4 Training Procedure

Based on the description of the mobility prediction model and optimization mechanism, we summarize the whole training procedure of our system in Algorithm 1. For the local training in the mobile devices, we choose cross-entropy loss as the loss function and select basic SGD algorithm as the default optimizer.

4 EXPERIMENTS

4.1 Datasets

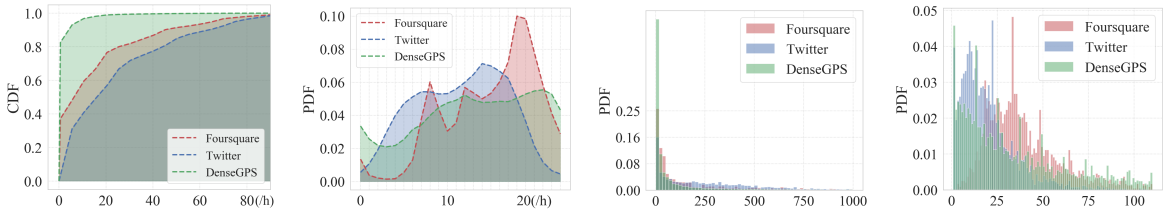
Three representative mobility datasets are used to evaluate the system performance in our paper. The basic information of these datasets is shown in Table 1.

Foursquare [46]: It collects 0.5 million check-ins in Tokyo lasting for about 10 months (from 12 April 2012 to 16 February 2013). Each check-in includes an anonymized user ID, timestamp and location information, e.g., GPS coordinates and semantic meaning (represented by fine-grained venue-categories).

Twitter [48]: It contains around 1.1 million geo-tagged tweets from Los Angeles. These tweets are collected from 1 August 2014 to 30 November 2014. Every geo-tagged tweet consists of four parts, e.g., an anonymized user ID, location information (GPS coordinates), timestamp and the message published by the user. Compared with the other two platforms, Twitter data is very sparse when location service is not frequently-used function for Twitter users.

DenseGPS [14]: This private data is from the location service of one of the major mobile application providers in China, which contains 5000 users with one-month dense location records. Each record contains user id, the GPS coordinates, and the timestamp. Due to the frequent operation in the mobile application, this data is much denser than the former two datasets, where average number of records for each user is at least 10 times than other two datasets.

To better understand the characteristics of the data, we analyze the detailed information of data and draw their distribution in Fig. 6. From Fig. 6(a), we can see that time interval of records on DenseGPS data is much shorter than Foursquare and Twitter data, where more than 80% of the time intervals are shorter than 1 hour. Due to the different cultures of cities and different functions of service, Fig. 6(b) shows the different timestamp distribution of data records from two datasets in one day. In Fig. 6(c) from the spatial view, we find that the visiting frequency of locations in three datasets follow the similar long-tail distribution while this phenomenon



(a) Distribution of time interval of nearby location records. (b) Time distribution of location records. (c) Visiting frequency distribution of locations. (d) Distribution of the number of the visited locations of each user.

Fig. 6. Detailed information of three mobility datasets.

is heavier in DenseGPS dataset. We also compare the distribution of the number of locations that user visited from different datasets in Fig. 6(d).

Following [14, 49], we perform several pre-processing steps on original mobility data to filter the harmful effects of noising data and too much missing data. First, we filter out the users whose number of records is less than 10 to eliminate the effects of data sparsity. Second, we split the records of respective users into several sessions based on the time interval (larger than 72 hours for Foursquare and Twitter data, 10 hours for DenseGPS data) between two neighbor records. Then, all the users with too few sessions (less than 5 sessions) and all the session not long enough are removed (less than 5 records). Besides, the locations on three datasets are projected into $500m \times 500m$ grids. The spatial resolution and the shape of the region can be easily adapted based on the requirement. After the spatial projection, the number of locations in the Foursquare, Twitter, and DenseGPS becomes 3021, 5736 and 11836.

4.2 Baselines

We compare our methods with two kinds of baselines: *privacy-preserving personal models* and *joint models without protecting privacy*.

Personal Model: Personal model is directly executed in the mobile device with only local private data. Without sharing private data, personal model protects the personal privacy but fails to provide competitive performance.

- **P-Markov [16]**: Markov model is widely used to predict human mobility [24, 44] for a long time. For personal mobility modelling, it regards all the visited locations of user as states and builds a transition matrix to capture the first order transition probabilities between them.
- **HMM (Hidden Markov Model) [31]**: Similar to the personalized Markov model, each user mobility is modeled by a private state transition probability matrix and a private state-observation probability matrix.
- **LSTM**: This is a private version of our mobility prediction model in the paper. Each user trains and evaluates a private LSTM model with only his data.

Joint Model: By sharing data from different users, joint model succeeds in capturing the universal mobility patterns and achieves better modelling performance. However, none of joint models have considered the privacy issue.

- **J-Markov**: Different from the personalized Markov chain model (**P-Markov**), we utilize the records of all users to generate the location transition probability matrix. And this shared matrix is used to perform all users' mobility prediction.
- **FPMC [37]**: Factorizing personalized Markov chains (FPMC) is the combination of matrix factorization and Markov chains. Further, sequential bayesian personalized ranking is used in the training process.

- **LSTM**: This is the joint version of our mobility prediction model in the paper. All the user data is utilized to train a joint model and the evaluation is done on each user.
- **DeepMove [14]**: This is the state-of-the-art method for mobility prediction task, which combines neural attention with the recurrent network to capture the periodical pattern in human mobility.

4.3 Metrics

We set top-k accuracy as our evaluation metric. To calculate the top-k prediction accuracy, we first use the model to generate k most likely predictions based on the given historical records and then check whether the real next location is included in these top-k predictions.

4.4 Parameters

The main parameters of our privacy-preserving system can be divided into three groups, the default settings of them are as follows: 1) parameters for the mobility prediction model: the size of location embedding=64, the size of time embedding=10 and the size of hidden state=64; 2) hyper-parameters for the local training of mobility model: learning rate=0.02, dropout rate=0.5, weight decay=1e-6; 3) hyper-parameters for the global aggregation: clients=400, local epoch=1. It is noted that parameters for specific experiments are assigned by parameter search. The parameters of LSTM (personal and joint) are set as the aforementioned settings. In HMM model, we set the number of hidden states to be half of the number of locations for the corresponding user based on the parameter study on the prediction performance. According to [37], the performance of the FPMC model will continue to improve as the number of latent factors increases. Thus, we set the number of latent factors to the number of locations that appear in the training set.

4.5 Overall Performance

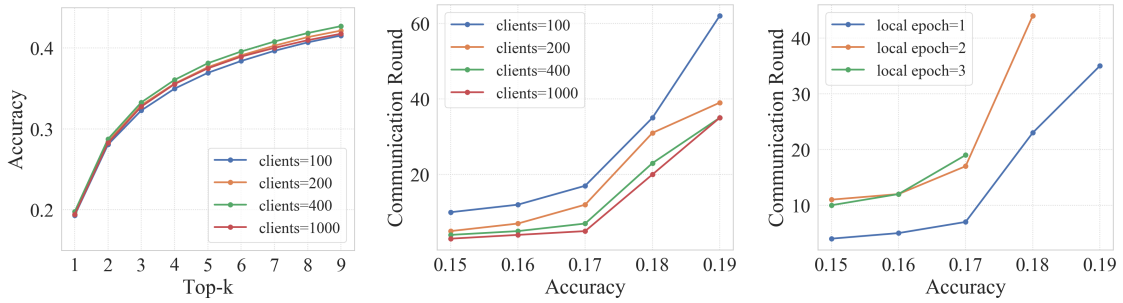
In this section, we first present the comparison results with baseline models under the constrain of privacy-preserving settings and then analyze the effects of key parameters in the training procedure of the proposed system.

4.5.1 Performance Comparison. We compare our proposed methods with two kinds of baselines in Table 2 on three datasets. From Table 2, we can draw three key conclusions. First, compared with personal models, joint models indeed achieve better performance on mobility modelling. The best performance achieved by the joint models on Foursquare dataset is up to 0.217 which improves the best performance of personal model by 32%. The performance gain achieved by the joint models on the Twitter dataset is also similar. Second, as the deep learning based model, DeepMove and LSTM with the joint training settings achieve the best performance on three datasets. This result demonstrates the superiority of deep models on mobility modeling. Furthermore, we find that the performance of LSTM with personal settings is much worse than the joint setting and other personal models, which shows the importance of joint training and knowledge sharing for deep models. Besides, we also observe that DeepMove only achieves similar results to joint LSTM on Twitter data and DenseGPS data. Third, constrained with the privacy-preserving settings (*e.g.*, preserve data on local devices), our proposed models especially the model with fine-tuned personal adaptor still achieve competitive performance compared with the best joint model on three datasets. Compared with the best privacy-preserving personal model, our proposed models improve the modelling performance by 15%∼28% on three datasets. In summary, by preserving private data on local devices and effective optimization mechanisms, our proposed methods achieve promising performance when protecting personal privacy.

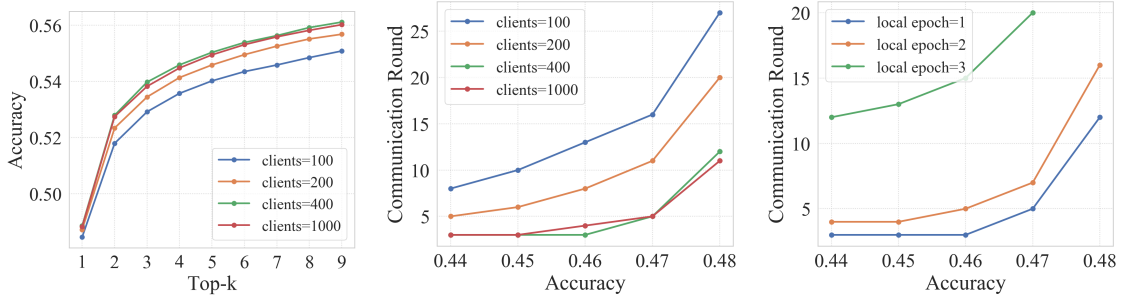
4.5.2 The Effects of the Selected Clients. In the experiment, the number of total clients is equal to the number of users in the dataset, which is more than 2000. In each training round of the proposed system, we only choose a

Table 2. Overall top-1 prediction performance on three mobility datasets. "+personal" denotes the model with fine-tuned personal adaptor on local devices.

Privacy-Level	Methods	Top-1@Foursquare	Improv.	Top-1@Twitter	Improv.	Top-1@DenseGPS	Improv.
Personal Model (privacy-preserving)	P-Markov	0.151±0.000	-7.93%	0.345±0.000	-10.62%	0.618±0.000	-0.80%
	HMM	0.164±0.023	0	0.386±0.003	0	0.623±0.005	0
	LSTM	0.135±0.002	-17.68%	0.371±0.005	-3.89%	0.617±0.002	-0.96%
Joint Model (privacy-leakage)	J-Markov	0.186±0.000	+13.41%	0.441±0.000	+14.25%	0.694±0.000	+11.40%
	FPMC	0.195±0.001	+18.90%	0.460±0.003	+19.17%	0.570±0.000	-8.51%
	LSTM	0.217±0.002	+32.32%	0.497±0.001	+28.76%	0.718±0.001	+15.25%
	DeepMove	0.232±0.000	+41.46%	0.498±0.001	+29.01%	0.714±0.001	+14.07%
Our Model (privacy-preserving)	Our Basic Model	0.209±0.001	+27.44%	0.489±0.000	+26.68%	0.694±0.000	+11.40%
	+Personal Adaptor	0.213±0.001	+29.88%	0.495±0.001	+28.24%	0.715±0.000	+14.77%



(a) Top-k accuracy with selected clients on Foursquare data (b) Communication rounds with selected clients on Foursquare data (c) Communication rounds with local epoch on Foursquare data



(d) Top-k accuracy with selected clients on Twitter data (e) Communication rounds with selected clients on Twitter data (f) Communication rounds with local epoch on Twitter data

Fig. 7. The effects of the number of selected clients and the number of local training epoch.

fraction of clients to participate in the current optimization procedure. Fig. 7(a) and Fig. 7(d) present the effects of the number of selected clients on the final performance on two datasets. In general, we can observe that the final performance is improved continuously with the number of clients increasing from 100 to 400. However, we also observe that when the number of clients becomes 1000, the performance stops to be improved and

Table 3. Parameter number and communication cost of different methods on Foursquare data. The value in the table is the typical value. "SD" denotes the data volume of one user which is smaller than the weights of the trained model, "N" denotes the average iteration rounds during the federated training process of each mobile device, the typical value of "N" in our experiment is smaller than 10.

@Foursquare	P-Markov	HMM	J-Markov	FPMC	LSTM	DeepMove	Ours
Model Size	∼10KB	∼2KB	∼50MB	∼100MB	∼10MB	∼10MB+	∼10MB
Communication	0	0	∼10KB+SD	∼100MB+SD	∼10MB+SD	∼10MB+SD	∼10MBxN

even turns to be degraded. This phenomenon is similar to the idea of mini-batch SGD: data with proper 'batch' size in each optimization step is better than only one data or just all the data. Besides, as Fig. 7(b) and Fig. 7(e) shows, more participants will also lead to higher convergence rate. For example, when 400 clients participate in the optimization procedure, the system only needs 34 communication rounds to achieve more than 0.19 top-1 accuracy. As for 100 clients, they need 60 communication rounds to achieve the similar accuracy. It is noted that all the clients are running in the parallel way, which means that the time cost of one communication round will not be increased significantly with more clients participating in the system. In other words, more communication rounds require more training time and face more random risk. Based on the two aforementioned results, by considering the final performance and convergence rate, choosing proper clients to participate in the optimization becomes important for the system. We also study the effects of local training epoch on the system performance in Fig. 7(c) and Fig. 7(f). We find that only one local training epoch is enough in our experiments, which may be due to the small size of the dataset and neural network. Detailed analysis of this can be further conducted in the future.

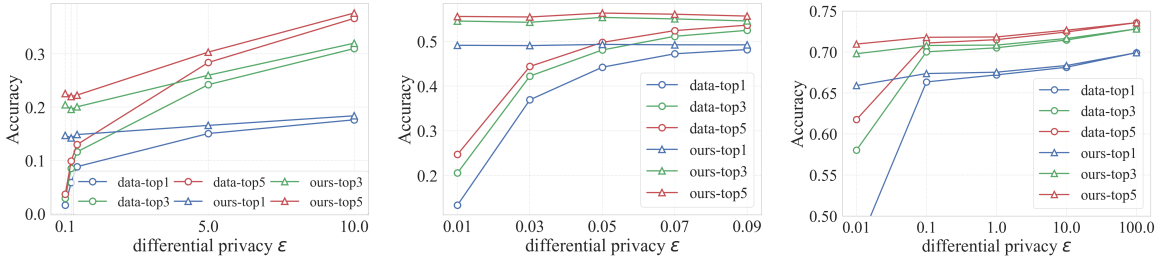
4.5.3 Study on Computing Complexity and Communication Cost. In Table 3, we choose the experiment results on Foursquare data to compare different methods from the view of model size and communication cost. Compared with the personal model with fewer parameters and low communication cost, joint models have more parameters and also need specific communication cost to update data and download model. Compared with the conventional joint model, the proposed model protects personal privacy by preserving the data on local devices and only transferring the model weights. The model size of our whole model is similar to LSTM and smaller than DeepMove with the historical attention module. In general, the communication cost of our methods is N times of the model size, where N is the average iteration rounds during the federated training process of each mobile device. Based on the practical experiments on Foursquare data, the value of N is around 10.

4.6 Privacy Risk Analysis

After evaluating the superior performance of the proposed system in Section 4.5, we analyze the privacy risk of it and demonstrate the effectiveness of group optimization on local devices in this section. Here, we compare our method with the data protection method, which is written as "data" in the following figures. The data protection method works by directly training models with data protected by the differential privacy mechanism in paper [4].

4.6.1 The Effects of Differential Privacy on Prediction Accuracy. Fig. 8 presents the effects of differential privacy parameter ϵ on the final prediction performance on three datasets. We first analyze the results on Foursquare data. From three lines with "o" marker in Fig. 8(a), we can see that the performance of the conventional method "data" decreases significantly when differential privacy parameter ϵ becomes smaller. However, as lines with "Δ" shows, the performance of our method decreases slowly when the differential privacy requirement becomes strict by using smaller ϵ. For example, when ϵ reduces from 20 to 0.05, the top-1 accuracy of "data" method falls from more than 0.20 to less than 0.09 on Foursquare data while the top-1 accuracy of our method only falls from

more than 0.20 to about 0.15. We can observe the similar results from the other two datasets in Fig. 8. The results demonstrate the robustness of our method on strict differential privacy settings. Besides, it is noted that the performance of our method on Twitter data keeps almost unchanged during the change of differential privacy parameter ϵ. This is due to the sparse and duplicated characteristics of Twitter data presented in Fig. 6. Learned on these sparse and noisy data, the model cannot understand the importance of spatial adjacency but focus on the transition relations between a small fraction of isolated locations, which can be directly captured by the recurrent network. In other words, a good recurrent neural network is enough for the accurate mobility modelling and "bad" or "noisy" location embedding table will only have limited influence on the final prediction performance.



(a) Results on Foursquare dataset. (b) Results on Twitter dataset. (c) Results on DenseGPS dataset.

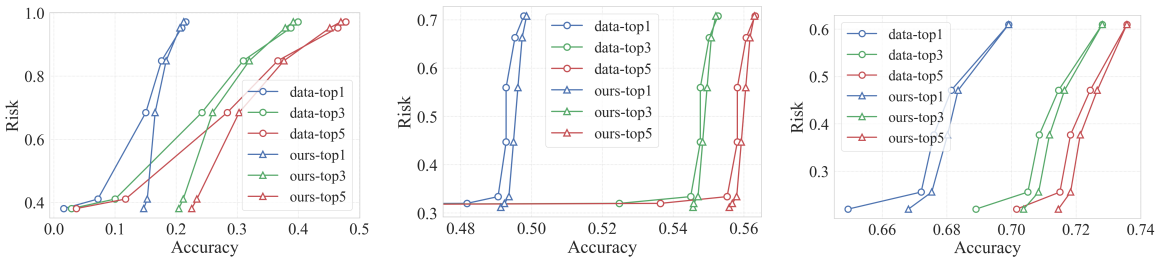
Fig. 8. The effectiveness of our method with changing differential privacy parameter ϵ.

Table 4. The variation of attack risk with different differential privacy parameter ϵ.

Differential privacy ϵ		0.1	0.5	1	5	10	20	50	100	∞
Attack Risk	Foursquare	0.381	0.395	0.411	0.685	0.849	0.953	0.972	0.972	0.987
	Twitter	0.320	0.324	0.334	0.447	0.560	0.663	0.708	0.708	0.779
	DenseGPS	0.222	0.234	0.256	0.377	0.471	0.549	0.583	0.584	0.606

4.6.2 The Trade-off Between Attack Risk and Prediction Accuracy. Here, we evaluate the effectiveness and security of our method on the attack introduced in Section 3.3.1. Based on the attack methods introduced before, the attack risk is defined as follows, $risk = \frac{||ls_{attack} \cap ls_{truth}||}{||ls_{truth}||}$, where ls is the abbreviation of location set, $||ls_{attack}||$ denotes the estimated location set based on the difference analysis of the downloaded model and uploaded model. It is noted that this definition is just a simple proxy of the real attack risk, which can be better measured by other metrics. Table 4 shows the attack risk of the model with different differential privacy parameter ϵ. We can observe that the attack risk can be reduced significantly by utilizing differential privacy mechanisms.

Furthermore, by combining the information from Table 4 and Fig. 8, we obtain Fig. 9 which reflects the relationship between the attack risk and performance of our method. We also first analyze the results on Foursquare data. As Fig. 9(a) shows, to reduce the attack risk from 1 to 0.4, the performance of "data" method is also reduced significantly from more than 0.20 to about 0.08. With the support of the proposed efficient optimization mechanism, the performance degradation of our method for equal attack risk reduction is smaller than 0.05, which is only 40% of the conventional "data" method. Fig. 9 shows the similar conclusion on other two datasets. In summary, the effectiveness and security of our method on protecting personal privacy from the



(a) Results on Foursquare dataset. (b) Results on Twitter dataset. (c) Results on DenseGPS dataset.

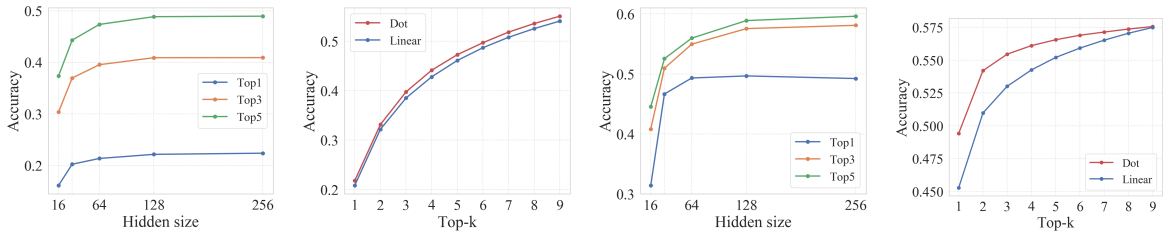
Fig. 9. The attack risk during the uploading procedure on three datasets.

difference attack is confirmed in Fig. 8 and Fig. 4. Our method can achieve better performance with a lower risk of difference attack. It is noted that the trade-off between risk and performance on Twitter data of our model is not so remarkable, which is similar to the results introduced in the previous subsection. As mentioned before, we find that the role of location embedding in the whole model on Twitter data is not so important due to the sparse and duplicated characteristics of it. In other words, the location embedding trained on Twitter data can is not so useful for the whole model and the powerful recurrent network can capture most of limited mobility patterns in data itself. Thus, when we change the differential privacy parameter to reduce the attack risk on Twitter data, the well-trained model with noisy-trained location embedding can still work well.

4.7 Effects of Model Size on the Performance

To efficiently utilize the limited storage space and weak computing power on the local mobile devices, we expect to design parameter efficient model in the ubiquitous computing system. However, smaller model always leads to the limited performance on the target task. Thus, how to balance the limited parameter requirement and better performance becomes an important topic in ubiquitous computing. In this section, we discuss the effects of model size on the mobility prediction performance to give a proper solution to our problem. For flexible modelling and simplified architecture, we set the dimension H_r of hidden state in recurrent neural network as the same value with H_l. Due to the huge dimension L of available locations, the parameters of whole model is mainly depend on the parameters in location embedding table LxH_l.

By rewriting H_l and H_r as the same H, Fig. 10(a) and Fig. 10(c) show us the accuracy variations with different H on two datasets. We find the small value of H will limit the performance of whole model. Thus, we choose 64 as the default value of H on Foursquare data and choose 128 as the default value of H for Twitter data. We also test two different output modules introduced in Section 3.2.3 in Fig. 10(b) and Fig. 10(d). In these two figures, projection-based output module is denoted as “Linear” and correlation-based output module denoted as “Dot”. We observe that the “Dot” schema is not only parameter-efficient than the “Linear” method but also leads to better performance. The failure of the “Linear” schema on the performance is due to the fact that more parameters will require more training data and more ideal optimizer. These two criteria are naturally not met in our problem settings. In summary, we recommend using the “Dot” schema as the default output mode for mobility prediction task in the privacy-preserving settings.



(a) The effects of hidden size on Foursquare data (b) The effects of output mode on Foursquare data (c) The effects of hidden size on Twitter data (d) The effects of output mode on Twitter data

Fig. 10. The effects of model size on the performance on two datasets.

5 RELATED WORK

5.1 Mobility Prediction Model

Mobility prediction is an important topic in the field of ubiquitous computing. Existing works on mobility prediction can be classified into two categories: personal models and joint models.

Personal Model: Markov model and its variations are common models of this approach. In Markov-based models [7, 16], they model the probability of the future action by building a transition matrix between visited locations based on the past trajectories. To capture the unobserved characteristics between location transition, Mathew et al. [31] cluster the locations from the trajectories and train a personal Hidden Markov Model for each user. Besides, Kalman filter model [13], decision tree [21] are also utilized to model the personal mobility behaviors.

Joint Model: With the limited information, personal models fail to solve the cold start problem and the data sparsity issue. Thus, more and more researchers try to utilize the mobility pattern and popular route from group users to enhance the mobility modelling on individuals. Matrix factorization [8, 37] is introduced to model the mobility. Rendle et al. [37] propose factorized personalized Markov model (FPMC) to combine the advantages of MF and Markov model for next basket prediction. By iteratively clustering the user and modelling the mobility of group users, Zhang et al. [49] propose GMove to utilize the shared movement regularity to improve the modelling performance. In recent years, deep learning based methods [11, 14, 23, 26, 28, 50] for mobility modelling are proposed and achieve state-of-the-art performance. Feng et al. [14] combine recurrent neural network with attention to capture the periodic regularity in human mobility. Liao et al. [26] utilize the multi-task learning method to predict the location and activity simultaneously. Kong et al. [23] utilize memory network to model the long-term regularity. Zhao et al. [50] modify the general LSTM with time and distance gates to better capture the spatial-temporal correlations. These methods usually train a joint model for all users, they can be regarded as joint models. While joint models perform well in mobility modelling, the privacy issues of them become the stumbling block for preventing their application and development.

5.2 Privacy Protection Methods

We introduce related works of privacy protection from two directions: data protection mechanism (*e.g.*, differential privacy) and privacy preserving models (*e.g.*, federated learning). Related works for data encryption are not discussed here.

Data Protection Mechanisms: It aims to protect the private information in the database from random querying. The representative mechanisms include k-anonymity [19], l-diversity [30], t-closeness [25] and differential privacy [12]. Based on these mechanisms, lots of trajectory data protection solutions [1, 19, 41, 45] are proposed

and achieve promising privacy protection results. However, all of these methods only care about the protection of data and realize the protection by destroying the data structure, which directly harms the performance of mobility modelling.

Federated Learning: With the emerging requirement of privacy protection, researchers try to directly consider the privacy issue in the modelling and system [3, 32, 35]. As one of the most potential solutions for privacy preserving modelling, federated learning [32, 47] is designed to train the joint machine learning models with decentralized data on mobile devices. Further, researchers propose advanced methods [2, 5, 22, 38] to improve the communication efficiency and optimization performance of federated learning. While some works [17, 33] also combine differential privacy into federated learning, they all apply differential privacy operation on the whole model in the aggregation stage, which is inapplicable and inefficient in mobility prediction task. Meanwhile, some researchers directly apply federated learning to different application scenarios to protect the user privacy, like recommendation [6], language modelling [33], transfer learning [29] and so on.

Different from these existing works, we focus on the mobility prediction task and try to propose a privacy-preserving framework to solve the unique privacy issue in it with maintaining the competitive prediction performance. Our methods can also be applied to other similar scenarios where the private item (as location in mobility prediction task) feature is modeled by one-hot embedding component. For example, after a slight modification for the differential privacy mechanism, our method can also be used to protect the private information of which item is purchased by users in the recommendation system.

6 CONCLUSION

In this paper, we investigated the problem of modelling human mobility with privacy constraints. Based on the concept of federated learning, we proposed a practical mobility prediction framework to achieve promising prediction performance while preserving the personal data on local devices. We designed a group optimization mechanism on local devices by training different parts (risky/secure) of the overall model with different types (protected/normal) of data. In this way, the effects of protecting privacy on modelling performance can be easily controlled and significantly reduced. Furthermore, we proposed an efficient global optimization strategy and an effective polling schema for client selection to accelerate the convergence rate of the whole system. Finally, we also proposed a personal adaptor to be fine-tuned in the local device for better prediction performance. The future work of our work can be divided into two directions: 1) designing advanced mobility model for better personal modelling. In this paper, we only consider the basic mobility model for the simplicity of the whole system. 2) expanding our framework to more types of machine learning models and different scenarios.

ACKNOWLEDGMENTS

This work was supported in part by The National Key Research and Development Program of China under grant SQ2018YFB180012, the National Nature Science Foundation of China under 61971267, 61972223, 61861136003, and 61621091, Beijing Natural Science Foundation under L182038, Beijing National Research Center for Information Science and Technology under 20031887521, and research fund of Tsinghua University - Tencent Joint Laboratory for Internet Innovation Technology.

REFERENCES

[1] Osman Abul, Francesco Bonchi, Mirco Nanni, et al. 2008. Never Walk Alone: Uncertainty for Anonymity in Moving Objects Databases.. In *ICDE*, Vol. 8. 376–385.

[2] Naman Agarwal, Ananda Theertha Suresh, Felix Xinnan X Yu, Sanjiv Kumar, and Brendan McMahan. 2018. cpSGD: Communication-efficient and differentially-private distributed SGD. In *Advances in Neural Information Processing Systems*. 7564–7575.

[3] Mohammad Al-Rubaie and J Morris Chang. 2019. Privacy-Preserving Machine Learning: Threats and Solutions. *IEEE Security & Privacy* 17, 2 (2019), 49–58.

[4] Miguel E Andrés, Nicolás E Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. 2012. Geo-indistinguishability: Differential privacy for location-based systems. *arXiv preprint arXiv:1212.1984* (2012).

[5] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konecny, Stefano Mazzocchi, H Brendan McMahan, et al. 2019. Towards federated learning at scale: System design. *arXiv preprint arXiv:1902.01046* (2019).

[6] Fei Chen, Zhenhua Dong, Zhenguo Li, and Xiuqiang He. 2018. Federated meta-learning for recommendation. *arXiv preprint arXiv:1802.07876* (2018).

[7] Meng Chen, Yang Liu, and Xiaohui Yu. 2014. Nlpmm: A next location predictor with markov modeling. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 186–197.

[8] Chen Cheng, Haiqin Yang, Irwin King, and Michael R Lyu. 2012. Fused matrix factorization with geographical and social influence in location-based social networks. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*.

[9] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[10] Yves-Alexandre De Montjoye, César A Hidalgo, Michel Verleysen, and Vincent D Blondel. 2013. Unique in the crowd: The privacy bounds of human mobility. *Scientific reports* 3 (2013), 1376.

[11] Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. 2016. Recurrent marked temporal point processes: Embedding event history to vector. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1555–1564.

[12] Cynthia Dwork. 2011. Differential privacy. *Encyclopedia of Cryptography and Security* (2011), 338–340.

[13] Huifang Feng, Chunfeng Liu, Yantai Shu, and Oliver WW Yang. 2015. Location prediction of vehicles in VANETs using a Kalman filter. *Wireless personal communications* 80, 2 (2015), 543–559.

[14] Jie Feng, Yong Li, Chao Zhang, Funing Sun, Fanchao Meng, Ang Guo, and Depeng Jin. 2018. Deepmove: Predicting human mobility with attentional recurrent networks. In *Proceedings of the 2018 World Wide Web Conference*. International World Wide Web Conferences Steering Committee, 1459–1468.

[15] Jie Feng, Mingyang Zhang, Huandong Wang, Zeyu Yang, Chao Zhang, Yong Li, and Depeng Jin. 2019. DPLink: User Identity Linkage via Deep Neural Network From Heterogeneous Mobility Data. In *The World Wide Web Conference*. ACM, 459–469.

[16] Sébastien Gambs, Marc-Olivier Killijian, and Miguel Núñez del Prado Cortez. 2012. Next place prediction using mobility markov chains. In *Proceedings of the First Workshop on Measurement, Privacy, and Mobility*. ACM, 3.

[17] Robin C. Geyer, Tassilo Klein, and Moin Nabi. 2017. Differentially Private Federated Learning: A Client Level Perspective. (2017).

[18] Marta C Gonzalez, Cesar A Hidalgo, and Albert-Laszlo Barabasi. 2008. Understanding individual human mobility patterns. *nature* 453, 7196 (2008), 779.

[19] Marco Gramaglia and Marco Fiore. 2015. Hiding mobile traffic fingerprints with glove. In *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies*. ACM, 26.

[20] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[21] Shan Jiang, Yingxiang Yang, Siddharth Gupta, Daniele Veneziano, Shounak Athavale, and Marta C González. 2016. The TimeGeo modeling framework for urban mobility without travel surveys. *Proceedings of the National Academy of Sciences* 113, 37 (2016), E5370–E5378.

[22] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. 2016. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492* (2016).

[23] Dejiang Kong and Fei Wu. 2018. HST-LSTM: A Hierarchical Spatial-Temporal Long-Short Term Memory Network for Location Prediction.. In *IJCAI*. 2341–2347.

[24] Vaibhav Kulkarni and Benoit Garbinato. 2019. 20 Years of Mobility Modeling & Prediction: Trends, Shortcomings & Perspectives. *arXiv preprint arXiv:1906.07451* (2019).

[25] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. 2007. t-closeness: Privacy beyond k-anonymity and l-diversity. In *2007 IEEE 23rd International Conference on Data Engineering*. IEEE, 106–115.

[26] Dongliang Liao, Weiqing Liu, Yuan Zhong, Jing Li, and Guowei Wang. 2018. Predicting Activity and Location with Multi-task Context Aware Recurrent Neural Network.. In *IJCAI*. 3435–3441.

[27] Ziqian Lin, Jie Feng, Ziyang Lu, Yong Li, and Depeng Jin. 2019. DeepSTN+: Context-aware Spatial-Temporal Neural Network for Crowd Flow Prediction in Metropolis.

[28] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. Predicting the next location: A recurrent model with spatial and temporal contexts. In *Thirtieth AAAI Conference on Artificial Intelligence*.

[29] Yang Liu, Tianjian Chen, and Qiang Yang. 2018. Secure Federated Transfer Learning. *arXiv preprint arXiv:1812.03337* (2018).

[30] Ashwin Machanavajjhala, Johannes Gehrke, Daniel Kifer, and Muthuramakrishnan Venkitasubramaniam. 2006. l-diversity: Privacy beyond k-anonymity. In *22nd International Conference on Data Engineering (ICDE'06)*. IEEE, 24–24.

[31] Wesley Mathew, Ruben Raposo, and Bruno Martins. 2012. Predicting future locations with hidden Markov models. In *Proceedings of the 2012 ACM conference on ubiquitous computing*. ACM, 911–918.

[32] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Artificial Intelligence and Statistics*. 1273–1282.
[33] H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. 2018. Learning differentially private recurrent language models. (2018).
[34] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.
[35] Payman Mohassel and Yupeng Zhang. 2017. Secureml: A system for scalable privacy-preserving machine learning. In *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 19–38.
[36] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. 2018. Film: Visual reasoning with a general conditioning layer. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
[37] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*. ACM, 811–820.
[38] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. 2017. Federated multi-task learning. In *Advances in Neural Information Processing Systems*. 4424–4434.
[39] Jun Tang, Aleksandra Korolova, Xiaolong Bai, Xueqiang Wang, and Xiaofeng Wang. 2017. Privacy loss in apple's implementation of differential privacy on macos 10.12. *arXiv preprint arXiv:1709.02753* (2017).
[40] Zhen Tu, Kai Zhao, Fengli Xu, Yong Li, Li Su, and Depeng Jin. 2017. Beyond k-anonymity: protect your trajectory from semantic attack. In *2017 14th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE, 1–9.
[41] Zhen Tu, Kai Zhao, Fengli Xu, Yong Li, Li Su, and Depeng Jin. 2018. Protecting Trajectory From Semantic Attack Considering k-Anonymity, k-Diversity, and t-Closeness. *IEEE Transactions on Network and Service Management* 16, 1 (2018), 264–278.
[42] Jingyuan Wang, Ning Wu, Wayne Xin Zhao, Fanzhang Peng, and Xin Lin. 2019. Empowering A* Search Algorithms with Neural Networks for Personalized Route Recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 539–547.
[43] Hao Wu, Ziyang Chen, Weiwei Sun, Baihua Zheng, and Wei Wang. 2017. Modeling trajectories with recurrent neural networks. IJCAI.
[44] Ruizhi Wu, Guangchun Luo, Junming Shao, Ling Tian, and Chengzong Peng. 2018. Location prediction on trajectory data: A review. *Big Data Mining and Analytics* 1, 2 (2018), 108–127.
[45] Yonghui Xiao and Li Xiong. 2015. Protecting locations with differential privacy under temporal correlations. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1298–1309.
[46] Dingqi Yang, Daqing Zhang, Vincent. W. Zheng, and Zhiyong Yu. 2015. Modeling User Activity Preference by Leveraging User Spatial Temporal Characteristics in LBSNs. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 45, 1 (2015), 129–142.
[47] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)* 10, 2 (2019), 12.
[48] Chao Zhang, Keyang Zhang, Quan Yuan, Haoruo Peng, Yu Zheng, Tim Hanratty, Shaowen Wang, and Jiawei Han. 2017. Regions, periods, activities: Uncovering urban dynamics via cross-modal representation learning. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 361–370.
[49] Chao Zhang, Keyang Zhang, Quan Yuan, Luming Zhang, Tim Hanratty, and Jiawei Han. 2016. Gmove: Group-level mobility modeling using geo-tagged social media. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1305–1314.
[50] Pengpeng Zhao, Haifeng Zhu, Yanchi Liu, Jiajie Xu, Zhixu Li, Fuzheng Zhuang, S. Sheng Victor, and Zhou Xiaofeng. 2019. Where to Go Next: A Spatio-Temporal Gated Network for Next POI Recommendation. In *Thirty-Third AAAI Conference on Artificial Intelligence*.