SoK: Can Trajectory Generation Combine Privacy and Utility?

Erik Buchholz
University of New South Wales
CSIRO's Data61, Cyber Security CRC
Sydney, NSW, Australia
e.buchholz@unsw.edu.au

Alsharif Abuadbba
CSIRO's Data61
Cyber Security CRC
Sydney, NSW, Australia
sharif.abuadbba@data61.csiro.au

Shuo Wang
CSIRO's Data61
Cyber Security CRC
Sydney, NSW, Australia
shuo.wang@data61.csiro.au

Surya Nepal
CSIRO's Data61
Cyber Security CRC
Sydney, NSW, Australia
surya.nepal@data61.csiro.au

Salil S. Kanhere
University of New South Wales
Sydney, NSW, Australia
salil.kanhere@unsw.edu.au

ABSTRACT

While location trajectories represent a valuable data source for analyses and location-based services, they can reveal sensitive information, such as political and religious preferences. Differentially private publication mechanisms have been proposed to allow for analyses under rigorous privacy guarantees. However, the traditional protection schemes suffer from a limiting privacy-utility trade-off and are vulnerable to correlation and reconstruction attacks. Synthetic trajectory data generation and release represent a promising alternative to protection algorithms. While initial proposals achieve remarkable utility, they fail to provide rigorous privacy guarantees. This paper proposes a framework for designing a privacy-preserving trajectory publication approach by defining five design goals, particularly stressing the importance of choosing an appropriate Unit of Privacy. Based on this framework, we briefly discuss the existing trajectory protection approaches, emphasising their shortcomings. This work focuses on the systematisation of the state-of-the-art generative models for trajectories in the context of the proposed framework. We find that no existing solution satisfies all requirements. Thus, we perform an experimental study evaluating the applicability of six sequential generative models to the trajectory domain. Finally, we conclude that a generative trajectory model providing semantic guarantees remains an open research question and propose concrete next steps for future research.

KEYWORDS

Trajectory Privacy, Differential Privacy, Location Privacy, Deep Learning, Generative Adversarial Networks

1 INTRODUCTION

Location trajectories are valuable for several applications, from navigation services over research to pandemic control. Moreover, a large amount of location data is collected daily through the increasing number of sensor-equipped devices, particularly smartphones. However, the information content of location trajectories poses a risk of exposing sensitive details about individuals, such as religious, political, or sexual beliefs [2, 58]. For instance, researchers showed in 2013 that only four locations provided through the connection to mobile phone antennas suffice to uniquely identify 95 % of users [16]. In the case of an anonymised dataset released by a New York taxi company, a Redditor identified which taxi drivers are practising Muslims by correlating the drivers' break times with mandatory prayer times [25]. Such examples illustrate that mobility datasets require appropriate protection when released.

Various protection mechanisms based on k-Anonymity [70] and Differential Privacy (DP) [26, 33, 35, 36, 42] have been proposed. Recent works favour DP [19] for its resistance to background knowledge attacks [10, 30, 37]. However, DP introduces a privacy-utility trade-off by adding noise [59]. Numerous works point out that the existing DP approaches degrade utility to a level that is no longer useful for analyses [46, 60, 63]. Moreover, the added noise can lead to structural differences, enabling reconstruction attacks [5, 64]. Importantly, Miranda-Pascual et al. [52] uncovered flawed proofs in key DP works like Hua et al. [35] and Li et al. [42], affecting their privacy claims. Recent approaches [10, 77] based on these mechanisms are subsequently impacted by the same issues.

Hence, we propose a design framework for privacy-preserving trajectory publication approaches based on five design goals, aiming to address and avoid the shortcomings evident in prior research. These design goals include: (i) ensuring proven privacy guarantees, (ii) defining and emphasising the Unit of Privacy (UoP), (iii) valuing practical privacy assessments alongside theoretical guarantees, (iv) prioritising high utility and its thorough evaluation, and (v) considering the practical implementation of the approach. Using this framework, we systematically examine existing DP trajectory protection mechanisms, underscoring their limitations.

Acknowledging the difficulty of balancing high utility with strict privacy in (DP) methods, a vision paper [44] proposed *trajGANs* – Generative Adversarial Networks designed for trajectory data synthesis. Synthetic data could replace real data in analyses, thereby safeguarding individual privacy. Our work systematises approaches inspired by this vision, focusing on their alignment with our proposed design goals. LSTM-TrajGAN [63] represents a widely recognised work following this vision. While LSTM-TrajGAN achieves high utility with the generated data, the approach fails to provide rigorous semantic privacy guarantees. We show the practical implications by successfully applying the attack proposed in [5] to

Privacy Enhancing Technologies 2024(3), 1–19

LSTM-TrajGAN. Moreover, we find that no other proposed generative trajectory model achieves sufficient privacy guarantees yet.

These findings motivate an experimental study evaluating the applicability of generative models from other domains to trajectory generation. In particular, we consider two simple Recurrent Neural Network (RNN)-based models [34], a privacy-enhanced adaption of LSTM-TrajGAN [63], Recurrent GAN (RGAN) [23] successful for medical signal generation, WaveGAN [18] used for audio generation, and a sequential adaptation of GeoPointGAN (GPG) [14] used for location generation. First, we test the models on a toy dataset, MNIST Sequential (MNIST-Seq), before evaluating all models on two well-known trajectory datasets, Geolife and Foursquare NYC [17] (FS-NYC). While all models produce acceptable results on the toy dataset, no model can adequately capture the point distribution of the trajectory datasets. Despite the underwhelming performance, our findings suggest that Conv1D-based models may outperform RNN-based ones and indicate the potential of an ensemble approach that combines point and sequential properties. We also identify the use of Differentially Private Stochastic Gradient Descent (DP-SGD) as a promising method to attain genuine DP guarantees, considering an appropriate UoP in generative trajectory models. Our analysis concludes that designing a trajectory-generating model that offers robust privacy guarantees remains an urgent and open research gap. This systematisation of knowledge contributes to the field of trajectory privacy in the following ways:

- We propose a framework for developing privacy-preserving trajectory publication approaches based on five pivotal design goals (Section 3).
- We contrast traditional trajectory protection methods with synthetic trajectory generation (Sections 4 and 5).
- We critically assess the prevailing privacy-preserving trajectory synthesis techniques, identifying gaps in the current state-of-the-art (Section 6).
- We reproduce the results of proposed generative models and perform additional measurements, e.g., showing the success of the Reconstruction Attack on Protected Trajectories (RAoPT) attack against LSTM-TrajGAN (Section 6).
- We evaluate the adaptability of six generative models to trajectory data in an experimental study, finding that none of the models is easily applicable (Section 7).
- We make all our experimental code available online[1], ensuring reproducibility and facilitating future work.

2 BACKGROUND

This section discusses the background knowledge for this work. Section 2.1 formally defines trajectory datasets and introduces the datasets used for evaluation. Section 2.2 explains DP, and Section 2.3 looks at DP-SGD, the most common application of DP to deep learning. Then, we introduce generative models in Section 2.4. Finally, Section 2.5 describes attacks against trajectory privacy methods.

2.1 Trajectory Datasets

A trajectory dataset D consists of a number of trajectories $D = \{T_{u1}, T_{u2}, \ldots, T_{zm}\}$ where T_{ui} refers to the i^{th} trajectory of user u. Each user might contribute one or multiple trajectories to the dataset. A trajectory T can be represented as an ordered sequence of locations: $T = (l_1, \ldots, l_n)$. Each location has at least two coordinates, typically latitude and longitude $l_i = (lat_i, lon_i)$, with optional additions like elevation for greater precision [81]. Additionally, trajectories can include semantic information such as Point of Interests (POIs) (e.g., restaurant, shop, gym) to provide context to locations [71]. While such additional information can increase the utility of a dataset for analyses, semantic information facilitates Trajectory User Linking (TUL) [50, 71]. In a pure semantic trajectory, each location only represents a semantic location, such as a shop, without associated location information, e.g., credit card transaction datasets. In this work, we assume each location consists of spatial coordinates, i.e., latitude and longitude. Semantic information is optional as not all datasets record it.

2.2 Differential Privacy

Privacy notions are typically categorised into two types [47]: *syntactic* and *semantic* notions. *Differential Privacy (DP)* [20] represents the main semantic privacy notion used to protect personal information [30, 47]. The central intuition of differential privacy is that adding or removing any user's data to a dataset does not significantly change the output. Accordingly, participation does not harm users' privacy as they have *plausible deniability* regarding participation. The mathematical definition is as follows [20]:

Definition 2.1 (Differential Privacy). A mechanism $\mathcal{K}$ provides (ε, δ)-differential privacy if for all *neighbouring* datasets D_1 and D_2, and all $S \subseteq Range(\mathcal{K})$ holds

$$\mathbb{P}[\mathcal{K}(D_1) \in S] \leq e^{\varepsilon} \times \mathbb{P}[\mathcal{K}(D_2) \in S] + \delta \tag{1}$$

Based on this definition, determining when two datasets are considered *neighbouring* is pivotal for ensuring privacy guarantees. Therefore, we dedicate Section G2 to discussing the definition of *neighbourhood* in the context of trajectory datasets.

Consider mechanism $\mathcal{K}$ computing a noisy average over a dataset. If the data of another user is added to the dataset D_1 yielding dataset D_2, the change of the probabilities for the outputs of $\mathcal{K}$ is bounded through ε. The smaller ε is chosen, the higher the provided privacy level. In the literature, common values for ε range from 0.01 to 10 [22]. However, in machine learning using DP-SGD, larger values for ε might be used in practice [57]. The value δ represents a failure probability for which ε-DP can be violated. The best practice is to choose $\delta = \frac{1}{n}$ where n is the number of records in the dataset, such that no entire record can remain unprotected [57]. The most common way to design a differential private mechanism is through the Laplace mechanism [20], Gaussian mechanism [20], or Exponential Mechanism (EM) [51]. Differential privacy offers a few valuable properties that aid in the design of more complex mechanisms:

Post-Processing. Differential privacy is *immune to post-processing* [20]. This means that the output of any differential private mechanism can be post-processed to enhance the utility of the output without reducing the provided privacy guarantees [20]. However, the post-processing *must not* access the original data.

Sequential Composition. Moreover, the *Sequential Composition Theorem* states that the sequential composition of n $(\varepsilon_i, \delta_i)$-DP mechanism provides $\sum_{i=1}^{n}(\varepsilon_i, \delta_i)$-DP. This allows designing a DP algorithm as a combination of multiple individual algorithms.

[1] https://github.com/erik-buchholz/SoK-TrajGen

Parallel Composition. In cases where the records of a dataset are partitioned and processed by separate algorithms providing $(\varepsilon_i, \delta_i)$-DP, such that each record is only accessed by exactly one algorithm, the combined algorithm provides $(\max_i \varepsilon_i, \max_i \delta_i)$-DP.

2.3 Differentially Private Deep Learning

Giving users or third parties access to a trained Deep Learning (DL) model represents a common business practice [15]. Used datasets are commonly private because they contain sensitive information about individuals and business secrets, or sharing is limited by legislative constraints. For instance, a DL-powered text-to-image converter might be trained on a business's internal and private texts and images. However, Membership Interference Attacks (MIAs) [66] can recover (sensitive) training data from a released model even if the data itself is not published. To prevent the leakage of private training data, differential privacy has been introduced to DL [1, 57].

We refer the reader to a recent report by Ponomareva et al. [57] for a thorough analysis of differentially private DL. Here, we briefly describe *Differentially Private Stochastic Gradient Descent (DP-SGD)* [1], which represents the most widely implemented application of DP to deep learning [57]. The first step of DP-SGD is clipping the gradients to a clipping norm C such that the influence of each training sample is bounded. Second, Gaussian noise is added to the gradients based on this clipping norm C and a noise multiplier σ. Third, privacy accounting tracks the ε budget during training to ensure a (ε, δ)-DP guarantee upon completion.

2.4 Generative Models

Generative models for sequential data [21], such as trajectories, have been proposed to generate more data for domains with limited data available or to generate synthetic data in order to replace real data that is sensitive, e.g., in the medical domain [23]. Here, we focus on those architectures used in the context of trajectories.

Recurrent Neural Network (RNN). *RNNs* sequentially process inputs by maintaining a memory of previous inputs through a hidden state that is passed on from one cell to the next. This makes RNNs well-suited for trajectories, as each location relies on previous ones. The most common type of RNN in the context of trajectories is the Long Short-Term Memory (LSTM) [34], which improves the ability to understand long-term dependencies.

Autoencoder (AE). *AEs* [4] consist of an encoder that encodes a given input into a lower-dimensional latent representation and a decoder that aims at reconstructing the original input. Common applications are dimensionality reduction and denoising. The latent bottleneck forces outputs to be different, while the most important information is preserved if the model is trained correctly.

Variational Autoencoder (VAE). *VAEs* [41] are a probabilistic extension of AEs. The encoder maps inputs to latent *distributions*, enforced by minimising the KL divergence to the target distribution. The decoder then samples from the latent space distributions to generate data. This enables VAEs to both reconstruct and generate.

Generative Adversarial Network (GAN). *GANs* [29] are composed of two separate models, the generator G and the discriminator D. The generator G receives noise as input and tries to generate samples that are indistinguishable from real data samples. Meanwhile, the discriminator D judges whether a given input sample comes from the real dataset or the generator. The resulting zero-sum game is represented by the adversarial loss [29]:

$$\min_G \max_D \mathbb{E}_{x\sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z)))] \quad (2)$$

Here, p_{data} is the real data distribution, and p_z is G's input noise distribution. The generator and discriminator train iteratively until G produces samples that can barely be distinguished from real data.

Adversarial Autoencoder (AAE). *AAEs* [48] combine AEs with the adversarial concept of GANs. The encoder transforms inputs into latent representations that approximate a target distribution. In contrast to VAEs, AAEs employ an adversarial loss, i.e., a discriminator's feedback, to ensure the latent space adheres to the designated distribution. Like VAEs, AAEs can reconstruct input data and generate new synthetic data.

2.5 Practical Attacks against Trajectory Privacy

Several attacks on trajectory protection highlight the risks of insufficient protection. Miranda-Pascual et al. [52] provide a comprehensive overview. Here, we focus on two practical attacks, namely RAoPT [5] and TUL [50], which provide open-source code [6, 45] such that we propose them for empirical evaluation in Section G3.

TUL. While a single trajectory might not reveal sensitive information or allow identification, linking multiple trajectories to the same user increases these risks [50, 63]. Therefore, some protection mechanisms [24, 63, 65, 67] utilise *TUL* to evaluate the provided privacy level, with a lower TUL success rate implying better privacy. One recent and openly available [45] TUL algorithm is MARC [50]. This DL model consisting of embedding, RNN, and softmax layers matches multi-aspect trajectories containing semantic information, e.g., POI information, to users.

RAoPT. Buchholz et al. [5] note that the perturbation of DP protection mechanisms causes structural differences in the outputs, e.g., zigzag patterns. They develop the RAoPT DL model, which trains on protected samples as inputs and real samples as targets to understand their relationship. The trained model can then reconstruct versions closer to the originals from unseen protected trajectories, thereby reducing perturbations and compromising privacy.

3 DESIGN GOALS

This section introduces our design framework for privacy-preserving trajectory publication mechanisms. The overarching goal can be defined as: *The release of high-utility trajectory data without revealing private information about the individuals participating in the dataset.* In the following, we discuss the five goals forming our design framework. These goals are *not* ordered by importance, as this depends on the specific dataset and use case.

G1: Formal Privacy Guarantees

Ensuring stringent privacy guarantees is essential when releasing trajectory datasets. While practical privacy evaluations against specific attacks have advantages, as we discuss in Section G3, formal guarantees are irreplaceable. First, testing against a designated attack only demonstrates resistance to that specific threat. It does not vouch for protection against other existing attacks, unforeseen future threats, or enhanced algorithms of the same attack. Second, this method lacks quantifiable privacy guarantees. In a commercial

context, evidence of rigorous efforts to safeguard shared data is necessary. Leveraging formal guarantees, like DP with established ε and δ parameters, provides such evidence. It signifies that state-of-the-art protection, endorsed by the research community at the time of release, was implemented. In case of a privacy incident, these guarantees testify to the releasing entity's earnest attempts at privacy preservation. Differential privacy stands out as the de facto standard for semantic privacy guarantees. However, specific scenarios might allow for DP relaxations. One notable example is Local Label Differential Privacy (LLDP) used by GPG (ref. Section 6).

Local Differential Privacy (LDP). In standard (global) DP, a central entity applies protections, such as noise addition, to data collected from clients before sharing it. This requires trust from the clients in the central entity. LDP eliminates this trust requirement by applying protection *locally* on the client's device before sending it to the central entity, enhancing privacy guarantees at the cost of potentially greater utility loss. This work presumes trust in the central data collector as global DP is the most common notion for trajectory privacy. However, we highlight when approaches provide the stronger LDP guarantees.

Common Pitfalls. Miranda-Pascual et al. [52] demonstrate that multiple foundational works on DP trajectory protection rely on erroneous proofs, making quantifying the provided privacy level challenging. We highlight similar issues affecting further works in Sections 5 and 6. While the formal proofs for these flaws are provided in [52], we summarise the most common pitfalls:

(1) **Not Considering All Elements [52]:** A DP mechanism must be able to output any *possible* value *independent of the dataset* [52]. However, some methods [73, 77–80] only add noise to existing elements, leaving unobserved elements with a count of 0. Consequently, an element e not present in dataset D would always have a count of 0. If neighbouring dataset D' contains e, the difference between D and D' cannot be bounded by ε (ref. Equation 1). This contradicts the definition of ε-DP.

(2) **Exponential Mechanism (EM) Application [52]:** The EM's outputs depend on a score function $u(d, r) : D \times R \rightarrow \mathbb{R}$ that assign a score u to each possible output $r \in R$ based on the dataset $d \in D$. However, the set of all possible datasets D and outputs R must be well-defined. Several studies [10, 32, 35, 42, 46, 75] inaccurately define R as dependent on the selected dataset d. This renders the EM indefinable and invalidates any DP claim based on it [52].

(3) **Access to Real Data:** The post-processing property permits any manipulation of DP outputs *not accessing the original data* (ref. Section 2.2). However, certain methods [74] access the raw data via side channels, undermining any DP protection measures.

Conclusion. Semantic privacy guarantees are inevitable even in the presence of privacy evaluations. The gold standard is Differential Privacy, but relaxations might be justifiable. LDP provides stronger guarantees, making it a preferable choice when achievable. Rigorous verification of privacy claims is essential, highlighted by some baseline studies' flawed DP proofs.

G2: Unit of Privacy

For privacy guarantees, choosing the correct unit of data to protect is of utmost importance. The choice of a *Unit of Privacy (UoP)* is equivalent to the question of what makes two inputs (datasets) of a DP mechanism "neighbouring" (ref. Section 2.2). A larger UoP requires more obfuscation to achieve the same privacy level. Thus, the UoP should be chosen as small as possible yet large enough to ensure privacy. Protecting a too-small unit of data is a common problem and can lead to vulnerability to correlation [52] or reconstruction attacks [5]. Recent literature discussed the UoP, e.g., Ponomareva et al. [57] focusing on machine learning and Miranda-Pascual et al. [52] (using the term *level of granularity*) for trajectory data. We use the term UoP and provide a consolidated systematisation specific to location trajectories. This shall support researchers in selecting the appropriate UoP for their application. In the following, we consider the dataset D_1 consisting of multiple trajectories $D_1 = (T_{u1}, T_{u2} \ldots, T_{zm})$, where T_{ui} is the i^{th} trajectory of user u.

User-level Privacy corresponds to the original definition of differential privacy [20]. For a trajectory dataset D_1, D_2 can be obtained by removing all trajectories belonging to one user u: $D_2 = D_1 \setminus \{T_{vi} | v = u\}$. While user-level privacy offers the most robust protection, it often leads to a notable reduction in data utility. As user-level privacy represents the original definition of DP, all of the following units of privacy constitute relaxations thereof.

Instance-Level Privacy, or *example-/trajectory-level* privacy, safeguards individual trajectories T_{ui} with a specific privacy guarantee. According to the composition theorem (ref. Section 2.2), ε instance-level differential privacy yields $m\varepsilon$ user-level differential privacy for a user contributing m trajectories. Hence, instance-level and user-level privacy are equivalent for datasets containing one trajectory per user. Due to the protection of a trajectory as one unit, this level protects against any attack exploiting the correlation within a trajectory. Instance-level DP is the most common level in DL [57], as DP-SGD provides instance-level privacy for the training samples.

Location-Level Privacy refers to *event-level* privacy in the general setting [52]. In the case of a trajectory $T = (l_1, \ldots, l_n)$, each location l_i is protected with the privacy guarantee independent of the other locations. This weakest level of privacy requires the least perturbation. While literature [36, 40] uses location-level privacy for trajectory protection, it is vulnerable to correlation and reconstruction attacks. Early works on location privacy [3] already warned against using such solutions for trajectory projection.

Multi-event-level Privacy bridges instance- and location-level privacy, accommodating notions such as w-event-level and l-trajectory-level privacy (see [52] for a detailed discussion). It covers a window of multiple events within a trajectory, equating to location-level privacy at $w = 1$ and instance-level at $w = \max\{|T_i| \in D\}$. This approach is useful when sensitive information, like a hospital visit, spans multiple locations such that location-level privacy is insufficient, yet trajectory-level privacy is too restrictive.

Example. To demonstrate the significance of selecting the appropriate UoP, consider the trajectory protection methods CNoise and the SDD mechanism [36]. Despite their DP guarantees, reconstruction attacks [5, 64] successfully target these mechanisms. These attacks do not contradict the rigorous definition DP. Instead, both attacks leverage the correlations between and within trajectories. The wrong UoP, i.e., protecting individual locations instead of the entire trajectory, makes these mechanisms vulnerable. Nevertheless, recent works [40] still use location-level approaches in the context of trajectory privacy (ref. Section 6).

	Preservation	Statistics
Point Level	HD Density	Range Query Hotspot Preservation
Trajectory Level	DTW† HD†	Travelled Distance Segment Length

Table 1: Utility Metric Systematisation. Metrics with † require a 1:1-mapping between input and output trajectories.

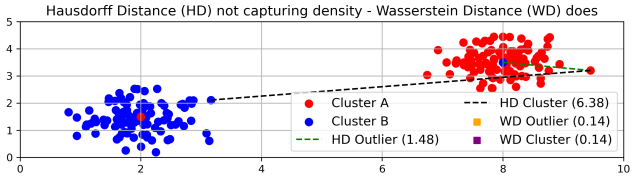

Figure 1: Density problem of Hausdorff Distance (HD). A single outlier significantly reduces the HD but not the WD.

Conclusion. Given these findings, we strongly advise researchers to carefully select the UoP and refrain from using location-level privacy for trajectories. We view instance-level privacy as a promising balance between utility and privacy due to the protection against intra-trajectory correlations and its applicability to DL. However, employing instance-level privacy requires special care for recurring locations in multiple trajectories, such as home or work.

G3: Empirical Privacy

Proof of formal privacy guarantees (**G1**) and adherence to the appropriate UoP (**G2**) might make a practical privacy assessment seem unnecessary. However, as Miranda-Pascual et al. [52] showed, numerous DP trajectory protection mechanisms are built on flawed DP proofs. Additionally, vulnerabilities can arise from the incorrect selection of the UoP, as elaborated in Section G2. Given these common challenges, we advocate for integrating practical privacy evaluations into future publications to identify potential issues and emphasise the achieved privacy levels.

Recent studies [24, 63, 65, 67] have employed the publicly available *Trajectory User Linking (TUL) algorithm MARC* [45, 50] as practical privacy assessment. Additionally, the novel *reconstruction attack RAoPT [5]* provides publicly available code [6], making it a suitable evaluation tool for trajectory protection mechanisms. Several approaches [10, 42, 43, 74, 75, 77] use the *Mutual Information (MI)* metric to measure the provided privacy level. MI measures the dependence of the protected dataset on the original dataset such that a lower value indicates better privacy. However, as the goal for the published dataset is to retain as much useful information as possible while removing all private information, a low MI might also indicate low utility. Therefore, we consider the aforementioned practical attacks more meaningful in quantifying practical privacy.
Conclusion. While not strictly necessary, we recommend evaluating trajectory publication approaches against at least one practical attack, even with formal privacy guarantees. The open-source attacks TUL [45], and RAoPT [6] represent suitable candidates.

G4: Utility

The primary aim of a released dataset is the usability for downstream tasks and analyses. Any protection mechanism has to trade off privacy and utility [59]. Thus, utility represents our fourth design goal. We systematise utility metrics, discuss data aggregation and precision, and address environmental constraints.
Utility Metrics. Recent studies [52, 68] have analysed various trajectory utility metrics. Rather than reiterating these, we aim to establish a selection framework. The large number of metrics highlights the core issue. The absence of a universally adopted metric makes comparing different methodologies difficult. Thus, we recommend using established metrics in future research.

We propose classifying utility metrics along two axes as shown in Table 1. First, metrics can be separated into *data preservation* and *statistical* metrics [52]. Data preservation metrics evaluate original data retention, while statistical metrics assess the conservation of overall patterns. Second, metrics can be oriented towards *point properties* or *sequential properties*. A prevailing trend is emphasising point-level metrics, potentially neglecting sequential coherence.

We present example metrics covering all categories, focusing on continuous, spatial coordinate trajectories. For a comprehensive list, see [52, 68]. Metrics concentrate on spatial dimensions as minimal information of trajectories (ref. Section 2.1). Datasets with additional properties, like semantics, should incorporate these, e.g., via the *Spatial-Temporal-Categorical Distance (STC)* [12, 52]. For *point preservation*, the *Hausdorff Distance (HD)* is commonly used [10, 35, 43], comparing the distance of points in one dataset to their nearest counterparts in another. However, it may not effectively capture discrepancies in point distributions' densities. Consider the two clusters A (red) and B (blue) in Figure 1. First, assume that the clusters are distinct, i.e., ignore the single point with the opposite cluster. Then, the black line represents the HD with the value 6.38. However, with a single outlier of B in the centre of A, the HD reduces to the green line, resulting in a value of 1.48. This sensitivity to outliers limits the HD's effectiveness for comparing generated (B) and real (A) point clouds. Therefore, we suggest additionally comparing the spatial distributions using metrics like the Wasserstein Distance (WD) or Jensen Shannon Distance (JSD). In the example, the value of the WD remains consistent at 0.14. In terms of *point statistics*, *range queries* [12–14, 31, 35, 43] and its discrete counterpart count queries [8, 9, 32, 69] have gained prominence, often paired with *hotspot preservation queries* [12–14, 31, 72]. *Trajectory preservation* metrics usually require a *1:1-mapping* for comparison, i.e., each output trajectory should match an original trajectory. However, some generative models, such as GANs, create new samples from noise input (ref. Section 2.4) without a clear link to an original sample. This makes it unclear which trajectories to match for comparison. Hence, applying trajectory preservation metrics to some generative models is difficult. One possible solution is matching each generated trajectory to the closest original. If a 1:1 mapping is given, the spatial metrics HD [38, 40, 43, 46, 63] and Dynamic Time Warping (DTW) [36, 38, 76] represent suitable candidates. Note that in this case, the HD between trajectories are computed, whereas, for point preservation, the HD between (an equal-sized subset of) all points of the original and protected dataset is evaluated. We note a research gap in trajectory preservation metrics for generative models without 1:1 mapping. Lastly, *Trajectory*

statistics should focus on the sequential properties. Therefore, we deem the total *Travelled distance* [26, 31–33, 38, 72] and the *Segment lengths* [31, 33] as appropriate metrics. The former computes the total distance travelled from start to end point for each trajectory, while the latter computes the distance between any two consecutive points. Then, the resulting distributions of the original and generated dataset are compared, e.g., via JSD or WD. However, metrics are not the only utility evaluation factor.

Aggregation. Various DP methods release aggregated datasets, presenting counts for representative trajectories [10, 35, 42, 43]. This approach requires less location obfuscation since the information from multiple trajectories can be consolidated. However, aggregation invariably results in detail loss. Thus, individual trajectory release is preferable for maximal utility, especially if the data's eventual application is unknown at release time.

Precision. Utility in data preservation is directly linked to precision. Broadly, trajectories can be categorised into *POI-trajectories* and *coordinate trajectories*. The former involves locations from a predefined, countable set of POIs. In contrast, the latter typically involves arbitrary coordinates, like latitude and longitude. Generally, POI-based methods yield higher utility at a given privacy level than coordinate-based ones. However, this article focuses on coordinate trajectories due to their broader applicability.

Grid Approaches. Some solutions [26, 33] convert the continuous space into a grid, discretising the data by assigning each location to a grid cell. While this shares some benefits with POI trajectories, it introduces challenges. The precision is constrained by grid granularity [72], which is limited by computational considerations. A finer grid translates into a more extensive range of values, increasing the computational effort. Furthermore, the grid's spatial extent is typically limited (e.g., to a city) to maintain meaningful cell sizes.

Environmental Constraints. Recent literature [5, 12–14, 52, 54, 69] underlines the importance of *environmental* constraints. Protection methods can cause violations such that obfuscated trajectories may pass through rivers or physical barriers [5, 12], consecutive locations may not be reachable [13], or cars do not follow the road network [5]. Considering these constraints improves utility and prevents using this knowledge for reconstruction attacks [5].

Conclusion. For meaningful comparisons among approaches, it is vital to employ *established utility metrics*. These metrics should cover both *point* and *trajectory aspects* and ideally evaluate both *data preservation* and *statistical attributes*. The release of *individual trajectories* is preferable over aggregated data. Solutions for *coordinate trajectories* are most generally applicable, but *POI datasets* or *grid-based* approaches might present optimised results for some cases. Finally, considering *environmental constraints* is essential.

G5: Practicality

Releasing a dataset usually occurs once, making computational cost a secondary concern to utility and privacy. However, *practicality* remains crucial, i.e., protection approaches must not require high-cost, specialised equipment but commodity hardware should suffice. Furthermore, a predictable runtime is preferable. For example, the Sampling Distance and Direction (SDD) mechanism's [36] probabilistic sampling can yield uncertain and extended runtimes of many hours as confirmed by the RAoPT artefacts [6]. Moreover, evaluations should consider at least two datasets, ideally publicly available. Trajectory datasets differ significantly in granularity (5 s vs. per minute), span (city-level vs. global), and additional information. Beyond the challenges of differing datasets and metrics, reproducibility remains a hurdle, often due to insufficient publication details and unresponsive authors. LSTM-TrajGAN [63] demonstrates the benefits of publishing source code (ref. Section 6). While LSTM-TrajGAN influences many generative trajectory models, models without public code often receive limited attention.

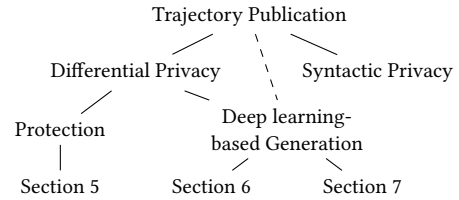

Figure 2: Categorisation of trajectory publication.

Conclusion. The trajectory publication approach should (i) run on *commodity hardware*, (ii) have a *deterministic runtime*, (iii) be evaluated on *at least two public datasets*, and (iv) ideally *share the source code* for reproducibility. Based on these five goals, we examine existing trajectory publication schemes in the following.

4 SYSTEMATISATION OF APPROACHES

We outlined our design framework for trajectory publication mechanisms in the preceding section. Such mechanisms are classified based on several attributes, highlighted in Figure 2. The primary distinction hinges on the targeted privacy notion, typically categorised into *syntactic* and *semantic* notions [47]. *Syntactic privacy notions*, such as k-Anonymity [70], though common, are susceptible to background knowledge attacks and do not provide rigorous guarantees [3, 10, 74, 77]. Conversely, DP (ref. Section 2.2), the leading *semantic privacy notion*, represents the current de facto privacy standard due to its strong guarantees. Furthermore, some methods, like GANs-based ones, ensure privacy by generating new data rather than targeting specific privacy notions. Yet, these mechanisms may also incorporate privacy notions alongside their generative properties.

This work discusses *DP protection mechanisms* (Section 5) and *Deep Learning (DL)-based trajectory generation* (Section 6 and 7). To explore k-anonymity-based methods, readers are referred to [39]. The separation between protection and generation approaches overlaps, as some obfuscation-based protections, such as DPT [33], record noisy statistics of the original dataset and sample from these to generate protected trajectories. We achieve clarification by categorising approaches based on DL as *DL-based trajectory generation* while we discuss all other approaches jointly as *DP protection mechanisms*. The DL-based trajectory generation approaches represent a recent development, with the first vision proposed in 2018 [44]. To the best of our knowledge, they have not been thoroughly discussed and systemised regarding privacy preservation and represent the focus of this work. Next, we discuss the protection mechanisms and describe the state-of-the-art trajectory generation in Section 6.

Category	Approach(es)	In [52]	UoP	G1	G2	G3	G4	G5	Main Shortcoming
Ⓐ Tree-based	[8, 9]	✓	Instance		✓	–	✗	✗	Requires pre-defined location domain
Ⓑ Noisy Count	[73, 78–80]	✓	Instance	✗	✓	–	✗	✓	Broken DP proof: Only existing traj. considered (**Pitfall 1**)
Ⓒ Clustering Based	[10, 35, 42]	✓	Instance	✗	✓	–	✗	○	Broken DP proof (EM – **Pitfall 2**); Aggregated output
	[46, 75]	✗	Instance	✗	✓	–	✗	○	Broken DP proof (EM – **Pitfall 2**); Aggregated output
Ⓓ Location Privacy	[3, 7, 36]	✗	Location		✗	✗	○	○	Wrong UoP; Successful attack known for [36]; Length-based utility decline
Ⓔ Grid-Cell based	[26, 33, 69]	✗	Instance		✓	–	○	○	Grid cell granularity limits precision or performance
Ⓕ Environmental	[32]	✗	Instance	✗	✓	–	✓	✓	Broken DP Proof (EM – **Pitfall 2**)
Constraints	[12]	✗	Location		✗	–	✓	✓	Generates points only
Ⓖ POI protection	[13]	✓	Instance		✓	–	✗	✓	Only for POI data; Requires public knowledge to be available

Table 2: Overview of Differentially Private Protection Mechanisms discussed in Section 5. The table lists each approach's inclusion in the SoK [52], targeted UoP, adherence to design goals from Section 3, and primary shortcomings. Symbols: ✓ (goal satisfied), ○ (partially satisfied), and ✗ (unsatisfied). For G1, ✗ denotes known DP proof errors; the absence of a symbol means we are unaware of any flaw. For G3, ✗ indicates demonstrated attacks and approaches with – have not been evaluated.

5 DIFFERENTIAL PRIVACY PROTECTION

Chen et al. [8] first introduced the application of Differential Privacy (DP) to location trajectories in 2011. Since this pioneering work, various approaches have evolved in the DP trajectory protection field. This Section briefly addresses these protection methods, highlighting their limitations in meeting the design goals outlined in Section 3. These limitations motivate alternative solutions, such as DL-based approaches for trajectory generation (Section 6), which represent the focus of this work. Miranda-Pascual et al. [52] provide a comprehensive analysis of DP trajectory protection mechanisms in their SoK. Table 2 summarises our observations and indicates the approaches detailed by Miranda-Pascual et al. [52].

Ⓐ **Tree-based Approaches.** The first DP mechanism [8] is based on noisy prefix trees, i.e., a tree representing all possible sequences of locations. The occurrences of each prefix are counted, and Laplace noise is added to achieve DP before reconstructing output trajectories based on the noisy tree. The authors improved the approach's utility by using n-grams in subsequent work [9]. However, all mechanisms in this line of research rely on a predefined set of discrete locations. Hence, these approaches only apply to POI datasets but not to general coordinate trajectories, violating **G4: Utility**. Enumerating all locations within a particular area, e.g., using a grid, incurs unreasonable computational overhead as the tree size increases exponentially [52], violating **G5: Practicality**.

Ⓑ **Noisy Count-based Approaches.** Other approaches [73, 78–80] utilising noisy counts are summarised in [52]. The authors show that these methods contain an error in their DP proofs, violating **G1: Formal Privacy Guarantees**. Only the counts for existing sequences are perturbed, but not for all possible (count zero) sequences (**Pitfall 1**). However, a DP mechanism has to produce any output with a probability larger than zero. Otherwise, the mechanism's outputs cannot be bounded (ref. [52] for a formal proof).

Ⓒ **Clustering-based Approaches.** One line of research [10, 35, 42, 46, 75] is based on clustering the locations at timestamps, followed by sampling from these clusters. Miranda-Pascual et al. [52] show that the EM is used incorrectly by these approaches (**Pitfall 2**), breaking the formal DP proof: Proper EM application requires dataset-independent output sets, whereas these methods use the dataset-dependent set of possible partitions, breaching **G1: Formal Privacy Guarantees**. Recent studies [46, 75], not covered in [52], repeat this error. Furthermore, they aggregate data into reference trajectories with noisy counts, thereby reducing utility (**G4: Utility**). They also risk generating implausible trajectories due to poor cluster representation choices and assume uniform lengths and sampling rates, which is impractical for real-world data [52].

Ⓓ **Location-level Privacy.** Other approaches protect each trajectory location individually, such as the SDD mechanism and CNoise [36]. The former samples a distance and direction from one location to the next, while the other adds Laplace noise to each location. These mechanisms violate **G2: Unit of Privacy**, as the privacy budget ε is used for a single location. This makes the protection vulnerable to reconstruction attacks like RAoPT [5] (violating **G3: Empirical Privacy**). Moreover, the SDD mechanism requires probabilistic sampling until a suitable next location is found, which can lead to run times of several hours for datasets such as Geolife [81] (see Algorithm 5 [36] or implementation [6]). This stays in conflict with **G5: Practicality**. Other approaches like Geo-Indistinguishability (Geo-Ind)[3], aimed at location privacy, are not directly applicable to trajectory privacy due to location correlation, as outlined in **G2: Unit of Privacy**. The predictive mechanism [7] addresses these correlations. The next location is predicted based on the previously protected locations. Only a small privacy budget is used to test if the predicted location is close to the next real location. Then, the location is released without using further budget. Otherwise, a standard noise mechanism is used to perturb the next location, spending more budget. While the predictive mechanism significantly reduces the required privacy budget from $n\varepsilon$, necessary to protect a trajectory with standard Geo-Ind

and DP composition, it still causes severe utility degradation for long trajectories (**G4: Utility**). An advantage of these approaches is strong LDP guarantees (ref. Section G1).

Ⓔ **Grid-based Approaches**, such as DPT [33], TGM [26], and Sun et al. [69], first divide the map covered by the dataset into grid cells. Then, the transition probabilities between cells are recorded, and synthetic trajectories are constructed based on these. Grid usage allows blocking certain cells, e.g., based on environmental constraints. However, the accuracy of the generated trajectories depends on the size of the smallest grid cells, and increased granularity yields increased computational effort (ref. Section G4). Moreover, capturing movements within grid cells is not possible. While grid-based approaches offer some appealing properties, they limit the achievable level of utility (**G4: Utility**) [72].

Ⓕ **Environmental Constraints.** Existing methods often overlook environmental constraints (ref. Section G4), resulting in trajectories that ignore geographical features like road networks [5, 14, 52, 54, 69]. Haydari et al. [32] address this shortcoming by incorporating geographical constraints through DP-based map matching. Yet, this approach shares the EM application issue of the aforementioned clustering-based approaches, violating **G1: Formal Privacy Guarantees**. The use of dataset-dependent candidate paths for the EM contradicts the need for dataset-independent outputs (**Pitfall 2**). Cunningham et al. [12] leverage the publicly available road network to enhance the quality of generated locations. However, they produce sets of locations, not trajectories, making trajectory reconstruction from outputs non-trivial. Despite focusing on locations rather than trajectories, we include this work as an example of successful environmental constraint incorporation for DP.

Ⓖ **POI protection.** Cunningham et al. [13] also accounts for environmental constraints, utilising public knowledge to refine outputs. Moreover, this approach offers LDP for enhanced guarantees compared to standard DP. However, it is limited to POI datasets, making it unsuitable for general coordinate trajectories (ref. Section 2.1).

Conclusion. Despite the development of various trajectory protection mechanisms, no optimal solutions exist. Many approaches lack sufficient privacy due to incorrect UoP selection (**G2: Unit of Privacy**) [36], flawed DP proofs (**G1: Formal Privacy Guarantees**) [10, 35, 42], or susceptibility to reconstruction attacks (**G3: Empirical Privacy**) [5, 64]. Additionally, high utility is hard to achieve, with methods releasing only aggregated data [10, 35, 42, 43] or relying on grid cells [26, 33, 69], restricting granularity. These limitations inspired researchers to explore DL-based trajectory generation as a promising alternative [44], discussed in the following.

6 DL-BASED TRAJECTORY GENERATION

In 2018, a vision paper [44] proposed the idea of using a Generative Adversarial Network (GAN) (ref. Section 2.4) to generate synthetic trajectory data. This concept relies on the principle that releasing synthetic data safeguards privacy by distributing generated *fake* data, not real individuals' data. This section explores implementations of the trajGAN concept summarised in Table 3.

❶ *LSTM-TrajGAN.* The most notable implementation of this concept is *LSTM-TrajGAN [62, 63]*, which consists of a trajectory generator and a trajectory discriminator with similar architectures. The generator aims to produce realistic trajectories, while the discriminator seeks to differentiate them from actual trajectories (ref. Section 2.4). Real trajectories are normalised by encoding location offsets from a central point and using one-hot encodings for temporal and semantic attributes. These encoded trajectories are fed into the generator's embedding layer, which embeds each feature individually. A Fully Connected (FC) feature fusion layer fuses these embeddings with a noise vector. A sequence-to-sequence LSTM layer processes the resulting latent representation. Finally, synthetic trajectories are generated using a FC layer for continuous properties, like locations, and softmax layers for categorical properties. The discriminator, mirroring the generator's structure with embedding, feature fusion, LSTM, and output layers, evaluates these synthetic trajectories during training, returning a value in the range $[0, 1]$ for each input. LSTM-TrajGAN employs a specialised loss function, *TrajLoss*, as an alternative to the standard adversarial loss, defined by:

$$\begin{aligned}\text{TrajLoss}(y_r, y_p, t_r, t_s) = \alpha L_{BCE}(y_r, y_p) + \beta L_s(t_r, t_s) \\ + \gamma L_t(t_r, t_s) + c L_c(t_r, t_s)\end{aligned} \tag{3}$$

In this equation, L_{BCE} represents the standard adversarial loss, L_s is a Mean Squared Error (MSE) loss measuring spatial similarity, and L_t and L_c are cross-entropy losses assessing temporal and categorical similarities, respectively.

LSTM-TrajGAN demonstrates remarkable utility and is the first to evaluate against TUL practically, reducing its top-1 accuracy success from 93.8 % to 45.9 %. Despite its important contribution, this approach faces limitations. Primarily, it lacks formal privacy guarantees, violating **G1: Formal Privacy Guarantees**. A practical evaluation is valuable but insufficient (ref. Section G1). LSTM-TrajGAN is robust against the MARC [50] TUL model, but it might be susceptible to other attacks or improved TUL algorithms. Secondly, the model's architecture raises further privacy concerns. The model uses an encoded real trajectory as generation input, with noise only concatenated and intermingled with the remaining input in the FC feature fusion layer. This structure allows the model to learn to disregard the noise to generate more realistic outputs, as detailed in Appendix B theoretically and practically in Appendix C.1. Training LSTM-TrajGAN for the specified 2,000 batches [63] results in differing prediction outputs: about 8 % of predicted hours vary, and the average location distance exceeds 800 m on the FS-NYC dataset (ref. Table 4). Extending training to 10× this amount, which only takes ≈ 25 min on our system (specified in Section 7), leads to 0 differently predicted hours and reduces the location distance to 183 m. This suggests that LSTM-TrajGAN may learn to neglect the input noise with longer training, yielding outputs nearly identical to the input trajectories. Hence, LSTM-TrajGAN's privacy relies heavily on the model not being trained for too long, a decision dependent on the end-user, and not providing robust guarantees. It could be argued that LSTM-TrajGAN's latent space acts as a bottleneck, akin to AE architectures, preventing precise replication of original trajectories. However, quantifying the modification caused by this compression is challenging and should not represent the primary basis for the provided privacy preservation.

Reconstruction Attack. We evaluated the performance of the RAoPT [5] reconstruction attack on samples produced by LSTM-TrajGAN. This attack significantly reduces distances between the

Category	Approach	UoP	G1	G2	G3	G4	G5	Main Shortcoming
LSTM-based	❶ LSTM-TrajGAN [63]	Instance	✗	✓	✓ (TUL) / ✗ (RAoPT)	✓	✓	No privacy guarantees
	❷ Shin2023 [65]	Instance	✗	✓	✓ (TUL)	✓	✓	Inherited from LSTM-TrajGAN
	❸ Ozeki2023 [55]	Instance	✗	✓	∘ (MIA)	✓	✓	Inherited from LSTM-TrajGAN
	❹ Song2023 [67]	Instance	✗	✓	✓ (TUL)	✓	✓	Inherited from LSTM-TrajGAN
	❺ Fontana2023 [24]	Instance	✗	✓	✓ (TUL)	✓	✓	Inherited from LSTM-TrajGAN
	❻ LGAN-DP [77]	Instance	✗	✓	–	∘	✓	Flawed DP proof (**Pitfall 1**)
	❼ DP-TrajGAN [74]	Instance	✗	✓	–	∘	✓	Flawed DP proof (**Pitfall 1**)
AAE-based	❽ Kim2022 [40]	Location		✗	–	∘	✓	UoP; Grid-based
Clustering-based	❾ RNN-DP [10]	Instance	✗	✓	–	✓	✓	Flawed DP proof (**Pitfall 2**)
Two Stage GANs	❿ TSG [72]	Instance	✗	✓	–	✓	✓	No guarantees
	⓫ TS-TrajGEN [38]	Instance	✗	✓	–	✓	✓	No guarantees
Point Generation	⓬ GeoPointGAN [14]	Location		✗	–	✓	✓	Targets location generation

Table 3: Overview of Trajectory Generation Approaches. The table lists the targeted UoP, adherence to design goals from Section 3, and primary shortcomings. Symbols used are ✓ (goal satisfied), ∘ (partially satisfied), and ✗ (unsatisfied). Regarding G1, ✗ denotes known DP proof errors; the absence of a symbol means we are unaware of any flaw. For G3, ✗ indicates demonstrated attacks and approaches with – have not been evaluated against practical attacks.

original and generated trajectories: over 20 % for FS-NYC and between 33 % and 50 % for Geolife. It also improved the overlap of the convex hulls, determined by the Jaccard index, by over 60 % for Geolife and by over 14 % for FS-NYC. For detailed measurements and discussion, see Appendix C.3. These findings underscore the insufficiency of practical guarantees alone, as elaborated above. Although LSTM-TrajGAN effectively protects against the TUL attack, it remains susceptible to other attacks, like RAoPT.

Is LSTM-TrajGAN a GAN? Lastly, we claim that LSTM-TrajGAN operates mainly as a transformative model with minimal use of the discriminator. To verify this, we removed the discriminator feedback from the loss function (L_{BCE} in Equation 3) and evaluated the model with and without a discriminator on the same dataset (FS-NYC) used in [63]. Table 5 in Appendix Section C.2 shows the results. Performance remains similar whether or not the discriminator's feedback is included in the loss. Location quality drops slightly without the discriminator, but categorical properties (e.g., hour) improve. Suppose we remove all components of the loss and train the generator based only on the discriminator's feedback. In that case, the model provides barely any utility, outputting over 85 % different categorical values (compared to 1 %-10 % before), and increasing the location distances more than 20-fold. In other words, the discriminator has little impact on the model's performance.

LSTM-TrajGAN Extensions. Due to the promising results obtained by LSTM-TrajGAN, especially in terms of utility and the openly available source code, several works [24, 65, 67, 74, 77] aim to improve the model. ❷ *Shin et al. [65]* introduced a Trajectory Category Auxiliary Classifier (TCAC) to counteract mode collapse, a common issue in GANs. However, unlike conventional GANs that rely solely on noise input, LSTM-TrajGAN uses real samples as generator inputs, avoiding mode collapse. We observed no mode collapse during our experiments. Thus, the effectiveness of the auxiliary classifier remains unclear, and without access to the source code, we could not verify these claims. Moreover, the approach does not improve on LSTM-TrajGAN's privacy limitations.

❸ The model used by *Ozeki et al. [55]* appears identical to LSTM-TrajGAN, mirroring even parameters and loss function. Their model generates privacy-preserving trajectories for taxi demand prediction. Yet, without any alterations from LSTM-TrajGAN, the same limitations persist in their approach. The authors claim their solution would offer equivalent privacy to DP. However, this statement is based on an evaluation regarding the MIA success rate, which is problematic for two reasons: First, using one existing attack does not allow extending the findings to any attack, as discussed in Section G1. Second, the approach is compared to CNoise [36] and Geomasking, which do not provide trajectory-level DP as discussed in Section 5, Consequently, it is uncertain how their method measures up against an instance-level DP mechanism.

❹ *Song et al. [67]* extend LSTM-TrajGAN by an Except-Condition GAN (exGAN). This approach allows sensitive regions to be excluded from the generated data by specifying certain labels to be excluded. The model is evaluated on the FS-NYC dataset, where it achieves similar utility to LSTM-TrajGAN while reducing the TUL success rate. However, the approach neither adds any privacy guarantees nor addresses the other shortcomings of LSTM-TrajGAN.

❺ *Fontana et al. [24]* combine LSTM-TrajGAN [63] and MARC [50], a model used for assessing the TUL success rate by some LSTM-based approaches [63, 65, 67]. Moreover, they introduce a novel *Next Week Trajectory Prediction* model to evaluate the utility of generated trajectories. This model mostly equals LSTM-TrajGAN's generator. However, instead of generating synthetic trajectories, the model receives one week's trajectories and predicts the following week's trajectories for a given user. This model underlines the adaptability of the LSTM-TrajGAN architecture to other tasks. Moreover, the authors evaluate the models on two datasets, FS-NYC, used by most related works, and the breadcrumbs dataset [53]. This evaluation confirms the effective balance of utility and privacy of LSTM-TrajGAN. However, the work does not enhance the generation process and thus shares LSTM-TrajGAN's privacy limitations.

❻ *LGAN-DP* [77] targets the extension of LSTM-TrajGAN by DP guarantees. First, synthetic trajectories are generated through a simplified version of LSTM-TrajGAN, only considering spatial information. Differential Privacy is not achieved through the actual generation model but through post-processing. In particular,

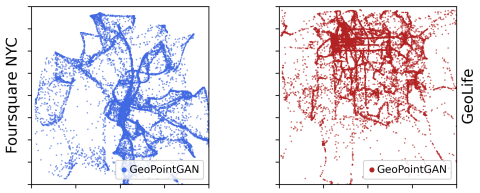

Figure 3: Application of GeoPointGAN [14] to the FS-NYC and Geolife datasets. Compare Figure 5 for baseline.

a combination of the clustering-based approach by Hua et al. [35] and the constrained Laplace noise from [79] is implemented. Both these baselines are discussed in Section 5 and have been critiqued for flawed DP proofs [52]. Without a comprehensive DP proof for LGAN-DP, it likely inherits similar flaws. As thoroughly discussed by Miranda-Pascual et al. [52], DP requires adding Laplace noise to all possible outputs, not only to a restricted number of candidates (**Pitfall 1**). Based on the available information, LGAN-DP appears to construct a candidate set from the cluster centres and only perturbs the corresponding counts, thus failing to meet **G1: Formal Privacy Guarantees**. Applying a DP protection mechanism to LGAN-DP's outputs likely results in lower utility than directly applying such a mechanism to the real dataset, making the benefits of LGAN-DP over the methods from Section 5 unclear. Moreover, the post-processing is expected to significantly reduce utility compared to other generative approaches such as LSTM-TrajGAN. We could not confirm this assumption due to unavailable source code.

❼ *DP-TrajGAN [74]* also targets DP guarantees. Its architecture differs slightly from LSTM-TrajGAN. However, the general flow remains the same, i.e., a real trajectory concatenated with random noise serves as input for the generator. Unlike LSTM-TrajGAN, the real trajectory is perturbed to achieve DP before being fed into the generator. This method could ensure DP for the output via the post-processing property (ref. Section 2.2), provided no further steps access raw data. However, the work states: "But the discriminator takes the real trajectory as input without adding noise because D needs to identify the original trajectory distribution more accurately, thus providing more accurate weight gradients for G" [74]. Since the discriminator's feedback indirectly exposes the generator to unprotected trajectories through gradient updates, this contradicts DP (**Pitfall 1**). Thus, DP-TrajGAN fails to offer robust privacy guarantees, violating **G1: Formal Privacy Guarantees**.

AAE-based Approaches. While the architectures above mirror LSTM-TrajGAN closely, alternative architectures have been proposed. ❽ *Kim and Jang [40]* propose a *AAE-based* architecture for the trajectory generation. First, the real trajectory data is protected with the established location DP mechanism Geo-Indistinguishability (Geo-Ind) [3]. Then, the perturbed data is encoded with a Geo-Indistinguishability (Geo-Ind)-aware location encoding to offset some of the utility loss. This client-side protection offers LDP guarantees, an advantage over the standard DP guarantees of other approaches. The AAE's encoder consists of a many-to-many LSTM followed by a FC layer, while the decoder consists of a FC layer followed by an LSTM. The discriminator of the AAE compares the latent space representation produced by the encoder with the real data distribution. During generation, noise is sampled from a Gaussian distribution and fed into the decoder to generate a synthetic trajectory. Unlike the LSTM-based models, this method does not use real trajectories during generation, preventing privacy leaks. While similar to DP-TrajGAN, the DP protection is achieved during pre-processing, the raw data is not accessed in later steps, ensuring DP guarantees via the post-processing property (ref. Section 2.2).

However, the approach has two key drawbacks. First, the used privacy mechanism Geo-Ind provides location privacy but not trajectory privacy, violating **G2: Unit of Privacy** (ref. Section 5). While the usage of an AAE for trajectory generation adds another layer of privacy on top of Geo-Ind, quantification of the trajectory-level privacy this solution can provide is challenging (conflicting with **G1: Formal Privacy Guarantees**). Second, the approach relies on splitting the geographical area into a grid. The results provided do not clarify the accuracy of the data generated. However, the finest used grid, dividing Beijing into a 20×20 grid, seems coarse. Grid cell usage limits the provided utility (ref. **G4: Utility**) [72].

Clustering-based Approaches. ❾ *Chen et al. [10]* proposed another RNN-based model for trajectory generation. To provide DP guarantees, they post-process the generated trajectories with the trajectory publication mechanism based on clustering proposed by Hua et al. [35]. However, this mechanism cannot provide DP guarantees due to the incorrect application of the EM [52] violating **G1: Formal Privacy Guarantees** (**Pitfall 2**).

Two Stage GANs. ❿ *Wang et al. [72]* propose a Two-Stage-GAN (TSG) for trajectory generation. The first stage GAN consists of a Convolutional Neural Network (CNN), generating a grid representation of a trajectory where each grid cell records the stay duration. The link module transfers this representation into a sequence of grid cells. The result serves as input for the second-stage GAN, which combines it with an encoded road map image to incorporate geographical constraints and outputs coordinate trajectories. Through the two-stage approach, grid cells can be used to generate the overall structure. Still, the second stage allows for generating coordinate trajectories for improved accuracy. While this approach introduces a promising architecture, TSG neither provides any privacy guarantees nor is it evaluated against any practical attacks.

⓫ *TS-TrajGEN [38]* is a second two-stage GAN for trajectories. The first stage generates a region-level (i.e., coarse) trajectory based on recorded origins and destinations. Based on this regional-level trajectory, the second stage creates a more detailed coordinate Trajectory. TS-TrajGEN incorporates the road network through the usage of the A*-Search algorithm. While the approach thoroughly evaluates the utility based on seven metrics, it neither provides privacy guarantees (**G1: Formal Privacy Guarantees**) nor evaluates the achieved level of privacy (**G3: Empirical Privacy**) in any way.

Point Generation. ⓬ *GeoPointGAN (GPG) [14]* represents a generative model for location sets, not trajectories, which we would like to highlight for its impressive and reproducible point-generation capabilities. GPG employs a classic GAN structure: the generator uses Gaussian noise to create synthetic points, and the discriminator classifies points as real or synthetic. For privacy, GPG applies Local Label Differential Privacy (LLDP), a variant of LDP treating only labels as sensitive. Labels indicating whether a location is real or fake are probabilistically flipped to achieve LLDP. Our evaluations show that GPG accurately captures point distributions of

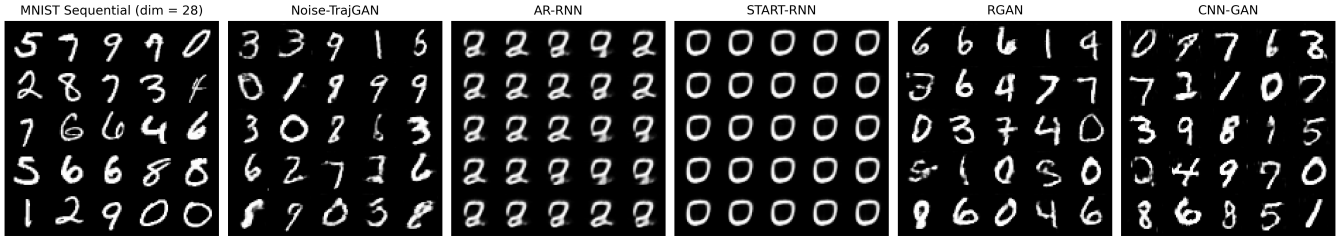

Figure 4: MNIST-Seq dataset generation. All models perform reasonably well on this dataset. The two RNN models generate the same image as each step depends on the previous, but the first row consists only of black pixels.

the considered datasets, as depicted in Figure 3, and consistently yields stable results. However, adapting GPG for trajectory generation is challenging, as we discuss in Section 7. The model provides location-level privacy by design, so transferring to trajectory-level privacy might significantly degrade utility. Moreover, while LLDP yields quantifiable privacy guarantees and some researchers argue that the notion provides sufficient privacy guarantees for the deep learning setting [14, 27], ultimately, a solution providing standard DP guarantees would represent the gold standard.

Conclusion. Several generative models have been published since the first vision paper [44] proposing the usage of GANs for the privacy-preserving generation of trajectory data. However, none of the existing solutions can provide sufficient privacy guarantees yet. The largest group of approaches [24, 38, 55, 63, 65, 67, 72] relies on the obfuscation caused by the generation itself. However, this does not allow quantifying the provided privacy level as discussed in **G1: Formal Privacy Guarantees**. Other approaches cannot provide acceptable guarantees due to flaws in their privacy proofs, either due to privacy leakage in the model design [74] or errors inherited from baseline works [10, 77]. The design of a fully differential private generative model remains an open research question. However, solutions such as GPG [14], which can provide LLDP guarantees for robust spatial point generation, show a promising direction.

7 GENERATIVE MODELS

Current deep learning-based generative models show promise but fail to meet all our design goals (ref. Section 3), with inadequate privacy guarantees as the primary issue (Table 3). The ideal model would offer at least *instance-level (preferable user-level)* **(G2)** *differential privacy* guarantees **(G1)**, while achieving *practical privacy* **(G3)** and *high utility* **(G4)** with *reasonable computational costs* **(G5)**, and is evaluated on *public datasets* with *established metrics*.

DP-SGD (ref. Section 2.3) is the prevalent method for DP in DL [57], offering instance-level DP for training data. Thus, models trained with DP-SGD ensure trajectory-level privacy when each trajectory is a training sample. The fundamental concept aligns with the objectives of trajectory generation. DP-SGD ensures that models capture overall data distribution while minimising individual samples' impact. Likewise, trajectory generators aim at encapsulating dataset traits without specific trajectory retention. This finding motivates the following proposal: (1) Create a generative model independent of input trajectories during prediction, enabling the integration of DP-SGD. As DP-SGD only protects the training dataset, the method cannot be applied to a model like LSTM-TrajGAN requiring input during prediction. (2) Upon achieving this, the next step involved applying DP-SGD [57] to ensure DP for the dataset. In the following, we evaluate six generative models for sequential data to determine their suitability. All source code is made available[2].

Evaluation Setup. We performed all measurements on a server (2x Intel Xeon Silver 4208, 128 GB RAM, Ubuntu 20.04.01 LTS) with 4 NVIDIA Tesla T4 GPUs (16 GB RAM each) using one GPU per experiment. Appendix D lists all used (hyper-)parameters.

Datasets. We use three datasets for our evaluation: MNIST Sequential (MNIST-Seq) [23] as a simple sequential toy dataset to confirm the correctness of our implementations and the two trajectory datasets Geolife [81] and Foursquare NYC [17] (FS-NYC). While FS-NYC contains relatively coarse (restaurant) check-ins, Geolife consists of finer trajectories, 91 % having a sampling rate of smaller or equal 5 s [81]. The diversity between datasets allows for comparing the approaches for various use cases. Results on MNIST-Seq are shown in Figure 4, and on trajectory datasets in Figure 5. We detail the three datasets and their pre-processing below.

MNIST-Seq. The standard MNIST dataset consists of handwritten digit images and serves as a benchmark for image processing. The MNIST-Seq dataset transforms these standard 28×28 images into sequences of length 28 with 28 values each [23], i.e., each row of the images represents one time step. It has been used as a benchmark for sequence generation [23]. Though not a trajectory dataset, we utilise this simple sequential dataset to confirm our models' capacity to correctly generate sequences, ensuring that weak results on trajectory datasets are not due to implementation errors.

FS-NYC. Like related works [50, 63], we use FS-NYC as provided in the LSTM-TrajGAN repository [62] without further pre-processing. This version reduces the size of the original dataset [17] to 66 962 check-ins and 193 users and uses categorical features.

Geolife. The Geolife dataset spans a large area of the globe. Therefore, using it as-is is unsuitable for three reasons: i) Most existing trajectory generation models target city-scale datasets, ii) it makes visualisation very challenging, and iii) dataset cleaning is required for use with DL models. We decided on the following pre-processing steps: (1) All points outside a bounding box defined by the 5th ring of Beijing are removed. (2) All trajectories are re-sampled to a constant sampling rate of one location per 5 s because over 90 % have a finer granularity than that [81]. (3) Trajectories are split on ≥ 60 s breaks as we consider separate trips as distinct trajectories. (4) We truncate all remaining trajectories to 200 points as some approaches require a defined upper length. (5) Finally, all trajectories shorter than 10 points are dropped. Both datasets are normalised to $[0; 1]$

[2] https://github.com/erik-buchholz/SoK-TrajGen

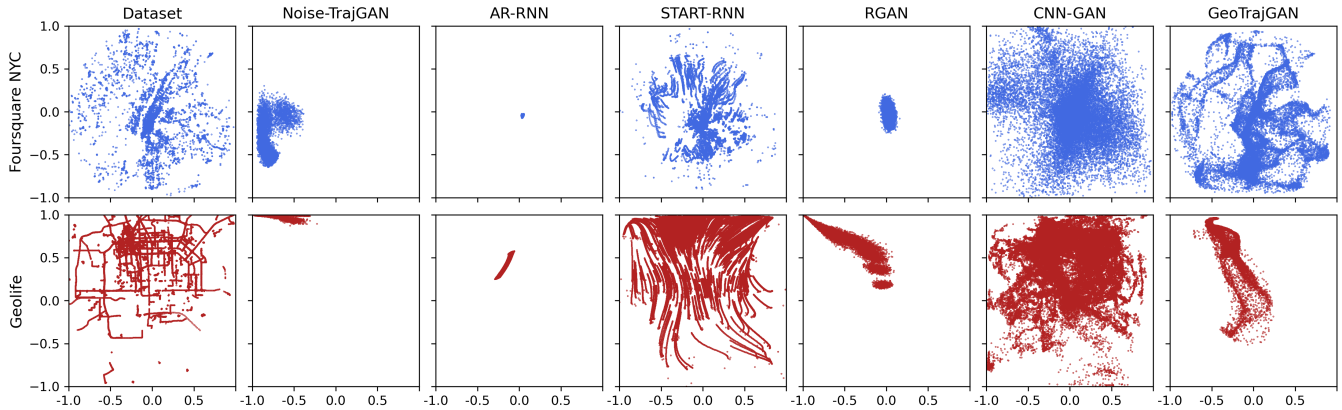

Figure 5: Overview of generative models. No considered sequential generative model can adequately capture the point distribution of the original trajectory datasets FS-NYC and Geolife.

by subtracting a reference point defined as:

$$ref = \left(\frac{\max\{lon\} - \min\{lon\}}{2}, \frac{\max\{lat\} - \min\{lat\}}{2} \right)$$

and division through a scaling factor defined as:

$$sf = (\max\{lon\} - ref_{lon}, \max\{lat\} - ref_{lat})$$

Noise-TrajGAN (NTG). Considering LSTM-TrajGAN's [63] success (ref. Section 6), incorporating DP-SGD in its training process may appear promising. However, LSTM-TrajGAN-based models require real trajectory input for generation, while DP-SGD only protects training samples. Therefore, a mere integration with a DP-SGD library is insufficient. To overcome this limitation, we developed a variant of LSTM-TrajGAN, named *NTG*, adhering to a more traditional GAN structure. The key difference is that the generator only receives a (Gaussian) noise vector instead of trajectory as input. The discriminator remains mostly unmodified. Figure 4 shows that NTG can sequentially generate MNIST digits, although the digits are slightly blurrier than those produced by other models.

However, Figure 5 shows that the model cannot capture the point distribution of trajectory datasets. Despite over 20 trials with varied loss functions (like WGAN-GP and modified trajLoss), noise shapes, optimizers, learning rates, and RNN layers, no notable improvements emerged. Based on this finding, we examined LSTM-TrajGAN in more detail, described in Section 6, and found that the discriminator minimally impacts the model's performance. Accordingly, solely relying on the discriminator's feedback does not appear promising. This insight directed us to explore different architectures.

Simple RNN Models. RNN-based models have been applied successfully to various domains. *LSTM* [34] and *GRU* [11] are the most widely used RNN types. Indeed, most related works (ref. Section 6) incorporate RNNs in their models. We evaluated multiple simple RNN-based models, focusing on *Autoregressive RNN (AR-RNN)* and *Start-Point RNN (START-RNN)* in the following.

The initial RNN cell of the *AR-RNN* model receives Gaussian noise and produces a latent encoding, which is processed by an output layer into the desired output format, e.g., (lat, lon). This serves as input to the next cell alongside the previous hidden (and cell) state. During training, real values replace each cell's output after every step to prevent error propagation. This results in high-quality images generated during training for MNIST-Seq. Yet, in the prediction phase, AR-RNN consistently yields identical outputs (ref. Figure 4). The initial row of black pixels, shared by all digits, predetermines subsequent rows, yielding near-identical output. Moreover, Figure 5 reveals the model's failure to represent point distributions, possibly because initial noise lacks placement information, causing clustering around the dataset's centre of gravity.

Inspired by these findings, we developed *START-RNN*, using a dataset's real starting point for guidance. As this represents a privacy leakage, a final solution would have to generate start points privately, e.g., using GPG (ref. Section 6). On the MNIST-Seq dataset, START-RNN still only produces identical digits because the digits cannot be distinguished based on the first row. Trajectory dataset outputs improve, but Figure 5 shows challenges in capturing details like road networks. Consequently, we consider more sophisticated generative models that are successful in other domains.

RGAN [23]. The *Recurrent GAN (RGAN)* [23] represented the first application of the GAN architecture to real-valued sequences, avoiding discretization drawbacks (ref. Section G4). This model employs a standard GAN setup: A generator creates sequences from noise, and a discriminator differentiates these from real training samples. Both models consist of a LSTM layer followed by a FC output layer. The input shape is $(batch\_size, sequence\_len, noise\_dim)$, with $noise\_dim$ set to 5 in [23]. We reproduced the results on MNIST-Seq but observed mode collapse with the original hyper-parameters, yielding only some of the 10 digits. Applying WGAN-LP [56] improved results in Figure 4. Nevertheless, RGAN failed on the trajectory datasets (ref. Figure 5). Though effective for sequential data like MNIST-Seq and medical signals, RGAN fails to capture spatial point distributions. The generated point distributions vary greatly between gradient updates, indicating unstable training. Multiple practitioners report difficulties in training GANs containing RNN layers and report better results using Conv1D layers instead [49].

CNN-GAN. One notable example is WaveGAN [18], which proposes a Conv1D-based GAN for the generation of audio signals and performs better than compared RNN-based models. WaveGAN represents an adaption of the popular and robust [28] DCGAN [61] architecture to the 1D setting. In particular, the 5×5 convolutional filters are flattened to Conv1D filters with kernel size 25 and both stride and upsampling are increased from 2 to 4 [18]. As WaveGAN produces a minimum sequence length of 16 384, we modified the

model to accommodate shorter sequences required by MNIST-Seq (length 28) and our trajectory datasets (lengths up to 200). We retained a kernel size of 25 to enable the model to consider multiple past and future points during generation. To reduce upsampling, we reduced the stride to 1. We calculate the initial FC layer's output size based on the dataset's maximum sequence length. The resulting model, trained with the WGAN-LP loss, achieves results comparable to RGAN on MNIST-Seq. Although the generated points' distribution is much closer to the dataset's and the model trains more stable than previous models, the CNN-GAN remains unable to replicate the datasets' detailed spatial distributions (ref. Figure 5). However, the improved results indicate that Conv1D layers are more suitable for capturing point distributions than RNNs.

GeoTrajGAN. A prevailing pattern of the previous experiments is the models' inability to capture the datasets' location distribution. Therefore, we considered GPG [14], discussed in Section 6, which can generate detailed point distributions (ref. Figure 3) and intended to extend it to a trajectory generation model called *GeoTrajGAN (GTG)*. The first challenge relates to the selection of batches. Originally, GPG randomly samples 7 500 points per batch, so each batch resembles the dataset's distribution. We found that using randomly sampled trajectories and concatenating their points already impairs the model's ability to represent point distributions accurately. We attributed this to the use of batch norm, as the trajectory batches provide less diversity than point batches because locations within one trajectory are correlated. Replacing batch norm by layer norm solved this problem but required reducing the learning rate from 4E-5 to 1E-6 and adding a *STN* to the discriminator. In contrast, GPG only uses an STN in the generator. These changes increase the discriminator's parameter count 12.9×, slowing down training. GTG takes ≈ 500 epochs to reach a comparable state to GPG after 100 epochs. Moreover, the street network details are not as sharp when using layer norm, but most roads remain visible. However, the resulting model still produced points only. To produce trajectories, we tried several modifications, achieving the best results with a bidirectional LSTM in the generator and two separate discriminators: one similar to GPG's (with layer norm adjustments) and a new LSTM-based discriminator for assessing the sequential quality.

Figure 5 shows that GTG somewhat captures FS-NYC's point distribution but still fails on Geolife despite its larger size and longer training. Moreover, the generated distributions do not reach the detail captured by the original GPG model (ref. Figure 3). While being the most promising of all evaluated models, the GTG approach requires substantial future work to be suitable for privacy-preserving trajectory generation. Nevertheless, the findings highlight that an ensemble model consisting of a model specialised in capturing the distribution of locations and another model capturing sequential properties appears to be a promising direction for future research.

Discussion. Our evaluations highlight that applying sequential GAN models from other domains to trajectory datasets is challenging. We identified the application of DP-SGD as a promising methodology for generating trajectories with rigorous privacy guarantees. Relying on this well-researched algorithm could solve the privacy limitations of existing generative trajectory models (ref. Section 6). However, to enable the application of DP-SGD, developing a generative model that does not access the real dataset during generation is essential. We are unaware of an existing model achieving this for trajectories with a continuous location domain. Moreover, we empirically evaluated multiple generative models targeting continuous sequential data but found none could capture the distribution of locations in a trajectory dataset. The design of an ensemble model combining one part for point distribution and another for sequential properties presents a promising direction for future research. Additionally, our experiments indicated that Conv1D layers perform better than RNN-based models for generating trajectories.

Our experiments' main takeaway is that developing a generative model for trajectory datasets with strong semantic privacy guarantees remains an important open research question. Concrete next steps for future work could be: (1) Developing a generative model for trajectory data without real data access during generation. Such a model could be based on a model ensemble favouring Conv1D over RNN layers. (2) Integrating this model with the DP-SGD algorithm to provide trajectory-level privacy guarantees. As DP-SGD introduces noise to the training process, further model adaptions might be required. (3) Considering alternative approaches like label DP, as used by GPG, if the impact of DP-SGD on utility is too detrimental. (4) Special purpose solutions for POI and discrete trajectories could offer improved utility under full privacy guarantees for certain use cases. These developments should be guided by the design framework proposed in Section 3. With these suggestions for future work, we conclude in the following section.

8 CONCLUSION

This study examined the current state-of-the-art privacy-preserving trajectory dataset generation, focusing on deep learning techniques. We proposed a framework comprised of five design goals to guide the future development of generative trajectory approaches. We emphasised the "Unit of Privacy", which is frequently overlooked in current methods, and proposed a novel systematisation of utility metrics. Our analysis of current trajectory protection methods revealed a lack of robust solutions, with multiple works grounded in flawed differential privacy proofs. The influence of these flawed proofs on subsequent research highlights the importance of diligent assessment of such guarantees. Deep learning-driven generative models reveal their potential as an alternative to protection mechanisms, especially regarding utility. Yet, they fall short of delivering robust trajectory-level differential privacy guarantees. Furthermore, our extensive experimental study suggests that generative models designed for other sequential domains are not readily transferable to trajectory datasets. The main outcome of this work is that the design of a generative deep-learning model providing DP guarantees represents a compelling open research question. We identify a GAN-based model trained with DP-SGD as a promising research direction that we intend to pursue as future work.

ACKNOWLEDGMENTS

The authors would like to thank UNSW, the Commonwealth of Australia, and the Cybersecurity Cooperative Research Centre Limited for their support of this work. The authors thank all the anonymous reviewers for their valuable feedback.

REFERENCES

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep Learning with Differential Privacy. In *Proc. 2016 ACM SIGSAC Conf. Comput. Commun. Secur. (CCS '16)*. Association for Computing Machinery, New York, NY, USA, 308–318. https://doi.org/10.1145/2976749.2978318

[2] Osman Abul, Francesco Bonchi, and Mirco Nanni. 2008. Never Walk Alone: Uncertainty for Anonymity in Moving Objects Databases. In *2008 IEEE 24th Int. Conf. Data Eng.*, Vol. 00. IEEE, Cancun, Mexico, 376–385. https://doi.org/10.1109/ICDE.2008.4497446

[3] Miguel E. Andrés, Nicolás E. Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. 2013. Geoindistinguish-Abilty: Differential Privacy for Location-Based Systems. In *Proc. ACM Conf. Comput. Commun. Secur. (CCS '13, Vol. 13)*. Association for Computing Machinery, Berlin, Germany, 901–914. https://doi.org/10.1145/2508859.2516735

[4] Dor Bank, Noam Koenigstein, and Raja Giryes. 2021. Autoencoders. http://arxiv.org/abs/2003.05991

[5] Erik Buchholz, Alsharif Abuadbba, Shuo Wang, Surya Nepal, and Salil Subhash Kanhere. 2022. Reconstruction Attack on Differential Private Trajectory Protection Mechanisms. In *Proc. 38th Annu. Comput. Secur. Appl. Conf. (ACSAC '22)*. Association for Computing Machinery, New York, NY, USA, 279–292. https://doi.org/10.1145/3564625.3564628

[6] Erik Buchholz, Sharif Abuadbba, Shuo Wang, Surya Nepal, and Salil S. Kanhere. 2022. Reconstruction Attack on Protected Trajectories (RAoPT). https://github.com/erik-buchholz/RAoPT

[7] Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Marco Stronati. 2014. A Predictive Differentially-Private Mechanism for Mobility Traces. In *Privacy Enhancing Technologies*. Springer International Publishing, Cham, 21–41. https://doi.org/10.1007/978-3-319-08506-7_2

[8] Rui Chen, Benjamin C. M. Fung, and Bipin C. Desai. 2011. Differentially Private Trajectory Data Publication. *arXiv* abs/1112.2 (Dec. 2011), 1–12. https://arxiv.org/abs/1112.2020

[9] Rui Chen, Gergely Acs, and Claude Castelluccia. 2012. Differentially Private Sequential Data Publication via Variable-Length n-Grams. In *Proc. 2012 ACM Conf. Comput. Commun. Secur.*, Vol. 2012. Association for Computing Machinery, New York, NY, USA, 638–649. https://doi.org/10.1145/2382196.2382263

[10] Si Chen, Anmin Fu, Jian Shen, Shui Yu, Huaqun Wang, and Huaijiang Sun. 2020. RNN-DP: A New Differential Privacy Scheme Base on Recurrent Neural Network for Dynamic Trajectory Privacy Protection. *J. Netw. Comput. Appl.* 168, February (2020), 102736. https://doi.org/10.1016/j.jnca.2020.102736

[11] Kyunghyun Cho, Bart Van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations Using RNN Encoder–Decoder for Statistical Machine Translation. In *Proc. 2014 Conf. Empir. Methods Nat. Lang. Process. EMNLP*. Association for Computational Linguistics, Doha, Qatar, 1724–1734. https://doi.org/10.3115/v1/D14-1179

[12] Teddy Cunningham, Graham Cormode, and Hakan Ferhatosmanoglu. 2021. Privacy-Preserving Synthetic Location Data in the Real World. In *17th Int. Symp. Spat. Temporal Databases (SSTD '21)*. Association for Computing Machinery, New York, NY, USA, 23–33. https://doi.org/10.1145/3469830.3470893

[13] Teddy Cunningham, Graham Cormode, Hakan Ferhatosmanoglu, and Divesh Srivastava. 2021. Real-World Trajectory Sharing with Local Differential Privacy. *Proc. VLDB Endow.* 14, 11 (July 2021), 2283–2295. https://doi.org/10.14778/3476249.3476280

[14] Teddy Cunningham, Konstantin Klemmer, Hongkai Wen, and Hakan Ferhatosmanoglu. 2022. GeoPointGAN: Synthetic Spatial Data with Local Label Differential Privacy. https://doi.org/10.48550/arXiv.2205.08886

[15] Emiliano De Cristofaro. 2020. An Overview of Privacy in Machine Learning. http://arxiv.org/abs/2005.08679

[16] Yves-Alexandre de Montjoye, César A. Hidalgo, Michel Verleysen, and Vincent D. Blondel. 2013. Unique in the Crowd: The Privacy Bounds of Human Mobility. *Sci. Rep.* 3, 1 (Dec. 2013), 1–5. https://doi.org/10.1038/srep01376

[17] Dingqi Yang, Daqing Zhang, Vincent W. Zheng, and Zhiyong Yu. 2015. Modeling User Activity Preference by Leveraging User Spatial Temporal Characteristics in LBSNs. *IEEE Trans. Syst. Man Cybern, Syst.* 45, 1 (Jan. 2015), 129–142. https://doi.org/10.1109/TSMC.2014.2327053

[18] Chris Donahue, Julian McAuley, and Miller Puckette. 2019. Adversarial Audio Synthesis. http://arxiv.org/abs/1802.04208

[19] Cynthia Dwork. 2008. Differential Privacy: A Survey of Results. In *Theory and Applications of Models of Computation*, Manindra Agrawal, Dingzhu Du, Zhenhua Duan, and Angsheng Li (Eds.). Vol. 4978 LNCS. Springer Berlin Heidelberg, Berlin, Heidelberg, 1–19. https://doi.org/10.1007/978-3-540-79228-4_1

[20] Cynthia Dwork and Aaron Roth. 2013. The Algorithmic Foundations of Differential Privacy. *Found. Trends Theor. Comput. Sci.* 9, 3-4 (2013), 211–407. https://doi.org/10.1561/0400000042

[21] Peter Eigenschink, Thomas Reutterer, Stefan Vamosi, Ralf Vamosi, Chang Sun, and Klaudius Kalcher. 2023. Deep Generative Models for Synthetic Sequential Data: A Survey. *IEEE Access* 11 (2023), 47304–47320. https://doi.org/10.1109/ACCESS.2023.3275134

[22] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response. In *Proc. 2014 ACM SIGSAC Conf. Comput. Commun. Secur.* ACM, New York, USA, 1054–1067. https://doi.org/10.1145/2660267.2660348

[23] Cristóbal Esteban, Stephanie L. Hyland, and Gunnar Rätsch. 2017. Real-Valued (Medical) Time Series Generation with Recurrent Conditional GANs. https://doi.org/10.48550/arXiv.1706.02633

[24] Ivan Fontana, Marc Langheinrich, and Martin Gjoreski. 2023. GANs for Privacy-Aware Mobility Modeling. *IEEE Access* 11 (2023), 29250–29262. https://doi.org/10.1109/ACCESS.2023.3260981

[25] Lorenzo Franceschi-Bicchierai. 2015. Redditor Cracks Anonymous Data Trove to Pinpoint Muslim Cab Drivers. https://mashable.com/archive/redditor-muslim-cab-drivers

[26] Soheila Ghane, Lars Kulik, and Kotagiri Ramamohanarao. 2020. TGM: A Generative Mechanism for Publishing Trajectories With Differential Privacy. *IEEE Internet Things J.* 7, 4 (April 2020), 2611–2621. https://doi.org/10.1109/JIOT.2019.2943719

[27] Badih Ghazi, Noah Golowich, Ravi Kumar, Pasin Manurangsi, and Chiyuan Zhang. 2021. Deep Learning with Label Differential Privacy. https://doi.org/10.48550/ARXIV.2102.06062

[28] Ian Goodfellow. 2017. NIPS 2016 Tutorial: Generative Adversarial Networks. http://arxiv.org/abs/1701.00160

[29] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Networks. *Commun. ACM* 63, 11 (2014), 139–144. https://doi.org/10.1145/3422622

[30] Patricia Guerra-Balboa, Alex Miranda Pascual, Javier Parra-Arnau, Jordi Forne, and Thorsten Strufe. 2022. Anonymizing Trajectory Data: Limitations and Opportunities. https://aaai-ppai22.github.io/files/25.pdf

[31] Mehmet Emre Gursoy, Ling Liu, Stacey Truex, Lei Yu, and Wenqi Wei. 2018. Utility-Aware Synthesis of Differentially Private and Attack-Resilient Location Traces. In *Proc. 2018 ACM SIGSAC Conf. Comput. Commun. Secur. (CCS '18, Vol. 2018)*. Association for Computing Machinery, New York, NY, USA, 196–211. https://doi.org/10.1145/3243734.3243741

[32] Ammar Haydari, Chen-Nee Chuah, Michael Zhang, Jane Macfarlane, and Sean Peisert. 2022. Differentially Private Map Matching for Mobility Trajectories. In *Proc. 38th Annu. Comput. Secur. Appl. Conf. (ACSAC '22)*. Association for Computing Machinery, New York, NY, USA, 293–303. https://doi.org/10.1145/3564625.3567974

[33] Xi He, Graham Cormode, Ashwin Machanavajjhala, Cecilia M. Procopiuc, and Divesh Srivastava. 2015. DPT: Differentially Private Trajectory Synthesis Using Hierarchical Reference Systems. *Proc. VLDB Endow.* 8, 11 (July 2015), 1154–1165. https://doi.org/10.14778/2809974.2809978

[34] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (Nov. 1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[35] Jingyu Hua, Yue Gao, and Sheng Zhong. 2015. Differentially Private Publication of General Time-Serial Trajectory Data. In *2015 IEEE Conf. Comput. Commun. INFOCOM*, Vol. 26. IEEE, Hong Kong, China, 549–557. https://doi.org/10.1109/INFOCOM.2015.7218422

[36] Kaifeng Jiang, Dongxu Shao, Stéphane Bressan, Thomas Kister, and Kian-Lee Tan. 2013. Publishing Trajectories with Differential Privacy Guarantees. In *Proc. 25th Int. Conf. Sci. Stat. Database Manag. - SSDBM*. ACM Press, New York, New York, USA, 1. https://doi.org/10.1145/2484838.2484846

[37] Weiwei Jiang and Jiayun Luo. 2022. Graph Neural Network for Traffic Forecasting: A Survey. *Expert Systems with Applications* 207 (Nov. 2022), 117921. https://doi.org/10.1016/j.eswa.2022.117921

[38] Wenjun Jiang, Wayne Xin Zhao, Jingyuan Wang, and Jiawei Jiang. 2021. Continuous Trajectory Generation Based on Two-Stage GAN. *J. Data Sci.* 19, 1 (2021), 126–141. https://doi.org/10.6339/21-JDS1004

[39] Fengmei Jin, Wen Hua, Matteo Francia, Pingfu Chao, Maria Orlowska, and Xiaofang Zhou. 2021. A Survey and Experimental Study on Privacy-Preserving Trajectory Data Publishing. https://doi.org/10.36227/techrxiv.13655597.v1

[40] Jong Wook Kim and Beakcheol Jang. 2022. Deep Learning-Based Privacy-Preserving Framework for Synthetic Trajectory Generation. *Journal of Network and Computer Applications* 206 (Oct. 2022), 103459. https://doi.org/10.1016/j.jnca.2022.103459

[41] Diederik P. Kingma and Max Welling. 2013. Auto-Encoding Variational Bayes. http://arxiv.org/abs/1312.6114

[42] Meng Li, Liehuang Zhu, Zijian Zhang, and Rixin Xu. 2017. Achieving Differential Privacy of Trajectory Data Publishing in Participatory Sensing. *Inf. Sci.* 400–401 (Aug. 2017), 1–13. https://doi.org/10.1016/j.ins.2017.03.015

[43] Qi Liu, Juan Yu, Jianmin Han, and Xin Yao. 2021. Differentially Private and Utility-Aware Publication of Trajectory Data. *Expert Syst. Appl.* 180, March 2020 (Oct. 2021), 115120. https://doi.org/10.1016/j.eswa.2021.115120

[44] Xi Liu, Hanzhou Chen, and Clio Andris. 2018. trajGANs: Using Generative Adversarial Networks for Geo-Privacy Protection of Trajectory Data (Vision Paper). In *Locat. Priv. Secur. Workshop LoPaS*. github.io, Melbourne, Australia, 1–7. https://ptal-io.github.io/lopas2018/papers/LoPaS2018_Liu.pdf

[45] Lucas May Petry, Camila Leite Da Silva, Andrea Esuli, Chiara Renso, and Vania Bogorny. 2020. MARC: A Robust Method for Multiple-Aspect Trajectory Classification via Space, Time, and Semantic Embeddings. bigdata-ufsc. https://github.com/bigdata-ufsc/petry-2020-marc

[46] Tinghuai Ma and Fagen Song. 2021. A Trajectory Privacy Protection Method Based on Random Sampling Differential Privacy. *ISPRS Int. J. Geo-Inf.* 10, 7 (July 2021), 454. https://doi.org/10.3390/ijgi10070454

[47] Abdul Majeed and Seong Oun Hwang. 2023. Rectification of Syntactic and Semantic Privacy Mechanisms. *IEEE Secur. Privacy* 21, 5 (Sept. 2023), 18–32. https://doi.org/10.1109/MSEC.2022.3188365

[48] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. 2016. Adversarial Autoencoders. http://arxiv.org/abs/1511.05644

[49] martinarjovsky. 2023. Wasserstein GAN. https://github.com/martinarjovsky/WassersteinGAN

[50] Lucas May Petry, Camila Leite Da Silva, Andrea Esuli, Chiara Renso, and Vania Bogorny. 2020. MARC: A Robust Method for Multiple-Aspect Trajectory Classification via Space, Time, and Semantic Embeddings. *Int. J. Geogr. Inf. Sci.* 34, 7 (2020), 1428–1450. https://doi.org/10.1080/13658816.2019.1707835

[51] Frank McSherry and Kunal Talwar. 2007. Mechanism Design via Differential Privacy. In *48th Annu. IEEE Symp. Found. Comput. Sci. FOCS07*. IEEE, Providence, RI, USA, 94–103. https://doi.org/10.1109/FOCS.2007.66

[52] Àlex Miranda-Pascual, Patricia Guerra-Balboa, Javier Parra-Arnau, Jordi Forné, and Thorsten Strufe. 2023. SoK: Differentially Private Publication of Trajectory Data. *Proc. Priv. Enhancing Technol. PoPETs* 2023 (2023), 496–516. https://doi.org/10.56553/popets-2023-0065

[53] Arielle Moro, Vaibhav Kulkarni, Pierre-Adrien Ghiringhelli, Bertil Chapuis, Kévin Huguenin, and Benoît Garbinato. 2019. Breadcrumbs: A Rich Mobility Dataset with Point-of-Interest Annotations. In *Proc. 27th ACM SIGSPATIAL Int. Conf. Adv. Geogr. Inf. Syst.* ACM, Chicago IL USA, 508–511. https://doi.org/10.1145/3347146.3359341

[54] Elham Naghizade, Lars Kulik, Egemen Tanin, and James Bailey. 2020. Privacy- and Context-aware Release of Trajectory Data. *ACM Trans. Spat. Algorithms Syst.* 6, 1 (Feb. 2020), 1–25. https://doi.org/10.1145/3363449

[55] Ren Ozeki, Haruki Yonekura, Hamada Rizk, and Hirozumi Yamaguchi. 2023. Balancing Privacy and Utility of Spatio-Temporal Data for Taxi-Demand Prediction. In *2023 24th IEEE Int. Conf. Mob. Data Manag. MDM*, Vol. 24. IEEE, Los Alamitos, CA, USA, 215–220. https://doi.org/10.1109/MDM58254.2023.00044

[56] Henning Petzka, Asja Fischer, and Denis Lukovnicov. 2018. On the Regularization of Wasserstein GANs. https://doi.org/10.48550/arXiv.1709.08894

[57] Natalia Ponomareva, Hussein Hazimeh, Alex Kurakin, Zheng Xu, Carson Denison, H. Brendan McMahan, Sergei Vassilvitskii, Steve Chien, and Abhradeep Thakurta. 2023. How to DP-fy ML: A Practical Guide to Machine Learning with Differential Privacy. https://doi.org/10.48550/arXiv.2303.00654

[58] Vincent Primault, Antoine Boutet, Sonia Ben Mokhtar, and Lionel Brunie. 2019. The Long Road to Computational Location Privacy: A Survey. *IEEE Commun. Surv. Tutor.* 21, 3 (2019), 2772–2793. https://doi.org/10.1109/COMST.2018.2873950

[59] Vincent Primault, Sonia Ben Mokhtar, Cedric Lauradoux, and Lionel Brunie. 2014. Differentially Private Location Privacy in Practice. https://doi.org/10.48550/arXiv.1410.7744

[60] Youyang Qu, Jingwen Zhang, Ruidong Li, Xiaoning Zhang, Xuemeng Zhai, and Shui Yu. 2020. Generative Adversarial Networks Enhanced Location Privacy in 5G Networks. *Sci. China Inf. Sci.* 63, 12 (Dec. 2020), 220303. https://doi.org/10.1007/s11432-019-2834-x

[61] Alec Radford, Luke Metz, and Soumith Chintala. 2016. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. http://arxiv.org/abs/1511.06434

[62] Jinmeng Rao, Song Gao, Yuhao Kang, and Qunying Huang. 2020. LSTM-TrajGAN. GeoDS Lab @UW-Madison. https://github.com/GeoDS/LSTM-TrajGAN

[63] Jinmeng Rao, Song Gao, Yuhao Kang, and Qunying Huang. 2020. LSTM-TrajGAN: A Deep Learning Approach to Trajectory Privacy Protection. *Leibniz Int. Proc. Inform.* 177, GIScience (2020), 1–16. https://doi.org/10.4230/LIPIcs.GIScience.2021.I.12

[64] Minglai Shao, Jianxin Li, Qiben Yan, Feng Chen, Hongyi Huang, and Xunxun Chen. 2020. Structured Sparsity Model Based Trajectory Tracking Using Private Location Data Release. *IEEE Trans. Dependable Secure Comput.* 18, 6 (2020), 2983–2995. https://doi.org/10.1109/TDSC.2020.2972334

[65] Jihwan Shin, Yeji Song, Jinhyun Ahn, Taewhi Lee, and Dong-Hyuk Im. 2023. TCAC-GAN: Synthetic Trajectory Generation Model Using Auxiliary Classifier Generative Adversarial Networks for Improved Protection of Trajectory Data. In *2023 IEEE Int. Conf. Big Data Smart Comput. BigComp*. IEEE, Jeju, Republic of Korea, 314–315. https://doi.org/10.1109/BigComp57234.2023.00063

[66] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership Inference Attacks Against Machine Learning Models. In *2017 IEEE Symp. Secur. Priv. SP*. IEEE, San Jose, CA, USA, 3–18. https://doi.org/10.1109/SP.2017.41

[67] Yeji Song, Jihwan Shin, Jinhyun Ahn, Taewhi Lee, and Dong-Hyuk Im. 2023. Except-Condition Generative Adversarial Network for Generating Trajectory Data. In *Database Expert Syst. Appl. (Lecture Notes in Computer Science)*, Christine Strauss, Toshiyuki Amagasa, Gabriele Kotsis, A. Min Tjoa, and Ismail Khalil (Eds.). Springer Nature Switzerland, Cham, 289–294. https://doi.org/10.1007/978-3-031-39821-6_23

[68] Han Su, Shuncheng Liu, Bolong Zheng, Xiaofang Zhou, and Kai Zheng. 2020. A Survey of Trajectory Distance Measures and Performance Evaluation. *The VLDB Journal* 29, 1 (Jan. 2020), 3–32. https://doi.org/10.1007/s00778-019-00574-9

[69] Xinyue Sun, Qingqing Ye, Haibo Hu, Yuandong Wang, Kai Huang, Tianyu Wo, and Jie Xu. 2023. Synthesizing Realistic Trajectory Data With Differential Privacy. *IEEE Trans. Intell. Transport. Syst.* 24, 5 (May 2023), 5502–5515. https://doi.org/10.1109/TITS.2023.3241290

[70] Latanya Sweeney. 2002. Achieving K-Anonymity Privacy Protection Using Generalization and Suppression. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.* 10, 05 (Oct. 2002), 571–588. https://doi.org/10.1142/S021848850200165X

[71] Zhen Tu, Kai Zhao, Fengli Xu, Yong Li, Li Su, and Depeng Jin. 2019. Protecting Trajectory From Semantic Attack Considering K-Anonymity, l-Diversity, and t-Closeness. *IEEE Trans. Netw. Serv. Manag.* 16, 1 (March 2019), 264–278. https://doi.org/10.1109/TNSM.2018.2877790

[72] Xingrui Wang, Xinyu Liu, Ziteng Lu, and Hanfang Yang. 2021. Large Scale GPS Trajectory Generation Using Map Based on Two Stage GAN. *J. Data Sci.* 19, 1 (Feb. 2021), 126–141. https://doi.org/10.6339/21-JDS1004

[73] Shuilian Yuan, Dechang Pi, Xiaodong Zhao, and Meng Xu. 2021. Differential Privacy Trajectory Data Protection Scheme Based on R-tree. *Expert Systems with Applications* 182 (Nov. 2021), 115215. https://doi.org/10.1016/j.eswa.2021.115215

[74] Jing Zhang, Qihan Huang, Yirui Huang, Qian Ding, and Pei-Wei Tsai. 2022. DP-TrajGAN: A Privacy-Aware Trajectory Generation Model with Differential Privacy. *Future Gener. Comput. Syst.* 142, C (Dec. 2022), 25–40. https://doi.org/10.1016/j.future.2022.12.027

[75] Jing Zhang, Yi-rui Huang, Qi-han Huang, Yan-zi Li, and Xiu-cai Ye. 2023. Hasse Sensitivity Level: A Sensitivity-Aware Trajectory Privacy-Enhanced Framework with Reinforcement Learning. *Future Generation Computer Systems* 142 (May 2023), 301–313. https://doi.org/10.1016/j.future.2023.01.008

[76] Weiqi Zhang, Zhenzhen Xie, Akshita Maradapu Vera Venkata Sai, Qasim Zia, Zaobo He, and Guisheng Yin. 2024. A Local Differential Privacy Trajectory Protection Method Based on Temporal and Spatial Restrictions for Staying Detection. *Tsinghua Sci. Technol.* 29, 2 (April 2024), 617–633. https://doi.org/10.26599/TST.2023.9010072

[77] Ziming Zhang, Xiaolong Xu, and Fu Xiao. 2022. LGAN-DP: A Novel Differential Private Publication Mechanism of Trajectory Data. *Future Gener. Comput. Syst.* 141, C (Dec. 2022), 6392–703. https://doi.org/10.1016/j.future.2022.12.011

[78] Jianzhe Zhao, Jie Mei, Stan Matwin, Yukai Su, and Yuancheng Yang. 2021. Risk-Aware Individual Trajectory Data Publishing With Differential Privacy. *IEEE Access* 9 (2021), 7421–7438. https://doi.org/10.1109/ACCESS.2020.3048394

[79] Xiaodong Zhao, Yulan Dong, and Dechang Pi. 2019. Novel Trajectory Data Publishing Method under Differential Privacy. *Expert Systems with Applications* 138 (Dec. 2019), 112791. https://doi.org/10.1016/j.eswa.2019.07.008

[80] Xiaodong Zhao, Dechang Pi, and Junfu Chen. 2020. Novel Trajectory Privacy-Preserving Method Based on Prefix Tree Using Differential Privacy. *Knowledge-Based Systems* 198 (June 2020), 105940. https://doi.org/10.1016/j.knosys.2020.105940

[81] Yu Zheng, Lizhu Zhang, Xing Xie, and Wei-Ying Ma. 2009. Mining Interesting Locations and Travel Sequences from GPS Trajectories. In *Proc. 18th Int. Conf. World Wide Web (WWW '09)*. Association for Computing Machinery, New York, NY, USA, 791–800. https://doi.org/10.1145/1526709.1526816

A GLOSSARY

AAE Adversarial Autoencoder.
AdamW PyTorch's AdamW Optimizer.
AE Autoencoder.
AR-RNN Autoregressive RNN.
BCE Binary Cross Entropy.
CNN Convolutional Neural Network.
DL Deep Learning.
DP Differential Privacy.
DP-SGD Differentially Private Stochastic Gradient Descent, refer Section 2.3.
DTW Dynamic Time Warping.
EM Exponential Mechanism.

Dataset	Epochs	Batches	Haversine Mean [m]	Euclidean Mean	Hour [%]	Day [%]	Category [%]
Foursquare NYC	250	2000	837.74	1.50	7.78	0.01	0.79
Foursquare NYC	2500	20000	182.99	0.35	0.00	0.00	0.01
GeoLife	11	2000	492.06	8.52	3.24	0.01	N/A
GeoLife	110	20000	132.58	2.51	0.01	0.00	N/A
GeoLife Spatial	11	2000	230.02	4.12	N/A	N/A	N/A
GeoLife Spatial	110	20000	92.28	1.57	N/A	N/A	N/A

Table 4: LSTM-TrajGAN [63] Convergence. The generated trajectories converge towards the generation inputs if the model is trained long enough. The table records the mean distances of the predicted values from the original dataset after 2 000 batches as used in the paper [63] and after training 10× as long, i.e., after training for 20 000 batches.

exGAN Except-Condition GAN.
FC Fully Connected layer. Also called *Dense* or *Linear* layer.
FS-NYC Foursquare NYC [17].
GAN Generative Adversarial Network.
Geo-Ind Geo-Indistinguishability.
GPG GeoPointGAN [14].
GTG GeoTrajGAN.
HD Hausdorff Distance.
JSD Jensen Shannon Distance.
LDP Local Differential Privacy.
LLDP Local Label Differential Privacy.
LSTM Long Short-Term Memory.
MI Mutual Information.
MIA Membership Interference Attack.
MNIST-Seq MNIST Sequential Dataset: Images are transformed to sequences of length 28 with 28 features each [23].
MSE Mean Squared Error.
NTG Noise-TrajGAN.
POI Point of Interest.
RAoPT Reconstruction Attack on Protected Trajectories.
RGAN Recurrent GAN.
RNN Recurrent Neural Network.
SDD Sampling Distance and Direction.
START-RNN Start-Point RNN.
STC Spatial-Temporal-Categorical Distance.
STN Large Spatial Transformer Network.
TCAC Trajectory Category Auxiliary Classifier.
TSG Two-Stage-GAN.
TUL Trajectory User Linking.
UoP Unit of Privacy.
VAE Variational Autoencoder.
WD Wasserstein Distance.
WGAN-LP WGAN with Lipschitz Penalty [56].

B MINIMAL EXAMPLE TRAJLOSS

In Section 6, we describe LSTM-TrajGAN's [63] potential to ignore the input noise. This is due to noise being added by concatenation, followed by a FC layer. We demonstrate this with a simplified example:

(1) x is the encoded input of shape $batch\_size \times len \times dim$
(2) n is the random noise of shape $batch\_size \times noise\_dim$
(3) Generator $G(x, n)$: $G : \mathbb{R}^{len \times (dim + noise\_dim)} \rightarrow \mathbb{R}^{len \times dim}$
(4) Discriminator $D(x)$: $D : \mathbb{R}^{len \times dim} \rightarrow [0, 1]$

The noise vector n is repeated for all time steps len to map the shape $batch\_size \times noise\_dim$ to $batch\_size \times len \times nosie\_dim$. LSTM-TrajGAN uses *TrajLoss* as loss function [63]:

$$\begin{aligned}\text{TrajLoss}(y_r, y_p, t_r, t_s) = \alpha L_{BCE}(y_r, y_p) + \beta L_s(t_r, t_s) \\ + \gamma L_t(t_r, t_s) + c L_c(t_r, t_s)\end{aligned} \tag{4}$$

For generator training, y_r is 1, $y_p = D(G(x, n))$, t_r is the real sample ($t_r = x$), and t_s is the generated sample ($t_s = G(x, n)$). The generator's goal is to minimise this loss.

$$\underset{G(x,n)}{\text{argmin}} \quad \text{TrajLoss}(1, D(G(x, n)), x, G(x, n)) \tag{5}$$

$$= \underset{G(x,n)}{\text{argmin}} \quad \alpha L_{BCE}(1, D(G(x, n)) + \beta L_s(x, G(x, n)) + \gamma L_t(x, G(x, n)) + c L_c((x, G(x, n)) \tag{6}$$

The spatial loss L_s is the MSE, computed as $MSE(x, G(x, n))$ for generator training. This part of the loss is minimised for $G(x, n) = x$, as $MSE(x, x) = 0$. Temporal similarity L_t and L_c are softmax cross entropies, also minimised for $G(x, n) = x$. With an optimally trained discriminator, the adversarial loss L_{BCE} is minimised for $G(x, n) = x$, as in this case, the real and generated samples are identical. This leads in:

$$\underset{G(x,n)}{\text{argmin}}[\text{TrajLoss}(1, D(G(x, n)), x, G(x, n))] = x \tag{7}$$

The generator may learn to ignore the noise with sufficient training. The model employs a FC fusion layer F, merging encoding x with noise n: $F(x, n) = W(x||n) + b$. This formula simplifies to $F(x, n) = W_x x + W_n n + b$. As x and n are only concatenated, we can simplify this to $F(x, n) = W_x x + W_n n + b$. With prolonged training, if the loss is minimal for $F(x, n) = x$, weights in W_n related to n will tend towards zero. Although LSTM-TrajGAN's generator G is more complex than F, the key fusion component in G operates similarly to F. Given that the fusion layer is biased towards ignoring the noise, we infer this issue could extend to LSTM-TrajGAN's more intricate structure. This is concerning for privacy, as it suggests that the model's noise input does not ensure the output trajectories differ significantly from the input trajectories. Especially with a large latent space, the model could end up reproducing the real trajectories it was fed instead of creating distinct synthetic ones. We demonstrate this issue practically with experiments in Appendix C.1.

Dataset	Loss	Haversine Mean [m]	Euclidean Mean	Hour [%]	Day [%]	Category [%]
Foursquare NYC	STD	762.65	1.35	9.05	0.02	0.78
Foursquare NYC	NO BCE	788.88	1.36	5.56	0.00	1.42
Foursquare NYC	BCE ONLY	14973.26	26.09	95.71	86.82	90.90
GeoLife	STD	451.00	7.82	2.49	0.01	N/A
GeoLife	NO BCE	440.32	7.54	2.70	0.01	N/A
GeoLife	BCE ONLY	9090.44	163.93	94.13	85.80	N/A
GeoLife Spatial	STD	191.21	3.64	N/A	N/A	N/A
GeoLife Spatial	NO BCE	276.48	4.73	N/A	N/A	N/A
GeoLife Spatial	BCE ONLY	6403.59	126.27	N/A	N/A	N/A

Table 5: Evaluation of LSTM-TrajGAN [63] with different losses. The results show that the discriminator has a minimal impact on model performance and that the BCE-loss alone does not yield reasonable utility.

C LSTM-TRAJGAN EVALUATIONS

In Section 6, we report on three evaluations performed with LSTM-TrajGAN [63]. In this section, we describe these measurements in more detail. All experiments described in this section have been recorded with 5-fold cross-validation on the system specified in Section 7. Except for those parameters specified in the following sections, we used the parameters as specified in the respective publications. The FS-NYC and Geolife datasets introduced in Section 2.1 are used for all measurements. The dataset denoted as *Geolife Spatial* is equivalent to the Geolife dataset except that only the spatial information, i.e., latitude and longitude, are used, while temporal information is ignored. Section C.1 describes the evaluation testing whether LSTM-TrajGAN can learn to ignore the input noise. In Section C.2, we evaluate the influence of the discriminator's feedback on LSTM-TrajGAN's training. Finally, Section C.3 evaluates the effectiveness of RAoPT, proposed in [5], on LSTM-TrajGAN.

C.1 LSTM-TrajGAN Convergence

In Section 6, we describe that the LSTM-TrajGAN's architecture does not guarantee that the input noise influences the generated trajectories. This is because the input noise is only concatenated to the input trajectory before being processed by FC feature fusion layer, allowing the model to learn to ignore the noise. We elaborate on this theoretically in Section B. To show this behaviour practically, we performed the following evaluation. First, we trained LSTM-TrajGAN for 2 000 batches, as done in the original paper [63], on the FS-NYC and the Geolife datasets and recorded the difference between the input trajectories of the test set and the generated trajectories. Concretely, we measured the haversine and Euclidean distances between each pair of locations and the percentage of all categorical values that differ between input and output trajectories. Then, we trained the mode for 10× as long, i.e., 20 000 batches and recorded the same values. The results for these measurements are provided in Table 4. If LSTM-TrajGAN is trained for 2 000 batches, the results show a clear difference between the generation inputs and the generated trajectories. On the FS-NYC dataset, distances differ by over 800 m on average, and ≈8 % of hour values are different in the output. On the Geolife dataset, the predicted locations have an average distance of ≈500 m if the time is part of the model and 230 m if only the spatial information is considered. After 20 000 batches, the differences between input and output trajectories are significantly reduced. Instead of an average distance of 800 m, the output locations only differ by 180 m, which might be within line of sight for many places. Moreover, nearly all categorical features are identical to the values from the original dataset. The same is true for the Geolife dataset, where the locations are even closer to those from the original dataset, and the categorical features are nearly identical, too. These results highlight that LSTM-TrajGAN provides little privacy if trained for too long, as the generated trajectories closely resemble the trajectories from the original dataset used as generation inputs. Thus, as discussed in Section 6, the privacy provided by the model depends on the number of epochs the model is trained for, which provides limited privacy guarantees.

C.2 LSTM-TrajGAN Loss

In Section 6, we raise the question to what extent the model operates as a GAN [29], and to which extent the modification caused by the model are simply transformative due to the latent space compression of the input. To evaluate this question, we trained the model with three different loss functions and recorded all results in Table 5. First, we used the TrajLoss loss (ref. Equation 3) proposed by the authors [63] and denoted as *standard (STD)* in Table 5. Second, we excluded the discriminator from the model's training process by setting $\alpha = 0$ in Equation 3, i.e., excluding the Binary Cross Entropy (BCE)-loss from the loss computation. This case is denoted as *NO BCE*. Third, we trained the model with the BCE-loss only by setting all other factors in TrajLoss to 0, denoted as *BCE ONLY*. Again, we record the differences between the input and output trajectories of the test set by measuring the locations' mean haversine and Euclidean distance and the percentage of categorical attributes that differ. In this evaluation, lower values are better, as the model tries to generate trajectories that are as close as possible to those used as input (ref. Section C.1).

Table 5 shows that the difference between the models trained with (*STD*) and without (*NO BCE*) the discriminator is very small. While on the FS-NYC and the Geolife Spatial datasets, the location differences are slightly smaller when the model is trained with the discriminator, the model performs even better without the discriminator on the Geolife dataset with categorical features. Moreover, the results show that the discriminator's feedback alone is insufficient for successfully training the model. The mean distance is increased ≈ 20× on all datasets when the model is trained with the BCE-loss

Dataset	Euclidean Improvement [%]	Hausdorff Improvement [%]	Jaccard Improvement [%]
GeoLife Spatial	33.40	50.15	60.51
GeoLife	34.95	45.57	69.15
Foursquare NYC	24.94	19.81	14.30

Table 6: The table shows the percentage reduction of the Euclidean and Hausdorff distances and the increase of the Jaccard Index of the convex hull after applying RAoPT [5] on a dataset generated with LSTM-TrajGAN [63].

only compared to the standard TrajLoss. Likewise, the difference in categorical features increases from 9 % (hour) and ≤ 1 % (day and category) to over 85 % in all cases. These results highlight that LSTM-TrajGAN trains differently than traditional GANs, such as DCGAN [61], and explain why our Noise-TrajGAN (ref. Section 7) performed so poorly on the trajectory datasets.

C.3 RAoPT on LSTM-TrajGAN

Finally, in Section 6, we ask whether LSTM-TrajGAN is susceptible to existing attacks against trajectory protection approaches, such as RAoPT [5]. To evaluate this, we used the authors' RAoPT implementation [6] and ran it on trajectories generated by LSTM-TrajGAN. Concretely, we performed the measurements as follows. For each of the 5 runs, we randomly split the dataset into two equal-sized sets $train_{trajGAN}$ and $test_{trajGAN}$. Then, we trained LSTM-TrajGAN for 2 000 batches, as used in [63], on $train_{trajGAN}$. After completing the training, we generated trajectories based on $test_{trajGAN}$. The resulting set of generated trajectories, $gen_{trajGAN}$, was split in proportion 2 : 1 into the larger set $train_{RAoPT}$, and the smaller $test_{RAoPT}$. While $train_{RAoPT}$ was used for training of the RAoPT model, we performed the final measurements on $test_{RAoPT}$ and show the results in Table 6. As we used the RAoPT implementation without modification, we recorded the same values as used in [5]. RAoPT can reduce both the Euclidean and the Hausdorff distances and increase the Jaccard index of the trajectories' convex hull, which serves as a representation of the overlap of two trajectories' activity spaces [5]. While the reconstruction is not as significant as for those protection mechanisms considered in [5], the attack still reduces the Euclidean distance by over 30 % on both variants of the Geolife dataset, and the Hausdorff distance even by over 45 %. Moreover, on the Geolife dataset, the Jaccard index can be increased by over 60 % through the reconstruction. In summary, while LSTM-TrajGAN seems significantly less susceptible to the attack than some traditional trajectory protection mechanisms, the reconstruction still succeeds to some extent.

D MODEL PARAMETERS

In this section, we record the (hyper-)parameters used for the evaluation of generative models in Section 7, which produce Figures 4 and 5. Note that these parameters are also recorded in the provided code repository[3]. We follow the same order as Section 7. An overview of all (hyper-)parameters is provided in Table 7.

D.1 Noise-TrajGAN

On all datasets, we use a batch size of 10, PyTorch's *AdamW* optimiser with a learning rate of 0.0003 for the discriminator and 0.0001 for the generator, $\beta_1 = 0.5$ and $\beta_2 = 0.999$. We update the discriminator 5× more frequently than the generator. Both the dimensionality of the input noise vector and the LSTM's hidden size are chosen as 100. The discriminator's embedding dimensions are chosen as 28 for MNIST-Seq, 64 for the combined spatial features, and equal to the input size for categorical features. We train the models with the WGAN with Lipschitz Penalty (WGAN-LP) loss. On MNIST-Seq and Geolife, we train for 100 epochs, and on FS-NYC for 300 due to its smaller size. All other (hyper-)parameters are kept as their respective default values.

D.2 AR-RNN and START-RNN

Both RNN-based models use a batch size of 512, the *AdamW* optimiser with a learning rate of 0.001, $\beta_1 = 0.9$, and $\beta_2 = 0.999$ for all datasets. On MNIST-Seq, the models are trained for 30 epochs, on FS-NYC for 300, and on Geolife for 100. The dimensionality of the input noise used by AR-RNN is chosen as the feature dimensionality of the dataset, i.e., 28 for MNIST-Seq and 2 for the trajectory datasets. The initial FC layer uses the output size as the input size. An LSTM with a hidden size of 100 is used for both models. Both models use the MSE as the loss function. All other (hyper-)parameters are kept as their respective default values.

D.3 RGAN

The RGAN uses a batch size of 28, and PyTorch's *AdamW* optimiser with $\beta_1 = 0.5$ and $\beta_2 = 0.999$ for both generator and discriminator. On MNIST-Seq and Geolife, both learning rates are chosen as 0.0001, while the discriminator is trained with a learning rate of 0.0003 on FS-NYC. All cases use the WGAN-LP loss and train the discriminator 5× per generator update. For the input noise, a dimension of 5 is used as in the original paper [23], and the LSTM's hidden size is chosen as 100. On MNIST-Seq, the model is trained for 300 epochs, on FS-NYC for 500, and on Geolife for 100. All other (hyper-)parameters are kept as their respective default values.

D.4 CNN-GAN

Our CNN-gan uses a batch size of 32 on all datasets, PyTorch's *AdamW* optimiser with $\beta_1 = 0.5$ and $\beta_2 = 0.999$ and a learning rate of 0.0003 for the discriminator and 0.0001 for the generator. The generator and discriminator are updated with the same frequency ($n_{critic} = 1$), and WGAN-LP is used as the loss. On MNIST-Seq and Geolife, the model is trained for 300 epochs, and on FS-NYC for 1 000 epochs. For the input noise, a shape of $(batch\_size, 100)$ is used, i.e., the noise has a dimensionality of 100. The model uses batch norm on MNIST-Seq, while no batch norm is used on the trajectory datasets. All other (hyper-)parameters are kept as their respective default values.

[3] https://github.com/erik-buchholz/SoK-TrajGen

Model	Dataset	Epochs	Batch Size	Optimizer	LR Gen	LR Dis	β_1	β_2	Loss	n_{critic}
NTG	**MNIST-Seq**	100	10	AdamW	1E-4	1E-4	0.5	0.999	WGAN-LP	5
NTG	**FS-NYC**	300	10	AdamW	1E-4	3E-4	0.5	0.999	WGAN-LP	5
NTG	**Geolife**	100	10	AdamW	1E-4	3E-4	0.5	0.999	WGAN-LP	5
AR-RNN	**MNIST-Seq**	30	512	AdamW	1E-3	N/A	0.9	0.999	MSE	1
AR-RNN	**FS-NYC**	300	512	AdamW	1E-3	N/A	0.9	0.999	MSE	1
AR-RNN	**Geolife**	100	512	AdamW	1E-3	N/A	0.9	0.999	MSE	1
START-RNN	**MNIST-Seq**	30	512	AdamW	1E-3	N/A	0.9	0.999	MSE	1
START-RNN	**FS-NYC**	300	512	AdamW	1E-3	N/A	0.9	0.999	MSE	1
START-RNN	**Geolife**	100	512	AdamW	1E-3	N/A	0.9	0.999	MSE	1
RGAN	**MNIST-Seq**	300	28	AdamW	1E-4	1E-4	0.5	0.999	WGAN-LP	5
RGAN	**FS-NYC**	500	28	AdamW	1E-4	1E-4	0.5	0.999	WGAN-LP	5
RGAN	**Geolife**	100	28	AdamW	1E-4	1E-4	0.5	0.999	WGAN-LP	5
CNN-GAN	**MNIST-Seq**	300	32	AdamW	1E-4	3E-4	0.5	0.999	WGAN-LP	1
CNN-GAN	**FS-NYC**	1 000	32	AdamW	1E-4	3E-4	0.5	0.999	WGAN-LP	1
CNN-GAN	**Geolife**	300	32	AdamW	1E-4	3E-4	0.5	0.999	WGAN-LP	1
GTG	**FS-NYC**	1 000	256	AdamW	1E-5	1E-5	0.5	0.999	BCE	1
GTG	**Geolife**	1 000	256	AdamW	1E-5	1E-5	0.5	0.999	BCE	1

Table 7: Evaluation Hyperparameters. The table shows the hyperparameters of the models from Section 7 used for the evaluation and Figures 4 and 5.

D.5 GeoTrajGAN

We did not test GTG on MNIST-Seq as we already knew from our experiments with GPG (ref. Section 6), that the baseline model can capture the point distribution of the considered datasets (ref. Figure 3). For both FS-NYC and Geolife, we use a latent dimension of 256 for both the generator and the discriminator. The main architecture differences to GeoPointGAN [14] are the use of layer norm instead of batch norm in both models, an LSTM with hidden size 64 in the generator, and the usage of Large Spatial Transformer Network (STN) in the point-level discriminator. Moreover, we add a sequential discriminator consisting of an LSTM with hidden size 64 followed by two FC layers. The feedback of both discriminators is combined via addition. A standard BCE-loss is used as the loss function. Both the generator and the combined discriminator are trained with the same frequency ($n\_critic = 1$) using the $AdamW$ optimizer with a learning rate of 1E-5, $\beta_1 = 0.5$, $\beta_2 = 0.999$ and a batch size of 256. We train the model on both datasets for 1000 epochs as GTG only produced a reasonable point distribution for FS-NYC despite the larger size of the Geolife dataset (yielding more updates per epoch). All other (hyper-)parameters are kept as their respective default values.