Collecting Individual Trajectories under Local Differential Privacy

Jianyu Yang[1], Xiang Cheng[2*], Sen Su[3], Huizhong Sun[4], Changju Chen[5]
State Key Laboratory of Networking and Switching Technology,
Beijing University of Posts and Telecommunications, Beijing, China
{jyyang[1], chenchangjv[5]}@bupt.cn {chengxiang[2], susen[3], showlostage1[4]}@bupt.edu.cn

2022 23rd IEEE International Conference on Mobile Data Management (MDM) | 978-1-6654-5176-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/MDM55031.2022.00035

***Abstract*—In this paper, we tackle the problem of collecting individual trajectories under local differential privacy. The key challenge is how to achieve high utility of the collected trajectories while satisfying the strong privacy guarantee. To overcome this challenge, we present a novel approach, which is referred to as PrivTC. In PrivTC, we first design a locally differentially private grid construction method to instruct the aggregator to lay an appropriate grid on the given geospatial domain. Then we propose a locally differentially private spectral learning method to help the aggregator learn the Hidden Markov Model (HMM) from users' trajectories discretized by the constructed grid. Finally, the aggregator generates a synthetic trajectory dataset as a surrogate for the original one from the learned HMM. Extensive experiments on real datasets confirm the effectiveness of PrivTC.**

***Index Terms*—trajectory, collection, local differential privacy**

I. Introduction

An individual trajectory is a sequence of geographic locations in chronological order, which describes a person's movement in space. With the deep penetration of smart mobile devices, users' trajectories have become widely collected by many service providers such as Google and Uber via their location-based applications. With these collected trajectories, service providers can offer better user experience and generate new revenue opportunities. For example, through analyzing users' movement patterns, service providers can help users plan routes to avoid traffic jams and guide business companies to place advertisements in frequently visited locations. However, users' trajectories are highly sensitive since they may disclose their home addresses or visits to sensitive locations such as hospitals. Without proper privacy protection mechanisms, directly collecting users' trajectories either puts individual privacy at risk or hinders business operations as users are unwilling to share their true trajectories. Thus, developing solutions to address such privacy concerns in collecting individual trajectories is an urgent need.

Local differential privacy (LDP) [1] has recently emerged as the *de facto* standard for individual privacy protection. It has been adopted in many real world applications, including Google Chrome browser [2] and iOS [3]. Different from the traditional differential privacy (DP) setting [4], which assumes a trusted data aggregator that has access to the private data, LDP does not make assumptions about the credibility of the data aggregator. In LDP, data are perturbed on each individual's device before being sent to the aggregator. Since real data never leave from users' devices, LDP protects both users against privacy risks and aggregator against damages due to potential privacy breaches. Existing studies [5]–[10] mainly focus on generating synthetic trajectories in the traditional DP setting. Their approaches cannot be applied to privately collecting individual trajectories.

In this paper, we tackle the problem of collecting individual trajectories under LDP. Given a large number of users, each of which has a trajectory in the given geospatial domain, an untrusted aggregator aims at collecting trajectories from these users while satisfying LDP. A general idea for solving this problem is to learn the well-known Hidden Markov Model (HMM) [11] to approximate the overall distribution of users' trajectories in a locally differentially private manner and generate a synthetic trajectory dataset as a surrogate for the original one from the learned model. However, based on this idea, designing an effective approach which enables the aggregator to obtain the synthetic trajectory dataset with high utility is nontrivial due to the following two technical challenges.

The first challenge comes from the fact that an individual trajectory consists of geographic locations whose latitude and longitude coordinates are continuous. Directly building the HMM on users' original trajectories will lead to an unbounded number of possibilities to simulate a move from one location to another when synthesizing a trajectory. The second one is how to accurately learn the HMM under LDP. Intuitively, one may attempt to enforce LDP on the canonical Baum-Welch algorithm [12] to achieve this task. However, such solution requires many interactions between the aggregator and users, leading to high network latency and excessive noise added for achieving LDP, and thus becomes infeasible in the real-world setting.

Overcoming these two challenges, we present a novel approach for collecting individual trajectories under LDP, which is referred to as PrivTC. In PrivTC, we design a locally differentially private grid construction method called PrivAG, which can adaptively partition the given geospatial domain into a reasonable grid. According to the constructed grid, the users can discretize their trajectories to bound the number of probabilities. Therefore, the first challenge is solved. We further propose a locally differentially private spectral learning method called PrivSL, which employs spectral learning [13]

*Corresponding author

2375-0324/22/$31.00 ©2022 IEEE
DOI 10.1109/MDM55031.2022.00035

Authorized licensed use limited to: National Yang Ming Chiao Tung University. Downloaded on December 25,2023 at 12:01:28 UTC from IEEE Xplore. Restrictions apply.

to learn the HMM from users' discretized trajectories. Since PrivSL requires the aggregator to interact with each user only once, it overcomes the second challenge.

Overall, PrivTC works as follows. At the beginning of PrivTC, the aggregator randomly divides users into two groups U_1 and U_2. Then, the aggregator uses PrivAG to partition the given geospatial domain into a grid G by interacting with U_1 and broadcasts the grid G to each user in U_2. After that, the aggregator uses PrivSL to learn the HMM from the U_2's trajectories discretized according to G. Finally, the aggregator generates a synthetic trajectory dataset from the learned HMM.

Contributions. The contributions of this work are summarized as follows:

- We formulate the problem of collecting individual trajectories under LDP and present a novel approach called PrivTC for solving this problem.
- We design PrivAG for private grid construction to help users discretize their trajectories, which includes a guideline for properly choosing the grid granularities. We propose PrivSL for private spectral learning to obtain the HMM from users' discretized trajectories.
- We conduct extensive experiments to evaluate the performance of different approaches over real datasets. The results demonstrate the effectiveness of PrivTC.

Roadmap. Section II provides the preliminaries and problem statement. Section III gives the details of our approach. Section IV shows our experimental results. Section V reviews related work. Finally, Section VI concludes this paper.

II. Preliminaries and Problem Statement

A. Local Differential Privacy

Local differential privacy (LDP) [1] is defined as follows.

Definition 1 (ε-LDP): Given a privacy budget ε, a randomized algorithm $\mathcal{R}$ satisfies ε-LDP, if for any two inputs v and v', and for any possible output $O \in Range(\mathcal{R})$,

$$\Pr[\mathcal{R}(v) = O] \leq \exp(\varepsilon) \cdot \Pr[\mathcal{R}(v') = O], \tag{1}$$

where the probability space is over the coin flips of $\mathcal{A}$.

B. Optimized Local Hashing

The optimized local hashing (OLH) protocol [14] is the state-of-the-art LDP oracle for frequency estimation in a relatively large domain.

In OLH, the i-th user first randomly chooses a hash function H_i from a family of hash functions to hash his/her real value v in the input domain $[c]$ into a new value $H_i(v)$ in a smaller domain $[c']$. Then, the user uses generalized randomized response (GRR) [14], [15] to perturb the hashed value $H_i(v)$ as $y_i = \text{GRR}(H_i(v))$, where $\text{GRR}(\cdot)$ is the perturbation function defined in the following:

$$\forall_{y \in [c']} \ \Pr[y = \text{GRR}(H(v))] = \begin{cases} p = \frac{e^\varepsilon}{e^\varepsilon + c' - 1}, & \text{if } y = H(v) \\ p' = \frac{1}{e^\varepsilon + c' - 1}, & \text{if } y \neq H(v) \end{cases}.$$

Finally, the user reports $\langle H_i, y_i \rangle$ to the aggregator.

Due to $\frac{p}{p'} = e^\varepsilon$, OLH satisfies ϵ-LDP. For each value $v \in [c]$, to compute its frequency, the aggregator first computes $|\{i \mid H_i(v) = y_i\}| = \sum_{i \in [n]} \mathbb{1}_{\{H_i(v) = y_i\}}$, and then transforms it to its unbiased estimation

$$f_v = \frac{1}{n} \sum_{i \in [n]} \frac{\mathbb{1}_{\{H_i(v) = y_i\}} - 1/c'}{p - 1/c'},$$

where $\mathbb{1}_{\{H_i(v) = y_i\}}$ is the indicator function that returns 1 if $H_i(v)$ equals y_i while returns 0 if not, and n is the total number of users.

In [14], it is shown that setting $c' = e^\varepsilon + 1$ can minimize the estimation variance of OLH, which is

$$\mathsf{Var}[f_v] = \frac{4e^\varepsilon}{n \cdot (e^\varepsilon - 1)^2}, \tag{2}$$

TABLE I: Notations

Notation	Description
$\mathcal{L}$	The 2-D geospatial domain
n	The total number of users
T	The trajectory in $\mathcal{L}$
t	The trajectory length
G	The constructed grid on $\mathcal{L}$
I^G	The set of indexes of the cells in G
T^G	The discretized trajectory according to G

C. User Division Principle

The user division principle is that when gathering multiple pieces of information from users under LDP, dividing users into groups and having each group report information with the full privacy budget can achieve higher utility of results than splitting privacy budget. This principle has been applied in many existing LDP works [16]–[19]. We will also adopt the user division principle in our proposed approach.

D. Problem Statement

Consider there is a two-dimensional (2-D) geospatial domain $\mathcal{L}$. A location l is represented by the latitude and longitude coordinates in $\mathcal{L}$. Let n be the total number of users. The i-th user's trajectory is a t-length ordered list of locations, denoted by $T_i = \langle l_1^i, l_2^i, \cdots, l_t^i \rangle$ where $T_i(j) = l_j^i$ means the location at the j-th timestamp in T_i.

We study the problem of collecting individual trajectories under LDP. In our problem setting, we assume that there is a data aggregator that does not have access to the users' raw trajectories. Our goal is to design an approach to enable the untrusted data aggregator to collect the overall distribution of trajectories from the n users and generate a synthetic trajectory dataset that approximates the users' original trajectories while satisfying ε-LDP.

Please see Table I for the list of notations.

III. Our Approach

In this section, we first elaborate our approach PrivTC for collecting individual trajectories under LDP in Section III-A-III-D. Then we give its analysis in Section III-E.

Authorized licensed use limited to: National Yang Ming Chiao Tung University. Downloaded on December 25,2023 at 12:01:28 UTC from IEEE Xplore. Restrictions apply.

A. Overview of PrivTC

The approach PrivTC consists of three phases in total. In PrivTC, the aggregator first randomly divides n users into two groups U_1 and U_2, which report information for Phases 1 and 2, respectively.

Specifically, the three phases are described in the following:

Phase 1: Grid Construction. With users in U_1, the aggregator first invokes a locally differentially private grid construction method, called PrivAG, to construct a grid G by partitioning the given 2-D domain $\mathcal{L}$ into a set of cells. Then, the aggregator broadcasts grid G to users in the other group U_2. The details of PrivAG are presented in Section III-B.

Phase 2: HMM Learning. According to the grid G, each user in U_2 discretizes his/her trajectory by replacing every location with the index of cell which it belongs to. Then the aggregator invokes a locally differentially private spectral learning method, namely PrivSL, to learn the HMM over these discretized trajectories. The details of PrivSL are illustrated in Section III-C.

Phase 3: Trajectory Synthesizing. The aggregator independently synthesizes n trajectories from the learned HMM. The details of synthesizing a trajectory are shown in Section III-D.

B. Grid Construction under LDP

A high-quality grid on the given 2-D geospatial domain can help users reasonably discretize their trajectories and enhance the utility of the generated synthetic trajectory dataset.

A straightforward method for constructing a grid is to uniformly partition the 2-D domain $\mathcal{L}$ into $g \times g$ cells of equal size. However, due to treating all regions in $\mathcal{L}$ in the same way, the method may result in irrational partitioning, losing the statistic features of the original dataset and reducing the utility of the synthetic trajectory dataset. In particular, for a sparse region that has very few locations, this method may produce over-partitioning of this region, generating too many cells with nearly zero locations; for a dense region that has many locations, this method may lead to under-partitioning of the region, making users' discretized trajectories indistinguishable.

To solve this problem, we design a locally differentially private grid construction method called Private Adaptive Grid (PrivAG). Its main idea is to firstly lay a coarse grid G_1 over the given 2-D domain $\mathcal{L}$, and then further partition each cell in G_1 based on its noisy frequency to form the final grid G.

Specifically, in PrivAG, the aggregator first uniformly partitions the given 2-D geospatial domain $\mathcal{L}$ into a $g_1 \times g_1$ grid G_1, and then broadcasts it to the users in group U_1. Next, based on the received grid G_1, each i-th user in U_1 discretized his/her trajectory $T_i = \langle l_1^i, l_2^i, \cdots, l_t^i \rangle$ into $T_i^{G_1} = \langle d_1^i, d_2^i, \cdots, d_t^i \rangle$, where $T_i^{G_1}(j) = d_j^i$ is the index of the cell containing the location l_j^i.

After that, the aggregator uses OLH to collect the noisy frequency of each cell in G_1 from the users in U_1. More concretely, the aggregator evenly divides the group U_1 into t subgroups, where each subgroup reports the information for one timestamp in the discretized trajectories. Then, for each j-th timestamp, the aggregator uses OLH to obtain the noisy frequency f_k^j of the k-th cell in G_1 at the timestamp from the j-th subgroup's discretized trajectories. The overall noisy frequency f_k of the k-th cell in G_1 is computed as the average of those on all t timestamps, i.e.,

$$f_k = \frac{1}{t} \cdot \sum_{j=1}^{t} f_k^j.$$

Finally, for each k-th cell in G_1, the aggregator further partitions it into $g_2^k \times g_2^k$ cells based on its noisy frequency f_k.

Choosing Granularities. Since the granularities g_1 and g_2 have a direct impact on the utility of the constructed grid, we propose the following guideline for appropriately choosing them.

Guideline: The domain $\mathcal{L}$ should be firstly partitioned into $g_1 \times g_1$ grid G_1, where g_1 should be

$$g_1 = \sqrt{2\alpha \cdot (e^{\varepsilon} - 1) \cdot \sqrt{\frac{n}{e^{\varepsilon}}}}. \tag{3}$$

Then, for each k-th cell in G_1 ($1 \leq k \leq g_1 \times g_1$) with a noisy frequency f_k, it should be further partitioned into $g_2^k \times g_2^k$ cells, where g_2^k is computed as

$$g_2^k = \sqrt{2\alpha \cdot f_k \cdot (e^{\varepsilon} - 1) \cdot \sqrt{\frac{(1-\sigma)n}{e^{\varepsilon}}}}, \tag{4}$$

where n is the total number of users, t is the length of a trajectory, ε is the total privacy budget of PrivTC, α is some small constant depending on the trajectory dataset, and σ is a constant that denotes the proportion of the number of users in U_1 to n, i.e., $\sigma = \frac{|U_1|}{n}$. Our experimental results suggest that setting α to be in the range of $[0.01, 0.02]$ and σ to be in the range of $[0.1, 0.3]$ can typically achieve good performance across different datasets. Due to space limitation, we refer the readers to our technical report [20] for these results and their analysis. In our subsequent experiments, we set $\alpha = 0.02$ and $\sigma = 0.2$.

We show the analysis to support this guideline as follows. Similar to [21], we evaluate the quality of the constructed grid via the accuracy of range queries on it. Specifically, we assume that the aggregator will use OLH to collect the frequency of each cell in the grid from users' trajectories discretized according to the grid. Given a range query q that specifies a rectangle in the domain $\mathcal{L}$ and asks for the sum of the frequencies of locations within the rectangle. To get the answer f_q of the query q, we check all cells in the grid. If a cell is completely contained in the query rectangle, we include its noisy frequency in f_q; if a cell is partially contained, we first estimate the sum of frequencies of locations in the intersection between the cell and the query rectangle, assuming that the locations within the cell are uniformly distributed and then include the estimated sum in f_q.

In the estimated answer f_q, there are three kinds of errors including sampling error, noise error, and non-uniformity error.

Authorized licensed use limited to: National Yang Ming Chiao Tung University. Downloaded on December 25,2023 at 12:01:28 UTC from IEEE Xplore. Restrictions apply.

The sampling error is caused by dividing users into groups and using the frequencies collected from a group to represent those collected from all users, since the distribution over a group may be different from the global one over all users. The noise error is due to the use of OLH for achieving LDP in the collecting process.

The sampling error and noise error can be quantified together. We first analyse the squared error for the noisy frequency f_k^j of the k-th cell in G_1 at the j-th timestamp, where f_k^j is obtained from the discretized trajectories in the j-th group by OLH. Assuming that in the collecting process, there are n users in total and they are divided into t groups, where each group reports for one timestamp. Then, based on the analysis in [19], we can easily derive that the expected squared sampling and noise error for f_k^j is dominated as

$$\mathsf{Var}\left[f_k^j\right] = \frac{4te^{\varepsilon}}{n \cdot (e^{\varepsilon}-1)^2}.$$

(Due to space limitation, we present the detailed derivation in our technical report [20].) Thus, the overall noisy frequency f_k of the k-th cell in G_1 has a variance of

$$\begin{aligned}\mathsf{Var}\left[f_k\right] &= \mathsf{Var}\left[\frac{1}{t}\cdot\sum_{j=1}^{t} f_k^j\right] = \frac{1}{t^2}\cdot\sum_{j=1}^{t}\mathsf{Var}\left[f_k^j\right] \\ &= \frac{1}{t^2}\cdot t\cdot\frac{4t\cdot e^{\varepsilon}}{n\cdot(e^{\varepsilon}-1)^2} = \frac{4e^{\varepsilon}}{n(e^{\varepsilon}-1)^2}.\end{aligned}$$

The total squared sampling and noise error in f_q equals the sum of variance of cells included in the query rectangle.

The non-uniformity error is incurred by the cells partially included in the query rectangle. Since locations within the same cell in a grid are reported together, when answering the query with the cells of this kind, the aggregator needs to estimate the frequencies of locations in the common area between the cells and the query by assuming that the locations in these cells are uniformly distributed. The accurate amount of the non-uniformity error depends on the true distribution of the dataset, which is not accessible for the aggregator in our problem setting. Therefore, we decide to calculate the approximate non-uniformity error in the subsequent analysis.

Intuitively, for a fine-grained grid, the query will include more cells, leading to larger sampling and noise error and lower non-uniformity error; while for a coarse-grained grid, the query will include less cells, resulting in smaller sampling and noise error and higher non-uniformity error. Hence choosing proper granularities can be considered as a trade-off between these two types of errors.

Analysis on g_1. To analyse the value of g_1, we suppose that the aggregator will collect the frequencies of cells in the $g_1 \times g_1$ grid G_1 from all n users. For a range query q that selects the portion r of the 2-D domain, the query rectangle includes about $r \cdot (g_1)^2$ cells. When answering the query q, the total sampling and noise error is $r \cdot (g_1)^2 \cdot \frac{4e^{\varepsilon}}{n(e^{\varepsilon}-1)^2}$.

The non-uniformity error is proportional to the sum of the frequencies of locations in the cells that fall on the four edges of the query rectangle. For the query q's rectangle, there are $\sqrt{r}\cdot g_1$ cells on each edge and thus $4\sqrt{r}\cdot g_1$ cells on the four edges in total. The expected sum of the frequencies of locations included in these cells is $4\sqrt{r}\cdot g_1 \cdot \frac{1}{g_1\cdot g_1} = \frac{4\sqrt{r}}{g_1}$. Assuming that the non-uniformity error on average is $\frac{4\sqrt{r}}{g_1}\cdot\alpha$, where α is some small constant. Then it has a squared error of $\left(\frac{4\alpha\cdot\sqrt{r}}{g_1}\right)^2$.

To minimize the sum of the two squared errors $r\cdot(g_1)^2\cdot\frac{4e^{\varepsilon}}{n(e^{\varepsilon}-1)^2} + \left(\frac{4\alpha\cdot\sqrt{r}}{g_1}\right)^2$, we should set $g_1 = \sqrt{2\alpha\cdot(e^{\varepsilon}-1)\cdot\sqrt{\frac{n}{e^{\varepsilon}}}}$.

Analysis on g_2. To find the proper value of g_2, we need to assume that the aggregator will collect the frequencies of cells in the final grid G from the $(1-\sigma)\cdot n$ users in group U_2, since the users in group U_1 has been used for estimating the frequencies of cells in the grid G_1.

Similar to the analysis of g_1, for the k-th cell in G_1 that has the frequency f_k and is further partitioned into $g_2^k \times g_2^k$ cells, the range query q's rectangle roughly contains $r\cdot(g_2^k)^2$ cells. Thus the squared sampling and noise error is $r\cdot(g_2^k)^2\cdot\frac{4e^{\varepsilon}}{(1-\sigma)\cdot n(e^{\varepsilon}-1)^2}$.

For the non-uniformity error, there are $4\sqrt{r}\cdot g_2^k$ on four edges of the query rectangle in total. Then the expected sum of the frequencies of locations included in these cells is $4\sqrt{r}\cdot g_2^k\cdot\frac{f_k}{g_2^k\cdot g_2^k} = \frac{4\cdot f_k\cdot\sqrt{r}}{g_2^k}$. The squared error from non-uniformity is $\left(\frac{4\alpha\cdot f_k\cdot\sqrt{r}}{g_2^k}\right)^2$, where α is the same small constant in the analysis of g_1.

For minimizing the sum of the two squared errors

$$r\cdot(g_2^k)^2\cdot\frac{4e^{\varepsilon}}{(1-\sigma)\cdot n(e^{\varepsilon}-1)^2} + \left(\frac{4\alpha\cdot f_k\cdot\sqrt{r}}{g_2^k}\right)^2,$$

the value of g_2^k should be $\sqrt{2\alpha\cdot f_k\cdot(e^{\varepsilon}-1)\cdot\sqrt{\frac{(1-\sigma)n}{e^{\varepsilon}}}}$.

C. Spectral Learning under LDP

According to the constructed grid G, each i-th user in U_2 discretizes his/her trajectory $T_i = \langle l_1^i, l_2^i, \cdots, l_t^i\rangle$ into $T_i^G = \langle d_1^i, d_2^i, \cdots, d_t^i\rangle$, where $T_i^G(j) = d_j^i$ is the index of the cell that l_j^i lies in. To learn the HMM from user group U_2's discretized trajectories, we propose a locally differentially private spectral learning method called PrivSL. Before giving the details of PrivSL, we first introduce the non-private spectral learning algorithm for HMM.

In the non-private algorithm, for each i-th user in U_2, he/she first randomly samples a triplet $\langle T_i^G(x_1), T_i^G(x_2), T_i^G(x_3)\rangle$ from his/her discretized trajectory T_i^G, where (x_1, x_2, x_3) are adjacent timestamps. Then, from these triplets, the aggregator computes three sets of probabilities:

$$\begin{aligned}S_1 =& \{\Pr[T^G(x_1) = d_1] | d_1 \in I^G\},\\ S_2 =& \{\Pr[T^G(x_2) = d_2, T^G(x_1) = d_1] | d_1, d_2 \in I^G\},\\ S_3 =& \{\Pr[T^G(x_3) = d_3, T^G(x_2) = d_2, T^G(x_1) = d_1] | d_1, d_2, d_3 \in I^G\},\end{aligned}$$

where I^G is the set of indexes of grid G's cells.

Next, using the probabilities in $S_1 \cup S_2 \cup S_3$, the aggregator forms a length-$|I^G|$ vector P_1, a $|I^G| \times |I^G|$ matrix $P_{2,1}$, and

Authorized licensed use limited to: National Yang Ming Chiao Tung University. Downloaded on December 25,2023 at 12:01:28 UTC from IEEE Xplore. Restrictions apply.

a set of $|I^G| \times |I^G|$ matrices $\{P_{3,y,1}|y \in I^G\}$, respectively. In particular, the d_1-th value in P_1 is

$$P_1[d_1] = \Pr[T^G(x_1) = d_1];$$

the value of the d_2-th row and d_1-th column in matrix $P_{2,1}$ is

$$P_{2,1}[d_2, d_1] = \Pr[T^G(x_2) = d_2, T^G(x_1) = d_1];$$

the value of d_3-th row and d_1-th column in matrix $P_{3,y,1}$ is

$$P_{3,y,1}[d_3, d_1] = \Pr[T^G(x_3) = d_3, T^G(x_2) = y, T^G(x_1) = d_1].$$

After that, with the given number of the hidden states h, the aggregator computes the singular value decomposition (SVD) of $P_{2,1}$, and obtains M which is the matrix of left singular vectors corresponding to the h largest singular values. Finally, the aggregator computes the spectral learning parameters

$$\vec{b}_1 = M^\top P_1, \quad \vec{b}_\infty = (P_{2,1}^\top M)^+ P_1,$$
$$\{B_y = M^\top P_{3,y,1}(M^\top P_{2,1})^+ | y \in I^G\},$$

which can be applied to synthesizing trajectories. The value of h appears to be less critical. We found that the results are almost the same so long as h is larger than 10 and thus set $h = 10$ for our experiments.

We then clarify our solution to convert the non-private algorithm into a locally differentially private one. In the non-private algorithm, the aggregator only interacts with the users to compute the probabilities in $S_1 \cup S_2 \cup S_3$. Therefore, to make the non-private algorithm locally differentially private, we merely need to instruct the aggregator to collect $S_1 \cup S_2 \cup S_3$ in a private manner.

A naive strategy for such a procedure is to let the aggregator collect probabilities in S_3 using OLH and then derive probabilities in S_1 and S_2 from S_3 via the following sum process:

$$\Pr[T^G(x_1)] = \sum_{T^G(x_3), T^G(x_2)} \Pr[T^G(x_3), T^G(x_2), T^G(x_1)],$$
$$\Pr[T^G(x_2), T^G(x_1)] = \sum_{T^G(x_3)} \Pr[T^G(x_3), T^G(x_2), T^G(x_1)].$$

However, such a strategy will result in a large amount of added noise in S_1 and S_2, since the sum process causes the accumulation of noise in S_3.

To address this deficiency, we propose to further evenly divide users in group U_2 into three subgroups that report information for S_1, S_2 and S_3, respectively. By avoiding the sum process in the naive strategy, this new strategy can significantly improve the accuracy of the collected probabilities in $S_1 \cup S_2 \cup S_3$. Nevertheless, due to using OLH to ensure privacy, the collected probabilities may be negative, which violates the prior knowledge that the true ones are non-negative. Besides, since the three sets of probabilities are collected separately using OLH, they may not hold the natural dependencies among S_1, S_2 and S_3 as shown in the sum process above, leading to inconsistency issue. To further increase the accuracy of collected probabilities, we let the aggregator postprocess these probabilities to remove the negativity and inconsistency among them in the following manner.

Post-Process for Collected Probabilities. The post-process contains the following two basic steps:

1) *Removing Negativity.* In this step, the aggregator employs Norm-Sub [22] to handle the probabilities in S_1, S_2 and S_3, respectively. Its main idea is to firstly convert all negative estimates to 0 and then evenly assigns the total difference between the sum of positive estimates and 1 to each positive estimate. The process is repeated until all estimates become non-negative and sum up to 1.

2) *Removing Inconsistency.* To remove inconsistency among S_1, S_2 and S_3, we need to ensure consistency on timestamps x_1 and x_2. In particular, x_1 is related to $X_1 = \{S_1, S_2, S_3\}$, while x_2 is related to $X_2 = \{S_2, S_3\}$. It is unnecessary to perform consistency operation on x_3, since x_3 is only related to S_3.

Below we describe how to achieve consistency on a timestamp x. For each $d \in I^G$, we define $W_{S_i}(x, d)$ to be the sum of S_i's probabilities corresponding to $T^G(x) = d$. Our goal is to make all $W_{S_i}(x, d)$ consistent. To this end, we first compute their weighted average as $W(x, d) = \sum_{S_i \in X} \theta_i \cdot W_{S_i}(x, d)$. The optimal value of θ_i is obtained by minimizing the variance of $W(x, d)$, i.e., $\mathsf{Var}[W(x, d)] = \sum_{S_i \in X} {\theta_i}^2 \cdot \mathsf{Var}[W_{S_i}(x, d)] = \sum_{S_i \in X} {\theta_i}^2 \cdot |C_i| \cdot \mathsf{Var}_0$, where C_i is a S_i's subset whose probabilities are corresponding to $W_{S_i}(x, d)$, and Var_0 is the basic variance for estimating a single probability. Apparently, we have $|C_1| = 1, |C_2| = |I^G|$ and $|C_3| = |I^G|^2$. Based on the analysis in [18], we should set $\theta_i = \frac{1}{|C_i|} / \sum_{S_i \in X} \frac{1}{|C_i|}$ to obtain the optimal weighted average $W(x, d)$. After $W(x, d)$ is calculated, we update each $W_{S_i}(x, d)$ as $W(x, d)$ in the following manner. For each probability in C_i, we update its value by adding the amount of change $(W(x, d) - W_{S_i}(x, d)) / |C_i|$.

To remove inconsistency, the aggregator can first use the above method for x_1 and then for x_2. It is shown in [23] that the later consistency operation on x_2 will not invalidate consistency established in the previous operation on x_1. Note that removing inconsistency may incur negativity, and vice versa. To resolve this problem, we interchangeably invoke these two steps several times.

To sum up, in PrivSL, users in U_2 are evenly divided into three subgroups, each of which only reports information for one set in $S_1 \cup S_2 \cup S_3$. After collecting the probabilities in $S_1 \cup S_2 \cup S_3$ via OLH, the aggregator runs the post-process to remove negativity and inconsistency among them, and finally uses them to compute the spectral learning parameters as in the non-private algorithm.

D. Synthesizing A Trajectory

To synthesize a trajectory $\hat{T} = \langle l_1, l_2, \cdots, l_t \rangle$, the aggregator first synthesizes its discretized trajectory $\hat{T}^G = \langle d_1, d_2, \cdots, d_t \rangle$, from the learned HMM parameterized by $\{\vec{b}_1, \vec{b}_\infty, \{B_y | y \in I^G\}\}$. In particular, the elements in $\hat{T}^G$ are

Authorized licensed use limited to: National Yang Ming Chiao Tung University. Downloaded on December 25,2023 at 12:01:28 UTC from IEEE Xplore. Restrictions apply.

sampled one by one. The first element $T^G(1) = d_1$ is sampled from the distribution

$$\left\{ \Pr[\hat{T}^G(1) = d_1] = \frac{\vec{b}_\infty^\top B_{d_1} \vec{b}_1}{\sum_y \vec{b}_\infty^\top B_y \vec{b}_1} \,|d_1 \in I^G \right\}.$$

The subsequent k-th element $\hat{T}^G(k) = d_k$ ($2 \leq k \leq t$) is sampled from the distribution $\left\{ \Pr[\hat{T}^G(k) = d_k | d_{1:k-1}] \,|d_k \in I^G \right\}$, where

$$\begin{cases} \vec{b}_k = \frac{B_{d_{k-1}} \vec{b}_{k-1}}{\vec{b}_\infty B_{d_{k-1}} \vec{b}_{k-1}}, \\ \Pr[\hat{T}^G(k) = d_k | d_{1:k-1}] = \frac{\vec{b}_\infty^\top B_{d_k} \vec{b}_k}{\sum_y \vec{b}_\infty^\top B_y \vec{b}_k} \end{cases}.$$

To generate the trajectory $\hat{T}$ from $\hat{T}^G$, for each k-th location l_k in $\hat{T}$, the aggregator sets l_k to be the coordinates of the point that is randomly generated in $\hat{T}^G(k)$-th cell of grid G.

E. Analysis of PrivTC

In the following, we give the analysis of PrivTC including privacy guarantee and communication cost.

Privacy Guarantee. In PrivTC, all the information from each user to the aggregator was sanitized via OLH with privacy budget ε and no other information is leaked. Thus, PrivTC satisfies ε-LDP.

Communication Cost. In PrivTC, the aggregator communicates with users in Phases 1 and 2. In Phase 1, there are two types of messages. One is the coarse-grained grid G_1 sent from the aggregator to the users in group U_1, leading to a cost of $O\left(|U_1| \cdot |I^{G_1}|\right)$, where I^{G_1} denotes the set of indexes of cells in G_1. The other one is the perturbed reports generated by OLH, which are sent to the aggregator by the users in U_1. The total cost of this type is $O(|U_1|)$.

Similar to Phase 1, there are also two types of messages in Phase 2. The first one is the final constructed grid G sent from the aggregator to each user in group U_2, resulting in the cost of $O\left(|U_2| \cdot |I^G|\right)$. The second one is also due to the using of OLH by the users in U_2, generating the cost of $O(|U_2|)$. Therefore, the total communication cost in PrivTC is $O\left(|U_1| \cdot |I^{G_1}| + |U_1| + |U_2| \cdot |I^G| + |U_2|\right) = O\left(n \cdot |I^G|\right)$.

IV. Experiments

In this Section, we first evaluate the performance of PrivTC. Then we judge the effectiveness of our guideline for choosing grid granularities in PrivAG, which is a building block of PrivTC.

A. Experimental Settings

Datasets. We make use of the following two real datasets in our experiments.

- Gowalla [24]: It is collected by the location-based social networking website Gowalla, where users share their locations by checking-in. It contains a total of 6,442,890 check-ins of these users from Feb. 2009 to Oct. 2010.
- Taxi [25]: It describes a complete year from July 2013 to June 2014 of the trajectories for all the 442 taxis running in the city of Porto, in Portugal.

For the two datasets Gowalla and Taxi, we sample 200k and 500k trajectories at equal time intervals, respectively. To experiment with different lengths of trajectories, we generate multiple test datasets with the length ranging from 3 to 15 from the original datasets.

Baseline Approaches. We compare PrivTC against the following three carefully designed baseline approaches:

- **UG.** The approach UG is designed to verify the effectiveness of PrivAG. Different from PrivTC, in UG, the aggregator only uniformly partitions the given domain $\mathcal{L}$ into a $g_1 \times g_1$ grid G_1, where the value of g_1 is set according to the guideline in Section III-B. Then the aggregator invokes PrivSL to learn the HMM from users' trajectories discretized by G_1.
- **NSL.** The approach NSL is used to estimate the effectiveness of PrivSL. The difference between NSL and PrivTC is that NSL adopts the naive strategy to collect the probabilities in $S_1 \cup S_2 \cup S_3$, where S_3 is only collected via OLH and then S_1 and S_2 are derived from S_3 as described in Section III-C.
- **Ngram.** Based on another canonical trajectory model, namely n-gram [6], we further design a baseline approach Ngram to evaluate the effectiveness of HMM model with spectral learning. The distinction between Ngram and PrivTC is that in Ngram, after constructing grid G via PrivAG, the aggregator leverages the n-gram model to model users' trajectories discretized by G and then synthesizes trajectories from the learned n-gram model. In our experiments, we employ the 3-gram to model discretized trajectories under LDP, where the probabilities in S_3 are only required and collected using the naive strategy presented in Section III-C.

To configure PrivAG invoked in PrivTC, NSL and Ngram, with our provided guideline in Section III-B, we first set the recommended $\sigma = 0.2$, $\alpha = 0.02$. Then, for handling a dataset, we use its public information including the number of users n and the privacy budget ε to derive the values of g_1 and g_2 from the equations in our guideline.

Utility Metrics. To evaluate the performance of different approaches, we choose metrics to measure the difference between real trajectory dataset D_r and synthetic dataset D_s. Following three evaluation categories for the trajectory dataset in [26] including spatial density, frequent travel pattern, and spatial-temporal travel, we employ the following three metrics under these categories, respectively.

- **Query MAE.** Given a set Q of range queries, the Query Mean Absolute Error (Query MAE) is computed as

$$\text{Query MAE} = \frac{1}{|Q|} \sum_{q \in Q} |f_q - \overline{f_q}|,$$

where $f_q = \frac{q(D_s)}{n \cdot t}$ and $\overline{f_q} = \frac{q(D_r)}{n \cdot t}$ are the estimated and true answers of query q, function $q(\cdot)$ denotes the count

Authorized licensed use limited to: National Yang Ming Chiao Tung University. Downloaded on December 25,2023 at 12:01:28 UTC from IEEE Xplore. Restrictions apply.

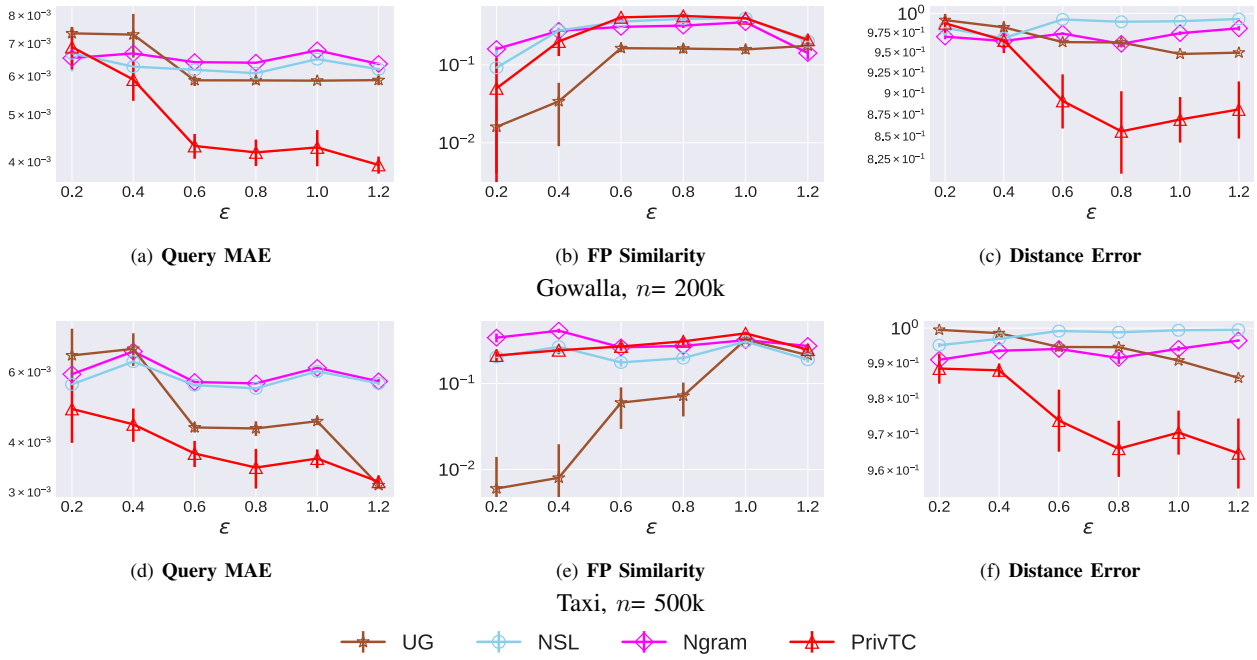

Fig. 1: **Varying ε under setting of t= 9. Results are shown in log scale.**

of locations included in the query rectangle in a dataset, n is the total number of users, and t is the trajectory length of the dataset. To comprehensively estimate the range queries in the entire 2-D domain $\mathcal{L}$, in our experiments, we uniformly partition $\mathcal{L}$ into a 15×15 test grid G_t and use cells in G_t to specify the set of range queries.

- **FP Similarity.** It measures the set similarity between the top-k patterns in D_r and the top-k patterns in D_s. To obtain the top-k patterns of dataset D, we also lay the 15×15 test grid G_t on the given 2-D domain $\mathcal{L}$, then discretize the trajectories in D according to G_t, and finally mine the k patterns with highest support, denoted by $\mathcal{F}_k^{G_t}(D)$. In particular, we define

$$\text{FP Similarity} = F1\left(\mathcal{F}_k^{G_t}(D_r), \mathcal{F}_k^{G_t}(D_s)\right),$$

where $F1(\cdot)$ is the F1-measure, i.e., the harmonic mean of precision and recall. In our experiments, we consider the patterns that are two cells long and set $k = 100$.

- **Distance Error.** We define the distance of a trajectory as the sum of Euclidean Distance between every two locations at adjacent timestamps. Based on the the maximum distance in the real trajectory dataset D_r, we split the maximum distance into 20 equal-width bins. Then, for each of the datasets D_r and D_s, we generate a histogram of distance by counting how many trajectories' distance fall into each bin. Let $H(D_r)$ and $H(D_s)$ denote the two histograms of datasets D_r and D_s, respectively. The distance error is calculated as

$$\text{Distance Error} = JSD(H(D_r), H(D_s)),$$

where $JSD(\cdot)$ is the Jensen-Shannon divergence.

Note that the value of FP similarity is between 0 and 1. Higher values of FP similarity mean the lower difference between the real and synthetic datasets, and thus the higher utility of the synthetic dataset. Conversely, lower values of Query MAE or Distance Error indicate the better performance of approaches.

Methodology. We use $\varepsilon \in \{0.2, 0.4, 0.6, 0.8, 1.0, 1.2\}$ for our experiments. In all subsequent experiments, unless explicitly stated, we use the following default values for other relevant parameters: $\varepsilon = 1.0$, $t = 9$, $\sigma = 0.2$ and $\alpha = 0.02$.

We implemented all approaches using Python3.7. The source code of our approach is publicly available at [20]. All experiments were conducted on servers running Linux kernel version 4.4.0 with Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz and 128GB memory. For each dataset and each approach, we repeat each experiment 5 times and report result mean and standard deviation.

B. Performance of PrivTC

In general, there are two parameters that can affect the performance of PrivTC.

Impact of ε. Figure 1 gives the results for comparing PrivTC against all the baseline approaches under different ε values. As expected, we can observe that the accuracy of all approaches tends to be better when ε increases. Among these approaches, we find that NSL and Ngram have the similar performance under these three metrics. This is because NSL adopts the naive strategy to only collect the probabilities in S_3 using OLH and then derive S_1 and S_2 from S_3, leading to excessive noise in S_1 and S_2, and thus cancelling out the benefit of HMM for

Authorized licensed use limited to: National Yang Ming Chiao Tung University. Downloaded on December 25,2023 at 12:01:28 UTC from IEEE Xplore. Restrictions apply.

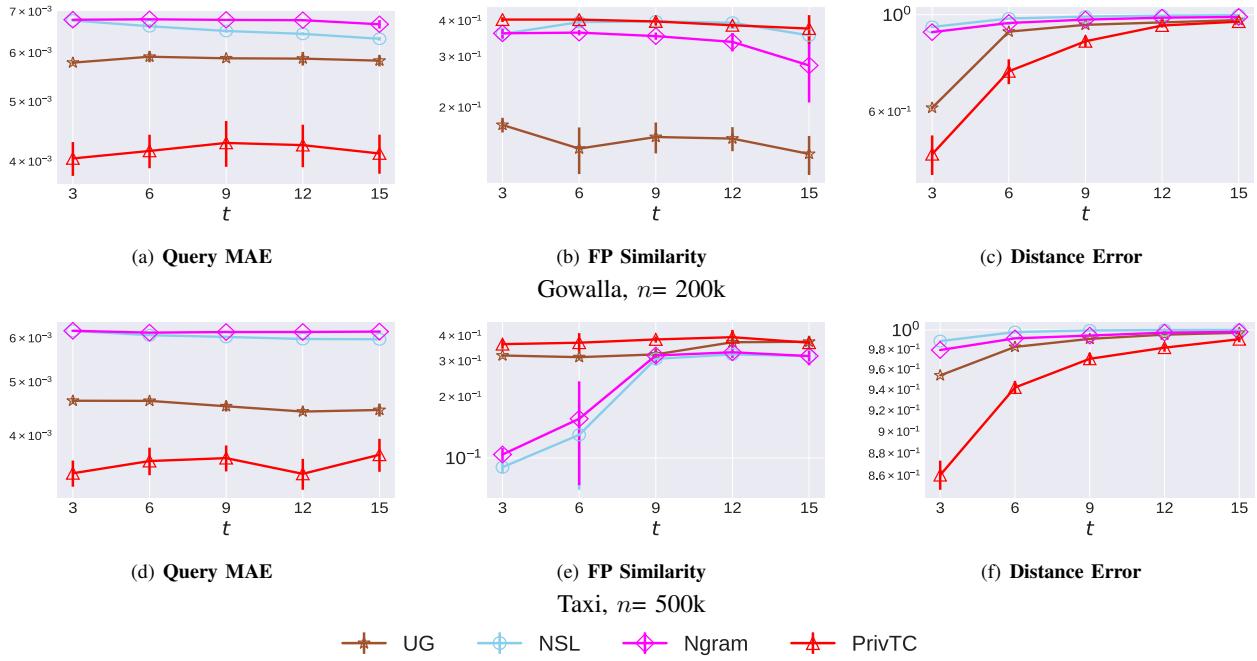

Fig. 2: **Varying t under setting of ε= 1.0. Results are shown in log scale.**

better modelling the trajectory data. For UG, the results of FP Similarity are obviously lower than those of other approaches, making its disadvantage more pronounced, i.e., uniformly partitioning the 2-D geospatial domain is insufficient, losing the statistic features of the original trajectories.

Figure 1 shows that PrivTC has a clear advantage on over all baseline approaches. Compared with UG, PrivTC achieves the higher accuracy, which confirms the effectiveness of PrivAG that adaptively partitions the 2-D geospatial domain. Moreover, we can see that PrivTC outperforms much better than NSL, which indicates the improvement of PrivSL over the naive strategy for collecting probabilities. In addition, compared with Ngram, the superiority of PrivTC is remarkable. The reason can be explained as follows. In PrivTC, the HMM with spectral learning is more effective than the n-gram for modeling trajectories. The intuition behind this is that Ngram only needs the probabilities in S_3 while HMM requires those in $S_1 \cup S_2 \cup S_3$, which is more comprehensive for capturing the features of trajectory. We also see that there are some jumping points of PrivTC. This is because the PrivAG invoked PrivTC chooses different granularities based on the ε value and the total number of users n in the dataset. Although these choices are generally good, they are not optimal for every dataset at every ε value.

Impact of t. Figure 2 presents the results varying t from 3 to 15. From Figure 2, we observe that, for all approaches, their utilities slightly degrade as t grows. The reason can be explained as follows. When t is larger, the users' random sampling of triplets for collecting the probabilities required for model learning introduces a small amount of bias, since the distribution of triplets is used to represent that of the full-length trajectories.

Among these approaches, PrivTC consistently performs the best, especially for the Query MAE, which confirms its good scalability for handling long trajectory data. In particular, for the Distance Error, we can see that when t is relatively small, the advantage of PrivTC compared to Ngram becomes much more remarkable, which confirms its effectiveness for modeling short trajectories.

C. *Effectiveness of Guideline*

As a building block of PrivTC, PrivAG for grid construction has an important influence on the performance of PrivTC. To evaluate the effectiveness of granularities provided by our guideline in PrivAG, we first set a granularity g to implement a version of UG, referred to as UG(g), where the aggregator uniformly partitions the 2-D domain $\mathcal{L}$ into a $g \times g$ grid. Then we compare PrivTC with the implemented versions including UG(5), UG(10), UG(15) and UG(20), to judge if our guideline can give good choices of granularities.

Figure 3 shows the results varying ε from 0.2 to 1.2. From Figure 3, we find that for a fixed ε, when the value of g changes from 5 to 20, the utility of UG(g) first increases, and then degrades. The reason is that $g = 5$ leads to under-partitioning of the 2-D domain while $g = 20$ produces over-partitioning. We also find that even for the same version of UG, it may have distinct performance on different datasets. This is because UG approaches simply uniformly partition the 2-D domain without any information about the distribution of the dataset, and thus the performance cannot be guaranteed when handling diverse datasets. However, we observe that PrivTC

Authorized licensed use limited to: National Yang Ming Chiao Tung University. Downloaded on December 25,2023 at 12:01:28 UTC from IEEE Xplore. Restrictions apply.

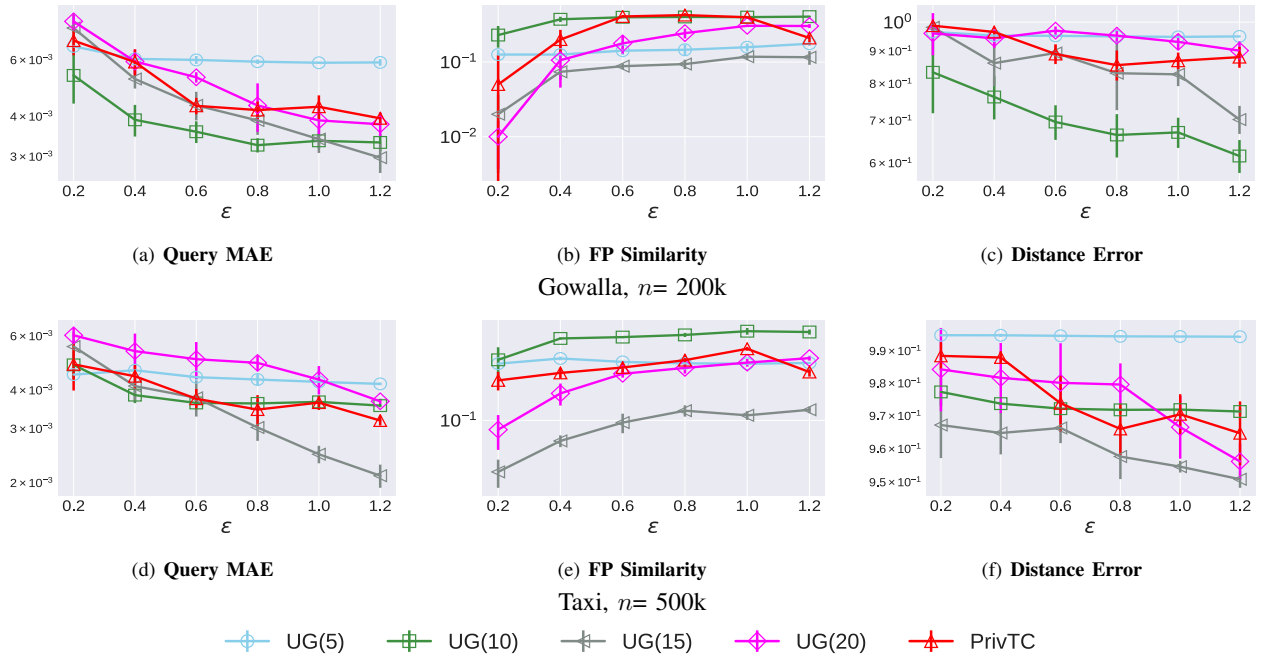

Fig. 3: **Verifying guideline in PrivAG under setting of ε= 1.0, t= 9. Results are shown in log scale.**

performs reasonably well for all ε values on both datasets, which confirms that our guideline can give helpful guidance.

V. Related Work

A series of works [5]–[10] focus on releasing trajectories under traditional differential privacy (DP) [4]. Chen et al. [5] first apply DP into publishing large volume of real-life sequential data by recursively building a noisy prefix tree to group sequences with the identical prefix into the same branch. To improve the utility, Chen et al. [6] then propose a novel approach for differentially private sequential data publication based on a variable-length n-gram model. We have designed a baseline approach by leveraging the n-gram model and examined its performance in Section IV. Hua et al. [7] further present a new approach to publish general time-serial trajectory data under DP by removing the assumption that the trajectories to be published contain a lot of identical prefixes or n-grams. He et al. [8] develop DPT, a system for synthesizing mobility data, where raw trajectories are discretized by hierarchical reference systems for capturing individual movements at different speeds. Gursoy et al. [26] present AdaTrace, a utility-aware location trace synthesizer with differential privacy guarantee and attack resilience. After that, Gursoy et al. [9] propose a framework called DP-Star for differentially private and utility preserving publishing of trajectory data. Both AdaTrace and DP-Star opt to discretize users' trajectories using an adaptive grid, which is first introduced in [21]. In particular, Qardaji et al. [21] propose an Adaptive Grids (AG) approach to adaptively partition the 2-D geospatial domain into a two-level hierarchical grid, which can achieve better utility of the released synopsis than those hierarchy approaches [27]–[29]. Note that the building block method PrivAG of our approach PrivTC also adopts the idea of adaptive grid. However, there are two distinctions between AG and PrivAG. One is that AG is only for 2-D data, while PrivAG is for trajectory data which is multi-dimensional, combining information from multiple timestamps. The other one is that PrivAG collects information for constructing grid by dividing users instead of privacy budget due to the effectiveness of user division principle in LDP setting, and thus provides a novel analysis of different sources of errors and guideline.

The notion of local differential privacy (LDP) was formalized in [1]. As the basic building block for achieving LDP, frequency oracles for categorical and numerical domains are widely investigated by a number of works [2], [14], [30]–[34]. About geospatial data, Chen et al. [35] propose a novel approach for private spatial data aggregation under LDP by taking into account the users' personalized privacy needs. Cunningham et al. [36] develop a n-gram-based method for perturbing spatio-temporal trajectories under LDP. However, since their method needs to incorporate a range of publicly known external knowledge, which cannot be done in our problem scenario, it cannot be applied to solving our problem.

In addition, many subsequent works apply LDP to other data collection tasks, such as frequent items or itemsets [37]–[44], key-value data [45], [46], social graphs [47], [48], telemetry data [49], preference rankings [50], evolving data [51], and marginals [18], [52], [53]. However, all of their approaches are not suitable for collecting trajectories.

VI. Conclusion

In this paper, we present PrivTC, a novel approach for collecting individual trajectories under LDP. In PrivTC, we

Authorized licensed use limited to: National Yang Ming Chiao Tung University. Downloaded on December 25,2023 at 12:01:28 UTC from IEEE Xplore. Restrictions apply.

design a locally differentially private grid construction method PrivAG for adaptively partitioning the given geospatial domain into a grid. In addition, we propose a locally differentially private spectral learning method PrivSL for effectively learning the HMM. We claim that PrivTC guarantees ε-LDP. Our experimental results demonstrate the effectiveness of PrivTC.

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China (Grant No. 61872045 and Grant No. 62072052), and the Innovation Research Group Project of NSFC (61921003).

REFERENCES

[1] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. D. Smith, "What can we learn privately?" in *FOCS*, 2008.
[2] Ú. Erlingsson, V. Pihur, and A. Korolova, "RAPPOR: randomized aggregatable privacy-preserving ordinal response," in *CCS*, 2014.
[3] J. Tang, A. Korolova, X. Bai, X. Wang, and X. Wang, "Privacy loss in apple's implementation of differential privacy on macos 10.12," *CoRR*, vol. abs/1709.02753, 2017.
[4] C. Dwork, F. McSherry, K. Nissim, and A. D. Smith, "Calibrating noise to sensitivity in private data analysis," in *TCC*, 2006.
[5] R. Chen, B. C. M. Fung, B. C. Desai, and N. M. Sossou, "Differentially private transit data publication: a case study on the montreal transportation system," in *KDD*, 2012.
[6] R. Chen, G. Ács, and C. Castelluccia, "Differentially private sequential data publication via variable-length n-grams," in *CCS*, 2012.
[7] J. Hua, Y. Gao, and S. Zhong, "Differentially private publication of general time-serial trajectory data," in *INFOCOM*, 2015.
[8] X. He, G. Cormode, A. Machanavajjhala, C. M. Procopiuc, and D. Srivastava, "DPT: differentially private trajectory synthesis using hierarchical reference systems," *PVLDB*, 2015.
[9] M. E. Gursoy, L. Liu, S. Truex, and L. Yu, "Differentially private and utility preserving publication of trajectory data," *TMC*, 2019.
[10] X. Ding, W. Zhou, S. Sheng, Z. Bao, K. R. Choo, and H. Jin, "Differentially private publication of streaming trajectory data," *Inf. Sci.*, 2020.
[11] L. E. Baum, J. A. Eagon *et al.*, "An inequality with applications to statistical estimation for probabilistic functions of markov processes and to a model for ecology," *Bulletin of the American Mathematical Society*, 1967.
[12] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains," *The annals of mathematical statistics*, 1970.
[13] D. J. Hsu, S. M. Kakade, and T. Zhang, "A spectral algorithm for learning hidden markov models," in *COLT*, 2009.
[14] T. Wang, J. Blocki, N. Li, and S. Jha, "Locally differentially private protocols for frequency estimation," in *USENIX Security*, 2017.
[15] S. L. Warner, "Randomized response: A survey technique for eliminating evasive answer bias," *Journal of the American Statistical Association*, 1965.
[16] G. Cormode, T. Kulkarni, and D. Srivastava, "Answering range queries under local differential privacy," *PVLDB*, 2019.
[17] T. Wang, B. Ding, J. Zhou, C. Hong, Z. Huang, N. Li, and S. Jha, "Answering multi-dimensional analytical queries under local differential privacy," in *SIGMOD*, 2019.
[18] Z. Zhang, T. Wang, N. Li, S. He, and J. Chen, "CALM: consistent adaptive local marginal for marginal release under local differential privacy," in *CCS*, 2018.
[19] J. Yang, T. Wang, N. Li, X. Cheng, and S. Su, "Answering multi-dimensional range queries under local differential privacy," *PVLDB*, 2021.
[20] "Technical report and source code of privtc," [Online], 2021, http://github.com/YangJianyu-bupt/privtc.
[21] W. H. Qardaji, W. Yang, and N. Li, "Differentially private grids for geospatial data," in *ICDE*, 2013.
[22] T. Wang, M. Lopuhaä-Zwakenberg, Z. Li, B. Skoric, and N. Li, "Locally differentially private frequency estimation with consistency," in *NDSS*, 2020.
[23] W. H. Qardaji, W. Yang, and N. Li, "Priview: practical differentially private release of marginal contingency tables," in *SIGMOD*, 2014.
[24] "Gowalla data," http://snap.stanford.edu/data/loc-gowalla.html.
[25] "Taxi trajectory data," http://www.kaggle.com/crailtap/taxi-trajectory.
[26] M. E. Gursoy, L. Liu, S. Truex, L. Yu, and W. Wei, "Utility-aware synthesis of differentially private and attack-resilient location traces," in *CCS*, 2018.
[27] G. Cormode, C. M. Procopiuc, D. Srivastava, E. Shen, and T. Yu, "Differentially private spatial decompositions," in *ICDE*, 2012.
[28] W. H. Qardaji and N. Li, "Recursive partitioning and summarization: a practical framework for differentially private data publishing," in *ASIACCS*, 2012.
[29] W. H. Qardaji, W. Yang, and N. Li, "Understanding hierarchical methods for differentially private histograms," *PVLDB*, 2013.
[30] J. Acharya, Z. Sun, and H. Zhang, "Hadamard response: Estimating distributions privately, efficiently, and with little communication," in *AISTATS*, 2019.
[31] R. Bassily and A. D. Smith, "Local, private, efficient protocols for succinct histograms," in *STOC*, 2015.
[32] M. Ye and A. Barg, "Optimal schemes for discrete distribution estimation under locally differential privacy," *TIT*, 2018.
[33] S. Wang, Y. Qian, J. Du, W. Yang, L. Huang, and H. Xu, "Set-valued data publication with local privacy: Tight error bounds and efficient mechanisms," *PVLDB*, 2020.
[34] Z. Li, T. Wang, M. Lopuhaä-Zwakenberg, N. Li, and B. Skoric, "Estimating numerical distributions under local differential privacy," in *SIGMOD*, 2020.
[35] R. Chen, H. Li, A. K. Qin, S. P. Kasiviswanathan, and H. Jin, "Private spatial data aggregation in the local setting," in *ICDE*, 2016.
[36] T. Cunningham, G. Cormode, H. Ferhatosmanoglu, and D. Srivastava, "Real-world trajectory sharing with local differential privacy," *PVLDB*, 2021.
[37] R. Bassily, K. Nissim, U. Stemmer, and A. G. Thakurta, "Practical locally private heavy hitters," in *NIPS*, 2017.
[38] M. Bun, J. Nelson, and U. Stemmer, "Heavy hitters and the structure of local privacy," in *PODS*, 2018.
[39] X. Gu, M. Li, L. Xiong, and Y. Cao, "Providing input-discriminative protection for local differential privacy," in *ICDE*, 2020.
[40] J. Hsu, S. Khanna, and A. Roth, "Distributed private heavy hitters," in *ICALP*, 2012.
[41] Z. Qin, Y. Yang, T. Yu, I. Khalil, X. Xiao, and K. Ren, "Heavy hitter estimation over set-valued data with local differential privacy," in *CCS*, 2016.
[42] N. Wang, X. Xiao, Y. Yang, T. D. Hoang, H. Shin, J. Shin, and G. Yu, "Privtrie: Effective frequent term discovery under local differential privacy," in *ICDE*, 2018.
[43] S. Wang, L. Huang, Y. Nie, P. Wang, H. Xu, and W. Yang, "Privset: Set-valued data analyses with locale differential privacy," in *INFOCOM*, 2018.
[44] T. Wang, N. Li, and S. Jha, "Locally differentially private frequent itemset mining," in *SP*, 2018.
[45] X. Gu, M. Li, Y. Cheng, L. Xiong, and Y. Cao, "PCKV: locally differentially private correlated key-value data collection with optimized utility," in *USENIX Security*, 2020.
[46] Q. Ye, H. Hu, X. Meng, and H. Zheng, "Privkv: Key-value data collection with local differential privacy," in *SP*, 2019.
[47] Z. Qin, T. Yu, Y. Yang, I. Khalil, X. Xiao, and K. Ren, "Generating synthetic decentralized social graphs with local differential privacy," in *CCS*, 2017.
[48] H. Sun, X. Xiao, I. Khalil, Y. Yang, Z. Qin, W. H. Wang, and T. Yu, "Analyzing subgraph statistics from extended local views with decentralized differential privacy," in *CCS*, 2019.
[49] B. Ding, J. Kulkarni, and S. Yekhanin, "Collecting telemetry data privately," in *NIPS*, 2017.
[50] J. Yang, X. Cheng, S. Su, R. Chen, Q. Ren, and Y. Liu, "Collecting preference rankings under local differential privacy," in *ICDE*, 2019.
[51] M. Joseph, A. Roth, J. Ullman, and B. Waggoner, "Local differential privacy for evolving data," in *NIPS*, 2018.
[52] G. Cormode, T. Kulkarni, and D. Srivastava, "Marginal release under local differential privacy," in *SIGMOD*, 2018.
[53] X. Ren, C. Yu, W. Yu, S. Yang, X. Yang, J. A. McCann, and P. S. Yu, "Lopub: High-dimensional crowdsourced data publication with local differential privacy," *TIFS*, 2018.

Authorized licensed use limited to: National Yang Ming Chiao Tung University. Downloaded on December 25,2023 at 12:01:28 UTC from IEEE Xplore. Restrictions apply.