Federated TimeGAN for Privacy Preserving Synthetic Trajectory Generation

Saloua Bouabba, Karine Zeitouni, Bassem Haidar, Nazim Agoulmine, Zaineb Chelly Dagdia

Federated TimeGAN for Privacy Preserving Synthetic Trajectory Generation

Saloua Bouabba
Université Paris-Saclay, UVSQ, DAVID, France
ESIEA, CNS, France
saloua.bouabba@uvsq.fr

Karine Zeitouni
Université Paris-Saclay, UVSQ, DAVID, France
karine.zeitouni@uvsq.fr

Bassem Haidar
ESIEA, CNS, France
bassem.haidar@esiea.fr

Nazim Agoulmine
Université Paris-Saclay, Univ Evry, IBISC, France
nazim.agoulmine@univ-evry.fr

Zaineb Chelly Dagdia
Université Paris-Saclay, UVSQ, DAVID, France
zaineb.chelly-dagdia@uvsq.fr

***Abstract*—Mobility datasets are crucial for various applications. However, sharing this data raises privacy concerns due to the sensitive nature of geolocation information. Synthetic data generation has recently emerged as a promising solution to protect geo-privacy of trajectory data. Current approaches rely on having a large set of authentic trajectories collected from individual users to train generative networks. However, this assumption proves impractical in many real-world scenarios due to the sensitive personal information typically embedded within trajectories. Our approach leverages federated learning to generate privacy-preserving synthetic trajectories without the need for centralized data collection. Experimental results demonstrate that our distributed framework effectively produces synthetic trajectories with distributions comparable to baseline, offering a privacy-conscious alternative for geo-privacy protection in mobility datasets.**

I. Introduction

The widespread of location-based technology, such as mobile devices and various connected sensors has resulted in the collection of vast amount of data, including location information and user trajectories, which reflect individuals' mobility patterns. Examining and extracting insights from mobility data holds the potential to enhance our understanding of traffic and public transportation systems and address urban planning challenges more effectively. Furthermore, trajectory data is of paramount interest for business purposes, public health and environmental studies. While mobility data is highly valuable, acquiring and sharing extensive real-world trajectories raises serious privacy concerns as it can reveal highly sensitive information about individual's habit and surrounding context; in fact even when personal identifiers such as user IDs are removed to anonymize trajectory data, users can often be re-identified with very little location information [1].

Data sanitization techniques have been proposed to tackle the problem of privacy preserving trajectory publishing, leveraging the principle of k-anonymity [2], or differential privacy [3]. Nevertheless, attackers can use geographical information or background knowledge to reconstruct user trajectories [4]. Synthetic data generation has recently emerged as a promising solution to protect the geo-privacy of trajectory data [5]. By replacing actual data with synthetically generated trajectories, it is possible to achieve performance that is comparable to real data for various tasks. Synthetic records can also maintain statistical properties, making them a useful tool for a variety of applications. Although the generated data provides a good balance between privacy and utility, these methods assume the availability of a significant number of authentic trajectories collected from individual users, enabling the training of generative networks to produce synthetic trajectories based on these real datasets. This assumption proves impractical in many real-world scenarios. There are a number of factors that contribute to this issue. For one, privacy concerns are increasingly prevalent; people are becoming more aware of the value of their personal information and are therefore more cautious about who has access to it. This wariness extends to the sharing of location data, which could potentially be misused by a centralized entity. In addition to privacy concerns, there are also logistical constraints that make the collection of trajectory data impractical. Gathering this type of data requires coordination across different cities and regions, a task that is not easily achievable. Furthermore, data regulations can often pose a barrier to the centralized collection of this type of data. These regulations, which are designed to protect user privacy and ensure ethical data usage, can limit the amount of data that can be collected and stored by a single entity. As a result, the assumption that a large quantity of authentic trajectory data will be readily available may not hold true in many cases.

An alternative and highly practical approach involves harnessing the distributed nature of trajectories to generate synthetic trajectories in a privacy-preserving manner while maintaining users' data locally by exploiting federated learning, a technique that enables collaborative model training across decentralized devices. To the best of our knowledge,

This work is supported by the "Investissements d'Avenir", programme IDEX under grant ANR-11-IDEX-0003.

this has not been explored in the existing literature. In this paper, we adapt TimeGAN (a model designed for multivariate time series) to a federated setting, and use it for the generation of synthetic trajectories. Our experimental results demonstrate that the performance of our federated TimeGAN is comparable to the baseline, establishing a novel and effective method for privacy-preserving trajectory synthesis within a federated learning framework.

After an overview of the related work in Section 2, we present our proposal in Section 3. We provide the experiments settings in Section 4, followed by the results and a discussion. Section 5 concludes the paper and envisions future directions.

II. Related Work

A. Synthetic Trajectory Generation

Previous studies that tackled the challenge of generating synthetic trajectories mainly relied on sequential models under simplified human mobility assumptions. These models encompass Markov chains [6], [7] and Recurrent Neural Networks [8], [9]. While they prove effective in capturing valuable movement patterns within smaller sub-trajectories, their accuracy tends to diminish when applied as generative models for longer trajectories.

Motivated by the success of deep learning models, particularly generative adversarial networks (GANs) [10] in creating synthetic data across various applications, numerous researchers have sought to harness these methods for the artificial generation of trajectory data. In fact, GANs can simultaneously capture various mobility patterns, spanning spatial, temporal, and social dimensions of mobility data that traditional approaches often fail to grasp. [11] discretize trajectories onto an N1 $\times$ N2 grid, with each visit mapped to a specific cell. They use a Convolutional Neural Network (CNN)-based Wasserstein GAN [12] to generate synthetic trajectories in matrix form, preserving real mobility data's semantic and geographic patterns. [13] proposed Movesim, a framework incorporating a self-attention sequential modeling network to simulate human mobility data. This generator captures temporal transitions in mobility data and utilizes an attention-based network to model relations between locations. A mobility regularity-aware loss for the discriminator ensures the quality of generated trajectories. [14] proposed a Two-Stage GAN (TSG) that first converts GPS points into a discrete grid representation. This initial GAN captures spatio-temporal patterns, and in the second stage, precise coordinates of trajectory points are generated. The model leverages road information from map images and employs Long Short-Term Memory (LSTM) networks to generate realistic trajectories that align with road networks. DeltaGAN [15] takes a two-step generative approach, using a Wasserstein GAN-GP to generate stay durations. By interpreting trajectories as sequences of events occurring at irregular time intervals, DeltaGAN produces detailed and continuous trajectories. The generated timestamps are then used to train a location generator, capturing the realistic range of mobility for location modeling. [16] introduce TrajGAIL, focusing on privacy concerns by mapping coordinates directly to a road network. Unlike previous models, TrajGAIL directly maps coordinates to a road network instead of grid cells, allowing it to generate synthetic data aligned with the road network layout. These methods rely on having a significant number of real trajectories from individual users for training generative networks. However, obtaining such data is impractical due to privacy concerns, as users are hesitant to share their trajectory information without a trusted relationship. Contrary to these traditional approaches, this paper delves into the distributed nature of trajectories, leveraging it to craft privacy-preserving synthetic trajectories via federated learning within the realm of generative adversarial networks. This methodology, to the best of our knowledge, has not been previously investigated.

B. Federated Learning

In the traditional machine learning paradigm, a central server is in charge of collecting data from users in order to train the machine learning model, thus giving rise to many privacy challenges. To tackle this issue, Federated Learning [17] has emerged as an alternative distributed machine learning approach that enables model learning on a large corpus of decentralized data. In a federated learning setting, a model is collaboratively trained between users by updating a global model with locally trained parameters. In fact, raw data is never shared with another entity, only updated parameters are reported to a central server for aggregation. Federated Learning is considered as an iterative process where, at each cycle, the core machine learning model is improved. The implementation of federated learning as described by [18] can be generalized into three phases. First, according to its configuration, the server selects a subset of the different devices announcing their availability to start the cycle. Second, the server is configured according to an aggregation mechanism and sends the global model to each of the selected devices and finally, the devices train the global model using their local data through several iterations and then send their updates to the server which incorporates them into its global state as they occur. If enough devices report updates in time, the cycle is successfully completed and the server updates its global model, otherwise the cycle is aborted.

III. Proposed Framework

A. Problem Definition

Raw trajectory: a trajectory is represented as a sequence of points, denoted as $\hat{S} = [\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n]$. Each point, $\hat{x}_i$, within this trajectory is described by a triple (l_i, y_i, t_i), where l_i and y_i represent the spatial coordinates of the object at a given time instant t_i.

Multivariate time series: a multivariate time series is a sequence of observations or data points collected over time, where each observation consists of multiple variables or components. It can be represented as a collection of vectors, each of which contains values for two or more variables at a specific time point. Suppose we have a time series with observations at

discrete time points $t = 1, 2, 3, \ldots, T$. For each time point t, we have a vector of observations $X_t = (X_{1,t}, X_{2,t}, \ldots, X_{p,t})$, where p is the number of variables or components in the time series. The multivariate time series can then be represented as:

$$\{X_1, X_2, X_3, \ldots, X_T\}$$

X_t represents the vector of observations at time t, and p is the number of variables in the multivariate time series.

Synthetic Trajectory Generation: The objective is to generate a realistic mobility trajectory $\hat{S} = [\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n]$ using a θ-parameterized generative model $\mathcal{G}$. The formulation involves expressing the probability of the entire trajectory $\hat{S}$ as the product of conditional probabilities for each spatial-temporal point $\hat{x}$, as represented by the equation:

$$p_\theta(\hat{S}) = \prod_{i=1}^{n} p_\theta(\hat{x}_i | \hat{x}_1, \ldots, \hat{x}_{i-1}) \tag{1}$$

Here, p_θ denotes the generation distribution from the generator $\mathcal{G}$, and the trajectory generation process is conceptualized as a sequential decision process.

B. Federated TimeGAN for Synthetic Trajectory Generation

The existing GAN-based methods for generating synthetic trajectories exhibit similarities despite variations in details, such as the choice of frameworks, training datasets, and encoding methods. The common approach involves encoding trajectory datasets for input into a generative network, which, using an adversarial model, undergoes training with the prepared datasets to generate synthetic trajectories. It is worth noting that our solution's trajectory generation framework is based on a similar approach. However, unlike existing methods, our approach does not involve collecting the data beforehand. For our chosen GAN model, we have opted for TimeGAN [19], which is specifically designed for generating multivariate time series. Given that a trajectory can be considered as a multivariate time series with latitude and longitude as spatial dimensions, TimeGAN is well-suited for our purpose.

TimeGAN employs the autoregressive decomposition of the joint distribution to focus on the conditionals, simplifying the learning objective to best approximate the density at any time. This breaks down the sequence-level goal into stepwise objectives. The first is global, minimizing the distance between distributions. The second is local, minimizing the sum of distances for any time. While minimizing the former requires a perfect adversary, minimizing the latter only needs ground-truth trajectories, leading to a training procedure that includes a supervised loss for adversarial learning.

TimeGAN utilizes the autoregressive decomposition of the joint distribution: $p_\theta(\hat{S}) = \prod_{i=1}^{n} p_\theta(\hat{x}_i | \hat{x}_1, \ldots, \hat{x}_{i-1})$ (1) to focus specifically on the conditionals. This leads to the complementary—and simpler—objective of learning a density $p(\hat{S}_{1:t} | \hat{S}_{1:t-1})$ that best approximates $p_\theta(\hat{S}_{1:t} | \hat{S}_{1:t-1})$ at any time t. This breaks down the sequence-level goal into stepwise objectives. The first is global, minimizing the distance between distributions. The second is local, minimizing the sum of distances for any time. While minimizing the former requires a perfect adversary, minimizing the latter only needs ground-truth trajectories, leading to a training procedure that includes a supervised loss for adversarial learning.

TimeGAN leverages the strengths of both Generative Adversarial Networks (GANs) and autoencoders to achieve this. In TimeGAN, the GAN component allows the generator to create synthetic data that matches the real data's feature distribution, thanks to the guidance of the discriminator, minimizing an unsupervised loss:

$$L_u = \mathbb{E}_{x_{1:T} \sim p} \left[\sum_t \log y_t \right] + \mathbb{E}_{x_{1:T} \sim \hat{p}} \left[\sum_t \log(1 - \hat{y}_t) \right] \tag{2}$$

where y_t and $\hat{y}_t$ denote the classification of real or fake data, respectively. This aspect ensures that the generated data captures essential statistical characteristics.

Additionally, TimeGAN incorporates an autoencoder (an embedding and recovery network) which reduces the dimensionality of the data to a hidden space and then reconstructs it using these latent features. This dimensionality reduction allows the GAN to focus on learning the temporal dynamics within a smaller-dimensional space through minimizing a reconstruction loss between x_t and $\tilde{x}_t$ real and reconstructed features:

$$L_r = \mathbb{E}_{x_{1:T} \sim p} \left[\sum_t \| x_t - \tilde{x}_t \|_2^2 \right] \tag{3}$$

The generator receives the actual data's embedding sequences $h_{1:t-1}$ and generates the next latent vector, allowing it to learn the transition dynamics between different time points. Gradients are computed based on a supervised loss that measures the difference between the real conditional distributions and the model's distributions in the latent space:

$$L_s = \mathbb{E}_{x_{1:T} \sim p} \left[\sum_t \| h_t - g(h_{t-1}, z_t) \|_2^2 \right] \tag{4}$$

This approach ensures that the generator captures the stepwise conditional distributions and temporal dependencies of the trajectory data.

Our approach involves integrating TimeGAN into a federated learning framework as showed in Algorithm 10. First the server broadcast the initial parameters of TimeGAN to the participating clients. Each client independently train TimeGAN on their local datasets. This step allows each client to capture the unique temporal patterns present in its specific geographical region or dataset characteristics. The training involves iteratively updating the model parameters to enhance its ability to generate realistic synthetic trajectories. After a set number of training iterations, each client shares its locally updated TimeGAN model parameters with the central server. This periodic sharing ensures that the server receives continuous insights into the evolving trajectory patterns learned by each client. The server aggregates the received model updates from all participating clients. This aggregation process involves calculating a weighted average of the model parameters, giving

each client's contribution a proportional influence. The server communicates the aggregated model updates back to each client. This iterative process of sharing, aggregating, and propagating updates continues for a predefined number of rounds.

At the end of the federated learning process, the server utilizes the final aggregated model parameters to generate synthetic trajectories. This model, enriched by the collaborative insights from all clients, produces trajectories that encapsulate the diverse spatio-temporal patterns learned across the distributed datasets.

Algorithm 1 Federated TimeGAN for Synthetic Trajectory Generation

Require: Set training period N, local training iterations L

1: Initialize local TimeGAN parameters for each client i: $\theta_G^0, \theta_D^0, \theta_E^0, \theta_R^0$

2: Set learning rates $a(n), b(n), c(n), d(n)$ for discriminator, generator, embedder, and recovery for $n = 1, 2, \ldots, N-1$

3: **for** $n = 1, 2, \ldots, N-1$ **do**

4: **for** $l = 1$ to L **do**

5: Update local TimeGAN parameters:

$$\theta_G^n = \theta_G^{n-1} + a(n-1)gi(\theta_G^{n-1})$$
$$\theta_D^n = \theta_D^{n-1} + b(n-1)di(\theta_D^{n-1})$$
$$\theta_E^n = \theta_E^{n-1} + c(n-1)ei(\theta_E^{n-1})$$
$$\theta_R^n = \theta_R^{n-1} + d(n-1)ri(\theta_R^{n-1})$$

6: **end for**

7: All clients send TimeGAN parameters to the server.

8: Calculate aggregated parameters by averaging:

$$\theta_G^n\Delta = \sum_{j=1}^{B} p_j \theta_{Gj}^n \qquad \theta_D^n\Delta = \sum_{j=1}^{B} p_j \theta_{Dj}^n$$
$$\theta_E^n\Delta = \sum_{j=1}^{B} p_j \theta_{Ej}^n \qquad \theta_R^n\Delta = \sum_{j=1}^{B} p_j \theta_{Rj}^n$$

9: The server sends back aggregated parameters to all clients

10: **end for**

- The server uses the final aggregated TimeGAN parameters to generate synthetic trajectories.

IV. Experiments

A. Data

In order to experiment federated TimeGAN for synthetic trajectory generation we use GeoLife [20]; a publicly available dataset describing the GPS trajectories of 182 users over 4.5 years, collected using different devices (e.g., GPS receivers and mobile phones) every 1–5 seconds or 5–10 meters. GeoLife covers many users' outdoor movements, from life routines such as going home and work to entertainment and sports activities. Each point in a GeoLife's trajectory contains latitude, longitude, altitude, and timestamp.

We filter out the region of interest within Beijing, China area. We select trajectories that last less than 1 hour, which constitutes 58% of the trajectories. We use a three-minute time interval, resulting in 20 points for each trajectory ensuring uniform length across all trajectories without the need for padding. We note that our attempts to utilize the entire dataset and standardize trajectory length through padding resulted in significant performance degradation.

We note that for the federated part, "clients" refers to the original GeoLife users. After preprocessing the trajectories as described above, only a subset of these users remains, specifically 50 clients. Each client has a different number of trajectories, with each trajectory consisting of 20 points.

B. Metrics

In accordance with standard practice in previous work [11]; [13], we compute the Jensen–Shannon divergence (JSD) between two probability distributions P and Q using the formula:

$$JSD(P||Q) = \frac{1}{2}D_{KL}(P||M) + \frac{1}{2}D_{KL}(Q||M) \tag{5}$$

where $D_{KL}(R||S)$ is the Kullback–Leibler divergence, defined as:

$$D_{KL}(P||Q) = \sum_{x \in X} P(x) \log\left(\frac{P(x)}{Q(x)}\right) \tag{6}$$

and $M = \frac{1}{2}(P||Q)$ represents the midpoint distribution.

The JSD serves as a symmetric measure to evaluate the similarity between the distributions of the following mobility metrics in the generated trajectory data and real trajectory data:

- Distance: the daily average per trajectory travel distance
- Radius of gyration per trajectory
- Duration: the length of time spent at each location
- DailyLoc: the total number of distinct places within a single trajectory
- G-rank: Reflects the frequency of visits to specific locations, specifically the visiting frequency of the top 100 locations
- I-rank: An individualized version of G-rank, considering the ranking of locations for each trajectory

We discretize GPS coordinates into a grid of 250m x 250m cells. This involves converting the continuous geographical area into discrete units or cells, with each point of a trajectory falling into one of these cells.

C. Baseline

We compare federated TimeGAN with:

- MoveSim: The original work [13] integrates three distinct urban structures into its methodology. These structures encompass the physical distance between all pairs of locations, functional similarity determined by the correlation between Points of Interest (POI) distributions, and historical transitions between locations. The computation of the first and third structures directly utilizes

training trajectories. However, the second structure relies on additional POI information, which is not available in the dataset. To ensure a fair and comparable evaluation, MoveSim is implemented without utilizing this second urban structure as done in [21].

- Decentralized TimeGAN: Through our experiments, we observed that TimeGAN is designed for processing chronologically ordered time series data. Therefore, it cannot be efficiently trained on clients with overlapping timestamps in a centralized fashion. This limitation necessitates the decentralization of TimeGAN to enable training on separate clients in parallel. So we compare the federated TimeGAN approach where clients collaboratively update a global model to an approach where every client apply TimeGAN on its own dataset to generate synthetic data in local and then we regroup all the synthetic data of all clients (there is no parameter sharing during the training unlike the federated setting).

D. Default Setting

All of the components (embedding network, generator, discriminator and recovery) are implemented with 3-layer Long short-term memory (LSTM) with hidden dimension of 24. In our experiments, we have 50 clients, each with different number of trajectories (non-iid setting). Using random search we set the number of the federated learning rounds to 100. The number of epochs in each client (i.e., local epochs) is set to 5000 per round. We select 10 clients to participate at each round. All experiments were conducted on a machine with NVIDIA A10 graphic processing unit (GPU), 128 GB of RAM, 32 CPU cores and 400GB of storage.

V. Results and Discussion

A. Mobilty Metrics

We report the results (i.e., the JSD value for the mobility metrics) in Table I. Both decentralized TimeGAN and federated TimeGAN demonstrate superior performance compared to MoveSim across multiple metrics. In our evaluation, Decentralized TimeGAN outperforms MoveSim by a significant margin, achieving approximately 80% better performance on average, while Federated TimeGAN shows around 50 % improvement compared to MoveSim.

This notable performance gap could be attributed to various factors. Firstly, MoveSim's reported gains in the original work might have heavily relied on leveraging auxiliary Points of Interest (POI) information, which was intentionally omitted in our study to ensure a fair comparison. Additionally, the more complex model architecture of MoveSim may require a larger training dataset. In our experiments, we utilized a dataset with only 50 clients, whereas the original MoveSim work used 181 clients.

Moreover, Decentralized TimeGAN exhibits superior performance compared to federated TimeGAN. Decentralized TimeGAN showcases approximately 25% better performance on average compared to Federated TimeGAN. This observation can be attributed to several factors. Decentralized TimeGAN avoids the limitations associated with regrouping data from all clients by individually training on each client's data. In contrast, federated TimeGAN shares model parameters among clients during each round, potentially leading to model divergence. Additionally, federated TimeGAN's performance is sensitive to hyperparameter tuning, such as the number of local and global rounds, which was done through a random search due to resource constraints. Furthermore, the non-IID nature of client settings contributes to the performance gap, as each client exhibits distinct mobility patterns and trajectory distributions, posing challenges in achieving a homogeneous learning environment.

B. Visualization

We use QGIS to plot the processed real Geolife data points and the synthetic data from the decentralized TimeGAN, MoveSim, and federated TimeGAN. This illustrates the comparison between these methods in terms of preserving the patterns in the original trajectories.

Movesim's data distribution (Fig. 1c) closely resembles the real Geolife dataset, providing an accurate representation due to its generation of exact coordinates, particularly utilizing 70% of the original Geolife data for training. However, the privacy risk inherent in synthetic generated data is heightened in Movesim, given that it retains the exact GPS points from the training data. While the order and duplication of points may differ in the generated trajectories, the presence of identical distinct points poses an increased privacy risk.

Despite its visually accurate representation, Movesim's metrics does not surpass those of decentralized TimeGAN. This discrepancy arises from the fact that these metrics are trajectory-based, and even if the distinct points match, the generated trajectories differ.

In contrast, Federated TimeGAN and Decentralized TimeGAN create new trajectories with novel points, resulting in visualizations that deviate from the original data. Fig. 1d suggests that certain clients have a more pronounced influence on the training process, possibly attributed to the non-iid setting with distinct distributions of trajectories and mobility patterns across clients.

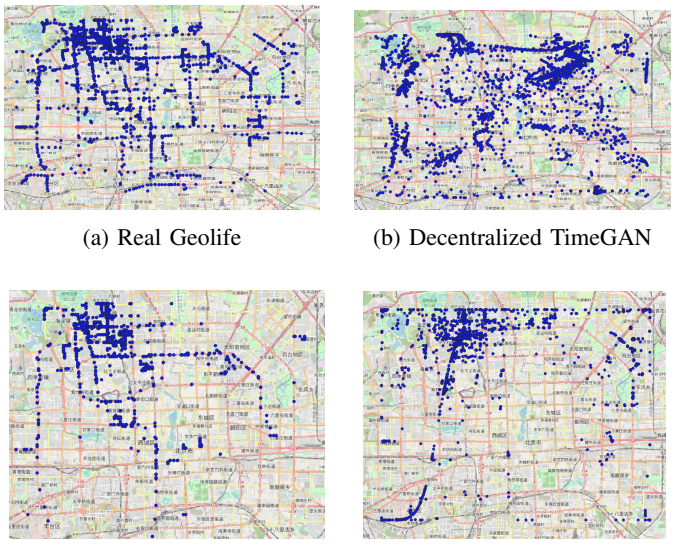
VI. Conclusion

In conclusion, this paper introduces a federated version of TimeGAN for trajectory generation, aiming to ensure privacy preservation in data synthesis. Our comprehensive evaluations reveal that the proposed federated TimeGAN performs comparably to our mentioned baseline, laying the groundwork for privacy-preserving trajectory generation. However, it is evident that there is room for improvement, particularly in addressing non-IID aspects and fine-tuning parameters. Additionally, our findings highlight the efficacy of decentralized TimeGAN, suggesting its potential application for local data augmentation and addressing data imbalances among clients, especially in non-IID scenarios for other federated learning tasks.

Looking forward, our future work will delve into refining the federated TimeGAN model by tackling non-IID challenges

TABLE I: Performance comparison of our model and baselines on geolife dataset, where lower results are better.

	Distance	***Radius of gyration***	***Duration***	***DailyLoc***	***G-rank***	***I-rank***
Decentralized TimeGAN	0.05922	0.04077	0.02104	0.18193	0.0084	0.03925
MoveSim	0.46628	0.35962	0.06477	0.48256	0.01764	0.0734
Federated TimeGAN	0.15438	0.38221	0.03796	0.36198	0.06871	0.03404



(a) Real Geolife (b) Decentralized TimeGAN

(c) Movesim (d) Federated TimeGAN

Fig. 1: Real and generated data distribution

and optimizing hyperparameters. Furthermore, we plan to extend the scope of our research to semantically enriched trajectories, acknowledging the multivariate nature of TimeGAN. The privacy aspect will be scrutinized further through attack simulations, and we aim to enhance privacy guarantees by exploring the incorporation of differential privacy.

REFERENCES

[1] 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. 1, pp. 1–5, 2013.

[2] B. Liu, W. Zhou, T. Zhu, L. Gao, and Y. Xiang, "Location privacy and its applications: A systematic study," *IEEE access*, vol. 6, pp. 17 606–17 624, 2018.

[3] S. Chen, A. Fu, J. Shen, S. Yu, H. Wang, and H. Sun, "Rnn-dp: A new differential privacy scheme base on recurrent neural network for dynamic trajectory privacy protection," *Journal of Network and Computer Applications*, vol. 168, p. 102736, 2020.

[4] E. Buchholz, A. Abuadbba, S. Wang, S. Nepal, and S. S. Kanhere, "Reconstruction attack on differential private trajectory protection mechanisms," in *Proceedings of the 38th Annual Computer Security Applications Conference*, 2022, pp. 279–292.

[5] A. Kapp, J. Hansmeyer, and H. Mihaljević, "Generative models for synthetic urban mobility data: A systematic literature review," *ACM Computing Surveys*, vol. 56, no. 4, pp. 1–37, 2023.

[6] L. Song, D. Kotz, R. Jain, and X. He, "Evaluating location predictors with extensive wi-fi mobility data," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 7, no. 4, pp. 64–65, 2003.

[7] R. Shokri, G. Theodorakopoulos, J.-Y. Le Boudec, and J.-P. Hubaux, "Quantifying location privacy," in *2011 IEEE symposium on security and privacy*. IEEE, 2011, pp. 247–262.

[8] Z. Lin, M. Yin, S. Feygin, M. Sheehan, J.-F. Paiement, and A. Pozdnoukhov, "Deep generative models of urban mobility," *IEEE Transactions on Intelligent Transportation Systems*, 2017.

[9] Q. Gao, F. Zhou, K. Zhang, G. Trajcevski, X. Luo, and F. Zhang, "Identifying human mobility via trajectory embeddings." in *IJCAI*, vol. 17, 2017, pp. 1689–1695.

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[11] K. Ouyang, R. Shokri, D. S. Rosenblum, and W. Yang, "A non-parametric generative model for human trajectories." in *IJCAI*, vol. 18, 2018, pp. 3812–3817.

[12] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," *Advances in neural information processing systems*, vol. 30, 2017.

[13] J. Feng, Z. Yang, F. Xu, H. Yu, M. Wang, and Y. Li, "Learning to simulate human mobility," in *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 2020, pp. 3426–3433.

[14] X. Wang, X. Liu, Z. Lu, and H. Yang, "Large scale gps trajectory generation using map based on two stage gan," *Journal of Data Science*, vol. 19, no. 1, pp. 126–141, 2021.

[15] N. Xu, L. Trinh, S. Rambhatla, Z. Zeng, J. Chen, S. Assefa, and Y. Liu, "Simulating continuous-time human mobility trajectories," in *Proc. 9th Int. Conf. Learn. Represent*, 2021, pp. 1–9.

[16] S. Choi, J. Kim, and H. Yeo, "Trajgail: Generating urban vehicle trajectories using generative adversarial imitation learning," *Transportation Research Part C: Emerging Technologies*, vol. 128, p. 103091, 2021.

[17] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[18] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečnỳ, S. Mazzocchi, B. McMahan *et al.*, "Towards federated learning at scale: System design," *Proceedings of machine learning and systems*, vol. 1, pp. 374–388, 2019.

[19] J. Yoon, D. Jarrett, and M. Van der Schaar, "Time-series generative adversarial networks," *Advances in neural information processing systems*, vol. 32, 2019.

[20] Y. Zheng, X. Xie, W.-Y. Ma *et al.*, "Geolife: A collaborative social networking service among user, location and trajectory." *IEEE Data Eng. Bull.*, vol. 33, no. 2, pp. 32–39, 2010.

[21] M. Zhang, H. Lin, S. Takagi, Y. Cao, C. Shahabi, and L. Xiong, "Cs-gan: Modality-aware trajectory generation via clustering-based sequence gan," in *2023 24th IEEE International Conference on Mobile Data Management (MDM)*. IEEE, 2023, pp. 148–157.