Real-World Trajectory Sharing with Local Differential Privacy

Teddy Cunningham
University of Warwick
Coventry, United Kingdom
teddy.cunningham@warwick.ac.uk

Graham Cormode
University of Warwick
Coventry, United Kingdom
g.cormode@warwick.ac.uk

Hakan Ferhatosmanoglu
University of Warwick
Coventry, United Kingdom
hakan.f@warwick.ac.uk

Divesh Srivastava
AT&T Chief Data Office
Bedminster, NJ, USA
divesh@att.com

ABSTRACT

Sharing trajectories is beneficial for many real-world applications, such as managing disease spread through contact tracing and tailoring public services to a population's travel patterns. However, public concern over privacy and data protection has limited the extent to which this data is shared. Local differential privacy enables data sharing in which users share a perturbed version of their data, but existing mechanisms fail to incorporate user-independent public knowledge (e.g., business locations and opening times, public transport schedules, geo-located tweets). This limitation makes mechanisms too restrictive, gives unrealistic outputs, and ultimately leads to low practical utility. To address these concerns, we propose a local differentially private mechanism that is based on perturbing hierarchically-structured, overlapping *n*-grams (i.e., contiguous subsequences of length *n*) of trajectory data. Our mechanism uses a multi-dimensional hierarchy over publicly available external knowledge of real-world places of interest to improve the realism and utility of the perturbed, shared trajectories. Importantly, including real-world public data does not negatively affect privacy or efficiency. Our experiments, using real-world data and a range of queries, each with real-world application analogues, demonstrate the superiority of our approach over a range of alternative methods.

PVLDB Reference Format:
Teddy Cunningham, Graham Cormode, Hakan Ferhatosmanoglu, and Divesh Srivastava. Real-World Trajectory Sharing with Local Differential Privacy. PVLDB, 14(11): 2283 - 2295, 2021.
doi:10.14778/3476249.3476280

1 INTRODUCTION

Sharing trajectories has obvious benefits for many real-world applications, including managing disease spread through contact tracing, and tailoring public services (such as bus routes) to a population's travel patterns. However, widespread public concern over privacy and data protection has limited the extent to which this data is currently shared. Differential privacy (DP) is an increasingly popular technique for publishing sensitive data with provable guarantees on individual privacy. In contrast to centralized DP, local differential privacy (LDP) allows users to share a perturbed version of their data, thus allaying fears of an untrusted data collector.

Although LDP provides a more practical setting and more attractive privacy properties, its mechanisms often have lower utility due to its stronger privacy requirements. This is (in part) because existing mechanisms fail to incorporate the wide range of real-world knowledge that is publicly available. This is a major shortcoming, especially as a wealth of accessible, open source information about the real world exists: detailed mapping data describing roads and points of interest; transit schedules; business opening hours; and unstructured user-generated data, in the form of reviews, check-ins, photos, and videos. These provide a rich and detailed (if somewhat non-uniform) description of the real world within which people navigate their lives. Within this context, approaches to data sharing that rely on crude abstractions, such as describing counts within uniform grids, appear simplistic. Moreover, they lead to synthetic data that fails to respect common sense: showing movement patterns that cross a mountain range as if there was a highway through it, or trajectories in which travelers visit a sports stadium in the middle of the night. We argue that, to be of value, efforts to share trajectories must more explicitly model the real world, and combine the private data with public information. To better capture realistic behavior, we propose solutions that include a wide range of external knowledge in a utility-enhancing manner, and show empirically that including real-world information greatly improves the utility of the perturbed data.

External knowledge can be incorporated in two ways. The first is a series of (deterministic) constraints that simply state whether one instance is feasible or not (e.g., someone cannot be 'located' in the sea). The second (probabilistic) approach is to make certain outputs more likely than others. Indeed, the second approach relates to another limitation of existing LDP mechanisms, wherein many assume equal sensitivity across data points. That is, with respect to place A, they treat places B and C to be equally sensitive, even if place A is much 'closer' to place B than to place C. Although a number of existing approaches instead use non-uniform probabilities in a utility-minded manner [1, 3, 4, 26], these works use relaxed definitions of ϵ-LDP, which is unsatisfying.

Our framework includes both of these approaches. First, a 'reachability' constraint ensures adjacent points in a trajectory can be reached in the respective time gap. Second, we use a multi-attributed distance function that ensures semantically similar locations are more likely to be returned by a perturbation mechanism. However,

Proceedings of the VLDB Endowment, Vol. 14, No. 11 ISSN 2150-8097.
doi:10.14778/3476249.3476280

privately perturbing trajectories in the local setting while incorporating real-world knowledge effectively is non-trivial. Furthermore, we go further than existing methods by proposing a solution that satisfies the strict requirements of ϵ-LDP.

Our first mechanism – which models trajectories as individual points in high-dimensional space – can be seen as the elegant, 'global' solution. However, its time and space complexity makes it computationally infeasible in most scenarios, which leads us to introduce our more efficient and scalable solution, based on perturbing overlapping, hierarchically-structured *n*-grams (i.e., contiguous subsequences of length *n*) of trajectory data. *n*-gram perturbation allows us to capture the spatio-temporal relationship between adjacent points, while remaining computationally feasible. Moreover, using *overlapping n*-grams allows us to capture more information for each point, whilst continuing to satisfy LDP. Our semantic distance function incorporates a rich set of public knowledge to adjust the probability of certain perturbations in a utility-enhancing manner. We also exploit the (publicly-known) hierarchies that are inherent in space, time, and category classifications to structure *n*-grams in a multi-dimensional hierarchy, which has notable benefits for utility. Exploiting a hierarchically-structured space in this manner also reduces the scale of the problem, which ensures that our solution is scalable for large urban datasets. Finally, we use optimization techniques to reconstruct the optimal realistic output trajectory from the set of perturbed *n*-grams.

We compare our mechanism to a number of alternative approaches by comparing the real and perturbed trajectory sets, and answering a range of application-inspired queries. Our mechanism produces perturbed trajectory sets that have high utility, preserve each location's category better than alternatives in all settings, and also preserve the temporal location of hotspots at a range of granularities. Our solution scales well with city size (unlike some other baselines), while remaining efficient and accurate.

In summary, the main contributions of our work are:

- an outline of the global solution, which we argue to be computationally infeasible in most cases;
- a robust, scalable, and efficient mechanism for perturbing spatio-temporal trajectory data in a way that satisfies ϵ-LDP;
- a method for integrating public knowledge into private mechanisms to give significant utility improvements with no cost to privacy; and
- extensive empirical analysis, including through a range of queries, that indicate our work's relevance in addressing important data analytics problems in a private manner.

We consider related work (Section 2) before formally introducing our problem and the guiding principles for our solution (Section 3). We introduce key definitions that are integral to our solution in Section 4. We begin Section 5 with the 'global' solution before outlining our hierarchical, *n*-gram-based solution. We also outline the semantic distance function and discuss the efficiency of our mechanism. We set out our experiments in Section 6 and present the results in Section 7. We discuss future work in Section 8.

2 RELATED WORK

Differential privacy [20] has become the de facto privacy standard. While centralized DP assumes data aggregators can be trusted, LDP [19] assumes that aggregators cannot be trusted and relies on data providers to perturb their own data. Many early LDP mechanisms [e.g., 5, 23, 44] assume all data points have equal sensitivity (i.e., the probability of any other data point being returned is equal), which can be unrealistic in practical settings, especially for spatial data.

There have been a number of recent relaxations of (L)DP to allow perturbation probabilities to be non-uniform across the domain. In d_{χ}-privacy [9], and its location-specific variant geoindistinguishability [4], the indistinguishability level between any two inputs is a function of the distance between them. This concept has since been generalized to any metric, and extended to the local setting to give metric-LDP [3]. Context-aware LDP [1] goes further by allowing an arbitrary (non-metric) measure of similarity between points, and input-discriminative LDP [26] assigns each data point its own privacy level. Other relaxations to LDP rely on the provision of some additional information. For example, personalized LDP [11] lets users specify a desired privacy level, whereas local information privacy [30] utilizes knowledge of users' priors.

Location data privacy (surveyed in [31]) has received a reasonable degree of attention in both centralized and local DP studies, (summarized in [24]). In addition to the aforementioned LDP relaxations, there have been some specific relaxations to LDP in the location domain (summarized in [37]). Applying (L)DP techniques to trajectory data, however, is less well-studied. Most early work used trajectory data to answer common queries [e.g., 6, 10, 35], which focus on returning summary level statistics, as opposed to individual-level data, which gives end users more flexibility. More recent DP-related work has focused on publishing and synthesizing differentially private trajectories [e.g., 27–29].

There has been DP-related work that considers user-specific context in which user priors are utilized [e.g., 30, 33, 36], and Desfontaines et al. [15] study the notion of DP with 'partial knowledge' for a user. Finally, Cunningham et al. [14] use publicly available geographic knowledge to improve the quality of private synthetic location data (in the centralized setting). However, we are the first to explicitly use external domain knowledge (i.e., user-independent prior information that is known to all, such as the locations, business opening hours, etc.) in the local setting to enhance utility.

In summary, our work introduces a rigorous and unified LDP-based and utility-minded mechanism for publishing individual-level trajectory data that incorporates a wide range of public knowledge.

3 PROBLEM MOTIVATION

Imagine a city in which each resident visits a number of places of interest (POIs) each day. These POIs link together in a time-ordered sequence to form a trajectory. The city's government wishes to learn aggregate information on where residents are traveling but, wary of governmental oversight, many residents are unwilling to share their entire trajectories truthfully. However, they are willing to share a slightly perturbed version of their trajectory, especially if it came with privacy guarantees. Hence, we wish to create a mechanism for users to share their trajectories in a privacy-preserving way, whilst ensuring that the shared trajectories preserve the major underlying patterns and trends of the real data at the aggregate level. We now describe the three principles that motivate and guide our solution: protecting privacy, enhancing utility, and ensuring efficiency.

Protecting Privacy. Our primary aim is to protect the individual privacy of each user so that she has plausible deniability within the dataset. We seek to achieve this by perturbing each user's trajectory in order to satisfy the requirements of DP. Specifically, we will use LDP wherein the data aggregator is not trusted. We assume that each user shares one trajectory each, and it is shared at the end of the data collection period. We discuss the privacy implications of these assumptions in Section 5.7.

Enhancing Utility by Incorporating External Knowledge. Although the primary aim of the mechanism is to preserve privacy, our practical goal is to ensure that the perturbed trajectories have high utility. Information to preserve to ensure high utility can range from hotspot information to co-location patterns and previous travel history. We argue that an important way to boost utility is to link the probability of perturbation from one location to any other with the semantic distance between the two locations. That is, one is more likely to be perturbed to another location if it is more semantically similar to its current location.

Furthermore, traditional (L)DP models impose strong privacy guarantees to protect against external information being used in an adversarial attack. However, in real-world applications of (L)DP, we argue that these protections can be too strong and can negatively affect the utility of the output dataset. To improve utility, we propose incorporating a range of *publicly known* external information to influence the output of our mechanisms. Examples of this type of information include business opening hours, sports teams schedules, and published maps. This knowledge can be used to influence the likelihood of certain perturbations, with the aim of boosting realism and utility. As this knowledge is publicly available, it is assumed adversaries have access to it.

Ensuring Efficiency. As our solutions utilize a wide range of public information from the real world, it is equally important that our solution can be applied to real world settings, at scale. Consequently, we complement our privacy and utility goals with the desire for our solution to be efficient and scalable for large urban datasets.

Applications. Our work focuses on perturbing trajectories such that aggregate statistics are preserved as much as possible, which leads to many important end applications of our work. A notable (and timely) one is the idea of *societal* contact tracing that seeks to identify the places and times in which large groups of people meet (so-called 'superspreading' events), as opposed to chance encounters between individuals. Knowledge of such events can be used for location-specific announcements and policy decisions. Other applications include advertising and provision of public services. For example, if a city council can identify popular trip chains among residents, they can improve the public transport infrastructure that links these popular places. Likewise, if a restaurant owner knows that many museum-goers eat lunch out after visiting a museum, she may consider advertising near museums.

4 DEFINITIONS

In this section, we introduce necessary notation and definitions; commonly-used notation is summarized in Table 1. We denote a set of POIs, as $\mathcal{P}$, where an individual POI is denoted by p_i. Each POI $p_i \in \mathcal{P}$ has a number of attributes – α_{it}, β_{it}, etc. – associated with it, which could represent the popularity, privacy level, category, etc. of the POI, and they can vary temporally. We quantize the time domain into a series of timesteps t, the size of which is controlled by the time granularity, g_t. For example, if g_t = 5 minutes, the time domain would be: T = {...10:00, 10:05, 10:10,...}.

Table 1: Commonly-Used Notation

Notation	Meaning
$p, \mathcal{P}$	Point of interest (POI), and set of POIs
g_t, g_s	Time and space granularity
t, T	Timestep, and set of all timesteps
$\tau, \mathcal{T}$	Trajectory, and set of trajectories
χ	Dimension subscript (e.g., t for temporal dimension)
$r_\chi, \mathcal{R}_\chi$	STC region, and set of regions
n	Length of trajectory fragment (i.e., n-gram)
$w, \mathcal{W}^n$	n-gram, and set of (reachable) n-grams
$\tau(a, b)$	Trajectory fragment; covers a^{th} to b^{th} elements of τ
θ	Reachability threshold
$d_\chi(\cdot)$	Distance function for dimension χ
ϵ	Privacy budget

We define a trajectory, τ, at the POI level as a sequence of POI-timestep pairs such that $\tau = \{(p_1, t_1), ...(p_i, t_i), ..., (p_{|\tau|}, t_{|\tau|})\}$, where $|\tau|$ denotes the number of POI-timestep pairs in a trajectory (i.e., its length). For each trajectory, we mandate that $t_{i+1} > t_i$ (i.e., one cannot go back in time, or be in two places at once). Each trajectory is part of a trajectory set, $\mathcal{T}$. Perturbed trajectories and trajectory sets are denoted as $\hat{\tau}$ and $\widehat{\mathcal{T}}$, respectively.

We use combined space-time-category (STC) hierarchical partitions in which we assign POIs to different STC regions. r_χ denotes an individual region where χ denotes the dimension of the region (i.e., s for space, t for time, etc.). Regions can be combined to form STC regions r_{stc} and $\mathcal{R}_\chi$ denotes region sets. A trajectory can be represented on the region level as $\tau = \{r_1, ...r_i, ..., r_{|\tau|}\}$ whereby r_i represents the i-th STC region in the trajectory. Consider the first point in a trajectory being {Central Park, 10:54am}. This might give r_s = {Upper Manhattan}, r_c = {Park}, and r_t = {10-11am}, leading to $r_{stc} = r_1$ = {Upper Manhattan, 10-11am, Park}.

Chaining regions (or POIs) together forms n-grams, denoted as w^n, whereby: $w^n = \{r_i, \ldots, r_{i+n-1}\}$. $\mathcal{W}^n$ is the set of all possible n-grams. We use $\tau(a, b)$ to specify a sub-sequence of τ such that: $\tau(a, b) = \{r_a, ..., r_i, ...r_b\}$, where a and b are the indices of τ. For example, $\tau(1, 3)$ denotes the first three STC regions (or POI-timestep pairs) of τ. $d_s(p_i, p_j)$ denotes the physical distance between p_i and p_j, and $d(r_i, r_j)$ denotes the distance between r_i and r_j.

4.1 Reachability

The notion of reachability is needed to ensure realism in perturbed trajectories. We begin identifying the subset of POIs that can be reached from any particular POI, based on the relative physical distance between them. A threshold, θ, represents the maximum distance that one can travel in a certain time period. θ can be specified directly, or be a function of g_t and a given travel speed.

Definition 4.1 (Reachability). A POI $p_b \in \mathcal{P}$ is reachable from p_a at time t if $d_s(a, b) \leq \theta(t)$, where $p_a \in \mathcal{P}$.

Reachability prevents illogical trajectories from being produced. For example, a trajectory of {New York City, Tokyo, London} would be unrealistic if the time granularity was one hour. Alternatively, imagine the trajectory: {Grand Central Station, Empire State Building, Central Park}. A realistic bigram to perturb to might be {Empire State Building, Times Square} (i.e., the two locations can be reached within one hour), whereas {Empire State Building, Mount Rushmore} is unrealistic as it would not satisfy the reachability constraint. The definition of reachability accommodates time-varying and asymmetric distances (e.g., caused by congestion and one-way roads, respectively). For n-gram perturbations, $\mathcal{W}^n$ is the set of all n-grams that satisfy the requirements of reachability. Formally, for the n-gram $w = \{p_a, ...p_i, p_{i+1}, ..., p_b\}$, the reachability constraint requires p_{i+1} to be reachable from p_i at time t_i for all $a \leq i < b$.

4.2 Privacy Mechanisms

Definition 4.2 (ϵ-local differential privacy [19]). A randomized mechanism $\mathcal{M}$ is ϵ-local differentially private if, for any two inputs x, x' and output y:

$$\frac{\Pr[\mathcal{M}(x)=y]}{\Pr[\mathcal{M}(x')=y]} \leq e^{\epsilon} \tag{1}$$

Whereas centralized DP allows the aggregator to add noise, LDP ensures that noise is added to data before it is shared with an aggregator. Like its centralized analogue, LDP possesses two fundamental properties that we use in our mechanism [13]. The first is the composition theorem, which states that one can apply k ϵ_i-LDP mechanisms, with the result satisfying ϵ-LDP, where $\epsilon = \sum_i \epsilon_i$. The second property allows post-processing to be performed on private outputs without affecting the privacy guarantee.

Definition 4.3 (Exponential Mechanism (EM) [39]). For an input x and output $y \in \mathcal{Y}$, the result of mechanism $\mathcal{M}$ is ϵ-differentially private if one randomly selects y such that:

$$\Pr[\mathcal{M}(x) = y] = \frac{\exp\left(\epsilon q(x,y)/2\Delta_q\right)}{\sum_{y_i \in \mathcal{Y}} \exp\left(\epsilon q(x,y_i)/2\Delta_q\right)} \tag{2}$$

where, $q(x, y)$ is a quality (or utility) function, and Δ_q is the sensitivity of the quality function, defined as $\max_{y,y'} |q(x, y) - q(x, y')|$.

The utility of the EM can be written as:

$$\Pr\left[q(x,y) \leq OPT_q - \frac{2\Delta_q}{\epsilon}\left(\ln\frac{|\mathcal{Y}|}{|\mathcal{Y}_{OPT}|} + \zeta\right)\right] \leq e^{-\zeta} \tag{3}$$

where OPT_q is the maximum value of $q(x, y)$, and $\mathcal{Y}_{OPT} \subseteq \mathcal{Y}$ is the set of outputs where $q(x, y) = OPT_q$ [22].

Although more commonly used for centralized DP, the EM can be applied in LDP, where we consider different inputs (for LDP) to be equivalent to neighboring datasets of size 1 (for centralized DP). By using the EM with a distance-based quality function, we achieve ϵ-LDP (i.e., the probability ratio for any perturbation is upper-bounded by e^{ϵ}). This is in contrast to relaxations of LDP, such as metric-LDP [3], context-aware LDP [1], or input-discriminative LDP [25]. All of these have upper bounds for the probability ratio of a perturbation of the form $e^{f(x,x')}$, where f is a function that quantifies the distance between two inputs x and x'. Note that our setting is a specific (stricter) case in which $f(x, x') = \epsilon$ for all x, x'.

5 TRAJECTORY PERTURBATION

In this section, we first present a global solution that perturbs the whole trajectory (Section 5.1), before outlining our n-gram-based mechanism that addresses the infeasible aspects of the global solution (Sections 5.2–5.6). We provide theoretical analysis (Sections 5.7 and 5.8), a number of alternative approaches (Section 5.9), and an outline of our multi-attributed distance function (Section 5.10).

5.1 Global Solution

In the global solution, we model entire trajectories as points in high-dimensional space. Having instantiated all possible trajectories, we determine the distance between these high-dimensional points and the real trajectory, and use this distance to determine the probability distribution. The probability of τ being perturbed to $\hat{\tau}$ is:

$$\Pr(\hat{\tau} = \tau_i) = \frac{\exp\left(-\epsilon d_\tau(\tau,\tau_i)/2\Delta_{d_\tau}\right)}{\sum_{\tau_i \in \mathcal{S}} \exp\left(-\epsilon d_\tau(\tau,\tau_i)/2\Delta_{d_\tau}\right)} \tag{4}$$

where, $\mathcal{S}$ is the set of all possible trajectories, and d_τ is the distance function that represents the distance between trajectories (see Section 5.10). We use the EM (2) to perturb trajectories. Proof that the global solution satisfies ϵ-LDP follows from (4) and Definition 4.3.

Theorem 5.1. *The utility of the global solution is expressed as:*

$$\Pr\left[d_\tau(\tau, \hat{\tau}) \leq -\frac{2\Delta_{d_\tau}}{\epsilon}\left(\ln|\mathcal{S}| + \zeta\right)\right] \leq e^{-\zeta} \tag{5}$$

Proof. The quality function is the distance function d_τ, and its maximum value is obtained iff $\hat{\tau} = \tau$. Hence, $OPT_q = 0$ and $|\mathcal{Y}_{OPT}| = 1$. Substituting this into (3) yields (5). □

To assess the feasibility of the global solution, we first analyze its worst-case time complexity. The number of possible timestep sequences is $\frac{|T|!}{|\tau|! \times (|T|-|\tau|)!}$ where $|T|$ is the number of timesteps, which is a function of g_t (in minutes): $|T| = \frac{24 \times 60}{g_t}$. The number of possible POI sequences is $|\mathcal{P}|^{|\tau|}$. Hence, the maximum size of $\mathcal{S}$ is: $|\mathcal{S}| = \frac{|\mathcal{P}|^{|\tau|} \times |T|!}{|\tau|! \times (|T|-|\tau|)!}$ The global solution requires instantiating all trajectories in $\mathcal{S}$, the size of which grows exponentially with $|\tau|$.

In reality, the reachability constraint reduces the number of possible trajectories. Assuming that (on average) μ% of all POIs are reachable between successive timesteps, the number of possible trajectories is reduced by a factor of $\mu^{|\tau|-1}$. To illustrate this, imagine a small-scale example where $|\tau| = 5$, $g_t = 15$ minutes, $|\mathcal{P}| = 1{,}000$, and $\mu = 20\%$. Even under these settings, $|\mathcal{S}| \approx 9.78 \times 10^{19}$, which means $\mathcal{S}$ remains computationally infeasible to compute and store.

Variants of the EM approach have been proposed, with the aim of tackling the high cost of enumerating all possible outputs. The subsampled exponential mechanism applies the EM to a sample of possible outputs [34]. In our case, we would need the sampling rate to be very small to make this approach computationally feasible. However, the highly skewed distribution of d_τ means that the sampling rarely selects trajectories with low d_τ values, which ultimately leads to poor utility in the perturbed dataset. The ‘Permute and Flip’ approach [38] instead considers each output in a random order, and performs a Bernoulli test to see if it can be output. However, the success probability, which is proportional to $\exp(-\epsilon d_\tau)$, is generally low, meaning that efficiency gains are limited. Instead, we look to alternative models to make our approach feasible.

5.2 *n*-gram Solution Overview

Instead of considering trajectories as high-dimensional points, we propose using overlapping fragments of the trajectories to capture spatio-temporal patterns with efficient privacy-preserving computations. Specifically, we consider a hierarchical n-gram-based solution that aims to privately perturb trajectories more quickly. Using (overlapping) n-grams allows us to consider the spatio-temporal link between any n consecutive points, which is necessary to accurately model trajectory data.

Our solution (summarized in Figure 1) has four main steps: hierarchical decomposition, n-gram perturbation, optimal STC region-level reconstruction, and POI-level trajectory reconstruction. Hierarchical decomposition is a pre-processing step that only uses public information, so it can be done a priori and without use of the privacy budget. Similarly, both trajectory reconstruction steps do not interact with private data, thus allowing us to invoke LDP's post-processing property without using the privacy budget.

5.3 Hierarchical Decomposition

As $|\mathcal{P}|$ increases, the number of feasible outputs for POI-level perturbation grows exponentially. To address this challenge, we utilize hierarchical decomposition to divide POIs into STC regions.

STC Region Composition. We first divide the physical space into R_s spatial regions. For each $r_s \in \mathcal{R}_s$, we create R_c regions – one for each POI category. For each space-category region, we create R_t regions, which represent coarse time intervals. POIs are then assigned to STC regions, based upon their location, opening hours, and category. POIs can appear in more than one STC region (e.g., they are open throughout the day and/or the POI has more than one category). We remove all STC regions that have zero POIs within them (e.g., r_{stc} = (top of mountain, 3am, church)), which ensures that these regions will not be included in $\mathcal{W}^n$. As all this information is public, it does not consume any privacy budget.

$\mathcal{R}_s$ can be formed using any spatial decomposition technique, such as uniform grids or clustering, or it can use known geography, such as census tracts, blocks, or boroughs. We find that our mechanism is robust to the choice of spatial decomposition technique. $\mathcal{R}_c$ can be derived from known POI classification hierarchies, such as those published by OpenStreetMap, etc. $\mathcal{R}_t$ is most easily formed by considering a number of coarse (e.g., hourly) time intervals.

STC Region Merging. Depending on the number of STC regions, and the number of POIs within them, STC regions can be merged across any (or all) of the three dimensions. For example, instead of r_{stc} = (Main Street, 1am, Nightclub) and r_{stc} = (Main Street, 11pm, Bar), they could be merged into r_{stc} = (Main Street, 11pm-2am, Nightlife), which represents merging in the time and category dimensions.

Merging regions is done primarily for efficiency reasons as it prevents many semantically similar, but sparsely populated, regions from existing. Additional POI-specific information (e.g. popularity) can be included into merging criteria to prevent significant negative utility effects. For example, if the data aggregator wishes to preserve large spatio-temporal hotspots, they will want to prevent merging very popular POIs with semantically similar but less popular POIs. For example, consider a conference center complex. Although all conference halls are semantically similar, one hall might have a large trade show, whereas the others may have small meetings. It is important not to merge all halls in this case, as this might result in less accurate responses to downstream data mining tasks. Hence, we require that each STC region has κ POIs associated with it where κ is a pre-defined function of POI attributes.

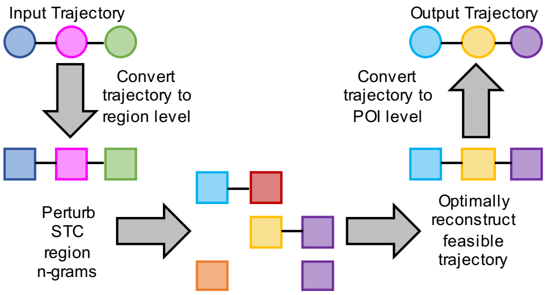

Input Trajectory
Convert trajectory to region level
Perturb STC region n-grams
Optimally reconstruct feasible trajectory
Convert trajectory to POI level
Output Trajectory

Figure 1: Solution Overview

To further illustrate this, consider Figure 2a, which shows ten POIs, divided in a 3 × 3 grid where larger circles indicate a more popular POI. Figure 2b shows how regions might be merged if we only consider geographic proximity, whereas Figure 2c shows the resultant regions when merging accounts for perceived popularity. We see more POIs in regions with less popular POIs, whereas very popular POIs exist singly in a region. Deciding along which dimensions to merge regions, as well as the priority and extent of merging, depends ultimately on the utility goals of the data aggregator. For example, if preserving the category of POIs is important, merging in the time and space dimensions first would be preferable.

n-gram Set Formation. As a final pre-processing step, we define $\mathcal{W}^n$ by first instantiating all possible n-gram combinations of STC regions. We then remove all n-gram combinations that do not satisfy the reachability constraint. For the region level, we deem any r_a and r_b to be reachable if there is at least one $p_i \in r_a$ and at least one $p_j \in r_b$ that satisfy reachability.

5.4 *n*-gram Perturbation

Once $\mathcal{W}^n$ has been defined, we convert each trajectory from a sequence of POI-timestep pairs to a sequence of STC regions. The next step is to perturb the STC regions of τ by using overlapping n-grams and the EM.

Notation. We define Z to be a set that holds all the perturbed n-grams of τ. We then use $z(a,b) = \{\hat{r}_a, ..., \hat{r}_i, ...\hat{r}_b\}$ to be the perturbed n-gram, where a and b are the indices of τ and $z(a,b) \in Z$. Importantly, there is a subtle difference between $\hat{\tau}(a,b)$ and $z(a,b)$. In Z, for any timestep, there are multiple possible regions associated with each trajectory point, whereas $\hat{\tau}$ is the final reconstructed trajectory and so there is only one region for each trajectory point.

Main Perturbation. For each perturbation, we take $\mathcal{W}^n$ and define the probability that $\tau(a,b)$ is perturbed to $w_i \in \mathcal{W}^n$ as:

$$\Pr(z(a,b) = w_i) = \frac{\exp\left(-\epsilon' d_w(\tau(a,b), w_i)/2\Delta_{d_w}\right)}{\sum_{w \in \mathcal{W}^n} \exp\left(-\epsilon' d_w(\tau(a,b), w_i)/2\Delta_{d_w}\right)} \tag{6}$$

where $\epsilon' = \frac{\epsilon}{|\tau|+n-1}$, and $d_w(\tau(a,b), w_i)$ is the function that quantifies the distance between n-grams. To ensure that n-grams are perturbed, we specify the ranges of a and b such that $a = (1, |\tau| - n + 1)$ and $b = (n, |\tau|)$. Once these probabilities have been defined, we use

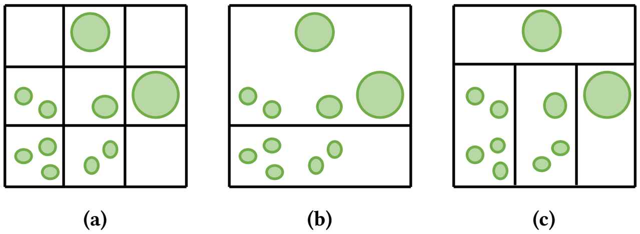


Figure 2: Illustrative example of the STC region merging

the EM to sample from $\mathcal{W}^n$ and we store $z(a, b)$ in Z. We repeat this for increasing values of a and b (see Figure 3).

Using overlapping n-grams gives better accuracy than using non-overlapping n-grams or merely perturbing points independently. It lets us repeatedly 'query' a point multiple times, meaning that we gather more information about it while continuing to guarantee LDP. Overlapping n-grams simultaneously allows us to query a larger portion of the entire trajectory, which enables us to base each perturbation upon a wider range of semantic information. For example, $\hat{\tau}(3)$ is determined based on information from $\tau(1, 3)$, $\tau(2, 4)$, and $\tau(3, 5)$, assuming $n = 3$. Hence, $\tau(3)$ is 'queried' n times, neighboring points $n-1$ times, etc., in addition to using information from $2n - 1$ points to influence the perturbation of $\tau(3)$.

End Effects. When $n \geq 2$, the start and end regions in a trajectory are not covered n times. For example, when $|\tau| = 4$ and $n = 2$, the main perturbation step covers the first and last regions once only (with $z(1, 2)$ and $z(3, 4)$, respectively). To ensure that all timesteps have the same number of perturbed regions, we conduct extra perturbations with smaller n-grams. This supplementary perturbation is performed in the same manner as (6), but with different $\mathcal{W}^n$ sets and different bounds for a and b. In our example, we would obtain $z(1, 1)$ and $z(4, 4)$ using $\mathcal{W}^1$ (as illustrated in Figure 3).

Theoretical Utility. As the solution has multiple post-processing stages, a theoretical utility guarantee for the entire mechanism remains elusive. However, we can analyze the utility of the n-gram perturbation step.

Theorem 5.2. *The utility of the n-gram perturbation stage is:*

$$\Pr\left[d_w(\tau(a, b), w) \leq -\tfrac{2\Delta_{d_w}}{\epsilon'}\left(\ln|\mathcal{W}^n| + \zeta\right)\right] \leq e^{-\zeta} \tag{7}$$

Proof. The quality function is the distance function d_w, and its maximum value is obtained iff $z(a, b) = \tau(a, b)$. Hence, $OPT_{d_w} = 0$ and $|\mathcal{W}^n_{OPT}| = 1$. Substituting this into (3) yields (7). □

Hence, the utility is dependent on the size of the feasible n-gram set, which itself is influenced by n, the granularity of hierarchical decomposition, and the strictness of the reachability constraint. Utility is also affected by trajectory length, as ϵ' is a function of $|\tau|$.

5.5 Region-Level Trajectory Reconstruction

Given a collection of perturbed n-grams, we define an optimization problem that reconstructs a trajectory between points in $\hat{\tau}$ and the perturbed n-grams in Z. This is post-processing of the LDP output, and does not consume any privacy budget. We define two error terms (illustrated in Figure 4) that measure the similarity of regions to the perturbed data. By perturbing each point in τ n times (by using overlapping n-grams), we magnify this effect.

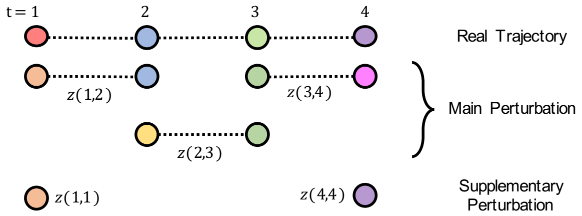


Figure 3: Main and supplementary perturbation mechanisms; different colors indicate different STC regions

The first error term is the 'region error' $e(r_j, i)$, which is the distance between r_j and the perturbed n-grams in location i in Z. It is defined as:

$$e(\hat{r}_j, i) = \sum d(\hat{r}_j, y_i) \tag{8}$$

where y_i is the region from $z(a, b) \in Z$ iff $a \leq i \leq b$, where a, b, and i are trajectory indices. The second error term is the 'bigram error', $e(i, w)$, which is the sum of the two relevant region error terms. More formally, it is defined as:

$$e(i, w) = e\left(w(1), i\right) + e\left(w(2), i + 1\right) \tag{9}$$

where w is a region-level bigram in $\mathcal{W}^2$, with $w(1)$ and $w(2)$ being the first and second regions in w, respectively.

We now define the minimization problem as:

$$\min \ \textstyle\sum_{i=1}^{|\tau|-1} x_i^w e(i, w) \tag{10}$$

$$\text{s.t.} \quad x_i^w \cdot q(w_i, w_{i+1}) = x_{i+1}^w \cdot q(w_i, w_{i+1}) \quad \forall\, 1 \leq i < |\tau| \tag{11}$$

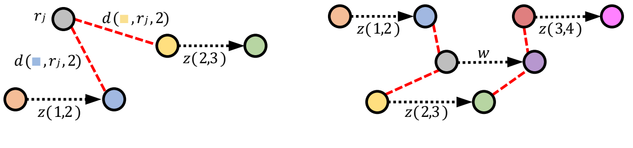
$$q(w_i, w_{i+1}) = \begin{cases} 1 & \text{if } w_i(2) = w_{i+1}(1) \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

$$\textstyle\sum_{i=1}^{|\tau|-1} x_i^w = |\tau| - 1 \tag{13}$$

$$\textstyle\sum_{w \in \mathcal{W}^2} x_i^w = 1 \quad \forall\, 1 \leq i < |\tau| \tag{14}$$

where, x_i^w is a binary variable encoding whether w is selected for index i. The objective (10) is to minimize the total bigram error across the trajectory. (11) and (12) are continuity constraints that ensure consecutive bigrams share a common region. (13) ensures that the number of bigrams selected is correct, and (14) ensures only one bigram is associated with each point in the trajectory.

Efficiency Discussion. Assuming that the space, time, and category granularities are well-chosen such that $|\mathcal{W}^2| \ll |\mathcal{R}|^2$, the scale of the optimization problem will generally be within the scope of most linear programming solvers (see Section 5.8). Nevertheless, we introduce a step to further limit the set of possible bigrams that can appear in the reconstructed trajectory. Once n-gram perturbation is complete, we obtain the minimum bounding rectangle (MBR) defined by all $r_{stc} \in Z$. From this, we define $\mathcal{P}_{mbr} \subseteq \mathcal{P}$, which contains all the POIs in this MBR, and $\mathcal{R}_{mbr}$, which is the set of STC regions that contain at least one POI in $\mathcal{P}_{mbr}$. From this, we define $\mathcal{W}^2_{mbr}$ as the set of feasible bigrams formed from $\mathcal{R}_{mbr}$, and we use this set in the reconstruction. Performing this step does not prevent the optimal reconstructed trajectory from being found, as the reconstruction seeks to minimize the error with respect to the perturbed n-grams in Z, all of which are included in $\mathcal{R}_{mbr}$.



(a) Node Error

(b) Bigram Error

Figure 4: Illustrative example of error terms: error equals sum of red dashed line distances

5.6 POI-Level Trajectory Reconstruction

The final step is to express the output trajectories in the same format as the input trajectories. Whereas converting a trajectory from POIs to STC regions is relatively straightforward, the converse operation is non-trivial as there can be many possible POI-level trajectories corresponding to a certain sequence of STC regions. Furthermore, the reachability requirement means that some trajectories are infeasible, and should not be published.

We expect most POI-level trajectories to be feasible as $\mathcal{W}^n$ is defined based on the reachability criterion. Hence, we generate an individual trajectory at random and check that it satisfies the reachability constraint. If it does, we output it; if not, we generate another trajectory. We continue this until we generate a feasible trajectory, reach a threshold (γ), or exhaust all possible combinations. Experimentally, the threshold of $\gamma = 50{,}000$ was rarely reached.

When this trajectory sampling fails, it implies that the perturbed region sequence does not correspond to a feasible trajectory. If so, we randomly select a POI and time sequence and 'smooth' the times such that they become feasible. For example, consider the region-level trajectory: {(Restaurant, 9-10pm, Downtown), (Bar, 9-10pm, Downtown), (Bar, 9-10pm, Suburb)}. Reachability may mean that the suburban bar is only reachable from the downtown bar in 55 minutes, meaning that it is impossible to visit all three venues in an hour. Accordingly, we smooth the timesteps such that either $\tau(1)$ occurs between 8 and 9pm, or $\tau(3)$ occurs between 10 and 11pm.

5.7 Privacy Analysis

We now analyze the privacy of the n-gram-based solution through a sketch proof, and discuss some possible adversarial attacks.

Theorem 5.3. *The perturbation of trajectory τ satisfies ϵ-LDP.*

Proof. Each n-gram $\tau(a, b)$ – where $a = (1, |\tau| - n + 1)$ and $b = (n, |\tau|)$ – is perturbed with privacy budget ϵ' using (6), and the EM. This means each perturbation satisfies ϵ'-DP, and there are $|\tau| - n + 1$ of these perturbations. Because of end effects, there are an additional $2(n - 1)$ perturbations, each of which also satisfy ϵ'-DP. Using sequential composition, the resultant output satisfies $(|\tau|+n-1)\epsilon'$-DP. As $\epsilon' = \frac{\epsilon}{(|\tau|+n-1)}$, the overall mechanism therefore satisfies ϵ-DP. As the size of adjacent datasets is 1, ϵ-DP results are equivalent to ϵ-LDP results. □

As we use publicly available external knowledge, we assume that an adversary has access to all the same knowledge. However, external knowledge is *only* used to enhance utility, whereas privacy is provided through the application of the EM (which could be done with no external knowledge). Hence, an adversary cannot use this information to learn meaningful information with high probability. As our solution is predicated on a 'one user, one trajectory' basis, inference attacks based on repeated journeys from the same user are prevented by definition. We can use sequential composition to extend our solution to the multiple release setting; assuming each of k trajectories is assigned a privacy budget of ϵ, the resultant release provides $(k\epsilon)$-LDP to each user. Finally, unlike in other works that consider continuous data sharing [e.g. 2, 8, 21, 32], our setting sees the user share all data at the end of their trajectory. Hence, as we provide user-level ϵ-LDP, the LDP privacy guarantee protects against spatial and temporal correlation attacks.

5.8 Computational Cost

We now discuss the computational costs of the proposed n-gram-based approach, which highlight how it is highly practical. Also, as each perturbation can be done locally on a user's device, the entire data collection operation is inherently distributed and scalable.

Choice of n. Precomputing $\mathcal{W}^n$ requires $O(|\mathcal{P}|^n)$ space, which becomes infeasible for large cities and $n \geq 3$. If $n \geq 3$ and $|\mathcal{P}|$ is large, feasible n-grams can be computed 'on-the-fly', although this attracts a significant runtime cost. While these effects are partially mitigated by using STC regions, we recommend choosing $n = 2$ (bigrams) as $n \geq 3$ will be unrealistic in most practical settings.

Time Complexities. Converting a trajectory from the POI-level to STC region level has time complexity $O(|\tau|)$ and the perturbation phase has time complexity $O(n|\tau|)$. Converting trajectories from the STC region level back to the POI level (assuming time smoothing does not need to be performed) has a worst-case time complexity of $O(\gamma(|\tau| + |\tau - 1|))$. This is because $|\tau|$ POIs need to be selected, and then reachability checks need to be performed on each link (of which there are $|\tau| - 1$ in total). In the worst-case, this process is repeated γ times, hence $O(\gamma(|\tau| + |\tau - 1|))$. We find that time smoothing is needed for around 2% of trajectories on average, and so we devote little focus to its runtime effects here.

Optimal Reconstruction Complexity. Section 5.5 presents the optimal reconstruction phase, which uses integer linear programming. Here, we briefly discuss the scale of the problem in terms of the number of variables and constraints. Let $\phi = |\mathcal{R}_{mbr}|$. The number of feasible bigrams will be ϕ^2 in the worst case, although the reachability constraint reduces this in practice. From the definition of x_i^w, we see that there are $\phi(|\tau| - 1)$ variables (i.e., one x_i^w per bigram, per trajectory point). The continuity constraints – (11) and (12) – impart $\phi(|\tau| - 1)$ constraints, and the capacity constraints – (13) and (14) – impart $|\tau| - 1$ constraints. Hence, the optimization problem has $(\phi|\tau| + |\tau| - \phi - 1)$ constraints in total. Closed-form expressions for the expected runtime of optimization problems depend on the exact solver chosen, and are typically fast in practice [see 12, 43].

5.9 Alternative Approaches

We compare our solution to other approaches, summarized here.

Using Physical Distance Only. The most basic distance-based perturbation mechanism (called PhysDist) would ignore external knowldge and only use the physical distance between POIs/regions.

POI-level n-Gram Perturbation. Our mechanism can be applied just on the POI-level. NGramNoH perturbs the time and POI dimensions separately in order to control the size of $\mathcal{W}^n$. This requires splitting the privacy budget more (i.e., $\epsilon' = \frac{\epsilon}{2|\tau|+n-1}$).

Independent POI Perturbation. The simplest approach is to perturb each POI independently of all others. We consider two variations of this approach: one where the reachability constraint is considered during perturbation (IndReach), and one where it is not (IndNoReach). To ensure that feasible trajectories are output when using IndNoReach, we use post-processing to shift the perturbed timesteps to ensure a 'realistic' output. While such methods make less intensive use of the privacy budget, they fail to account for the intrinsic relationship between consecutive points. However, when temporal gaps between POIs are large, the reachability constraint becomes less influential, making these methods more attractive.

Other LDP Relaxations. As discussed in Section 2, a number of distance-based perturbation mechanisms that were inspired by the principles of (L)DP exist. However, although these approaches possess their own theoretical guarantees, they do not satisfy ϵ-LDP, which makes them incomparable with our mechanism.

5.10 Distance Function

We now outline the semantic distance functions used throughout our work. Note that our mechanism is not reliant on any specific distance/quality function – any other distance function can be used, without needing to change the mechanism.

Physical Distance. We use $d_s(p_a, p_b, t)$ to denote the physical distance from p_a to p_b at time t, which can be derived using any distance measure (e.g., Euclidean, Haversine, road network). To get the distances between STC regions, we obtain the distance between the centroids of the POIs in the two regions. We similarly use $d_s(r_a, r_b)$ to denote the physical distance between r_a and r_b.

Time Distance. The time distance between regions is defined as the absolute time difference between two STC regions. That is, $d_t(r_a, r_b) = |t_a - t_b|$. We limit time distances to ensure that no time distance is greater than 12 hours. Where STC regions are merged in the time dimension, we use the time difference between the centroids of the merged time intervals. For example, if two regions cover 2-4pm and 5-7pm, $d_t = |3 - 6| = 3$ hours.

Category Distance. Category distance, d_c, is quantified using a multi-level hierarchy. (We use three, although any number of levels can be used.) Figure 5 illustrates how d_c varies across hierarchical levels, relative to the leftmost level 3 (white) node. We define category distance to be symmetric (i.e., $d_c(\text{Shoe Shop}, \text{Shopping}) = d_c(\text{Shopping}, \text{Shoe Shop})$. If two POIs or regions do not share a level 1 category, we deem them to be unrelated and $d_c = 10$ (indicated by the dotted line and purple node in Figure 5).

Combining Distances. Distance functions are combined as follows:

$$d(r_a, r_b) = \left(d_s(r_a, r_b)^2 + d_t(r_a, r_b)^2 + d_c(r_a, r_b)^2\right)^{1/2} \qquad (15)$$

To determine the 'distance' between two n-grams, we use element-wise summation. For example, the distance between two bigrams – $w_i = \{i_1, i_2\}$ and $w_j = \{j_1, j_2\}$ – is calculated as $d(i_1, j_1) + d(i_2, j_2)$. More generally, if w_1 and w_2 are two n-grams:

$$d_w(w_1, w_2) = \sum_{a=1}^{n} d(i_a, j_a) \qquad (16)$$

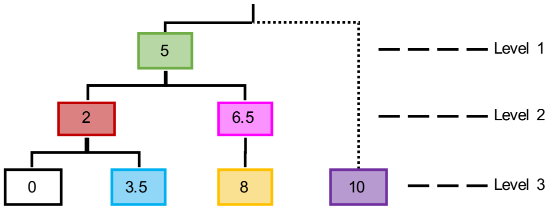


Figure 5: d_c values, relative to left-most level 3 node

6 EXPERIMENTAL SET-UP

Our experiments seek to: a) analyze our mechanism and gather insights from its behavior; b) compare our approach to comparable alternatives; and c) demonstrate the practical utility of our mechanism in the context of application-inspired queries.

6.1 Data

We use a range of real, synthetic, and semi-synthetic trajectory datasets. There is a chronic lack of high quality, publicly available POI sequence data. Hence, we augment existing real datasets to make them suitable for a comprehensive evaluation.

6.1.1 Real Data. We combine Foursquare check-in [45] and historic taxi trip [42] data, both from New York City. The set of POIs is taken from all POIs that appear in the raw Foursquare dataset, from which we take the $|\mathcal{P}|$ most popular as our set $\mathcal{P}$. We concatenate the pick-up and drop-off locations of each taxi driver's daily trips in order to protect their business-sensitive movements. We match the co-ordinate data with the nearest POI. If no POIs within 100m are found, we discard the point. We clean the data by removing repeat points with the same venue ID or exact latitude-longitude location. Where points occur less than g_t minutes apart, we remove all but one point at random. For category information, we use the publicly available Foursquare category hierarchy [16] to assign a single category to each POI. We manually specify opening hours for each broad category (e.g., 'Food', 'Arts and Entertainment'), and set all POIs of that (parent) category to have those hours. However, the mechanism is designed to allow POI-specific opening hours.

6.1.2 Semi-Synthetic Data. We use Safegraph's Patterns and Places data [41] to semi-synthetically generate trajectories. We randomly determine $|\tau|$ using a uniform distribution with bounds (3,8), and the start time using a uniform distribution with bounds (6am, 10pm). The starting POI is selected at random from the popularity distributions of the day/time in question. We sample from the distribution of dwell times at each POI to determine the time spent in one location, and we sample the time spent traveling to the next POI uniformly from (0, 60) minutes. The next POI is sampled at random, based on the popularity distribution at the expected arrival time (based on the POIs that are 'reachable'). This process continues until a trajectory is generated. Safegraph uses the NAICS hierarchy [7], and we use this system for the category hierarchy. Opening hours information for POIs is sparse, and so we manually assign general opening hours to categories, as in the Taxi-Foursquare data.

6.1.3 Campus Data. We generate trajectories based on the University of British Columbia campus [40]. The 262 campus buildings act as POIs, and nine POI categories exist, such as ‘academic building’, ‘student residence’, etc. We determine trajectory length and start time in the same way as for the Safegraph data. For each subsequent timestep in the trajectory, we sample from a uniform distribution with bounds (g_t, 120) minutes. The category of the first POI is chosen at random, and the exact POI is chosen at random from all POIs in the selected category. For each subsequent POI, the POI is chosen from the set of reachable POIs based on the preceding POI, the time gap, and the time of day. We artificially induce three popular events into the synthetic trajectories by picking a point in the trajectory, and controlling the time, POI, and category of the trajectory point. The remainder of the trajectory is generated as per the previously outlined method. The three popular events are: 500 people at Residence A at 8-10pm; 1000 people at Stadium A at 2-4pm; and 2000 people in some academic buildings at 9-11am.

6.1.4 External Knowledge Specification. Although we specify external knowledge manually, more scalable, operator-independent methods are possible. For example, APIs of major location-based services (e.g., Google Maps) can be used to query thousands of POIs efficiently and cheaply. In the case of Google Maps, information such as location, opening hours, category, price levels, etc. can be obtained directly through their API. This information can be stored in the POI-level database, with which the mechanism interacts.

6.1.5 Pre-Processing Costs. The pre-processing necessary for our experiments is split into three parts: (a) POI processing, hierarchical decomposition, and region specification; (b) trajectory composition; and (c) trajectory filtering. Part (a) is a one-time operation that creates the necessary data structures. The impact of specifying external knowledge is negligible as the data structures (e.g., $\mathcal{R}$, $\mathcal{W}^n$) need to be created regardless. Figure 7 shows the runtime costs for pre-processing step (a) for the two large-scale datasets. The runtime is heavily dependent on the size of $\mathcal{P}$, but less influenced by the reachability constraint. It is independent of other variables, such as trajectory length and privacy budget. Although it is a one-time operation, localized updates can be performed to reflect changes in the real world (e.g., changes in POI opening times, new roads affecting reachable POIs). Despite the large runtime, we argue that it is an acceptable cost, especially as it is a one-time process. Parts (b) and (c) are only necessary as we are simulating the perturbation of thousands of trajectories. In a practical setting, parts (b) and (c) are negligible as the trajectory data is created by each user and, by definition, a real trajectory should satisfy reachability and other feasibility constraints. If there are infeasible aspects in a trajectory, smoothing operations can be performed in sub-second time.

6.2 Experimental Settings

We set $g_t = 10$ minutes, $n = 2$ for all n-gram-based methods, and $|\mathcal{P}| = 2{,}000$. We set $\epsilon = 5$, in line with other real-world deployments of LDP by Apple [17] and Microsoft [18]. We assume all travel is at 4km/hr (campus data) and 8km/hr (Taxi-Foursquare and Safegraph data). These speeds correspond to approximate walking and public transport speeds in cities, once waiting times, etc. have been included. We consider the effects of varying these parameters in Section 7.2. We use Haversine distance throughout. We filter $\mathcal{T}$ to remove trajectories that do not satisfy the reachability constraint or where POIs are ‘visited’ when they are closed. In general, the size of $\mathcal{T}$ (once filtered) is in the range of 5,000-10,000.

When creating STC regions, we divide the physical space using a $g_s \times g_s$ uniform grid. The finest granularity we consider is $g_s = 4$, and we use coarser granularities ($g_s = \{1, 2\}$) when performing spatial merging. We use the first three levels of the Foursquare and NAICS category hierarchies (for the Taxi-Foursquare and Safegraph data, respectively), and use the category distance function outlined in Section 5.10. Using these levels ensures that $d_c = 10$ for POIs with completely different categories. STC regions have a default time granularity of one hour. By default, we perform spatial merging first, followed by time merging, and category merging, and $\kappa = 10$.

We do not find any suitable alternatives in the literature—all existing work is based in the centralized DP domain, or uses relaxed definitions of LDP. Instead, we compare to the alternative approaches outlined in Section 5.9.

6.3 Utility Measures

We assess the accuracy of our perturbed trajectories through a range of measures. First, we examine the distance between the real and perturbed trajectories. We normalize the distance values by $|\tau|$ and use the term ‘normalized error’ (NE) henceforth. For this, we use the same distance definitions as outlined in Section 5.10.

6.3.1 Preservation Range Queries. We also examine a set of ‘preservation range queries’ (PRQs). That is, for each point in each trajectory, we check to see whether the perturbed POI is within δ units of the true POI. For example, a location PRQ might examine whether $\hat{p}_i$ is within 50 meters of p_i. We conduct PRQs in all three dimensions, and define the utility metric PR_χ as:

$$PR_\chi = \frac{1}{|\mathcal{T}|} \sum_{\tau \in \mathcal{T}} \left(\frac{1}{|\tau|} \sum_{i=1}^{|\tau|} \pi(p_i, \hat{p}_j, \delta_\chi) \right) \times 100\% \qquad (17)$$

where $\pi(p_i, \hat{p}_j, \delta)$ equals 1 if $d_\chi(p_i, \hat{p}_j) \leq \delta$, and zero otherwise. For time PRQs, t_i and $\hat{t}_i$ replace p_i and $\hat{p}_i$, respectively.

6.3.2 Hotspot Preservation. For each POI, spatial region, or category, we define a *spatio-temporal hotspot* as the time interval during which the number of unique visitors is above a threshold η. A hotspot is characterized by $h = \{t_s, t_e, p_i, c\}$, where t_s and t_e are the start and end times, p_i is the POI, and c is the maximum count reached in the interval. Note that multiple hotspots can be associated with the same POI if popularity changes over time (e.g., a train station might have hotspots during the AM and PM peaks).

We consider three spatial granularities: POI-level, and spatial regions defined by 4×4 and 2×2 grids, and $\eta = \{20, 20, 50\}$, respectively. We consider three category granularities (i.e., each hierarchical level), and $\eta = \{50, 30, 20\}$, for levels $\{1, 2, 3\}$ respectively. We quantify hotspot preservation by calculating the ‘hotspot distance’ between the hotspots in the perturbed and real data. If $\mathcal{H}$ and $\widehat{\mathcal{H}}$ are the hotspot sets in the real and perturbed data respectively, the average hotspot distance (AHD) between sets is:

$$AHD(\mathcal{H}, \widehat{\mathcal{H}}) = \frac{1}{|\widehat{\mathcal{H}}|} \sum_{\hat{h} \in \widehat{\mathcal{H}}} \min_{h \in \mathcal{H}} \left(|t_s - \hat{t}_s| + |t_e - \hat{t}_e| \right) \qquad (18)$$

Note that, for each perturbed hotspot, we calculate the hotspot distance to each real hotspot (for the same space-category granularity) and report the minimum value. This protects against cases

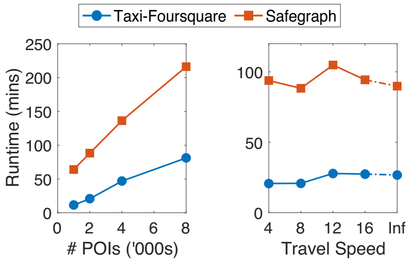


Figure 7: Pre-processing runtime costs

Table 2: Mean NE between real and perturbed trajectory sets

Method	Taxi-Foursquare			Safegraph			Campus		
	d_t	d_c	d_s	d_t	d_c	d_s	d_t	d_c	d_s
IndNoReach	1.44	3.81	1.99	1.47	2.87	2.14	2.06	1.35	0.87
IndReach	1.43	3.80	2.03	1.50	2.94	2.31	2.03	1.39	0.89
PhysDist	1.61	8.74	**1.85**	1.62	8.38	**2.10**	2.16	3.04	0.90
NGramNoH	1.63	4.25	2.07	1.62	3.37	2.33	2.14	1.46	0.88
NGram	**1.18**	**1.82**	2.24	**0.93**	**1.31**	2.12	**1.21**	**0.81**	**0.83**

in which there is not a one-to-one relationship between real and perturbed hotspots. We exclude hotspots in $\widehat{\mathcal{H}}$ for which there is no corresponding hotspot in $\mathcal{H}$. We also record the absolute difference between c and $\hat{c}$ for each hotspot pair. When averaged across all hotspots, we obtain the average count difference, ACD.

7 RESULTS

We compare our hierarchical solution to baseline methods in terms of NE and runtime. We also vary experimental and mechanism parameters, before evaluating on application-inspired queries.

7.1 Baseline Comparison

7.1.1 Normalized Error. Table 2 shows the distances between the real perturbed trajectories in all three dimensions. NGram is generally the best performing method across all datasets. Comparison with NGramNoH demonstrates that a hierarchical approach provides accuracy benefits as well as efficiency benefits (as we will see). The importance of including external knowledge, such as category information, is emphasized when comparing performance with PhysDist, which performs worse than all other methods. Performance gains are primarily achieved in minimizing the category distance between real and perturbed trajectories.

NGram has lower d_c and d_t values than all other methods, although it performs less well (comparatively) when analyzing d_s. This indicates that, although the category and time dimensions of the STC region merging seem well-suited, the spatial merging may be too coarse. Less merging in the spatial dimension would help to minimize accuracy losses here, although a moderate decrease in efficiency would have to be tolerated. Space limitations prohibit deeper analysis of different STC region merging approaches.

7.1.2 Runtime Analysis. Table 3 shows the average runtime of each perturbation method, including a breakdown of time spent on each stage of the mechanism. The 'Other' column incorporates overheads and mechanism stages unique to one perturbation method (e.g., time smoothing in IndNoReach and IndReach, or the POI-level reconstruction in NGramH). As expected, IndNoReach and IndReach are exceptionally quick as they rely solely on indexing operations. For the remaining mechanisms, the majority of the runtime is reserved for solving the optimization problem during the trajectory reconstruction phase. All other phases are performed in sub-second times. This demonstrates that even quicker results are feasible if time is spent selecting the best LP solver and tuning the optimization parameters – aspects of work that were beyond the scope of this paper. Importantly, however, NGram complements its accuracy superiority with efficiency prowess over NGramNoH and PhysDist, being nearly two and four times faster on average, respectively. The performance gain is primarily achieved from having a smaller optimization problem as a result of STC region merging.

7.2 Parameter Variation

We now examine how performance is influenced by the trajectory characteristics, and the mechanism or experiment parameters. Figures 8 and 9 show the variation in NE and runtime, respectively.

7.2.1 Trajectory Length. Figures 8a and 8e show an increase in error as trajectory length increases. NGram consistently outperforms other methods, which are broadly comparable in accuracy terms, with the exception of PhysDist. This is because ϵ' decreases as $|\tau|$ increases, which decreases the likelihood that the true n-gram is returned. Although the reconstruction stage seeks to minimize the effects of this, the reconstruction error is defined with respect to the *perturbed* n-grams (not the *real* n-grams), which limits the ability for the mechanism to correct itself. An alternative privacy budget distribution would be to assign a constant ϵ' value for each perturbation, but this means trajectories experience a different amount of privacy leakage (i.e., $\epsilon = (n + |\tau| - 1)\epsilon'$).

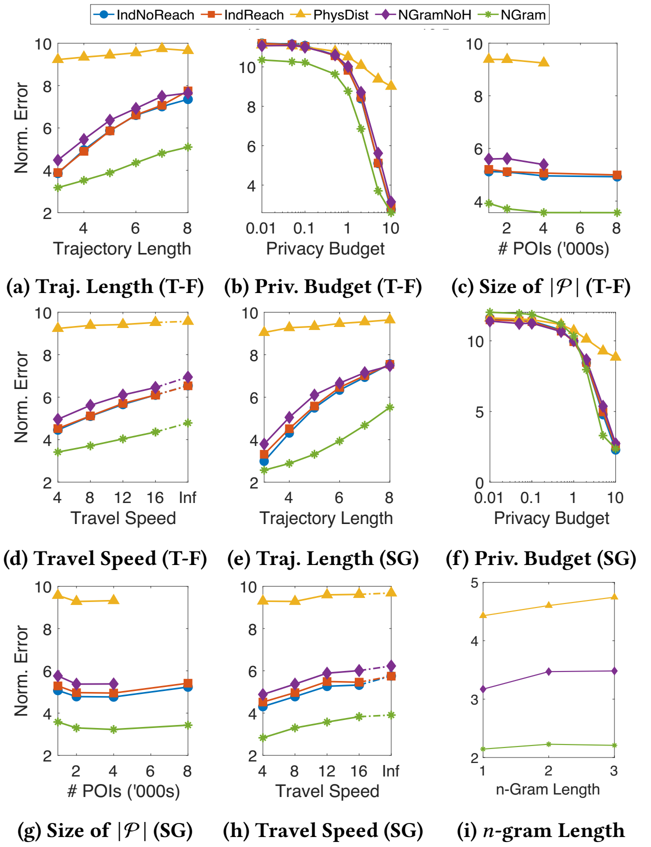
Figures 9a and 9e show how the runtime changes with trajectory length. As expected, IndNoReach and IndReach show little runtime variability. Of the optimization-based approaches, NGram is consistently the fastest method, and its rate of increase as $|\tau|$ increases is lower than other approaches. Finally, as most trajectories were less than eight POIs in length, NGram produces output trajectories in a reasonable time for the vast majority of trajectories.

7.2.2 Privacy Budget. Figures 8b and 8f show how NE is influenced by ϵ. All methods produce expected error profiles—as ϵ increases, the error decreases—although this behavior is less notable for PhysDist. When $\epsilon < 1$, the drop-off in utility is less pronounced. This behavior is likely to be indicative of the DP noise overwhelming the characteristics of the true data (i.e., the output data offers little value due to the added noise). Hence, we recommend setting $\epsilon \geq 1$ as a 'starting point' in applications of our solution.

Importantly, Figures 9b and 9f show that the runtime of NGram is relatively immune to the privacy budget, which indicates that the scale of the optimization problem is unaffected by the privacy budget. Most remaining methods also exhibit this immunity, although PhysDist does not, which further emphasizes the benefits of including external information in trajectory perturbation.

Table 3: Average runtime in seconds; breakdown by main mechanism stages; values of 0.000s indicate runtimes that are less than 10^{-3}s; sum of individual runtime stages may not equal 'Total' due to rounding

Method	Taxi-Foursquare					Safegraph				
	Perturb	Reconst. Prep.	Optimal Reconst.	Other	Total	Perturb	Reconst. Prep.	Optimal Reconst.	Other	Total
IndNoReach	**0.005**	–	–	0.714	0.720	0.006	–	–	0.786	0.791
IndReach	**0.005**	–	–	**0.000**	**0.006**	**0.005**	–	–	**0.000**	**0.006**
PhysDist	0.449	0.497	67.618	**0.000**	68.564	0.431	0.473	60.561	**0.000**	61.464
NGramNoH	0.446	0.561	30.872	**0.000**	31.879	0.426	0.509	24.389	**0.000**	25.325
NGram	0.056	**0.132**	**4.892**	0.502	5.582	0.126	**0.235**	**3.196**	0.178	3.735

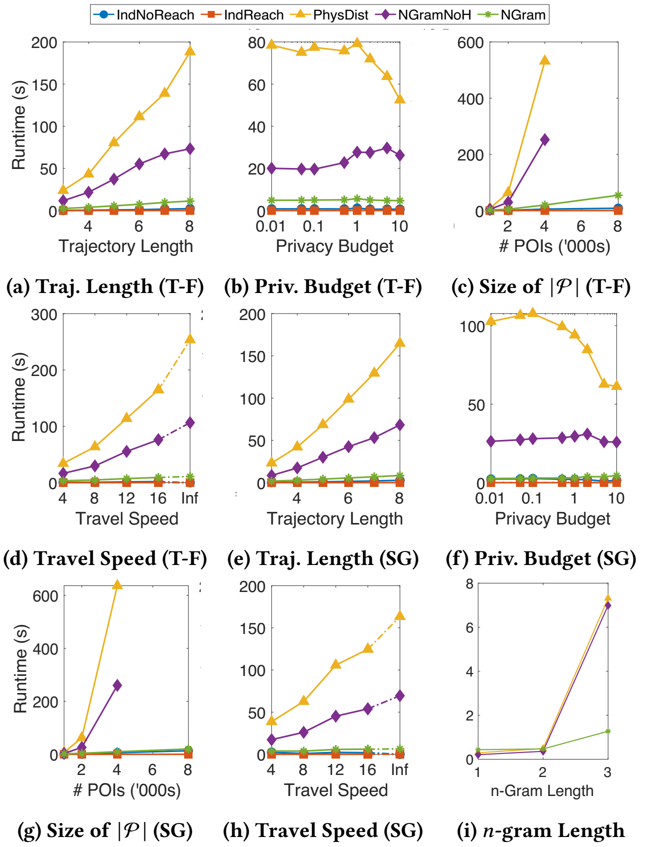


(a) Traj. Length (T-F) (b) Priv. Budget (T-F) (c) Size of $|\mathcal{P}|$ (T-F)

(d) Travel Speed (T-F) (e) Traj. Length (SG) (f) Priv. Budget (SG)

(g) Size of $|\mathcal{P}|$ (SG) (h) Travel Speed (SG) (i) n-gram Length

Figure 8: Normalized error as experimental settings vary; Figures 8a-8d use Taxi-Foursquare (T-F) data, Figures 8e-8h use Safegraph (SG) data, and Figure 8i uses Campus data



(a) Traj. Length (T-F) (b) Priv. Budget (T-F) (c) Size of $|\mathcal{P}|$ (T-F)

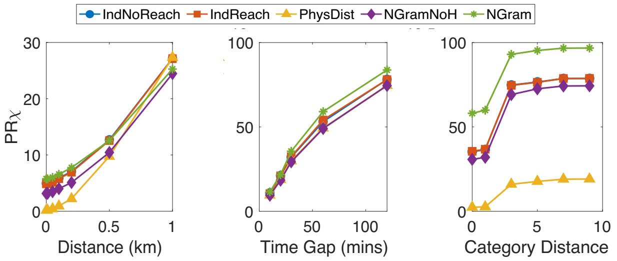
(d) Travel Speed (T-F) (e) Traj. Length (SG) (f) Priv. Budget (SG)

(g) Size of $|\mathcal{P}|$ (SG) (h) Travel Speed (SG) (i) n-gram Length

Figure 9: Average runtime as experimental settings vary; Figures 9a-9d use Taxi-Foursquare (T-F) data, Figures 9e-9h use Safegraph (SG) data, and Figure 9i uses Campus data

7.2.3 Size of POI Set. Figures 8c and 8g show the effect that $|\mathcal{P}|$ has on NE. We omit PhysDist and NGramNoH when $|\mathcal{P}|$ = 8,000, owing to their high runtime. Interestingly, the error profiles are relatively immune to the effects of changing $|\mathcal{P}|$. This suggests that the optimal reconstruction phase can effectively identify the best trajectory from the perturbed n-grams. Figure 9c and 9g show a moderate runtime increase for NGram, which still perturbs trajectories in a reasonable time, even for large POI sets. In all n-gram-based methods, at least 95% of runtime is spent during reconstruction, indicating the area of focus if substantial time-savings are necessary.

7.2.4 Reachability Constraint. We experiment with assumed travel speeds of $\{4, 8, 12, 16\}$km/hr, and we also consider imposing no reachability constraint (i.e., $\theta = \infty$). Error increases as the reachability constraint becomes less strict or is removed entirely (Figures 8d and 8h). This is because more n-grams are feasible and so the likelihood of the true n-gram being returned is reduced. NGram



(a) Space **(b) Time** **(c) Category**

Figure 10: Variation in PR_χ values as δ_χ changes

consistently outperforms all other methods in accuracy terms and, in terms of runtime, it is relatively immune to changes in assumed travel speed, unlike other n-gram approaches (Figures 9d and 9h). Importantly, NGram is up to 31% better than other methods when the reachability constraint is applied, and it remains up to 22% better than other methods when the reachability constraint is omitted.

7.2.5 n-gram Length. We consider n-grams of length $\{1, 2, 3\}$ for the three n-gram-based methods, using the Campus data. The NE and runtime results are shown in Figures 8i and 9i, respectively. NGram consistently outperforms other methods for all values of n, and, for NGram, $n = 2$ offers the best results. This is to be expected given the trade-off between capturing more information between neighboring points (achieved with high n) and the division of ϵ and sensitivity of the distance function (where low n is good). As expected and discussed in Section 5.8, runtime costs start to become undesirable when $n = 3$, supporting our recommendation that bigrams should be used in most real-world applications.

7.3 Application-Inspired Queries

Figure 10 shows the results for the PRQs for each dimension. For space and time PRQs, all methods perform similarly, although NGram slightly outperforms the other methods in general. There is a more notable difference in performance for category PRQs, with NGram clearly superior for all δ_c values. Interestingly, there is an evident step at $\delta_c = 3.5$, which suggests strong preservation of category within levels 2 and 3, which demonstrates robustness in our solution. The ability to preserve the general category of POIs indicates the solution's suitability for societal contact tracing as relevant agencies can, say, advise people who have recently visited sports stadia to monitor their health. Table 4 shows the AHD and ACD values under default settings (AHD is in hours, ACD has no units). NGram is much better than other methods in preserving the temporal location of hotspots. Again, PhysDist performs worst of all and the remaining methods are broadly comparable. Interestingly, NGram performs less well when considering ACD values. This suggests that, while hotspots are broadly preserved in time, they are 'flatter' in the perturbed trajectory sets. In practice, preserving the spatio-temporal location of hotspots will probably be more important to policy makers and researchers than preserving the hotspot strength (i.e., the maximum number of unique visitors).

Based on these results and those in Section 7.2.2, we recommend $\epsilon \geq 1$ for most practical applications.

Table 4: AHD and ACD values for default trajectory sets

Method	Taxi		Safegraph		Campus	
	AHD	ACD	AHD	ACD	AHD	ACD
IndNoReach	1.58	8.21	2.52	13.07	2.36	**15.72**
IndReach	1.72	9.64	2.54	**9.07**	2.54	17.83
PhysDist	2.22	10.76	3.34	16.24	4.38	23.48
NGramNoH	1.71	**9.36**	2.81	11.25	3.29	18.23
NGram	**1.49**	13.53	**2.01**	16.30	**2.03**	18.74

8 FURTHER WORK

We have developed an efficient and scalable n-gram-based method for perturbing trajectory data in accordance with LDP. However, there are a number of areas in which our solution can be extended, and these form the basis for future work for us and others. Although the external knowledge we use is limited to the data that is widely available, our framework can accommodate other data sources without difficulty. For example, temporally-varying POI popularity and POI-specific opening hours can be incorporated easily. Less-structured data (e.g., inferred popularity from public comments) could also be incorporated into the semantic distance function. We anticipate that incorporating more, richer data sources would further enhance utility, without negatively affecting efficiency. POI attributes can also be extended to a personalized LDP setting (e.g., the privacy level of a hospital differs if one is a doctor or patient).

Whereas the focus of this paper has been on devising a general approach for trajectory sharing, our solution can be adapted for specific applications or to consider the setting where single location points are shared continuously. Applications with specific utility aims may necessitate tuning of parameters or distance functions which is another direction of future work. Our problem framework and solution can also be applied to any notion of trajectory in space-time. To illustrate this, consider sharing shopping habits (e.g., credit card transactions). Here, $\mathcal{P}$ represents the set of purchasable products, with attributes such as category, price, etc. We can exploit intrinsic hierarchies such that $\mathcal{R}_c$ represents the set of stores from which products are purchased (which can be online or physical stores). The reachability constraint remains to ensure that adjacent stores in τ are reachable in the real world (as is currently done to identify and prevent credit card fraud). Online stores would always be 'reachable' given their non-physical presence. Other concepts, such as utility-enhancing semantic distance functions and the impossibility of some combinations (e.g., purchasing a car from a florist), translate naturally. Hence, this framework can be applied more generally.

ACKNOWLEDGMENTS

This work is supported by the UK Engineering and Physical Sciences Research Council (EPSRC) under Grant No. EP/L016400/1, European Research Council under Grant No. ERC-2014-CoG 647557, and by The Alan Turing Institute under EPSRC Grant No. EP/N510129/1. The authors would also like to acknowledge the University of Warwick Scientific Computing Research Technology Platform for assistance in the research described in this paper.

REFERENCES

[1] Jayadev Acharya, Keith Bonawitz, Peter Kairouz, Daniel Ramage, and Ziteng Sun. 2019. Context-Aware Local Differential Privacy. arXiv:1911.00038

[2] Gergely Acs and Claude Castelluccia. 2014. A Case Study: Privacy Preserving Release of Spatio-Temporal Density in Paris. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1679–1688. https://doi.org/10.1145/2623330.2623361

[3] Mario Alvim, Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Anna Pazii. 2018. Invited Paper: Local Differential Privacy on Metric Spaces: Optimizing the Trade-Off with Utility. In *IEEE Computer Security Foundations Symposium*. 262–267. https://doi.org/10.1109/CSF.2018.00026

[4] Miguel E. Andrés, Nicolás E. Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. 2013. Geo-indistinguishability: differential privacy for location-based systems. In *ACM SIGSAC Conference on Computer and Communications Security*. ACM, 901–914. https://doi.org/10.1145/2508859.2516735

[5] Raef Bassily, Kobbi Nissim, Uri Stemmer, and Abhradeep Thakurta. 2017. Practical Locally Private Heavy Hitters. In *International Conference on Neural Information Processing Systems*. 2285–2293. https://papers.nips.cc/paper/2017/file/3d779cae2d46cf6a8a99a35ba4167977-Paper.pdf

[6] Luca Bonomi and Li Xiong. 2013. A Two-Phase Algorithm for Mining Sequential Patterns with Differential Privacy. In *ACM International Conference on Information Knowledge Management*. 269–278. https://doi.org/10.1145/2505515.2505553

[7] US Census Bureau. 2020. *North American Industry Classification System*. Retrieved May 12, 2021 from https://www.census.gov/naics/

[8] Yang Cao, Masatoshi Yoshikawa, Yonghui Xiao, and Li Xiong. 2017. Quantifying Differential Privacy under Temporal Correlations. In *IEEE International Conference on Data Engineering*. 821–832. https://doi.org/10.1109/ICDE.2017.132

[9] Konstantinos Chatzikokolakis, Miguel E. Andrés, Nicolás Emilio Bordenabe, and Catuscia Palamidessi. 2013. Broadening the Scope of Differential Privacy Using Metrics. In *Privacy Enhancing Technologies*. 82–102.

[10] Rui Chen, Gergely Acs, and Claude Castelluccia. 2012. Differentially Private Sequential Data Publication via Variable-Length n-Grams. In *ACM SIGSAC Conference on Computer and Communications Security*. 638–649. https://doi.org/10.1145/2382196.2382263

[11] Rui Chen, Haoran Li, A. K. Qin, Shiva Prasad Kasiviswanathan, and Hongxia Jin. 2016. Private spatial data aggregation in the local setting. In *IEEE International Conference on Data Engineering*. 289–300. https://doi.org/10.1109/ICDE.2016.7498248

[12] Michael B. Cohen, Yin Tat Lee, and Zhao Song. 2018. Solving Linear Programs in the Current Matrix Multiplication Time. (2018). arXiv:1810.07896 http://arxiv.org/abs/1810.07896

[13] Graham Cormode, Somesh Jha, Tejas Kulkarni, Ninghui Li, Divesh Srivastava, and Tianhao Wang. 2018. Privacy at Scale: Local Differential Privacy in Practice. In *ACM SIGMOD International Conference on Management of Data*. 1655–1658. https://doi.org/10.1145/3183713.3197390

[14] Teddy Cunningham, Graham Cormode, and Hakan Ferhatosmanoglu. 2021. Privacy-Preserving Synthetic Location Data in the Real World. In *International Symposium on Spatial and Temporal Databases*. https://doi.org/10.1145/3469830.3470893

[15] Damien Desfontaines, Esfandiar Mohammadi, Elisabeth Krahmer, and David Basin. 2020. Differential privacy with partial knowledge. (2020). arXiv:1905.00650

[16] Foursquare Developers. 2020. *Venue Categories*. Retrieved May 24, 2021 from https://developer.foursquare.com/docs/build-with-foursquare/categories/

[17] Apple Differential Privacy Team. 2017. *Learning with Privacy at Scale*. https://machinelearning.apple.com/research/learning-with-privacy-at-scale

[18] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. 2017. Collecting Telemetry Data Privately. In *International Conference on Neural Information Processing Systems*. 3574–3583. https://papers.nips.cc/paper/2017/file/253614bbac999b38b5b60cae531c4969-Paper.pdf

[19] John C. Duchi, Michael I. Jordan, and Martin J. Wainwright. 2013. Local Privacy and Statistical Minimax Rates. In *IEEE Symposium on Foundations of Computer Science*. 429–438. https://doi.org/10.1109/FOCS.2013.53

[20] Cynthia Dwork. 2006. Differential Privacy. In *Automata, Languages and Programming*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1–12.

[21] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N. Rothblum. 2010. Differential Privacy under Continual Observation. In *ACM Symposium on Theory of Computing*. 715–724. https://doi.org/10.1145/1806689.1806787

[22] Cynthia Dwork and Aaron Roth. 2014. The Algorithmic Foundations of Differential Privacy. *Found. Trends Theor. Comput. Sci.* 9, 3–4 (2014), 211–407. https://doi.org/10.1561/0400000042

[23] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response. In *ACM SIGSAC Conference on Computer and Communications Security*. 1054–1067. https://doi.org/10.1145/2660267.2660348

[24] Fatima Zahra Errounda and Yan Liu. 2020. An Analysis of Differential Privacy Research in Location and Trajectory Data. (2020). https://assets.researchsquare.com/files/rs-94765/v1_stamped.pdf

[25] Xiaolan Gu, Ming Li, Yang Cao, and Li Xiong. 2019. Supporting Both Range Queries and Frequency Estimation with Local Differential Privacy. In *IEEE Conference on Communications and Network Security*. 124–132. https://doi.org/10.1109/CNS.2019.8802778

[26] Xiaolan Gu, Ming Li, Li Xiong, and Yang Cao. 2020. Providing Input-Discriminative Protection for Local Differential Privacy. In *IEEE International Conference on Data Engineering*. 505–516. https://doi.org/10.1109/ICDE48307.2020.00050

[27] Mehmet Emre Gursoy, Ling Liu, Stacey Truex, and Lei Yu. 2019. Differentially Private and Utility Preserving Publication of Trajectory Data. *IEEE Transactions on Mobile Computing* 18, 10 (2019), 2315–2329.

[28] Mehmet Emre Gursoy, Vivekanand Rajasekar, and Ling Liu. 2020. Utility-Optimized Synthesis of Differentially Private Location Traces. (2020). arXiv:2009.06505

[29] Xi He, Graham Cormode, Ashwin Machanavajjhala, Cecilia M Procopiuc, and Divesh Srivastava. 2015. DPT: differentially private trajectory synthesis using hierarchical reference systems. *PVLDB* 8, 11 (2015), 1154–1165.

[30] Bo Jiang, Ming Li, and Ravi Tandon. 2018. Context-aware Data Aggregation with Localized Information Privacy. In *IEEE Conference on Communications and Network Security*. 1–9. https://doi.org/10.1109/CNS.2018.8433200

[31] Hongbo Jiang, Jie Li, Ping Zhao, Fanzi Zeng, Zhu Xiao, and Arun Iyengar. 2021. Location Privacy-Preserving Mechanisms in Location-Based Services: A Comprehensive Survey. *ACM Comput. Surv.* 54, 1, Article 4 (2021). https://doi.org/10.1145/3423165

[32] Georgios Kellaris, Stavros Papadopoulos, Xiaokui Xiao, and Dimitris Papadias. 2014. Differentially Private Event Sequences over Infinite Streams. *PVLDB* 7, 12 (2014), 1155–1166. https://doi.org/10.14778/2732977.2732989

[33] Daniel Kifer and Ashwin Machanavajjhala. 2012. A Rigorous and Customizable Framework for Privacy. In *ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. 77–88. https://doi.org/10.1145/2213556.2213571

[34] Eric Lantz, Kendrick Boyd, and David Page. 2015. Subsampled Exponential Mechanism: Differential Privacy in Large Output Spaces. In *ACM Workshop on Artificial Intelligence and Security*. 25–33. https://doi.org/10.1145/2808769.2808776

[35] Chao Li, Balaji Palanisamy, and James Joshi. 2017. Differentially Private Trajectory Analysis for Points-of-Interest Recommendation. In *IEEE International Congress on Big Data*. 49–56.

[36] Ninghui Li, Wahbeh Qardaji, Dong Su, Yi Wu, and Weining Yang. 2013. Membership Privacy: A Unifying Framework for Privacy Definitions. In *ACM SIGSAC Conference on Computer Communications Security*. 889–900. https://doi.org/10.1145/2508859.2516686

[37] Ashwin Machanavajjhala and Xi He. 2018. Analyzing Your Location Data with Provable Privacy Guarantees. In *Handbook of Mobile Data Privacy*, Aris Gkoulalas-Divanis and Claudio Bettini (Eds.). Springer International Publishing, Cham, 97–127. https://doi.org/10.1007/978-3-319-98161-1_5

[38] Ryan McKenna and Daniel R Sheldon. 2020. Permute-and-Flip: A new mechanism for differentially private selection. In *Advances in Neural Information Processing Systems*, Vol. 33. 193–203. https://proceedings.neurips.cc/paper/2020/file/01e00f2f4fbcbb7505cb641066f2859b-Paper.pdf

[39] Frank McSherry and Kunal Talwar. 2007. Mechanism Design via Differential Privacy. In *IEEE Symposium on Foundations of Computer Science*. 94–103. https://doi.org/10.1109/FOCS.2007.41

[40] University of British Columbia. 2016. *ubcv-buildings*. Retrieved May 12, 2021 from https://github.com/UBCGeodata/ubcv-buildings

[41] Safegraph. 2020. *Safegraph Places Schema*. Retrieved December 7, 2020 from https://docs.safegraph.com/v4.0/docs/places-schema

[42] New York City Taxi and Limousine Commission. 2013. *TLC Trip Record Data*. Retrieved March 2, 2013 from https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page

[43] Jan van den Brand. 2020. A Deterministic Linear Program Solver in Current Matrix Multiplication Time. In *ACM-SIAM Symposium on Discrete Algorithms*. 259–278.

[44] Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha. 2017. Locally Differentially Private Protocols for Frequency Estimation. In *USENIX Security Symposium*. 729–745.

[45] Dingqi Yang, Daqing Zhang, Vincent. W. Zheng, and Zhiyong Yu. 2015. Modeling User Activity Preference by Leveraging User Spatial Temporal Characteristics in LBSNs. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 45, 1 (2015), 129–142.