Hindawi
Security and Communication Networks
Volume 2022, Article ID 2045586, 14 pages
https://doi.org/10.1155/2022/2045586

Research Article

Secure Data Publishing of Private Trajectory in Edge Computing of IoT

Xinyao Liu, Baojiang Cui, Junsong Fu, Zishuai Cheng, and Xuyan Song

School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing 100876, China

Correspondence should be addressed to Baojiang Cui; cuibj@bupt.edu.cn

Received 22 January 2022; Revised 13 April 2022; Accepted 9 May 2022; Published 8 June 2022

Academic Editor: Marimuthu Karuppiah

Copyright © 2022 Xinyao Liu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Secure data publishing of private trajectory is a typical application scene in the Internet of Things (IoT). Protecting users' privacy while publishing data has always been a long-term challenge. In recent years, the mainstream method is to combine the Markov model and differential privacy (DP) mechanism to build a private trajectory generation model and publishes the generated synthetic trajectory data instead of the original data. However, Markov cannot effectively model the long-term trajectory data spatio-temporal correlation, and the DP noise results in the low availability of the synthetic data. To protect users' privacy and improve the availability of synthetic trajectory data, we propose a trajectory generation model with differential privacy and deep learning (DTG). In DTG, we design a private hierarchical adaptive grid method. It divides the geospatial region into several subregions according to the density of positions to realize the discretization of coordinates of the trajectory data. Second, GRU is used to capture the temporal features of the trajectory sequence for good availability, and we generate synthetic trajectory data by predicting the next position. Third, we adopt the optimizer perturbation method in gradient descent to protect the privacy of model parameters. Finally, we experimentally compare DTG with the state-of-the-art approaches in trajectory generation on actual trajectory data T-Drive, Portotaxi, and Swedishtaxi. The result demonstrates that DTG has a better performance in generating synthetic trajectories under four error metrics.

1. Introduction

With the development of the Internet of Things (IoT), the surge in IoT devices has boosted the growth of trajectory data, and users' trajectory data could be recorded timely and accurately by the advanced positioning technology. The great and accurate trajectory data contribute to the development of location-based services (LBS), where the reports indicate that the market value of location-based services in 2020 is \$44.47 billion and will reach \$155.13 billion by 2026 [1]. Currently, it has become a trend to provide users better services based on the trajectory information from IoT. The increasing location-based services, such as navigation, car-hailing, and living services, provide users with convenient life [2, 3] by mining the trajectory data with intelligence technologies. Furthermore, governments and organizations guide social construction by human mobility data [4, 5]. The trajectory data have important implications for our society.

Protecting users' privacy is the basis of service in the use of trajectory data [6–8]. On the one hand, users' trajectory data may be collected without approval, and the user's track would be recorded by various devices [9]. Even after the user turns off the track recording function, the trajectory information can still be recorded by the operating system or SIM card [10]. The collection and publication of the trajectory data are a direct threat to users' privacy.

On the other hand, if the data owner does not use the appropriate privacy protection mechanism when publishing the trajectory data, the attackers will have the opportunity to obtain the sensitive information of mobile users [11–13]. As shown in Figure 1, IoT converges the real-time trajectory data, collected by various devices, to edge servers and publishes the data to the third-party partners which are trustless or honest but curious. The edge servers generally desensitize the aggregated data before publishing, but the traditional anonymity-based privacy mechanisms are

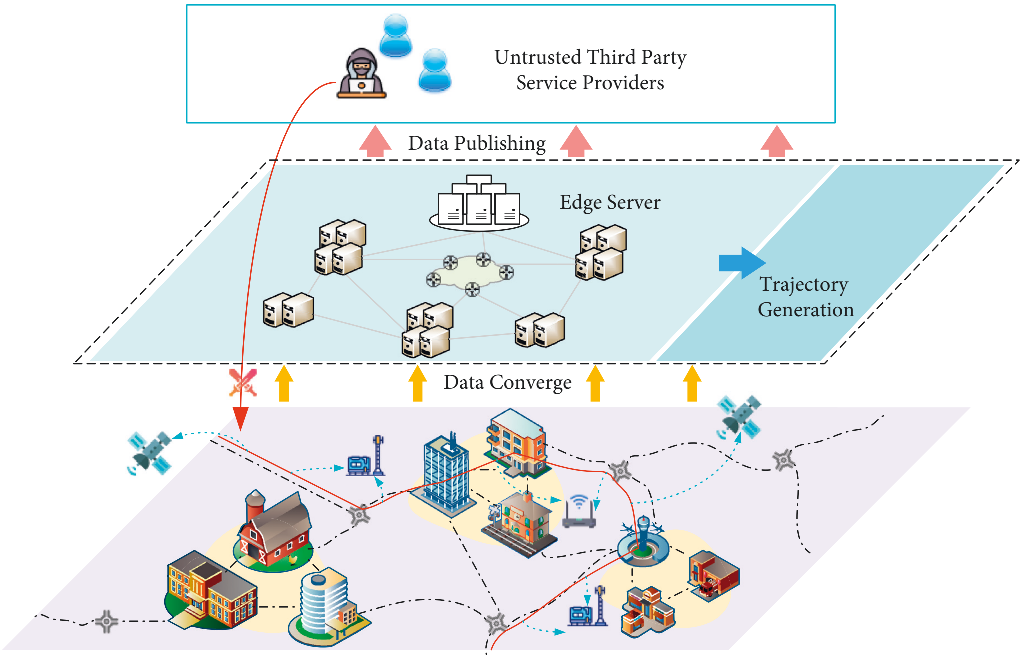


Figure 1: The scene graph of aggregating and desensitizing data in IoT edge computing.

inadequate strength. The malicious third party can infer the data hidden in the anonymous area through the background knowledge of the published data and get the user's sensitive information. The sensitive information involves frequently visited sites such as users' homes, offices, hospitals, clinics, entertainment venues, and religious places [9, 12]. According to users' trajectories, applications could accurately serve ads, such as catering, medical, and other services. And some applications are paid promotion fees, so they consider the interests of the payer more than those of users. These are common violations of users' privacy, and even more than that, criminals may formulate crime plans based on the leaked trajectory information.

These problems have inspired trajectory generation technology. Researchers hope to extract mobility features from the trajectory dataset and construct models to generate synthetic trajectories based on the features. Recent studies have developed differentially private-based [14, 15] trajectory generation [16–21] to provide strong privacy guarantees. The key challenge of developing a differential privacy-based generation model mechanism is to preserve the mobility features accurately when adding noise. There are three main works in the existing model. The first is to divide the geospatial region of the trajectory dataset and transform continuous spatial coordinates into discrete region identifications. Second, it is to build a model that can model long-term spatio-temporal trajectory data. The third is to design the privacy protection mechanism to protect the spatial and temporal privacy of trajectory data. Traditional generations that mostly use the Markov model are hard to accurately extract and preserve the mobility patterns, and the differential privacy mechanism distorts the features of trajectory, such as position distribution, trajectory diameter, and frequent pattern. Researchers turn to the study of generation models based on deep learning [22–24]. The deep learning model can learn the hidden patterns contained in the original dataset. Particularly, a recurrent neural network (RNN) has the advantages of modelling long-term time-dependent sequence data, and it can capture geographic features of trajectory data preeminently. Nevertheless, most of the current research on trajectory generation based on the deep learning model [22, 23, 25–27] ignores the protection of mobility privacy. The deep learning model has mass parameters with a huge semantic space, which may contain the features of the original trajectory data. The attackers could obtain the information of the original data by model inference attack.

To solve the above problems, this paper presents a trajectory generation model with differential privacy and deep learning (DTG). DTG preserves the spatial-temporal correlation of trajectories and provides high-strength privacy protection by combining differential privacy mechanisms and the deep learning model. First, we propose the private hierarchical adaptive grid model and index the coordinates of trajectory data by the identifiers of grid cells as the training data. The model could divide the extremely dense regions of geospatial space hierarchically, and the adaptive mechanism is adopted to flexibly determine the size of the grid according to the density of the local region. The count of each cell of the grid is perturbed by the Laplace mechanism to protect spatial privacy. Second, we extract the features of the trajectories by GRU. A subsequence of a trajectory is generated over a sliding window and the GRU model train according to the sequence taking the first part of a sequence except for the last node as input and the last node as the output. Third, to protect the temporal correlation privacy, we take the optimizer perturbation method in the training process. The optimizer perturbation method is implemented by adding the random noise with the Gaussian distribution to satisfy differential privacy.

The major contributions of this paper are as follows:

(i) For the first time, we introduce the private hierarchical adaptive grid model. It is a density-aware grid model that reduces the number of empty cells and fully divides the dense regions to keep the cells free of excess data.

(ii) We develop a trajectory generation model with a differential privacy mechanism and the GRU model. The GRU model could guarantee good availability of the generated trajectory data, and the differential privacy mechanism could protect the real model parameters from being accessed by attackers.

(iii) Third, we conduct an experiment on DTG over real-life datasets and demonstrate that our solution outperforms the state-of-the-art techniques [6, 20, 22] in terms of the point distribution error metric, frequent pattern error metric, region query error metric, and diameter error metric.

The remainder of this paper is outlined as follows. We first discuss the state-of-art technology of trajectory synthesis in Section 2 and introduce the background and main theorems in Section 3. Section 4 presents the core components of DTG. Section 5 describes the evaluation metrics and the experiment setup to prove the superiority of our method using real datasets. Furthermore, some research emphasis is put forward in Section 6.

2. Related Work

We classify related work into two categories and discuss each category.

2.1. Trajectory Generation with Differential Privacy. Most of the existing trajectory generation schemes are mainly divided into stochastic methods [28–32] and simulation-based methods [33–37]. The simulation-based method generates simulated trajectory data by simulating the human mobility pattern in various road networks. The stochastic modelling method generates random variables following a particular probability distribution to fit the real data. And with the development of differential privacy technology, using the stochastic model with perturbation based on differential privacy to protect the privacy of trajectory datasets [6, 16, 18, 20, 21] has become the mainstream method.

Chen et al. [16] use the variable-length n-gram method to process sequential data, counting the sequential transition probability and generating synthetic data. In the process, they take differential privacy methods to add noise and design an exploration tree to improve data availability. He et al. [18] protect the privacy of the trajectory generation process by adding Laplace noise to the prefix tree. In addition, they propose hierarchical reference systems and direction weighted sampling to improve the availability of generated data. AdaTrace [20] uses the low-order Markov model to generate the synthetic trajectory data. They add Laplace noise to the grid method, length histogram, start and end distribution diagram, Markov model, and so on. TGM [21] models the encoded data as the graphical generative model method, specifies the starting point, calculates the direction of advance, and transfers to the adjacent grid or maintains the current position, thus generating features of arbitrary length and retaining the stop point.

These methods provide strong privacy protection, but the availability of the generated trajectory data is inadequate.

2.2. Trajectory Generation with Deep Learning. Deep learning has brought new developments to trajectory synthesis, having significant advantages in modelling sequential data compared to the Markov model [38].

Kulkarni et al. [22] take four evaluation ways to measure seven kinds of formation models, including Char-RNN [39], RNN-LSTM [40], RHN (recurrent highway networks) [41], PSMM (pointer sentinel mixture model) [42], SGAN [43], RGAN [44], and Copulas [45]. The results show that it is feasible to use RNNs to generate trajectory data. Huang et al. [46] combine variational autoencoder (VAE) based on LSTM to get a well-constructed potential space to capture the salient features of training data, showing excellent performance. TrajGANs [23] framework applies generative adversarial nets (GAN) to trajectory generation, and the generator and discriminator of trajGANs are RNN networks. The generator generates a synthetic trajectory according to a random vector, and the discriminator is used to identify the authenticity of the trajectory. Ouyang et al. [25] propose trajectory generation technology based on GAN, and its generator and discriminator use a convolutional neural network. This method treats the trajectory as a stay sequence, and three features such as geographical location, start time, and duration of the trajectory are extracted for trajectory generation. Song et al. [26] use a four-layer convolutional neural network within the GAN framework to generate trajectories which are represented by a 512×512 matrix. Movesim [27] uses a model-free generative adversarial framework to generate synthetic trajectory data. The generator uses a self-attention-based sequential modelling network to model human mobility, and the discriminator distinguishes the generated trajectory sequence by a mobility regularity-aware loss.

The above methods by deep learning model ignore the privacy protection of the generation model and training data.

It is suggested that we should combine differential privacy and deep learning in trajectory generation for improving the availability and privacy of the generated synthetic trajectory data.

3. Preliminaries

This section introduces the preliminaries of DTG. We review the trajectory dataset and the basic terminology of differential privacy.

3.1. Trajectory Dataset and Notation. Let $T = \langle P_1, P_2, \ldots, P_{|T|} \rangle$ be a trajectory with P_1 as the start point and $P_{|T|}$ as the endpoint of the trajectory. T is a time-ordered sequence composed of $|T|$ sequential points. And a trajectory dataset $D = \{T_1, T_2, \ldots, T_{|D|}\}$ contains $|D|$ trajectories. A spatial-temporal point is a pair $P_i = (l_i, t_i)$, where l_i indicates the geographic location and the timestamp and t_i indicates the time when l_i is visited. l_i is a pair of space coordinates, which is usually represented by longitude and latitude, as $l_i = (\text{longitude}_i, \text{latitude}_i)$.

To simplify the problem of trajectory generation, we preprocess the original dataset before trajectory generation. In the dataset D, the intervals between the adjacent points are the same. Hence, we simplify $P_i = (l_i, t_i)$ to $P_i' = (l_i)$ representing a point after data preprocessing because of the same intervals. The detailed process is in Section 4. And a trajectory is denoted by $T' = \langle P_1' P_2', \ldots, P_{|T|}' \rangle$, and the dataset is denoted by $D' = \{T_1', T_2', \ldots, T_{|D'|}'\}$. We define two-dimensional space $\Omega(D)$ as the geospatial region of D. $\Omega(D)$ is the boundaries of space coordinates of the points in D.

3.2. Grid Method. The grid method could discretize the continuous two-dimensional (latitude and longitude) coordinates. The grid method divides the geospatial region $\Omega(D)$ into multiple disjoint subregions, as shown in Figure 2, and we take the identifiers of subregions to index the points instead of the coordinates of latitude and longitude.

Definition 1 (grid model). Let G be a grid model. For the geospatial region $\Omega(D)$ of the dataset D, G contains n independent regions, represented as $G = \{g_1, g_2, \ldots, g_n\}$, and it satisfies for $\forall i \in n$, $g_i \neq \varnothing$ and for $\forall i, j \in n (i \neq j)$, $g_i \cap g_j = \varnothing$. $g_i \in G$ has a unique identifier C_i.

If a pair of coordinates l of point P is in the region of g_i, we take C_i for P. Therefore, the grid model G could map spatial coordinates to the identifiers of grid units.

3.3. Differential Privacy. The standard privacy definitions used in our paper are derived from the work of Dwork [47]. Differential privacy [15, 23] is a robust database privacy protection standard. Dwork proves that it is impossible to provide absolute privacy protection in the presence of background knowledge [48] and further proposes the concept of differential privacy based on indistinguishability. Differential privacy [47] requires that the output of any computation is insensitive to changes in a single data; namely, the effect of an adversary learning information from a database containing a record is the same as learning information from a database that does not have this information. Therefore, the adversary cannot violate the privacy of any piece of data in the database. Furthermore, it cannot violate the privacy of the whole database.

Definition 2 (neighboring datasets). If datasets, D_1 and D_2, differ in only one record such that $|D_1 \oplus D_2| = 1$, D_1 and D_2 are neighboring datasets.

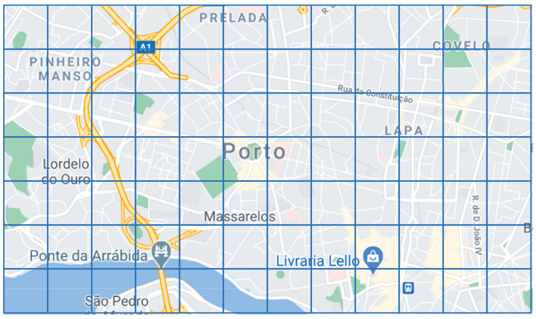


FIGURE 2: The grid method sketch. The diagrammatic drawing of discretizing geospatial space by the grid method. This area is part of Porto. DTG divides this region into multiple geographical units by the grid method. The sample map image is from Google Earth, and the grid size in the figure does not represent the real grid size used in the experiments.

Definition 3 (differential privacy [47]). Let D_1 and D_2 be neighboring datasets. A randomized algorithm $Q\colon D \longrightarrow O$ is $\varepsilon-$ differentially private $(\varepsilon - DP)$ for $\varepsilon > 0$, and $O \subseteq \text{Range}(Q)$ if equation (1) is workable. And Q is $(\varepsilon, \delta)-$differentially private ($(\varepsilon, \delta) - \text{DP}$) for $\varepsilon > 0, \delta \in (0, 1)$, and $O \subseteq \text{Range}(Q)$ if equation (2) is workable.

$$\Pr[Q(D_1) \in O] \leq e^{\varepsilon} \cdot \Pr[Q(D_2) \in O]. \tag{1}$$

$$\Pr[Q(D_1) \in O] \leq e^{\varepsilon} \cdot \Pr[Q(D_2) \in O] + \delta. \tag{2}$$

Definition 4 (sensitivity). Let $q\colon D \longrightarrow \mathbb{R}^d$ be a function on the dataset D, and its output is a fixed dimension vector of d numbers. The sensitivity Δq of q is defined as

$$\Delta q = \max_{D, D'\colon \|D - D'\| = 1} \left\| q(D_1) - q(D_2) \right\|_2. \tag{3}$$

The D_1 and D_2 in (3) are neighboring datasets. $\|q(D_1) - q(D_2)\|_2$ represents the L_2 norm of $q(D_1) - q(D_2)$. The Laplace mechanism uses the L_1 norm, and the Gaussian mechanism uses the L_2 norm.

The most popular $\varepsilon - \text{DP}$ algorithm is the Laplace mechanism [47], and the most popular $(\varepsilon, \delta) - \text{DP}$ algorithm is the Gaussian mechanism [49]. They perturb the returned values by adding random noise according to the sensitivity.

Definition 5 (Laplace mechanism [47]). $\text{Lap}(\lambda)$ denotes a random variable from Laplace distribution with mean 0 and scale parameter λ. For a function $q\colon D \longrightarrow \mathbb{R}^d$ with sensitivity Δq, the randomized function $M(D) = q(D) + \text{Lap}(\lambda)$ satisfies $\varepsilon - \text{DP}$ when $\lambda \geq \Delta q / \varepsilon$.

The Laplace mechanism provides a strict ε-differential privacy, and some work considers the availability of data and does not require excessive privacy. We use the Laplace mechanism in the grid model and use the Gaussian mechanism, which provides relaxed (ε, δ)-differential privacy constraints, in the deep learning model.

Definition 6 (Gaussian mechanism [49]). Let $N(0, \sigma^2)$ denotes a random variable from Gaussian distribution with mean 0 and variance σ^2. For a function $q\colon D \longrightarrow \mathbb{R}^d$ with sensitivity Δq, the randomized function $M(D) = q(D) + N(0, \sigma^2)$ satisfies $(\varepsilon, \delta) - \mathrm{DP}$ when $\sigma > \sqrt{2\ln(1.25/\delta)}\Delta q/\varepsilon$ for $\forall\delta \in (0,1)$.

We will use the serial combination theorem when designing the privacy protection mechanism of trajectory composition.

Theorem 1 (serial composition theorem). *For* $\forall\varepsilon > 0$, $\forall\delta \in (0,1)$, *where* $\varepsilon = \sum\varepsilon_i$, $\delta = \sum\delta_i$, *if each step of the serial mechanism satisfies, the whole process satisfies* $(\sum\varepsilon_i, \sum\delta_i)$*-differential privacy, namely* (ε, δ)*-differential privacy.*

3.4. Problem Statement. A trajectory generation model M is an algorithm able to generate a set of n synthetic trajectories $D = \{T_1, T_2, \ldots, T_n\}$, which describe the movements of the given population. The generated trajectory T_i should be a time-ordered sequence $T_i = \langle P_1, P_2, \ldots, P_{n_i}\rangle$ composed of n_i spatio-temporal points. And the attackers could not violate the trajectory privacy of users. The availability of the synthetic trajectory dataset is evaluated concerning the point distribution error metric, frequent pattern error metric, region query error metric, and diameter error metric.

4. Synthetic Trajectory Generation

Figure 3 illustrates the system architecture of DTG. It mainly includes data preprocessing, GRU model, gradient descent algorithm with differential privacy mechanism, and trajectory generation algorithm.

4.1. Data Preprocessing. Usually, what we get is the raw trajectory dataset $D_{\text{raw}} = \{P_1, P_2, \ldots, P_n\}$, which is composed of spatio-temporal points. We should first transform D_{raw} to the original dataset, denoted by D.

4.1.1. Transformation of the Trajectory Dataset. P_i of $D_{\text{raw}} = \{P_1, P_2, \ldots, P_n\}$ contains geographical location and time, including user identification, transportation, and other information. We divide D_{raw} into subsets, denoted by $D_{\text{raw,id: } i}$, by user identification, and in each subset, the time of P_i is earlier than the time of P_{i+1}. We require that the user's location should be collected at the same sampling rate. Therefore, the sampling interval $\overline{t}$ between adjacent locations in a trajectory is the same. $D_{\text{raw,id: } i}$ is movement records of the user i for a certain period, and we need to divide this data into multiple trajectories. Given the sampling interval $\overline{t}$, if the interval between the adjacent nodes is greater than $\overline{t}$, we divide it into subtrajectories. For P_i and P_{i+1}, if $t_{i+1}(\in P_{i+1}) - t_i(\in P_i) > \overline{t}$, we take P_i as the last point of the previous trajectory and P_{i+1} as the first point of the next trajectory. Then, D_{raw} is transformed to $D' = \{T_1', T_2', \ldots, T_n'\}$ by this way. T_i' denotes a trajectory and consists of the sequence of locations $\langle P_1, P_2, \ldots, P_{n_i}\rangle$ where $P_i = (\text{latitude}_i, \text{longitude}_i)$. The time intervals between adjacent points in each trajectory of D' are the same.

4.1.2. Discretization of the Trajectory Dataset. The discretization process of DTG is as follows: DTG partitions the geographic region by the private hierarchical adaptive grid method. It divides the region according to the given parameters, including the count threshold of position points n_θ, the upper limit of the hierarchy H, the privacy budget $\varepsilon(\sum_{i=1}^{H}\varepsilon_i = \varepsilon)$, and the partition velocity parameters $\beta = \{\beta_i \mid i \in Z, 1 \leq i \leq H\}$. β_i is used to reduce the speed of partition, and its range is $(0,1]$. The maximum hierarchy of the grid model of the target geographic space shall not exceed H, and the partition shall be stopped if the number of location points in the grid unit is lower than n_θ. The partition process of hierarchy $h\,(h \in Z, 1 \leq h \leq H)$ is as follows: firstly, count points n_j in each grid cell, respectively, and each grid cell is divided into $m_j \times m_j$ subgrid cells of equal size, where $m_j = \left\lceil \sqrt{n_j/n_\theta} \times \beta_h + 0.5 \right\rceil$; then, add noise with differential privacy mechanism to the counts and take the noisy counts as the final counts of the grid cells.

Then, DTG has discretized the $\Omega(D)$, and the grid model G is already built. We index the points of D' by the identifiers of the grids. As shown in Figure 4, the curves represent the user's movement trajectory, and the points on the curves are the sampling points. For example, the point P_i of the trajectory $T_1 = \langle P_1, P_2, P_3, P_4, P_5, P_6\rangle$ stands for the location of the trajectory. DTG indexes the locations of the trajectories by the identifications of grid cells. DTG transforms the location sequence T_1 to the cell identification sequence $S_1 = \langle C_1, C_5, C_5, C_{10}, C_{11}, C_{15}\rangle$, where C_i represents the identification of the grid cells. DTG discretizes the trajectory dataset D' to the trajectory sequence dataset $\mathbb{D} = \{S_1, S_2, \ldots, S_n\}$ by mapping spatial coordinates to the gird identifiers.

4.2. GRU Model. RNN is qualified for modelling sequential data. It can transfer the output and state of the current moment to the next moment as input. Therefore, this serial structure can reserve the relationship between each moment. Considering it is difficult for RNN to reserve long-term dependence, and there are problems of gradient disappearance and gradient explosion, researchers further propose many excellent evolution models based on RNN, such as LSTM (long short-term memory) and GRU. These models solve long-term dependence by adding memory units and avoiding gradient explosions by gating units. Compared with LSTM, GRU has fewer parameters and faster training. Hence, GRU is a better choice for our mechanism.

4.2.1. The GRU Unit. The input and output structure of GRU is shown in Figure 5.

There is a current input x^t and a hidden state h^{t-1}, containing the relevant information, passed down from the previous time steps. The GRU model gets the output y^t and the hidden state h^t of the current time step based on x_t and h^{t-1}. And GRU passes the hidden state h^t to the next time step.

The GRU unit structure, shown in Figure 6, consists of two gates, the update gate and the reset gate. These are two vectors that determine what information should be passed to

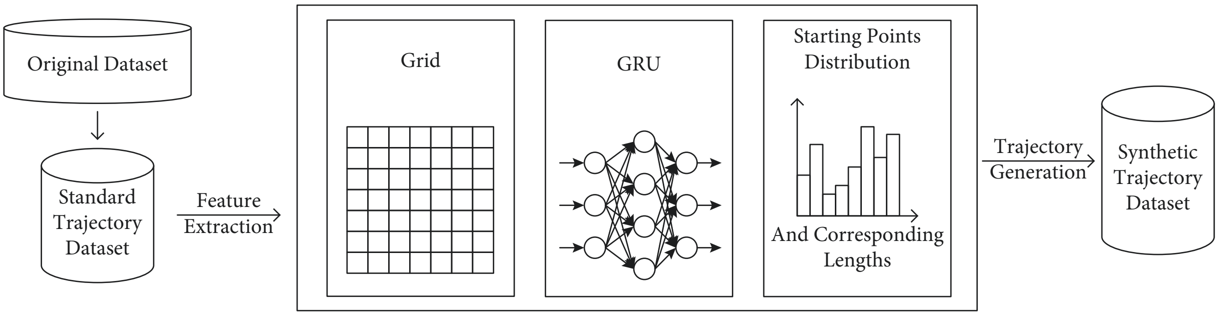

Original Dataset
Standard Trajectory Dataset
Feature Extraction
Grid
GRU
Starting Points Distribution
And Corresponding Lengths
Trajectory Generation
Synthetic Trajectory Dataset

Figure 3: DTG system architecture.

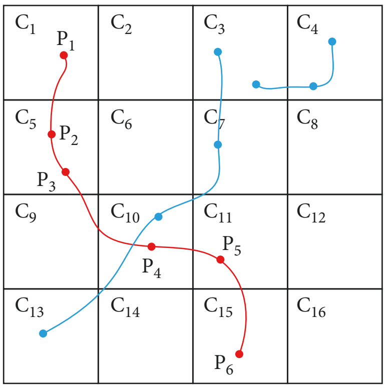


Figure 4: Data discretization by grid method.

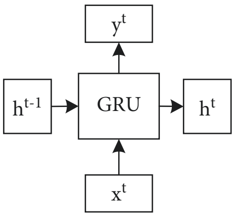


Figure 5: The input/output structure of GRU.

the next steps and finally to the output, and GRU could retain useful information from long ago by them. We will introduce the mathematics of the GRU unit.

The update gate helps the GRU model to determine how much of the past information, and we calculate the update gate z_t at time step t by equation (4). In equation (4), x_t is multiplied by W_z, the weight matrix from the input layer to the update gate, and the hidden state h^{t-1} from $t-1$ time step is multiplied by U_z, the weight matrix from h^{t-1} to the update gate. Both result and the offset vector of the update gate b_z are added together, and we get a result between 0 to 1 by the sigmod function σ.

$$z_t = \sigma\left(W_z x_t + U_z h_{t-1} + b_z\right). \tag{4}$$

The reset gate helps the GRU model to determine how much of the past information to forget. We calculate the reset gate r_t by equation (5). In equation (5), W_r is the weight matrix from the input layer to the reset gate, U_r is the weight matrix from the hidden state to the reset gate, and b_r is the offset vectors of the reset gate.

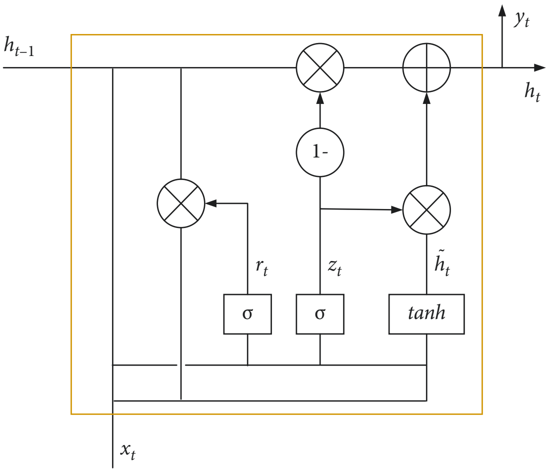


Figure 6: The unit structure of GRU.

$$r_t = \sigma\left(W_r x_t + U_r h_{t-1} + b_r\right). \tag{5}$$

The candidate's hidden state $\tilde{h}_t$ retains the relevant information from the past by the reset gate r_t. And we calculate $\tilde{h}_t$ by (6), where U_h is the connection weight matrix between the hidden states, W_h is the weight moment from the input layer to the hidden state, b_h is the offset vector of the hidden unit, and $\odot$ is denoted as the Hadamard product.

$$\tilde{h}_t = \tanh\left(W_h x_t + U_h\left(r_t \odot h_{t-1}\right) + b_h\right). \tag{6}$$

The current hidden state h_t contains the information of the current unit which also includes the past information. The formula of h_t is as follows:

$$h_t = z_t h_{t-1} + \left(1 - z_t\right) \odot \tilde{h}_t. \tag{7}$$

4.2.2. *The Training Set and the Generation Method.*

DTG generates highly available and realistic synthetic trajectories by the GRU model. GRU model helps model sequential data and could simulate the internal associations of the trajectories. DTG generates trajectories by iteratively predicting the next position. For trajectory $T_u = \langle C_1, C_2, \ldots, C_i \rangle$, the probability that the next position is C_{i+1} is $\Pr[C_{i+1}]$. The context constrains the choice of C_{i+1}, and we introduce the Markov assumption to simplify the problem as

$$\Pr[C_{i+1}] = \Pr[C_{i+1}|C_iC_{i-1}\ldots C_1]. \tag{8}$$

And we use the n-gram model where the current location C_i is determined by the n locations before it, to solve the problem of oversize spatial dimension and data sparsity, as shown in the following equation:

$$\Pr[C_{i+1}] = \Pr[C_{i+1}|C_iC_{i-1}\ldots C_{i-n}]. \tag{9}$$

Based on the n-gram model, if $n = N$, DTG segments the trajectory sequences into subsequences by sliding window with width $N + 1$. For example, if $n = 4$, a trajectory sequence $\langle C_1, C_5, C_5, C_{10}, C_{11}, C_{15}\rangle$ could be divided into 2 subsequences: $\langle C_1, C_5, C_5, C_{10}, C_{11}\rangle$ and $\langle C_5, C_5, C_{10}, C_{11}, C_{15}\rangle$. We take the first n nodes of each subsequence as the input data and the last node as the output data, and then, we get input-output pairs as $(\langle C_1, C_5, C_5, C_{10}\rangle, \langle C_{11}\rangle)$ and $(\langle C_5, C_5, C_{10}, C_{11}\rangle, \langle C_{15}\rangle)$. According to this method, DTG transforms the dataset $\mathbb{D}$ to the training set (X, Y), where $x_i \in X$ is input with length N, and $y_i \in Y$ is output with length 1.

The GRU model is trained on the (X, Y). And, in the process of generation, GRU takes the last N nodes of the given sequence as input, and DTG appends the output from GRU to the end of the given sequence. DTG processes the above process iteratively to generate synthetic trajectories.

4.3. Differentially Private Gradient Descent Algorithm. This section describes the algorithm by which DTG combines the differential privacy mechanism and the gradient descent algorithm. The form of empirical risk minimization (ERM) in machine learning is shown in equation (10). Our target is to make the arg min function satisfy the differential privacy mechanism. Hence, the output model parameter w^* has a similar distribution for any adjacent data D and D'. We could take differential privacy mechanisms at every step, from processing the input data to generating the output data of the machine learning process to protect privacy. We choose to add noise to the process of gradient descent for better availability. This method is an optimization perturbation.

$$w^* = \operatorname{argmin} \frac{1}{n}\sum_{i=1}^{n} \ell(w, x_i, y_i)_w. \tag{10}$$

In Algorithm 1, it shows the process of minimizing the loss function $L(\theta)$ by adjusting the parameter $L(\theta, x_i, y_i)$. During training, DTG calculates the gradient of loss function for each pair of input and output data, then truncates the gradient, and adds noise by the Gaussian mechanism to the truncated gradient. Finally, DTG updates the parameters.

We adopt the $(\varepsilon, \delta) - \mathrm{DP}$ mechanism, which provides looser privacy protection than $\varepsilon - \mathrm{DP}$, improving the availability of the generated data. Algorithm 1 shows the training process of an epoch, and we choose to divide ε and δ equally between the parameters of each epoch, sample, and layer of the model.

DTG sets the threshold as C and truncates the gradients. In $\mathbf{g}_i^{(x_i)}/\max(1, \|\mathbf{g}_i^{(x_i)}\|_2/C)$, if the l_2 norm of $\mathbf{g}_i$ is equal or greater than C, the gradient will take C, and if it is less than C, the gradient will be reserved. The global sensitivity Δq of the function $\mathbf{g}_i^{(x_i)}/\max(1, \|\mathbf{g}_i^{(x_i)}\|_2/C)$ is C.

DTG adopts the Gaussian mechanism to add noise. As required by the Gaussian mechanism, $\forall\delta \in (0, 1)$, $\sigma > \sqrt{2\ln(1.25/\delta)}\Delta q/\varepsilon$, the Gaussian distribution satisfies (ε, δ)-differential privacy. Therefore, the noise added in the gradient provides privacy protection and meets the differential privacy requirements. DTG allocates privacy, ε and δ, to each epoch equally, and the privacy budgets satisfy $\varepsilon = \sum_{n_{\text{epoch}}}\varepsilon_i$ and $\delta = \sum_{n_{\text{epoch}}}\delta_i$. DTG adds Gaussian noise $\mathrm{N}(0, \sigma^2)$, where $\sigma > \sqrt{2\ln(1.25/\delta_i)}\Delta q/\varepsilon_i$, to the parameters of gradient in each epoch. According to the composition theorem of differential privacy, the GRU satisfies the differential privacy.

4.4. Privacy Analysis. If we adopt the differential privacy mechanism at any step in the process of DTG from data input to output and ensure that the process after adopting the differential privacy mechanism does not obtain the data from the step before the differential privacy mechanism, the whole process can be guaranteed to meet the differential privacy in the theory.

We take (ε, δ) as the overall privacy budget and allocate them to each calculation in the optimization process of the neural network, and the privacy budget of each epoch is $(\varepsilon_i, \delta_i)$. In the training process, adding noise consumes the allocated privacy budget $(\varepsilon_i, \delta_i)$, and the overall disturbance consumes the total privacy budget (ε, δ), where $\varepsilon = \sum_{n_{\text{epoch}}}\varepsilon_i$ and $\delta = \sum_{n_{\text{epoch}}}\delta_i$. Because the calculation of each round is based on the same input dataset, hence, this process satisfies the serial combination theorem. Based on the serial combination theorem, the whole process also satisfies (ε, δ)-differential privacy.

The subsequent operation on the processed data does not affect the sequential combination principle of differential privacy.

5. Experiment Evaluation

We evaluate the availability of the generated data in terms of geographic and semantic characteristics and select peer trajectory generation tools as competitors.

5.1. Experiment Setup. We experiment on real datasets T-Drive [50, 51], Portotaxi [3], and Swedishtaxi [52]. The T-Drive trajectory dataset records the trajectories of 10,357 taxis in Beijing, China, during a week. The Portotaxi dataset describes the trajectory of all 442 taxis in Porto, Portugal, during the whole year (January 7, 2013, to June 30, 2014). The Swedishtaxi dataset is the trajectory data for Swedish taxi cars during October and November 2018. This paper samples the T-Drive trajectory data at a sampling interval of 300 seconds. A total of 7,881 trajectories sample data are obtained, including 90590 positions. We randomly selected 3962 trajectories from the Portotaxi dataset with 191868 positions in total and 10,000 trajectories from the Swedishtaxi dataset. They are used as experiment datasets for model training. Considering the average length of the

```
Input: input data $X = x_1, x_2, \ldots, x_N$, output data $Y = y_1, y_2, \ldots, y_N$, the loss function $L(\theta) = 1/N\sum_i L(\theta, x_i, y_i)$, the data size $N$, the learning rate $\eta$, the privacy parameters $(\varepsilon, \delta)$, the gradient norm bound $C$, total epochs $T$, and the current epoch $t$.
    $\varepsilon_i = \varepsilon/T$, $\delta_i = \delta/T$, $\sigma = \sqrt{2\ln(1.25/\delta_i)}C/\varepsilon_i$
for $t$ in $T$ do
    for $(x_i, y_i) \in (X, Y)$ do
        compute gradient $g_t^{(x_i)} \leftarrow \nabla_\theta L(\theta_i, x_i, y_i)$
        clip gradient $\tilde{g}_t^{(x_i)} \leftarrow g_t^{(x_i)}/\max(1, \|g_t^{(x_i)}\|_2/C)$
        add noise $\tilde{g}_j' \leftarrow \sum_i^N g_t^{(x_i)} + \mathrm{N}(0, \sigma^2)$
    $\theta_{t+1} \leftarrow \theta_t - \eta\tilde{g}_t'$
    end for
end for
```

Algorithm 1: Gradient descent with differential privacy for one epoch.

trajectories, the length of the subtrajectories should not be longer than half of the average length, and it should not be overly short so that the model cannot obtain the long-term dependent correlation. Therefore, the length of the subtrajectories in the T-Drive dataset is set to 5, and these of the Portotaxi and Swedish datasets are 20 and 6, respectively.

To verify the effectiveness of the proposed method, we choose to compare it with the current mainstream privacy trajectory generation models DPstar [6], Adatrace [20], and RNN [22]. Due to the small difference in the actual performance of the two methods, they are combined into a differentially private Markov (DPMarkov) model for experiments. In addition, to further demonstrate the performance of the proposed method, this section takes the differentially private RNN model (DPRNN) as the performance baseline. To eliminate the influence of the grid method on the experiment results, three models are trained and generated based on the private trajectory grid method proposed in this paper.

In the experiment, the performance of the mechanisms under different privacy budgets is compared, where the privacy budget ε and δ adopt $[0.01, 0.1, 0.5, 1, 5, 10]$ and $10^{(-5)}$, respectively. Undersized privacy budgets will add massive noise to the gradient, resulting in the model cannot converge, while oversize privacy budgets will increase the possibility of privacy disclosure. Therefore, the privacy budget is between 0.01 and 10 to compare various methods efficiently. The availability of the generated data is quantified by the geographics and semantics similarity with the original data, including the points' distribution error, diameter error, region query error, and frequent patterns error. The experiment is built on the Google Colab platform, configured with Intel (R) Xeon (R) 2.20 GHz processor and 12991 MB memory, and the operating system is Linux version 5.4.144. The simulation experiment is implemented based on PyTorch 1.8.1 and Python 3.7.10.

5.2. Utility Metrics. We generate the synthetic data set $\mathbb{D}'$ according to the original trajectory sequence dataset $\mathbb{D}$, where $|\mathbb{D}| = |\mathbb{D}'|$. Due to the difference between the generated data and the original data, there will be errors in the results when the data user uses the synthetic data compared with the real data. The smaller the error between the results from the generated data and the results from the original data, the higher the availability of the generated data is considered. Since the metrics of data availability are difficult to define, therefore, the similarity between the generated data and the original data is used to approximate the availability of the generated data. We use four evaluation metrics to quantify the similarity of the generated data. In this experiment, we calculated the similarity between the original dataset and the generated trajectory dataset of different methods. The methods are the point distribution error metric, frequent pattern error metric, region query error metric, and diameter error metric.

5.2.1. The Point Distribution Error Metric. The point distribution error metric can directly measure the similarity of the point distribution of two trajectory data sets in the same geographic region. Its disadvantage is that the evaluation results cannot reflect the context between locations in the trajectory. Its advantage is that the metric can reflect the spatial density of locations in the geographic space. The point distribution error is defined as $\mathrm{JSD}(P_p, P_p')$, where $\mathrm{JSD}(\cdot)$ denotes the Jensen-Shannon divergence and P_p, P_p' are the point distributions of $\mathbb{D}$ and $\mathbb{D}'$. JSD measures the similarity of two probability distributions. It is a variant of KL divergence. JSD is symmetric, whose value is from 0 to 1. We calculate the probability distribution of points by the ratio of the number of locations in each region to the total number of locations for each data set and then measure the difference between the distributions by JSD.

5.2.2. The Diameter Error Metric. The diameter of the trajectory is a good evaluation metric with practical value [18] and is critical to the user's range of movement. The diameter of a trajectory is defined as the maximum distance between any two points in the trajectory. We use the histogram of the equiwidth interval to count the distribution of diameters. In this paper, the diameter histogram is set with 50 equal width intervals, and the range of each interval is the extreme difference divided by 50. Then, we use JSD to measure the distance between the real diameter distribution and the synthetic diameter distribution as the diameter error metric.

5.2.3. The Region Query Error Metric. Region query refers to counting the number of trajectories in a query region. Region query error (RQE) measures the relative error between the number of trajectories of the generated dataset passing through a region and that of the real data. The definition of the region query error metric is shown in equation (11). For the query area $\mathbb{A}$, if the position of the trajectory $T'_j \in \mathbb{D}'$ exists within the area, the trajectory T'_j is considered to pass through the area. The function crad$(\cdot)$ is used to count the number of the passed trajectories, and the same statistical methods are used for the real dataset $\mathbb{D}$. This section divides the geospatial region of the dataset into 625 (25×25) query regions and takes the mean of the region query error of all query regions as the region query error of the generated dataset.

$$\mathrm{QRE}(\mathbb{A}) = \frac{\left|\left(\mathrm{crad}\left(T'_j \in \mathbb{D}'\right)\mathrm{if}\ T'_j \mathrm{pass}\ \mathbb{A}\right) - \left(\mathrm{crad}\left(T_i \in \mathbb{D}\right)\mathrm{if}\ T_i \mathrm{pass}\ \mathbb{A}\right)\right|}{\max\left(\left(\mathrm{crad}\left(T_i \in D\right)\mathrm{if}\ T_i \mathrm{pass}\ \mathbb{A}\right), \rho\right)}. \tag{11}$$

5.2.4. Frequent Pattern Error Metric. We take the frequent pattern error metrics to measure the difference between datasets [6] because most services such as urban traffic and advertising require frequent patterns of users. The evaluation method is to select the trajectory segments of top-k in the data set, count the percentage of each trajectory segment in the whole data set, and then compare the differences between the distribution of real data sets and synthetic data sets. We define the trajectory pattern as an ordered sequence of cells; for example, $C_4 \longrightarrow C_5 \longrightarrow C_8$. We count the distributions of $\mathrm{top} - k$ frequent patterns in the trajectory data sets and compare the differences between generated and real data sets. In the following experiment, the sequence lengths of the frequent pattern of the T-Drive, Portotaxi, and Swedishtaxi datasets are 4, 19, and 6, respectively.

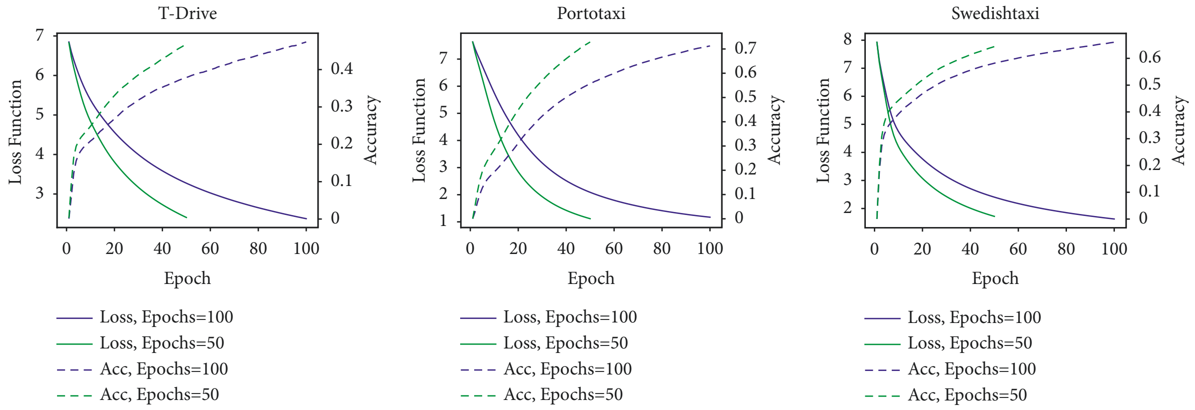
5.3. Comparison with Existing Generators. As shown in Figure 7, we first compare the results of loss function and accuracy of DTG in 50 epochs and 100 epochs when the privacy budget is 10. When DTG does not use the differential privacy mechanism, its accuracy reaches a stable maximum after 50 epochs. Each epoch costs $\varepsilon = 0.1, \delta = 10^{-7}$ in 100 epoch and $\varepsilon = 0.2, \delta = 2 \times 10^{-7}$ in 50 epochs. With the increase of epochs, the number of noises added in each round will also increase. The values of loss function about 50 epochs and 100 epochs are similar in the end. Unilaterally, increasing the number of epochs cannot improve the performance of the model. Therefore, in subsequent experiments, the epochs of DTG and DPRNN are set to 50.

Figure 8 shows the accuracy of different models in predicting the next location. The input of the model is the trajectory sequence, and the output is the next location of the sequence. The length of the input sequence of the T-Drive dataset is 4, the Portotaxi dataset is 19, and the Swedishtaxi is 6. Considering that the 1-order Markov model is better than the high-order Markov model, we adopt the 1-order Markov model in the experiment, and the Markov model predicts the next location based on the last location node of the sequence. As can be seen from Figure 8, with different datasets of T-Drive, Portotaxi, and Swedishtaxi, the increase of privacy budgets has not significantly improved the accuracy of DPMarkov, while the accuracy of DTG and DPRNN has significantly improved. When the privacy budgets are 0.01 and 0.1, the accuracy of DTG and DPRNN is lower than DPMarkov. DTG gradually surpasses DPMarkov when the privacy budget is greater than (equal to) 0.5, and when it is greater than (equal to) 0.1, the accuracy of DPRNN is gradually higher than DPMarkov. In addition, the accuracy of DTG is higher than that of DPRNN under different privacy budgets. It can be concluded that the Markov model has a weak ability to extract the features of trajectory data, but its accuracy will not be significantly reduced when there is more differential privacy noise; DTG and DPRNN have strong modelling ability for trajectory data, but these two models are sensitive to noise. Excessive noise will significantly degrade the performance of the deep learning model, even lower than that of the Markov model. When the privacy overhead is large, DTG performance is much higher than the Markov model.

We compare the results of loss function and accuracy with the number of epochs under different privacy to verify the impact of different noise on the deep learning models. In Figure 9, DTG and DPRNN cannot converge when $\varepsilon = 0.01, 0.1, 0.5$, and the convergence speed is slow when $\varepsilon = 1$. And the models can converge normally when $\varepsilon = 5, 10$. In addition, under different privacy budgets, DTG converges faster than DPRNN. In Figure 10, the models have good accuracy in predicting the next location of the sequence when $\varepsilon = 5, 10$, and DTG is better than DPRNN with the same privacy budgets.

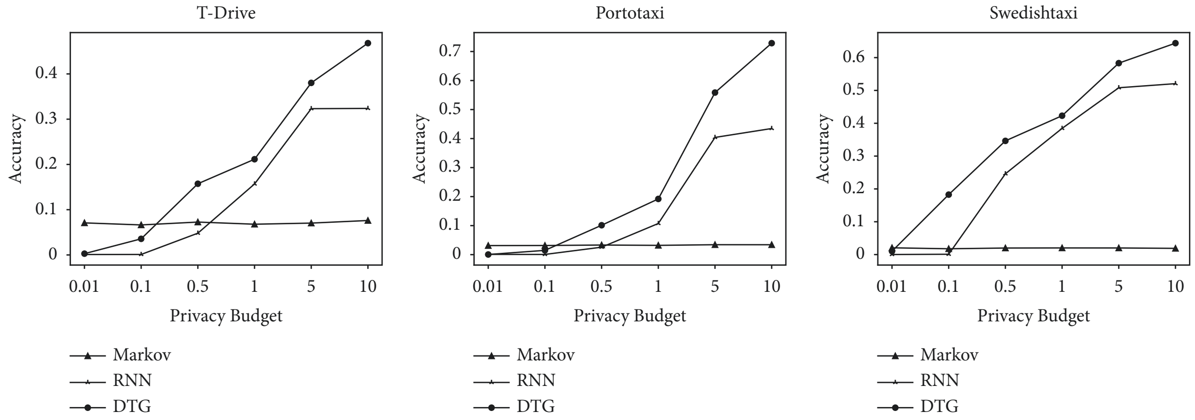
Figures 11–14 show the results of evaluating the similarity between the generated trajectory data and the original trajectory data using four metrics: point distribution error, diameter error, region query error, and frequent pattern error under different privacy budgets. The similarity of data generated by DPMarkov is inferior to that of DTG and DPRNN in four metrics, and there is little difference in the similarity of DPMarkov's generated trajectories under different privacy budgets.

With the increase in privacy budgets, the results of the four metrics of DTG and DPRNN gradually decrease, which indicates that generated trajectory data becomes better. When $\varepsilon = 5, 10$, the similarity of DTG generated data is significantly improved compared with that when $\varepsilon = 0.01, 0.1, 0.5, 1$. Combined with the model performance in Figures 9 and 10, it shows that when ε is small, the error



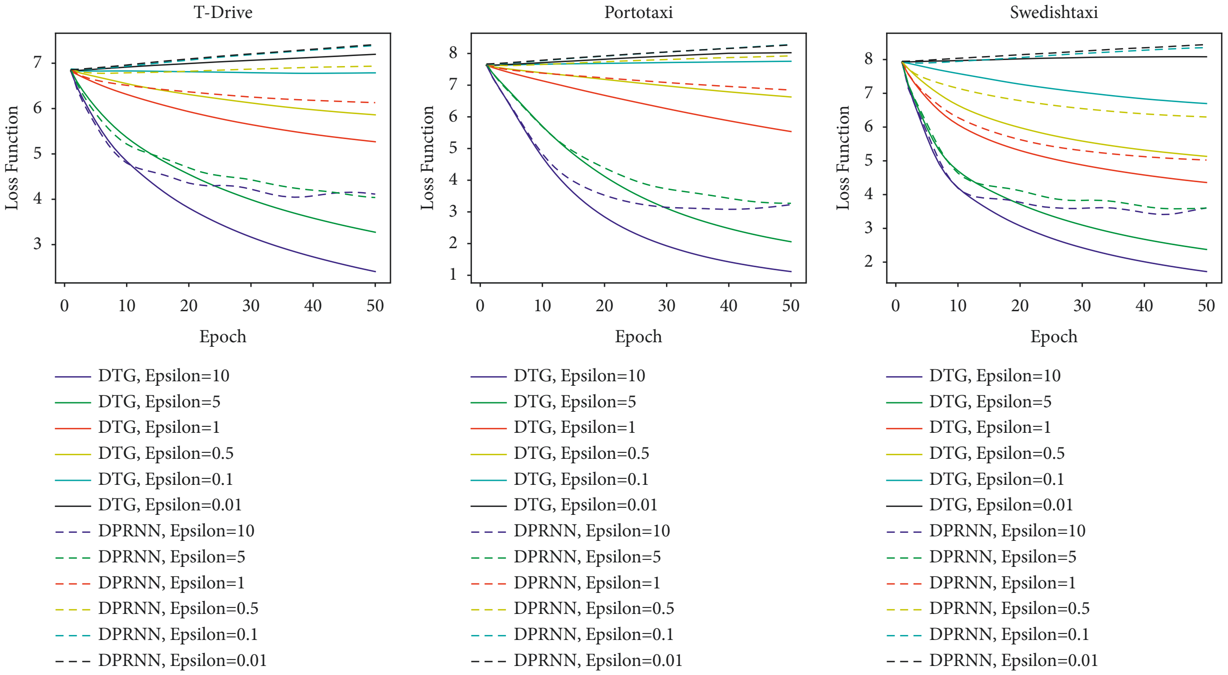
(a) (b) (c)

FIGURE 7: The results of DTG with different epochs ($\varepsilon = 10$). (a) T-Drive. (b) Portotaxi. (c) Swedishtaxi.



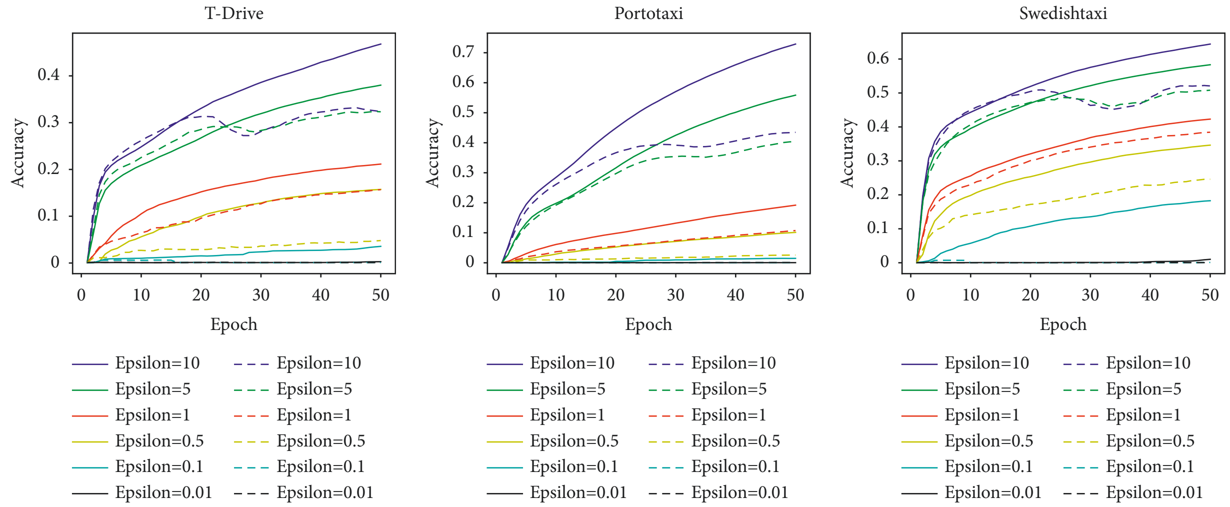
(a) (b) (c)

FIGURE 8: The accuracy of models on different datasets and privacy budgets. (a) T-Drive. (b) Portotaxi. (c) Swedishtaxi.



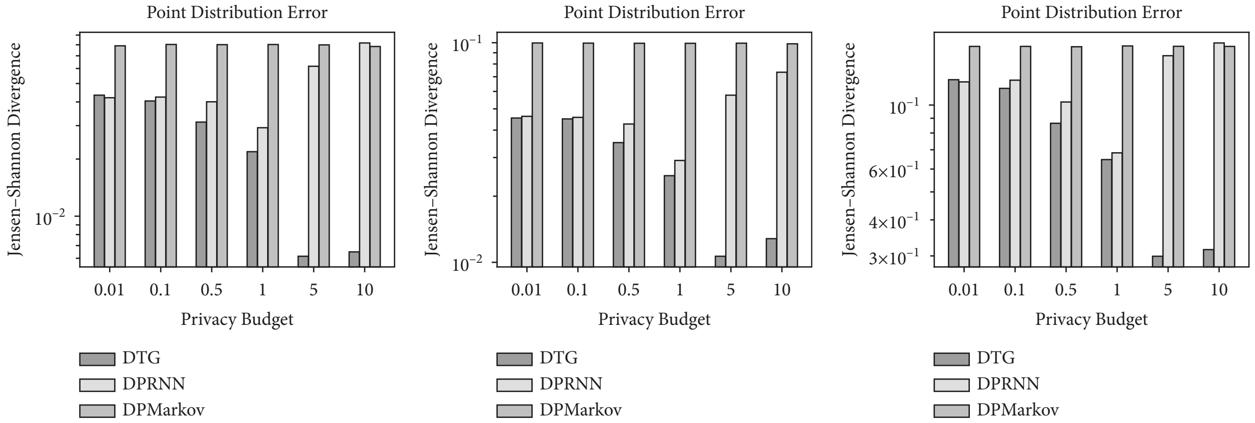
(a) (b) (c)

FIGURE 9: The value of loss function of DTG and DPRNN. (a) T-Drive. (b) Portotaxi. (c) Swedishtaxi.



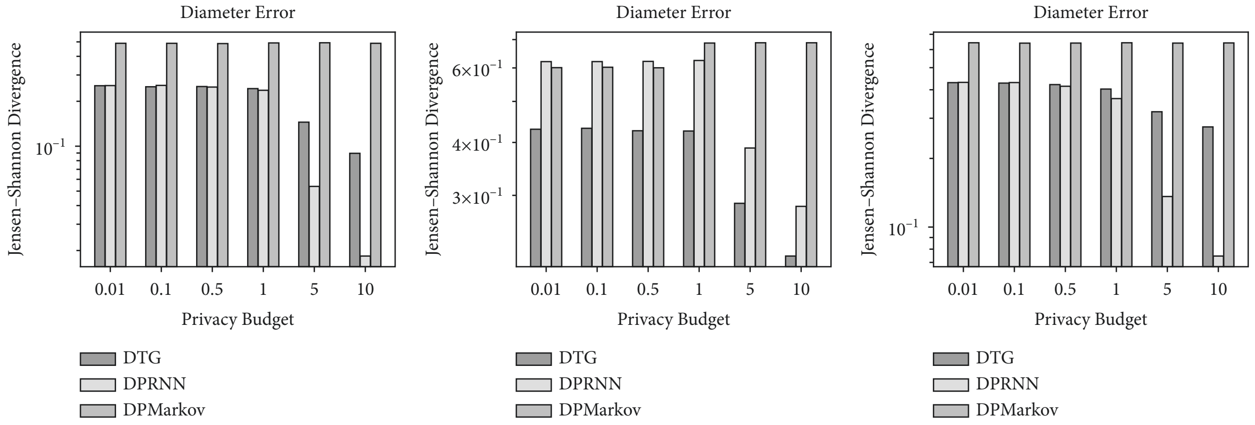
(a) (b) (c)

Figure 10: The accuracy of DTG and DPRNN. (a) T-Drive. (b) Portotaxi. (c) Swedishtaxi.



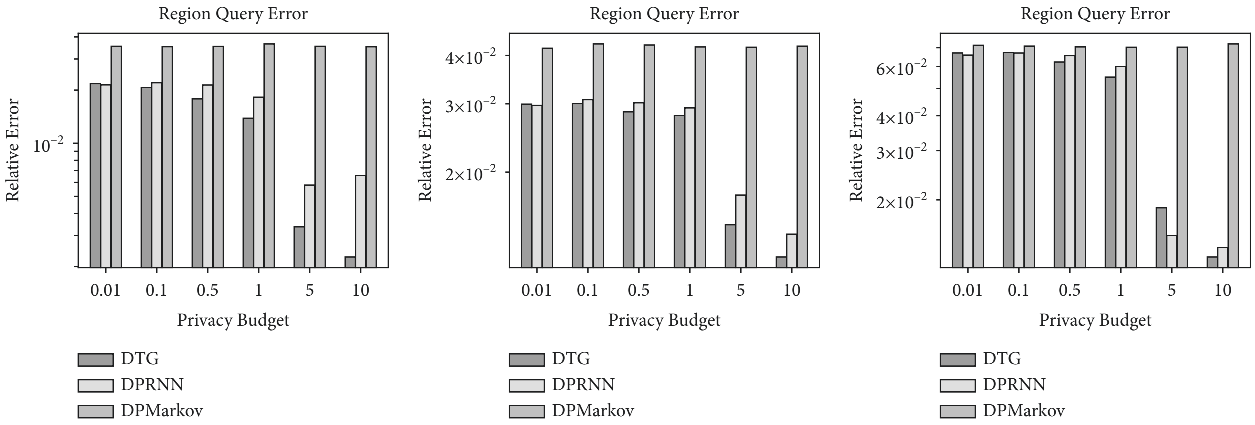
(a) (b) (c)

Figure 11: The point distribution error of models. (a) T-Drive. (b) Portotaxi. (c) Swedishtaxi.



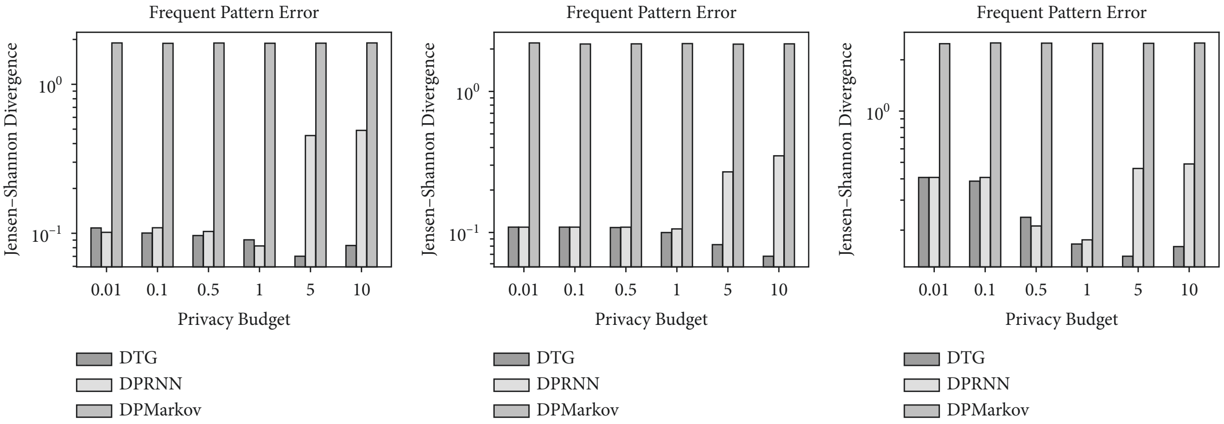
(a) (b) (c)

Figure 12: The diameter error of models. (a) T-Drive. (b) Portotaxi. (c) Swedishtaxi.



(a) (b) (c)

Figure 13: The region query error of models. (a) T-Drive. (b) Portotaxi. (c) Swedishtaxi.



(a) (b) (c)

Figure 14: The frequent pattern error of models. (a) T-Drive. (b) Portotaxi. (c) Swedishtaxi.

between the generated data and the original trajectory is large. In addition, except when $\varepsilon = 5, 10$ in the T-Drive dataset, the data quality generated by DTG is better than that of DPRNN.

In conclusion, by comparing with DPMarkov and DPRNN, experiments show that DTG performs better under the same privacy budgets in most cases. DTG has the advantage of extracting the temporal and spatial relationship of trajectory data.

6. Conclusion

This paper presents DTG. It is a trajectory generation model with high privacy and high availability by combining differential privacy and deep learning. In DTG, we first normalize the original trajectory dataset by sampling with the same interval. Second, we transform the spatial coordinates to grid identifiers by partitioning the geospatial space with the private hierarchical adaptive grid model. This model is density-aware and protects the privacy of the dataset's spatial correlation by the Laplace mechanism. Third, DTG adopts the differentially private gradient descent algorithm to protect the privacy of the dataset's temporal correlation by perturbing gradients. We experimentally compare DTG with the state-of-the-art approaches in trajectory generation taking four metrics, and DTG has a better performance in the process of generating synthetic trajectories.

In the field of trajectory synthesis, we plan to expand in the following aspects in the future: (i) more suitable deep learning models for trajectory generation to better model human mobility patterns, especially in terms of stopping points, etc; (ii) more efficient privacy allocation mechanisms to improve generated data availability; (iii) enhanced grid method to solve the problem of data sparseness.

Data Availability

The experiment data used to support the findings of this study are included in the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (NSFC), Nos. 61941114, 61872836, 62001055, and 61802025.

References

[1] Mordor Intelligence, "Location-based Services Market - Growth, Trends, Covid-19 Impact, and Forecasts," 2020, https://www.mordorintelligence.com/industry-reports/location-based-services-market.

[2] L. C. Nyc Taxi, "Trip record data," 2020, http://www.nyc.gov/html/tlc/html/about/trip record data.shtml.

[3] L. Moreira-Matias, J. Gama, M. Ferreira, J. M. Moreira, and L. Damas, "Predicting taxi–passenger demand using streaming data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1393–1402, 2013, https://doi.org/10.1109/TITS.2013.2262376.

[4] M. Luca, G. Barlacchi, B. Lepri, and L. Pappalardo, "Deep Learning for Human Mobility: A Survey on Data and Models," 2020, https://arxiv.org/abs/2012.02825.

[5] S. M. Nagarajan, P. Chatterjee, W. Alnumay, and V. Muthukumaran, "Integration of IoT based routing process for food supply chain management in sustainable smart cities," *Sustainable Cities and Society (Elsevier*, vol. 76, Article ID 103448, 2022.

[6] M. E. Gursoy, L. Liu, S. Truex, and L. Yu, "Differentially private and utility preserving publication of trajectory data," *IEEE Transactions on Mobile Computing*, vol. 18, no. 10, pp. 2315–2329, 2018.

[7] S. M. Nagarajan, U. Kumaran, M. Thirunavukkarasan, M. D. Alshehri, and S. Alkhalaf, "Secure data transmission in Internet of medical Things using RES-256 algorithm," *IEEE Transactions on Industrial Informatics*, vol. 99, 2021.

[8] V. Muthukumaran, C.-H. Hsu, M. Karuppiah, Y.-C. Chung, and Y.-H. Chen, "Public key encryption with equality test for Industrial Internet of Things system in cloud computing," *Transactions on Emerging Telecommunications Technologies(Wiley)*, vol. 33, no. 4, 2021.

[9] S. Meyer, "Location Privacy Is More Important than You Think," 2018, https://www.cpomagazine.com/data-privacy/location-privacy-is-more-important-than-you-think.

[10] K. Collins, "Google Collects Android Users' Locations Even when Location Services Are Disabled," 2020, https://qz.com/1131515/google-collects-android-users-locations-even-when-location-services-are-disabled/.

[11] F. Xu, Z. Tu, Y. Li, P. Zhang, X. Fu, and D. Jin, "Trajectory recovery from ash: user privacy is not preserved in aggregated mobility data," in *Proceedings of the 26th International Conference on World Wide Web*, pp. 1241–1250, International World Wide Web Conferences Steering Committee, Geneva, Switzerland, April2017.

[12] Y. A. de Montjoye, C. A. Hidalgo, M. Verleysen, and V. D. Blondel, "Unique in the crowd: the privacy bounds of human mobility," *Scientific Reports*, vol. 3, no. 3, p. 1376, 2013.

[13] K. Sashi Kanth, S. Setty, and K. Tainwala, "Scalable and Secure Data Sharing for Dynamic Groups in cloud," in *Proceedings of the IEEE International Conference on Advanced Communication Control and Computing Technologies (ICACCCT)*, pp. 1697–1701, IEEE, Ramanathapuram, India, 8 May 2014, https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7019398.

[14] C. Dwork, "A firm foundation for private data analysis," *Communications of the ACM*, vol. 54, no. 1, pp. 86–95, 2011.

[15] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, pp. 211–407, 2014.

[16] R. Chen, G. Acs, and C. Castelluccia, "Differentially private sequential data publication via variable-length n-grams," in *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, pp. 638–649, ACM, New York, NY, USA, 16 October 2012.

[17] L. Bonomi and Li Xiong, "A two-phase algorithm for mining sequential patterns with differential privacy," in *Proceedings of the 22nd ACM International Conference on Conference on Information Knowledge Management*, pp. 269–278, ACM, Atlanta, GA USA, 27 October 2013.

[18] X. He, G. Cormode, A. Machanavajjhala, C. M. Procopiuc, and D. Srivastava, "Dpt: differentially private trajectory synthesis using hierarchical reference systems," *Proceedings of the VLDB Endowment*, vol. 811, pp. 1154–1165, Seoul, Korea, July2015.

[19] R. Chen, B. C. Fung, B. C. Desai, and N. M. Sossou, "Differentially private transit data publication: a case study on the montreal transportation system," in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 213–221, ACM, New York, NY, USA, 12 August 2012.

[20] M. E. Gursoy, L. Liu, S. Truex, L. Yu, and W. Wei, "Utility-aware synthesis of differentially private and attack-resilient location traces," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 196–211, ACM, New York, NY, USA, 15 Octoer2018.

[21] S. Ghane, L. Kulik, and K. Ramamohanarao, "Tgm: a generative mechanism for publishing trajectories with differential privacy," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 2611–2621, 2019.

[22] V. Kulkarni, N. Tagasovska, T. Vatter, and B. Garbinato, "Generative models for simulating mobility trajectories," 2018, https://arxiv.org/abs/1811.12801.

[23] Xi Liu, H. Chen, and C. Andris, "trajGANs: using generative adversarial networks for geo-privacy protection of trajectory data (Vision paper),"vol. 1, pp. 1–7, in *Proceedings of the Location Privacy and Security Workshop*, vol. 1, pp. 1–7, Tu wien, Vienna, Austria, 19 August 2018.

[24] S. M. Nagarajan, A. K. Bashir, R. P. Mahapatra, S. Mohammed, and Al. Numay, "Iadf-Cps: Intelligent Anomaly Detection Framework towards Cyber Physical Systems," *Computer Communications (Elsevier)*, vol. 188, pp. 81–89, 2022.

[25] K. Ouyang, R. Shokri, D. S. Rosenblum, and W. Yang, "A Non-parametric Generative Model for Human trajectories," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pp. 3812–3817, IEEE, California, CL. USA, 13 July 2018.

[26] H. Y. Song, M. S. Baek, and M. Sung, "Generating human mobility route based on generative adversarial network," in *Proceedings of the Federated Conference on Computer Science and Information Systems (FedCSIS)*, pp. 91–99, Leipzig, Germany, 6 September 2019.

[27] J. Feng, Z. Yang, F. Xu, Y. Haisu, M. Wang, and Y. Li, "Learning to simulate human mobility," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 3426–3433, ACM, New York, NY, USA, 6 August 2020.

[28] V. A. Davies ann, "Evaluating Mobility Models within an Ad Hoc Network," *Mathematical and Computer Sciences*, 2000.

[29] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless Communications and Mobile Computing*, vol. 2, no. 5, pp. 483–502, 2002.

[30] S. M. Mousavi, H. R. Rabiee, M. Moshref, and A. Dabirmoghaddam, "Mobisim: A Framework for Simulation of Mobility Models in mobile Ad-Hoc Networks," in *Proceedings of the Third IEEE International Conference on Wireless and mobile Computing, Networking and Communications (WiMob 2007)*, p. 82, IEEE, Berlin, Heidelberg, 8 October 2007.

[31] J. Harri, F. Filali, and C. Bonnet, "Mobility models for vehicular ad hoc networks: a survey and taxonomy," *IEEE Communications Surveys & Tutorials*, vol. 11, no. 4, pp. 19–41, 2009.

[32] M. Baratchi, N. Meratnia, P. J. Havinga, A. K. Skidmore, and B. A. Toxopeus, "A hierarchical hidden semi-markov model for modeling mobility data," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pp. 401–412, ACM, New York, NY, USA, 13 September 2014.

[33] G. Gidofalvi and T. B. Pedersen, "St–acts: a spatio-temporal activity simulator," in *Proceedings of the 14th Annual ACM International Symposium on Advances in Geographic Information Systems*, pp. 155–162, ACM, Virginia, VA, USA, 10 November 2006.

[34] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz, "Sumo–simulation of urban mobility: an overview," in *Proceedings of the SIMUL 2011, The Third International Conference on Advances in System Simulation*, Barcelona, Spain, 23 October 2011.

[35] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of sumosimulation of urban mobility," *International Journal of Agile Systems and Management*, vol. 5, no. 3&4, 2012.

[36] F. J. Ros, J. A. Martinez, and P. M. Ruiz, "A survey on modeling and simulation of vehicular networks: communications, mobility, and tools," *Computer Communications*, vol. 43, pp. 1–15, 2014.

[37] N. Pelekis, S. Sideridis, P. Tampakis, and Y. Theodoridis, "Hermoupolis: a semantic trajectory generator in the data science era," *SIGSPATIAL Special*, vol. 7, no. 1, pp. 19–26, 2015.

[38] V. Kulkarni and B. Garbinato, "Generating synthetic mobility traffic using rnns," in *Proceedings of the 1st Workshop on Artificial Intelligence and Deep Learning for Geographic Knowledge Discovery*, pp. 1–4, New York, NY, USA, November 2017.

[39] L. R. Medsker and L. C. Jain, "Recurrent neural networks," *Design and Applications*, vol. 5, pp. 64–67, 2001.

[40] A. Graves, "Long short-term memory," *Supervised Sequence Labelling with Recurrent Neural Networks*, pp. 37–45, Springer, Berlin, Germany, 2012.

[41] Z. J. Georg, R. K. Srivastava, J. Koutník, and J. Schmidhuber, "Recurrent highway networks," in *Proceedings of the International Conference on Machine Learning*, jmlr.org, Sydney NSW Australia, 13 August 2017.

[42] S. Merity, X. Caiming, J Bradbury, and R. Socher, "Pointer sentinel mixture models," 2016, https://arxiv.org/abs/1609.07843.

[43] L. Yu, W. Zhang, J. Wang, and Y. Yong, "Seqgan: sequence generative adversarial nets with policy gradient," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. No. 1, California, CL. USA, Febrary 2017.

[44] C. Esteban, S. L. Hyland, and G. Rätsch, "Real-valued (medical) time series generation with recurrent conditional gans," 2017, https://arxiv.org/abs/1706.02633.

[45] G. Geenens, A. Charpentier, and P. Davy, "Probit transformation for nonparametric kernel estimation of the copula density," *Bernoulli*, vol. 23, no. 3, pp. 1848–1873, 2017.

[46] D. Huang, X. Song, Z. Fan, R. Jiang, R. Shibasaki, and Y. Zhang, "A Variational Autoencoder Based Generative Model of Urban Human Mobility," in *Proceedings of the 2nd IEEE International Conference on Multimedia Information Processing and Retrieval (MIPR)*, pp. 425–430, IEEE, San Jose, CA, USA, 28 March2019.

[47] C. Dwork, "Calibrating noise to sensitivity in private data analysis," *Lecture Notes in Computer Science*, vol. 3876, no. 8, pp. 265–284, 2012.

[48] C. Dwork, "Differential privacy," 33rd International Colloquium on Automata," *Languages and Programming*, vol. 4052, pp. 1–12, 2006.

[49] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our data, ourselves: privacy via distributed noise generation," *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, vol. 4004, pp. 486–503, 2006.

[50] J. Yuan, Y. Zheng, X. Xie, and G. Sun, "Driving with knowledge from the physical world," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 316–324, ACM, New York, NY, USA, 21 August 2011.

[51] J. Yuan, Y. Zheng, C. Zhang et al., "Driving directions based on taxi trajectories," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 1, pp. 220–232, 2013.

[52] Kaggle, "Taxi Movement Concatenated," 2016, https://www.kaggle.com/datasets/henrikengdahl/taximovementconcatenated.