WARWICK
THE UNIVERSITY OF WARWICK

Generating and Sharing Differentially Private Spatio-Temporal Data Using Real-World Knowledge

by

Teddy Cunningham

Thesis

Submitted to the University of Warwick in partial fulfilment

of the requirements for admission to the degree of

Doctor of Philosophy

in

Computer Science

Department of Computer Science

July 2022

Contents

List of Tables

List of Figures

Acknowledgements

My first thanks go to Hakan Ferhatosmanoglu for his dedicated and enthusiastic supervision over the past four years. Because of you, I have learned more than I ever expected to, and I have become a better researcher. You have also given me the opportunity to review papers, help write bids for funding, and work as part of collaborative projects, all of which have made my PhD experience even more enriching.

I would also like to thank my long-term collaborators, Divesh Srivastava and Graham Cormode, who have both helped to make my research immeasurably better. I have always enjoyed our regular meetings, and it has been a real privilege to collaborate with two world-leading experts. I am also grateful to Konstantin Klemmer and Hongkai Wen, who have provided invaluable help for parts of this research. Thank you also to my examiners, Carsten Maple and Li Xiong, for their positive and constructive feedback.

Even though a pandemic kept us apart for most of our PhDs, I must also thank Chris Conlan, Aparajita Haldar, Nicole Hengesbach, Jon Davies, Matteo Mazzamurro, and everyone else I have met during my time at Warwick. The strength of our collective friendship has made this journey so much more enjoyable, and I continue to miss being in the office with all of you.

I wish to thank all of my friends and family for their support over the past four years. Even if they have not always really known what my research has been about, they have always shown an interest, and that is all I can ask for. My final thanks must go to Jenna, whose continual love, support, and encouragement has helped me to persevere through to the end.

Declarations

I, Teddy Cunningham, declare that the work included in this thesis is my own work, unless otherwise stated below. This thesis has not been submitted for a degree at any other university.

This work was primarily funded by the UK Engineering and Physical Sciences Research Council (EPSRC) under Grant No. EP/L016400/1. Additional funding was also provided by the European Research Council under Grant No. ERC-2014-CoG 647557, and by The Alan Turing Institute under EPSRC Grant No. EP/N510129/1.

Parts of this thesis have been previously published by the author in the following:

[68] Teddy Cunningham, Graham Cormode, and Hakan Ferhatosmanoglu. Privacy-Preserving Synthetic Location Data in the Real World. In *SSTD*, pages 23–33, 2021. doi:10.1145/3469830.3470893

[69] 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

[71] Teddy Cunningham, Konstantin Klemmer, Hongkai Wen, and Hakan Ferhatosmanoglu. GeoPointGAN: Synthetic Spatial Data with Local Label Differential Privacy. 2022. doi:10.48550/ARXIV.2205.08886[1]

[67] Teddy Cunningham. Sharing and Generating Privacy-Preserving Spatio-Temporal Data Using Real-World Knowledge. In *IEEE MDM*, pages 331–333, 2022. doi:10.1109/MDM55031.2022.00074

Other parts of this thesis are currently under review:

[70] Teddy Cunningham, Hakan Ferhatosmanoglu, and Divesh Srivastava. Sharing and Integrating Event Sequences from Multiple Sources with Local Differential Privacy. 2022. Currently under review.

[1]Teddy Cunningham and Konstantin Klemmer are joint first-authors of this article. Konstantin's contributions focus on developing the architecture of GeoPointGAN. Teddy's contributions focus on developing the privacy mechanism for GeoPointGAN. Both contributed equally to writing the manuscript and conducting the experiments.

The following research was performed in collaborations during the development of this thesis, but does not form part of the thesis:

[59] Chris Conlan, Teddy Cunningham, Gunduz Vehbi Demirci, and Hakan Ferhatosmanoglu. Collective Shortest Paths for Minimizing Congestion on Temporal Load-Aware Road Networks. In *ACM SIGSPATIAL Workshop on Computational Transportation Science*, pages 1–10, 2021. doi:10.1145/3486629.3490691

[106] Susan M. Grant-Muller, Mahmoud Abdelrazek, Hannah Budnitz, Caitlin D. Cottrill, Fiona Crawford, Charisma F. Choudhury, Teddy Cunningham, Gillian Harrison, Frances C. Hodgson, Jinhyun Hong, Adam Martin, Oliver O'Brien, Claire Papaix, and Panagiotis Tsoleridis. Technology Enabled Data for Sustainable Transport Policy. In *International Encyclopedia of Transportation*, pages 135–141. Elsevier, 2021. doi:10.1016/B978-0-08-102671-7.10627-X

[114] Aparajita Haldar, Teddy Cunningham, and Hakan Ferhatosmanoglu. RAGUEL: Recourse-Aware Group Unfairness Elimination. In *ACM CIKM*, pages 666–675, 2022. doi:10.1145/3511808.3557424

[60] Chris Conlan, Teddy Cunningham, Samuel Watson, Jason Madan, Alex Sfyridis, Jo Sartori, Hakan Ferhatosmanoglu, and Richard Lilford. Perceived Quality of Care and Choice of Healthcare Provider in Informal Settlements. 2022. Currently under review.

Abstract

Privacy-preserving spatio-temporal data sharing is vital for addressing many real-world problems, such as managing disease spread or tailoring public services to a population's travel patterns. Differential privacy has become the de facto privacy standard owing to its strong privacy guarantees, although existing mechanisms make very restrictive assumptions regarding what outside knowledge is known beyond the data itself. This limits the practical utility of the private data, and has prevented the widespread deployment of differentially private algorithms in the real world. This thesis aims to show that incorporating publicly available information, such as the road network or characteristics of places of interests, can enhance the practical utility of the output data without negatively affecting privacy.

This thesis focuses on two main problems, both of which are fundamental in enabling location analytics with private data. The first considers the synthesis of spatial point data, and three solutions are proposed. The first solution uses a private adaptation of kernel density estimation to generate data within small private partitions, and the second uses the road network as the basis for data generation. The third solution combines randomised response with generative adversarial networks to develop a generative model that satisfies label local differential privacy – a more practical and realistic privacy setting. The second problem focuses on sharing trajectory data using local differential privacy. The proposed solution uses the exponential mechanism to efficiently perturb overlapping, hierarchically structured *n*-grams of trajectory data, which help to preserve the spatio-temporal correlations inherent in trajectory data. This problem, and its solution, is then extended to a setting in which two services wish to privately share event sequence data with each other.

All solutions incorporate publicly available external knowledge by imposing hard constraints on feasible outputs, exploiting the intrinsic hierarchies and underlying structures of real-world data, and using distance functions to ensure that semantically similar values are more likely to be output. Experiments with real data show that including this information helps to produce private data that performs very well in many spatio-temporal analytical tasks, including range, hotspot, and facility location queries. These strong results demonstrate the potential for more widespread use of differential privacy in the real world.

Acronyms

ACD	Average Count Difference
AHD	Average Hotspot Distance
CD	Chamfer Distance
DP	Differential Privacy
EMD	Earth Mover's Distance
GAN	Generative Adversarial Network
KDE	Kernel Density Estimation
LDP	Local Differential Privacy
LLR	Lower Level Region
MAE	Mean Absolute Error
MBR	Minimum Bounding Rectangle
MCC	Matthews Correlation Coefficient
MEDD	Mean Edge Distance Difference
MPE	Mean Percentage Error
MSD	Mean Semantic Distance
NCE	Normalised Cell Error
POI	Place of Interest
RQ	Research Question
SDC	Sørensen-Dice Coefficient
STC	Space-Time-Category
STN	Spatial Transformer Network
ULR	Upper Level Region
WUD	Weighted Uniform Distribution

Symbols

A	Area of a region
$\mathcal{A}$	Set of attributes
B	Batch size
$\mathcal{B}$	Bipartite graph
c	Number of points in a region; in Ch. 6, the number of events in a lower level region
C	Number of points in an upper level region
$d(\cdot,\cdot)$	Distance function
$d_{\|}$	Distance between a point's projection onto its nearest edge and the edge's vertex
$d_{\perp}$	Perpendicular distance between a point and its nearest edge
D	Discriminator
$\mathcal{D}$	Dataset
e	Edge
E	Event period
$\mathcal{E}$	Set of edges
f	Candidate facility
$\widehat{\mathrm{f}}$	Kernel density estimator
F	Cumulative distribution function
$\mathcal{F}$	Set of candidate facilities
g	Granularity (h = hotspot, t = time, s = space, c = category)
G	Generator
$\mathcal{G}(\mathcal{E},\mathcal{V})$	Road network graph
h	Spatio-temporal hotspot
$\tilde{h}$	Smoothing parameter used in the construction of kernel density estimator
H	Number of histogram bins
$\mathcal{H}$	Set of hotspots
J	Number of boundaries of a region
k	Number of selected facilities
K	Number of partitioned regions
l	Label attached to a point
$\mathcal{L}$	Set of labels
m	Number of points in a neighbouring region
M	Grid size (in one direction)
$\mathcal{M}$	Randomised mechanism
n	Number of points in a region; in Ch. 5 and Ch. 6, the length of an n-gram

N	Number of points in a dataset
N_p	Number of perturbations
N_{steps}	Number of training steps
p	Real point
$\mathcal{P}$	Set of real points
$q(\cdot, \cdot)$	Quality function
Q	Quantile
r	Parameter of the kernel function; in Ch. 6, a lower level region
R	Region; in Ch. 6, an upper level region
$\mathcal{R}$	Set of regions
$\mathcal{RL}$	Set of lower level regions
$\mathcal{RU}$	Set of upper level regions
s	Synthetic point
$\mathcal{S}$	Set of synthetic points
t	Timestep; in Ch. 2, a tuple
T	Set of timesteps
$\mathcal{T}$	Set of trajectories
u	User
$\mathcal{U}$	Set of users
v	Vertex
$V(G, D)$	Min-max game between generator and discriminator
$\mathcal{V}$	Set of vertices
w^n	n-gram; in Ch. 6, a lower level region n-gram
W^n	Upper level region n-gram
$\mathcal{W}^n$	Set of space-time-category region n-grams
$\mathcal{WL}^n$	Set of lower level region n-grams
$\mathcal{WU}^n$	Set of upper level region n-grams
$\mathbf{x}$	2- or 3-dimensional vector of co-ordinates
X	Random variable representing uniform distribution of real points
Y	Random variable representing the kernel distribution
z	Event
$\mathbf{z}$	2- or 3-dimensional vector of random noise
α	Probability of returning some output
β	Probability of returning some output
γ	Threshold on the number of samples taken when reconstructing a trajectory
Γ	Number of constraints in trajectory reconstruction optimisation problem
δ	Cross-over penalty imposed when integrating data from two services
Δ	Global sensitivity
ϵ	Privacy budget
ζ	Error term when reconstructing trajectory
η	Minimum count for a spatio-temporal hotspot to exist
θ	Parameter of the kernel function
Θ	Set of parameters used to define generator or discriminator
κ	Minimum number of places of interest when merging space-time-category regions
λ	Parameter used to model event periods

Λ	Threshold on the number of times a real point can be sampled
μ	Mean value of a distribution
ξ	Constant used to define the utility of the exponential mechanism
π	Place of interest
Π	Set of places of interest
ρ	Range query radius
ϱ	Average percentage of reachable places of interest
σ	Service
ς	Indicator variable used to define preservation range query utility measure
Σ	Set of services
τ	Trajectory
$\overline{\tau}$	Holistic trajectory
τ^*	Composite trajectory
Υ	Semantic distance threshold when integrating data between services
ϕ	Bijection used to define earth mover's distance
φ	Threshold for the amount of noise added to edges
Φ	Kernel function
χ	Semantic dimension (t = time, s = space, c = category)
ψ	Reachability threshold
Ψ	Parameter of the subsampled exponential mechanism
Ω	Set of all possible global trajectories
ω	Weighting factor

Chapter 1

Introduction

Each day people interact with many services (e.g., banks, social media, or shops), each of which collect data on our interactions with them. And, in a digital world in which mobile technologies are ubiquitous, these services are collecting this data to an increasingly larger extent, and with ever-growing ease. This data is highly valuable to the services as it allows them to learn more about their users, and it enables them to improve their service. For example, if a mapping platform knows where users wish to start and end their trips, they can develop collective routing algorithms that minimise traffic congestion [59]. Similarly, if a bank knows how different customer demographic groups spend their money, they can use this information to tailor their range of loans, mortgages, and current accounts. Sharing data with other entities (e.g., other businesses, researchers, or the public) also has its benefits. For example, if a bank suspects that a transaction is fraudulent, it can seek data from other location-based services to determine whether the transaction in question is concordant with other events of the user. Sharing data openly with the public can also have multiple societal benefits, including improved healthcare [66, 160], increased transparency and accountability [156, 163], and improved transport systems [32].

However, despite these compelling use cases, the data that services collect on users is extremely private, for numerous personal, social, and financial reasons, and the risks of violating an individual's privacy presents a major impediment to the free sharing of such data. Several techniques have been proposed, and used, to provide a degree of privacy, whilst accommodating the desire for greater data sharing. Most of the early techniques, such as k-anonymity [211], l-diversity [158], and t-closeness [151], typically anonymise, generalise, omit, and/or perturb database values or records such that the shared database contains no information that would allow users to be identified directly. While these techniques can be effective from a privacy perspective (especially if the sanitised database is viewed in isolation), the effect on the database's ultimate utility can be severe. If, for example, the exact co-ordinates of people's locations must be generalised to the postcode-level, this limits the range of analyses that can be conducted using the data. Even when a good balance between privacy and utility is achieved, none of these techniques offer guarantees of privacy, and many have been shown to be vulnerable to homogeneity, background knowledge, and linkage attacks that ultimately lead to deanonymisation [72] or re-identification of sensitive values [155]. This has meant that high-profile data leakage cases have continued to occur in recent years, including the release of former Massachusetts governor William Weld's medical data [210], the deanonymisation of Netflix users [175], the publication of actor Bradley Cooper's tipping tendencies [223], and the identities of those who stormed the US Capitol in 2021 [215].

Differential privacy (DP) was proposed by Dwork et al. [83], in part, to address these deficiencies, and it has since become the de facto privacy standard in academia (and increasingly in industry) owing to its strong mathematical guarantees of privacy. In short, it ensures that every individual in a dataset has plausible deniability regarding their inclusion in that dataset. It is typically achieved by a trusted aggregator adding carefully curated noise to the answers of statistical queries (e.g., count, sum, average). Its simplicity and mathematical rigour has resulted in several real-world applications, including by Uber [130] and for the 2020 United States Census [3, 228].

One limitation of the original form of DP is that it relies on users providing their true data to a trusted aggregator, which may not always be realistic or desirable, especially when very sensitive data (e.g., medical, location, or financial data) is shared. Local differential privacy (LDP) [81] offers a decentralised alternative in which users privatise their own data before sharing it with an untrusted aggregator. This affords every individual plausible deniability with respect to the actual data included in the dataset. Owing to its decentralised setting, LDP typically requires a higher degree of noise to ensure that its privacy goals are met. This means that, to achieve high utility, very large datasets are normally necessary, which has limited real-world applications of LDP to big technology companies, such as Apple [13] and Google [86].

Despite their rigorous privacy guarantees and real-world applicability, many DP and LDP algorithms make very restrictive assumptions regarding what outside knowledge is known beyond the data itself (e.g., provenance, structure, or hierarchy), even though context-aware privacy has been shown to offer better privacy-utility trade-offs [124]. This is a major shortcoming, especially as a wealth of accessible, open source information about the real world exists: detailed mapping data describes roads and places of interest; transit schedules; business opening hours; and unstructured user-generated data, in the form of reviews, check-ins, photos, and videos. These sources provide a rich and detailed (if somewhat non-uniform) description of the real world within which people navigate their lives. Excluding this wealth of knowledge can even harm utility if a mechanism is oblivious to real-world conditions (e.g., a synthetic data generator could 'locate' people in the middle of the ocean, as it does not know the location of bodies of water). As this knowledge is publicly available, it can be used by adversaries to compromise privacy, yet current literature fails to utilise this same information to enhance utility, which limits its practicality in analytics tasks.

Given the increased societal desire for data to be shared, DP's ability to provide strong guarantees of privacy, and the increasing use of DP in real-world settings, it is intriguing that most existing algorithms fail to actively incorporate publicly available real-world knowledge. This thesis examines this issue and proposes methods for sharing private data in both the centralised and local DP settings, while utilising public knowledge to enhance utility without affecting the privacy guarantee.

1.1 Research Overview

With this broad motivation in mind, the scope, aims, and content of this thesis can now be outlined.

Although algorithms for privately sharing several types of data have been proposed, including medical [92], internet search history [86], and financial data [2], this thesis primarily focuses on spatio-temporal data. This decision can be justified by considering that, not only are spatio-temporal databases (e.g., taxi trajectories, public transport smartcard data) widespread, but many other databases (e.g., social media, shopping, bank transactions) contain spatio-temporal information also. For example, many social media posts are geotagged, social media users often check-in to places they visit, and all in-person credit card transactions record the time and place of purchase. This ubiquity of spatio-temporal data makes it an important and compelling domain upon which to focus.

Spatio-temporal data itself can be divided into two segments: point data and trajectory[1] data. Similarly, private data sharing can be divided into two main areas of research: publication and synthesis. Combining these gives four potential areas of study, each of which have their own challenges, applications, and research gaps. This thesis primarily focuses on two areas: synthesis of spatial point data, and publication of trajectory data. The reasons for these choices are explained by substantiating the following three research questions (RQ), and by assessing the current state of the art, which is surveyed in Chapter 2.

RQ1: How can spatial point data be generated privately, such that the synthetic data is practical and useful?

In order to preserve the privacy of individuals, sensitive location data typically has to be sanitised before it is published. This can involve aggregation into pre-defined regions, location perturbation, or truncation of longitude-latitude data. The sanitisation operation is normally controlled and performed by the data owner, whose primary concern is to minimise the privacy risk to the data subjects and their consequent liability. In many cases, this considerably limits the utility of the published data.

In contrast to crude sanitisation, releasing a synthetic dataset in the same format as the original data can give more flexibility in how clients can use the published data. In many practical scenarios, the recipient of the data will want to use their own data analytics tools without any restrictions from the data provider on the way in which the data can be used, or the type of queries used. Hence, the aim of this research question is to develop approaches for privately generating realistic, usable synthetic point data from real location data.

Although methods for generating differentially private trajectories have been proposed [e.g., 110, 112, 119], and this appears to be a more complex variant of this research question, many of the solutions therein all produce outputs that correspond to arbitrary grid cells, which is not concordant with the format of the original data. While one could extend these solutions to generate individual points (e.g., by using uniform sampling), generating

[1]Throughout this thesis, the term ‘trajectory’ broadly refers to any temporally ordered set of events or points in space-time. This differs from other, stricter notions in which a trajectory is a set of spatio-temporal data points with a very small sampling rate (e.g., less than ten seconds).

a spatial point dataset that maintains privacy, has high practical utility, and preserves the underlying distributions of the original dataset is a non-trivial exercise (as we will see). In any case, generating large-scale, private spatial point datasets is a core problem in spatial data management, and its importance and practicality warrants its study.

As a decentralised privacy model can provide even better protection to individuals, it is important to extend the scope of this research question to the local setting. This ensures that practical mechanisms with meaningful levels of privacy and security can be developed for the important problem of synthesising spatial point data.

RQ2: How can trajectory information be shared by individuals in a locally differentially private manner?

Although highly granular spatial point data has many uses, individual points can only tell part of a story, whereas trajectory data allows for a richer understanding of a population's movements and interactions. Privately sharing trajectories also has concrete 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. Although spatio-temporal trajectory synthesis has been addressed in the centralised setting, no prior work has addressed it in the local setting (in which publication and synthesis can be seen to be quasi-identical). Hence, RQ2 focuses on publishing trajectories/location sequences such that the requirements of local differential privacy are met.

Publishing trajectory data is not without its challenges. In particular, spatio-temporal data can have strong correlations between attributes of the same record, and between adjacent points in the trajectory. These correlations and patterns are difficult to protect from a privacy perspective, and difficult to preserve from a utility perspective. Furthermore, as LDP requires noise to be added to the data itself, 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.

The utility goals in RQ2 focus on both the user and the trajectory level, each of which have their own practical value. High utility at the aggregate level enables a range of queries with socially beneficial applications, such as societal contact tracing (e.g., where and when will the next 'superspreader' events occur) and facility location (e.g., where should a business owner build their new cinema). Maintaining high utility at the trajectory level gives end users the flexibility to use the output data effectively for their own unique purposes.

RQ3: How can trajectory information be shared between services in a practical and locally private manner?

Most existing private data sharing mechanisms only consider that services share data in isolation, meaning that services and end users only gain a partial picture of each individual in the population. For example, a transit agency knows when and where a user is travelling, but not why they are travelling or what they do at their destination. Moreover, in many cases, the same event is recorded by multiple services, but each service collects a different subset of data on the event. Having partial data like this is restrictive, as it heavily limits the utility of the data and the range of analyses that can be conducted, as well as introducing the potential for missed (or incorrect) inferences and insights. As such, the aim of this research question is to build on RQ2 by tackling the issue of how a user can privately share their event sequence data from one service (the donating service) with another service (the receiving service). The receiving service can then integrate this private data with trajectory data that it may already have on a user.

Sharing private data with other services in this way can enhance services and benefit users further. For example, consider a series of bank transactions, one of which is flagged as potentially fraudulent. With no other knowledge, the bank has limited information to confirm whether the event is likely to be fraudulent and therefore may not block the user's card to prevent other fraudulent transactions. However, the bank could use data from other services to check if there is concordance with the suspicious transaction (i.e., if the other services all locate the user in another city at the same time, the transaction is likely to be fraudulent).

While this research question possesses many of the same challenges as RQ2 with regard to dealing with trajectory data, it has two key challenges that are unique to the multiple service setting. The first challenge is to develop a data sharing protocol that gives the user control over their data, provides strong privacy, and achieves high practical utility. The second challenge is the task of integrating data from multiple services. In particular, as both services may collect data on the same event (but different attributes), it is crucial that this link is preserved when integrating data. A complementary challenge is ensuring that events that are not linked in reality are not linked when the two services integrate their data. Given that one half of the data being used in this integration will be perturbed in accordance with LDP, the difficulty of this task is evident.

Finally, this research question focuses on sharing trajectory data from multiple different services, not all of which will exclusively collect spatio-temporal data. As such, RQ3 requires the development of a general, all-purpose trajectory sharing solution, rather than one that is specifically designed for spatio-temporal data.

1.2 Solution Objectives

Having considered the motivation, scope, and proposed research questions of this thesis, we now turn to the main properties that are desirable for the proposed solutions. While some of these properties are desirable for any (private) algorithm, this thesis' focus on developing solutions that are practical for real-world application gives each a new context.

Provide strong, practical levels of privacy

As noted already, DP and LDP offer strong privacy guarantees, and it is reasonable to expect that any solution should use them as the base privacy model. However, just as many DP algorithms can be restrictive in how they treat external knowledge, many private data sharing solutions use unnecessarily restrictive privacy settings. Hence, it is prudent for solutions to use a privacy setting, or privacy model, that reflects the real-world problems they seek to address, whilst still providing DP-like guarantees.

Operate effectively at large scales

Naturally, any computational algorithm aims to be efficient and scalable. This is especially important for ones that seek to be applied to real-world, large-scale problems in which millions of records are shared. Scalability manifests itself in several ways in this thesis. Most notably, the solutions must be able to accommodate large data sets and be applied to urban-scale problems. Incorporating real-world data must also be scalable, especially in terms of the resources (e.g., time, cost, availability, labour) needed to collate and format the external knowledge. Any solution should also have a practical operational set-up that is conscious of the privacy requirements inherent to the problem. For example, a solution should consider what data is transferred between users and aggregators, and how this is implemented. Where appropriate, these aspects are considered (at a high level) to ensure that the proposed solutions can be implemented realistically.

Utilise real-world data to enhance utility

A main motivation of this thesis is to develop algorithms that actively incorporate real-world knowledge with the aim of enhancing practical utility. It is natural, therefore, to expect that any proposed solution should seek to do this, and this can be achieved in two main ways. First, solutions should seek to utilise all types of publicly available external knowledge (e.g., pre-defined schemas, historic data, commonsense knowledge). Second, solutions should exploit this data in several ways, including through hard constraints and by influencing the data synthesis or publication mechanisms. Together, these approaches should realise the full potential of real-world knowledge.

Preserve the underlying characteristics of the true data

In synthesising or publishing any data privately, values in the output dataset will differ (possibly significantly) from values in the input dataset. While this is to be expected, there is an implicit limit at which point the datasets are too different, and any meaningful level of utility is lost. Hence, any solution should aim to preserve the underlying characteristics present in the original data, which will differ depending on the data or problem setting. For example, when generating point data, preserving the overall distribution of points is key, whereas preserving spatio-temporal correlations between adjacent points is important when dealing with trajectory data.

Demonstrate strong performance in real-world application queries

The ultimate goal in designing algorithms that are fit for real-world use is to offer strong performance in analytics tasks in the real-world itself. These tasks range from simple, yet abstract, range and hotspot queries to more complex applications, such as facility location and fraud detection. If a mechanism can output private data that performs as well as the true data in these queries, it demonstrates that these objectives have, as a whole, been met.

Caveat

As Thomas Cromwell was once told, "a man can not have his cake and eat his cake" [121]. Privacy researchers must acknowledge a similar trilemma when developing algorithms that seek to provide high levels of privacy and utility, whilst remaining efficient. While any two of these three aspects can be obtained trivially, achieving an appropriate balance between privacy, utility, *and* efficiency is decidedly non-trivial, as we will see. Nevertheless, this thesis will show that incorporating real-world knowledge can make this challenge somewhat easier to address.

1.3 Contributions

This thesis offers several substantial contributions to the field, which are briefly summarised here.

Chapter 3 presents two robust methods for generating synthetic spatial data using centralised DP, both of which use real-world data to enhance the practical utility of the output data. The first method combines existing private partitioning methods with novel private data generation methods to generate spatial data within known geographic boundaries. These data generation methods include a new mechanism for differentially private kernel density estimation that is designed for our use-case: multiple point sampling for synthetic data generation. The second method goes further and explicitly uses the road network to control the data generation process. Specifically, private micro-histograms are composed along each edge in the road network to effectively model the distribution of real points. Sampling from these noisy micro-histograms allows high quality synthetic data to be generated. Both of these methods contribute towards answering RQ1.

RQ1 is also addressed in Chapter 4 with the proposal of GeoPointGAN, which uses a generative adversarial network (GAN) to generate spatial point data. The advantage of this solution is that it gives more flexibility to the end user as they can create datasets of any size, which is a functionality not directly provided with the mechanisms of Chapter 3. Furthermore, GeoPointGAN includes a novel 'label-flipping' mechanism, inspired by the randomised response mechanism of LDP, to satisfy label-LDP, which is proposed as a more practical variant of LDP. This local setting confers a higher level of privacy to individual users than is achieved in Chapter 3. Finally, privately flipping the labels of true and fake points offers potential regularisation and generalisation benefits that enable GeoPointGAN to be robust and, intriguingly, achieve a certain level of privacy for free.

Chapter 5 proposes two solutions for sharing sequences of places of interest using LDP, as set out in RQ2. The first mechanism models trajectories as individual points in high-dimensional space, and can be seen as the elegant, 'global' solution. However, its high time and space complexity makes it computationally infeasible in most scenarios, which leads us to the central contribution of this chapter: a scalable, efficient mechanism 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. The mechanism also uses a semantic distance function to incorporate a rich set of public knowledge to adjust the probability of certain perturbations in a utility-enhancing manner. Exploiting the (publicly known) hierarchies that are inherent in space, time, and category classifications to structure n-grams in a multi-dimensional hierarchy has notable benefits for utility. Doing so also reduces the scale of the problem, ensuring that the solution is scalable for large urban datasets.

Chapter 6 tackles the problem of privately sharing and integrating event sequences between two services, as outlined in RQ3. The proposed solution uses the successful n-gram-based solution presented in Chapter 5, but with two fundamental improvements. First, variable-length n-grams are used to prevent the mechanism from trying to preserve correlations between consecutive events where no such correlation is likely to exist, whilst also utilising the privacy budget as best as possible. Second, the solution incorporates popularity information in a generic way, covering a comprehensive spectrum of cases where such information is public knowledge, collated from multiple sources, privately learned from data, or unavailable entirely. The novel challenge of integrating data from the donating service with the existing data of the receiving service is addressed with three efficient solutions to a bipartite matching-based optimisation problem that links semantically similar events. This integration step exploits the proposed practical privacy setting in which the receiving service can utilise previously collected, unperturbed data to enhance utility, without affecting privacy guarantees.

Throughout Chapters 3–6, the aforementioned algorithmic contributions are supported by extensive experiments that typically use real-world data. These experiments consistently show that these contributions outperform baseline methods (where such methods exist) in terms of utility, efficiency, and privacy. In keeping with the practical focus of this work, the experiments focus on real-world analytics tasks, such as range, hotspot, and facility location queries. Another consistent finding of these experiments is that incorporating publicly available external knowledge has a clear benefit for utility. This finding, and the overall strong performance in these tasks, together demonstrate that the proposed solutions possess the five desirable properties outlined in Section 1.2.

1.4 Thesis Structure

The remainder of this thesis is structured as follows.

Chapter 2 introduces differential privacy, its properties, the mechanisms used to achieve it, and its local variant, LDP. Along with a thorough summary of recent literature, this chapter also explores the question of what knowledge can be assumed to be public, and how it can be used to boost the utility of private data sharing mechanisms when operating in the real world.

The main technical methods and contributions of this thesis are presented in Chapters 3–6. Chapter 3 details the two methods for generating differentially private spatial data, and Chapter 4 proposes label-LDP and GeoPointGAN. Chapter 5 presents the n-gram-based solution for publishing location sequences with LDP. Chapter 6 extends this work and addresses the problem of sharing trajectories between multiple services, also using LDP.

The final chapter of this thesis, Chapter 7, includes a summary of the main technical contributions, a discussion on the limitations of this work, and proposals for future work.

Chapter 2

Background

This chapter provides the background material to support the technical contributions presented in the remainder of the thesis. Section 2.1 formally introduces both DP and LDP, as well as their associated properties and mechanisms. Section 2.2, provides an extensive summary of recent literature that is related to the work of this thesis. This review is complemented, where necessary, by problem-specific literature summaries in subsequent chapters. Finally, in Section 2.3, we discuss the notion of public knowledge in the context of this thesis. Specifically, we outline what publicly available external knowledge can be used to enhance utility, how it can be used, and how previous work has utilised external knowledge in various capacities.

2.1 Differential Privacy

To illustrate the ideas of this section, consider a toy example in which four people are sharing their eye colours. Alice and Bob both have green eyes, Chiara has blue eyes, and David has brown eyes. Although eye colour is not inherently sensitive (as it can be observed publicly), it acts as a proxy for more sensitive information (e.g., blood type, income, location).

2.1.1 Definition and Properties

The core idea behind differential privacy is that the output of any function (i.e., query) on a dataset should be approximately the same regardless of whether any one record in the dataset is included or not. It can be formally stated as follows:

Definition 1 (ϵ-differential privacy [83]). *A randomised mechanism $\mathcal{M}$ satisfies ϵ-differential privacy if, for any two neighbouring datasets $\mathcal{D}$ and $\mathcal{D}'$, and for all outputs $y \in \mathcal{Y}$ where $\mathcal{Y} = \mathrm{Range}(\mathcal{M})$, we have:*

$$\frac{\Pr[\mathcal{M}(\mathcal{D}) = y]}{\Pr[\mathcal{M}(\mathcal{D}') = y]} \leq e^{\epsilon} \tag{2.1}$$

Two datasets $\mathcal{D}$ and $\mathcal{D}'$ are said to be neighbouring if they differ only by the inclusion of one tuple t (i.e., $\mathcal{D}' = \mathcal{D} \pm t$). A relaxed variant of this notion is one in which $\mathcal{D}$ and $\mathcal{D}'$ are neighbouring if they differ owing to the substitution of one element. That is, they have the same $N-1$ elements, but $\mathcal{D}$ has t whereas $\mathcal{D}'$ has t' as their N^{th} element respectively.

The parameter ϵ denotes the privacy budget, which influences the ubiquitous trade-off between utility and privacy. Lower ϵ values correspond to stronger privacy but lower utility due to an exponentially tighter probabilistic bound, whereas higher ϵ values provide higher utility at the expense of privacy.

To illustrate how DP works, consider that Alice, Bob, and Chiara are in a room and someone asks a trusted aggregator how many people in the room have brown eyes. The answer is reported truthfully to be 0. If David enters the room and the same question is asked, the answer would be 1, which would leak information about David's eye colour (i.e., their eyes are brown). DP protects against these membership attacks through a range of mechanisms (as we shall see), such that each member has a degree of plausible deniability regarding their membership in the dataset. This prevents an adversary from definitively identifying any true sensitive information about the people in the room.

Properties

DP has three key properties – sequential composition, parallel composition, and post-processing – that can be utilised to create larger differentially private mechanisms. Proofs for all three properties can be found in Dwork and Roth [82].

Sequential composition states that, if multiple differentially private mechanisms are applied to the same dataset, the overall privacy leakage is the sum of the individual privacy leakages. Formally, for a set of mechanisms $\mathcal{M}_1, \ldots, \mathcal{M}_i, \ldots, \mathcal{M}_n$, each of which satisfies ϵ_i-DP, any function of them $\mathrm{f}(\mathcal{M}_1, \ldots, \mathcal{M}_i, \ldots, \mathcal{M}_n)$ is ϵ-differentially private, where $\epsilon = \sum_i^n \epsilon_i$. This result also implies that, if the same ϵ-DP mechanism is applied k times, the overall privacy leakage is $k\epsilon$.

Parallel composition states that, if differentially private mechanisms are applied to *disjoint* subsets of $\mathcal{D}$, the overall privacy leakage is bounded by the maximum privacy leakage of the mechanisms $\mathcal{M}_i$. That is, the function f would satisfy $(\max_i \epsilon_i)$-differential privacy instead.

The post-processing property states that, for any two mechanisms $\mathcal{M}_1$ and $\mathcal{M}_2$, if $\mathcal{M}_1$ satisfies ϵ-DP, then the composition $\mathcal{M}_2(\mathcal{M}_1(\cdot))$ satisfies ϵ-DP, regardless of whether $\mathcal{M}_2$ satisfies ϵ-DP itself. In a practical sense, the post-processing property allows the output of any DP mechanism to be used and manipulated infinitely, without affecting its privacy guarantee, as long as there is no further interaction with the sensitive data. Examples of post-processing operations include rounding non-integer values, rounding negative values to zero, and summing multiple private histogram counts to obtain aggregate histogram counts.

Privacy Budget Allocation

Choosing the value of the privacy budget, and how to divide it between multiple steps (if using sequential composition), is ultimately a decision about whether to conduct a privacy-led or utility-led analysis. A privacy-led approach will first fix a value for ϵ, corresponding to the desired (possibly mandated) level of privacy. A mechanism is then developed within this constraint in order to maximise utility, which can include investigating how to allocate the privacy budget across multiple steps. This is allowed as the overall privacy leakage of the mechanism is still bounded by ϵ, as stated by DP's composition properties. Conversely, a utility-led approach will aim to achieve a minimum level of utility, sacrificing the level of privacy in order to do so. This can include using a larger value for the privacy budget, or using variants of DP or LDP that have less stringent privacy requirements.

It is intuitive to seek to balance these competing approaches, and it is not uncommon to switch between a privacy-led and a utility-led approach throughout development. Indeed, in this thesis, we begin with a clear privacy requirement: all solutions should use strict DP and LDP as the base privacy model. Given this privacy constraint, our focus shifts to achieving a high level of utility, most notably through the inclusion of external knowledge. Achieving such utility is ultimately dependent on the overall value for the privacy budget, as well as its division across stages of a mechanism. Throughout this thesis, each solution is evaluated for a range of privacy budgets and, where appropriate, we also consider several ways of distributing the privacy budget. This helps to demonstrate a solution's robustness, and provides insight into the privacy-utility trade-off inherent in this work.

2.1.2 Local Differential Privacy

The traditional form of differential privacy relies on a trusted aggregator who collects the users' true data, before adding noise to a dataset to ensure plausible deniability. However, in many practical settings, this is unfeasible, unrealistic, or undesirable. Local differential privacy, proposed initially by Duchi et al. [81], allows users to perturb their data before sharing it with an aggregator, who can be trusted or not. Although LDP provides stronger privacy, and benefits from additional security compared to centralised DP, it typically involves the addition of more noise, which makes it harder to achieve high utility easily.

Definition 2 (ϵ-local differential privacy)**.** *A randomised mechanism $\mathcal{M}$ satisfies ϵ-local differential privacy if, for any two inputs $x, x' \in \mathcal{X}$ and output $y \in \mathcal{Y}$:*

$$\frac{\Pr[\mathcal{M}(x) = y]}{\Pr[\mathcal{M}(x') = y]} \leq e^{\epsilon} \tag{2.2}$$

The intuition with LDP is that, given the output y, an adversary cannot (with high confidence) identify the input value. Importantly, LDP possesses the same composition and post-processing properties as centralised DP [63].

To achieve LDP, users can perturb their data in several ways. In the case of non-numeric data, users randomly select the value they report from a finite output domain. Numeric data can also be perturbed this way, or additive noise can be used to change the value that is reported.

2.1.3 Mechanisms

There are two fundamental methods for achieving DP in the centralised setting: the Laplace mechanism and the exponential mechanism. Both mechanisms are used widely in DP research, and throughout this thesis. For their proofs of DP, readers can consult the original references. There are many core mechanisms for providing LDP to data, depending on the query or application (e.g., frequency oracles are well-outlined by Wang et al. [240]), and this thesis uses the widely used randomised response mechanism.

Laplace Mechanism

The Laplace mechanism is used to release the values of numeric functions of data [83]. For a function f acting on $\mathcal{D}$, it adds random noise to the value of $\mathrm{f}(\mathcal{D})$ such that:

$$\mathcal{M}_{\mathrm{f}} = \mathrm{f}(\mathcal{D}) + \mathrm{Lap}\left(\frac{\Delta_{\mathrm{f}}}{\epsilon}\right) \tag{2.3}$$

where $\mathrm{Lap}(\cdot)$ denotes the Laplace distribution, and the scale of the noise is set by the global sensitivity of f, which is defined as: $\Delta_{\mathrm{f}} = \max_{\mathcal{D},\mathcal{D}'} \ |\mathrm{f}(\mathcal{D}) - \mathrm{f}(\mathcal{D}')|$.

Queries for which the Laplace mechanism is used include sum, count, and average queries. In our example, noise from the Laplace mechanism can be added to our count query to provide plausible deniability ($\Delta_{\mathrm{f}} = 1$ for count queries [82]). For example, the number of people with green eyes might be reported as $2 + \mathrm{Lap}(\frac{1}{\epsilon}) = 2 - 0.124 = 1.876$. As an aside, the private answer is likely to be a decimal, which is nonsensical as there cannot be 1.876 people with blue eyes. Hence, post-processing can be invoked to round the private answers to the nearest integer, without affecting the privacy of any member of the dataset.

Exponential Mechanism

Whereas the Laplace mechanism is only suitable for numerical data, the exponential mechanism [168] can be used to return differentially private responses to non-numeric queries. For any dataset $\mathcal{D}$ and output $y \in \mathcal{Y}$, the result of mechanism $\mathcal{M}$ is ϵ-differentially private if one randomly selects y such that:

$$\Pr[\mathcal{M}(\mathcal{D}) = y] = \frac{\exp\left(\epsilon q(\mathcal{D}, y)/2\Delta_q\right)}{\sum_{y_i \in \mathcal{Y}} \exp\left(\epsilon q(\mathcal{D}, y_i)/2\Delta_q\right)} \tag{2.4}$$

where $q(\mathcal{D}, y)$ is a quality function, and Δ_q is the global sensitivity of the quality function (defined as for Δ_f).

The quality function is typically defined to be a positive function in which high values correspond to desirable outputs and, in turn, high probabilities. However, throughout this thesis, we define the quality function to be a semantic distance function such that: $q(\mathcal{D}, y) = -d(\mathcal{D}, y)$, where $d(\cdot,\cdot)$ is some distance function. This means that datasets, or values, that have a high semantic similarity will have low distance values, but high quality function values, and high probabilities in the exponential mechanism. The opposite is true for entities with low semantic similarity. To ensure that $\Delta_q = 1$, the distance function is normalised such that all values are within the range $(0, 1)$. More information on the exact uses and definitions for the semantic distance (quality) functions is given in Section 2.3 and Chapters 5 and 6, where they are used.

The utility of the exponential mechanism can be written as:

$$\Pr\left[q(x, y) \leq OPT_q - \frac{2\Delta_q}{\epsilon}\left(\ln\frac{|\mathcal{Y}|}{|\mathcal{Y}_{OPT}|} + \xi\right)\right] \leq e^{-\xi} \tag{2.5}$$

where OPT_q is the maximum value of $q(x, y)$, $\mathcal{Y}_{OPT} \subseteq \mathcal{Y}$ is the set of outputs where $q(x, y) = OPT_q$ [82], and ξ is some constant.

Finally, although more commonly used for centralised DP, the exponential mechanism can be applied in LDP, where different inputs (for LDP) are considered to be equivalent to neighbouring datasets of size 1 (for centralised DP).

Randomised Response

Randomised response was initially proposed by Warner [245] to give survey respondents plausible deniability when answering sensitive questions. In the original setting, users are asked a yes-no question and, before answering, each user secretly tosses a biased coin. If it comes up as 'heads' the user responds truthfully; otherwise, they say 'yes', regardless of whether this is true. This prevents an adversary from definitively knowing the true response of any one respondent. Despite these noisy responses from each individual, the (approximate) true number of yes-no responses can be found by debiasing the noisy answers, thus providing a reasonable level of utility at the aggregate level.

Randomised response can provide LDP if the probabilities of returning any one response are carefully tuned according to the domain size and privacy budget. Although the original randomised response setting had two outcomes, it can be generalised to any number of outcomes by setting the probability that a user reports true information to be $\alpha = \frac{e^\epsilon}{e^\epsilon + |\mathcal{Y}| - 1}$, where $|\mathcal{Y}|$ is the size of the output set [86]. Hence, the probability of reporting any other single output is $\beta = \frac{1}{e^\epsilon + |\mathcal{Y}| - 1}$. To prove that setting α and β in this way satisfies LDP, one can simply consider the ratio of the two entities:

$$\frac{\alpha}{\beta} = \frac{e^\epsilon}{e^\epsilon + |\mathcal{Y}| - 1} \times \frac{e^\epsilon + |\mathcal{Y}| - 1}{1} = e^\epsilon$$

In our example, consider that there are four possible eye colours: brown, blue, green, and grey. If Alice wanted to share their eye colour with someone, they would report their true eye colour with probability $\alpha = 0.475$ (if $\epsilon = 1$), and any other colour with probability $\beta = 0.175$.

2.2 Private Data Sharing

To contextualise this thesis' work, it is necessary to review the relevant literature. Owing to the large body of existing work on private data sharing, this review generally focuses on papers that use DP, LDP, or one of their many variants. Even within the field of DP and LDP, the body of existing work is too large to survey completely here. As a result, this review primarily focuses on work that uses DP and LDP with spatio-temporal data, as this is the main focus of this thesis. For more general summaries, readers can consult recent surveys on DP [76, 176] and LDP [73, 261]. Similarly, for a wider review of location privacy, readers can consult Jiang et al. [129], whereas Errounda and Liu [88] provide a wider survey on DP research with spatio-temporal data.

2.2.1 Overview

The first private data sharing methods focused on publishing datasets. We have already noted how early data publication methods relied heavily on anonymisation and perturbation, which made them vulnerable to attack and deanonymisation. Early DP works on data publication focused on releasing answers to popular queries, such as search [143], predicate [232], and k-mean queries [91]. In particular, the release of private histograms attracted a lot of early attention. Hay et al. [117] utilise the idea of consistency across overlapping queries to release accurate private histograms, and Xiao et al. [249] propose multi-dimensional partitioning strategies for the same task. Xu et al. [256] advance these works by proposing NoiseFirst and StructureFirst, which differ by the order in which DP noise is added and the histogram is structured. Since these seminal works, histograms have continued to attract the focus of DP researchers [e.g., 61, 116, 146, 192], and are utilised in one of the data generation approaches in Chapter 3.

However, as these methods are generally restricted to a pre-defined and limited range of queries, more recent work has focused on the private release of entire high-dimensional datasets and marginals. PrivBayes [274] uses Bayesian networks to capture the correlations between attributes before adding noise to the marginals and releasing representative data. PrivView [193] is a two-step process in which a strategically chosen set of marginals are composed (called 'views'), before maximum entropy optimisation is used to reconstruct k-way marginals from these views. Chen et al. [54] instead use a sampling-based framework to construct a dependency graph, which is then used to generate noisy marginals and synthetic data. DPPro [255] meanwhile uses random projection to model high-dimensional data in a lower dimensional space, where noise can be added and synthetic data created. Though tangential to the work presented in Chapters 5 and 6, these works highlight the inherent difficulty in privately publishing high-dimensional data with high utility, which is a challenge addressed in these chapters.

In contrast to data publication, data synthesis gives more flexibility to end users. Beyond private data sharing, synthetic datasets are important for many data science tasks, including benchmark generation [102, 107], and database management system testing [28, 221]. When generating synthetic data, one must try to achieve high levels of privacy, fidelity, and downstream utility. Compared to data publication, data synthesis results in a weaker relationship between the sensitive input data and the (supposedly) private output data. Nevertheless, strong privacy notions need to be implemented in order to ensure that inference-based attacks are not successful [e.g., 78, 97]. The plausible deniability-based protections of DP and LDP help to protect against these threats, which is why DP is increasingly used for data synthesis [e.g., 1, 99, 125, 219, 220, 252].

At a basic level, any synthetic dataset should have high statistical similarity with the original dataset (i.e., high fidelity), as this ensures that the synthetic data can be used by a wide range of end users. However, in realistic settings, those who create synthetic data often do so because they have particular end tasks in mind [14, 167]. Hence, they want to maximise the synthetic data's utility for these specific analytical tasks. Indeed, in outlining their solution for the NIST competition, McKenna et al. [167] advocate this utility-led approach. Specifically, the authors prioritise preserving fidelity in some low-dimensional marginals (selected based on their importance for the end task), before using these to generate higher dimensional data. This approach is particularly pertinent when dealing with high-dimensional data owing to the curse of dimensionality, as McKenna et al. [167] note themselves. Given the real-world focus of this thesis, more attention is paid to achieving high utility in location analytics tasks, all within the constraints of DP. Notwithstanding this focus, the solutions in Chapters 3 and 4 are also developed with fidelity in mind. Consequently, evaluation in both chapters focuses on statistical fidelity and utility in real-world inspired queries.

Machanavajjhala et al. [159] were the first to generate synthetic data while offering privacy guarantees under a revised probabilistic version of DP. Since then, several generic approaches to generating privacy-preserving synthetic data have been proposed, including ones using deep learning [1], Bayesian networks [274], Gaussian mixture models [47], and noise-induced marginals [167]. GAN-based solutions for private data synthesis are also widespread, and these are surveyed in more depth in Chapter 4. Some other methods for specific data types, such as social network graphs [177], have been proposed but these methods are not applicable to spatial data. This leaves spatial data synthesis as a core and outstanding problem of study.

2.2.2 Private Location Data in the Centralised Setting

Sharing raw location data in the centralised domain can be divided into two main methods: obfuscation and perturbation. Obfuscation-based methods aim to lower the granularity of sensitive information in a systematic, controlled, and statistically rigorous way [20, 88]. To do so, Wang et al. [235, 236, 237] use an obfuscation matrix to minimise the expectation of data uncertainty between the true and obfuscated locations, before applying their work to the task of mobile crowdsensing. Yang et al. [262] extend this idea to consider the temporal dimension, again with the application of mobile crowdsensing. Obfuscation also provides a degree of indistinguishability to users and their data. This notion has been extended to the spatial domain by Andrés et al. [12], who propose geoindistinguishability, in which two locations are indistinguishable if the probability ratio is within some exponential bound that is based on the distance between the points. This exponential bound is similar to the one provided through DP, although geoindistinguishability's privacy level is weaker than strict DP. Geoindistinguishability has since been extended by Ren et al. [198] who propose DistPreserv in which geoindistinguishability-like privacy is provided to users, whilst also maintaining the location distribution of the users. This provides better utility to the service collecting the data while still providing practical levels of privacy to the users.

Whereas obfuscation methods rely on reporting data at a coarser granularity, perturbation methods actually change the values of the points, but generally retain the same spatial granularity. Indeed, in Andrés et al. [12], geoindistinguishability is provided to locations through a perturbation mechanism that uses a Laplace-like distance-based mechanism to ensure that closer points are more likely to be returned than points further away. However, this method involves adding a large amount of noise to points, which has downstream effects on utility. More refined mechanisms for providing geoindistinguishability have been proposed [7, 36, 49]. Bordenabe et al. [36] introduce the optimal mechanism, which is effective but experiences large runtimes owing to its linear programming-based solution. Chatzikokolakis et al. [49] instead propose a Bayesian remapping procedure to capture the utility benefits of the optimal mechanism while improving efficiency, while Ahuja et al. [7] use spatial hierarchies and the composition property to efficiently prune space to improve utility further. Despite these advances, all geoindistinguishability-based perturbation approaches fail to provide the level of privacy that is required for DP, which limits their applicability to this work.

Aggregating and structuring spatial data are also integral techniques for answering location analytics queries effectively. In particular, private spatial decompositions are important for answering spatial count, range, and histogram queries. While many of the aforementioned private histogram methods can be applied to this task with minor modification, there have also been works that specifically focus on decomposing spatial data. For example, Cormode et al. [62] propose a class of private spatial decompositions, including quad-trees and k-d trees. To et al. [218] propose a two-level h-tree to minimise the division of the privacy budget, and Kim et al. [138] tackle the challenge of decomposition in skewed distributions, which are common in spatial datasets. Recently, Shaham et al. [202] build on these works and propose the homogeneous tree framework, which seeks to achieve homogeneous data density within each resulting partition to minimise the effect of the added DP noise. Whereas these methods are data dependent, there are also decomposition methods that are data independent, such as the uniform and adaptive grids proposed by Qardaji et al. [191]. PrivTree [273] is also data-independent and uses a hierarchical approach without pre-defining the tree height. These spatial decomposition methods have served as the basis for spatial query-specific papers. For example, Ghane et al. [101] release private spatial histograms whilst ensuring consistency in the synthetic output, Shaham et al. [203] use differentially private partitioning to release origin-destination matrices, and Zeighami et al. [271] use neural networks to answer spatial range queries. We utilise some of these spatial decomposition techniques [i.e., 191] in Chapter 3 for our own practical task: differentially private partitioning. Other techniques can also be used as alternative partitioning methods, or to augment the data generation methods [e.g., 101].

2.2.3 Private Trajectories in the Centralised Setting

In contrast to single locations, the release of private trajectories (or, more generally, sequences) is an important and popular research area. Chen et al. [53] were the first to publish synthetic sequential datasets by using a prefix tree that stores the private noisy counts of each sequence. Chen et al. [52] extend this work by using an n-gram model to store variable-length sequences in the tree. While neither of these works were explicitly designed for trajectory data, the authors note that they could be extended relatively naturally. We also note that the use of n-grams is similar to the work in Chapters 5 and 6. However, the settings are incomparable: Chen et al. [52] release DP counts at the n-gram level, whereas we use n-grams as the basis for releasing LDP trajectories at the user level. Indeed, many early works used trajectory data to answer common queries [e.g., 34, 52, 148] that focus on returning summary level statistics, as opposed to individual-level data that gives end users more flexibility.

Consequently, several methods for synthesising trajectories in the centralised setting have been proposed. DPT [119] uses a Markov chain-like model to generate trajectories, and the authors also use hierarchical decomposition to structure their space, as we do in Chapters 5 and 6. DPStar [111] also uses a Markov model, alongside density-aware spatial partitioning, to obtain better performance than DPT. DPStar has since been surpassed by AdaTrace [110] and OptaTrace [112], both of which primarily focus on location traces, which have more defined spatio-temporal correlations. Recently, Yao et al. [264] use a graph-based method to publish trajectories, and this is also the first paper to explicitly consider outliers in trajectory data. Lestyán et al. [145] meanwhile use graph neural networks to develop a synthetic trajectory generator that can preserve fine-grained characteristics of the input data. Other works have combined techniques from k-anonymity research with DP to produce DP-compliant trajectories [e.g., 150].

Several variants and relaxations of DP have been applied to the problem of privately sharing trajectories. These include w-event privacy [136], l-trajectory privacy [42], and spatio-temporal event privacy [44, 45]. While many of these variants achieve good utility, their weaker privacy notions limit their relevance to this work, and any comparison between them lacks meaning.

A somewhat orthogonal, yet complementary research area is work that focuses on temporal correlations under DP, which is of particular importance when considering trajectories. Xiao and Xiong [248] use the notion of δ-location sets to protect these correlations. A δ-location set is the set of feasible locations that any one person can be located, given a previous point and current timestep. This idea is similar to the hard constraints we use when generating data in Chapter 3, and the reachability constraints we use in Chapters 5 and 6. The idea of δ-location sets has since been studied further by the same research group [44, 45, 250]. In related work, Cao et al. [43] study the 'reverse' problem when quantifying DP under temporal correlations. Although this set of work does not focus on publishing private trajectories per se, it is important to consider it within the wider context of private trajectory release.

Most existing private trajectory release methods focus solely on geographic utility, which can broadly be viewed as location preservation, as measured through count and range queries. However, in any practical setting, it is necessary for private trajectory data to achieve high semantic utility, which includes preserving spatio-temporal hotspots and correlations (e.g., rush hour tendencies), as well as achieving strong performance in location analytics tasks (e.g., facility location). Although some trajectory release methods do aim to maximise semantic utility [e.g., 27, 212, 216], none of them satisfy any form of DP, although Bindschaedler and Shokri [27] satisfy a notion of plausible deniability, which has similarities with DP. This leaves the recent work of Li et al. [147] as the state-of-the-art, not only because it outperforms AdaTrace and DPT, but because it is also designed for both types of utility, and satisfies DP. Throughout this thesis, we seek to achieve high geographic and semantic utility in order to develop real-world-focused solutions.

2.2.4 Spatio-Temporal Data in the Local Setting

Spatio-temporal data is increasingly studied in the local setting. Wang et al. [234] combine RAPPOR [86] and randomised response with modified Hilbert curves (to preserve spatial correlations) and the genetic algorithm (to effectively segment the space) to allow users to share location data for mobile crowdsensing. Meanwhile, Xiong et al. [254] combine δ-location sets with generalised randomised response, and a time-specific definition of (ϵ, δ)-

LDP to enable real-time spatio-temporal data aggregation. Zhao et al. [277] propose LDPPart, which uses a probabilistic, tree-based partitioning algorithm to publish single location records with LDP. Errounda and Liu [87, 89] use w-event privacy, combined with randomised repsonse and the Laplace mechanism, to release continuous [87] and collective [89] location statistics. Hong et al. [122] propose the square mechanism, which perturbs locations to nearer locations with higher probability, to collect geospatial data for frequency estimation-based tasks. Finally, Asada et al. [16] conduct an interesting study that uses machine learning (specifically, matrix factorisation) to provide users with LDP-compliant recommendations for when they should share their location data, and when they should not.

The expectation maximisation algorithm [75] is also used in many works, normally to maximise utility, and typically where randomised response has been used to privatise the original data. For example, Ren et al. [196] use expectation maximisation and lasso regression to publish high-dimensional data with LDP, whereas Kim et al. [139] use the technique to collect indoor positioning data with LDP. Ye et al. [265] note that accuracy decreases when a space is discretised and frequency oracles are applied, which is a common approach for sharing spatial data with LDP. To combat this, they propose a continuous perturbation mechanism that uses the expectation maximisation algorithm, which is shown to outperform existing methods for frequency estimation in the spatial domain.

Other data publication methods have been developed that, despite not exclusively focusing on spatio-temporal data, are nevertheless important for spatio-temporal queries. These include mechanisms for answering multi-dimensional range queries [e.g., 65, 80, 260]; publishing 2-way marginals [64], which are useful for preserving spatio-temporal correlations; and frequent itemset mining [e.g., 238, 241], which are useful for identifying spatio-temporal hotspots.

As in the centralised setting, there have also been relaxations of LDP specifically for use with spatial data in the local setting. For example, Chen et al. [55] introduce personalised LDP to allow users to specify a geographic 'safe region' when sharing their data for the purposes of spatial data aggregation. Personalised LDP is also used by Bao et al. [22], who combine it with the expectation maximisation algorithm for the purposes of POI recommendation in the local setting. Similarly, although geoindistinguishability was proposed in the centralised setting, it can easily be applied in the local setting by performing the location perturbation on the user's device.

These aforementioned works, while useful, cannot be easily translated to the problems we study in Chapters 4–6 as they fail to deal with trajectory data, and/or they adhere to a different (typically weaker) privacy definition. Despite the large body of literature in the centralised setting, there is very little work focusing on trajectories in the local setting. This is due to the added complexity of dealing with trajectory data, combined with the additional noise that is necessary when using local privacy settings. Xiong et al. [253] use randomised response to continually share location data, using (ϵ, δ)-LDP, and Gursoy et al. [113] propose a method for sharing data (including sequence data) using condensed LDP. Naghizade et al. [174] model trajectories as sequences of 'stop-and-go' movements, and privatise trajectories by perturbing the 'stops'. However, this work does not provide any meaningful DP-like privacy level, although it does provide a guarantee based on the number of stops in the trajectory. Given this limited body of literature, Chapter 5 is the first work that releases LDP-compliant trajectories. Moreover, in Chapter 6, where the focus shifts beyond spatio-temporal data, we present a more generalised mechanism for trajectory sharing with LDP. Chapter 6 also considers the problem of sharing and integrating the data of multiple services; this differs from existing work, which only considers services sharing data in isolation.

2.2.5 Distance-Based LDP

Many early LDP mechanisms, such as generalised randomised response [86], (optimised) unary encoding [240], and (optimised) local hashing [23], assume that 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. Hence, there have been several recent relaxations of (L)DP to allow perturbation probabilities to be non-uniform across the domain. In $d_{\mathcal{X}}$-privacy [48], and its location-specific variant geoindistinguishability [12], the indistinguishability level between any two inputs is a function of the distance between them. This concept has since been generalised to any metric, and extended to the local setting to give metric-LDP [11]. Context-aware LDP [4] goes further by allowing an arbitrary (non-metric) measure of similarity between points, and input-discriminative LDP [108, 109] assigns each data point its own privacy level. Other relaxations to LDP rely on the provision of some additional information. For example, personalised LDP [55] lets users specify a desired privacy level, whereas local information privacy [127, 128] utilises knowledge of users' priors.

Although these variants have been shown to be effective, they all share a common limitation: they provide a weaker level of privacy than traditional ϵ-LDP. Specifically, all of them have upper bounds for the probability ratio of the form $e^{\mathrm{f}(x,x')}$, where f is some function that quantifies the distance or similarity between two inputs x and x'. In Chapters 5 and 6, however, we use the exponential mechanism to create a distance-based LDP mechanism. When this is the case, $\mathrm{f}(x,x') = \epsilon$ for all x, x', which conforms to the requirements of LDP, and hence provides a stronger level of privacy.

2.3 What is Public Knowledge?

As outlined in Chapter 1, a primary aim of this thesis is to demonstrate how using publicly available external knowledge can enhance the utility of methods for private data synthesis and publication. This section discusses this idea in more detail, and explains why using such knowledge in this manner does not leak privacy.

As the external knowledge is already assumed to be public knowledge, it is non-sensitive and does not need to be privatised. Importantly, it is *only* used to enhance utility, whereas privacy is provided through the application of standard DP or LDP mechanisms (e.g., exponential mechanism), which can be done with no external knowledge. This means that external knowledge can be used independently of the sensitive data, which leaves any DP or LDP guarantee unaffected. This also means that an adversary (who is assumed to have access to any public, external knowledge) cannot use this data to learn meaningful information about the sensitive data with high probability.

2.3.1 Types of Public Knowledge

There are many different types of public knowledge, and a vast number of sources from which to obtain this knowledge. The best types and sources of knowledge will be problem- or domain-specific and, given this thesis' focus on spatio-temporal data, a primary source of external knowledge is geographic data. Geographic data sources (e.g., Google Maps, OpenStreetMap) outline the locations of seas, rivers, military compounds, and the structure of the road network, among others. These sources also contain large databases of POIs, including their opening hours, 'price points', category, etc., which are a valuable source of information that can be exploited. Data from other domains can also be utilised. For example, the schedules of sports teams, cinemas, theatres, etc. can be used to determine whether it is realistic for people to be associated with that POI at a certain point in space-time. Even unstructured, user-generated content, such as public social media comments and videos, can be harnessed. For example, semantic analysis and natural language processing techniques can be used to infer whether a POI is popular or offers good services.

Public knowledge does not need to be codified; it can also be informed by historic real-world observations. The popularity of POIs is an example of an attribute that can be quantified using historic data. For example, in many cities, suburban train stations will exhibit strong peaks in usership during the morning and evening weekday rush-hours, but will be relatively quiet during a weekday and at weekends. Hence, if these trends in usership can be modelled using publicly available data (e.g., bikesharing data), then popularity information can be incorporated into mechanisms easily. We first consider POI popularity to be public knowledge in Chapter 5. We then generalise the problem in Chapter 6 to consider settings where popularity is learned from data, crowdsourced from multiple sources, or not known at all.

Finally, public knowledge also extends to commonsense reasoning, which may be informed by pervasive socio-cultural knowledge. For example, if Alice were to be located at a church at 3am on a Tuesday, this is unlikely to be feasible as people tend to visit churches on Sunday mornings.

2.3.2 Utilising Public Knowledge

Public knowledge can be utilised in three main ways, which are explained at a high level here. Chapters 3–6 contain more detail on the exact methods that are specific to each problem. First, a series of hard constraints can be imposed to control what values, or outputs, are feasible. Second, the intrinsic structure or hierarchies of real-world entities (e.g., road network, POI categories) can be exploited to control data generation or influence data perturbation. Finally, the distance (quality) functions used with the exponential mechanism can be structured to ensure that outputs that are semantically similar to the input are more likely to be returned.

Hard Constraints

The first way in which public knowledge can be used is through hard constraints that impose boundaries on whether certain outputs, or combinations thereof, are feasible. In Chapter 3, we use geographic knowledge to restrict points from being generated within 'out-of-bounds' areas, such as seas and rivers. In Chapters 5 and 6, we use hard constraints to control the output domain when using the exponential mechanism. For example, POIs that are closed from 9pm–9am cannot be viable if the event occurs at 11pm. Similarly, we deem it to be unrealistic for someone to purchase a banana or a car for £100.

Hard constraints can also be applied to prevent infeasible relationships between pairs of points in trajectories. In Chapters 3–6, a reachability constraint prevents two consecutive points from being output if the physical distance between them is greater than the maximum distance one could travel in the time between points. That is, if Bob is located at Big Ben at 9am, it would be feasible for a mechanism to 'locate' them at the London Eye at 10am, but not the Eiffel Tower. In Chapter 6, we introduce additional non-spatio-temporal hard constraints between trajectory points, including an ordering constraint that dictates permissible orders in which events can occur. For example, one cannot return a product to a shop before buying it.

Intrinsic Structures

Many real-world entities have intrinsic structures or hierarchies. For example, the road network has a clearly defined structure as a set of roads and intersections. Geographic space also has an intrinsic hierarchy: countries can be subdivided recursively into regions, counties, cities, suburbs, postcode areas, streets, and houses. Another domain that has strong hierarchical structure is the category classification for POIs. For example, Pizza Express can have the tags 'pizza restaurant', 'Italian restaurant', and 'food and drink'. These hierarchies can be defined through formal means, such as the US Census Bureau's Geographic Entity Hierarchy [227], or Foursquare's POI category hierarchy [93]. Alternatively, they can be defined arbitrarily by discretising the data space using a range of granularities. For example, space can be divided into 100 m $\times$ 100 m cells, and time can be divided into 15- or 60-minute intervals. In Chapter 3, we use the structure of the road network to control the data generation process, and in Chapters 5 and 6, we use both types of hierarchy to structure our space and inform the semantic distance functions.

Semantic Distance Functions

The final way to incorporate public knowledge into our algorithms is when using the exponential mechanism for perturbation. Specifically, the quality function used can be tuned such that the probability of returning outputs that are semantically similar to the input data is increased. For example, suppose that Jane is at a pizza restaurant at 9pm in the city centre, and they wish to perturb their location. Intuitively, higher utility should be obtained if the mechanism makes it more likely that they are perturbed to a pasta restaurant than a burger restaurant, which should be more likely than a shoe shop. To maximise utility, these semantic distance functions can operate across multiple dimensions or attributes. For example, time, space, and category can be utilised together: the tuple [Pasta Restaurant, 9pm, City Centre] should be more likely than the tuples [Burger Restaurant, 9pm, City Centre]; [Pasta Restaurant, 6pm, City Centre]; and [Pasta Restaurant, 9pm, Suburb]. These semantic distance functions are fundamental to the algorithms presented in Chapters 5 and 6.

2.3.3 Previous Context-Aware Work

The general notion of incorporating external or prior knowledge into a privacy definition or mechanism to enhance utility can be described as 'context-awareness'. Whereas this thesis is the first to actively utilise a wide range of domain-specific external knowledge, there are some recent works that provide the functionality to do so.

Pufferfish privacy [137], which is a relaxation of DP, is designed for domain experts to create their own domain-specific privacy notions. Specifically, domain users specify a set of secrets (i.e., the sensitive data) and data evolution scenarios that are typical for that domain (i.e., processes that model how the sensitive data is likely to have been generated), both of which can utilise real-world knowledge. Blowfish privacy [118] builds on this work and offers a more generalised setting for incorporating external knowledge. The notion of mutual information (i.e., information held by users and adversaries) is explored by Asoodeh et al. [17, 18] and Wang et al. [244]. Similarly, both centralised information privacy [41] and local information privacy [127] consider the user-specific context by utilising user priors. This notion is extended by Jiang et al. [128] who investigate the context-awareness of local information privacy further. In the spatial domain, Acharya et al. [4] propose context-aware LDP, as well as block-structured LDP, which uses geographic hierarchical knowledge to enhance utility. Similarly, personalised LDP [55] utilises a geographic 'safe region' (specified by users) to control the level of privacy provided. Finally, Naghizade et al. [174] use contextual information from the road network to enhance the perturbation of trajectories.

It is reasonable to expect that these privacy notions would all give good levels of utility if used as the basis for solutions to the problems in this thesis. However, all of these works, and any other existing context-aware works, suffer from the same limitation: they do not conform to ϵ-(L)DP. For example, both pufferfish and blowfish privacy focus on the looser notion of indistinguishability, whereas Jiang et al. [127, 128] show that local information privacy achieves at least 2ϵ-LDP, but not ϵ-LDP. As Chapter 1 establishes that satisfying ϵ-(L)DP is a pre-requisite for any solution, these alternative notions become unviable. Furthermore, most of these works are theoretical in nature and, although they hint at incorporating prior real-world knowledge, they fail to explicitly outline how this would be done, which limits their overall applicability. Although it is beyond the scope of this thesis, developing extensions of these weaker privacy notions to include real-world knowledge would be a valuable avenue for future research. This would then enable an important analysis of the relative levels of privacy and utility that are achieved when using these variants.

Chapter 3

Generating Synthetic Location Data with Differential Privacy

This chapter focuses on generating differentially private synthetic spatial point datasets – a core issue of private spatial data publication that has many important applications, such as advertising and the better provision of public services. This problem is motivated further because spatial datasets are also used in several societally important fields, such as ecology [230], geology [279], and epidemiology [98], many of which contend with the need to preserve the privacy of the data subjects (e.g., animals from being poached illegally, individuals being identified from contact tracing apps).

As discussed previously, many existing DP methods for spatial data publication harm practical utility unnecessarily by failing to publish point (i.e., co-ordinate) data, and by being oblivious to real-world conditions (e.g., by ignoring coastline data). This chapter proposes two methods for generating differentially private spatial point data, both of which use external knowledge (to varying degrees) to enhance practical utility and maintain statistical fidelity. The first approach considers a richer set of ways to model the input location data. A differentially private partitioning-based framework restricts data generation to be within small private regions, before data is generated through three methods of increasing complexity. This includes a novel adaptation of kernel density estimation that is specifically suited to multiple point generation and maintains privacy. In the second approach, we utilise the road network structure to enhance utility, based on the observation that spatial data is heavily influenced by the underlying road network, which is public knowledge. Data is generated by sampling from edge-level micro-histograms, a process designed to ensure the underlying distribution of the real data is maintained as much as possible.

The chapter is organised as follows. We introduce the problem in Section 3.1. Section 3.2 outlines the partitioning-based approach, and Section 3.3 details the road network-based approach for synthetic data generation. Both approaches are evaluated using statistical and location analytics queries in Section 3.4. The chapter concludes in Section 3.5 with a discussion of the relative merits of each approach, alongside recommendations for their real-world deployment.

3.1 Problem Setting

Given a dataset containing the real locations of individuals, this chapter aims to develop methods for generating synthetic spatial point data that satisfies ϵ-DP, and preserves as much of the underlying distribution of the real data (i.e., statistical fidelity) as possible. Specifically, the objective is to *protect the existence and location of each individual in the dataset* by using DP. We use p and s to denote real and synthetic locations (in co-ordinate form), and $\mathcal{P}$ and $\mathcal{S}$ to denote the sets of real and synthetic locations, respectively.

This work utilises the sequential composition of DP (see Section 2.1). Specifically, we add Laplace noise in at most three places and divide our privacy budget across these steps, where each step has a privacy budget of ϵ_i. That is, $\epsilon = \epsilon_1 + \epsilon_2 + \epsilon_3$. This means that the maximum privacy leakage of any single point is upper bounded by ϵ.

Threat Model

We assume that the aim of an adversary is to *identify the true location of a certain individual.* As the proposed methods make use of external knowledge (e.g., the road network), which is *public knowledge*, we assume it can also be utilised by any adversary. Given this aim, there are two primary adversary targets: membership inference and location identification. DP provides protection in both regards. Through its definition (see Definition 1), each individual has a degree of plausible deniability with respect to their inclusion in the synthetic dataset (governed by a probabilistic bound; see Equation 2.1). This assures us that the output $\mathcal{S}$ does not provide the adversary with an advantage in determining the true location of a certain individual in the input. By synthesising location data (as opposed to merely publishing it), we further weaken the relationship between real and synthetic points. This is because one cannot (with a high degree of certainty) link a point in the synthetic dataset with one in the real dataset and conclude that they are both associated with the same individual.

As each point is treated independently, each point has its own (composable) DP guarantee. As such, the proposed methods can be applied to trajectory data without adverse downstream consequences. That is, it would not be possible to link individual points in the synthetic data and re-identify a real trajectory.

3.2 Partitioning-Based Data Generation

This section details our two-stage partitioning-based approach. We first restrict data generation to be within small regions, and then generate a noisy number of points within these regions. We propose a private version of kernel density estimation (KDE) to obtain representative probability distributions of point data. A well-defined kernel function requires access to points in the dataset, which makes it difficult to satisfy DP requirements *and* maintain high utility. Privatising KDE is further complicated by the need to repeatedly sample from the private KDE to generate multiple synthetic points, a process that could lead to high levels of privacy leakage ordinarily. Hence, we develop a kernel density estimate that satisfies ϵ-DP, achieves high utility, and is robust to multiple sampling.

3.2.1 Private Data Partitioning

We partition our space using differentially private grid- and clustering-based approaches from the literature. Importantly, any private partitioning method, such as those cited in Section 2.2.2, can be used to perform this first step. However, some of these methods (i.e., recursive tree-based partitioning) have been shown to be less effective than the methods included in this thesis [191, 217], albeit in other tasks.

Grid-Based Partitioning

A simple method to privately partition data is to use a uniform grid (UGrid) that is independent of the data, thus maintaining privacy. Choosing the correct granularity is important, as too coarse or too fine a grid can lead to poor results. Consequently, to determine the dimensions of the grid, we utilise a guideline proposed by Qardaji et al. [191]. For an $M \times M$ uniform grid, the number of cells in each direction is set to be:

$$M = \left\lceil \sqrt{\frac{N\epsilon_1}{10}} \right\rceil \tag{3.1}$$

where N is the number of points in the real dataset, and ϵ_1 is the privacy budget assigned to this task. This ensures that the average number of points per cell is suitably larger than the noise magnitude. Consequently, the total number of cells, or regions, into which the data is partitioned is $K = M^2 \approx \frac{N\epsilon_1}{10}$. Adding noise to the number of points n_i in each region R_i using the Laplace mechanism gives: $n'_i = n_i + \text{Lap}(\frac{1}{\epsilon_1})$.

In many realistic situations (e.g., non-uniform distribution of points), a uniform grid is unsuitable as it is likely to fail to capture the distribution accurately and/or add noise to the dataset in a biased manner. Therefore, adaptive grids (AGrid) [191], where denser regions have more grid cells and sparser regions have fewer cells, are advantageous. We follow the recommendation of Qardaji et al. [191] by first dividing the data region into an $M_1 \times M_1$ uniform grid where:

$$M_1 = \max\left(10, \frac{1}{4}\left\lceil \sqrt{\frac{N\epsilon_1}{10}} \right\rceil\right) \tag{3.2}$$

We add Laplace noise, controlled by ϵ_1, to the count in each cell and then divide each cell i into an $M_2^i \times M_2^i$ grid where:

$$M_2^i = \left\lceil \sqrt{\frac{n'_i\epsilon_2}{5}} \right\rceil \tag{3.3}$$

Partitioning concludes by adding Laplace noise, controlled by ϵ_2, to the count in each of the new smaller regions.

Theorem 1. *Both UGrid and AGrid partitioning have a time complexity of $O(N)$.*

Proof. Both methods have two phases: data partitioning and noise addition. For UGrid, each data point is assigned to a grid cell. For AGrid, each data point is assigned twice: once to a cell in the $M_1 \times M_1$ grid, and once to a cell in the $M_2 \times M_2$ grid. Hence, the complexity of this stage is $O(N)$, for both methods. After each grid assignment, each cell has noise added to it, a process with time complexity $O(K)$. This yields an overall time complexity for both methods of $O(N+K)$. However, as K is itself of the order of N, the process can be reduced to a linear operation with time complexity $O(N)$. ■

Cluster-Based Partitioning

We also generate regions using private clustering. Most early differentially private clustering methods require substantial effort to handle issues concerning the initial selection of centroids, number of necessary iterations, and the necessary additive noise [21, 29, 30, 169, 172, 272]. We adapt the expanded uniform grid K-means (EUGKM) method [206, 207], which has been shown to perform well while satisfying ϵ-DP.

In short, EUGKM consists of two steps: initial cluster centroid generation and K-means-style clustering. To generate the locations of an initial set of K centroids, EUGKM uses the concept of sphere packing to randomly generate points within the bounds of the dataset that ensures that all centroids are evenly (but not necessarily equally) spaced across the data space. The main advantage of this method is that it can be done without access to individual data records, thus maintaining privacy. A uniform grid is then generated using Equation 3.1, and ϵ_1 is used to control the grid size. Data points are assigned to a grid cell, the total number for each cell is calculated, and Laplace noise of $\text{Lap}(\frac{1}{\epsilon_1})$ is added to the count in each cell. Grid cells are then 'allocated' to their nearest centroid and a weighted K-means style procedure for optimisation is initiated, where the cell-centroid distances are weighted by the (noisy) number of points in each cell. Once the centroid locations no longer change, the clustering procedure terminates. We use these centroid locations to generate K Voronoi regions to which each real data point is assigned. For each cluster region, we obtain the number of points and, as we have interacted with the real data again, we add noise to each Voronoi region's count: $n_i' = n_i + \text{Lap}(\frac{1}{\epsilon_2})$.

Theorem 2. *EUGKM has time complexity of* $O(NK)$.

Proof. EUGKM has three main stages: the creation of a uniform grid, the clustering of grid cells, and the addition of Laplace noise to regions. Creating initial centroids is independent of the data and the number of regions. From Theorem 1, the time complexity to create the uniform grid is $O(N)$. The complexity of the K-means clustering stage is $O(NKdi)$, where d is the number of dimensions (i.e., two), and i is the number of iterations. As these are constants, the complexity reduces to $O(NK)$. Adding noise to clusters is intuitively an $O(K)$ operation. Summing these terms together gives an overall time complexity of $O(N+NK+K)$, which reduces to $O(NK)$ as this term dominates. ■

3.2.2 Private Data Generation

Generating synthetic data from a domain without imposing any constraints can be done in many ways. For example, sampling from a uniform distribution over the entire domain will maximise the entropy, but it will adversely affect fidelity as it is unlikely to preserve the underlying characteristics or properties of the real data. The task is made more difficult as we aim to preserve the complex patterns of the true spatial data while imposing the strict requirements of ϵ-DP.

This section introduces differentially private synthetic data generation methods for use in conjunction with any partitioning method. Note that, when generating synthetic points with any method, we can ensure that points are not generated in regions that are unlikely to contain points (e.g., seas and rivers) by specifying ‘out-of-bounds’ regions. Any synthetic data points that lay within these regions are discarded, and alternative points are generated.

Uniform Distribution

As private partitioning already approximately captures the overall distribution of the points, a simple method for synthetic point generation is to sample at random from a uniform distribution. Uniform random sampling is data independent, so no further noise is needed at this stage to preserve privacy (i.e., $\epsilon_3 = 0$). We further reduce the size of the region by dividing each region into triangles, where each triangle consists of the region’s centroid and two adjacent vertices of the region. We generate points randomly within each triangle in proportion to each triangle’s area, using the triangle point picking method [246]. Alternatively, one could find the minimum bounding rectangle, generate points uniformly within this rectangle, and perform rejection sampling with respect to the region boundaries. However, clustering-based partitioning can occasionally create oddly shaped regions, which makes the latter approach time-consuming.

Weighted Uniform Distribution

A more nuanced approach is to use information from neighbouring regions to define the point distribution. The weighted uniform distribution (WUD) approach subdivides each region and distributes points uniformly across each sub-region, with the number of points in each sub-region influenced by the characteristics of the sub-region and neighbouring region [266].

Each region R_i is split into J sub-regions, where J is the number of boundaries that R_i has. The number of points $n'_{i,j}$ in sub-region $R_{i,j}$ is based on its area and the noisy number of points in the neighbouring region. It is obtained using the following guideline:

$$n'_{i,j} = n'_i \left(\omega \frac{A_{i,j}}{A_i} + (1 - \omega) \frac{m'_{i,j}}{m'_i} \right) \tag{3.4}$$

where $A_{i,j}$ and A_i are the areas of $R_{i,j}$ and R_i, respectively; $m'_{i,j}$ and m'_i are the noisy number of points in the neighbouring region(s) to $R_{i,j}$ and R_i, respectively; and $0 \leq \omega \leq 1$ is a weighting factor. By definition, $m'_i = \sum_j m'_{i,j}$. Setting $\omega = 0.5$ to give equal weight between the areas and populations of (sub-)regions is recommended. Figure 3.1 shows an example of the weighted uniform method where $n'_i = 200$. Once the number of points in each sub-region is determined, points are generated using the triangle point picking method. As the boundary regions are private (due to the partitioning method) and only the noisy number of points in any region are ever used, the post-processing property of DP negates further noise addition. Hence, $\epsilon_3 = 0$ here.

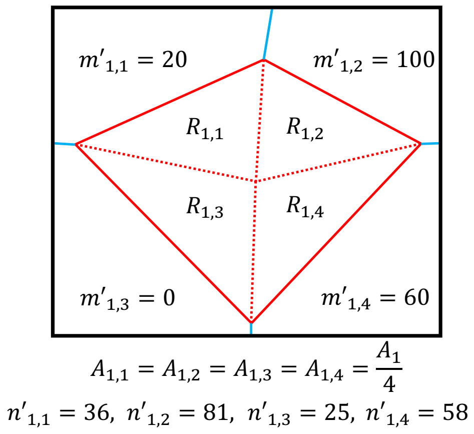

$m'_{1,1} = 20$
$m'_{1,2} = 100$
$R_{1,1}$
$R_{1,2}$
$R_{1,3}$
$R_{1,4}$
$m'_{1,3} = 0$
$m'_{1,4} = 60$
$A_{1,1} = A_{1,2} = A_{1,3} = A_{1,4} = \frac{A_1}{4}$
$n'_{1,1} = 36,\ n'_{1,2} = 81,\ n'_{1,3} = 25,\ n'_{1,4} = 58$

Figure 3.1: Example of the weighted uniform distribution method used to determine the number of points in each sub-region

Kernel Density Estimation

Kernel density estimation is a statistical approach to estimate the density function of a distribution. Using KDE as a basis for synthetic data generation can ensure higher fidelity between the synthetic and original data. In our setting, the kernel density estimator, $\widehat{\mathrm{f}}(\mathbf{x})$, is defined as:

$$\widehat{\mathrm{f}}(\mathbf{x}) = \frac{1}{N} \sum_{j=1}^{N} \Phi(\mathbf{x} - \mathbf{x}_j) \tag{3.5}$$

where $\mathbf{x}$ is a two-dimensional vector consisting of x- and y-co-ordinates, N is the number of points in the dataset (that is the basis for the kernel), and Φ is the kernel function.

Kernel Density Estimator Construction. While there have been numerous attempts to privatise KDE [10, 115, 123], these methods are not well-suited to this problem (i.e., sampling multiple times from a private KDE). Prior efforts adopt relaxed privacy definitions, such as (ϵ, δ)-DP [115], or perform post hoc testing of KDE samples for privacy [123]. Aldà and Rubinstein [10] use the Gaussian kernel, which results in oversmoothing in this setting, leading to poor quality synthetic data.

We instead use a two-dimensional Laplace kernel, owing to the widespread use of its one-dimensional counterpart in other DP work. Specifically, we use the polar Laplace distribution, which has the probability density function:

$$\Phi(\mathbf{x} - \mathbf{x}_j) \equiv \Phi(r, \theta) = \frac{\exp(-r/\tilde{h})}{2\pi\tilde{h}} \tag{3.6}$$

where $r = \|\mathbf{x} - \mathbf{x}_j\|$, θ is the angle between $\mathbf{x}$ and $\mathbf{x}_j$, and $\tilde{h}$ is a normalisation (or smoothing) factor. Other kernels (e.g., uniform, triangular, Gaussian) can be used and, while most can be privatised relatively easily, they generally produce synthetic data that has low utility. Conversely, kernels that generate data with high utility cannot be shown to satisfy strict ϵ-DP, even if they might do so in most realistic circumstances.

To obtain a differentially private kernel for region R_i, it is necessary to tune the kernel function in each region R_i such that the probability ratio between the two most distal points in R_i is no more than e^{ϵ}, as required by Definition 1. Hence, if $||R_i||$ is the maximum distance between any two locations (not necessarily in $\mathcal{P}$) in R_i, the smoothing parameter for R_i is set to:

$$\tilde{h}_i = \frac{||R_i||}{\epsilon *} \tag{3.7}$$

Theorem 3. *The kernel function in Equation 3.6 with $\tilde{h} = \frac{||R||}{\epsilon}$ satisfies ϵ-differential privacy*

Proof. For the kernel function to satisfy ϵ-DP, the probability ratio between the kernel function at the two most distal points (p_A and p_B) in R must be no greater than e^{ϵ}. That is,

$$e^{-\epsilon} \leq \frac{\Phi_A}{\Phi_B} \leq e^{\epsilon} \tag{3.8}$$

Defining $\Phi_A = \Phi(0, \theta)$ and $\Phi_B = \Phi(||R||, \theta)$ yields:

$$\Phi_A = \Phi(0, \theta) = \frac{\epsilon \exp(0)}{2\pi ||R||} = \frac{\epsilon}{2\pi ||R||}$$
$$\Phi_B = \Phi(||R||, \theta) = \frac{\epsilon \exp(-\epsilon ||R|| / ||R||)}{2\pi ||R||} = \frac{\epsilon \exp(-\epsilon)}{2\pi ||R||}$$

Substituting these into Equation 3.8 gives:

$$\frac{\Phi_A}{\Phi_B} = \frac{\epsilon}{2\pi ||R||} \times \frac{2\pi ||R||}{\epsilon \exp(-\epsilon)} = \frac{1}{\exp(-\epsilon)} = e^{\epsilon}$$

■

Synthetic Data Generation. To generate a synthetic point s, we can utilise a convenient property of kernel density estimation: sampling from the full KDE is equivalent to first sampling one of the n points $\mathbf{x}_j$, then sampling from the kernel around $\mathbf{x}_j$ (see Theorem 4). From Equation 3.6, we can see that r and θ can be sampled independently – that is, $\Phi(r, \theta) = \Phi(r)\Phi(\theta)$, where:

$$\Phi(r) = \frac{\exp(-r/\tilde{h})}{\tilde{h}} \tag{3.9}$$

$$\Phi(\theta) = \frac{1}{2\pi} \tag{3.10}$$

To this end, we first sample from $\Phi(r)$, and then sample from $\Phi(\theta)$; the latter being equivalent to sampling randomly from the uniform distribution with bounds $(0, 2\pi]$. Once values for r and θ are obtained, this displacement is added to the sampled real point to give s, where $x_s = x_p + r\cos\theta$, and $y_s = y_p + r\sin\theta$.

There is a risk that real points could be sampled many times, which would lead to privacy leakage that could reveal the true location of an individual. To avoid this, the sample procedure is modified such that $\epsilon^* = \epsilon_3/\Lambda$, which allows each real point to be sampled at most Λ times (using sequential composition). This ensures that the target level of privacy protection is reached, but not exceeded. If this limit is reached, or if $n_i = 0$ and $n'_i > 0$, excess points are generated uniformly at random, which has no negative privacy consequences. Setting $\Lambda = 2$ is appropriate, as $n'_i \leq 2n_i$ in most cases. This sampling process is repeated until n'_i points are generated in each region R_i. Finally, as a sample generated this way has the same distribution as the KDE and the KDE satisfies DP, it follows that the synthetic data satisfies DP.

As an aside, we can briefly go over why this sampling process is equivalent to sampling from the full KDE. Consider two random variables: X and Y, where X represents all the points in the real dataset (from which we sample with uniform probability), and Y has the same distribution as the kernel function. The sampling process outlined is equivalent to sampling from the combined distribution $X + Y$, which we now show is equivalent to sampling from the KDE directly (i.e., Y).

Theorem 4. *The kernel density estimate of $X + Y$ is the same as the kernel density estimate of Y.*

Proof. Using F_Φ to denote the cumulative density function of the kernel function, the cumulative density function of Y is given by:

$$F_Y(\mathbf{x}) = \Pr(Y \le \mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} F_\Phi(\mathbf{x} - \mathbf{x}_i) \tag{3.11}$$

We can sample from X such that $\Pr(X = \mathbf{x}_i) = \frac{1}{N}$. Therefore,

$$\begin{aligned}
F_{X+Y}(\mathbf{x}) &= \Pr(X + Y \le \mathbf{x}) \\
&= \sum_{i=1}^{N} \Pr(X + Y \le \mathbf{x} \mid X = \mathbf{x}_i) \Pr(X = \mathbf{x}_i) \\
&= \sum_{i=1}^{N} \Pr(\mathbf{x}_i + Y \le \mathbf{x}) \frac{1}{N} \\
&= \frac{1}{N}\sum_{i=1}^{N} \Pr(Y \le \mathbf{x} - \mathbf{x}_i) \\
&= \frac{1}{N}\sum_{i=1}^{N} F_\Phi(\mathbf{x} - \mathbf{x}_i) \\
&= F_Y(\mathbf{x})
\end{aligned}$$

■

When generating the synthetic points within a region, all points would ideally be located within the pre-determined geographic boundaries that are informed by real-world knowledge. However, points will be generated within these out-of-bounds regions occasionally, which will require additional rounds of point generation. Importantly, this does not affect the overall time complexity of point generation, which remains $O(K)$ for all three data generation methods.

3.3 Road Network-Aware Data Generation

The methods presented thus far follow the common assumption that there is limited knowledge of the underlying geography. In many cases, however, more significant information is available both to the data owner and to the public. For example, for a dataset of vehicle trajectories, it is reasonable to assume that all points in the dataset will correspond to points on (or very close to) segments of a city's road network. Therefore, when generating points, one should ensure that all synthetic points are similarly aligned to road segments. We can again use external knowledge to infer where individuals are unlikely to be located (e.g., in seas, rivers, military bases). Importantly, enforcing any of these constraints does not use any information not already in the public domain, and can therefore be done without using any of the privacy budget. For example, the location of roads and boundaries of seas are available (often to a high level of detail) through a range of mapping platforms and government open data repositories. This section proposes a novel differentially private method for generating synthetic location data that exploits the underlying structure of the road network.

Notation

Let the graph $\mathcal{G}(\mathcal{E}, \mathcal{V})$ represent the road network, in which $\mathcal{E}$ and $\mathcal{V}$ represent the road segments and road intersections, respectively. For each individual location $p \in \mathcal{P}$, there exists an edge $e_p \in \mathcal{E}$ that is the closest edge (distance-wise) to p. Two variables help to map p onto e_p. The first, $d_{\perp}^{p}$, gives the perpendicular distance from p to e_p. The second, $d_{\parallel}^{p}$, gives the parallel distance along e_p between $v_i^{e_p}$ and p. These variables are illustrated in Figure 3.2.

Noise Addition

If the real data points are not perfectly aligned with the assumed road network, it is necessary to map-match them to edges in the graph (i.e., obtain e_p for all $p \in \mathcal{P}$). Once complete, the number of points for which that edge is the nearest is determined, and denoted as n_e. We use this count to determine the noisy number of points that will be generated along each edge by first adding Laplace noise to n_e. The privacy budget is represented as ϵ_1. A naïve approach would be to use these values as the noisy counts. However, this would result in a large amount of additional noise throughout the dataset, especially when a large proportion of edges have low/zero counts. Therefore, we reduce the influence of the noise by denoting this 'intermediate' count as n_e^*, and performing a post-processing step to obtain $n_e' = \frac{N \times n_e^*}{N^*}$, where $N^* = \sum_e n_e^*$ is the sum of intermediate noisy counts for all edges. Furthermore, we set $n_e' = 0$ for all edges where $n_e' \leq \varphi$, where φ is a threshold value. Imposing this threshold also reduces the impact of the added Laplace noise. DP is still satisfied as these are post-processing operations.

Determining the threshold value. The value of φ can impact the quality of the synthetic data and may vary dynamically with ϵ_1 (as the magnitude of added noise depends on ϵ_1). The optimal value for φ will balance the number of points added to edges where $n_e = 0$, and the number of points 'lost' for edges where $n_e' \leq \varphi$ and $n_e \neq 0$. However, trying to find this equilibrium directly requires knowing the true number of points on each edge, which would violate DP. To obtain a good approximation for φ, we use the inverse cumulative distribution function of the Laplace distribution, defined as:

$$Q = \begin{cases} \mu + \frac{\ln(2F)}{\epsilon_1} & \text{if } F \leq 0.5 \\ \mu + \frac{-\ln(2-2F)}{\epsilon_1} & \text{if } F \geq 0.5 \end{cases} \tag{3.12}$$

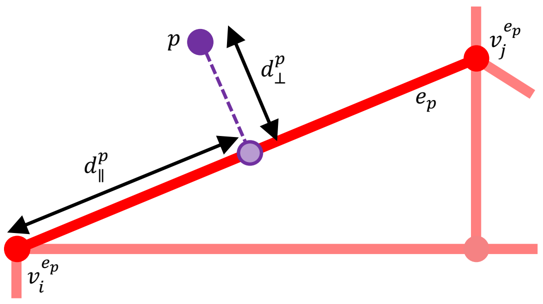


Figure 3.2: Diagram showing p, e_p, $d_{\perp}^{p}$, and $d_{\parallel}^{p}$

where Q is the quantile of the Laplace distribution, μ is the mean of the distribution (i.e., n_e), and F is the value of the cumulative distribution function.

The intuition is that setting $\varphi = Q_{95}$ removes approximately 95% of the added noise, for example. When $n_e = 0$, then $\mu = n_e = 0$, and so $Q = 0$ when $F \leq 0.5$ (disbarring negative counts). As such, only the second term is needed. Furthermore, when ϵ_1 is very small, the above term can be very large, which also causes adverse distortion to the dataset. Thus, an upper limit on the value φ can take is imposed (here, set to be 10). Hence, φ is defined as:

$$\varphi = \min\left(\frac{-\ln(2-2F)}{\epsilon_1}, 10\right) \tag{3.13}$$

Experimentally, setting $F = 0.9$ (i.e., removing about 90% of the noise) gives good results, and this setting is used henceforth.

Synthetic Data Generation

To generate a synthetic point s along an edge, we must fix (a) the distance along e that s is, (b) the perpendicular distance from e that s is, and (c) the 'side' of the edge that s is in relation to e. For (a), we could assign a distance at random from a uniform distribution. However, for very long roads, this could result in synthetic points being far from the real point locations, which would possibly reduce the synthetic data's utility. Instead, each edge can be summarised with a micro-histogram using the values of $d_{\parallel}^{p}$. We use H bins of equal width as this is data-independent, although bins of uneven width can be used with minimal modifications. To preserve privacy, noise (= $\text{Lap}(\frac{1}{\epsilon_2})$) is added to the count of each bin. We sample from this noisy histogram to determine the bin in which s lies. The exact value for $d_{\parallel}^{s}$ is determined by sampling from a uniform distribution with bounds corresponding to the bounds of the histogram bin. We sample from the histogram n'_e times to generate the necessary values for $d_{\parallel}^{s}$ (note that $e_s \equiv e_p$). This process is illustrated in Figure 3.3.

For (b), we use the same approach to determine the values of $d_{\perp}^{s}$, with ϵ_3 as the privacy budget when adding noise to the histogram. When the values for $d_{\perp}^{s}$ and $d_{\parallel}^{s}$ are set, there are two possible locations for s. For (c), we select between these two locations with equal probability to determine the final location of s. When $n_e = 0$, we define the range of histogram values such that $d_{\perp}^{s}$ takes a value in the range (0, 10) metres and $d_{\parallel}^{s}$ takes a value in the range (0, $|e_s|$), where $|e_s|$ is the length of e_s. This process is applied to all edges in $\mathcal{E}$ where $n'_e > 0$ until the entire synthetic dataset, $\mathcal{S}$, is created.

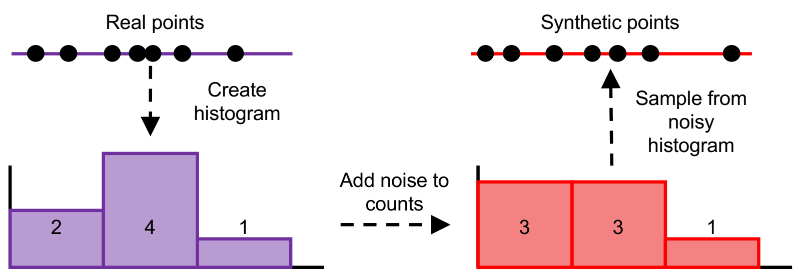


Figure 3.3: Example showing the process for sampling from the edge-level micro-histograms to find $d^s_{\parallel}$ and $d^s_{\perp}$

Histogram bin choice. We now discuss how to choose H, which affects the downstream utility of the synthetic data. The aim is to balance the overall noise added to an edge with the location accuracy along an edge. For example, having a high number of bins will be beneficial for describing locations accurately, but will involve high noise addition, which will negatively affect accuracy during the histogram sampling stage. The converse is true for low values of H.

To assess this trade-off and to determine the optimal value for H, we can consider a range count query. Specifically, consider that a road segment with N points is divided into H histogram bins, and that our range query covers a proportion k of the road (i.e., kH bins). The error in answering this range query has two competing components: privacy noise error and non-uniformity error. Privacy noise error comes from the noise added to the count of each bin, as required by DP, and it increases as H increases as Laplace noise needs to be added less often. A non-uniformity error arises when answering a query that partially intersects a bin. That is, we do not know whether points in a bin will be included in the query response, owing to the non-uniformity of the data distribution. The non-uniformity error decreases with more bins, as it is more likely that a query covers entire bins and all points lie within these bins. Finding the optimal value for H requires equating these two competing error terms.

Theorem 5. *The optimal value for H is $O\left(\sqrt{\epsilon N}\right)$.*

Proof. The Laplace mechanism for count queries has a variance of $\frac{2}{\epsilon^2}$, corresponding to a standard deviation of $\frac{\sqrt{2}}{\epsilon}$. Hence, the expected magnitude of the noise error per bin is $\frac{\sqrt{2}}{\epsilon}$. Given the query covers kH bins, the noise error will be proportional to $\frac{\sqrt{2kH}}{\epsilon}$. Assuming points are distributed evenly, each bin has, on average, $\frac{N}{H}$ points. Hence, the average non-uniformity error for each partially intersected bin is proportional to $\frac{N}{H}$. As there are (at most) two partially covered bins, the total non-uniformity error is proportional to $\frac{2N}{H}$. From these two definitions, the total error in answering the range count query is: $\frac{\sqrt{2kH}}{\epsilon} + \frac{2N}{H}$. Treating k as a constant and equating the two terms, ultimately yields: $H = O\left(\sqrt{\epsilon N}\right)$. ■

In this proof, N corresponds to n'_e, the noisy count of the edge; and ϵ corresponds to either ϵ_2 or ϵ_3. The value for ϵ can be chosen empirically, and we find that setting $H = \sqrt{N}$ gives effective results, as demonstrated in Section 3.4.2.

Time Complexity

Whereas the theoretical runtimes of the partitioning-based methods are dependent on the number of points in the dataset and the number of regions, it is intuitive to see that the expected runtime of the road network-based method is also dependent on the size of the road network. Specifically, runtime is dependent on the number of edges in the road network, $|\mathcal{E}|$.

Theorem 6. *The time complexity for the road network-based approach is* $O\left(N + (N|\mathcal{E}|\epsilon)^{1/2}\right)$.

Proof. When generating points along a single edge, two micro-histograms are created along each edge. For each micro-histogram, each point along the edge is assigned to a bin, with Laplace noise then added to the count of each bin. This stage has an overall time complexity of $O(n_e + \sqrt{H})$, which is equivalent to $O(n_e + \sqrt{\epsilon n_e})$. This process is repeated for each edge in the road network, leading to an overall time complexity of $O\left(|\mathcal{E}|(n_e + \sqrt{\epsilon n_e})\right)$. However, assuming that points are distributed evenly, $n_e = \frac{N}{|\mathcal{E}|}$, which reduces the overall time complexity to: $O\left(N + (N|\mathcal{E}|\epsilon)^{1/2}\right)$. ■

Given the differences between the time complexities of the partitioning- and road network-based methods, it is hard to compare them on a theoretical level. Furthermore, the real-world focus of this work leads to differences between the theoretical and practical runtimes of the solutions. For example, location data is rarely spread evenly across a city's road network. Similarly, including geographic constraints in the data generation stages will sometimes lengthen runtimes. Hence, it is more practical to compare them empirically.

3.4 Evaluation

The accuracy and efficiency of the data generation methods can now be assessed using three utility measures and common location analytics queries.

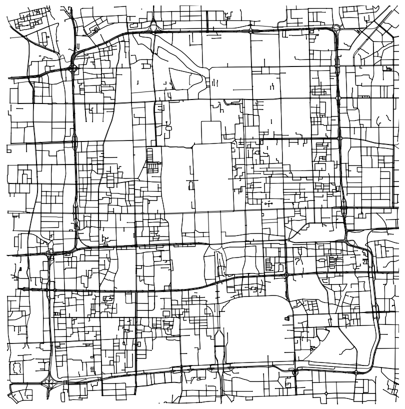
3.4.1 Outline

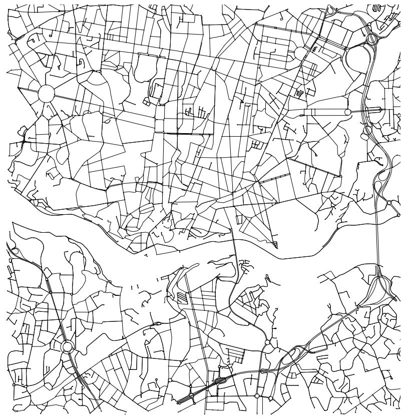
Datasets. Synthetic data is generated using real location data from three cities with different topographies and sizes: Beijing, Porto, and New York City. Information on the datasets is included in Table 3.1, and Figure 3.4 shows the structure of the road network in each city. Beijing covers the largest area, and its road network has a moderately strong grid-like structure. New York has a very strong grid structure, which is common in North American cities, and this contrasts with Porto where the road network has a less defined structure, which is characteristic of European cities.

Only the longitude-latitude pairs of each record are extracted (i.e., any temporal information is excluded). Although the taxi trajectory points are correlated, we model each point to represent an independent individual in the dataset. Coastline data is extracted from OpenStreetMap [184], and it is used to define 'out-of-bounds' regions that represent major bodies of water, such as seas and rivers. Any points in the original data that are located within these regions are removed, and the boundaries are used to ensure that no synthetic points are created in these regions. The same technique can be used to add further geographical restrictions (e.g., forests, military bases) on the presence of real or synthetic individuals.

Table 3.1: Dataset information

City	Data Type	Number of Points	Number of Edges	Area (km^2)	Density (km^{-2})	Boundaries North	South	West	East	Reference
Beijing	Taxi Trajectories	158,260	7,913	60.6	2,612	39.954	39.862	116.330	116.450	[268, 269]
Porto	Taxi Trajectories	79,360	3,968	32.5	2,442	41.168	41.123	-8.635	-8.576	[100]
New York City	311 Calls	163,220	8,161	59.1	2,762	Manhattan Island				[178]


(a) Beijing


(b) Porto

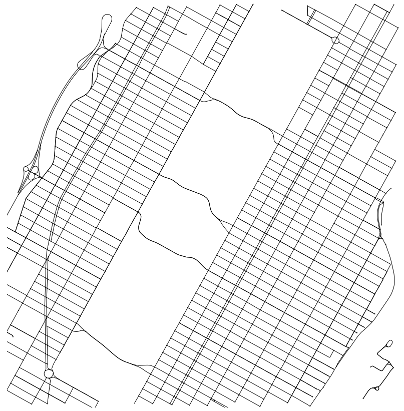

(c) New York City

Figure 3.4: Road network structures of Beijing, Porto, and New York City

Data for the 'driveable' road network is extracted from OpenStreetMap, using the `osmnx` Python package [31], with boundaries matching those detailed in Table 3.1. As pre-processing steps, we map-match each point in the cleaned datasets to the corresponding road network, remove any edges that are within the out-of-bounds areas, and calculate the values $d_{\perp}^{p}$ and $d_{\parallel}^{p}$ for each point. The final number of points and edges for each city is shown in Table 3.1.

Baselines. As discussed in Section 2.2, most existing work only publishes count data for grid cells/clusters, as opposed to generating co-ordinate data. Such data can be generated in these partitions using simple uniform sampling (Section 3.2.2), and so these extensions of existing methods act as realistic baselines. The terms 'UGrid-Uni', 'AGrid-Uni', and 'Clust-Uni' refer to the extension of the uniform grid, adaptive grid, and clustering-based partitioning methods, respectively.

Parameter Selection. For each dataset, we set $N = 20|\mathcal{E}|$, where $|\mathcal{E}|$ is the number of edges in the road network graph. This is done so that the number of grid cells is approximately equal to the number of edges for the road network-based solution (henceforth referred to simply as 'Road'). This allows for a fairer comparison between the methods as the amount of added noise will be more comparable. However, for clustering-based methods, having $K \approx |\mathcal{E}|$ would result in the regions exhibiting a grid-like structure, and so we set $K = 1{,}000$. By default, $\epsilon = 1$; the impact of varying and splitting the privacy budget is evaluated in Section 3.4.2.

Implementation. All experiments were conducted using Matlab 2020b on a Macbook Pro, with a 2.3GHz Intel Core i5 processor and 8GB RAM. Matlab's in-built parallelisation toolbox was utilised to speed-up the data generation process.

Utility Measures. We initially assess utility through three measures – Chamfer distance (CD), normalised cell error (NCE), and mean edge distance difference (MEDD) – all of which aim to quantify the synthetic data's fidelity with the real data.

As a continuous and pairwise smooth function, Chamfer distance is a well-established measure for quantifying the distance between two point datasets. For a set of real points and synthetic points, CD is defined as:

$$CD(\mathcal{P}, \mathcal{S}) = \sum_{p \in \mathcal{P}} \min_{s \in \mathcal{S}} ||p - s||^2 + \sum_{s \in \mathcal{S}} \min_{p \in \mathcal{P}} ||p - s||^2 \tag{3.14}$$

where $|| \cdot ||$ is some distance measure (here, normalised Euclidean distance is used).

To calculate NCE, we divide the region into uniformly sized cells and obtain c_i^{real} and c_i^{synth}, which are the number of points in cell i for the real and synthetic datasets, respectively. To avoid the cell granularity from influencing results, NCE is normalised by the number of real points. Hence, it is defined as:

$$NCE = \frac{1}{|\mathcal{P}|} \sum_{i} \left| c_i^{\text{real}} - c_i^{\text{synth}} \right| \tag{3.15}$$

In our evaluation, we divide the entire region into a uniform grid where each individual grid cell has approximate real-life dimensions of 100m × 100m.

Table 3.2: Normalised cell errors (NCE), Chamfer distances (CD), mean edge distance differences (MEDD; in metres), and runtimes (in seconds) for default settings; baselines denoted by asterisks (*)

Data Gen.		Beijing				Porto				New York City			
Method		NCE	CD	MEDD	Time	NCE	CD	MEDD	Time	NCE	CD	MEDD	Time
UGrid	Uni*	0.360	3.49	10.64	64.25	0.165	3.89	6.46	62.58	0.374	2.49	15.36	91.44
UGrid	WUD	0.332	3.02	8.59	295.97	**0.152**	3.16	5.36	131.62	0.366	2.30	15.04	233.39
UGrid	KDE	0.297	**2.91**	8.62	39.89	0.160	3.32	5.22	99.63	0.309	2.02	12.86	749.28
AGrid	Uni*	0.379	4.20	11.83	55.92	0.188	4.19	6.23	65.33	0.310	3.01	14.99	159.19
AGrid	WUD	0.362	3.79	10.02	1336.85	0.180	3.79	5.20	399.73	0.307	2.84	14.63	1469.85
AGrid	KDE	**0.285**	3.36	8.84	63.82	0.160	3.18	4.76	265.71	0.259	2.33	11.34	1876.09
Clust	Uni*	0.876	10.47	27.83	10.81	0.407	6.75	13.00	**17.51**	0.610	5.82	19.48	**25.17**
Clust	WUD	0.866	10.18	26.19	28.88	0.391	6.39	12.38	32.50	0.591	4.82	18.63	29.11
Clust	KDE	0.616	6.69	19.23	**8.23**	0.272	4.42	8.85	85.93	0.463	3.46	16.41	842.54
Road		0.316	3.54	**1.97**	29.09	0.184	**2.66**	**0.94**	16.87	**0.200**	**0.70**	**0.70**	51.40

While CD and NCE quantify the error between just the synthetic and real datasets, MEDD quantifies the error between the two datasets with respect to a graph – here, the road network. MEDD can be used to quantify the preservation of network alignment of the synthetic points. It is defined as:

$$MEDD = \left| \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} d_{\perp}^{p} - \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} d_{\perp}^{s} \right| \tag{3.16}$$

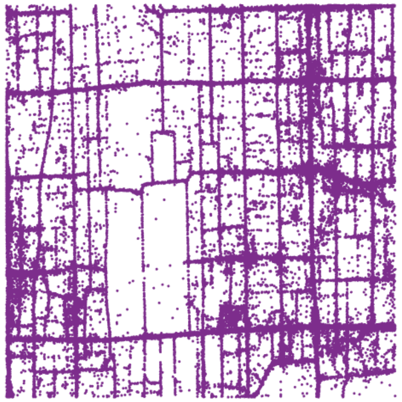
3.4.2 Results

Summary

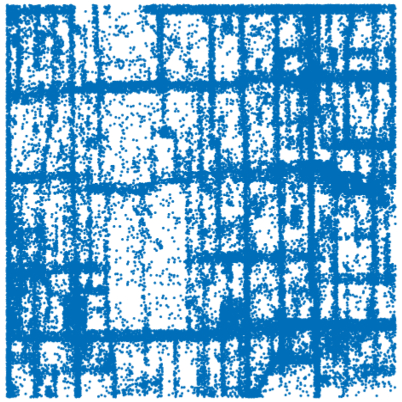
Figure 3.5 shows the visual similarity between the real and synthetic data. Although all methods preserve the underlying structure to some degree, utilising the geographical constraints explicitly in the data generation stage produces synthetic data that has much stronger visual similarity to the real data than the partitioning-based methods. Quantitatively, Table 3.2 shows the NCE, CD, and MEDD values for the four methods, as well as the three approaches for generating synthetic data points within defined regions, and the runtimes for each.

Adopting KDE for grid-based partitioning methods improves data quality compared to extensions of existing methods. As KDE almost always outperforms WUD in accuracy terms, we adopt it as the default choice for data generation. AGrid performs similarly to UGrid, unless the city's network is more structured (e.g., New York) in which case it is markedly better (in terms of NCE). However, it generally takes longer to run, which may make UGrid preferable. For clustering-based partitioning, KDE offers more significant improvements compared to other approaches, although it fails to match the grid-based approaches in accuracy terms. This is primarily because larger regions lead to flatter kernels due to the requirements of DP (i.e, $||R||$ is larger, meaning $\tilde{h}$ is larger).

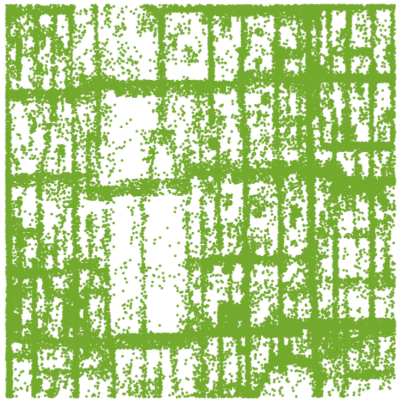
Using Road offers even greater improvements by being up to 28x more accurate over the baselines (vs. Clust-Uni, MEDD, New York). Furthermore, Road is up to 3.9x faster than the baselines (vs. AGrid-Uni, Porto), and up to 37x faster than KDE approaches (vs. AGrid-KDE, New York). This highlights its suitability for generating large city-scale synthetic datasets



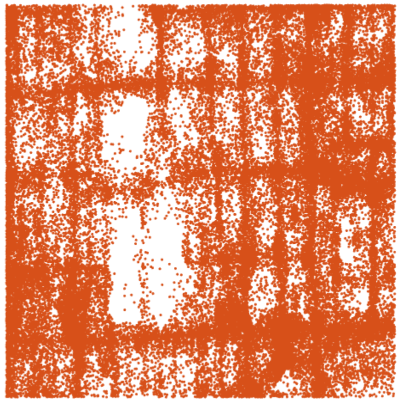
(a) Real points



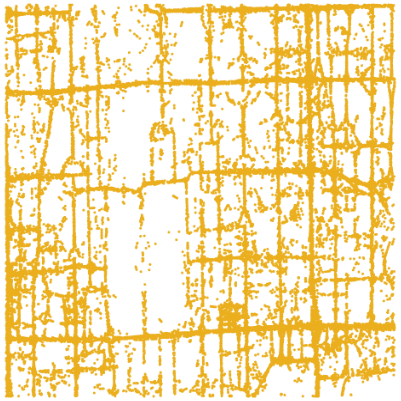
(b) UGrid-KDE



(c) AGrid-KDE

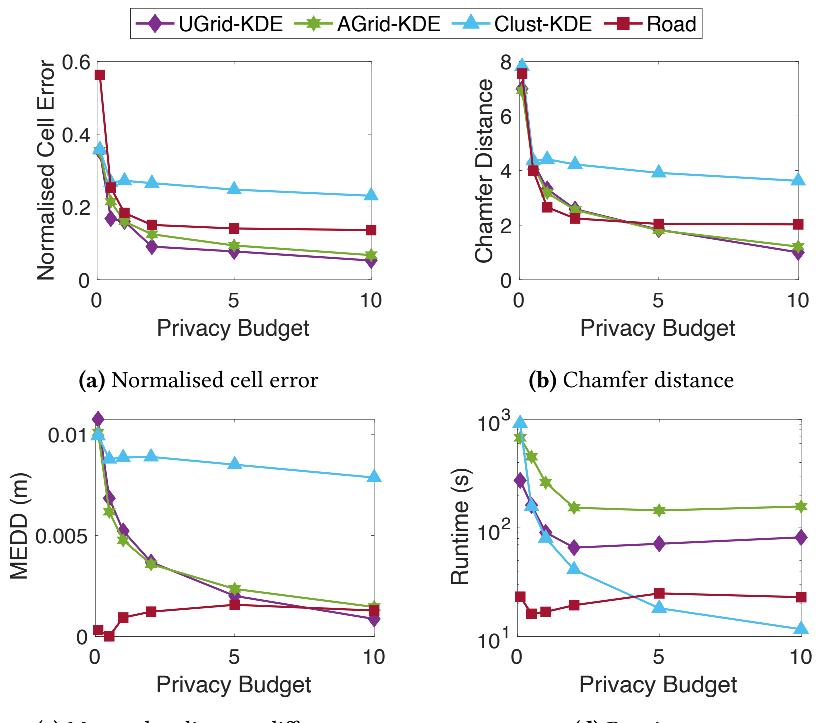


(d) Clust-KDE



(e) Road

Figure 3.5: Plots of real and synthetic data for data generation methods (Beijing)



(a) Normalised cell error

(b) Chamfer distance

(c) Mean edge distance difference

(d) Runtime

Figure 3.6: Privacy budget vs. utility (Porto)

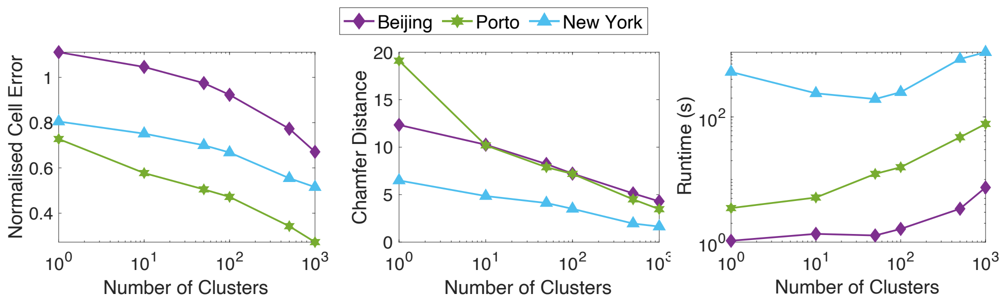
of high utility. In Porto and Beijing, where points are less closely aligned with the road network and the road network is less ordered, grid-based approaches are generally superior in accuracy terms. In New York, however, the real data adheres more tightly to the road network, which means that Road is much better at creating high quality synthetic data and achieves better MEDD values.

Varying Parameters

We also examine the effects that varying the key parameters have on the quality of the data.

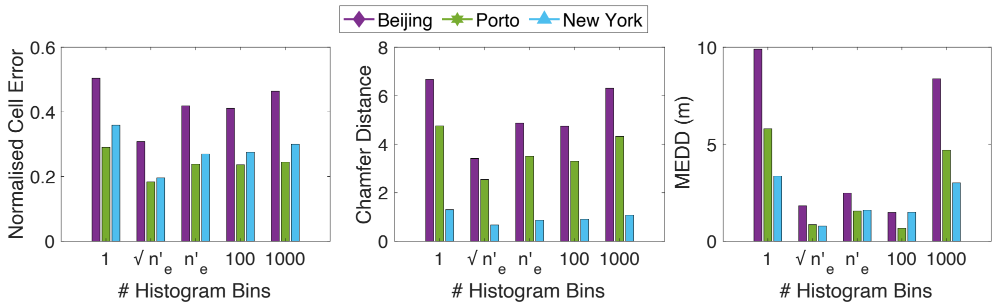
Privacy Budget. Figure 3.6 shows the effect of changing the privacy budget on the three utility measures and runtime (for Porto, although other cities exhibit similar profiles). In terms of accuracy, all methods behave as expected: accuracy decreases as ϵ decreases, due to the increase in the amount of added noise. For low ϵ values, runtime is higher for partitioning-based methods as it is more likely that generated points are 'out-of-bounds' or outside of the boundaries of R_i. Runtimes for grid-based methods increase slightly as ϵ increases beyond 5 as the number of cells grows in proportion to ϵ (cf. Equations 3.1–3.3). The runtimes for Road are consistently low for all ϵ, which further highlights its general suitability.

Number of Clusters. Figure 3.7 shows how utility and runtime change as the number of clusters changes. The following values for K are used: $\{1, 10, 50, 100, 500, 1{,}000\}$. As the number of clusters increases, both NCE and CD values decrease, although runtime increases exponentially. This is intuitive as smaller regions allow the kernel density estimate to be better tailored to the characteristics of the regions, at the expense of an increased runtime.



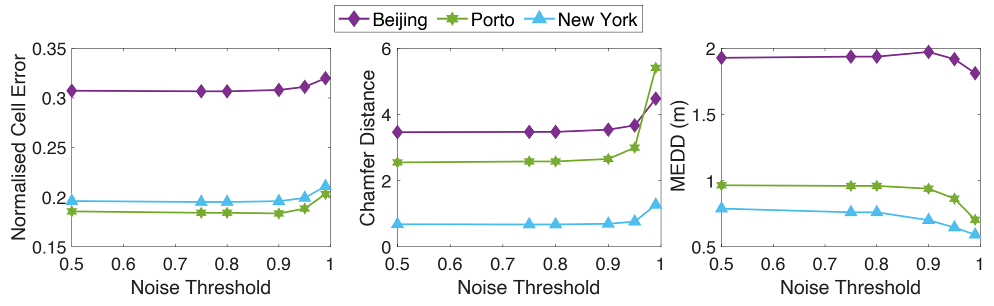
(a) Normalised cell error **(b)** Chamfer distance **(c)** Runtime

Figure 3.7: Number of clusters vs. utility



(a) Normalised cell error **(b)** Chamfer distance **(c)** MEDD

Figure 3.8: Number of histogram bins vs. utility



(a) Normalised cell error **(b)** Chamfer distance **(c)** MEDD

Figure 3.9: Noise threshold vs. utility

Number of Histogram Bins. In proving Theorem 5, setting $H = O(\sqrt{n'_e})$ was shown to be optimal, which can now be confirmed empirically. We consider five possible settings for H: $\{1, \sqrt{n'_e}, n'_e, 100, 1000\}$. Figure 3.8 shows values for the three utility measures for these five settings. As expected, setting $H = \sqrt{n'_e}$ is consistently the best performing approach, and this setting is recommended when Road is used.

Noise Threshold. In Section 3.3, we impose a threshold on the noise added to each edge, and discussed the utility trade-off in setting the threshold value. Although the optimal value for F cannot be obtained without violating privacy, its value can be obtained empirically. Figure 3.9 shows how values for the three utility measures change as F changes. Interestingly, when $F \leq 0.9$, there is little change in performance as the arbitrary upper threshold (set to 10) takes effect. As F increases from 0.9 towards 1, performance decreases, thus confirming that 0.9 is an appropriate setting to implement.

Privacy Budget Distribution

As well as choosing the privacy budget, one must also decide how to apportion it, where appropriate. This section examines several ways to apportion the privacy budget, and the results are included in Table 3.3. These allocations are used as the default settings for all other evaluations.

For all UGrid methods, $\epsilon_2 = 0$ as noise is only added once during the partitioning phase. Likewise, for data generation methods that do not use KDE, $\epsilon_3 = 0$. Hence, for UGrid methods that do not use KDE, $\epsilon_1 = \epsilon$ and no further investigation is needed. For UGrid methods with KDE-based data generation, we consider the following percentage splits between ϵ_1 and ϵ_3: 10–90, 20–80, 30–70, 40–60, 50–50, and their reverses. Table 3.3a shows how utility and runtime vary when these divisions are used. All three errors decrease as ϵ_1 increases, which is a consequence of less noise being added to the cell counts during partitioning. However, allocating more of the privacy budget to partitioning means that the kernel used during the generation phase is less well-defined. Therefore, it is not unexpected to see the runtime increase by up to 21x as a looser kernel means that points are more likely to be generated outside of region boundaries. As observed in Table 3.2 and supported by these experiments, both the partitioning and generation parts of UGrid-KDE benefit from high ϵ values. Hence, to accommodate this utility-efficiency trade-off, we set $\epsilon_1 = 0.6\epsilon$ and $\epsilon_3 = 0.4\epsilon$. This provides an approximate balance between utility and efficiency, although a higher ϵ_1 implies a desire to achieve greater utility at the expense of privacy.

For AGrid partitioning, Laplace noise is added twice when using adaptive grids, and so a higher proportion of the privacy budget is needed at this stage to achieve similar levels of utility compared to UGrid-KDE. Qardaji et al. [191] also recommend that $\epsilon_1 = \epsilon_2$. Hence, for KDE-based generation with AGrid partitioning, we consider the following percentage splits: 12.5–12.5–75; 25–25–50; 33–33–33; and 40–40–20. Table 3.3b shows that $\epsilon_1 = \epsilon_2 = 0.4\epsilon$ and $\epsilon_3 = 0.2\epsilon$ is generally best, although equal division of the privacy budget also gives strong results. A consequence of having larger values of ϵ_1 and ϵ_2 is that runtimes are higher, although this cost is reasonable given the utility benefits.

As noted previously, cluster-based partitioning generally leads to flatter kernels as regions tend to be larger. High ϵ_3 values help to keep $\tilde{h}$ at a value that prevents the kernel from becoming too flat (see Equation 3.7). This explains why, for cluster-based partitioning *with* KDE, setting $\epsilon_3 = 0.75\epsilon$ is best (see Table 3.3c). This leaves $\epsilon_1 = \epsilon_2 = 0.125\epsilon$, which suggests that setting $\epsilon_1 = \epsilon_2 = 0.5\epsilon$ is appropriate for cluster-based partitioning *without* KDE.

For Road, Laplace noise is added three times. It is intuitive to theorise that equal distribution of the privacy budget would give the best balance between the noise added to the number of points along an edge and the noise added to the micro-histograms on each edge. Empirical results, shown in Table 3.3d, support this theory, although all divisions of ϵ give similarly good results. Nevertheless, equal division of ϵ consistently provides the best utility, and it is therefore recommended as the suitable default setting.

Table 3.3: Privacy budget apportionment results; privacy budget as a percentage of ϵ shown

(a) UGrid-KDE

		Beijing				Porto				New York City			
ϵ_1	ϵ_3	NCE	CD	MEDD	Time	NCE	CD	MEDD	Time	NCE	CD	MEDD	Time
10	90	0.498	5.69	15.11	20.29	0.196	6.57	7.06	**40.66**	0.420	3.46	15.38	**333.04**
20	80	0.400	4.82	12.21	**14.49**	0.147	6.20	5.50	55.39	0.364	2.90	14.30	341.19
30	70	0.353	3.94	10.54	17.59	0.120	5.75	4.60	62.68	0.332	2.68	13.64	368.10
40	60	0.301	3.62	9.27	22.77	0.106	4.95	4.12	84.15	0.319	2.51	13.21	434.02
50	50	0.284	3.21	8.48	27.00	0.074	4.34	3.62	104.11	0.302	2.26	12.82	526.68
60	40	0.272	3.00	7.99	31.58	0.089	4.18	3.38	136.98	0.284	2.15	12.38	675.34
70	30	0.245	2.96	7.49	39.84	0.083	4.09	3.17	189.59	0.277	2.10	12.10	1008.98
80	20	0.232	2.85	7.09	49.71	0.080	4.15	2.97	379.82	0.268	2.03	11.84	1769.64
90	10	**0.224**	**2.67**	**6.74**	75.97	**0.075**	**3.91**	**2.75**	826.12	**0.248**	**1.97**	**11.55**	7051.78

(b) AGrid-KDE

			Beijing				Porto				New York City			
ϵ_1	ϵ_2	ϵ_3	NCE	CD	MEDD	Time	NCE	CD	MEDD	Time	NCE	CD	MEDD	Time
40	40	20	**0.261**	**3.55**	**8.27**	75.87	**0.100**	**3.92**	**3.28**	430.50	**0.244**	**2.55**	**10.71**	2027.89
25	25	50	0.306	3.99	9.58	40.75	0.124	4.26	3.90	114.94	0.263	2.91	11.88	591.92
12.5	12.5	75	0.398	4.99	12.31	**26.30**	0.157	5.61	4.93	**56.80**	0.335	3.92	13.77	**357.99**
33.3	33.3	33.3	0.270	3.75	8.67	52.22	0.106	3.93	3.45	189.25	0.251	2.84	11.24	974.39

(c) Clust-KDE

			Beijing				Porto				New York City			
ϵ_1	ϵ_2	ϵ_3	NCE	CD	MEDD	Time	NCE	CD	MEDD	Time	NCE	CD	MEDD	Time
40	40	20	0.617	7.48	19.37	33.83	0.252	6.71	9.10	160.94	0.456	4.16	16.50	1340.81
40	20	40	0.612	7.41	19.31	10.17	0.252	6.95	9.05	78.86	0.459	4.19	16.54	743.53
20	40	40	0.609	7.34	19.22	17.49	0.251	6.71	8.86	111.72	0.455	4.04	16.27	774.41
50	25	25	0.610	7.46	19.17	**7.92**	0.250	7.07	9.04	87.58	0.459	4.19	16.53	936.92
25	50	25	0.616	7.49	19.20	9.91	0.252	6.80	9.11	85.97	0.460	4.25	16.40	1211.57
25	25	50	0.607	7.43	19.08	16.58	0.248	6.76	8.93	77.05	0.464	4.17	16.29	621.45
75	12.5	12.5	0.614	7.71	19.41	16.60	0.249	7.15	9.06	210.84	0.454	4.26	16.51	2490.77
12.5	75	12.5	0.620	7.52	19.42	15.41	0.241	**6.70**	8.83	209.73	0.445	3.91	16.15	2725.65
12.5	12.5	75	**0.601**	**7.26**	**18.81**	12.90	**0.237**	7.00	**8.60**	**66.70**	**0.441**	**3.85**	**16.01**	**466.06**
33.3	33.3	33.3	0.610	7.54	19.26	9.93	0.246	6.79	8.90	82.16	0.457	3.99	16.30	775.51

(d) Road

			Beijing				Porto				New York City			
ϵ_1	ϵ_2	ϵ_3	NCE	CD	MEDD	Time	NCE	CD	MEDD	Time	NCE	CD	MEDD	Time
20	40	40	0.297	4.18	2.17	32.29	0.141	4.24	0.92	23.59	0.199	0.93	**0.71**	**42.37**
50	25	25	0.302	3.99	1.34	**26.30**	0.140	4.03	**0.36**	**22.38**	0.182	**0.74**	0.92	48.01
75	12.5	12.5	0.354	4.42	**0.77**	50.20	0.167	4.61	0.89	29.46	0.220	0.87	1.42	61.78
33.3	33.3	33.3	**0.286**	**3.90**	1.97	40.38	**0.135**	**3.92**	0.79	24.56	**0.179**	**0.74**	0.75	48.66

Real World Considerations

We next evaluate how well our methods model characteristics of real-world data, which is often messy and can exhibit high non-uniformity or skew.

Road Network Alignment. For Road, we assume that data points are well-aligned with the underlying road network. However, this is not always the case with real data, and there can be high error when map-matching raw data points to edges in the road network. This may be due to GPS sampling errors, map projection errors, or multi-lane roads being modelled as single lines of zero width. A city's topography can also influence the quality of synthetic data. For example, Porto's historic centre has many roads located close to each other, with no strong, regular underlying structure. This can make accurate map-matching difficult. Whereas 'uncorrected' data is used in the main experiments, we now perform experiments where we use the map-matched data as the input datasets (i.e., $d_{\perp}^{p} = 0$). In this new setting, Road is far superior to the other methods, which perform up to 18% worse. Hence, when the data is corrected, Road is up to 10%, 10%, and 120% more accurate than UGrid-KDE, AGrid-KDE, and Clust-KDE, respectively.

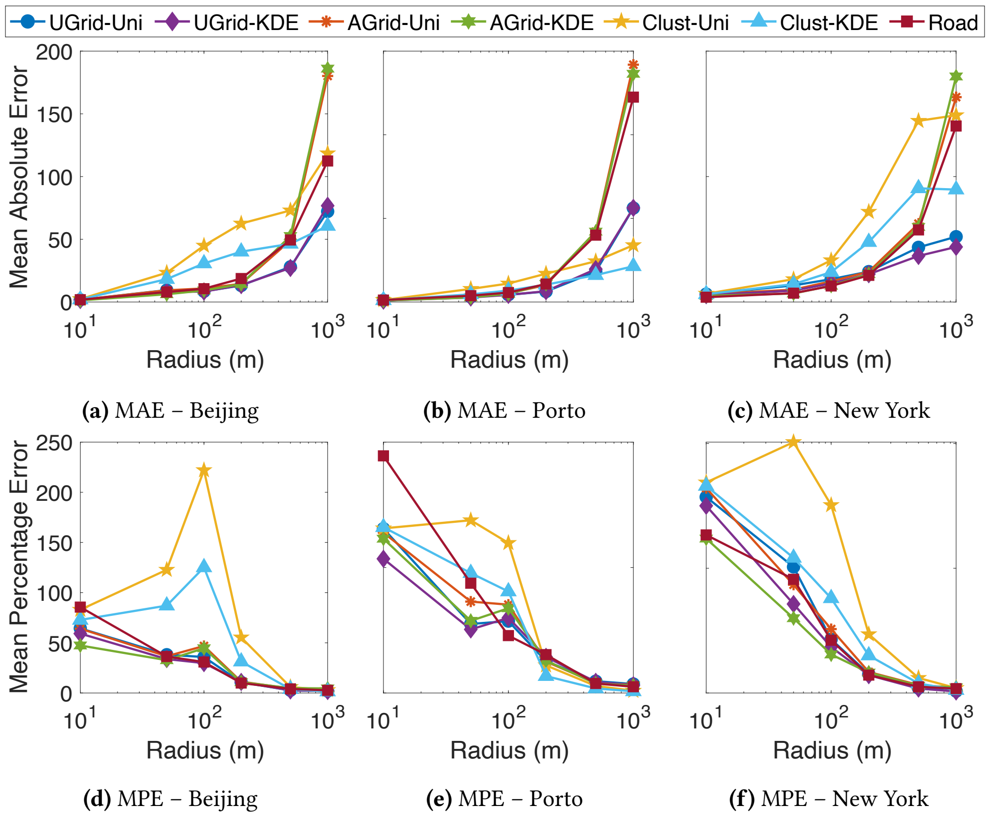
Uneven Population Densities. Population density in cities is rarely uniform, either across an area or along individual roads. In urban centres, point density may be reasonably uniform, while rural and suburban areas may experience more varied densities. To examine how our methods are affected by uneven densities, we create a dataset focused on a larger area of Beijing that includes more suburban areas. We set the expanded bounds of the studied region to the bounding box between (116.33, 39.97) and (116.48, 39.85), giving a new area of 90.0 km^2. In the new road network, $|\mathcal{E}| = 13{,}862$, and so we set $N = 20|\mathcal{E}| = 277{,}240$. The performance of UGrid-KDE actually improves – by 12.4% (NCE) and 15.2% (CD) – possibly because N is larger, which results in the added noise being less influential. Road's performance is essentially unchanged – 0.8% worse (NCE) and 4.8% better (CD) – as its performance is dependent on the road network, rather than point density. AGrid-KDE performs slightly worse – 7.8% (NCE) and 2.7% (CD) – possibly due to the adaptive grids being less well-aligned with the data. Clust-KDE performs much worse – 17.2% (NCE) and 21.0% (CD) – as a larger geographic extent leads to larger regions, which makes the kernels flatter.

3.4.3 Range and Hotspot Queries

Range Queries

Range queries are important in location analytics as they can be used to quickly assess how many customers are potentially available to a business, or measure accessibility to key services within a certain time, such as schools, hospitals, or vaccination centres. We specify a set, $\mathcal{F}$, of 200 arbitrary facilities in each city to be the basis of the range queries, and these places are randomly selected from the set of intersections in each city's road network. The extent of each range query is the circular region defined by the radius, ρ, centred on each facility f. To quantify the error, we use mean absolute error (MAE) and mean percentage error (MPE), defined as:

$$MAE = \frac{1}{|\mathcal{F}|} \sum_{f \in \mathcal{F}} \left| c_f^{\text{real}} - c_f^{\text{synth}} \right| \tag{3.17}$$



(a) MAE – Beijing (b) MAE – Porto (c) MAE – New York

(d) MPE – Beijing (e) MPE – Porto (f) MPE – New York

Figure 3.10: Range query radius vs. MAE and MPE

$$MPE = \frac{1}{|\mathcal{F}|} \sum_{f \in \mathcal{F}} \frac{|c_f^{\text{real}} - c_f^{\text{synth}}|}{c_f^{\text{real}}} \times 100\% \tag{3.18}$$

where c_f^{real} and c_f^{synth} respectively denote the number of real and synthetic points within ρ metres of facility f.

Figure 3.10 shows how the radius of the range query influences the error, for each method and city. For small ρ values, all partitioning-based methods outperform their respective baselines. Although MAE values are low for all cities, MPE values are higher for all methods in New York, where the data and road network are more structured. Interestingly, although Clust-KDE is generally less competitive overall, it performs better in the less-ordered Porto. Road is a viable alternative when ρ is small; although, as ρ increases, MAE increases rapidly. Likewise, AGrid methods perform notably worse for large ρ values. However, when one considers the error in relation to the dataset size, as well as the proportion of the query range to the entire dataset domain, this behaviour is acceptable. Despite this, UGrid methods offer strong alternatives, depending on the degree of road network alignment.

Hotspot Queries

Hotspot queries aim to identify locations in which there are a high number of individuals, They are fundamental in location analytics as they allow businesses to identify popular regions for advertising, city agencies to help manage congestion and traffic flow, etc. Here, we obtain kernel density estimates for the real and synthetic datasets at varying granularities.

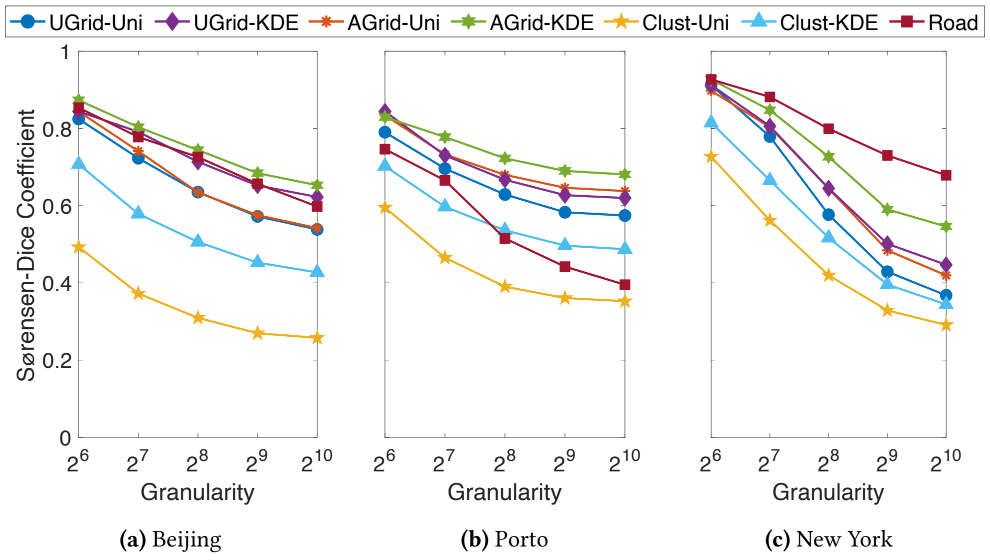


Figure 3.11: Hotspot granularity vs. SDC

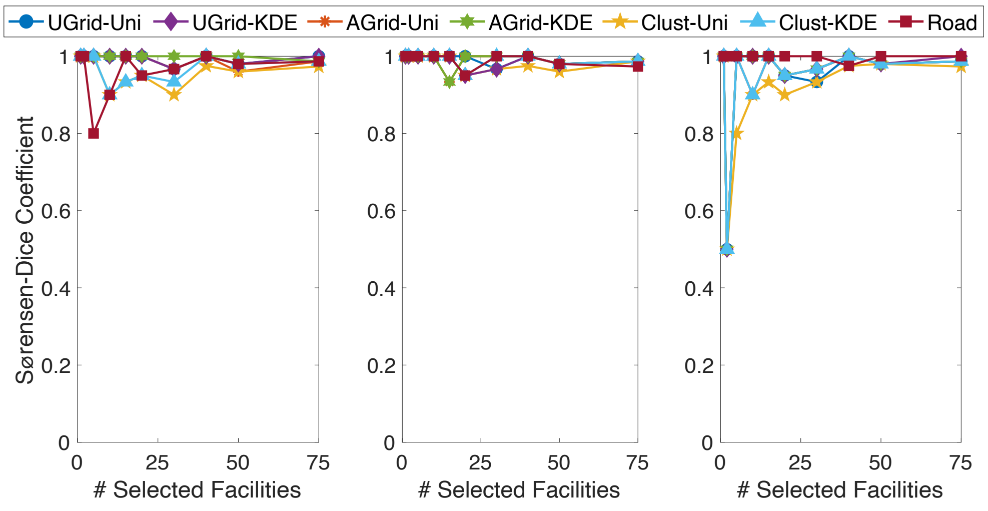
We use a Gaussian kernel over a $g_h \times g_h$ uniform grid, where g_h denotes the hotspot grid granularity; we use granularities: $g_h = \{2^6, 2^7, 2^8, 2^9, 2^{10}\}$. Note that the kernel function can be non-private (i.e., the kernel is tuned to the data) here as we are simply assessing the utility of the output data. Hotspots are defined to be grid cells in which the density is greater than the 95th percentile. $\mathcal{H}^{\text{real}}$ and $\mathcal{H}^{\text{synth}}$ are the set of hotspots obtained using the real and synthetic data, respectively To assess query response similarity between the two datasets, we use the Sørensen-Dice coefficient (SDC), defined as:

$$SDC = \frac{2\left|\mathcal{H}^{\text{real}} \bigcap \mathcal{H}^{\text{synth}}\right|}{\left|\mathcal{H}^{\text{real}}\right| + \left|\mathcal{H}^{\text{synth}}\right|} \tag{3.19}$$

Figure 3.11 shows that similarity decreases as granularity increases. This is because the kernel density estimates are more sensitive to small changes in the location of individual points. All partitioning-based methods outperform their respective baselines, and Road performs especially well when the original data is well-aligned with the road network (e.g., New York, Figure 3.11c). However, Road performs less well with dense road networks or poorly aligned data (e.g., Porto, Figure 3.11b). Conversely, grid-based methods perform better in less-structured environments, but perform worse when data is well-aligned with the road network.

3.4.4 Facility Location Queries

Facility location is a common analytics task for which individual location data is necessary, and it is one possible application for this work. Facility location queries are more complex as they are (essentially) a combination of range and hotspot queries. Given a set $\mathcal{F}$ of candidate facilities, a facility location query aims to find the best k locations that satisfy a stated objective function. The two most common facility location queries are the Max-Inf and Min-Dist queries. In the Max-Inf case, we seek to identify the most influential candidate facilities, where influence is commonly defined as the total number of customers that the



(a) Beijing **(b)** Porto **(c)** New York

Figure 3.12: Number of selected facilities vs. SDC for the MAX-INF query

facilities attract. In the MIN-DIST case, we find the facilities that minimise the total distance between customers and their nearest facility.

Outline

Consider the case where a food stand company wishes to locate a number of outlets in the centre of Beijing. We intuit that a lot of business would be generated if the outlets were located at the intersections of busy roads, and so we use the same location set as the range queries (as this was a selection of road intersections). It is assumed that there are no existing facilities currently in the city. $\mathcal{F}^{\text{real}}$ and $\mathcal{F}^{\text{synth}}$ denote the sets of selected facilities when the real and synthetic datasets are used, respectively. We consider a range of values for k: $k = \{1, 2, 5, 10, 15, 20, 30, 40, 50, 75\}$. The SDC is used to assess accuracy, as it will capture the extent to which synthetic data identifies the same top-k facilities as the real data. $\mathcal{F}^{\text{real}}$ and $\mathcal{F}^{\text{synth}}$ replace $\mathcal{H}^{\text{real}}$ and $\mathcal{H}^{\text{synth}}$ in Equation 3.19.

Results

Figure 3.12 shows how SDC values change with k for the MAX-INF query. Irrespective of the data generation method or city, the query is answered almost identically for all values of k compared to when the real data is used. For example, in Porto, at least 90% of the 'true' top-k facilities are selected when using synthetic data, which further highlights its suitability for use with facility location. Remarkably, when the MIN-DIST query is applied, SDC = 1 for all cities and values of k, and so this plot is omitted. This strong performance is because the optimal locations can tolerate the noise that is required in achieving DP.

However, there may be some cases in which using synthetic data does not obtain similar results for facility location queries. For example, when candidate facilities are close to each other, customers may be assigned to different facilities if their location is perturbed a little. Another example is in the capacitated facility location problem (when capacity constraints are strict) when 'additional' customers generated through additive noise cannot be accommodated at their nearest facility. However, overall, the proposed methods generate synthetic data that exhibits high levels of accuracy for facility location queries compared to using real data. In practice, this means that researchers and companies do not need to use real data for facility location. Instead, private synthetic data can be used without compromising on the accuracy of the facility location analysis.

3.5 Discussion

While both partitioning-based and road network-based approaches are effective in practice, different methods are more appropriate for different circumstances. These circumstances, and some accompanying recommendations, are summarised here.

All methods scale well in accuracy terms. In particular, Road accommodates large datasets easily, and the error decreases with input size. Hence, Road should be the default data generation method, especially when the raw data is well-aligned with the road network. Where road network data is unavailable, or the data is poorly aligned with the road network, partitioning-based approaches should be considered. UGrid-KDE and AGrid-KDE are generally comparable, although AGrid methods are particularly strong in more structured environments. For very large datasets, the difference in runtime costs between clustering- and grid-based methods is larger (cf. Equation 3.1 and Figure 3.6d – N and ϵ have similar effects on runtime), and so clustering-based methods should be considered in this case. Alternatively, merging regions based on a minimum count or area criterion can speed up computation. However, this involves an accuracy trade-off as the resultant regions would be larger, which means the kernel function would be flatter.

For facility location tasks, all methods perform well, and all methods can be recommended as a general purpose solution. For range queries, all methods are highly effective, especially when the range query radius is small. If the range query radius is large, UGrid approaches are recommended (with consideration paid to the degree of network alignment). For hotspot queries, using Road is advised for datasets that are well-aligned with the road network, which is the case for most applications. UGrid-KDE and AGrid-KDE are more effective when the datasets are less well-aligned, or when the road network is less well-structured.

Chapter 4

Generating Synthetic Location Data Using GANs and Label Local Differential Privacy

The data generation methods introduced in the previous chapter rely on centralised DP, which is often undesirable as it requires users to share all of their true data with an aggregator who must be assumed to be trusted. In this chapter, we develop GeoPointGAN – a machine learning-based solution for spatial data synthesis that uses a variant of LDP to offer a stronger privacy guarantee. As with the previous work, the aim is to generate synthetic spatial point data that achieves high fidelity with the original data, whilst also performing well in several location analytics tasks.

Although it may be attractive to use the traditional form of LDP as the basis for local data synthesis, when dealing with spatial data, this can be unnecessarily restrictive in terms of how sensitive data is treated. In particular, LDP normally adopts an "all-or-nothing" approach in which all data needs to be perturbed [161], which affects the utility of the synthetic data for common location analytics tasks. Label privacy [50] provides a more practical means for achieving the necessary privacy protection without sacrificing utility. It is based on the notion that the features of a point are public (and so do not need to be perturbed), whereas the label associated with the data is private (and so does need to be perturbed). Applied to the setting of spatial data, label privacy leads to the following idea: as all location information is public knowledge, it is only a person's *association* with a particular location point at a particular time that is private and in need of perturbation. From this, label privacy can be combined with LDP to formalise label-LDP to provide sufficient privacy protections when generating synthetic spatial data.

In theory, GANs offer a general purpose solution for the data generation problem due to their objective of learning the optimal functional mapping from some random noise input to a faithful representation of real data [104]. As they are a generative model, they can generate datasets of any size, which gives the end user more flexibility. Furthermore, as

GANs operate with data with 'real' and 'fake' labels, they present an intuitive setting for implementing label-LDP. GeoPointGAN incorporates label-LDP through a randomised response mechanism that flips the labels provided to the discriminator, thereby providing plausible deniability to each individual's *association* with a location. Beyond its privatisation properties, label flipping also has potential generalisation and regularisation effects on the model performance, which can be contextualised with related literature.

This chapter proceeds as follows. Section 4.1 presents necessary background information on GANs, followed by the definition and motivation for label-LDP in Section 4.2. In Section 4.3, we detail GeoPointGAN's novel architecture, alongside details on its training process and a theoretical analysis of the privacy mechanism. GeoPointGAN is evaluated in Section 4.4, before the chapter concludes in Section 4.5 with a discussion on the achieved level of privacy, and possible extensions.

4.1 GANs: A Summary

GANs seek to learn the data generating process of observed data $\mathbf{x} \sim p_{\mathcal{D}}(\mathbf{x})$ [104]. Learning is facilitated through two networks: a generator G, and a discriminator D. The generator $G(\mathbf{z}, \Theta_G)$, with parameters Θ_G, takes some random noise $\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})$ and maps it to the feature space of the real data $\mathbf{x}$: $G(\mathbf{z}) = \mathbf{x}$. The discriminator $D([\mathbf{x}, \widehat{\mathbf{x}}], \Theta_D)$, parameterised by Θ_D, then attempts to distinguish real data $\mathbf{x}$ from synthetic samples $\widehat{\mathbf{x}}$. That is, it assigns each data point with a 'real' or 'fake' label, denoted with 1 and 0, respectively. The learning process follows from a min-max game between G and D, which is given as $V(D, G)$:

$$\min_G \max_D V(D,G) = \mathbb{E}_{\mathbf{x}\sim p_{\mathcal{D}}(\mathbf{x})}\left[\log D(\mathbf{x})\right] + \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}(\mathbf{z})}\left[\log(1 - D(G(\mathbf{z})))\right] \tag{4.1}$$

In this chapter, the input feature vector $\mathbf{x} \in \mathbb{R}^{\{2,3\}}$ represents the spatial co-ordinates of a point in two- or three-dimensional space.

GANs have been utilised for a range of data types, including image data [104], audio streams [8], text data [51], traffic patterns [276], and gene expressions [79]. In the geo-spatial domain, GANs have been used for generating digital elevation maps [140], and global surface temperatures [142]. Existing GAN architectures for point data, mostly stemming from computer vision research, mostly deal with point clouds that are simplified, continuous representations of shapes and surfaces. The first GAN tailored to point clouds (r-GAN) [5] builds on advances in processing point clouds in neural networks, most notably PointNet [194]. Generating faithful shapes based on point cloud datasets, such as ShapeNet [46], is an active research challenge [e.g., 95, 149, 204]. Further studies have utilised GANs for point cloud upsampling [152], shape completion [201], or protection against adversarial attacks [278]. Their applications to real-world data have been limited thus far, with a few recent exceptions, such as an application to Lidar data [39].

Spatial point patterns, such as location data from mobile devices, have different characteristics that mean that their point patterns are very different from the point clouds that describe shapes and meshes. They typically have a noisy, multi-scale partitioned structure, they may cover the whole observational area (as opposed to having clear outlines), and they may be governed by underlying dynamics, such as self-excitement. For example, while the point cloud of a chair can roughly be segmented into six elements (i.e., four legs, seat, and back), spatial point patterns in the real world can consist of hundreds of intricate macroscopic (e.g., terrain, cities) and microscopic (e.g., roads, junctions) elements. This means that existing approaches are not optimised to handle the complex spatial patterns observed in the real world. Few studies have tackled this class of data using GANs: Xiao et al. [250] use Wasserstein GANs to learn temporal (one-dimensional) point processes, while Klemmer et al. [141] learn conditional GANs contextualised by the co-ordinates of continuous spatial point data. However, these works provide no intuition for point transformations or for producing new spatial point patterns similar to the input. The challenging nature of generating spatial point patterns and the lack of existing work addressing this problem helps to motivate this chapter.

In recent years, there has been an increasing amount of research into private GANs, surveyed generally by Cai et al. [40]. This research includes several differentially private GANs, such as DPGAN [252], DP-CGAN [220], PATE-GAN [267], and the work of Frigerio et al. [94], which extends DPGAN to continuous, discrete, and time series data. Existing private GANs have focused on other specific domains, such as medical [25, 219, 267], image [220], or time series data [239], as opposed to spatial data. Furthermore, all existing private GANs use centralised DP, normally by clipping and adding noise to the gradient during training [e.g., 94, 252] or by applying existing private frameworks [e.g., 19, 267]. This different privacy setting means we cannot directly compare them to this work.

4.2 Label Local Differential Privacy

Label differential privacy was formally introduced by Chaudhuri and Hsu [50] and has since been the focus of several studies [90, 103, 161, 231, 270]. All of these works are based on the same premise as this chapter: only the labels attached to data are sensitive, with the data itself being non-sensitive. However, almost all prior work has been in the centralised setting; only Busa-Fekete et al. [38] consider the local setting. The notion of LDP is extended to label-LDP if each feature vector $\mathbf{x}_i$ has a label $l_i \in \mathcal{L}$, together denoted as $(\mathbf{x}_i, l_i)$.

Definition 3 (ϵ-label local differential privacy)**.** *A randomised mechanism $\mathcal{M}$ satisfies ϵ-label local differential privacy if, for any labelled feature vector $(\mathbf{x}, l)$ with the input labels $l_i, l_j \in \mathcal{L}$ and output label $l_k \in \mathcal{L}$:*

$$\frac{\Pr[\mathcal{M}((\mathbf{x}, l_i)) = (\mathbf{x}, l_k)]}{\Pr[\mathcal{M}((\mathbf{x}, l_j)) = (\mathbf{x}, l_k)]} \leq e^{\epsilon} \tag{4.2}$$

It is intuitive to observe that label-LDP possesses the same post-processing property as traditional LDP, and that randomised response can be used to ensure label-LDP.

Label-LDP provides more practical, yet sufficiently private, protection to data by only perturbing the label attached to a feature vector. This is appropriate for this problem given that we deem information about locations in a region to be sufficiently public, with only one's *association* with a location being private information. From Definition 3, the intuition is that an adversary cannot (with high confidence) identify whether the person was at the reported location or not.

In using label-LDP, we assume that the real data covers a sufficiently large proportion of the domain in which fake points can be generated. This is to prevent the labels from being sufficiently correlated with features, which would undermine privacy [38]. For example, if the ratio between land area and total area is too small, many fake points would be generated in nonsensical locations (e.g., oceans), which would allow an adversary to identify fake points and, by extension, real points. While this assumption does not affect the mechanism, this soft constraint is imposed as an extra layer of protection against privacy leakage.

Finally, although one could naïvely apply randomised response directly to the real data, this limits the dataset size to approximately $\frac{Ne^{\epsilon}}{e^{\epsilon}+1}$, which heavily limits the range of analytics tasks for which the private data can be used. Hence, a more flexible solution that can generate datasets of any size is necessary.

Examples

To further illustrate the use of label-LDP in this setting, and its advantage over traditional LDP, consider the following two example scenarios. In both examples, although having the original locations with perturbed labels is useful in itself, the samples may not be representative. Even if the samples are representative, the labels attached to the locations will be noisy, owing to perturbation, which may be undesirable As such, being able to generate datasets of any size based on the original distribution is important, and gives end users more flexibility.

For the first example, consider that a city's government wants to know the distribution of its residents' locations at 10am. This might help to determine the approximate proportion of people working from home, which is helpful for managing working conditions or reducing disease spread. As the government will know every resident's address (e.g., through the electoral register or council tax information), this information is non-private for these purposes. The private element is where each person is at 10am, and residents will want a degree of plausible deniability, which is provided by label-LDP. In comparison, traditional LDP is unnecessarily restrictive here as it requires the perturbation of the location of each resident at 10am, even if their home address is known.

Moreover, if this same query was asked every day, traditional LDP-based solutions that rely on location perturbation would eventually risk revealing the true location. While continuous data sharing is a common need for many real-life applications, traditional LDP approaches do not address this most practical setting. Whereas, with label-LDP, as the location itself is deemed to be non-private, each daily report is essentially independent, and so repeated querying does not degrade the overall privacy level provided to each user. This further motivates label-LDP as a more robust privacy model for this problem.

The second example differs from the first as the government no longer has a plausible location for each individual for which they want to ask a yes-no question. Imagine a town with 100,000 people, all of whom are asked to privately share their location at 8pm, which is also the time at which a large number of residents are attending a concert. With traditional LDP, the locations of each individual would need to be perturbed, which would induce a large amount of noise into the dataset and greatly affect utility. However, public knowledge (e.g., news sources) would show that a large number of people were at the concert, and so we should expect a similarly large number of reported locations to be associated with the concert. As such, label-LDP still gives each concert-goer a degree of plausible deniability regarding their presence at the concert, while preserving the overall popularity distribution, which is publicly known (to some extent).

4.3 GeoPointGAN

This section presents GeoPointGAN's architecture, details for training GeoPointGAN, as well as a discussion on the privacy achieved by GeoPointGAN.

4.3.1 Model Architecture

GeoPointGAN includes several novel approaches to address the previously identified challenges of generating spatial point data using GANs. Its architecture is outlined in Figure 4.1.

Generator

Traditional GANs generate points by first sampling a Gaussian noise vector **z** from a lower-dimensional latent space. This noise is then 'upsampled' from this latent space to a higher-dimensional output space to generate synthetic points. GeoPointGAN instead samples a noise vector of the *same* dimensionality as the desired output. For example, when generating two-dimensional data, GeoPointGAN aims to learn a model that transforms data from a two-dimensional latent space to a meaningful representation in the same two-dimensional space. This is done by deploying a novel PointNet-based generator for point transformations during data generation.

PointNet [194] was originally devised for classifying and segmenting raw point clouds. While it has been used as the basis for GAN discriminators before [5], GeoPointGAN is the first GAN to utilise PointNet in the generator. In particular, PointNet's ability to provide transformation invariant properties for unordered data is desirable for spatial point generation. This is achieved by running the input data through symmetric functions (e.g., max pooling operators) to compute global point set features. This step is followed by a segmentation network that combines global information (e.g., city boundaries, rivers) with local, point-wise information (e.g., roads, junctions) to learn a combined representation. Lastly, a spatial transformer network (STN) [126] aligns the learned global and local point set features with the output space. However, the traditional STN architecture is unable to resolve the complexities of spatial point datasets sufficiently, owing to the shallowness of the neural networks deployed. In particular, while macroscopic structures (e.g., coastlines) can be captured reliably, the STN is incapable at learning small-scale patterns (e.g., minor roads). Consequently, we extend the STN such that the generator has five one-dimensional convolutional layers, with four fully connected layers. Batch normalisation is added between every layer and the ReLu function, except for the last layer. Altogether, this altered PointNet can be referred to as 'LargePointNet'.

As a final step, the generator takes the transformation invariant, aligned features of LargePointNet and projects them into the output space using four fully connected layers. This is similar to the prediction head for point segmentation, only that GeoPointGAN produces

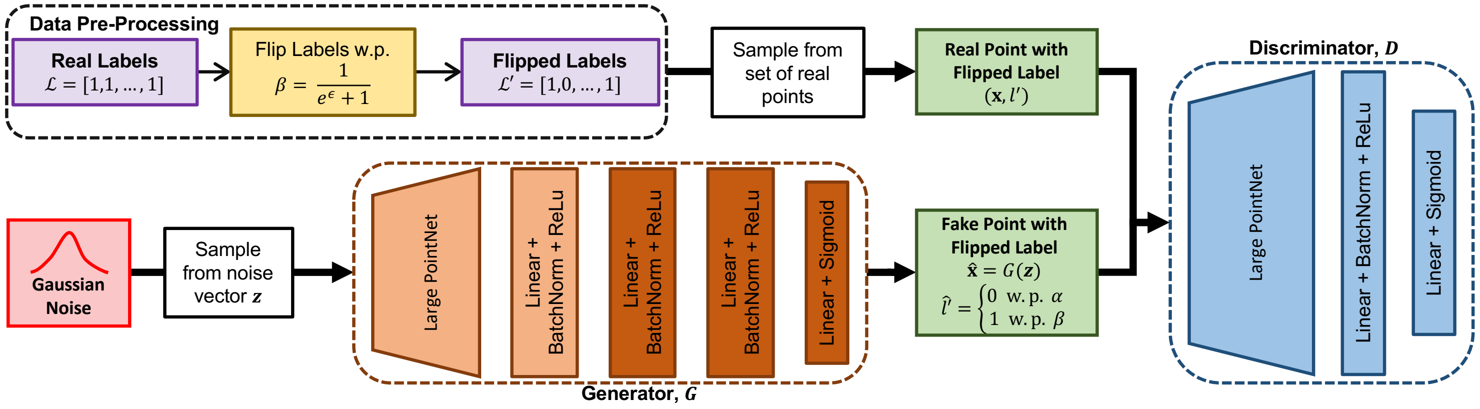

Data Pre-Processing
Real Labels
$\mathcal{L} = [1,1,\dots,1]$
Flip Labels w.p.
$\beta = \frac{1}{e^{\epsilon}+1}$
Flipped Labels
$\mathcal{L}' = [1,0,\dots,1]$
Sample from set of real points
Real Point with Flipped Label
$(\mathbf{x}, l')$
Gaussian Noise
Sample from noise vector $\boldsymbol{z}$
Large PointNet
Linear + BatchNorm + ReLu
Linear + BatchNorm + ReLu
Linear + BatchNorm + ReLu
Linear + Sigmoid
Generator, G
Fake Point with Flipped Label
$\hat{\mathbf{x}} = G(\boldsymbol{z})$
$\hat{l}' = \begin{cases} 0 & \text{w.p. } \alpha \\ 1 & \text{w.p. } \beta \end{cases}$
Discriminator, D
Large PointNet
Linear + BatchNorm + ReLu
Linear + Sigmoid

Figure 4.1: GeoPointGAN's architecture, including the label-LDP mechanism

two (or three) features per point (i.e., the synthetic co-ordinates), rather than one. These design choices are informed by extensive testing and are validated during evaluation.

Discriminator

GeoPointGAN's discriminator architecture is inspired by Achlioptas et al. [5], but comes with some fundamental technical improvements, and a critical change that allows label-LDP to be incorporated. First, points are run through the LargePointNet module. This balances the capacity for learning point set representations between the generator and discriminator, allowing for an evenly matched min-max game. The discriminator's prediction head consists of two fully connected layers and a sigmoid activation. Second, the last fully connected layer is altered to produce predictions on the point level, as opposed to the batch level. That is, the discriminator's task is to determine whether each *individual* point is real or fake, as opposed to each batch of points. Constructing the discriminator in this way allows a localised privacy mechanism to be incorporated into model training seamlessly.

Privacy Mechanism

Point-level privacy guarantees are incorporated into GeoPointGAN by probabilistically flipping the labels of the real and fake points (using randomised response) before showing the data samples to the discriminator. Traditional LDP would require each co-ordinate to be perturbed (using, say, the Laplace or exponential mechanism). With label-LDP, however, only perturbing the label is sufficient. As there are two (pseudo-)labels – 'real' and 'fake', flipping can be conducted using biased coin tosses in which the probability that a label's true status is maintained is $\alpha = \frac{e^{\epsilon}}{e^{\epsilon}+1}$, and the probability that a label's status is flipped is $\beta = 1 - \alpha = \frac{1}{e^{\epsilon}+1}$ (see Section 2.1.3). The labels of real points are flipped on users' devices during data pre-processing, which ensures that the central agent (i.e., the GAN networks) never has access to this information. If a real point is sampled several times throughout training, it will always have the same (flipped) label. From Definition 3, the intuition follows that the discriminator cannot determine (with high probability) that a point with a real label is actually real, or whether it is a fake point masquerading as a real one (or vice versa).

In a practical setting, all training is conducted by a central agent (who can be trusted or untrusted) on a remote server. Individual data is collected through mobile devices, and each individual is responsible for flipping the label associated with their location. Hence, the only data transferred from the user's device is the location and the flipped label, which means that no central agent can definitively determine the true label with absolute certainty. Importantly, the discriminator does not know which points are generated by the generator, and which points are transmitted to the server by users; it can only distinguish points based on their perturbed labels. In summary, the discriminator has no way of (definitively) knowing whether any one point is real with a real label, real with a fake label, fake with a real label, or fake with a fake label.

Effects on Training and Generalisation

This label flipping approach does not necessarily reduce model performance, but can even have beneficial effects. In predictive models, randomly flipping labels can act as a regulariser, which prevents the model from overfitting and improves generalisation [251]. When working with noisy labels, label flipping can incorporate the uncertainty of the labels into the model [180]. There is a vast collection of literature that focuses on GAN regularisation and robustness, and addresses related issues, such as limited data availability and generator-discriminator imbalance. Manipulating the (pseudo)-labels of GANs has proven to be a successful strategy to this end. Specifically, adding noise to the labels or applying one-sided label smoothing have been shown to improve GAN training, and these are common best practices [200]. Jiang et al. [129] propose to feed the discriminator with fake data masquerading as real data (i.e., fake data with real labels). While this is proposed mainly as an augmentation strategy for sparse data environments, it is very similar to GeoPointGAN's label flipping approach, although they do not feed the discriminator real data with fake labels. The authors also provide a theoretical intuition for training convergence and their approach. Their proof highlights how a GAN trained with label flipping augmentation minimises the Jensen-Shannon divergence between the (smoothed) real and synthetic data distributions and is, in theory, able to perfectly capture the data generating process. Hence, it is likely that a private GeoPointGAN will (to a certain extent) perform as well as a non-private GeoPointGAN (i.e., one with no label flipping).

4.3.2 Model Training

Algorithm 1 describes the training of GeoPointGAN. Lines 1–2 are conducted on user devices, although they are included here for completeness. Before training, each real point is assigned the 'real' label: $l_i = 1$ (Line 1). These labels are then flipped with probability β, where β is controlled by the privacy budget (Line 2). Perturbed labels are denoted as l_i'. We then initiate the training loop (Line 3). At each training step, B points are sampled from the real data *without* replacement (Line 4). This avoids oversampling points from high-density areas. B random points are also drawn from the noise prior $p_\mathbf{z}$ (Line 5) and transformed in G to generate B fake points: $\widehat{\mathbf{x}} = G(\mathbf{z})$. The label of each fake point, $\widehat{l_i} = 0$, is flipped with probability β to obtain $\widehat{l_i'}$ (Line 6). Real points $(\mathbf{x}, l')$ and fake points $(\widehat{\mathbf{x}}, \widehat{l'})$ are then classified as real or fake by D, after which D is updated using the optimiser (Line 7). We then train G by generating B new fake points (Line 8), flipping their labels with probability β (Line 9), and once more classifying them using D. G is then updated using its optimiser (Lines 10). This concludes one training step. D and G continuously play this game, with G getting better and better at generating synthetic data. After N_{steps} training steps, the label-LDP generator is published (Line 11).

4.3.3 Privacy Analysis

We now discuss some aspects of the mechanism from a privacy perspective. First, we show the proposed label flipping approach satisfies ϵ-label-LDP.

Theorem 7. *GeoPointGAN satisfies ϵ-label-LDP.*

Proof. For ease of understanding, in this proof, points with real and fake labels are denoted as $\mathbf{x}^1$ and $\mathbf{x}^0$, respectively The probability that a real label tells the discriminator that it is a real label is: $\Pr[\mathcal{M}(\mathbf{x}^1) = \mathbf{x}^1] = \alpha = \frac{e^\epsilon}{e^\epsilon+1}$. Hence, the probability that a real label tells the discriminator that it is a fake label is $1 - \alpha$. That is, $\Pr[\mathcal{M}(\mathbf{x}^1) = \mathbf{x}^0] = \beta = \frac{1}{e^\epsilon+1}$. Similarly,

Algorithm 1 GeoPointGAN training

```
Require: $\mathcal{D}, \epsilon, B, N_{steps}$
1:  Assign real points real labels: $\{\mathbf{x}_1, ..., \mathbf{x}_N\} \rightarrow \{(\mathbf{x}_1, l_1), ..., (\mathbf{x}_N, l_N)\}$ where $l_i = 1$
2:  Flip real labels with probability $\beta = \frac{1}{e^\epsilon+1}$ to obtain $\{(\mathbf{x}_1, l'_1), ..., (\mathbf{x}_N, l'_N)\}$
3:  for 1 to $N_{steps}$ do
4:      Sample $B$ real points with flipped labels: $\{(\mathbf{x}_1, l'_1), ..., (\mathbf{x}_B, l'_B)\}$
5:      Sample $B$ random co-ordinates $\{\mathbf{z}_1, ..., \mathbf{z}_B\}$ from noise prior $p_\mathbf{z}$
6:      Flip fake point labels with probability $\beta$ to obtain $\{\widehat{l'_1}, ..., \widehat{l'_B}\}$
7:      Update $D$ by ascending its stochastic gradient: $\nabla_{\Theta_D} \frac{1}{B} \sum_{i=1}^{B} \left[ \log(D(\mathbf{x}_i, l'_i)) + \log(1 - D(G(\mathbf{z}_i), \widehat{l'_i})) \right]$
8:      Sample $B$ random co-ordinates $\{\mathbf{z}_1, ..., \mathbf{z}_B\}$ from noise prior $p_\mathbf{z}$
9:      Flip fake point labels with probability $\beta$ to obtain $\{\widehat{l'_1}, ..., \widehat{l'_B}\}$
10:     Update $G$ by ascending its stochastic gradient: $\nabla_{\Theta_G} \frac{1}{B} \sum_{i=1}^{B} \left[ \log(D(G(\mathbf{z}_i), \widehat{l'_i}) \right]$
11: return $G$
```

$\Pr[\mathcal{M}(\mathbf{x}^0) = \mathbf{x}^0] = \alpha$ and $\Pr[\mathcal{M}(\mathbf{x}^0) = \mathbf{x}^1] = \beta$. From Equation 4.2, we have:

$$\frac{\Pr[\mathcal{M}(\mathbf{x}^1) = \mathbf{x}^1]}{\Pr[\mathcal{M}(\mathbf{x}^0) = \mathbf{x}^1]} = \frac{\alpha}{\beta} = \frac{e^\epsilon}{e^\epsilon + 1} / \frac{1}{e^\epsilon + 1} = e^\epsilon$$

■

As discussed in Section 4.2, label-LDP has the same post-processing properties as LDP, which means that privatised data can be manipulated freely without affecting the privacy guarantee (as long as the true data is not 'touched' again). In GeoPointGAN, the labels are perturbed by users before they are shown to the discriminator, and the true label is never used again. This means that the entire training procedure operates under post-processing, and the privacy guarantee remains intact throughout training.

Many DP mechanisms suffer when points are repeatedly sampled, which causes the privacy leakage to increase each time a point is sampled. GeoPointGAN is designed such that these attacks are redundant as point labels are flipped once, and once only, before training begins. That is, when $\mathbf{x}_i$ is sampled during any training step, it will always have the same perturbed label l'_i. This means that there is no privacy leakage even if a point is sampled more than once during different training steps. Note that, as points are sampled *without* replacement, the same point cannot be sampled multiple times during a single training step.

Threat Model

In this chapter, the aim of an adversary is to *identify a certain individual's association with a specific location.* We assume that the adversary has access to the database of all possible locations, and that they use this to try to reassociate an individual with their location. By definition, label-LDP provides sufficient protection against such attack, as each individual has a degree of plausible deniability with respect to their association with any location. In our model, this protection is provided when the label associated with each true point is perturbed using randomised response. An equal number of fake points are also generated, and their labels are perturbed similarly. This protects against frequency-based attacks based on the aggregate number of real and fake points in the output data. The fundamental structure of GeoPointGAN provides further protection as points are entirely dissociated from users when they are transmitted to the discriminator. As a consequence, with the published data being the output from a generative model, no location in the output data can (with a high degree of certainty) be associated with a location, or individual, in the sensitive input data.

4.4 Evaluation

We evaluate GeoPointGAN in two parts using four real spatial datasets. The first part evaluates GeoPointGAN against three alternative GAN-based approaches, before we study the effects that privatisation has on GeoPointGAN. The second part evaluates GeoPointGAN's practical query-based performance using the same location analytics tasks from Chapter 3.

4.4.1 Set-Up

Data. The same three datasets used in Chapter 3 are used in this evaluation, with no changes to the pre-processing. A fourth dataset – '3D Road' – provides three-dimensional spatial co-ordinates (latitude, longitude, and altitude) of the road network in Jutland, Denmark [134]. The dataset comprises over 430,000 points, covering an area of 185×135 km^2.

Training Setting. We follow a standardised training process. At each training step, 7,500 points are randomly sampled from the real dataset. We use the Adam optimiser with decoupled weight decay [157] and an initial learning rate of 4×10^{-5}. The learning rate is decreased by a factor of 10 after 5,000, 50,000, and 90,000 training steps. We train 1,000 steps per epoch for a total of 100 epochs. All training was conducted on a single RTX 2080 GPU. With this set-up, model training times do not exceed two hours. Overall, GeoPointGAN, like r-GAN, experiences training that is reliable and consistent. At no point during any of the training runs do we experience mode collapse or exploding gradients.

Benchmarks. We compare GeoPointGAN against three state-of-the-art GANs. The first – r-GAN [5] – is the method that is most closely related to GeoPointGAN and it is designed to operate on raw point clouds. The other two baselines, Tree-GAN [204] and PCGAN [15], are designed for graph-structured point clouds (e.g., meshes, shapes). All baselines are trained according to the configuration outlined by the original authors.

Despite the development of other private GANs (e.g., DPGAN, PATEGAN), these works all use the centralised DP, which is fundamentally different from our label-LDP setting. Similarly, although other private methods for synthetic spatial data generation exist, these methods also use different forms of privacy. For example, Chen et al. [55] use personalised LDP, and the work in Chapter 3 uses centralised DP. As our privacy setting is different from all of these works, any comparison between them is meaningless.

Evaluation Measures. To evaluate the extent to which GeoPointGAN preserves the underlying distribution of the real data, we use two widely used utility measures: Chamfer distance and earth mover's distance (EMD). The same definition for Chamfer distance used in Chapter 3 is used here.

EMD – a common measure for evaluating GANs – can be viewed as an optimisation problem that seeks to transform one probability distribution into another while minimising the cost of this operation. While the computational cost of obtaining the exact distance is too high for it

Table 4.1: Mean Chamfer and earth mover's distance values

Method	Chamfer Distance (CD)				Earth Mover's Distance (EMD)			
	Beijing	Porto	NYC	3D Road	Beijing	Porto	NYC	3D Road
Tree-GAN [204]	0.651	0.437	0.649	1.247	0.733	1.085	0.601	1.080
PCGAN [15]	0.092	0.348	0.160	0.796	0.283	0.831	0.305	0.896
r-GAN [5]	0.032	0.034	0.226	0.264	0.084	0.275	0.364	0.526
GeoPointGAN	**0.021**	**0.019**	**0.014**	**0.074**	**0.032**	**0.027**	**0.031**	**0.085**

to be used in deep learning algorithms (and hence approximations are used), its exact version can be used as an evaluation measure (especially when sample sizes are small). Defining $\phi : \mathcal{P} \rightarrow \mathcal{S}$ as a bijection, EMD is defined as:

$$EMD(\mathcal{P}, \mathcal{S}) = \min_{\phi:\mathcal{P}\rightarrow\mathcal{S}} \sum_{p\in\mathcal{P}} \|p - \phi(p)\| \tag{4.3}$$

4.4.2 GAN Evaluation

Baseline Comparison

Figure 4.2 depicts plots of the real data, alongside samples from GeoPointGAN and r-GAN. As Tree-GAN and PCGAN performed very poorly, their visualisations are excluded. While r-GAN succeeds at capturing macroscopic structures, such as the outline of Manhattan and Central Park, it lacks the capacity to model more intricate structures within the outline, such as individual streets or junctions. On the other hand, GeoPointGAN is able to reproduce microscopic structures to a reasonable extent.

We calculate mean CD and EMD values by taking 60 samples of 7,500 points each from the fully trained generators. These values are shown in Table 4.1. Note that, here, we use a non-private GeoPointGAN (i.e., $\epsilon = \infty$) to ensure a fair comparison with the baselines, which have no privacy mechanism. The tree-based methods perform particularly poorly as they fail to learn the spatial data distribution. This justifies the decision to use generative models that are capable of operating on raw point co-ordinates, rather than shapes or meshes. GeoPointGAN offers substantial improvements over r-GAN, decreasing CD values for New York by up to 16x, and EMD values by up to 12x. Given that r-GAN is generally the most competitive baseline, GeoPointGAN's superiority over the other baselines is even more pronounced. Overall, these results reflect GeoPointGAN's ability to preserve microscopic features and generate accurate spatial point data.

Effect of Privatisation

To assess how the label-LDP mechanism affects GeoPointGAN performance, we again draw 60 samples of 7,500 points from the real and synthetic datasets. We consider seven privacy budgets: $\epsilon = \{0.1, 0.25, 0.5, 1, 2, 5, 10\}$. As none of the baselines are competitive with GeoPointGAN, and they were not designed with privacy-preserving mechanisms in mind, they are excluded from this part of the evaluation. Figure 4.2 shows the effect of privatisation on the generated data. Lower privacy budgets have a negative impact on visual similarity; this is seen most clearly in the Beijing plots. Figure 4.3 further illustrates how

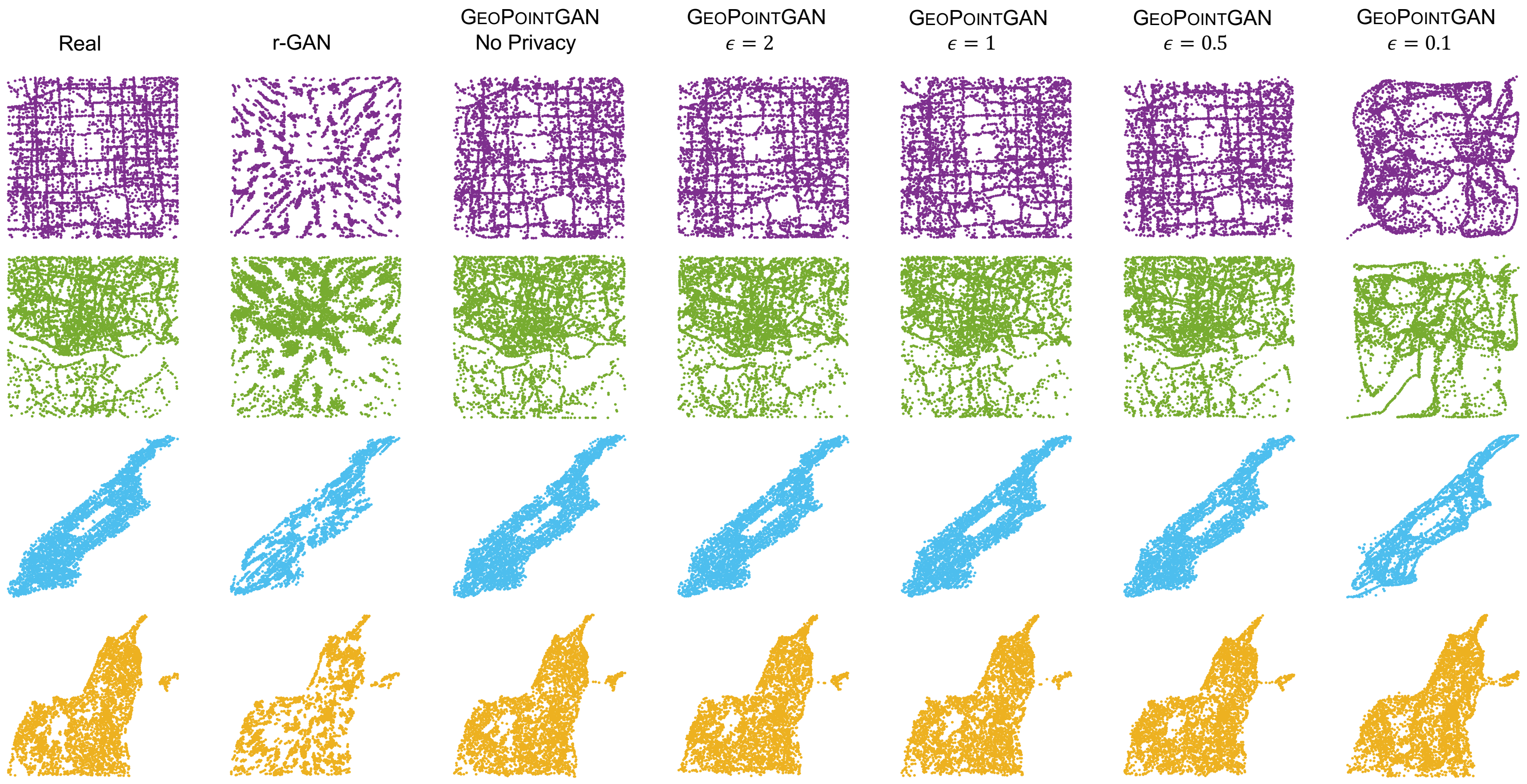


Figure 4.2: Sample plots of real and synthetic data; from top to bottom: Beijing, Porto, New York City, and 3D Road

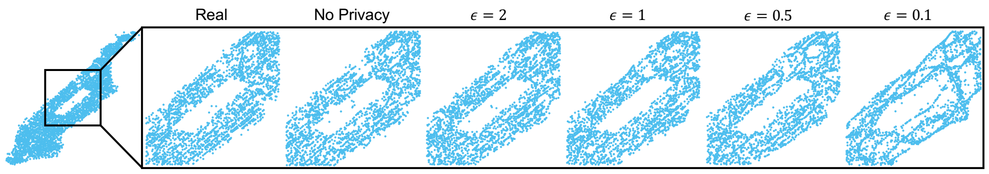


Figure 4.3: Real and synthetic data for different privacy budgets; data for New York City, zoomed in on Central Park

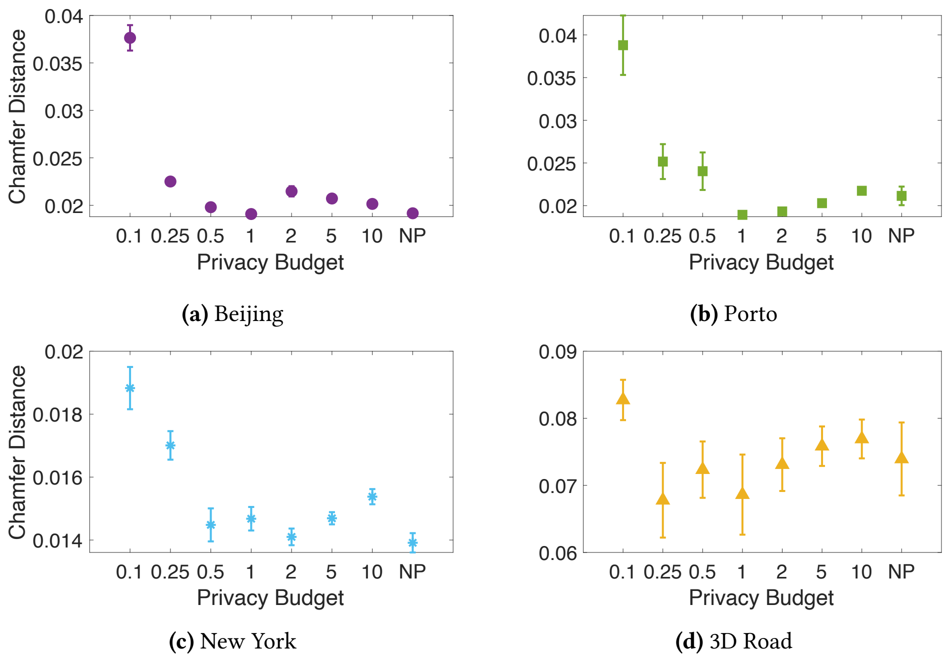
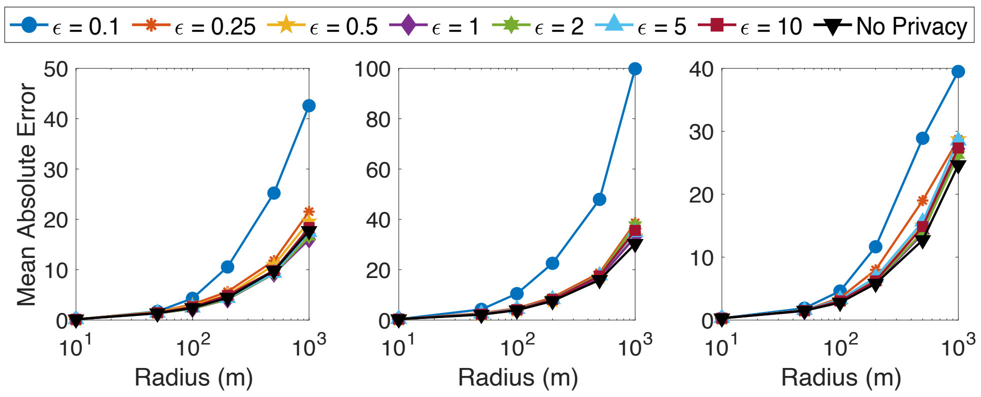


Figure 4.4: Privacy budget vs. Chamfer distance

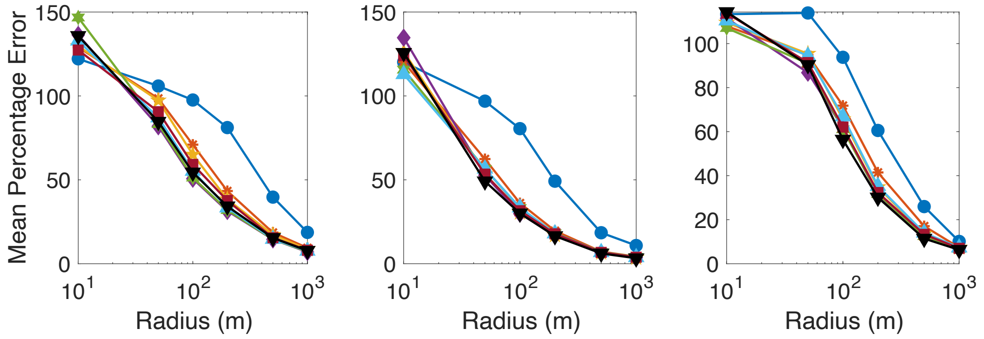
the preservation of macroscopic geographic features (e.g., Central Park in New York City) is affected by changing the privacy budget.

Figure 4.4, which shows the variation in CD as the privacy budget varies, highlights more interesting behaviour. The graphs for EMD exhibit very similar profiles, but they are excluded here for clarity. A very low privacy budget results in poor utility, with utility increasing as ϵ increases. This is to be expected, as it is more likely that a label is flipped when ϵ is low. However, as discussed in Section 4.3.1, GANs have a risk of overfitting, with GeoPointGAN particularly susceptible to this when there is little label flipping (i.e., when ϵ is high). Our empirical results indicate that, when $\epsilon > 1$, the CD values start to increase, leading to (albeit subtle) U-shaped curves. These results suggest that our hypothesis that label flipping can aid performance by generalising and regularising the GeoPointGAN is supported with empirical evidence. In some cases, better utility is gained using a *private* GeoPointGAN (cf. Porto, Figure 4.4b), which further demonstrates the power of regularisation through privatisation.

Interestingly, the structure of the underlying data also appears to influence this behaviour. In New York, where the data is more closely aligned with a strict grid structure, the U-shape is more pronounced, indicating that the regularisation properties of GeoPointGAN are more influential here. Conversely, in Porto, where the true data (and road network) lacks a clear,



(a) MAE – Beijing **(b)** MAE – Porto **(c)** MAE – New York



(d) MPE – Beijing **(e)** MPE – Porto **(f)** MPE – New York

Figure 4.5: Range query radius vs. MAE and MPE

regular structure, the U-shape is more subtle and very high privacy budgets correct the curve downwards. Finally, for 3D Road, CD values are larger (and show larger variations), which suggests that the complexity and spatial extent of the dataset pushes GeoPointGAN to its limits.

4.4.3 Data Analytics Tasks

We now evaluate GeoPointGAN using the three data analytics tasks introduced in Chapter 3. Given the poor qualitative and quantitative performance of the baselines, and their non-private nature, it is not meaningful to compare GeoPointGAN against them for these queries. As such, the aim of this evaluation is to compare GeoPointGAN to the optimum (i.e., maximum similarity with the real data).

Range Queries

To maintain consistency with the evaluation in Section 4.4.2, these range queries are conducted using 60 independent random samples. The mean of the MAE and MPE values is then reported. Otherwise, the settings and evaluation measures remain unchanged from Section 3.4.3. For simplicity, only two-dimensional range queries are used; hence 3D Road is excluded from evaluation.

Figure 4.5 shows the effect that ρ has on the MAE and MPE of the query answer. As expected, MAE increases as ρ increases, although MPE decreases; both of these trends are acceptable

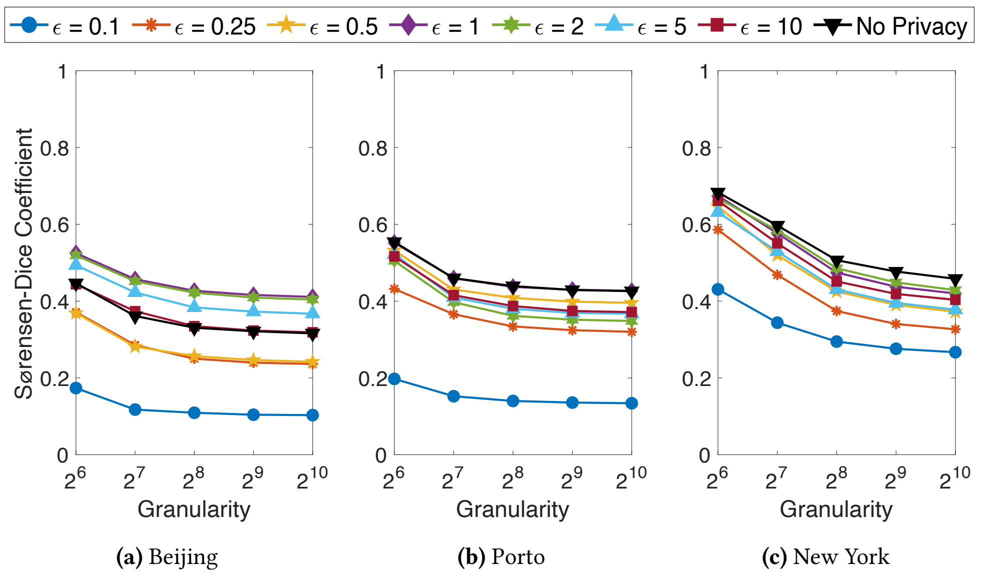


Figure 4.6: Hotspot granularity vs. SDC

when considered together. As a particularly impressive outcome, synthetic data generated with higher privacy budgets (i.e., $\epsilon \geq 0.25$) performs as well, and sometimes better, than the data generated using the non-private version of GeoPointGAN. Close inspection indicates that the privatised GeoPointGAN performs best when $\epsilon \approx 1$, which is concordant with the findings in Section 4.4.2.

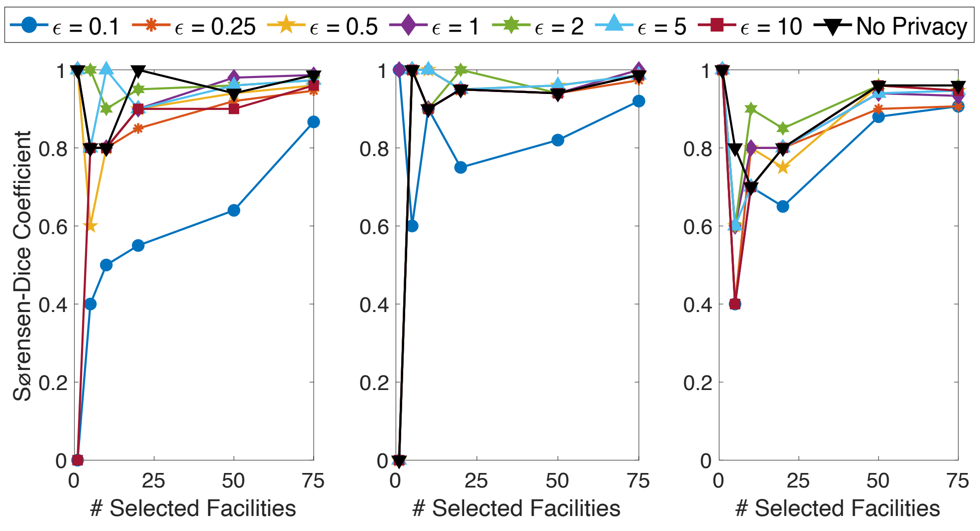
Hotspot Analysis

When conducting hotspot analysis, we take 60 independent samples and report the mean SDC value across these samples. Otherwise, the settings remain unchanged from Section 3.4.3. As the 3D Road dataset only consists of static locations (e.g., the road network), the concept of a 'hotspot' is meaningless, and so it is excluded from evaluation.

Figure 4.6 shows the variation in mean SDC values as the hotspot granularity increases and, once again, we observe similar findings to those discussed previously. Namely, (a) poor performance is observed when $\epsilon = 0.1$, while other ϵ values are competitive with the non-private GeoPointGAN; (b) a private GeoPointGAN sometimes outperforms the non-private version; and (c) a private GeoPointGAN performs best with a middling ϵ value, though the exact value depends on the city.

Facility Location Queries

For the facility location queries, we use the exact same set-up from Section 3.4.4. Figure 4.7 shows the variation in SDC values for the Max-Inf query. GeoPointGAN produces synthetic data that answers queries with high accuracy, and the non-private GeoPointGAN performs exceptionally well in most cases, especially in Porto for which it obtains near-optimal results. The effect of changing the privacy budget is also noticeable, and the phenomenon of privatised versions of GeoPointGAN performing as well as non-private versions persists. As in Chapter 3, when the Min-Dist query is applied, SDC = 1 for all cities and values of



(a) Beijing **(b)** Porto **(c)** New York

Figure 4.7: Number of selected facilities vs. SDC for the Max-Inf query

ϵ and k, and so this plot is omitted. These strong results demonstrate the practicality of GeoPointGAN, and illustrate that facility location can tolerate the noise that is inherent in GAN-based sampling, and required for label-LDP. This robustness can also be exploited for other data science tasks, such as nearest neighbour queries and clustering.

4.5 Discussion

As demonstrated throughout the experiments, GeoPointGAN is a robust method for generating large synthetic spatial datasets with practical levels of privacy and high utility, both statistically and with respect to several location analytics tasks. Indeed, in some settings, a private GeoPointGAN will perform better than non-private versions – a remarkable and important observation. This phenomenon is possible due to the design of the privacy mechanism, which exploits the inherent noise in label flipping to harness the regularisation effects that can be realised when training GANs.

Beyond these findings, this section provides further insights into the level of privacy provided. While GeoPointGAN demonstrably fulfils the requirements of label-LDP, the actual level of privacy provided is higher in many practical settings. As Malek et al. [161] note, simply removing the sensitive labels of a public dataset and training a model in an unsupervised fashion complies with label-(L)DP. GeoPointGAN takes this further by accounting for situations where, even if the identifier (e.g., taxi ID, 311 caller name) is removed, we are still conscious of leaking information based on knowledge regarding the veracity of each point (e.g., if locations can help to identify the caller). Specifically, GeoPointGAN uses pseudo-labels (to denote whether points are real or fake) to obfuscate the training data by removing certainty within the model as to what input represents real data. In this sense, the level of privacy provided is stronger than what is necessarily required with label-LDP, although quantifying this achieved level of privacy is non-trivial. And, while this level of privacy may not necessarily be strong enough to satisfy LDP in theory, it may satisfy LDP to some extent in practice. Further exploration of these ideas and challenges would form an interesting avenue for future work.

Finally, GeoPointGAN is notable in that, unlike the other solutions proposed in this thesis, it does not explicitly incorporate any publicly known geographic knowledge (e.g., coastline or road network data). This is because, in theory, GANs should be able to learn the boundaries of these features. Visual inspection (Figure 4.2) shows that GeoPointGAN can effectively do this and, coupled with strong quantitative performance, this demonstrates that including the external knowledge is unnecessary in achieving high practical utility. Nevertheless, some recent work has incorporated constraints into GANs to benefit utility: Yang et al. [263] use penalty terms in the loss function to model physical constraints (e.g., energy conservation laws), and Wu et al. [247] enforce statistical constraints on GANs to help solve partial differential equations. Meanwhile, Klemmer et al. [142] devise a new metric (SPATE) that measures spatio-temporal autocorrelation, which is then used to compute an embedding loss that considers spatio-temporal interactions and nudges the GAN to learn outputs that align better with the true data. To extend GeoPointGAN to accommodate external knowledge, one could incorporate these ideas to penalise points generated in out-of-bounds regions through the loss function, although this is left for future work.

Chapter 5

Sharing Location Sequences with Local Differential Privacy

This chapter focuses on perturbing trajectories using LDP, which is more complex than generating a set of synthetic spatial points. We model trajectories to be time-ordered sequences of visited POIs, as opposed to raw location traces. Scoping the problem this way is beneficial for achieving the aims of this thesis given that POIs are a rich source of external knowledge as they have several attributes that are public knowledge (e.g., opening hours, category, price point), all of which can be used to enhance utility. A natural way to incorporate this information is through semantic distance functions and the exponential mechanism, such that semantically similar POIs are more likely to be returned by the mechanism. Doing this also addresses the previously identified limitation of existing LDP mechanisms, which assumed equal sensitivity across all data points. But, unlike other distance-based mechanisms [4, 11, 12, 109], incorporating semantic distance functions with the exponential mechanism allows strict ϵ-LDP to be achieved.

The efficiency-utility trade-off discussed in Chapter 1 is even more influential when using the exponential mechanism as it requires the entire output set to be instantiated. From a utility perspective, modelling entire trajectories as singular points in a high-dimensional space appears to be most effective. This is because the spatio-temporal correlations between adjacent points are more likely to be preserved, which is crucial for achieving high utility when perturbing trajectories (also discussed in Chapter 1). However, this 'global' solution attracts a high space and time cost, which makes it unviable, as we shall see. Conversely, perturbing each POI independently is the fastest solution, although this fails to consider any information between adjacent points, thereby reducing utility. These intuitions motivate a more scalable solution that seeks to perturb smaller trajectory fragments, called n-grams, in order to capture the spatio-temporal relationship between adjacent points, while remaining computationally feasible.

This chapter presents this n-gram-based solution, as well as the global solution, and demonstrates how overlapping the n-grams allows more information for each point to be captured, whilst continuing to satisfy LDP. As well as forming the basis for the semantic distance functions, external knowledge can also be utilised to structure the space. In particular, the (publicly known) hierarchies that are inherent in space, time, and category classifications can be used 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 the n-gram-based solution remains scalable for large urban datasets.

The rest of this chapter is organised as follows. Section 5.1 formally introduces the problem, and the key definitions that are integral to the solution. Section 5.2 outlines the idealised 'global' solution, and Section 5.3 details the main n-gram-based solution. This is followed by Section 5.4, which discusses the privacy, theoretical utility, and computational cost of the two mechanisms. Section 5.5 focuses on evaluating the mechanism through an extensive set of experiments. The chapter concludes with Section 5.6, which discusses the generalisability of the mechanism, as well as potential applications of the work.

5.1 Preliminaries

Before either solution can be outlined, it is necessary to detail some preliminary material. This includes a more thorough discussion of the ideal characteristics of the solution, an outline of key notation, and the formal definition of the reachability constraint.

5.1.1 Problem Setting

Imagine a city in which each resident visits a number of POIs each day, which we can link together in a time-ordered sequence to form a trajectory. The city's government wishes to learn aggregate information on where residents are travelling 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. The aim of this chapter, therefore, is to create a mechanism for users to share their trajectories in a privacy-preserving way, whilst ensuring that the shared trajectories preserve as much utility as possible at the aggregate level. Despite the focus on aggregate level utility, it is still important for the data to be published on the trajectory level (i.e., in the same format as the input). This will give end users more flexibility to use the privatised data how they wish. With this is mind, the guiding aims and principles for any solution (as outlined in Section 1.2) can be contextualised with respect to this problem.

The primary aim is to protect the individual privacy of each user so that they have plausible deniability, which can be achieved by perturbing each user's trajectory in order to satisfy the requirements of LDP. It is assumed that each user shares one trajectory each, and it is shared at the end of the data collection period (e.g., one day). This contrasts with other work that focuses on the continual release/streaming setting [e.g., 132, 197, 243]. We discuss the privacy implications of these assumptions in Section 5.4.

Although the primary aim of the mechanism is to preserve privacy, the practical goal is to ensure that the perturbed trajectories have high utility (i.e., possess similar statistics and trends as the real data). Information to preserve to ensure high utility can range from hotspot information (e.g., what are the most popular POIs) to co-location patterns (e.g., how many other customers were at a shop at the same time as some infected person) and previous travel history (e.g., where were customers before visiting a shop). In theory, utility can be boosted by linking 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.

Example. Imagine that Alice is at a bar that is equidistant from another bar and a cinema, and there is also a school that is situated across town from them. Given their current location, being associated with the other bar should be more likely than being associated with a cinema (as it is more semantically similar), which itself should be more likely than the school (due to geographic proximity).

Furthermore, as discussed in previous chapters, 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, these protections can be too strong and can negatively affect the utility of the output dataset. To improve utility, publicly known external information can be utilised to influence the mechanism's output (e.g., by modelling popularity distributions, or the set of realistic or feasible outputs).

Example. Imagine Brian wishes to perturb their location one afternoon. Two equidistant POIs from them are a nightclub and a football stadium. From external information, it is known that the nightclub is currently closed, and there is a football game in progress. Using this knowledge, perturbing Brian's location to the stadium should be much more likely than to the nightclub.

Finally, as the solutions are set to utilise a wide range of public information from the real world, it is equally important that the solution can be applied to real-world settings, at scale. Consequently, the privacy and utility goals can be complemented with the desire for the solution to be efficient and scalable for large urban datasets.

Threat Model

There are two primary adversarial threats when releasing user-level trajectory data: location identification and journey tracing. As in previous chapters, we assume that an adversary has access to all of the same public knowledge as the proposed mechanisms. A location identification attack seeks to identify the exact time that a user was at a particular POI, or their entire spatio-temporal trajectory. By definition, LDP protects users against such attacks by providing users with plausible deniability with respect to their spatio-temporal location(s). This prevents an adversary from determining any of a user's exact spatio-temporal locations with high certainty. Moreover, even if an adversary were to know all but one of a user's true spatio-temporal locations, they would not be able to definitively identify the final location. This is because we provide privacy at the trajectory level, not the event level.

A journey tracing attack seeks to utilise a user's previous locations to determine their current or next location. Such attacks are especially pertinent when using spatio-temporal data, given the high correlation between adjacent points [e.g., 43, 243]. Both of the solutions presented in this chapter protect against journey tracing attacks by perturbing trajectories as one entity, either by modelling them as single points or by using overlapping n-grams. On a practical level, this means that the perturbation can only occur once a user has completed their daily movements. Consequently, an adversary only has access to a user's (perturbed) data at the end of the day, thus precluding journey tracing attacks.

5.1.2 Definitions

This section introduces the notation and definitions necessary for this chapter. Π denotes a set of POIs, where an individual POI is denoted by π. Each POI $\pi \in \Pi$ has a number of attributes associated with it, which can represent the popularity, privacy level, category, etc. of the POI. All of these attributes can be modelled to allow temporal variation. We discretise 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, ...}. These times are the lower bound of time intervals (i.e., all times between 10:05:00 and 10:09:59, would be represented as 10:05).

We define a trajectory, τ, at the POI level as a sequence of POI-timestep pairs such that $\tau = \{[\pi_1, t_1], ..., [\pi_i, t_i], ..., [\pi_{|\tau|}, t_{|\tau|}]\}$, 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 $\widehat{\tau}$ and $\widehat{\mathcal{T}}$, respectively.

Combined space-time-category (STC) hierarchical partitions are used to 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, and c for category). 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 $\tau = \{R_1, ..., R_i, ..., R_{|\tau|}\}$ where R_i represents the STC region for the i^{th} point in the trajectory. Consider that the first point in a trajectory is [Hyde Park, 10:54am]. This might give R_s = West London, R_t = 10–11am, and R_c = Park, leading to $R_{stc} = R_1$ = [West London, 10–11am, Park].

Chaining regions (or POIs) together forms n-grams, whereby: $w_k^n = \{R_k, ..., R_{k+n-1}\}$, with $\mathcal{W}^n$ denoting the set of all possible n-grams. $\tau(a, b)$ specifies 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. $d_\chi(\pi_i, \pi_j)$ denotes the distance in dimension χ between π_i and π_j, and $d(R_i, R_j)$ denotes the distance between R_i and R_j. Distance functions are discussed more in Section 5.3.5.

Popularity

Popularity is a characteristic of the data that is important to preserve if the output trajectories are to be useful for analytics tasks. The popularity of a value x is defined to be the number of times it occurs in the set $\mathcal{T}$. This notion of popularity is flexible to the granularity of aggregation (i.e., one can quantify the popularity of apples, or fruit in general), and it can be applied to multi-attributed tuples (i.e., STC regions). For example, if many people go to Wembley Stadium on a Saturday afternoon, we aim for the number of times the tuple [Wembley Stadium, 3–5pm] appears in the output data to be high, and similar to its count in the true data. In this chapter, popularity information is assumed to be public knowledge, and we explore the effects of relaxing this assumption in Chapter 6.

5.1.3 Reachability

As mentioned in Section 2.3, the notion of reachability is needed to ensure realism in perturbed trajectories by preventing illogical trajectories from being produced. In the context of this work, reachability can be defined as follows:

Definition 4 (Reachability). *π_b is reachable from π_a at time t if $d_s(\pi_a, \pi_b) \leq \psi(t)$, where $\pi_a, \pi_b \in \Pi$.*

The threshold, ψ, represents the maximum distance that one can travel in a certain time period. ψ can be specified directly, or it can be a function of g_t and a given travel speed. This definition also accommodates time-varying and asymmetric distances (i.e., congestion and one-way roads).

To illustrate the idea, consider the trajectory {New York City, Tokyo, London}. This would be unrealistic if the time granularity was one hour as it is infeasible to travel between these cities in less than one hour. Alternatively, imagine the trajectory: {Big Ben, London Eye, Tower Bridge}. A realistic bigram to perturb to might be {London Eye, Trafalgar Square} as the two locations can be reached within one hour, whereas {London Eye, Stonehenge} is unrealistic as it would not satisfy the reachability constraint. 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 $w_a^n = \{\pi_a, ..., \pi_i, \pi_{i+1}, ..., \pi_b\}$, the reachability constraint requires π_{i+1} to be reachable from π_i at time t_i for all $a \leq i < b$. At the region level, we deem any R_a and R_b to be reachable if there is at least one $\pi_i \in R_a$ and at least one $\pi_j \in R_b$ that satisfy reachability.

5.2 Global Solution

Before introducing the n-gram-based solution, we first consider what can be viewed as the 'global' solution, before examining why this solution is generally infeasible for all but the smallest problem settings. 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 $\widehat{\tau}$ is:

$$\Pr(\widehat{\tau} = \tau_i) = \frac{\exp\left(-\epsilon d_\tau(\tau, \tau_i)/2\Delta_{d_\tau}\right)}{\sum_{\tau_i \in \Omega} \exp\left(-\epsilon d_\tau(\tau, \tau_i)/2\Delta_{d_\tau}\right)} \tag{5.1}$$

where Ω is the set of all possible trajectories, represented as points in high-dimensional space, and d_τ is the distance function that represents the distance between trajectories. Once the probabilities have been defined, the exponential mechanism is used to perturb trajectories. Proof that the global solution satisfies ϵ-LDP follows from Equations 2.4 and 5.1.

Analysis of Global Solution

To assess the feasibility of the global solution, we first analyse its worst-case time complexity.

Theorem 8. *The worst-case time complexity of the global solution is: $O\left(\frac{|\Pi|^{|\tau|} \times |T|!}{|\tau|! \times (|T| - |\tau|)!}\right)$.*

Proof. As each timestep can only appear once, the number of possible timestep sequences is:

$$\frac{|T|!}{|\tau|! \times (|T| - |\tau|)!} \tag{5.2}$$

where $|T|$ is the number of possible timesteps, which is a function of g_t (in minutes) and the length of the data collection period (e.g., $|T| = \frac{24\times 60}{g_t}$ for one day). The number of possible POI sequences, in the worst-case scenario, is: $|\Pi|^{|\tau|}$. Hence, the maximum size of Ω is:

$$|\Omega| = \frac{|\Pi|^{|\tau|} \times |T|!}{|\tau|! \times (|T| - |\tau|)!} \tag{5.3}$$

The global solution requires instantiating all trajectories in Ω, hence the worst-case time complexity is $O\left(\frac{|\Pi|^{|\tau|}\times|T|!}{|\tau|!\times(|T|-|\tau|)!}\right)$. ■

In reality, the reachability constraint reduces the number of possible trajectories as not all POIs are reachable between successive timesteps for all timestep sequences. Assuming that (on average) ϱ% of all POIs are reachable between successive timesteps, the number of possible trajectories is reduced by a factor of $\varrho^{|\tau|-1}$. To illustrate this, imagine a small-scale example where $|\tau| = 5$, $g_t = 15$ minutes, $|\Pi| = 1{,}000$, and $\varrho = 20\%$. Even under these settings, $|\Omega| \approx 9.78 \times 10^{19}$, which means Ω remains computationally infeasible to compute and store.

Global Solution Improvements

As the exponential mechanism, as introduced by McSherry and Talwar [168], requires the instantiation of all possible trajectories, it naturally attracts a high time and space cost. This has led to many papers that focus on trying to increase its efficiency, without compromising on privacy or utility. Two of these approaches can be applied to the global solution, as we now outline.

The first is the subsampled exponential mechanism [144], which is shown by the authors to preserve ϵ-DP. To use the subsampled exponential mechanism, a proportion of the possible trajectories are sampled at random, and the exponential mechanism is applied to this subsampled set of outputs. The utility of the subsampled exponential mechanism is highly dependent on the shape of the probability density function of the quality function. In this setting, most high-dimensional distance functions will exhibit significant skew, with many more trajectories with moderate-to-large d_τ values than those with very small d_τ values. We use Ψ to denote the fraction of Ω that is sampled, which gives $|\Omega^*| = \Psi|\Omega|$, where Ω^* is the set of sampled trajectories. From above, we intuitively see that the worst-case time complexity of the global solution with subsampling is $O\left(\frac{\Psi|\Pi|^{|\tau|}\times|T|!}{|\tau|!\times(|T|-|\tau|)!}\right)$. As evident from the previous example, Ψ would need to be very small for the global solution to be computationally feasible and accurate.

Alternatively, the 'permute and flip' approach [164] can be used to avoid instantiating all d_τ values. It relies on selecting a possible output trajectory at random with uniform probability, determining the probability of selection (where $\alpha_\tau \propto \exp(-\epsilon d_\tau)$) before tossing a biased coin. If the coin shows heads, the selected trajectory is output; otherwise, the process is repeated. Although this variant is guaranteed to output a result, there is no guarantee of the time taken to achieve this. As with the subsampled exponential mechanism, the high dimensionality typically means that many possible trajectories have high d_τ values, corresponding to very low probabilities. Hence, it can take a long time to select an output using permute and flip. For example, if $g_t = 30$ minutes and $|\Pi| = 20$, 1.4×10^{10} trajectories exist, resulting in the runtime of *each* permute and flip-based perturbation being of the order of hours. Indeed, the worst-case time complexity of this adaptation of the global solution has the same time complexity as the unadapted version, which limits the practical appeal of this approach.

5.3 *n*-Gram-Based Solution

Instead of considering trajectories as high-dimensional points, we can instead use 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 for accurately modelling trajectory data.

This n-gram-based solution (summarised in Figure 5.1) 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, which allows us to invoke LDP's post-processing property without using the privacy budget.

5.3.1 Hierarchical Decomposition

As the size of the POI set increases, the number of feasible outputs for POI-level perturbation grows exponentially, quickly becoming computationally intractable. To address this challenge, we utilise hierarchical decomposition to divide POIs into STC regions.

STC Region Composition

We first divide the physical space into N_s spatial regions. For each $R_s \in \mathcal{R}_s$, we create N_c regions – one for each POI category. For each space-category region, we create N_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). STC regions that have zero POIs within them (e.g., R_{stc} = [top of mountain, 3–4am, church]) are removed, which ensures that these regions are not included in $\mathcal{W}^n$. As all this information is public, it does not consume any of the 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. $\mathcal{R}_c$ can be derived from known POI classification hierarchies, such as those published by OpenStreetMap, or it can be a user-specified hierarchy. $\mathcal{R}_t$ is most easily formed by considering a number of coarse (e.g., hourly) time intervals into which POIs can be assigned.

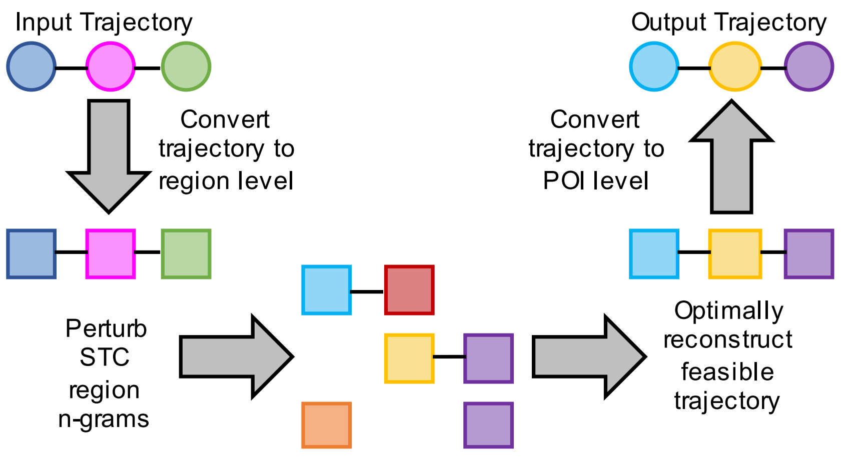

Input Trajectory
Output Trajectory
Convert trajectory to region level
Convert trajectory to POI level
Perturb STC region n-grams
Optimally reconstruct feasible trajectory

Figure 5.1: Overview of the n-gram-based solution

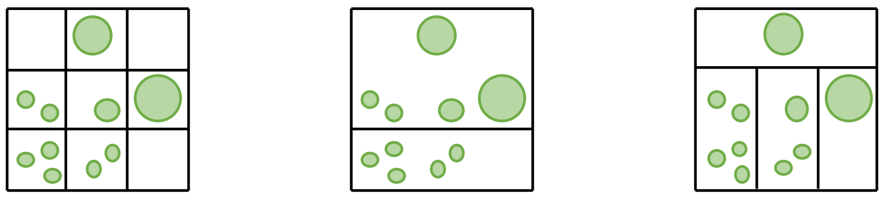
STC Region Merging

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, 1–2am, Nightclub] and R_{stc} = [Main Street, 11pm–12am, Bar], they can both 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 centre 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.

Additional rules can be enforced to further restrict the revealed information. The relative popularity and/or privacy of POIs can be taken into consideration in the merging stage. For example, consider bars surrounding a football stadium. Suppose most are clustered together in one STC region, while one outlying bar is the sole occupant of another STC region. Consequently, if an individual is associated with the second STC region, the exact bar can be inferred (which may be undesirable for some). Hence, we require that each STC region has κ POIs associated with it, where κ is a pre-defined function of POI attributes. A low κ value covers popular instances (e.g., $\kappa = 1$ is plausible for a sports stadium during a match), and a higher κ is appropriate for less popular, more private instances (e.g., drug rehabilitation hospitals).

To further illustrate this, consider Figure 5.2a, which shows ten POIs, divided in a 3×3 grid where larger circles indicate a more popular POI. Figure 5.2b shows how regions might be merged if we only consider geographic proximity, whereas Figure 5.2c shows the resultant regions when merging accounts for perceived popularity. We see more POIs in regions



(a) Unmerged regions **(b)** Geographic merging **(c)** Popularity-based merging

Figure 5.2: Illustrative example of merging space-time-category 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 is recommended.

n-gram Set Formation

As a final pre-processing step, $\mathcal{W}^n$ is defined by first instantiating all possible n-gram combinations of STC regions. All n-gram combinations that do not satisfy the reachability constraint are then removed. The semantic distances between all STC regions, and all STC region n-grams, are also pre-computed.

5.3.2 *n*-gram Perturbation

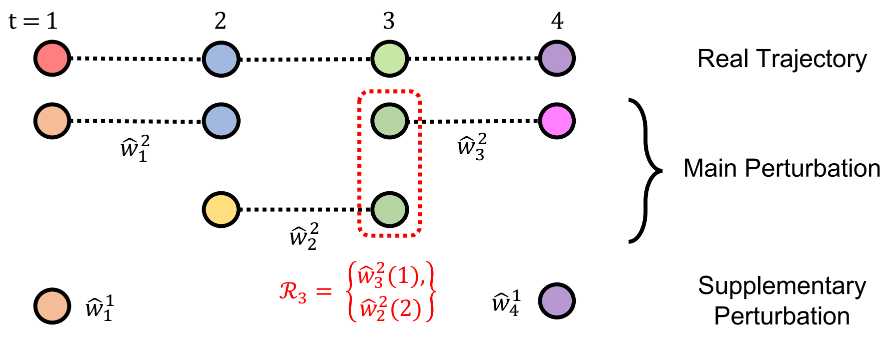
Once $\mathcal{W}^n$ has been defined, we convert each trajectory from a sequence of POI-timestep pairs to a sequence of STC regions. The next step is to perturb the STC regions of τ using overlapping fixed-length n-grams and the exponential mechanism.

Before outlining the method for perturbation, we need to introduce some new notation. $\widehat{\mathcal{W}}$ is the set that holds all the perturbed n-grams of τ, and $\widehat{w}_a^n = \{\widehat{r}_a, ..., \widehat{r}_i, ..., \widehat{r}_b\}$ is the perturbed n-gram, where a and $b (= a + n - 1)$ are the indices of τ, and $\widehat{w}_a^n \in \widehat{\mathcal{W}}$. Importantly, there is a subtle difference between $\widehat{\tau}(a, b)$ and $\widehat{w}_a^n$. In $\widehat{\mathcal{W}}$, for any timestep, there are multiple possible regions associated with each trajectory point, whereas $\widehat{\tau}$ is the final reconstructed trajectory and so there is only one region for each trajectory point. Finally, $\mathcal{R}_k$ is the set containing the STC regions from all of the perturbed n-grams in $\widehat{\mathcal{W}}$ that contain the k^{th} point in the trajectory. For example, $\mathcal{R}_3$ comprises the first region from $\widehat{w}_3^2$ and the second region from $\widehat{w}_2^2$, as illustrated in Figure 5.3.

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

$$\Pr(\widehat{w}_a^n = w_i^n) = \frac{\exp\left(-\epsilon' d_w(\tau(a,b), w_i^n)/2\Delta_{d_w}\right)}{\sum_{w_i^n \in \mathcal{W}^n} \exp\left(-\epsilon' d_w(\tau(a,b), w_i^n)/2\Delta_{d_w}\right)} \tag{5.4}$$

where $d_w(\tau(a,b), w_i^n)$ is the function that quantifies the distance between n-grams. The privacy budget is divided evenly by setting $\epsilon' = \frac{\epsilon}{N_p}$, where N_p is the number of perturbations. 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 the



t = 1
2
3
4
Real Trajectory
$\widehat{w}_1^2$
$\widehat{w}_3^2$
Main Perturbation
$\widehat{w}_2^2$
$\mathcal{R}_3 = \left\{ \begin{matrix} \widehat{w}_3^2(1), \\ \widehat{w}_2^2(2) \end{matrix} \right\}$
$\widehat{w}_1^1$
$\widehat{w}_4^1$
Supplementary Perturbation

Figure 5.3: Illustration of main and supplementary perturbations; different colours indicate different STC regions

exponential mechanism to sample from $\mathcal{W}^n$ and we store $\widehat{w}_a^n$ in $\widehat{\mathcal{W}}$. We repeat this for increasing values of a and b, as shown in Figure 5.3.

Using overlapping n-grams gives better accuracy than non-overlapping n-grams or merely perturbing points independently. It allows us to '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, $\widehat{\tau}(3)$ 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, neighbouring points $n-1$ times, etc., in addition to using information from $2n-1$ points to influence the perturbation of $\tau(3)$.

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 one time only. 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 Equation 5.4, but with different $\mathcal{W}^n$ sets and different bounds for a and b. In our example, we would use unigrams to obtain $\widehat{w}_1^1$ and $\widehat{w}_4^1$. Figure 5.3 illustrates the necessity for supplementary perturbation.

5.3.3 Region-Level Trajectory Reconstruction

Given a collection of perturbed n-grams, where each point in the trajectory is represented n times, we need to reconstruct a feasible trajectory where each point in the trajectory is associated with just one POI. To achieve this, we define an optimisation problem that reconstructs a trajectory, $\widehat{\tau}$, using the perturbed n-grams in $\widehat{\mathcal{W}}$. Importantly, this is post-processing of the LDP output, and does not consume any of the privacy budget.

First, we define two error terms that measure the similarity of regions to the perturbed data. The first error term is the 'region error' $\zeta_k^{R_i}$, which is the total distance between some region R_i and all the regions in $\mathcal{R}_k$. Formally:

$$\zeta_k^{R_i} = \sum_{R_j \in \mathcal{R}_k} d(R_i, R_j) \tag{5.5}$$

The second error term is the 'bigram error' ζ_i^w, which is the sum of the two relevant region error terms. More formally, it is defined as:

$$\zeta_k^w = \zeta_k^{w(1)} + \zeta_{k+1}^{w(2)} \tag{5.6}$$

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. Even if $n \neq 2$ when conducting the main perturbations, reconstruction is always conducted using bigrams as we seek to form a trajectory that links adjacent points in the trajectory, and these links are naturally obtained using bigrams.

Having defined the error terms, the minimisation problem can be defined as:

$$\min \sum_{k=1}^{|\tau|-1} x_k^w \zeta_k^w \tag{5.7}$$

such that:

$$x_k^w \cdot y(w_k, w_{k+1}) = x_{k+1}^w \cdot y(w_k, w_{k+1}) \quad \forall\ 1 \leq k < |\tau| \tag{5.8}$$

$$y(w_k, w_{k+1}) = \begin{cases} 1 & \text{if } w_k(2) = w_{k+1}(1) \\ 0 & \text{otherwise} \end{cases} \tag{5.9}$$

$$\sum_{k=1}^{|\tau|-1} x_k^w = |\tau| - 1 \tag{5.10}$$

$$\sum_{w \in \mathcal{W}^2} x_k^w = 1 \quad \forall\ 1 \leq k < |\tau| \tag{5.11}$$

where x_k^w is a binary variable encoding whether w is selected for index k. The objective (Equation 5.7) is to minimise the total bigram error across the trajectory. Equations 5.8 and 5.9 are continuity constraints that ensure that consecutive bigrams share a common region. Equations 5.10 ensures that the number of bigrams selected is correct, and Equation 5.11 ensures that 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 optimisation problem will generally be within the scope of most linear programming solvers (see Section 5.4). Nevertheless, we introduce a step to limit the set of possible bigrams that can appear in the reconstructed trajectory further. Once n-gram perturbation is complete, we obtain the minimum bounding rectangle (MBR) defined by all $R_{stc} \in \widehat{\mathcal{W}}$. The set created by this MBR contains all regions that appear in a perturbed n-gram in $\widehat{\mathcal{W}}$: $\mathcal{R}_{mbr} = \bigcup_{k=1}^{|\tau|} \mathcal{R}_k$. From this, we define $\Pi_{mbr} \subseteq \Pi$, which contains all of the POIs in this MBR, and $\mathcal{R}_{mbr}^*$, which is the set of STC regions that contain at least one POI in Π_{mbr}. From this, we define $\mathcal{W}_{mbr}^2$ 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 minimise the error with respect to the perturbed n-grams in $\widehat{\mathcal{W}}$, all of which are included in $\mathcal{R}_{mbr}^*$.

5.3.4 POI-Level Trajectory Reconstruction

The final step of the perturbation mechanism 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 are not be published.

In most cases, most POI-level trajectories are feasible as $\mathcal{W}^n$ is defined based on the reachability criterion. For simplicity and speed, we generate an individual trajectory by randomly sampling POIs from STC regions, before checking that it satisfies the reachability constraint. If it does, the trajectory is output; if not, another trajectory is generated. This continues until we generate a feasible trajectory, reach a threshold, γ, or exhaust all possible combinations. Experimentally, the threshold of $\gamma = 50{,}000$ was reached less than 2% of the time. More refined methods for sampling POIs can be implemented (e.g., by utilising popularity information), and this is explored more in Chapter 6.

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, Town Centre], [Pub, 9–10pm, Town Centre], [Bar, 9–10pm, Suburb]$\}$. Reachability may mean that the suburban bar is only reachable from the pub in 55 minutes, meaning that it is impossible to visit all three venues in an hour. Accordingly, we smooth the timesteps such that either the first POI is visited between 8 and 9pm, or the third POI is visited between 10 and 11pm.

5.3.5 Distance Functions

In order for the mechanism to exhibit the desired behaviour, we define a multi-attributed distance-based quality function. Note that the mechanism is not reliant on any specific distance/quality function – any other distance function can be used, without needing to change the mechanism. Finally, to ensure that $\Delta_{d_\chi} = 1$, all of the distance functions are normalised to have values in the range (0,1).

Physical Distance. $d_s(\pi_a, \pi_b, t)$ denotes the physical distance from π_a to π_b at time t, which can be derived using any distance measure (e.g., Euclidean, Haversine, road network). We similarly use $d_s(R_a, R_b)$ to denote the physical distance between R_a and R_b, with the value determined by the distance between the geographic centroids of the POIs in the two regions.

Time Distance. The time distance between regions is defined as the absolute time difference between two STC regions. That is, $d_t(t_a, t_b) = |t_a - t_b|$. 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.

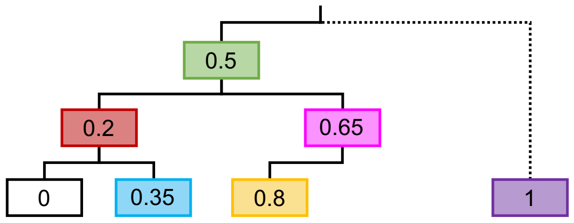


Figure 5.4: Category distance values, relative to left-most level 3 node

Category Distance. Category distance, d_c, is quantified using a multi-level hierarchy. In this chapter, a three-level category hierarchy is used, although any number of levels can be used. Level 1 is the highest level (i.e., uses the coarsest granularity), whereas level 3 is the lowest level (i.e., uses the finest granularity). Figure 5.4 illustrates how d_c varies across hierarchical levels, relative to the leftmost level 3 (white) node. Category distance is defined 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 (i.e., the uppermost (green) node), we deem them to be unrelated and $d_c = 1$ (indicated by the dotted line and purple node in Figure 5.4).

Combining Distances. Distance functions are combined as follows:

$$d(R_a, R_b) = \left(\omega_s d_s(R_a, R_b)^2 + \omega_t d_t(R_a, R_b)^2 + \omega_c d_c(R_a, R_b)^2\right)^{1/2} \tag{5.12}$$

where ω_χ are a set of non-negative weights such that $\omega_s + \omega_t + \omega_c = 1$. For simplicity, we set $\omega_\chi = \frac{1}{3}$ throughout. To determine the 'distance' between two n-grams, we use element-wise summation. For example, the distance between two bigrams ($w_i = \{R_1^i, R_2^i\}$ and $w_j = \{R_1^j, R_2^j\}$) is calculated as $d(R_1^i, R_1^j) + d(R_2^i, R_2^j)$. More generally, for any two n-grams:

$$d_w(w_i, w_j) = \frac{1}{n}\sum_{a=1}^{n} d(R_a^i, R_a^j) \tag{5.13}$$

5.4 Theoretical Analysis

Several aspects of the mechanism can now be analysed theoretically, including the level of privacy provided to trajectories, and the computational cost of various stages of the n-gram-based solution.

Number of Perturbations

With the main and supplementary perturbation processes outlined, it is possible to quantify the number of perturbations needed for each trajectory. First, each n-gram $\tau(a, b)$ is perturbed once, where $a = (1, |\tau| - n + 1)$ and $b = (n, |\tau|)$. This yields $|\tau| - n + 1$ perturbations, which are performed as the main perturbations. For points in τ in the range $(n, |\tau| - n + 1)$, each point has been perturbed in n separate n-grams, but the remaining points have been perturbed fewer times. Supplementary perturbations ensure that each point is covered equally, and there are $n - 1$ of these perturbations at each end of the trajectory, with each perturbation using a shorter n-gram each time. Hence, the total number of perturbations is $N_p = |\tau| - n + 1 + 2(n - 1) = |\tau| + n - 1$.

Privacy Analysis

As the privacy budget is divided into N_p equal portions (i.e., $\epsilon' = \frac{\epsilon}{N_p}$), sequential composition can be (trivially) applied to observe that the overall privacy loss for each trajectory is $N_p\epsilon' = N_p \times \frac{\epsilon}{N_p} = \epsilon$. Furthermore, as the size of adjacent datasets is 1, ϵ-DP results are equivalent to ϵ-LDP results here, which guarantees that the perturbation of each trajectory satisfies ϵ-LDP.

As discussed in Section 2.3, external knowledge is *only* used to enhance utility, whereas privacy is provided through the application of the exponential mechanism. Hence, an adversary (who is assumed to have the same information) cannot use this information to learn meaningful information with high probability. As the solution is predicated on a 'one user, one trajectory' basis, inference attacks based on repeated journeys from the same user are prevented by definition. Sequential composition can be used to extend the 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. 6, 43, 84, 136], this setting sees the user share all data at the end of their trajectory. Hence, as user-level ϵ-LDP is provided, the LDP privacy guarantee protects against spatial and temporal correlation attacks.

Theoretical Utility

Even though the global solution cannot be realised in practice owing to its high time and space complexity, its theoretical utility can be assessed.

Theorem 9. *The utility of the global solution is:*

$$\Pr\left[d_\tau(\tau, \widehat{\tau}) \leq -\frac{2}{\epsilon}\left(\ln|\Omega| + \xi\right)\right] \leq e^{-\xi} \tag{5.14}$$

Proof. The quality function is the distance function d_τ, and its sensitivity is $\Delta_{d_\tau} = 1$. The optimal value of the function is obtained iff $\widehat{\tau} = \tau$; hence, $OPT_{d_\tau} = 0$ and $|\Omega_{OPT}| = 1$. Substituting these values into Equation 2.5 yields Equation 5.14. ■

As the n-gram-based solution has multiple post-processing stages, a theoretical utility guarantee for the entire mechanism remains elusive. However, the utility of a single n-gram perturbation can be analysed.

Theorem 10. *The utility of a single n-gram perturbation is:*

$$\Pr\left[d_w(\tau(a,b), w_a^n) \leq -\frac{2}{\epsilon'}\left(\ln|\mathcal{W}^n| + \xi\right)\right] \leq e^{-\xi} \tag{5.15}$$

Proof. The quality function is the distance function d_w, and its sensitivity is $\Delta_{d_w} = 1$. The optimal value is obtained iff $w_a^n = \tau(a,b)$; hence, $OPT_{d_w} = 0$ and $|\mathcal{W}_{OPT}^n| = 1$. Substituting these values into Equation 2.5 yields Equation 5.15. ■

These two proofs indicate that two variables directly influence utility: the privacy budget, and the size of the output set. The size of the output set is dependent on the granularity of hierarchical decomposition, the strictness of the reachability constraint, and (for the n-gram-based solution) the value of n. For the n-gram-based solution, ϵ' is a function of $|\tau|$ and n, whereas ϵ is fixed for the global solution. Given these intricate dependencies, and the inclusion of external knowledge, properly assessing the practical utility of the mechanisms can only be done empirically.

Computational Cost

Finally, we discuss the computational costs of the n-gram-based approach, and highlight how it is highly practical. 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. When choosing n, it is important to consider the computational time and space requirements. Precomputing $\mathcal{W}^n$ requires $O(|\Pi|^n)$ space, which becomes infeasible for large cities and when $n \geq 3$. If $n \geq 3$ and $|\Pi|$ 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, setting $n = 2$ is recommended as $n \geq 3$ will be unrealistic in most practical settings.

Time Complexities. As optimisation is used in the solution, it is difficult to provide a closed-form expression for the time complexity for the entire perturbation mechanism. However, we can assess the time complexity of individual phases of the mechanism, and also discuss the factors controlling the scale of the optimisation problem. 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 $O(\gamma(|\tau| + |\tau - 1|))$.

Optimal Reconstruction Complexity. Section 5.3.3 presents the optimal reconstruction phase, which uses integer linear programming. Here, we assess the scale of the problem in terms of the number of variables and constraints. Let $\Gamma = |\mathcal{R}^*_{mbr}|$. The number of feasible bigrams will be Γ^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 $\Gamma(|\tau| - 1)$ variables (i.e., one x_i^w per bigram, per trajectory point). The continuity constraints (Equations 5.8 and 5.9) impart $\Gamma(|\tau| - 1)$ constraints, and the capacity constraints (Equations 5.10 and 5.11) impart $|\tau| - 1$ constraints. Hence, the optimisation problem has $(\Gamma|\tau| + |\tau| - \Gamma - 1)$ constraints in total. Closed-form expressions for the expected runtime of optimisation problems depend on the exact solver chosen; one is not given here, but readers are directed to state-of-the-art efficiency bounds [e.g., 58, 229].

5.5 Evaluation

The proposed mechanism can now be evaluated against a range of alternative approaches. The aims of these experiments are threefold: (a) analyse the mechanism and gather insights from its behaviour; (b) compare the approach to alternative methods; and (c) demonstrate the practical utility of the mechanism in the context of application-inspired queries.

5.5.1 Data

This evaluation uses a range of real, synthetic, and semi-synthetic trajectory datasets. As there is a chronic lack of high quality, publicly available POI sequence data, existing real datasets are augmented to make them suitable for a comprehensive evaluation.

Real Data

The first dataset combines Foursquare check-in [259] and historic taxi trip [179] 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 $|\Pi|$ most popular POIs to form the set Π. The pick-up and drop-off locations of each taxi driver's daily trips in order are concatenated, on the premise that we are privatising the driver's movements, as opposed to passenger movements, in order to protect their business. This co-ordinate data is matched with the nearest POI from the Foursquare dataset; if no POIs within 100 m are found, the point is discarded. For both datasets, the data is cleaned by removing repeat points with the same venue ID or exact latitude-longitude location. Where points occur less than g_t minutes apart, all points, bar one, are removed at random. The publicly available Foursquare category hierarchy [93] is used to assign a single category to each POI. As a simplifying assumption, each POI is assumed to only have one category associated with it (i.e., if POIs have more than one, we pick one at random from the attributed categories). Opening hours are specified manually for each broad category (e.g., 'Food', 'Arts and Entertainment'), and all POIs of that (parent) category have those hours. However, the mechanism is designed to allow POI-specific opening hours.

Semi-Synthetic Data

Safegraph is a company specialising in collected and sharing location data. We use their Patterns and Places data [199] is used to generate semi-synthetic trajectories. The length of the trajectory is determined using a uniform distribution with bounds (3,8), and the start time is found 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. The original data has information on the dwell time at each POI (i.e., how long the average customer spends at the POI). To model the time gap between adjacent POIs, we sample from the dwell time distribution to determine the time spent in one location, and also sample the time spent travelling 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 North American Industry Classification System hierarchy [226], which is used for the category hierarchy. Opening hours information for POIs is sparse, and so the process used with the Taxi-Foursquare data is repeated here.

Campus Data

To assess the effects of changing the n-gram length, a smaller dataset is necessary. The University of British Columbia campus [225] (used here) has 262 campus buildings, which act as POIs, with nine categories (e.g., 'academic building', 'student residence', etc.). Trajectory length and start time is determined in the same way as for the Safegraph data. For each subsequent timestep in the trajectory, the uniform distribution with bounds (g_t, 120) minutes is sampled. The category of the first POI is chosen at random, based on pre-defined probabilities, 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. Three popular events are artificially induced 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; 1,000 people at Stadium A at 2–4pm; and 2,000 people in some academic buildings at 9–11am.

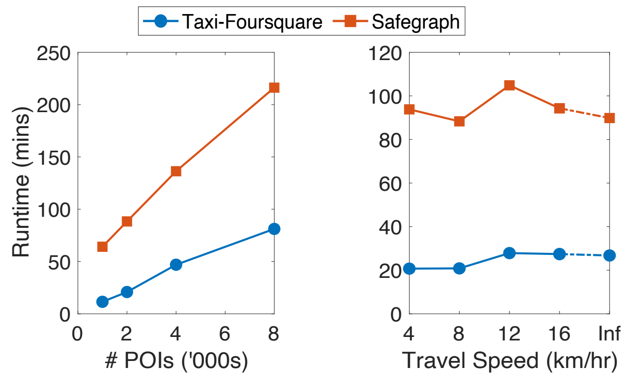
External Knowledge Specification

Although external knowledge is specified manually in these experiments, 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 points, etc. can be obtained directly through their API. This information can be stored in the POI-level database, with which the mechanism interacts. Importantly, specifying external knowledge is a pre-processing phase, and the overheads required to do so are minimal.

Pre-Processing Costs

The pre-processing necessary for these 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 5.5 shows the runtime costs for pre-processing step (a) for the two large-scale datasets. The runtime is heavily dependent on the size of Π, 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, localised 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, as this is a one-time process, this cost is (arguably) acceptable.

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 satisfies the reachability and other feasibility constraints. If there are infeasible aspects in a trajectory, smoothing operations can be performed in sub-second time. Given that these steps would have a negligible impact on runtime in a practical application, little focus is devoted to them here.



(a) Size of POI set (in 1000s) **(b)** Assumed travel speed

Figure 5.5: Pre-processing runtime costs, in minutes

5.5.2 Experimental Settings

By default, $g_t = 10$ minutes, $n = 2$, $|\Pi| = 2{,}000$, and $\epsilon = 5$. While this value for ϵ appears high, privacy budgets tend to be higher with LDP (as more noise is added overall) and this value is broadly similar to other real-world deployments of LDP by Apple [13] and Microsoft [77]. We assume all travel is at 4 km/hr (campus data) and 8 km/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 5.5.5. We use Haversine distance throughout. $\mathcal{T}$ is filtered 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, physical space is divided using a $g_s \times g_s$ uniform grid. The finest granularity is $g_s = 4$, and coarser granularities ($g_s = \{1, 2\}$) are used when performing spatial merging. We use the first three levels of the category hierarchies, and use the category distance function outlined in Section 5.3.5. Using these levels ensures that $d_c = 1$ 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 $\kappa = 10$.

Implementation. All experiments were conducted using Matlab 2020b on Linux servers comprising two Intel Xeon E5-2680 2.4 GHz 14-core processors, with 128 GB RAM per node.

5.5.3 Alternative Approaches

Owing to the novelty of this work, there are no suitable baseline methods in the literature; all existing work is based in the centralised DP domain, or uses relaxed definitions of LDP. Instead, the n-gram-based solution proposed in Section 5.3 is compared to several alternative approaches, summarised here.

Using Physical Distance Only. The most basic distance-based perturbation mechanism (called PhysDist) ignores external knowledge and only uses the physical distance between POIs/regions.

POI-Level n-Gram Perturbation. Instead of using hierarchically structured STC regions, the solution can be applied just on the POI-level. To control the size of $\mathcal{W}^n$, NGramNoH perturbs the time and POI dimensions separately. This approach requires a larger number of perturbations, which means that the privacy budget is divided into smaller portions, which will affect utility.

Independent POI Perturbation. The simplest approach is to perturb each POI independently of all others. There are 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, post-processing is used 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 (especially those close together in time). 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 2.2.5, 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 this mechanism.

5.5.4 Method Comparison

The hierarchical n-gram-based solution can now be compared to these alternative approaches in terms of the mean semantic distance (MSD) and runtime. Mean semantic distance is based on the distance between the real and perturbed trajectories, averaged across all three dimensions and all points in the trajectory. It is defined as:

$$MSD(\tau, \widehat{\tau}) = \frac{1}{|\tau|} \sum_{i=1}^{|\tau|} d(\pi_i, \widehat{\pi}_i) \tag{5.16}$$

where π_i and $\widehat{\pi}_i$ are (respectively) the i^{th} POIs in the real and perturbed trajectories. To determine MSD, we use the same distance definitions as outlined in Section 5.3.5.

Mean Semantic Distance

Table 5.1 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 next). The importance of including external knowledge, such as category information, is emphasised when comparing performance with PhysDist, which performs worse than all other methods. Performance gains are primarily achieved in minimising 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 analysing 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

Table 5.1: Mean semantic distance between real and perturbed trajectory sets

Method	Taxi-Foursquare			Safegraph			Campus		
	Time	Cat.	Space	Time	Cat.	Space	Time	Cat.	Space
IndNoReach	0.118	0.037	0.141	0.122	0.028	0.102	0.172	0.014	0.233
IndReach	0.117	0.037	0.143	0.124	0.029	0.111	0.169	0.014	0.238
PhysDist	0.133	0.087	**0.131**	0.135	0.084	**0.101**	0.180	0.030	0.241
NGramNoH	0.134	0.042	0.147	0.134	0.034	0.112	0.179	0.015	0.235
NGram	**0.098**	**0.017**	0.159	**0.077**	**0.013**	0.102	**0.101**	**0.008**	**0.222**

merging in the spatial dimension would help to minimise accuracy losses, although a moderate decrease in efficiency would have to be tolerated. A deeper analysis into different STC region merging approaches would be a valuable focus for future work.

Runtime Analysis

Table 5.2 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 optimisation 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 linear programming solver and tuning the optimisation parameters – aspects of work that were beyond the scope of this thesis. 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 optimisation problem as a result of STC region merging.

5.5.5 Parameter Variation

It is also important to examine how performance is influenced by the trajectory characteristics, and the mechanism or experiment parameters. Figures 5.6 and 5.7 show how mean semantic distance and runtime are influence by these factors, respectively.

Trajectory Length

Figures 5.6a and 5.6e 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 minimise 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 is to assign a constant ϵ' value for each perturbation, but this means that trajectories experience uneven privacy leakage (i.e., $\epsilon = (n + |\tau| - 1)\epsilon'$).

Figures 5.7a and 5.7e show how the runtime changes with trajectory length. As expected, IndNoReach and IndReach show little runtime variability. Of the optimisation-based

Table 5.2: Average runtimes 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

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

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.

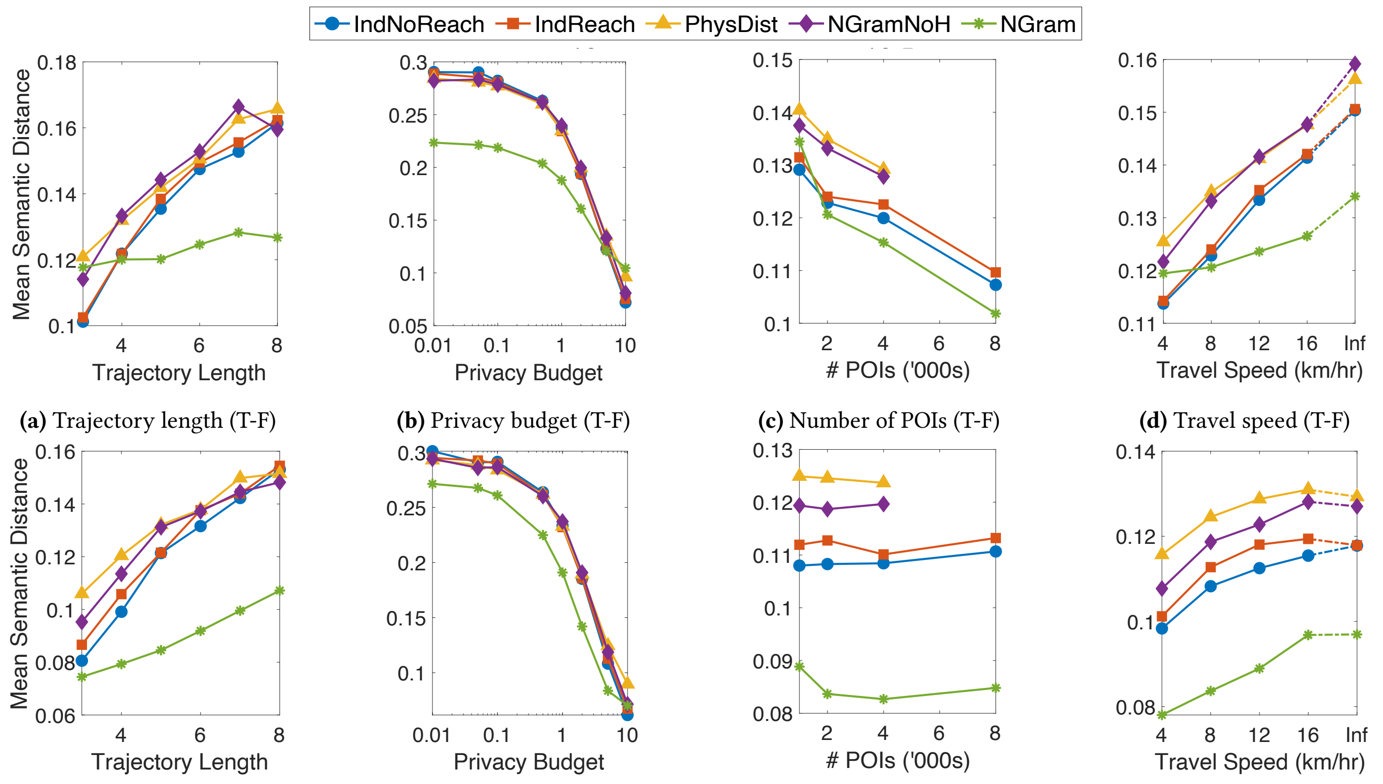
Privacy Budget

Figures 5.6b and 5.6f show how MSD is influenced by ϵ. All methods produce expected error profiles: as ϵ increases, the error decreases, although this behaviour is less notable for PhysDist. When $\epsilon < 1$, the drop-off in utility is less pronounced. This behaviour is likely to be indicative of the noise overwhelming the characteristics of the true data (i.e., the output data offers little value due to the added noise). Hence, when applying the mechanism, setting $\epsilon \geq 1$ is a good 'starting point'.

Figures 5.7b and 5.7f show that the runtime of NGram is relatively immune to the privacy budget, which indicates that the scale of the optimisation problem is unaffected by the privacy budget. Most remaining methods also exhibit this immunity, although PhysDist does not, which further emphasises the benefits of including external information in perturbation.

Size of POI Set

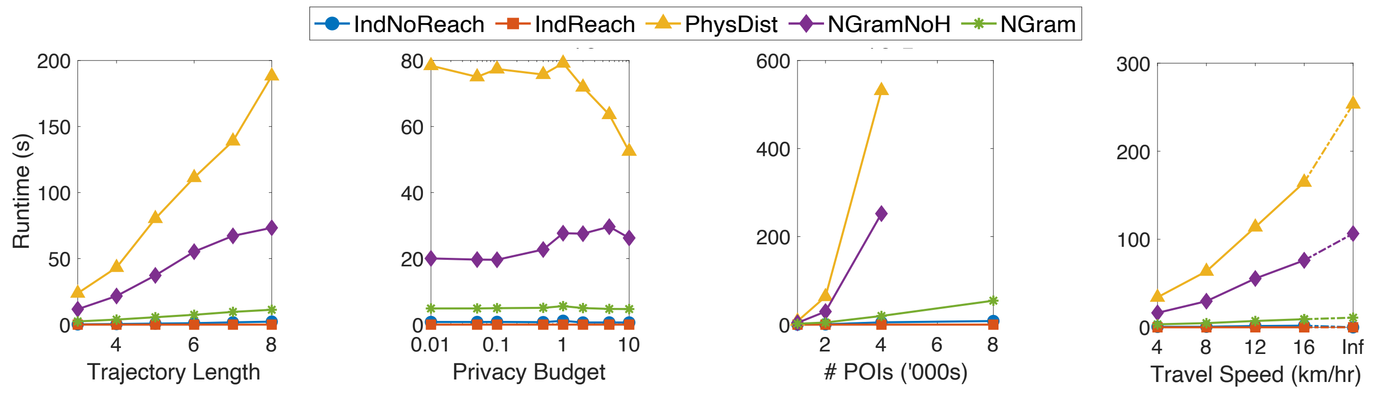
Figures 5.6c and 5.6g show the effect that the number of POIs has on MSD values. We omit PhysDist and NGramNoH when $|\Pi| = 8{,}000$, owing to their very high expected runtime. Interestingly, the error profiles are relatively immune to the effects of changing $|\Pi|$, with larger POI sets even bringing performance gains. This is primarily because STC regions are (initially) formed independently of the number of POIs. The error profiles also suggest that the optimal reconstruction phase can effectively prune the set of STC regions to identify the best trajectory from the perturbed n-grams. Figure 5.7c and 5.7g show a moderate runtime increase for NGram, which still perturbs trajectories in a reasonable time, even for large POI sets. This is in contrast to the other n-gram-based methods. As noted previously, at least 95% of runtime is spent during reconstruction, which indicates an important area of focus if real-world deployment is desired.



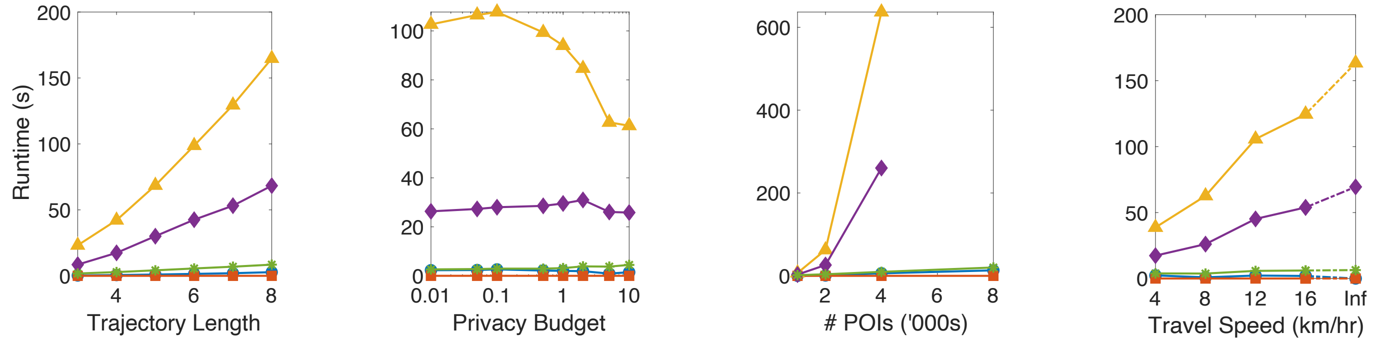
(a) Trajectory length (T-F) **(b)** Privacy budget (T-F) **(c)** Number of POIs (T-F) **(d)** Travel speed (T-F)

(e) Trajectory length (SG) **(f)** Privacy budget (SG) **(g)** Number of POIs (SG) **(h)** Travel speed (SG)

Figure 5.6: Mean semantic distance as experimental settings vary; Figures (a)–(d) use Taxi-Foursquare (T-F) data; Figures (e)–(h) use Safegraph (SG) data

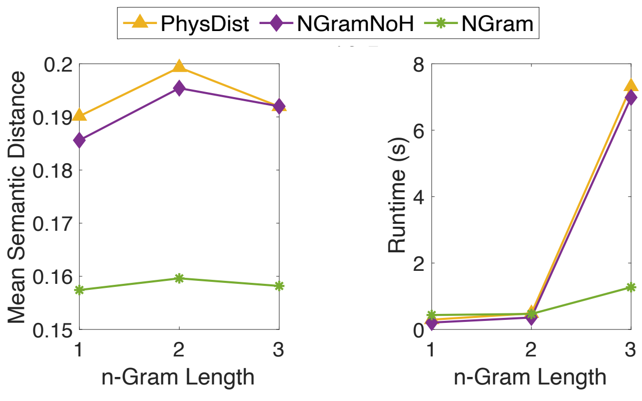


(a) Trajectory length (T-F) **(b)** Privacy budget (T-F) **(c)** Number of POIs (T-F) **(d)** Travel speed (T-F)



(e) Trajectory length (SG) **(f)** Privacy budget (SG) **(g)** Number of POIs (SG) **(h)** Travel speed (SG)

Figure 5.7: Average runtime as experimental settings vary; Figures (a)–(d) use Taxi-Foursquare (T-F) data; Figures (e)–(h) use Safegraph (SG) data



(a) Mean semantic distance **(b)** *n*-gram length

Figure 5.8: *n*-gram length vs. mean semantic distance and runtime (Campus data)

Reachability Constraint

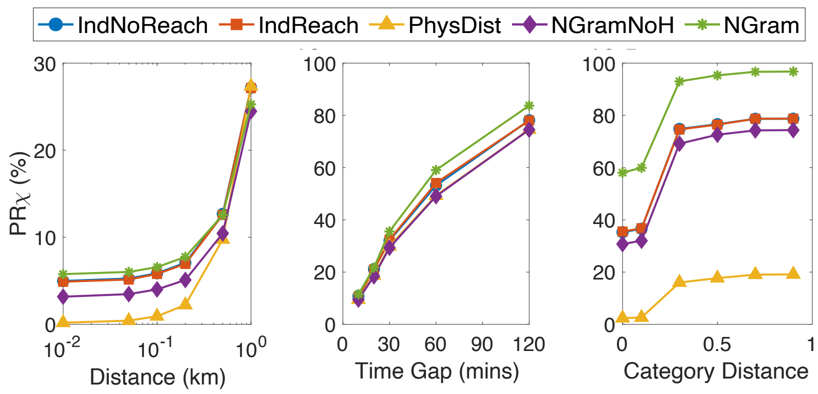
The effect of the reachability constraint on utility can be studied by experimenting with different assumed travel speeds; we use $\{4, 8, 12, 16\}$ km/hr. Additionally, setting $\psi = \infty$ models the case where there is no reachability constraint. MSD values increase as the reachability constraint becomes less strict or is removed entirely (Figures 5.6d and 5.6h). This is because more *n*-grams are feasible, and so the likelihood of the true *n*-gram being returned is reduced. NGram consistently outperforms all other methods in accuracy terms. In terms of runtime, it is relatively immune to changes in assumed travel speed, unlike other *n*-gram approaches (Figures 5.7d and 5.7h). Importantly, MSD is up to 31% lower for NGram compared to other methods when the reachability constraint is applied, and MSD values remain up to 22% lower when the reachability constraint is omitted.

n-gram Length

The smaller Campus dataset can be used to assess the effect of *n*-gram length for the three *n*-gram-based methods. The MSD and runtime values are shown in Figure 5.8 for when unigram, bigrams, and trigrams are used. NGram consistently outperforms other methods for all values of n, with $n = 2$ offering the best results. This is to be expected given the trade-off between capturing more information between neighbouring points (achieved with high n), and the division of ϵ and size of $\mathcal{W}^n$ (where low n is good). As expected, and discussed in Section 5.4, runtime costs start to become undesirable when $n = 3$, which supports the recommendation that bigrams are used in most real-world applications.

5.5.6 Preservation Range Queries

The range queries from Chapters 3 and 4 are helpful for analysing aggregate statistics, and these queries can be adapted to the trajectory level through so-called 'preservation range queries'. That is, for each point in each trajectory, we check to see whether the perturbed POI is within ρ units of the true POI. For example, a location preservation range query might examine whether $\widehat{\pi}_i$ is within 50 metres of π_i. We conduct these queries in all three



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

Figure 5.9: Variation in PR_χ values as ρ_χ changes

dimensions, and define the utility measure PR_χ as:

$$PR_\chi = \frac{1}{|\mathcal{T}|} \sum_{\tau \in \mathcal{T}} \left(\frac{1}{|\tau|} \sum_{i=1}^{|\tau|} \varsigma(\pi_i, \widehat{\pi}_i, \rho_\chi) \right) \times 100\% \tag{5.17}$$

where $\varsigma(\pi_i, \widehat{\pi}_i, \rho_\chi)$ equals 1 if $d_\chi(\pi_i, \widehat{\pi}_i) \le \rho_\chi$, and zero otherwise. For time queries, t_i and $\widehat{t_i}$ replace π_i and $\widehat{\pi}_i$, respectively.

Figure 5.9 shows the results for the preservation range queries for each dimension. For space and time queries, all methods perform similarly, although NGram slightly outperforms the other methods in general. There is a more notable difference in performance for category queries, with NGram clearly superior for all ρ_c values. Interestingly, there is an evident step at $\rho_c = 0.35$, suggesting strong preservation of category within levels 2 and 3, which demonstrates the solution's robustness. 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.

5.5.7 Hotspot Queries

As only spatial data was generated in Chapters 3 and 4, only spatial hotspots were considered. However, in reality, hotspots have a temporal component; indeed, they can span multiple dimensions and a range of granularities. Accordingly, in this chapter, a *spatio-temporal* hotspot is defined as the time interval during which the number of unique visitors at a location is above a threshold η. A hotspot is characterised by $h = \{t_s, t_e, \pi, c_{max}\}$, where t_s and t_e are the start and end times, π is the POI, and c_{max} 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 morning and afternoon rush-hour peaks). We consider three spatial granularities: the POI-level and spatial regions defined by 4×4 and 2×2 grids, with $\eta = \{20, 20, 50\}$ respectively. We consider three category granularities (i.e., levels $\{1, 2, 3\}$), with $\eta = \{50, 30, 20\}$ respectively. Hotspot preservation is quantified by calculating the 'hotspot distance' between the hotspots present in the real and perturbed data. If $\mathcal{H}$ and $\widehat{\mathcal{H}}$ are the hotspot sets when using the real and perturbed data

Table 5.3: AHD (in hours) 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

respectively, the average hotspot distance (AHD) between sets is:

$$AHD(\mathcal{H}, \widehat{\mathcal{H}}) = \frac{1}{\left|\widehat{\mathcal{H}}\right|} \sum_{\widehat{h} \in \widehat{\mathcal{H}}} \min_{h \in \mathcal{H}} \left(|t_s - \widehat{t_s}| + |t_e - \widehat{t_e}| \right) \tag{5.18}$$

Note that, for each hotspot in the perturbed data, we calculate the hotspot distance to each real hotspot (for the same space-category granularity) and report the minimum value. This protects against cases in which there is not a one-to-one relationship between real and perturbed hotspots. Hotspots in $\widehat{\mathcal{H}}$ for which there is no corresponding hotspot in $\mathcal{H}$ are excluded. We also record the absolute difference between the maximum counts for each hotspot pair (i.e., $|c_{max} - \widehat{c}_{max}|$), which, when averaged across all hotspots, gives the average count difference, ACD.

Table 5.3 shows the AHD and ACD values under default settings. NGram is much better than other methods at 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, which is to be expected. In practice, preserving the spatio-temporal location of hotspots will probably be more important to policymakers and researchers than preserving the hotspot strength (i.e., the maximum number of unique visitors).

5.6 Discussion

Whereas the focus of this chapter has been on devising a general approach for sharing POI sequences, the n-gram-based solution can be adapted for specific applications, especially where aggregate statistics are important. 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 (as done in existing contact tracing apps [e.g., 24, 222]). Knowledge of such events can be used for location-specific announcements (e.g., as used in New Zealand [170]) and policy decisions (e.g., as used for local lockdowns in Victoria, Australia), as opposed to targeted advice for individuals (e.g., most contact tracing apps). Other applications include advertising and the 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, they may consider advertising near museums.

When deploying the solution, the mechanism parameters will need to be tuned according to the privacy and utility goals of the end user. One important parameter to set is the privacy budget, and the results from Section 5.5 suggest that setting $\epsilon \geq 1$ is a good starting point. Similarly, the extent to which STC region merging is conducted will also influence results, and the best approach to do so will be dependent on domain-/problem-specific circumstances. Additionally, the external knowledge that is used in Section 5.5 is limited to the data that is widely available, although the proposed solution can accommodate other data sources without difficulty. Incorporating more, richer data sources is also recommended when using the solution for specific applications. Doing so is likely to enhance utility, without negatively affecting efficiency; this idea is discussed in more depth in Section 7.3.

Finally, the problem framework and solution can also be applied to other notions of trajectory in space-time. To illustrate this, consider sharing shopping habits (e.g., credit card transactions). Here, Π represents the set of purchasable products, with attributes such as category, price, etc. Intrinsic hierarchies can be exploited 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.

Chapter 6

Sharing and Integrating Event Sequences from Multiple Sources with Local Differential Privacy

Chapter 5 presents an efficient and scalable mechanism for services to privately share trajectory data with high utility. However, the proposed solution, and other similar work [e.g., 64, 242], can only share data from each data source in isolation, and provides no functionality for data from multiple services to be integrated together. This means that services and end users only gain a partial picture of each individual in the population. For example, a transit agency knows when and where a user is travelling, but not why they are travelling or what they do at their destination. Moreover, in many cases, the same event is recorded by multiple services, but each service collects a different subset of data on the event. To illustrate this, consider the example trajectory data in Figure 6.1; this example is used throughout this chapter. Jane's trip to the cinema is recorded by three services (Bank B, Film Review Service C, and Location Service E), all of which collect a different set of attributes (some of which overlap). Having partial data like this is restrictive as it heavily limits the utility of the data and the range of analyses that can be conducted, as well as introducing the potential for missed (or incorrect) inferences and insights. For example, although Bank B could identify fraud by analysing the buying patterns of millions of users, this can still result in false positives and false negatives, both of which are undesirable for users. However, if users were willing to share perturbed data about their other activities, even at a coarser granularity, this can reduce false positives and false negatives and provide a better experience for users.

To address this limitation, this chapter introduces the problem of sharing and integrating event sequences among multiple services with local differential privacy guarantees, and under the control of the user. The importance of, and a potential application for, this new problem can be illustrated with Jane's trajectory (Figure 6.1f). An in-person purchase in Peru

(a) Shop A

Time	Amount	Items	Payment
7:36	£1.09	Apple	Cash
13:03	£107.94	Bread, Milk, ...	Gift Card
20:32	£27.99	Batteries	Online

(b) Bank B

Time	Amount	Location	Payment
7:51	£24.99	Peru	Card
9:15	£12.99	Cinema	Online
20:32	£27.99	Shop A	Online

(c) Film Review Site C

Time	Film	Genre	Rating
11:41	Casablanca	Drama	2
22:51	Skyfall	Action	10
23:17	GoldenEye	Action	8

(d) Bikeshare Operator D

Time	Station ID	Start/End
7:47	582	Start
7:59	27	End
13:10	136	Start

(e) Location Service E

Arrive	Depart	Location
7:00	7:30	Park
9:12	11:30	Cinema
12:00	13:05	Shop A

(f) Subset of Jane's trajectory, which contains a suspected fraudulent event (in red)

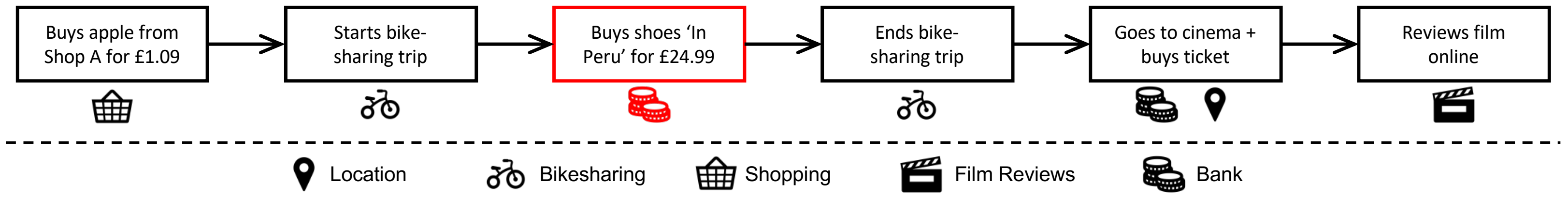


Figure 6.1: Example data collected by five services for one user (Jane) on one day (not all events shown in tables)

(in red) is attributed to Jane (who lives in the UK) and is recorded by the bank. With no other in-person transactions recorded, the bank has limited information to confirm whether the event is likely to be fraudulent. It may, therefore, decide not to block their card to prevent other fraudulent purchases. However, if Jane shares their privatised trajectory data from other services with the bank, the bank is likely to have sufficient evidence that Jane was not in Peru. As Jane has other in-person events in the UK that day, the bank can infer that the transaction is likely to be fraudulent, and so it can warn Jane, or block the card if the inference confidence is high enough.

There are two key challenges towards developing a privacy- and utility-enhancing solution for this new problem that are unique to the multiple source setting. The first challenge is to develop a data sharing protocol that gives the user control over their data, provides strong privacy, and achieves high practical utility. That is, how should the user share their data from the donating service with the receiving service, and how should the receiving service integrate this data with its own data. The second challenge is the task of integrating data from multiple services such that linked events are preserved across the dataset, which is fundamental in achieving high utility. In addition to these, all of the challenges from the single source sharing setting persist. These include the need to preserve correlations between consecutive events *and* between attributes of individual events, provide a strong level of privacy, and remain efficient and scalable for large datasets.

To address the first challenge, the user can allow services to utilise the raw data that they already collect (e.g., a bank needs to know the true transaction amount to execute the transfer of funds). As true data should not be shared between services in its original form, data from the donating service is perturbed locally on the user's device before it is shared, by the user, with the receiving service. The receiving service then integrates this privatised output with their own data with the user's consent. This practical and realistic setting enables all users to control what data they share and the purposes for which it is used.

The challenge of integrating data from multiple services can be solved by using bipartite matching-based optimisation to link semantically similar events. However, obtaining high utility when integrating data is dependent on achieving high utility when privately perturbing the trajectories of the donating service. Given its strong performance, the n-gram-based solution of Chapter 5 forms the basis for the perturbation mechanism in this solution. However, this work can be extended in two key ways to further improve the quality of perturbations and its overall practicality. First, using variable-length n-grams (as opposed to fixed-length n-grams) prevents the mechanism from trying to preserve correlations between consecutive events where no such correlation is likely to exist, whilst also utilising the privacy budget as best as possible. Second, the solution proposed in this chapter introduces a generic mechanism for incorporating popularity information that covers a spectrum of cases, where such information can be: (a) public knowledge, (b) collated from multiple services, (c) privately learned from data, or (d) unavailable entirely. This reflects the realistic setting in which services may not have complete and granular popularity data at their disposal.

Finally, as in Chapter 5, our perturbation mechanism uses a quality function that seamlessly combines semantic distance functions for the attributes being shared. As with the rest of this thesis, utility is boosted further by imposing viability constraints based on public knowledge to prevent nonsensical outputs. These include the reachability constraint introduced in the previous chapter, as well as other constraints, such as restrictions on the range and order of output values. Importantly, both the viability constraints and the semantic distance functions are adaptable to any domain setting, and this ensures that the overall mechanism is generalisable. This is important as, in the multiple service setting, services will wish to use data collected by services from other domains, and so any mechanism should accommodate the characteristics of event data from different domains (e.g., geospatial, social media, online shopping). This contrasts with Chapter 5, and other previous work [e.g., 110, 112], that focused on specific domains (e.g., the geospatial domain).

This chapter is structured as follows. First, recent relevant literature is surveyed in Section 6.1. This is followed by Sections 6.2 and 6.3, which outline the privacy setting and problem, respectively. Section 6.4 presents the proposed solution, which is complemented with theoretical analysis in Section 6.5. The solution is evaluated in Section 6.6, before the chapter concludes in Section 6.7 with a brief summary and discussion.

6.1 Related Work

This chapter uniquely combines three research areas – multi-dimensional data publication, multi-source data publication, and trajectory publication – to be the first to consider the problem of constructing private composite trajectories. Literature related to sharing private trajectories is surveyed in Section 2.2, and is not reviewed here. We review the other two research areas here, before summarising the limitations of existing work.

Multi-Dimensional Data Publication

The problem of publishing multi-dimensional data has been studied in several guises with centralised and local DP. For example, McKenna et al. [165, 166] seek to minimise the error for high-dimensional queries in the centralised setting. In the local setting, Cormode et al. [64] study the release of private marginals, and Ren et al. [196] examine the release of multi-dimensional crowdsourced data. Finally, Wang et al. [242] focus on publishing multi-dimensional data (from one source) with LDP for two analytical tasks – frequency estimation and machine learning-based tasks.

Multi-Source Data Publication

Using data from multiple sources (e.g., data owners) for a single (typically machine learning-based) goal is not new in the DP domain with secure multi-party computation and federated learning being notable fields in DP literature [e.g., 33, 133, 187, 209, 224]. However, these works make no attempt to publish data in the same format as the input data (i.e., they publish aggregate statistics or a model). Even when a generative model is published and data is synthesised using it, the link between users and their data is not preserved in current solutions, which limits their downstream utility.

Multi-party data publication has also attracted focus recently. Su et al. [208] (and the related follow-up work [56]) study it in the centralised setting. However, their setting is fundamentally different as it assumes that the parties share data covering the same attributes

Table 6.1: Summary of related work limitations

Reference	Local Setting	Multiple Service	Multiple Attribute	Trajectory Data	Same I/O Format
Wang et al. [242]	✓	✗	✓	✗	✓
Xu et al. [257]	✓	✓	✓	✗	✗
Tang et al. [213]	✗	✓	✗	✓	✗
Tang et al. [214]	✗	✗	✓	✗	✗
Chapter 5	✓	✗	✓	✓	✓

for disjoint user sets (e.g., three hospitals sharing patient data) with the aim of learning aggregate characteristics of the population (without providing tangible benefits to individual users). Tang et al. [213] extend this work by proposing a cryptographic, tree-based approach for multi-party trajectory release. Tang et al. [214] and Mohammed et al. [171] both focus on publishing vertically partitioned data by implementing a latent tree model-based solution and a cryptographic two-party protocol for the exponential mechanism, respectively. Similarly, private record linkage has also been used to tackle multi-party data sharing problems in the centralised setting [35, 120, 195]. He et al. [120] define the notion of output-constrained DP for the two-party sharing setting, in which data is shared between parties, and Rao et al. [195] extend this to include a third party.

There has been some recent research with respect to multi-source data release in the local setting. Xu et al. [257] answer aggregate queries using data from multiple sources by introducing the notion of user-level LDP, which requires LDP to be satisfied for each user, for each service, even when users provide different amounts of data. The authors only address the settings in which users share one piece of data with each service (i.e., single events), or users share multiple events (i.e., trajectories) with only one service. This work, however, focuses on the more general problem in which users can share trajectories with *both* services. This setting makes the integration of data between the two services a non-trivial operation that cannot be solved optimally by simply considering the (semantic) distance between events in the shared data.

Limitations

All of the previously cited work suffers from one (or more) of the following limitations: not using a local privacy setting, not considering multi-service sharing, not releasing multi-attributed data, not focusing on trajectory data, and not publishing data in the same format as the input data. Table 6.1 highlights the limitations of the existing work that is most similar to the problem in this chapter. All other cited work is less related, and still possesses many of these (or other) limitations.

Addressing these limitations with extensions of (most of) these works are either unfeasible or non-trivial. For example, changing from a centralised privacy setting to a local one fundamentally changes the nature of the problem, and means that existing solutions, which have been specifically designed for centralised DP, can no longer be applied. Similarly, works that do not consider sequence data (i.e., trajectories) provide no mechanism for protecting or preserving the correlations that exist between data points. And, as noted above, integrating data between two services is a more challenging operation when each service has multiple events. However, as the perturbation and integration of data are sequential steps, the perturbation mechanism proposed in Chapter 5 serves as the basis for a combined framework onto which a solution for integration can be added. This chapter's solution presents this combined framework and, therefore, becomes the first work to output locally private multi-attributed, multi-source trajectories in the same format as the input data.

6.2 Privacy Model

Although centralised DP typically involves less noise than LDP, it is inherently unsuitable for this problem. Most existing work (or possible extensions thereof) [e.g., 52, 54, 119, 153, 274] generate representative trajectories synthetically, based on noisy aggregate statistics. This is acceptable in many settings, but it is unsuitable here as it breaks the link between a trajectory and user ID, which is important when integrating data from different services. Conversely, a fully local setting, in which users only share perturbed data with services is unrealistic as each service will need to know some true data (e.g., to execute a transaction).

Instead, we can use a hybrid privacy setting in which the user is at the heart of the data sharing protocol. First, users have access to all of their original data. Each individual service has access to each user's original data, which they collect ordinarily. This ensures that information is recorded correctly, transactions are executed, etc. Users can decide what (additional) data from their holistic trajectory they are willing to give to services in a privacy-preserving manner. Any additional data that is provided is perturbed by the user using LDP, and it is perturbed on their device. Services can combine their previously-collected original state data with the additional LDP-perturbed data for enhanced utility. It is assumed that users 'opt-in' to this privacy-preserving data sharing protocol, and that all services are honest and do not collude with each other.

This privacy setting ensures that strict ϵ-LDP is satisfied throughout, whilst also providing a clear privacy boundary for users. This focus on preventing true data transfer beyond each (necessary) user-service relationship also ensures that users have full control over, and knowledge of, the extent to which their true data is shared. Finally, by enabling the receiving service to utilise true data, which has already been collected through necessity, it is possible to model a more realistic and practical data-sharing scenario than the common approach for using LDP.

6.3 Problem Outline

This section introduces notation, defines key terms, and formally outlines the composite trajectory problem. It also discusses the assumptions and constraints that scope the problem, and enhance the output data's utility.

6.3.1 Preliminaries

Consider a set of services $\Sigma = \{\sigma_1, \sigma_2, ..., \sigma_i\}$ in which each service σ_i (e.g., bank, shop, social network) records a user's interactions with the service (also known as events). Each user u_j is part of the user set $\mathcal{U} = \{u_1, u_2, ..., u_j\}$. Each event, z, is characterised by several attributes, which include location, transaction amount, items purchased, duration of visit, etc. When a user shares the data that σ_1 collects with σ_2, all events and attribute values are assumed to be *sensitive*. However, the set of attributes that σ_i collects (i.e., the names of the attributes) is considered to be *non-sensitive* and public knowledge (i.e., everyone knows that a bank collects transaction amounts, etc.). We use $\mathcal{A}_i$ to denote this set.

Events are recorded in a time-ordered sequence (i.e., a trajectory), and the trajectory for user u_j using service σ_i is denoted as $\tau_{ij} = \{z_{ij1}, ..., z_{ijk}, ..., z_{ij|\tau|}\}$. The set of trajectories recorded by σ_i is denoted as $\mathcal{T}_i$. As in Chapter 5, $\tau(a, b)$ denotes a fragment of τ where a and b denote the bounds of the fragment. For example, $\tau(1, 3)$ represents the first three events in τ. Hat notation denotes perturbed entities (e.g., $\widehat{z}_{ijk}$ denotes the k^{th} perturbed event of user u_j with service σ_i). For clarity, subscripts for variables are sometimes omitted when their exclusion is not integral to understanding.

Holistic Trajectories

$\overline{\tau}_j$ denotes the holistic trajectory for u_j, which is the time-ordered sequence of *every* interaction a user has with *any* service in Σ (i.e., all events for u_j). In non-trivial cases, $\overline{\tau}_j$ is only known in its entirety by u_j. However, each service σ_i will know a subset of $\overline{\tau}_j$ such that $\tau_{ij} \subseteq \overline{\tau}_j$. And, for each $z_{ijk} \in \tau_{ij}$, the service may have partial or full knowledge of the attributes of z_{ijk} represented in $\overline{\tau}_j$. That is, if $\overline{\tau}_j$ contains the time, location, and transaction amount, z_{ijk} may contain all these attributes (full knowledge) or just, say, the time and location (partial knowledge).

Linked Events

A linked event is one that ‘appears’ in the trajectories of two or more services. To definitively link events, there must be at least one overlapping attribute for the event between each pair of services. In the example, there is a linked event as the same time and transaction amount is recorded by Shop A and Bank B. Different services may have different knowledge of a linked event. For instance, both Shop A and Bank B know the transaction amount, but only Shop A knows the items purchased (i.e., batteries).

6.3.2 Problem Specification

With these preliminaries outlined, we can formally define the problem studied in this chapter. We can also discuss variants of the problem, which the proposed solution accommodates.

Consider two services $\sigma_1, \sigma_2 \in \Sigma$, and the users $u_j \in \mathcal{U}$. Each $u_j \in \mathcal{U}$ has an attributed trajectory τ_{ij} with each service, consisting of events recorded by that service. σ_1 and σ_2 have data for the attribute sets $\mathcal{A}_1$ and $\mathcal{A}_2$, respectively. $\mathcal{A}_{1\to 2}$ is the set of attributes that is shared by the donating service, via the user, with the receiving service. σ_2 integrates this data with their own data, τ_{2j}, and compiles the composite trajectory τ_j^* for each user. Henceforth, σ_1 is referred to as the *donating* service, and σ_2 is the *receiving* service.

The problem of this chapter is two-fold. First, given the unperturbed trajectory τ_{1j}, the user must perturb this trajectory in accordance with LDP. This perturbed trajectory, $\widehat{\tau}_{1j}$, is then shared by the user with the receiving service. Second, given the perturbed trajectory $\widehat{\tau}_{1j}$, the receiving service then seeks to integrate this data with its own unperturbed data on the user, τ_{2j}, to form τ_j^*, with the aim of preserving the linked events as much as possible.

This problem definition captures many possible variants of the problem. For example, in many practical situations, a service will not need to obtain all perturbed attributes from another service. In the example, Location Service E may want to know the transaction amount, but they may have no interest in learning what item was purchased. Similarly, in some cases, services may not record every event of a user, even if they could have done. For example, in Figure 6.1, Jane's visit to Shop A at 7:36 is not recorded by Location Service E. This may be the case if the service collects data 'actively' (i.e., users must check-in) rather than 'passively' (i.e., locations are recorded automatically), and Jane does not check-in. Importantly, incorporating these variants requires no changes to the mechanism.

Threat Model

As this problem explicitly relies on a user sharing their (potentially overlapping) data with other services, the most notable adversarial threat is a linkage attack. The most likely linkage attack is where a rogue receiving service uses the true data it has on a user to determine information about a user from the donating service's data. However, our privacy model prevents against this attack by ensuring that the only data (from another service) that a user provides with a receiving service is perturbed using LDP. The plausible deniability provided to users through LDP prevents the receiving service from definitively determining knowledge about any event in the donating service's data, thus thwarting any linkage attack. Additionally, as this problem is an extension of Chapter 5, it is intuitive to see that the same adversarial threats outlined in Section 5.1.1 remain relevant here. Importantly, however, the protections against these threats (provided by the solution presented in Chapter 5) persist.

6.3.3 Constraints

Once again, hard viability constraints, informed by external knowledge, can be imposed to enhance utility and ensure realism in the output data. These constraints are primarily enforced when defining n-gram sets, as detailed in Section 6.4, although post-processing ensures that the output data reflects these constraints. The precise constraints are domain-specific, but they can be generalised into four categories.

1 – Feasibility. This constraint uses external knowledge to ensure that an event in the output data aligns with common sense/real-world observations. For example, business opening hours can be used to ensure that the time-location pair for any event does not coincide with the times that a business is closed (e.g., being at a church at 3am on a Tuesday would be unfeasible).

2 – Reachability. This location-specific constraint, introduced as Definition 4 in Section 5.1.3, ensures that consecutive locations can be reached given the corresponding time gap in the trajectory. Reachability also implies that a person cannot be in two places at once and, in most cases, they cannot reasonably be doing two things at once.

3 – Ordering. Certain activities (or combinations thereof) can be subject to 'ordering' constraints. For example, one cannot return a product to a store before buying it. Similarly, one cannot exit a station without entering one.

4 – Output Domain. Output domain constraints impose limits on the values that are feasible. These could be minimum or maximum bounds (e.g., buying a new car will mean spending at least, say, £7,000), or simple restrictions on whether a value can exist (e.g., ATM withdrawals might only be possible in multiples of £10). Such limits can be specified by the service, learned from data, or obtained from public information. For example, maps can be used to define geographic regions in which no events can occur (as in Chapter 3).

Importantly, constraints are only informed using public knowledge (i.e., they do not rely on the values, distributions, or characteristics of the sensitive data). Where real data does not satisfy any imposed constraints (e.g., Jane travels between two locations sooner than reachability dictates, or Brian buys a new car for £1,000), this data can be excluded as an outlier. This means that there is no privacy risk from imposing the constraints as any input data is known to satisfy the constraints, which are, by definition, reflective of the real-world and therefore non-sensitive.

6.4 Solution

The solution in this chapter augments the work of Chapter 5 to perturb variable-length n-grams in a multi-attributed two-level space. It comprises four stages: incorporation of popularity data, n-gram perturbation, trajectory reconstruction, and integration of data between services.

6.4.1 Challenges and Design Details

This section discusses the key challenges in addressing the problem, alongside detail on the design choices used to combat these challenges.

Preserving Correlations Between Events. One characteristic we must preserve is the correlation that exists between adjacent events in time. For example, people are more likely to travel from their office to a nearby sandwich shop at lunchtime (as opposed to a sandwich shop across the city), before heading back to their office. As demonstrated in Chapter 5, fixed-length n-grams allow consecutive events to be perturbed together, which increases the likelihood that semantic links between them are maintained through perturbation. By using variable-length n-grams (as opposed to fixed-length n-grams), we seek to only preserve correlations between events close together in time (i.e., those likely to be correlated in reality), which prevents uncorrelated consecutive events from being perturbed together. By again using *overlapping* n-grams, the coverage of the event and its neighbouring events can be maximised, which helps to preserve the spatio-temporal links between events.

Preserving Correlations Between Attributes. Another characteristic to preserve is the correlation that exists between attributes within the same event. For example, people may have low spends in convenience stores, but high spends in luxury department stores. To preserve this, n-grams are composed within a multi-attributed space. This means that, instead of perturbing attributes independently, attributes are perturbed together through multi-attributed regions. An example region might be [7–8am, £0–10, Shop A] – this represents Jane's purchase of an apple in Figure 6.1.

Integrating Perturbed Data. If two or more services contain the same attributes of a linked event, the perturbed data is unlikely to match, which may give the perception that the data relates to different events. A process is needed to (try to) re-identify linked events using the donating service's perturbed data and the receiving service's true data. An integration stage, based on bipartite matching between events in the two services' data, aims to re-identify linked events and prevent the same event from appearing more than once in the output composite trajectory.

Balancing Efficiency and Utility. Many attributes (e.g., time, price, location) are continuous, and it is necessary to discretise them into bins, with a fine discretisation granularity better for utility. However, Equation 2.5 implies that smaller domain sets (obtained with a coarser granularity) lead to higher utility in perturbation. Moreover, large domain sets can be costly to compute (time-wise) and store (memory-wise), which can make the overall mechanism inefficient or even unviable (as seen with the global solution in Chapter 5). To balance this efficiency-utility trade-off, the proposed solution uses a two-level decomposition of multi-attributed space. Specifically, we discretise attributes and store popularity information at a finer granularity (called the lower level), and perform perturbations at a coarser granularity (called the upper level).

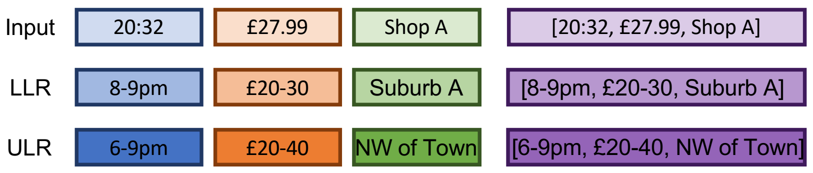
6.4.2 Preliminaries

Region Decomposition

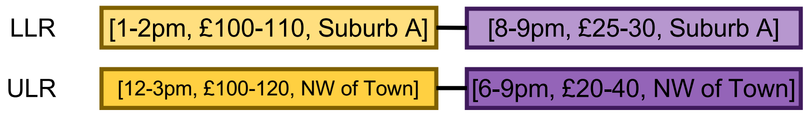
At the lower level, each attribute of τ is discretised, and $r_{\chi ijk}$ denotes the lower level region (LLR) of attribute χ for the k^{th} event of u_j, as recorded by σ_i. Multi-attributed LLRs are composed by combining attribute LLRs, with r_z used to denote these regions. For example, if z = [8:32pm, £27.99, Shop A], the corresponding attribute LLRs might be r_{time} = (8pm, 9pm), r_{amount} = (£20, £30), and r_{loc} = Shop A. Hence, r_z = [8–9pm, £20–30, Shop A]. The set of all LLRs for attribute χ is $\mathcal{RL}_{\chi}$, and the set of multi-attributed LLRs for attribute set $\mathcal{A}$ is $\mathcal{RL}_{\mathcal{A}}$.

Upper level regions (ULRs) are constructed by merging LLRs according to the bounds of a coarser granularity. They are denoted as R_{Aijk}, or R_z if referring to a specific event z. Example ULRs might be: R_{time} = (6pm, 9pm), R_{amount} = (£20, £40), and R_{loc} = NW of town. Hence, the multi-dimensional ULR would be R_z = [6–9pm, £20–40, NW of Town]. The set of all ULRs for attribute χ is $\mathcal{RU}_{\chi}$ and the set of multi-attributed ULRs for attribute set $\mathcal{A}$ is $\mathcal{RU}_{\mathcal{A}}$. Figure 6.2a shows how an event from Figure 6.1 is translated to an LLR and ULR.

As in Chapter 5, attributes can be decomposed hierarchically, exploiting intrinsic hierarchies where possible (e.g., POI categories: 'food + drink' > 'restaurant' > 'Italian' > 'pizza'). Whereas attributes can be decomposed into several hierarchical levels, only two levels are necessary for this mechanism. The granularity (for constructing LLRs and ULRs) is informed by the distribution of attribute values, the domain size of each attribute, the downstream use for the collected data, etc. If the distribution of attribute data is skewed, region bins



(a) LLRs and ULRs



LLR
[1-2pm, £100-110, Suburb A]
[8-9pm, £25-30, Suburb A]
ULR
[12-3pm, £100-120, NW of Town]
[6-9pm, £20-40, NW of Town]

(b) LLR and ULR *n*-grams

Figure 6.2: Illustration of how LLRs and ULRs are formed

can be non-uniform; otherwise, uniform bins suffice. If domain sizes or the number of attributes is small, finer granularities can be used to benefit utility. To ensure consistency when translating between LLRs and ULRs, the boundaries of ULRs should overlap those of LLRs. For example, if LLRs are defined 'on the hour' (i.e., boundaries are of the form XX:00), ULRs are also defined to be 'on the hour' and not, say, 10:00-11:30.

LLRs and ULRs are also used to construct n-grams. Lower level n-grams are denoted as $w_k^n = \{r_k, r_{k+1}, ..., r_{k+n-1}\}$. Similarly, n-grams that use ULRs are denoted as $W_k^n = \{R_k, R_{k+1}, ..., R_{k+n-1}\}$. The sets of n-grams for the lower and upper levels are $\mathcal{WL}^n$ and $\mathcal{WU}^n$, respectively. Figure 6.2b shows LLR and ULR n-grams for data from Figure 6.1.

Semantic Distance Functions

This mechanism relies on the notion of semantic similarity in both perturbation and integration. As in Chapter 5, these semantic distance functions incorporate public external knowledge such that more semantically similar outcomes are more likely to be output. In the example, Jane's location at 9:12 (the cinema) is more likely to be perturbed to a theatre than a football ground. Similarly, a high-value transaction in a jewellery store is more likely to be perturbed to a high-value transaction in a luxury shoe shop than a corner shop. These functions operate at any granularity (e.g., n-gram, single values), and to ensure that $\Delta_{d_\mathcal{X}} = 1$ and $\Delta_q = 1$, they are normalised to have values in the range (0,1). Any distance function can be used, although the ones used in this chapter are outlined here.

Although locations could be treated as numerical values (by using their co-ordinates), they typically have rich semantic information associated with them, as utilised in Chapter 5. Associating this semantic information with users (e.g., Jane likes to shop at luxury shoe shops) is valuable, and so it is advantageous to incorporate it into the semantic distance functions so that utility can be increased. These attributes are also linked to one another, which means that perturbing them independently is sub-optimal. Hence, locations are perturbed using a semantic distance function that combines two location-specific attributes:

$$d_{loc}(x, y) = \sqrt{\omega_{PD} d_{PD}(x, y)^2 + \omega_{CD} d_{CD}(x, y)^2} \tag{6.1}$$

where $d_{PD}(x, y)$ and $d_{CD}(x, y)$ are the physical and category distances between the two locations, respectively. The Haversine function is used for physical distances, and the category distance function from Chapter 5 is also used. For simplicity, we set $\omega_{PD} = \omega_{CD} = 0.5$ as they have the same normalised magnitude (i.e., $\Delta_{\mathcal{X}} = 1$).

For all other numerical attributes, the distance between values is simply defined as: $d_{\mathcal{X}}(x, y) = |x - y|$. For hierarchically structured categorical variables, we use a distance function, like the one outlined in Figure 5.4. For other categorical attributes (e.g., method of payment), all values are modelled to be equally distant from each other such that $d_{\mathcal{X}}(x, y) = 0$ if $x = y$, and $d_{\mathcal{X}}(x, y) = 1$ otherwise.

Pre-Processing

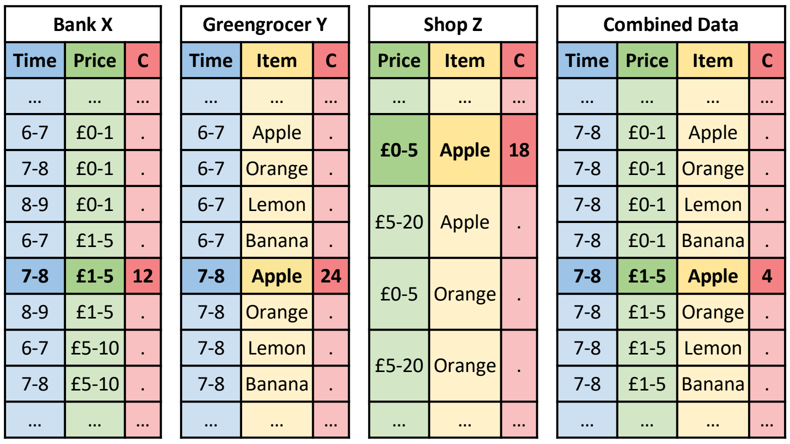
Several pre-processing steps are necessary. First, the set $\mathcal{A}_{1\to 2}$ is defined and communicated to the user and services. Second, the LLR and ULR sets are created. The value sets and the bounds for each attribute are private, although the names of the attributes collected are considered to be public. Trajectory data for each user is then collected and translated to both LLR and ULR format. The pairwise semantic distance matrices for $\mathcal{RL}$ and $\mathcal{RU}$, the relevant n-gram sets, and the pairwise semantic distance matrices for these n-gram sets are also pre-computed. Finally, utility constraints are imposed to remove nonsensical n-grams (e.g., those that fail to satisfy reachability, or those with businesses that are closed).

6.4.3 Incorporating Popularity

This section outlines a generalised solution for including popularity information, depending on the availability of such information. Popularity is incorporated into the mechanism through a feasibility constraint such that n-grams that appear less often than a specified threshold are viewed as unfeasible and removed. Initial experiments showed that setting the threshold to be 5% of the maximum count in an n-gram set offered good, robust results.

Public Popularity. In some cases, popularity data on all attributes being shared might be freely available in the public domain (e.g., census data, Google's Community Mobility Reports [105]). Where this is the case, this data can be pre-processed to the LLR and ULR granularities and incorporated directly.

Multi-Service Popularity. In most realistic settings, no one service will have full popularity knowledge for all shared attributes. However, this information may exist partially across multiple services. This potentially allows for popularity information to be crowdsourced from several other services, including those not involved in the data sharing protocol. For example, if $\mathcal{A}_{1\to 2} = \{\text{time, location, transaction amount}\}$, one service might provide highly granular spatio-temporal data, whereas another might provide data on how transaction amounts vary with time. This popularity data is converted to LLR and ULR granularities, using inference and aggregation where necessary. Similar techniques are used when multiple services provide data at different granularities. It is reasonable to assume that any popularity information provided here does not need to be privatised by our mechanism. That is, it is either already public knowledge and/or it has been sufficiently privatised by a service (e.g.,



Bank X		
Time	Price	C
...	...	...
6-7	£0-1	.
7-8	£0-1	.
8-9	£0-1	.
6-7	£1-5	.
7-8	**£1-5**	**12**
8-9	£1-5	.
6-7	£5-10	.
7-8	£5-10	.
...	...	...

Greengrocer Y		
Time	Item	C
...	...	...
6-7	Apple	.
6-7	Orange	.
6-7	Lemon	.
6-7	Banana	.
7-8	**Apple**	**24**
7-8	Orange	.
7-8	Lemon	.
7-8	Banana	.
...	...	...

Shop Z		
Price	Item	C
...	...	...
£0-5	**Apple**	**18**
£5-20	Apple	.
£0-5	Orange	.
£5-20	Orange	.
...	...	...

Combined Data			
Time	Price	Item	C
...	...	...	...
7-8	£0-1	Apple	.
7-8	£0-1	Orange	.
7-8	£0-1	Lemon	.
7-8	£0-1	Banana	.
7-8	**£1-5**	**Apple**	**4**
7-8	£1-5	Orange	.
7-8	£1-5	Lemon	.
7-8	£1-5	Banana	.
...	...	...	...

Figure 6.3: Example showing calculation of popularity data from multiple services

through the addition of DP-compliant noise). As such, there is no need to expend any of the privacy budget when using this information.

Figure 6.3 shows an example of this process. Three services collect data, with each service collecting popularity data for two of the three attributes – time, price, and item. There are three time bins at the LLR level – (6–7am), (7–8am), and (8–9am) – that are contained within one ULR: (6–9am). Similarly, there are four (uneven) price bins at the LLR level – (£0–1), (£1–5), (£5–10), and (£10–20) – that form two ULRs: (£0–5) and (£5–20). Finally, there are four items – apple, orange, lemon, and banana – each of which exist in their own LLR and ULR. The aim is to model popularity information for time-price-item LLRs by using data from all three services, each of which is given equal weight. The bold rows in Figure 6.3 relate to Jane's apple purchase in Figure 6.1. The bank collects time-price data at the LLR level and so, as there are four item bins, its contribution towards the total is divided evenly by four. Similarly, the greengrocer's contribution is divided by four as there are four price bins. Shop Z only has data at the ULR level, and so two steps are needed to obtain its contribution. First, its count is translated to the LLR level by dividing the count by two (i.e., to divide between the two LLRs that comprise the ULR). This count is then divided by three to cover the three possible time bins. As each service is given equal weighting, the modelled count for the LLR [7–8am, £1–5, Apple] is: $\frac{1}{3}(\frac{12}{4} + \frac{24}{4} + \frac{18}{2\times3}) = 1 + 2 + 1 = 4$.

Private Popularity. In some cases, no popularity data may exist or, if it does, it may not be sufficiently representative of the population being studied. If this is the case, input data can be utilised to obtain a private popularity distribution of the population.

A naïve way to privatise the data is to add noise to count data (e.g., by using Laplace noise), but this weakens the overall privacy guarantee to one based on the centralised model. Instead, a fraction of the privacy budget can be used to perform a set of dummy perturbations to obtain private popularity information. These dummy perturbations use the same perturbation model described in Section 6.4.4, and can be performed at the LLR or ULR level. Once the dummy perturbations are complete, counts are obtained for LLRs and ULRs: the noisy count for each ULR is $\widehat{C}_R = \frac{1}{n}\sum_{r_i \in R} \widehat{c}_{r_i}$, where $\widehat{C}_R$ is the noisy count for R and $\widehat{c}_{r_i}$ is the number of times $r_i \in R$ appears in the perturbed data. Note that $\widehat{C}_R$ is divided by n to ensure that regions are not double-counted, owing to the dummy n-gram perturbations.

The advantage of using private popularity data in this way is that it should better reflect any specific short-term trends exhibited by the population being studied that may not be replicated in the population at large. However, using a fraction of the privacy budget can adversely affect the accuracy of the 'real' perturbations, with consequential downstream effects on the output data's utility.

No Popularity Information. Where no popularity information is available, or dummy perturbations are not performed, the n-gram sets remains unchanged and data sharing proceeds as normal.

6.4.4 Perturbation

Perturbations are conducted using variable-length n-grams that seek to only preserve correlations between events close together in time (i.e., those likely to be correlated in reality). This decision can be justified and illustrated with a simple example (see Figure 6.4). Consider six events occurring across the day: the first two occur in the morning, the third around lunchtime, and the final three in the late evening. Intuitively, one can infer that z_1 and z_2 are likely to be correlated events, whereas z_3 and z_4 are unlikely to share a correlation. Using fixed bigrams would seek to preserve a correlation between z_3 and z_4, even though one is unlikely to exist. Hence, we split the trajectory into discrete event periods. Each event period is then modelled as a mini-trajectory and perturbed using fixed-length overlapping n-grams.

Event Periods

There are several ways to determine event periods; two are presented here, although others can be incorporated easily. The first uses a simple binary condition in which two consecutive events, z_k and z_{k+1}, are in the same event period if $t_{k+1} - t_k \leq t_{gap}$, where t_{gap} is some fixed time gap. However, this approach is crude, and it fails to reflect the likely patterns that are observable from true data.

A more advanced approach is to determine event periods probabilistically. The exponential distribution is ideal in this case as it can ensure that events closer together in time are more likely to be part of the same event period than events with a larger time gap. A random biased coin flip can decide whether two events are in the same event period, where the probabilities attached to the coin are influenced by the probability distribution function of the exponential distribution. Specifically, the probability that two events z_k and z_{k+1} (with times t_k and t_{k+1} respectively) are in the same event period is:

$$\Pr[z_k \text{ and } z_{k+1} \text{ are in the same event period}] = \lambda e^{-\lambda(t_{k+1}-t_k)} \tag{6.2}$$

where λ is a parameter of the exponential distribution. The cumulative distribution function of the exponential distribution can be used to set this parameter such that $\lambda = -\frac{\ln(1-F)}{t_{gap}}$, where F is the value of the cumulative distribution function. For example, setting $F = 0.9$ and $t_{gap} = 2$ hours means that the probability of z_k and z_{k+1} being in the same event period when they are separated by two hours is 0.1. Spatial information can also be included in determining event periods, if desired. Figure 6.4 shows the perturbations that are conducted when using unigrams, bigrams, and variable-length n-grams.

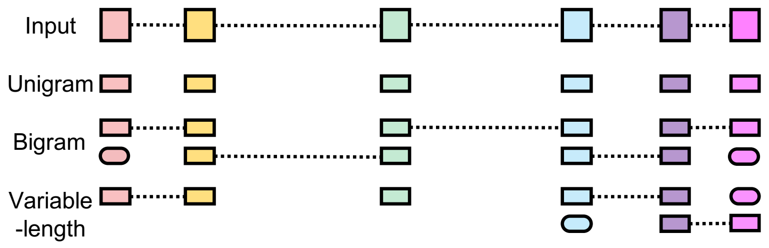


Figure 6.4: Illustration of perturbations needed when using unigrams, bigrams, and variable-length n-grams; rounded rectangles represent supplementary perturbations

Perturbation Model

The perturbation model used to perturb a service's data is similar to the one introduced in Chapter 5. Whereas Equation 5.12 was specific to perturbing POIs, the quality function used here is more generalised, and can cater for any number of attributes. It is defined as:

$$q(x,y) = -d(x,y) = -\sqrt{\sum_{\mathcal{X}_i} \omega_{\mathcal{X}_i} \left(d_{\mathcal{X}_i}(x,y) \right)^2} \tag{6.3}$$

where $x \in \mathcal{X}$ is the input datum, $y \in \mathcal{Y}$ is the output datum, and ω_i are non-negative weights such that $\sum_i \omega_i = 1$.

Perturbations are conducted at the ULR level as $|\mathcal{RU}| \ll |\mathcal{RL}|$, and so the perturbation probabilities of the exponential mechanism are less distorted by a smaller domain. The smaller domain size also maintains efficiency. The probability that $\tau(a,b)$ is perturbed to $W_i \in \mathcal{WU}^n$ is:

$$\Pr(\widehat{W}_k^n = W_i^n) = \frac{\exp\left(\epsilon' q(\tau(a,b), W_i^n)/2\Delta_q\right)}{\sum_{W_i^n \in \mathcal{WU}^n} \exp\left(\epsilon' q(\tau(a,b), W_i^n)/2\Delta_q\right)} \tag{6.4}$$

Perturbation continues for increasing values of a and b for each event period. We set $\epsilon' = \frac{\epsilon}{N_p}$.

As in Chapter 5, when $n > 1$, some events at the beginning and end of an event period will be perturbed an uneven number of times. To ensure that each event in an event period is perturbed equally, the same supplementary perturbation using shorter n-grams is implemented. Figure 6.4 illustrates the supplementary perturbation process, and when it is necessary (i.e., lone unigrams at the end of the bigram trajectory).

However, splitting trajectories into event periods can mean that events are perturbed unevenly at the trajectory level. For example, in Figure 6.4, the first three events are perturbed once each, and the remaining events are perturbed twice each. This is advantageous as it minimises the number of perturbations and maximises ϵ' (as we divide the overall privacy budget equally). This approach is not possible when using fixed-length n-grams as there is a necessity for n-grams to overlap along the entire trajectory. Importantly, this approach does not affect the overall privacy guarantee as we provide privacy at the trajectory level, as opposed to the event level. An alternative solution would be to perform additional perturbations such that each event is perturbed an equal number of times. However, this results in redundant perturbations that provide very little additional information as they

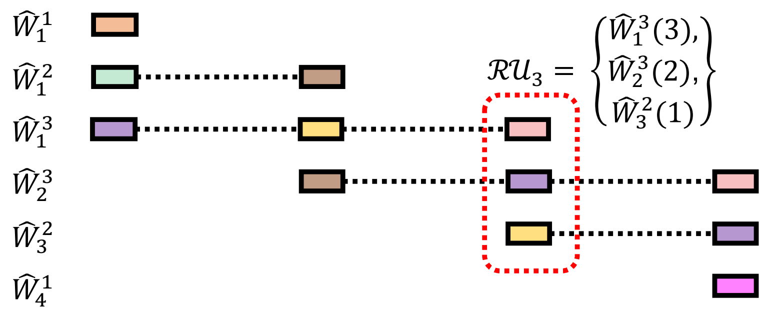

$\widehat{W}_1^1$
$\widehat{W}_1^2$
$\widehat{W}_1^3$
$\widehat{W}_2^3$
$\widehat{W}_3^2$
$\widehat{W}_4^1$
$\mathcal{RU}_3 = \left\{ \begin{matrix} \widehat{W}_1^3(3), \\ \widehat{W}_2^3(2), \\ \widehat{W}_3^2(1) \end{matrix} \right\}$

Figure 6.5: Formation of $\mathcal{RU}_k$ during reconstruction

are not overlapping n-grams. Furthermore, as this increases the total number of perturbations, the privacy budget has to be divided more, which impacts the overall utility of each perturbation (see Equation 2.5, and Section 6.5).

6.4.5 LLR Reconstruction

Once a service has perturbed data at the ULR level, up to four post-processing steps are necessary. As these are post-processing operations, the privacy guarantee is unaffected.

When supplementary perturbations have been performed and an event has been perturbed multiple times, the perturbed data must be consolidated such that each event only appears once. In Chapter 5, a linear programming-based solution was used and shown to be effective, if slow. Here, an indexing-based solution is implemented, which promises faster runtimes with little accuracy loss. To obtain $\widehat{\tau}(k,k)$, the ULR that is the minimum total semantic distance from ULRs in $\mathcal{RU}_k$ is found (ties are broken arbitrarily). $\mathcal{RU}_k$ is the set that contains the ULRs from all perturbed n-grams that contain the k^{th} event, as illustrated in Figure 6.5. Formally:

$$\widehat{\tau}(k,k) = \underset{R_i \in \mathcal{RU}}{\arg\min} \sum_{R_j \in \mathcal{RU}_k} d(R_i, R_j) \tag{6.5}$$

The trajectory is then translated back to the LLR level using weighted sampling, with weights determined using LLR popularity information, if available. For example, suppose that R can be decomposed into r_1, r_2, and r_3, and the noisy counts for the LLRs are 7, 2, and 1, respectively. If $\widehat{\tau}(k,k) = R$, it can be translated to the LLR level by sampling from the distribution with probabilities: $p_1 = 0.7$, $p_2 = 0.2$, and $p_3 = 0.1$. This method seeks to preserve the popularity distribution across LLRs within a ULR, as intricate correlations could be lost if uniform sampling is implemented. Uniform sampling is used, however, when no popularity information is available at the LLR level.

During perturbation, the times of events might be perturbed such that they are no longer in chronological order. A 'time smoothing' operation can rectify this such that the data provided to the receiving service has events in the same order in which they occurred. This operation minimally modifies the perturbed times until they are in chronological order, as outlined first in Chapter 5.

Where locations are shared, a final reachability check is made. In the unlikely event that the output trajectory violates the reachability constraint, localised updates to the time or location attributes in $\widehat{\tau}$ are performed such that the semantic distance between the initial and final output is minimised.

6.4.6 Integration

The final stage of the mechanism enables the receiving service to integrate the perturbed data from the donating service with their existing data. This section presents efficient bipartite matching procedures that link events based on their semantic similarity. The basic premise is that records with low semantic distance are likely to characterise the same event, and so they can be joined to give a more complete picture of the event.

Identifying Linked Events

In identifying linked events, the chronological order of the donating service's data should be preserved. This can be imposed with a hard constraint in which events cannot be linked (even if there is high semantic similarity) if it would mean that the chronological order of the donating service's data was not preserved. Alternatively, a softer constraint can be imposed in which events can be matched 'out-of-order' but such matches are penalised.

Set-Up. First, the pairwise semantic distances between each event in τ_{2j} and each event in $\widehat{\tau}_{1j}$ are found, using a semantic distance function defined for the attributes common to both datasets (i.e., $\mathcal{A}_2 \cap \mathcal{A}_{1\to 2}$). We define the bipartite graph $\mathcal{B}(\widehat{\tau}_{1j}, \tau_{2j}, \mathcal{E})$ in which the events in the two trajectories act as nodes, and the edge set $\mathcal{E}$ is defined as follows. An edge exists between two events if $d(\widehat{z}_{1jk}, z_{2jk}) \leq \Upsilon$, where Υ is a distance threshold that prevents events that are semantically dissimilar (e.g., an event at 7:38 and an event at 20:32) from being linked. $e_{a\to b}$ denotes an edge linking the a^{th} event in $\widehat{\tau}_1$ and the b^{th} event in τ_2. Some events (in either trajectory) can have a node degree of zero, which happens if an event is only recorded by one of the two services (e.g., Jane's cash transaction will not appear in Bank B's data). It also happens when the semantic distances between an event in $\widehat{\tau}_1$ and all events in τ_2 are greater than Υ, or vice versa.

Greedy Matching. A naïve approach is to select edges greedily. In this case, the edge with the lowest semantic distance is selected. Once this is done, edges with the same start or end node are removed, as well as edges that 'cross-over' the selected edge. Of the remaining edges, the edge with the lowest semantic distance is then selected. This process continues until no edges remain to be selected.

Strict Bipartite Matching. The greedy approach risks getting stuck in local optima, and so we propose a global optimisation problem. Although we seek to minimise semantic distance, we define the problem as a maximisation problem so that edge selection is encouraged. In most cases, both $|\widehat{\tau}_1|$ and $|\tau_2|$ are small, which means that the optimisation problem can be solved optimally by linear programming solvers. The problem is defined as:

$$\max \sum_{a=1}^{|\widehat{\tau}_1|} \sum_{b=1}^{|\tau_2|} \left(-d(\widehat{z}_{1ja}, z_{2jb}) \cdot y_{ab}\right) \tag{6.6}$$

such that:

$$y_{ab} = 0 \quad \text{if } y_{cd} = 1 \quad \forall\ 1 \leq c < a \leq |\widehat{\tau}_1| \land 1 \leq b < d \leq |\tau_2| \tag{6.7}$$

$$y_{ab} = 0 \quad \text{if } y_{cd} = 1 \quad \forall\ 1 \leq a < c \leq |\widehat{\tau}_1| \land 1 \leq d < b \leq |\tau_2| \tag{6.8}$$

$$\sum_{a=1}^{|\widehat{\tau}_1|} y_{ab} \leq 1 \quad \forall\ 1 \leq b < |\tau_2| \tag{6.9}$$

$$\sum_{b=1}^{|\tau_2|} y_{ab} \leq 1 \quad \forall\ 1 \leq a < |\widehat{\tau}_1| \tag{6.10}$$

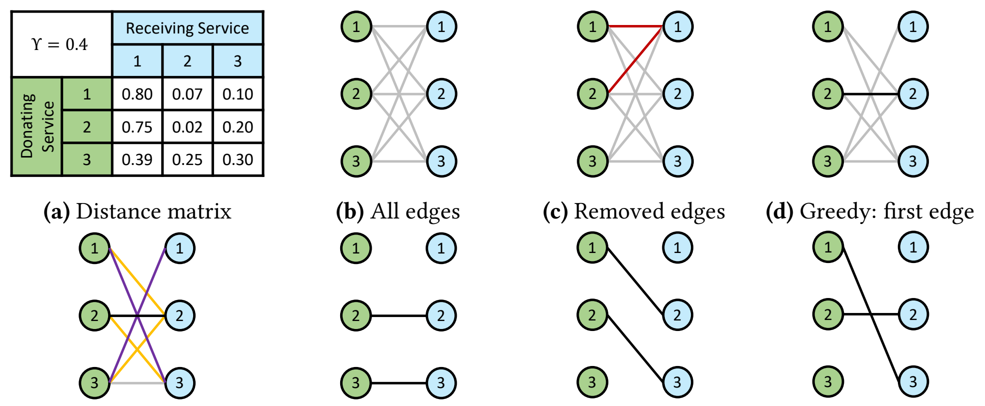
where y_{ab} is a binary indicator variable denoting whether $e_{a \to b}$ is selected. Equation 6.6 states the objective: maximise the total semantic similarity of the selected edges. Equations 6.7 and 6.8 enforce the ordering constraint that prohibits cross-overs. Equations 6.9 (6.10) state that each event in the receiving (donating) trajectory can be linked with at most one edge in the donating (receiving) trajectory.

Loose Bipartite Matching. If time smoothing is not performed, a looser form of bipartite matching can be implemented, in which cross-overs are allowed but their inclusion is penalised. Such behaviour is permissible as the receiving service knows the correct times that each event occurred, which means that the fully integrated trajectory will be in chronological order.

To enable looser matching, the objective function is altered to include a penalty term that penalises the inclusion of cross-over matches. This penalty can be fixed based on the number of cross-over matches, or it can vary depending on the extent of the cross-over matching. In the latter case, to obtain the penalty value, we first calculate the unrestricted Damerau-Levenshtein edit distance [37] in which transpositions are allowed but all other edit operations are prohibited. The edit distance (given by the integer number of swaps) is then multiplied by a constant δ to give the penalty value. Including δ balances the penalty term with the semantic distance attached to the edge weights.

In terms of the optimisation problem, as cross-overs are allowed, Equations 6.7 and 6.8 are redundant, and therefore removed as constraints. All other constraints remain unchanged. When variable penalties are used, the objective function becomes a non-linear function and, although the optimisation problem can still be solved using linear programming solvers, runtimes are longer (as we will see in Section 6.6.3). If speed is essential, other solvers or objective functions (e.g., a fixed penalty for each cross-over) can be used.

Examples. Figure 6.6 shows an example of the three matching processes. Figure 6.6a shows the semantic distance between events in the donating service's data (green) and the receiving service's data (blue), and $\Upsilon = 0.4$. First, all edges are instantiated (Figure 6.6b), before edges where $d(\widehat{z}_{1jk}, z_{2jk}) > \Upsilon$ are removed (highlighted in red; Figure 6.6c). These steps are common to all approaches. For the greedy process, the edge with the lowest semantic distance, $e_{2 \to 2}$, is selected first (Figure 6.6d). Edges emanating from these paired nodes (yellow) and edges that cross-over $e_{2 \to 2}$ (purple) are then removed as they are infeasible (Figure 6.6e).



$\Upsilon = 0.4$		Receiving Service		
		1	2	3
Donating Service	1	0.80	0.07	0.10
	2	0.75	0.02	0.20
	3	0.39	0.25	0.30

(a) Distance matrix **(b)** All edges **(c)** Removed edges **(d)** Greedy: first edge

(e) Greedy: infeasible edges **(f)** Greedy: final edges **(g)** Strict: final edges **(h)** Loose: final edges

Figure 6.6: Example of the three bipartite matching approaches

This leaves one remaining edge, $e_{3\to3}$, which is selected (Figure 6.6f). Greedily matching edges in this way yields an objective value of $-(0.02 + 0.30) = -0.32$.

The strict and loose matching processes can also be compared with the greedy approach. The global optimisation performed when using strict bipartite matching will select $e_{1\to2}$ and $e_{2\to3}$, which gives an objective value of $-(0.07 + 0.20) = -0.27$ (Figure 6.6g). This highlights how the greedy approach easily gets caught in local optima, as well as the superiority of the strict bipartite matching approach. Relaxing the cross-over constraint in the loose matching case will select $e_{2\to2}$ and $e_{1\to3}$, which results in an objective value of $-(0.02+0.10+2\delta)$ (Figure 6.6h). If $\delta \leq 0.075$, these edges are selected; otherwise, the edges from the strict matching case are used.

Figure 6.7 gives a better illustration of the differences between the strict and loose matching processes. Consider that the bank (green) shares data with the location service (blue) and no time smoothing is performed. Note how the event at 20:32 is perturbed to 9:10, which means it is matched with the cinema trip in the strict matching case. In the loose matching case, cross-overs are allowed, which allows this event (and others) to be (correctly) matched.

Choosing Integration Parameter Values. Integrating the two services' data successfully relies on setting the integration parameters Υ and δ appropriately. It is important to first note that, as both parameters are related to the semantic distance function, the optimal values for the parameters are dependent on the set of attributes common to both services (i.e., $\mathcal{A}_{1\to2} \cap \mathcal{A}_2$) and the characteristics of the semantic distance functions of those attributes. Unless these distance functions are simple (e.g., binary, linear), it is unlikely that the optimal settings can be derived theoretically. Hence, an empirical approach is recommended.

Deriving the integration parameters in this way is feasible for two reasons. First, the receiving service can tune the parameters independently of the donating service or the user, which minimises the communication cost. Second, each integration operation takes milliseconds, and so hundreds of parameter combinations can be considered without drastically affecting overall runtime. A simple way to find appropriate values is to iteratively consider possible parameter settings. More advanced, machine learning-based techniques could also be implemented, with the potential for external knowledge to be used as a basis for these models.

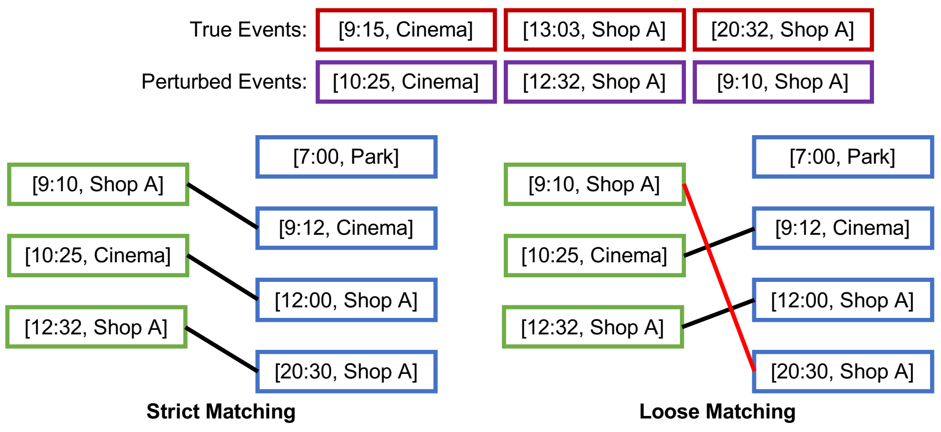


Figure 6.7: Illustration of strict and loose bipartite matching

For example, if the domain being studied has strong publicly observable correlations, these ground-truth linked events can be used to train a model to learn the parameters. Alternatively, users could provide historic data to the receiving service. The receiving service could then use this information as training data to learn Υ and δ.

Integrating Data

Once linked events have been identified, the data from both services is integrated as follows. First, τ_j^* is initialised. Then, for each selected edge $e_{a\to b}$, all attributes of z_{2jb} are added to τ_j^*. z_{2jb} is augmented with the perturbed attributes from $\widehat{z}_{1ja}$ for the attribute set $\mathcal{A}_{1\to 2} \setminus \mathcal{A}_2$. This continues for each selected edge. All unlinked events from $\widehat{\tau}_{1jk}$ and τ_{2jk} are then added to τ_j^*. One final check ensures that each composite trajectory satisfies the utility constraints, with perturbed attributes altered until the constraints are satisfied. This is done using the approach described in Section 6.4.5. Once complete, the receiving service can perform analysis on the composite trajectories.

6.5 Theoretical Analysis

This section contains theoretical analysis that supports the proposed solution and its key design choices.

Number of Perturbations. The number of perturbations needed to perturb a trajectory is:

$$N_p = \sum_{i=1}^{N_E} |E_i| + n_i - 1 \tag{6.11}$$

where N_E is the number of event periods in a trajectory, $|E_i|$ is the length of event period E_i, and $n_i = \min(n, |E_i|)$, which is the maximum n-gram length used in E_i. When fixed-length n-grams are used, there is one event period of length $|\tau|$ and so $N_p = |\tau|$ (when $n = 1$) and $N_p = |\tau| + 1$ (when $n = 2$).

For variable-length n-grams, the lower bound of N_p is $\frac{|\tau|}{n}$ when $|\tau|$ is perfectly divisible by n, or $\frac{|\tau|}{n} + 1$ otherwise. This lower bound is achieved when each event period has n points (with an extra event period of length $|\tau| \bmod n$ necessary when $|\tau| \bmod n$ is non-zero). The upper bound of N_p in the variable-length n-gram setting is harder to obtain as it depends on N_E and E_i, both of which are dependent on the data. It can be found numerically to be:

$$N_p = |\tau| + k(n-1) \quad k(n+1) \leq |\tau| < (k+1)(n+1) \tag{6.12}$$

where $k = \left\lfloor \frac{|\tau|}{n+1} \right\rfloor$.

Privacy. The privacy level achieved in perturbing each trajectory is the same as in Chapter 5. That is, as the privacy budget is divided into N_p equal portions (where $\epsilon' = \frac{\epsilon}{N_p}$), sequential composition dictates that the overall privacy loss for each trajectory is $N_p\epsilon' = N_p \times \frac{\epsilon}{N_p} = \epsilon$. Dividing the privacy budget in this way gives a privacy guarantee at the trajectory level, as opposed to the event level.

Once the perturbation of the donating service's trajectory is complete, the remainder of the data sharing and integration process is performed under post-processing. This is because none of these subsequent operations interact with 'sensitive' data. In traditional LDP settings, the true data used by receiving service for integration would be regarded as sensitive, and would need to be perturbed. However, the privacy setting outlined in Section 6.2 implies that this data is not sensitive *with respect to the receiving service*. This is because it is already shared with the receiving service in its unperturbed form. This contrasts with the data from the donating service, which needs to be privatised before the receiving service accesses it as this data would not ordinarily be shared by the user with the receiving service.

Utility. As this mechanism has multiple post-processing steps and (now) the data integration stage, a closed-form expression for the overall utility remains elusive. However, the utility of a single n-gram perturbation can be analysed.

Theorem 11. *The utility of a single n-gram perturbation is:*

$$\Pr\left[q(\tau(a,b), W_a^n) \leq -\frac{2}{\epsilon'}\left(\ln|\mathcal{WU}^n| + \xi\right)\right] \leq e^{-\xi} \tag{6.13}$$

Proof. Given its negativity, the maximum value of q is zero, and this is only obtained when $x = y = \tau(a,b)$. Hence, $OPT_q = 0$ and $|\mathcal{WU}^n_{OPT}| = 1$. Given that $\Delta_q = 1$, substituting these values into Equation 2.5 gives Equation 6.13. ■

Fixed- vs. Variable-Length n-grams. Theorem 11 dictates that utility is increased when the output domain set is small and/or ϵ is large, which occurs when N_p is small. While this suggests that using unigrams throughout is theoretically optimal (given that $|\mathcal{W}^1| \ll |\mathcal{W}^2|$), no consideration is paid to preserving correlations between consecutive events. This aspect of practical utility can only be realised by setting $n \geq 2$. Although using longer n-grams means larger domain sets, using variable-length n-grams can mean that ϵ' is larger (than it would be when $n = 1$) as there are fewer perturbations. This provides a slight counterbalance to the effects of the larger domain set. However, as the upper-bound for N_p is larger in the variable-length setting, this may not always be the case. Such a trade-off can only be examined empirically.

6.6 Evaluation

Having outlined the solution, we can now evaluate it using realistic data from five services. The evaluation has two stages. First, we assess how different mechanism settings affect the utility of the output data. We follow this by applying the solution to two data science tasks that allow practical utility to be assessed. A fraud detection study focuses on the individual level, and a facility location example quantifies utility at the aggregate level.

6.6.1 Data

Several real-world event sequence datasets exist, including film reviews [186], bikesharing trips [183], bank transactions [185], and online shopping [135]. However, each of these datasets covers a different set of users, and merely matching on user ID (where provided) has no guarantee of creating meaningful or sensible correlations. Hence, in the absence of a publicly available dataset that covers a sufficiently large number of users and attributes and allows for linked events to exist, realistic semi-synthetic data is used instead.

Data Summary

We use taxi trajectory data from San Francisco [189] and extract the latitude, longitude, and timestamp of each point. As the sampling rate is less than 10 seconds, each taxi's daily set of points is divided between users such that $|\tau_j| \approx 15$ (i.e., if the taxi trajectory has 600 points, we assign the points to 40 users). Each point is randomly assigned one of ten event types (detailed in Table 6.2) for which attribute values are randomly generated. Five services are modelled: a location service, bank, bikesharing service, film review service, and e-shopping platform. Some services (e.g., film review platform) can act as proxies for more general services. Most events are captured by one service only, although two services capture event types 5 (bank and location) and 9 (e-shopping and bank). These two event types allow linked events to be present in the holistic trajectories. Each service's data is obtained by taking vertical partitions of the holistic trajectories. Once processed, the data has 9.5 million events across 654,721 trajectories, 50,000 users and one month. Each holistic trajectory has 14.5 events on average, and each user has 13 trajectories on average. Each trajectory (as opposed to each user) has its own privacy budget, and the composition theorem can be used to quantify the overall privacy loss for each user.

Attributes

Table 6.2 also outlines the attributes associated with each event type. 'locID' refers to the location LLR/ULR for each point's co-ordinates. Some events recorded by the location service are check-ins to one of 100 randomly generated places of interest (POIs) and events are assigned to the nearest POI based on its co-ordinates. Each POI is associated with one of seven categories; users can purchase goods at three POI categories. Bikesharing trips start and end at one of 500 docking stations, with points assigned to their nearest station. 'tripStartID' is a binary variable denoting whether a trip is starting or ending. For film reviews, we generate 100 films, each of which is associated with one (or more) of five genres. These multi-genre groups each form an LLR and, for LLR distances, the Jaccard distance between groups is used. For ULRs, we select one genre for each film at random and assign the film to this ULR. For

Table 6.2: Event type summary

	Event Type	Attributes Collected
1	Bikesharing Trip	time, stationID, tripStartID, locID
2	Film Review	time, filmID, rating
3	Location (check-in)	time, poiID, locID
4	Bank Transaction (in-person)	time, payID, amount, locID
5	Bank Transaction (in-person, check-in)	time, payID, amount, poiID, locID
6	Bank Transaction (online)	time, amount
7	Online Shop (view/add to cart)	time, price, actionID, prodCatID
8	Online Shop (purchase, not bank)	time, payID, amount, price, actionID, prodCatID
9	Online Shop (purchase, bank)	time, payID, amount, price, actionID, prodCatID
10	Location (no check-in)	time, locID

bank transactions, 'payID' refers to one of three payment types: online, chip, or contactless. The shopping data has products across 20 categories ('prodCatID') that exist in a two-level hierarchy. Level 2 categories form the basis for LLRs, with each belonging to one of four level 1 categories, which form the ULRs. For products where both level 1 and 2 are the same, $d_c = 0$; for products with the same level 1 category but a different level 2 category, $d_c = 0.5$, and where level 1 categories are different $d_c = 1$. 'actionID' refers to one of three 'actions': add to cart, view, or buy. Where users purchase items, 'amount' is set equal to 'price'.

6.6.2 Experimental Set-Up

Baseline. As discussed in Section 2.2, there are no existing solutions to this new problem with which we can compare this work directly. Instead, the solution proposed in Chapter 5 can be adapted to this new setting. This is done by encoding every possible multi-attributed event as a 'location', and using each attribute's distance function as part of a multi-dimensional semantic distance function. As the original solution had no functionality for integrating data, the integration processes introduced in Section 6.4.6 are applied after the single source shares privatised data. This adapted version is henceforth referred to as the 'baseline'.

Default Settings. By default, we use variable-length n-grams (obtained using the exponential distribution method), utilise multi-service popularity information, and use strict bipartite matching with time smoothing. When perturbing trajectories, we set $\epsilon = 10$, and we explore the effect of changing this. When using multi-service popularity, counts are privatised using the Laplace mechanism with $\epsilon^* = 0.01$ for unigrams, and $\epsilon^* = 0.1$ for bigrams. Note that ϵ^* is independent of the privacy budget used in perturbation. When private popularity information is obtained, $\epsilon/4$ is used for the dummy perturbations, with the remainder used for the real perturbations. When using variable-length n-grams, $F = 0.9$ and $t_{gap} = 2$ hours, which gives $\lambda = 1.15$. Finally, the reachability threshold ψ is based on assumed travel speeds of cars (80 km/hr) and bikes (20 km/hr), and it does not vary with time.

Implementation. Experiments were conducted using Matlab 2021b on Linux servers comprising two Intel Xeon Platinum 8268 2.9GHz 24-core processors, with 32GB RAM per core.

Evaluation Measures. When assessing the different settings for popularity and n-gram length, we calculate the mean semantic distance, as in Chapter 5. We use the trajectory data before integration to ensure a one-to-one matching in events, and we report the mean semantic distance calculated across all $u_j \in \mathcal{U}$. As the number of linked events is much smaller than the number of unlinked events, it is better to use the Matthews correlation coefficient (MCC) [162] to evaluate the integration methods, as this has been shown to be a better evaluation measure than F1 score when the data is heavily unbalanced [57, 190]. With a range of $(-1, +1)$, it is defined as:

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{6.14}$$

where TP denotes the number of true positives, TN the number of true negatives, FP the number of false positives, and FN the number of false negatives. MCC values of +1, 0, and −1 represent totally correct, essentially random, and totally incorrect classification, respectively.

6.6.3 Mechanism Analysis

The first part of this evaluation studies the effect that changing mechanism parameters has on utility.

Incorporating Popularity

Table 6.3 shows that, in general, all methods for incorporating popularity information outperform the baseline. Unsurprisingly, public popularity is almost always the best performing method; this can be seen as the upper performance bound. Using multi-service popularity information outperforms the private and no popularity methods and performs comparatively well to the public popularity method. Obtaining popularity information privately performs less well, as using part of the privacy budget for dummy perturbations affects the real perturbations. In some cases, it still outperforms the baseline method. Interestingly, when the location service shares data, the baseline is the best-performing method. This is likely due to the fact that the baseline was specifically designed for sharing location sequences, whereas this solution is a more generalised perturbation mechanism.

n-gram Length

Table 6.4 shows that variable-length n-grams consistently outperform fixed-length bigrams, and generally perform better than fixed-length unigrams. This demonstrates that the effect of larger n-gram sets is outweighed by the lower number of perturbations on average, as discussed in Section 6.5. There is minimal difference in the methods for determining event periods; the best choice is likely to depend on the attributes being shared and/or the characteristics of the trajectory.

Privacy Budget

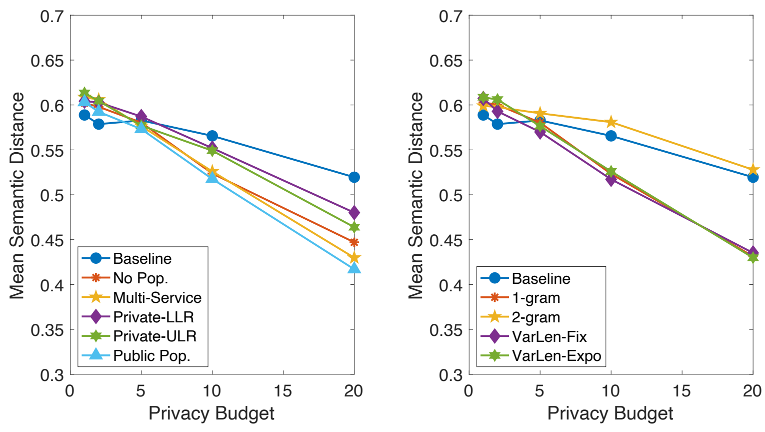
To examine how changing the privacy budget affects the accuracy for different perturbation settings, five values are used: $\epsilon = \{1, 2, 5, 10, 20\}$. Figure 6.8 shows how accuracy changes when the film review service shares data with the e-shopping service. As expected, a high privacy budget leads to greater utility. Using multi-service or public popularity information is consistently better than the private popularity and baseline methods. Likewise, unigram

Table 6.3: Effect of popularity incorporation methods on MSD; percentage compared to baseline in brackets

Donating Service	**Attributes Shared ($\mathcal{A}_{1\to 2}$)**	**Base.**	**No Pop. Info.**	**Multi-Service**	**Private LLR**	**Private ULR**	**Public Pop.**
Film	time, filmID, rating	0.566	0.529 (6.5)	0.526 (7.0)	0.552 (2.4)	0.549 (2.9)	0.518 (8.5)
Bank	time, amount	0.594	0.570 (3.9)	0.546 (7.9)	0.598 (-0.8)	0.603 (-1.6)	0.481 (18.9)
Shopping	time, actionID, prodCatID	0.711	0.676 (4.9)	0.665 (6.4)	0.700 (1.6)	0.693 (2.5)	0.663 (6.7)
Location	time, locID	0.473	0.490 (-3.5)	0.491 (-3.7)	0.500 (-5.7)	0.501 (-5.9)	0.497 (-4.9)
Bikeshare	time, stationID, tripStartID	0.553	0.394 (28.7)	0.367 (33.6)	0.397 (28.2)	0.397 (28.2)	0.330 (40.4)

Table 6.4: Effect of *n*-gram length on MSD; percentage compared to baseline in brackets

Donating Service	**Attributes Shared ($\mathcal{A}_{1\to 2}$)**	**Base.**	**Fixed 1-gram**	**Fixed 2-gram**	**Var. Len. Fix. Gap**	**Var. Len. Expo.**
Film	time, filmID, rating	0.566	0.523 (7.5)	0.581 (-2.7)	0.517 (8.6)	0.526 (7.0)
Bank	time, amount	0.594	0.550 (7.3)	0.546 (8.0)	0.536 (9.7)	0.546 (7.9)
Shopping	time, actionID, prodCatID	0.711	0.675 (5.1)	0.715 (-0.5)	0.667 (6.2)	0.665 (6.4)
Location	time, locID	0.473	0.491 (-3.8)	0.491 (-3.7)	0.496 (-4.7)	0.491 (-3.7)
Bikeshare	time, stationID, tripStartID	0.553	0.371 (32.9)	0.555 (-0.4)	0.392 (29.2)	0.367 (33.6)



(a) Popularity **(b)** n-gram length

Figure 6.8: Privacy budget vs. utility

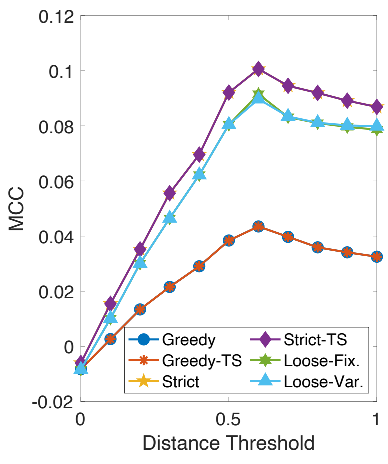
and variable-length n-gram methods perform well, whereas the baseline and bigram methods struggle, even when ϵ is high.

Integration

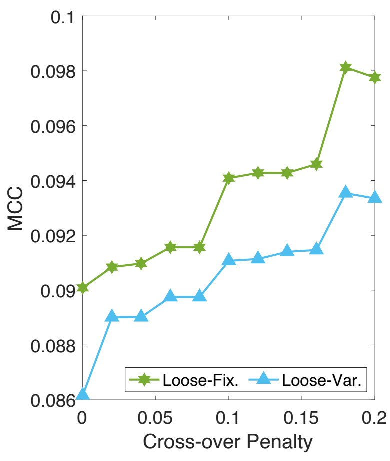
As discussed in Section 6.4.6, tuning Υ and δ is fundamental to successful integration. Figure 6.9 shows the effect that changing $\Upsilon$and δ has on the MCC when the bank is sharing data with the shopping service. We also study the effect that time smoothing (TS) has on results. There is a clear hump in Figure 6.9a that indicates the best setting for Υ. When Υ is too small, many true matches are missed; when Υ is too high, many false matches are identified. Strict matching offers the best results: it is up to 131% better than greedy matching, and up to 12% better than loose matching. Time-smoothing has no measurable effect, primarily due to the fact that the receiving service opts to use true data, including time, when integrating data. Figure 6.9b shows that, as δ increases, the penalty term begins to outweigh any benefit of including cross-over matches. Accordingly, the edges selected when using loose matching start to be the same as when using strict matching.

Runtime Analysis

Figure 6.10 shows how runtime is affected as mechanism settings change when the bank shares data with the film review service. Under its default settings, this chapter's solution is 357 times faster than the previous solution. This is partly due to the removal of the optimal reconstruction step outlined in Section 5.3.3. When private popularity information is used, runtime increases notably, due to the much larger n-gram sets required for dummy perturbations. Similarly, when fixed-length unigrams are used, runtime is much smaller, owing to the smaller n-gram sets. With the loose matching setting (when a variable penalty is used), slightly longer runtimes are observed, as expected given the added complexity of the objective function. Runtime is broadly unaffected as mechanism parameters (e.g., ϵ, Υ, δ) change.

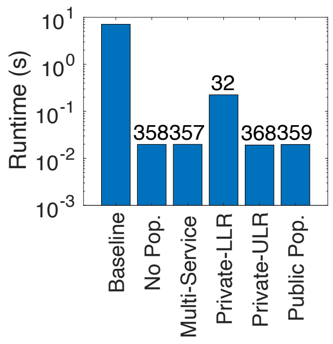


(a) Integration distance threshold

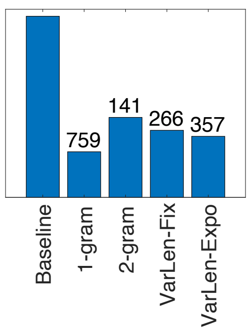


(b) Integration penalty

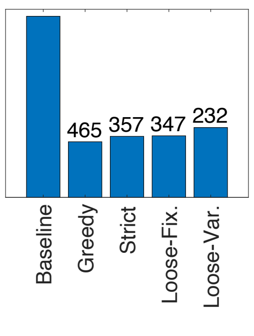
Figure 6.9: Integration parameters vs. utility



(a) Popularity



(b) *n*-gram length



(c) Integration

Figure 6.10: Runtime as mechanism settings vary; runtime as factor of baseline shown (e.g., 357x faster)

6.6.4 Fraud Detection

As outlined in the motivating example, fraud detection is a key potential application for this work. We now consider the case in which the bank suspects that the accounts of certain customers have been compromised. To verify this, they seek to cross-reference this with data from other services.

Outline. Fraud is assessed using the following logic. If a suspicious transaction occurs at, say, 19:35 in Shop A, it is not fraudulent with high probability if data from other services also locate Jane in the surrounding region at similar times. Conversely, the transaction is fraudulent with high probability if other services are not co-located with the transaction but are co-located with each other. If donating services (a) do not have enough data on the user, (b) do not have data on events close in time to the suspicious transaction, or (c) have conflicting information, the receiving service can report an 'inconclusive' finding. Although more sophisticated definitions of fraudulent activity and methods of fraud detection exist, they are beyond the scope of this work, but can be incorporated if desired.

Table 6.5: Fraud detection results

Number of	Fraud MCC		Not Fraud MCC		% Inconclusive		
Services	Chap. 5	Chap. 6	Chap. 5	Chap. 6	True	Chap. 5	Chap. 6
1	0.000	0.000	0.243	0.459	56.0	82.3	66.3
2	0.120	0.326	0.192	0.433	30.9	62.3	39.4
3	0.171	0.417	0.150	0.416	12.8	28.7	17.9

The likelihood of fraud is assessed using the true and perturbed data from up to three location services. As there are many more non-fraudulent transactions than fraudulent transactions, accuracy is reported using the MCC, as opposed to the F1 score, which relies on more evenly distributed data. Besides this, MCC is an appropriate measure for this setting as it includes all four classification quantities, each of which is important in assessing the overall quality of the proposed solution for fraud detection. Identifying true positives allow the bank to block a user's card and prevent further fraud, whereas minimising false positives helps to stop a user's card from being suspended unnecessarily. Preventing false negatives is important to stop fraud from going undetected, and true negatives can be helpful in training future fraud detection models. The percentage of 'inconclusive' flags is also reported as a way to assess the overall certainty in the fraud detection results. Where the number of services used to determine fraud is less than the total number of donating services, the mean across all possible service combinations is given.

Results. Table 6.5 shows that the proposed solution is up to 23% more accurate than the baseline, and also reports fewer inconclusive labels. As expected, the accuracy of fraud detection increases as the number of services increases, as there is a higher likelihood that services can confirm co-location of users away from the fraudulent transaction. The accuracy of identifying non-fraudulent events remains relatively stable for the proposed solution, although the baseline performs notably worse. As the number of services increases, the number of inconclusive labels decreases, with the proposed solution performing similarly compared with the true data. This highlights how data from more services brings a greater degree of certainty to the fraud detection system.

6.6.5 Facility Location

Another real-world application of the composite trajectory problem is in locating facilities. Consider that a film company (e.g., Netflix) has access to a set of facilities in a city where they can advertise (e.g., billboards, bus stops). To maximise the exposure of advertising to the right groups of people, it wants to assess what genre of film should be advertised in each facility, based on where active reviewers spend their time. For example, if users who watch family films spend time near playgrounds, they may consider advertising their family films there.

Outline. To perform this task, we consider the MaxCover problem, where a facility 'covers' a user if they are located within a certain radius of it (set to 1 km). Users can be covered by multiple facilities, and be covered by the same facility more than once. For each genre, a user is assigned a genre influence weight from the set $\{-2, -1, 0, 1, 2\}$ based on their average rating of films of that genre. +2 is associated with genres that they strongly like (i.e., high

Table 6.6: Facility location results

Setting	Scenario	$\lvert\mathcal{F}\rvert = 100$		$\lvert\mathcal{F}\rvert = 200$		$\lvert\mathcal{F}\rvert = 500$	
		Ch. 5	**Ch. 6**	**Ch. 5**	**Ch. 6**	**Ch. 5**	**Ch. 6**
Location	1	0.40	0.60	0.40	0.60	0.26	0.66
	2	0.20	0.30	0.15	0.25	0.24	0.24
	3	0.30	0.50	0.30	0.50	0.32	0.56
Time and Location	1	0.25	0.43	0.15	0.51	0.23	0.47
	2	0.18	0.28	0.16	0.23	0.21	0.23
	3	0.28	0.28	0.20	0.45	0.29	0.38

average rating) and −2 is assigned to genres that users strongly dislike. Each facility has a facility-genre score, which is adjusted by a user's genre influence weight if the facility covers the user. For example, in Jane's data (Table 6.1c), their genre influence weight for 'drama' films is −2. As such, the facility-genre score for any facility that covers Jane will decrease by 2 for this genre. Similarly, for 'action films', the facility-genre score will increase by 2, as Jane rates these films highly.

Once this score is obtained for all facility-genre pairs, facilities are selected, and assigned to genres. Maximum weight bipartite matching is used for facility-genre assignment, where facility-genre scores determine edge weights. We consider two sets of three scenarios. The first set of scenarios uses location information only, whereas the second set uses location and time information. This latter case allows us to model the realistic setting where the film company can change which films are advertised at each facility across the day. In Scenario 1, the film service asks the bikesharing service for data, whereas Scenario 2 sees the film review service ask the location service for data. Finally, in Scenario 3, the film service uses data from both services. In each case, 10% of the available facilities are assigned to each genre. Similarity between the true and perturbed data is quantified using the SDC (Equation 3.19). $\mathcal{F}_i$ is the set of facilities to which genre i is assigned when true data is used, and $\widehat{\mathcal{F}_i}$ is the corresponding set for the perturbed data. SDC values are averaged across all genres.

Results. Table 6.6 shows the new solution is up to 3.4x better than the baseline adaptation. This highlights its strength in aggregate-level queries. When the film service uses data from both services (Scenario 3), performance decreases, which may be due to disparities in the data due to perturbation. Performance also dips when time is considered as (a) the n-gram sets are larger and so perturbation is less accurate, and (b) the true number of users covered at each facility will be smaller and so perturbation has a larger effect. Finally, performance remains stable as the number of facilities vary – an important finding that underlines the solution's capability to be applied to large-scale analytics tasks.

6.7 Discussion

This chapter introduced the private composite trajectory release problem in which multiple services can share data in accordance with LDP, with the consent and control of each user. The key challenge with this problem is integrating the data between the two services privately. The proposed solution achieves this by using efficient semantic bipartite matching between the perturbed trajectory from the donating service and the previously collected unperturbed data of the receiving service. Whereas this chapter focuses on the two-service setting, the proposed solution can be applied with any number of services. When multiple services want the donating service's data (e.g., many services want to know where Jane is travelling and ask Location Service E for this data), Jane's data should be perturbed once and the same perturbed data should be shared with each receiving service. This prevents rapid degradation of the privacy budget. When the receiving service has data from multiple donating services, the biggest challenge is in integrating the data. The simplest solution is to apply the pairwise integration method outlined in Section 6.4.6 consecutively, although this risks missing linked events or the same event appearing multiple times. Implementing k-partite matching is likely to lead to better matching, although the problem is NP-hard [188], and is therefore left to future work.

Although the proposed solution's performance compared to the optimum can be improved, it is important to consider the limited information available to perform integration. Specifically, only the perturbed trajectory of the donating service, unperturbed trajectory of the receiving service, any popularity information, and any external knowledge can be used. Other techniques could be implemented, but many would require more of the privacy budget to be spent, or violate privacy entirely, which makes them unviable. Even adaptations of the proposed approach have their limitations. For example, one could give a larger weight to the temporal dimension in the semantic distance function, on the basis that it would be easier to preserve linked events based on the temporal attribute. However, this would mean that other attributes would be perturbed to a greater extent, which would affect the quality of the output data. The lack of viable alternatives only serves to emphasise the complexity of the integration problem, and motivates further refinement of the solution as the basis for future work.

Finally, whereas Chapter 5 focused exclusively on spatio-temporal trajectory data, the nature of this problem necessitates a more generalised perturbation mechanism that can handle other types of data (e.g., shopping or browsing data). This augmented perturbation mechanism, which uses variable-length n-grams and does not rely on publicly available popularity information, outperforms the n-gram-based solution of Chapter 5 when perturbing a wide range of trajectories. However, a consequence of a more generalised mechanism is that it is harder to demonstrate problem- or domain-specific utility, as evidenced in the facility location and fraud detection experiments. Hence, more work is needed to tune the mechanism, its parameters, and distance functions when applying the solution to specific applications.

Chapter 7

Discussion and Future Work

This chapter provides a critical summary of the material presented in this thesis, and provide recommendations for future research. Section 7.1 summarises the main technical contributions of the thesis, with reference to the research questions and desirable solution characteristics introduced in Chapter 1. These contributions are also summarised in Table 7.1. Section 7.2 discusses some of the limitations of the solutions in this thesis, and briefly outlines how they could be addressed. This is followed by Section 7.3, which presents several interesting and exciting avenues for future work, all of which would extend, enhance, or complement the work of this thesis. This chapter, and thesis, concludes with brief final remarks in Section 7.4.

7.1 Conclusions

Chapter 3 proposes two solutions that address the challenge of generating synthetic spatial point data. Both solutions produce synthetic data that has high fidelity with respect to the original data, whilst also satisfying the strict requirements of DP. The first approach partitions the data using existing effective differentially private methods, and then generates synthetic data within these private regions. A novel private adaptation of kernel density estimation that is specifically suited to the setting of multiple point generation is shown to capture the underlying distribution of the real data more effectively than other methods. The second data generation approach uses the underlying structure of the road network to control the data generation process by sampling from private micro-histograms modelled along edges in the road network. This helps to maintain the synthetic data's alignment with the road network, which is crucial for achieving high utility in location analytics tasks. Both methods further improve the real-life accuracy and utility of the generated data by using public geographic knowledge, such as coastlines and rivers, to define out-of-bounds regions in which points cannot be generated. An extensive set of experiments shows that the partitioning-based solutions perform significantly better than alternative approaches, and are up to 28x more accurate and 3.7x faster. The road network-based approach is even more effective, especially when the location data is well-aligned with the underlying road network, and it is up to 37x faster than partitioning-based approaches.

Table 7.1: Summary of thesis contributions

Ch.	Problem Summary	Proposed Solution(s)	Key Results	Applications
3	Generating synthetic spatial point data with centralised differential privacy	Framework that combines differentially private partitioning (grids and clustering) with private kernel density estimation to sample points; Road: a method that samples from noisy histograms modelled along edges in the road network	Road is best when data is well-aligned with (grid-like) road network; UGrid-KDE and AGrid-KDE are more effective with less-ordered data or less-structured road networks	Facility location; advertising; modelling accessibility to public services
4	Generating synthetic spatial point data with label-LDP	GeoPointGAN: a GAN-based solution that incorporates randomised response to flip the real and fake labels of associated with locations	GeoPointGAN outperforms three GAN-based baselines; some private GeoPointGANs perform as well as non-private GeoPointGANs due to regularisation effects	Monitoring disease spread; managing working conditions; urban planning
5	Sharing sequences of visited places of interest with LDP	Global solution that perturbs trajectories as single points in high-dimensional space; method that uses the exponential mechanism to perturb hierarchically structured overlapping *n*-grams	Global solution is infeasible; overlapping *n*-grams are better at preserving correlations than perturbing POIs individually; incorporating external knowledge and hierarchical decomposition also improves performance	Societal contact tracing; public service provision; advertising
6	Sharing and integrating event sequences from multiple services with LDP	Perturbation of two-level variable-length *n*-grams; data integrated using bipartite matching-based optimisation; popularity information included in a generic way	Variable-length *n*-grams offer improvements over fixed-length *n*-grams; perturbation is over 300x quicker than in Chapter 5; strict matching is 131% better than greedy matching	Fraud detection; facility location; better service quality

Chapter 4 presents GeoPointGAN, a novel GAN-based solution for generating data in accordance with label-LDP, which is itself introduced as a more practical privacy setting for generating synthetic spatial point data. GeoPointGAN provides label-LDP through a randomised response mechanism that flips the labels provided to the discriminator, thereby providing plausible deniability to each individual's *association* with a location. Beyond its privatisation properties, GeoPointGAN's architecture has several novel components. These include point-level classification, and a PointNet-based generator that transforms data from a latent space to a meaningful representation in the *same* dimensionality (as opposed to existing GANs, which use upsampling). Evaluation with real-world data also shows that GeoPointGAN is up to 10 times better than the most competitive baseline GAN. Experiments also show that, in some settings, a private GeoPointGAN performs as well as its non-private counterpart – a remarkable observation that indicates that GeoPointGAN is effective at minimising the impacts of the added noise. These findings illustrate that label flipping can also have generalisation and regularisation effects on the model performance, which can be further contextualised and studied with reference to related literature.

Turning to the problem of sharing sequences of visited locations, Chapter 5 presents two solutions, both of which satisfy strict ϵ-LDP. Despite its theoretically optimal qualities, the first solution is unfeasible owing to its high time and space complexity. This motivates a more efficient and scalable solution that is based on perturbing overlapping, hierarchically structured n-grams of trajectory data. n-gram perturbation allows the spatio-temporal relationship between adjacent points to be captured, while remaining computationally feasible. Moreover, using *overlapping* n-grams allows more information for each point to be captured, whilst continuing to satisfy LDP. Semantic distance functions incorporate a rich set of public knowledge to adjust the probability of certain perturbations in a utility-enhancing manner. Finally, exploiting the (publicly known) hierarchies that are inherent in space, time, and category classifications to structure n-grams in a multi-dimensional hierarchy has notable utility and efficiency benefits. Evaluation with real-world data highlights how the mechanism outperforms alternative approaches by up to 6.5x, and can effectively preserve the semantic characteristics of trajectories, all whilst satisfying LDP.

This problem, and its solution, is extended in Chapter 6 to the setting in which two services wish to privately share event sequences with each other. When integrating the donating service's perturbed data, a hybrid privacy setting allows the receiving service to utilise their previously collected data on a user in its unperturbed form to enhance utility. Two valuable improvements are also made to the perturbation mechanism. First, variable-length n-grams are used to prevent the mechanism from trying to preserve correlations between consecutive events where no such correlation is likely to exist, whilst also utilising the privacy budget as best as possible. Second, popularity information is incorporated in a more generic and realistic manner to cover a comprehensive spectrum of cases where such information is public knowledge, collated from multiple sources, privately learned from data, or unavailable entirely. These improvements improve utility by up to 33%, and the removal of linear programming-based optimisation when reconstructing trajectories leads to this solution being over 300 times faster. Linked events are preserved by the receiving service by solving a bipartite matching-based optimisation problem that links semantically similar events. The proposed approach, which optimally solves the problem in most cases, is up to 131% better than other greedy methods.

Finally, the consistent finding throughout this thesis has been that incorporating real-world knowledge can improve the utility of private data considerably. Importantly, by only utilising publicly available external knowledge, these utility benefits can be realised without affecting the privacy guarantee. Furthermore, in all four content chapters, the proposed solutions demonstrate strong performance in practical analytics tasks that are inspired by real-world queries and applications. For example, both data synthesis solutions perform well in range and hotspot queries, and obtain near-identical responses to facility location queries. Similarly, the n-gram-based solution in Chapter 5 can perturb trajectories privately, without affecting the presence of spatio-temporal hotspots. The solution is extended to the composite trajectory problem in which strong performance is achieved for two popular data science tasks – fraud detection and facility location. This practical utility underscores the potential for more widespread private data sharing. Moreover, it demonstrates that each solution exhibits the desired characteristics outlined in Section 1.2, which, in turn, indicates that the research questions of Section 1.1 have been addressed successfully.

7.2 Limitations

It would be remiss not to acknowledge the limitations of this work, and four primary limitations are outlined here. Addressing these limitations are natural avenues for future work, and potential ideas for this work are also discussed in this section.

Need for Clear Data Generation Guidelines

Chapter 3 concludes with guidance that outlines the scenarios in which each data generation method is appropriate. However, these recommendations (e.g., those based on the alignment with the road network) are somewhat subjective and could lead to sub-optimal utility in the synthetic data. To address this, it would be sensible to undertake a more rigorous evaluation of the methods to understand the scenarios in which each method performs best. This would include applying the methods to different dataset types across a range of cities and road network types. A more robust solution would be an integrated data generation framework that selects the best data generation method based on characteristics of the data (e.g., dataset size, privacy budget, alignment with the road network, geographic extent).

Spatio-Temporal Data Generation

Both Chapters 3 and 4 focus on generating spatial point data, and do not consider the temporal dimension, which is somewhat limiting considering that many analytics tasks benefit from using spatio-*temporal* data. The methods from Chapter 3 could be naïvely extended by using temporally divided subsets of data to generate synthetic datasets across a series of time intervals. Extending GeoPointGAN to the spatio-temporal domain is less trivial, although there has been recent work in the field, such as Xu et al. [258] who use a sequential version of optimal transport to account for temporal consistencies, and Klemmer et al. [142] who introduce a new embedding loss to help the learning of spatio-temporal autocorrelations.

These methods could serve as a basis for future work that would give GeoPointGAN the capability to generate label-LDP compliant spatio-temporal data.

High Privacy Budgets

In Chapters 5 and 6, the default privacy budgets used are 5 and 10 respectively (although decent performance is achieved when lower values are used). Although these values are within the range of those used in current real-world deployments of LDP (as noted in Section 5.5.2), lower privacy budgets are intuitively desirable. Utility-focused enhancements in other areas (e.g., by tuning mechanism parameters) would allow for lower default privacy budget values to be used. This is discussed more in Section 7.3, alongside more practical methods for determining the privacy budget.

Users with Multiple Trajectories

In Chapters 5 and 6, each trajectory is treated independently, regardless of user ID. Although this is acceptable when users only have one trajectory in the dataset, there is a risk of privacy leakage if they have multiple trajectories, especially if they all exhibit similar patterns (e.g., a commuter will have fairly similar trajectories during the working week). It is worth noting that this problem is not unique to this work, as the risk of privacy leakage through repeated queries is generally seen as one of DP's main vulnerabilities [82]. Nevertheless, addressing this issue is important in ensuring that the solutions in this thesis are successfully applied in real-world settings.

7.3 Future Work

Beyond addressing the limitations outlined above, there are five other strands of future work that would complement the work of this thesis, and these ideas are briefly outlined here.

7.3.1 Extensions

First, there is ample scope to extend the core problems and solutions presented in this work.

One extension opportunity is to create a combined data generation technique that utilises the strengths of the partitioning- and network-based approaches. Chapter 3 demonstrated that an adaptive grid-based approach is useful in producing high quality synthetic data. However, when partitioning the space, adaptive grids are agnostic towards the road network, which can negatively impact utility if the resultant grid is poorly aligned with the data (which itself tends to display strong alignment with the road network). Hence, better all-round utility could be achieved if the data space was partitioned using adaptive grids centred along major roads in the road network.

Another extension is to relax some of the assumptions upon which the solutions are built. For example, in Chapters 5 and 6, it was assumed that the length of the trajectory was non-private and did not need to be perturbed. Modelling trajectory length to be sensitive could be done in several ways. For example, randomised response could be used to determine whether a POI or event should be included in the perturbed trajectory. Alternatively, Laplace noise could be added to the trajectory length, with POIs/events added or removed as necessary.

While allowing the real and perturbed trajectory lengths to be different would provide an additional degree of privacy, the effect on utility could be large as many real POIs or events could disappear, and many fake POIs or events could appear, which would affect aggregate counts. This phenomenon somewhat mirrors the problems of adding noise to the number of edges in Section 3.3.

A final area for extension would be to model real-world behaviour better. For example, when trajectories are perturbed in Chapters 5 and 6, each trajectory is perturbed individually. Although this gives each user a strong level of privacy, it fails to consider that users may travel around in groups (e.g., as a family). This would lead to decreased utility as each family member would report a different set of POIs/events, even if they travelled as a group. This also presents a privacy risk as repeated querying of the same family could leak sensitive information about their activities. In any case, correlations between rows of data has been shown to highlight vulnerabilities of DP algorithms, especially those in the centralised setting, as noted by Liu et al. [154] and Wang et al. [233] (among others). One way to accommodate this would be to ensure that, where members of the same group have the same events or POIs, the output data for all family members is the same for these events. Where necessary, this could be performed by utilising the post-processing properties of DP and LDP. A similar technique could be applied when sharing social media or social interaction data, where discordant perturbed data would harm utility and risk privacy leakage.

7.3.2 Personalised Privacy

While this thesis has focused on incorporating real-world knowledge to enhance the utility of data sharing methods (i.e., the server side), it has not focused on the real-world characteristics of users and the potential effects they can have on utility (i.e., the client side). In particular, it is unreasonable to expect that all users will have the same attitude towards the privacy of their data. That is, some users may be happy to share all of their data at a fine granularity, whereas others may be willing to share only some of data and/or only if higher privacy protections are implemented. Furthermore, when dealing with trajectory data, it is also unreasonable to expect that a user will see each point in their trajectory as equally sensitive. For example, Alice may consider visiting a hospital to be more sensitive than visiting a supermarket. Therefore, assigning the same privacy budget to each user (or each point in a trajectory) is unrealistic, and potentially harmful from a utility perspective.

To account for this, notions of personalised differential privacy have been proposed in the centralised [9, 131] and local domains [55], and they have been used in several works since [e.g., 173, 182], including in the spatio-temporal domain [e.g., 22, 55, 74, 181]. As expected, these works have highlighted the utility improvements that can be realised by using the personalised setting. Although these personalised variants are relaxations of the traditional definitions, they arguably offer a more realistic and practical privacy setting, in addition to expected utility benefits. Hence, an important avenue for future work would be to extend the work of this thesis to the personalised privacy domain.

As Chapters 4–6 use the local setting, personalised LDP (or label-LDP) can be implemented easily. In Chapter 4, the probability that a label is flipped would be smaller (larger) for those who have liberal (conservative) attitudes towards privacy. The regularisation and generalisation effects that GeoPointGAN exhibited are expected to persist, and so it is likely that utility would remain strong when using personalised label-LDP. In Chapters 5 and 6, each user can have their own privacy budget for their trajectory, and assign it how they wish (e.g., users could split their privacy budget unevenly across their trajectory to protect more sensitive locations or events). Even if this were done, the overall trajectory-level privacy guarantee would be unaffected. While local perturbations would not require the value of the privacy budget to be transmitted by the user to the aggregator, an option to do so could be implemented. This would give the aggregator (or receiving service) more information regarding the likely accuracy of the data, which they could consider when using the output data for analyses. For example, if Alice and Bob have similar true trajectories, but they use privacy budgets of 0.1 and 10 respectively, users of the output data might wish to place more weight on the data reported by Bob as it is more likely to be accurate.

Extending the data generation methods proposed in Chapter 3 offers more opportunities. Specifically, the ideas of Chen et al. [55] could be combined with kernel density estimation, which operates effectively when it can be trained on real data. In their work, Chen et al. [55] allow users to specify a geographic 'safe region' in which they are willing to be associated. These (hierarchically structured) safe regions could form the basis for the regions in which to conduct data generation. That is, if some users provided their exact co-ordinates, this data could be used to train a kernel density estimator at the block-level. Data generated at this level could then be combined with data from those willing to reveal their block to train a kernel density estimator at the postcode level, etc.

7.3.3 User Study of Privacy Budgets

Whereas personalised (L)DP considers users to have non-uniform attitudes towards data sharing, selecting the appropriate privacy budget values for these users remains an outstanding issue. In keeping with this thesis' aims to develop practical, real world-minded data sharing mechanisms, it is important to consider how humans themselves perceive DP and LDP outputs, especially with respect to the privacy budget. For example, many people may be happy with LDP outputs where $\epsilon = 5$ (as used in Chapter 5). Alternatively, they may desire a smaller privacy budget value, depending on how they perceive their privatised data. Therefore, a valuable piece of future work would be to conduct a study that assess how people perceive outputs from the mechanisms developed in this thesis. The results of this study would help to further refine the mechanisms given a 'base' privacy budget from which to work.

While investigating how people assess their own privacy budget would be important for enhancing the work of this thesis, it would also offer a wider contribution towards DP research. To date, there have been no studies that examine how users perceive DP outputs determined using different privacy budgets, despite DP and LDP mechanisms being used in the real world. Indeed, Dwork et al. [85] found that there is little-to-no consensus among academics or practitioners on how to set the privacy budget in order to achieve high utility and meaningful privacy. Hence, a large-scale, wide-ranging study on humans would ensure that future DP solutions could achieve greater utility if the (practical) level of privacy that was necessary could be more easily quantified.

7.3.4 Is Synthetic Data Practical?

Despite its growing popularity, it is important to consider that using synthetic data in analytics or to train machine learning models is not without risk, and may not be the silver bullet solution that many claim [e.g., 14, 26]. Recently, Stadler et al. [205] have found that (differentially private) synthetic data fails to provide a better trade-off between privacy and utility for tabular data than traditional row sanitisation-based privacy solutions, especially for outlier detection. Similarly, Ganev et al. [96] show that training models that using differentially private synthetic data can risk treating different subgroups unevenly, which in turn could lead to unreliable, unfair, or discriminatory conclusions. Although the synthetic data generators presented in Chapters 3 and 4 were not evaluated against these tasks, it is important to recognise that spatio-temporal data is susceptible to attacks or privacy leakage based on outliers on marginalised subgroups. This is especially pertinent as they have been designed to be used to inform real-world decision-making. But, while it is beyond the scope of this thesis to fully consider the possible end effects of using this synthetic data in a practical setting, it would be sensible to devote future work to investigating this issue.

7.3.5 Real-World Deployment

This thesis has presented general purpose methods for private data generation and publication, although no solution has focused exclusively on any one specific deployment. As evidenced throughout, each problem, domain, city, and application have their own unique characteristics, which necessitates the careful tuning of mechanism parameters to maximise utility. This motivates the final strand of future work. For example, the fraud detection study in Chapter 6 highlighted that, although reasonable utility was obtained under the default settings, further tuning of mechanism parameters would improve utility. Tuning the mechanism includes choosing problem-specific values for user-specified parameters (e.g., the length of n-grams), incorporating more domain-specific external knowledge (e.g., geographic boundaries of islands and bridges, public transport and event schedule data), and further investigation into the values for empirically derived parameters (e.g., the harmonisation threshold in Chapter 6). Only by performing this tuning when deploying these solutions in real-world settings will the greatest utility benefits be realised.

More abstract applications for this thesis' work include data clean rooms, private data lakes, and private data marketplaces. For example, the synthetic data methods from Chapters 3 and 4 could be used to create data clean rooms in which researchers and analysts could generate and use large synthetic datasets for advanced data analyses, all with strong privacy guarantees. This synthetic data could then be combined with locally private trajectories to form part of a private data lake, which would allow analysts to devise complex machine learning models using high fidelity private data. Finally, the multiple service setting of Chapter 6 raises the possibility that private data marketplaces or auctions [275] could be created. In these settings, users could share their data with different services, where different pieces of data would have different monetary value depending on the data's uniqueness, density, etc., as well as the privacy budget used. Pursuing these blue sky goals would help to realise the full potential of the work in this thesis, whilst also unearthing new problems that can help to make private data sharing more widespread.

7.4 Final Remarks

This thesis aimed to develop differentially private algorithms for generating and sharing spatio-temporal data, with a particular focus on achieving high levels of utility in practical, real-world location analytics queries. The proposed solutions primarily achieve this by incorporating real-world knowledge into the mechanisms, and by using more realistic and practical privacy settings. This contrasts with existing differentially private solutions, which are unnecessarily restrictive with regard to how they treat public information. Extensive sets of experiments show that including real-world knowledge achieves the desired levels of utility, without affecting the level of privacy. The findings of this thesis can serve as the motivation for further research and development in which public knowledge is used to design and influence private data sharing mechanisms. This would eventually help to pave the way for everyday practical use of differential privacy in the real world.

Bibliography

[1] Nazmiye Ceren Abay, Yan Zhou, Murat Kantarcioglu, Bhavani Thuraisingham, and Latanya Sweeney. Privacy Preserving Synthetic Data Release Using Deep Learning. In *ECML KDD*, pages 510–526, 2018. doi:10.1007/978-3-030-10925-7_31.

[2] Emmanuel A. Abbe, Amir E. Khandani, and Andrew W. Lo. Privacy-Preserving Methods for Sharing Financial Risk Exposures. *American Economic Review*, 102(3):65–70, 2012. doi:10.1257/aer.102.3.65.

[3] John M. Abowd. The U.S. Census Bureau Adopts Differential Privacy. In *ACM SIGKDD*, page 2867, 2018. doi:10.1145/3219819.3226070.

[4] Jayadev Acharya, Kallista Bonawitz, Peter Kairouz, Daniel Ramage, and Ziteng Sun. Context Aware Local Differential Privacy. In *ICML*, pages 52–62, 2020. doi:10.5555/3524938.3524944.

[5] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning Representations and Generative Models for 3D Point Clouds. In *ICML*, pages 40–49, 2018. URL http://proceedings.mlr.press/v80/achlioptas18a.html.

[6] Gergely Acs and Claude Castelluccia. A Case Study: Privacy Preserving Release of Spatio-Temporal Density in Paris. In *ACM SIGKDD*, pages 1679–1688, 2014. doi:10.1145/2623330.2623361.

[7] Ritesh Ahuja, Gabriel Ghinita, and Cyrus Shahabi. A Utility-Preserving and Scalable Technique for Protecting Location Data with Geo-Indistinguishability. In *EDBT*, pages 217–228, 2019. doi:10.5441/002/edbt.2019.20.

[8] Mohammad Akbari and Jie Liang. Semi-Recurrent CNN-Based VAE-GAN for Sequential Data Generation. In *IEEE ICASSP*, pages 2321–2325, 2018. doi:10.1109/ICASSP.2018.8461724.

[9] Mohammad Alaggan, Sébastien Gambs, and Anne-Marie Kermarrec. Heterogeneous Differential Privacy. 2015. doi:10.48550/ARXIV.1504.06998.

[10] Francesco Aldà and Benjamin I. P. Rubinstein. The Bernstein Mechanism: Function Release under Differential Privacy. In *AAAI*, pages 1705–1711, 2017. doi:10.5555/3298483.3298488.

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

[12] Miguel E. Andrés, Nicolás E. Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. Geo-Indistinguishability: Differential Privacy for Location-Based Systems. In *ACM SIGSAC*, pages 901–914, 2013. doi:10.1145/2508859.2516735.

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

[14] Christian Arnold and Marcel Neunhoeffer. Really Useful Synthetic Data – A Framework to Evaluate the Quality of Differentially Private Synthetic Data. 2020. doi:10.48550/ARXIV.2004.07740.

[15] Mohammad Samiul Arshad and William J. Beksi. A Progressive Conditional Generative Adversarial Network for Generating Dense and Colored 3D Point Clouds. In *International Conference on 3D Vision*, pages 712–722, 2020. doi:10.1109/3DV50981.2020.00081.

[16] Maho Asada, Masatoshi Yoshikawa, and Yang Cao. "When and Where Do You Want to Hide?" – Recommendation of Location Privacy Preferences with Local Differential Privacy". In *Data and Applications Security and Privacy XXXIII*, pages 164–176, 2019. doi:10.1007/978-3-030-22479-0_9.

[17] Shahab Asoodeh, Fady Alajaji, and Tamás Linder. Notes on Information-Theoretic Privacy. In *Allerton Conference on Communication, Control, and Computing*, pages 1272–1278, 2014. doi:10.1109/ALLERTON.2014.7028602.

[18] Shahab Asoodeh, Fady Alajaji, and Tamás Linder. On Maximal Correlation, Mutual Information and Data Privacy. 2015. doi:10.48550/ARXIV.1510.02330.

[19] Sean Augenstein, H. Brendan McMahan, Daniel Ramage, Swaroop Ramaswamy, Peter Kairouz, Mingqing Chen, Rajiv Mathews, and Blaise Agüera y Arcas. Generative Models for Effective ML on Private, Decentralized Datasets. 2019. doi:10.48550/ARXIV.1911.06679.

[20] David E. Bakken, Rupa Rarameswaran, Douglas M. Blough, Andy A. Franz, and Ty J. Palmer. Data Obfuscation: Anonymity and Desensitization of Usable Data Sets. *IEEE Security and Privacy*, 2(6):34–41, 2004. doi:10.1109/MSP.2004.97.

[21] Maria-Florina Balcan, Travis Dick, Yingyu Liang, Wenlong Mou, and Hongyang Zhang. Differentially Private Clustering in High-Dimensional Euclidean Spaces. In *ICML*, pages 322–331, 2017. doi:10.5555/3305381.3305415.

[22] Ting Bao, Lei Xu, Liehuang Zhu, Lihong Wang, and Tielei Li. Successive Point-of-Interest Recommendation with Personalized Local Differential Privacy. *IEEE Transactions on Vehicular Technology*, 70(10):10477–10488, 2021. doi:10.1109/TVT.2021.3108463.

[23] Raef Bassily, Kobbi Nissim, Uri Stemmer, and Abhradeep Thakurta. Practical Locally Private Heavy Hitters. In *NeurIPS*, pages 2285–2293, 2017. doi:10.5555/3294771.3294989.

[24] Jason Bay, Joel Kek, Alvin Tan, Chai Sheng Hau, Lai Yongquan, Janice Tan, and Tang Anh Quy. BlueTrace: A Privacy-Preserving Protocol for Community-Driven Contact Tracing Across Borders. 2020. URL https://bluetrace.io/static/bluetrace_whitepaper-938063656596c104632def383eb33b3c.pdf.

[25] Brett K. Beaulieu-Jones, Zhiwei Steven Wu, Chris Williams, Ran Lee, Sanjeev P. Bhavnani, James Brian Byrd, and Casey S. Greene. Privacy-Preserving Generative Deep Neural Networks Support Clinical Data Sharing. *Circulation: Cardiovascular Quality and Outcomes*, 12(7):e005122, 2019. doi:10.1161/CIRCOUTCOMES.118.005122.

[26] Steven M. Bellovin, Preetam K. Dutta, and Nathan Reitinger. Privacy and Synthetic Datasets. *Stanford Technology Law Review*, 22(1), 2019. URL https://law.stanford.edu/wp-content/uploads/2019/01/Bellovin_20190129.pdf.

[27] Vincent Bindschaedler and Reza Shokri. Synthesizing Plausible Privacy-Preserving Location Traces. In *IEEE Security and Privacy*, pages 546–563, 2016. doi:10.1109/SP.2016.39.

[28] Carsten Binnig, Donald Kossmann, Eric Lo, and M. Tamer Özsu. QAGen: Generating Query-Aware Test Databases. In *ACM SIGMOD*, pages 341–352, 2007. doi:10.1145/1247480.1247520.

[29] Chandan Biswas, Debasis Ganguly, Dwaipayan Roy, and Ujjwal Bhattacharya. Privacy Preserving Approximate K-means Clustering. In *ACM CIKM*, pages 1321–1330, 2019. doi:10.1145/3357384.3357969.

[30] Avrim Blum, Cynthia Dwork, Frank McSherry, and Kobbi Nissim. Practical Privacy: The SuLQ Framework. In *ACM PODS*, pages 128–138, 2005. doi:10.1145/1065167.1065184.

[31] Geoff Boeing. OSMnx: New Methods for Acquiring, Constructing, Analyzing, and Visualizing Complex Street Networks. *Computers, Environment and Urban Systems*, 65:126–139, 2017. doi:10.1016/j.compenvurbsys.2017.05.004.

[32] Geoff Boeing, Carl Higgs, Shiqin Liu, Billie Giles-Corti, James F. Sallis, Ester Cerin, Melanie Lowe, Deepti Adlakha, Erica Hinckson, Anne Vernez Moudon, Deborah Salvo, Marc A. Adams, Ligia V. Barrozo, Tamara Bozovic, Xavier Delclòs-Alió, Jan Dygrýn, Sara Ferguson, Klaus Gebel, Thanh Phuong Ho, Poh-Chin Lai, Joan C. Martori, Kornsupha Nitvimol, Ana Queralt, Jennifer D. Roberts, Garba H. Sambo, Jasper Schipperijn, David Vale, Nico Van de Weghe, Guillem Vich, and Jonathan Arundel. Using Open Data and Open-Source Software to Develop Spatial Indicators of Urban Design and Transport Features for Achieving Healthy and Sustainable Cities. *The Lancet Global Health*, 10(6):e907–e918, 2022. doi:10.1016/S2214-109X(22)00072-9.

[33] Jonas Böhler and Florian Kerschbaum. Secure Multi-Party Computation of Differentially Private Median. In *USENIX Security Symposium*, pages 2147–2164, 2020. doi:10.5555/3489212.3489333.

[34] Luca Bonomi and Li Xiong. A Two-Phase Algorithm for Mining Sequential Patterns with Differential Privacy. In *ACM CIKM*, pages 269–278, 2013. doi:10.1145/2505515.2505553.

[35] Luca Bonomi, Li Xiong, Rui Chen, and Benjamin C. M. Fung. Frequent Grams Based Embedding for Privacy Preserving Record Linkage. In *ACM CIKM*, pages 1597–1601, 2012. doi:10.1145/2396761.2398480.

[36] Nicolás E. Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. Optimal Geo-Indistinguishable Mechanisms for Location Privacy. In *ACM SIGSAC*, pages 251–262, 2014. doi:10.1145/2660267.2660345.

[37] Leonid Boytsov. Indexing Methods for Approximate Dictionary Searching: Comparative Analysis. *ACM Journal of Experimental Algorithmics*, 16:1–91, 2011. doi:10.1145/1963190.1963191.

[38] Robert Istvan Busa-Fekete, Andres Munoz Medina, Umar Syed, and Sergei Vassilvitskii. Population Level Privacy Leakage in Binary Classification wtih Label Noise. In *NeurIPS*, 2021. URL https://openreview.net/forum?id=Gf2EuAB9Xj.

[39] Lucas Caccia, Herke Van Hoof, Aaron Courville, and Joelle Pineau. Deep Generative Modeling of LiDAR Data. In *IEEE International Conference on Intelligent Robots and Systems*, pages 5034–5040, 2019. doi:10.1109/IROS40897.2019.8968535.

[40] Zhipeng Cai, Zuobin Xiong, Honghui Xu, Peng Wang, Wei Li, and Yi Pan. Generative Adversarial Networks: A Survey Toward Private and Secure Applications. *ACM Computing Surveys*, 54(6): 1–38, 2021. doi:10.1145/3459992.

[41] Flavio du Pin Calmon and Nadia Fawaz. Privacy Against Statistical Inference. 2012. doi:10.48550/ARXIV.1210.2123.

[42] Yang Cao and Masatoshi Yoshikawa. Differentially Private Real-Time Data Release over Infinite Trajectory Streams. In *IEEE MDM*, volume 2, pages 68–73, 2015. doi:10.1109/MDM.2015.15.

[43] Yang Cao, Masatoshi Yoshikawa, Yonghui Xiao, and Li Xiong. Quantifying Differential Privacy under Temporal Correlations. In *IEEE ICDE*, pages 821–832, 2017. doi:10.1109/ICDE.2017.132.

[44] Yang Cao, Yonghui Xiao, Li Xiong, and Liquan Bai. PriSTE: From Location Privacy to Spatiotemporal Event Privacy. In *ICDE*, pages 1606–1609, 2019. doi:10.1109/ICDE.2019.00153.

[45] Yang Cao, Yonghui Xiao, Li Xiong, Liquan Bai, and Masatoshi Yoshikawa. Protecting Spatiotemporal Event Privacy in Continuous Location-Based Services. *IEEE TKDE*, 33(8):3141–3154, 2021. doi:10.1109/TKDE.2019.2963312.

[46] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. 2015. doi:10.48550/ARXIV.1512.03012.

[47] Thee Chanyaswad, Changchang Liu, and Prateek Mittal. RON-Gauss: Enhancing Utility in Non-Interactive Private Data Release. *PETS*, 2019(1):26–46, 2019. doi:10.2478/popets-2019-0003.

[48] Konstantinos Chatzikokolakis, Miguel E. Andrés, Nicolás Emilio Bordenabe, and Catuscia Palamidessi. Broadening the Scope of Differential Privacy Using Metrics. *PETS*, 2013:82–102, 2013. doi:10.1007/978-3-642-39077-7.

[49] Konstantinos Chatzikokolakis, Ehab ElSalamouny, and Catuscia Palamidessi. Efficient Utility Improvement for Location Privacy. *PETS*, 2017(4):308–328, 2017. doi:10.1515/popets-2017-0051.

[50] Kamalika Chaudhuri and Daniel Hsu. Sample Complexity Bounds for Differentially Private Learning. In *Conference on Learning Theory*, pages 155–186, 2011. URL https://proceedings.mlr.press/v19/chaudhuri11a.html.

[51] Liqun Chen, Shuyang Dai, Chenyang Tao, Dinghan Shen, Zhe Gan, Haichao Zhang, Yizhe Zhang, and Lawrence Carin. Adversarial Text Generation via Feature-Mover's Distance. In *NeurIPS*, pages 4671–4682, 2018. doi:10.5555/3327345.3327377.

[52] Rui Chen, Gergely Acs, and Claude Castelluccia. Differentially Private Sequential Data Publication via Variable-Length n-Grams. In *ACM SIGSAC*, pages 638–649, 2012. doi:10.1145/2382196.2382263.

[53] Rui Chen, Benjamin C. M. Fung, Bipin C. Desai, and Nériah M. Sossou. Differentially Private Transit Data Publication: A Case Study on the Montreal Transportation System. In *ACM SIGKDD*, pages 213–221, 2012. doi:10.1145/2339530.2339564.

[54] Rui Chen, Qian Xiao, Yu Zhang, and Jianliang Xu. Differentially Private High-Dimensional Data Publication via Sampling-Based Inference. In *ACM SIGKDD*, pages 129–138, 2015. doi:10.1145/2783258.2783379.

[55] Rui Chen, Haoran Li, A. K. Qin, Shiva Prasad Kasiviswanathan, and Hongxia Jin. Private Spatial Data Aggregation in the Local Setting. In *IEEE ICDE*, pages 289–300, 2016. doi:10.1109/ICDE.2016.7498248.

[56] Xiang Cheng, Peng Tang, Sen Su, Rui Chen, Zequn Wu, and Binyuan Zhu. Multi-Party High-Dimensional Data Publishing under Differential Privacy. *IEEE TKDE*, 32(8):1557–1571, 2020. doi:10.1109/TKDE.2019.2906610.

[57] Davide Chicco, Niklas Tötsch, and Giuseppe Jurman. The Matthews Correlation Coefficient (MCC) is More Reliable than Balanced Accuracy, Bookmaker Informedness, and Markedness in Two-Class Confusion Matrix Evaluation. *BioData Mining*, 14(1), 2021. doi:10.1186/s13040-021-00244-z.

[58] Michael B. Cohen, Yin Tat Lee, and Zhao Song. Solving Linear Programs in the Current Matrix Multiplication Time. 2018. doi:10.48550/ARXIV.1810.07896.

[59] Chris Conlan, Teddy Cunningham, Gunduz Vehbi Demirci, and Hakan Ferhatosmanoglu. Collective Shortest Paths for Minimizing Congestion on Temporal Load-Aware Road Networks. In *ACM SIGSPATIAL Workshop on Computational Transportation Science*, pages 1–10, 2021. doi:10.1145/3486629.3490691.

[60] Chris Conlan, Teddy Cunningham, Samuel Watson, Jason Madan, Alex Sfyridis, Jo Sartori, Hakan Ferhatosmanoglu, and Richard Lilford. Perceived Quality of Care and Choice of Healthcare Provider in Informal Settlements. 2022. Currently under review.

[61] Graham Cormode and Akash Bharadwaj. Sample-and-Threshold Differential Privacy: Histograms and Applications . In *AISTATS*, pages 1420–1431, 2022. URL https://proceedings.mlr.press/v151/cormode22a.html.

[62] Graham Cormode, Cecilia Procopiuc, Divesh Srivastava, Entong Shen, and Ting Yu. Differentially Private Spatial Decompositions. In *IEEE ICDE*, pages 20–31, 2012. doi:10.1109/ICDE.2012.16.

[63] Graham Cormode, Somesh Jha, Tejas Kulkarni, Ninghui Li, Divesh Srivastava, and Tianhao Wang. Privacy at Scale: Local Differential Privacy in Practice. In *ACM SIGMOD*, pages 1655–1658, 2018. doi:10.1145/3183713.3197390.

[64] Graham Cormode, Tejas Kulkarni, and Divesh Srivastava. Marginal Release under Local Differential Privacy. In *ACM SIGMOD*, pages 131–146, 2018. doi:10.1145/3183713.3196906.

[65] Graham Cormode, Tejas Kulkarni, and Divesh Srivastava. Answering Range Queries under Local Differential Privacy. *PVLDB*, 12(10):1126–1138, 2019. doi:10.14778/3339490.3339496.

[66] Nan Cui, Nick Malleson, Victoria Houlden, and Alexis Comber. Using VGI and Social Media Data to Understand Urban Green Space: A Narrative Literature Review. *ISPRS International Journal of Geo-Information*, 10(7), 2021. doi:10.3390/ijgi10070425.

[67] Teddy Cunningham. Sharing and Generating Privacy-Preserving Spatio-Temporal Data Using Real-World Knowledge. In *IEEE MDM*, pages 331–333, 2022. doi:10.1109/MDM55031.2022.00074.

[68] Teddy Cunningham, Graham Cormode, and Hakan Ferhatosmanoglu. Privacy-Preserving Synthetic Location Data in the Real World. In *SSTD*, pages 23–33, 2021. doi:10.1145/3469830.3470893.

[69] 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.

[70] Teddy Cunningham, Hakan Ferhatosmanoglu, and Divesh Srivastava. Sharing and Integrating Event Sequences from Multiple Sources with Local Differential Privacy. 2022. Currently under review.

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

[72] Yves-Alexandre de Montjoye, Cesar A. Hidalgo, Michel Verleysen, and Vincent D. Blondel. Unique in the Crowd: The Privacy Bounds of Human Mobility. *Scientific Reports*, 3:1376, 2013. doi:10.1038/srep01376.

[73] Angel M. Del Rey, Xingxing Xiong, Shubo Liu, Dan Li, Zhaohui Cai, and Xiaoguang Niu. A Comprehensive Survey on Local Differential Privacy. *Security and Communication Networks*, 2020. doi:10.1155/2020/8829523.

[74] Fatemeh Deldar and Mahdi Abadi. PLDP-TD: Personalized-Location Differentially Private Data Analysis on Trajectory Databases. *Pervasive and Mobile Computing*, 49:1–22, 2018. doi:10.1016/j.pmcj.2018.06.005.

[75] Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39 (1):1–22, 1977. doi:10.1111/j.2517-6161.1977.tb01600.x.

[76] Damien Desfontaines and Balázs Pejó. SoK: Differential Privacies. *PETS*, 2020(2):288–313, 2020. doi:10.2478/popets-2020-0028.

[77] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. Collecting Telemetry Data Privately. In *NeurIPS*, pages 3574–3583, 2017. doi:10.5555/3294996.3295115.

[78] Irit Dinur and Kobbi Nissim. Revealing Information While Preserving Privacy. In *ACM SIGMOD-SIGACT-SIGART*, pages 202–210, 2003. doi:10.1145/773153.773173.

[79] Kamran Ghasedi Dizaji, Xiaoqian Wang, and Heng Huang. Semi-Supervised Generative Adversarial Network for Gene Expression Inference. In *ACM SIGKDD*, pages 1435–1444, 2018. doi:10.1145/3219819.3220114.

[80] Linkang Du, Zhikun Zhang, Shaojie Bai, Changchang Liu, Shouling Ji, Peng Cheng, and Jiming Chen. AHEAD: Adaptive Hierarchical Decomposition for Range Query under Local Differential Privacy. In *ACM SIGSAC*, pages 1266–1288, 2021. doi:10.1145/3460120.3485668.

[81] John C. Duchi, Michael I. Jordan, and Martin J. Wainwright. Local Privacy and Statistical Minimax Rates. In *IEEE FOCS*, pages 429–438, 2013. doi:10.1109/FOCS.2013.53.

[82] Cynthia Dwork and Aaron Roth. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3–4):211–407, 2014. doi:10.1561/0400000042.

[83] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating Noise to Sensitivity in Private Data Analysis. In *Theory of Cryptography*, pages 265–284, 2006. doi:10.1007/11681878_14.

[84] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N. Rothblum. Differential Privacy under Continual Observation. In *ACM STOC*, pages 715–724, 2010. doi:10.1145/1806689.1806787.

[85] Cynthia Dwork, Nitin Kohli, and Deirdre Mulligan. Differential Privacy in Practice: Expose your Epsilons! *Journal of Privacy and Confidentiality*, 9(2), 2019. doi:10.29012/jpc.689.

[86] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response. In *ACM SIGSAC*, pages 1054–1067, 2014. doi:10.1145/2660267.2660348.

[87] Fatima Zahra Errounda and Yan Liu. Continuous Location Statistics Sharing Algorithm with Local Differential Privacy. In *IEEE Big Data*, pages 5147–5152, 2018. doi:10.1109/BigData.2018.8621876.

[88] Fatima Zahra Errounda and Yan Liu. An Analysis of Differential Privacy Research in Location Data. In *IEEE International Conference on Big Data Security on Cloud, IEEE International Conference on High Performance and Smart Computing, and IEEE International Conference on Intelligent Data and Security*, pages 53–60, 2019. doi:10.1109/BigDataSecurity-HPSC-IDS.2019.00021.

[89] Fatima Zahra Errounda and Yan Liu. Collective Location Statistics Release with Local Differential Privacy. *Future Generation Computer Systems*, 124:174–186, 2021. doi:10.1016/j.future.2021.05.020.

[90] Hossein Esfandiari, Vahab Mirrokni, Umar Syed, and Sergei Vassilvitskii. Label Differential Privacy via Clustering. 2021. doi:10.48550/ARXIV.2110.02159.

[91] Dan Feldman, Amos Fiat, Haim Kaplan, and Kobbi Nissim. Private Coresets. In *ACM STOC*, pages 361–370, 2009. doi:10.1145/1536414.1536465.

[92] Joseph Ficek, Wei Wang, Henian Chen, Getachew Dagne, and Ellen Daley. Differential Privacy in Health Research: A Scoping Review. *Journal of the American Medical Informatics Association*, 28(10):2269–2276, 08 2021. doi:10.1093/jamia/ocab135.

[93] Foursquare Developers. Venue Categories, 2020. URL https://developer.foursquare.com/docs/build-with-foursquare/categories/.

[94] Lorenzo Frigerio, Anderson Santana de Oliveira, Laurent Gomez, and Patrick Duverger. Differentially Private Generative Adversarial Networks for Time Series, Continuous, and Discrete Open Data. In *ICT Systems Security and Privacy Protection*, pages 151–164, 2019. doi:10.1007/978-3-030-22312-0_11.

[95] Rinon Gal, Amit Bermano, Hao Zhang, and Daniel Cohen-Or. MRGAN: Multi-Rooted 3D Shape Generation with Unsupervised Part Disentanglement. 2020. doi:10.48550/ARXIV.2007.12944.

[96] Georgi Ganev, Bristena Oprisanu, and Emiliano De Cristofaro. Robin Hood and Matthew Effects: Differential Privacy Has Disparate Impact on Synthetic Data. In *ICML*, pages 6944–6959, 2022. URL https://proceedings.mlr.press/v162/ganev22a/ganev22a.pdf.

[97] Simson Garfinkel, John M. Abowd, and Christian Martindale. Understanding Database Reconstruction Attacks on Public Data. *Communications of the ACM*, 62(3):46–53, 2019. doi:10.1145/3287287.

[98] Anthony C. Gatrell, Trevor C. Bailey, Peter J. Diggle, and Barry S. Rowlingson. Spatial Point Pattern Analysis and Its Application in Geographical Epidemiology. *Transactions of the Institute of British Geographers*, pages 256–274, 1996. doi:10.2307/622936.

[99] Chang Ge, Shubhankar Mohapatra, Xi He, and Ihab F. Ilyas. Kamino: Constraint-Aware Differentially Private Data Synthesis. 2021. doi:10.48550/ARXIV.2012.15713.

[100] Geolink. Taxi Service Trajectory Prediction Challenge, 2015. URL https://www.geolink.pt/ecmlpkdd2015-challenge/dataset.html.

[101] Soheila Ghane, Lars Kulik, and Kotagiri Ramamohanarao. Publishing Spatial Histograms under Differential Privacy. In *SSDBM*, 2018. doi:10.1145/3221269.3223039.

[102] Ahmad Ghazal, Tilmann Rabl, Minqing Hu, Francois Raab, Meikel Poess, Alain Crolotte, and Hans-Arno Jacobsen. BigBench: Towards an Industry Standard Benchmark for Big Data Analytics. In *ACM SIGMOD*, pages 1197–1208, 2013. doi:10.1145/2463676.2463712.

[103] Badih Ghazi, Noah Golowich, Ravi Kumar, Pasin Manurangsi, and Chiyuan Zhang. Deep Learning with Label Differential Privacy. In *NeurIPS*, 2021. doi:10.48550/arXiv.2102.06062.

[104] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In *NeurIPS*, pages 2672–2680, 2014. doi:10.5555/2969033.2969125.

[105] Google. COVID-19 Community Mobility Reports, 2022. URL https://www.google.com/covid19/mobility/.

[106] Susan M. Grant-Muller, Mahmoud Abdelrazek, Hannah Budnitz, Caitlin D. Cottrill, Fiona Crawford, Charisma F. Choudhury, Teddy Cunningham, Gillian Harrison, Frances C. Hodgson, Jinhyun Hong, Adam Martin, Oliver O'Brien, Claire Papaix, and Panagiotis Tsoleridis. Technology Enabled Data for Sustainable Transport Policy. In *International Encyclopedia of Transportation*, pages 135–141. Elsevier, 2021. doi:10.1016/B978-0-08-102671-7.10627-X.

[107] Ling Gu, Minqi Zhou, Zhenjie Zhang, Ming-Chien Shan, Aoying Zhou, and Marianne Winslett. Chronos: An Elastic Parallel Framework for Stream Benchmark Generation and Simulation. In *IEEE ICDE*, pages 101–112, 2015. doi:10.1109/ICDE.2015.7113276.

[108] Xiaolan Gu, Ming Li, Yang Cao, and Li Xiong. Supporting Both Range Queries and Frequency Estimation with Local Differential Privacy. In *IEEE CNS*, pages 124–132, 2019. doi:10.1109/CNS.2019.8802778.

[109] Xiaolan Gu, Ming Li, Li Xiong, and Yang Cao. Providing Input-Discriminative Protection for Local Differential Privacy. In *IEEE ICDE*, pages 505–516, 2020. doi:10.1109/ICDE48307.2020.00050.

[110] Mehmet Emre Gursoy, Ling Liu, Stacey Truex, Lei Yu, and Wenqi Wei. Utility-Aware Synthesis of Differentially Private and Attack-Resilient Location Traces. In *ACM SIGSAC*, pages 196–211, 2018. doi:10.1145/3243734.3243741.

[111] Mehmet Emre Gursoy, Ling Liu, Stacey Truex, and Lei Yu. Differentially Private and Utility Preserving Publication of Trajectory Data. *IEEE Transactions on Mobile Computing*, 18(10): 2315–2329, 2019. doi:10.1109/TMC.2018.2874008.

[112] Mehmet Emre Gursoy, Vivekanand Rajasekar, and Ling Liu. Utility-Optimized Synthesis of Differentially Private Location Traces. 2020. doi:10.48550/ARXIV.2009.06505.

[113] Mehmet Emre Gursoy, Acar Tamersoy, Stacey Truex, Wenqi Wei, and Ling Liu. Secure and Utility-Aware Data Collection with Condensed Local Differential Privacy. *IEEE Transactions on Dependable and Secure Computing*, 18(5):2365–2378, 2021. doi:10.1109/TDSC.2019.2949041.

[114] Aparajita Haldar, Teddy Cunningham, and Hakan Ferhatosmanoglu. RAGUEL: Recourse-Aware Group Unfairness Elimination. In *ACM CIKM*, pages 666–675, 2022. doi:10.1145/3511808.3557424.

[115] Rob Hall, Alessandro Rinaldo, and Larry Wasserman. Differential Privacy for Functions and Functional Data. *JMLR*, 14:703–727, 2013. doi:10.5555/2567709.2502603.

[116] Qilong Han, Bo Shao, Lijie Li, Zhiqiang Ma, Haitao Zhang, and Xiaojiang Du. Publishing Histograms with Outliers under Data Differential Privacy. *Security and Communication Networks*, 9(14):2313–2322, 2016. doi:10.1002/sec.1493.

[117] Michael Hay, Vibhor Rastogi, Gerome Miklau, and Dan Suciu. Boosting the Accuracy of Differentially Private Histograms through Consistency. *PVLDB*, 3(1–2):1021–1032, 2010. doi:10.14778/1920841.1920970.

[118] Xi He, Ashwin Machanavajjhala, and Bolin Ding. Blowfish Privacy: Tuning Privacy-Utility Trade-Offs Using Policies. In *ACM SIGMOD*, pages 1447–1458, 2014. doi:10.1145/2588555.2588581.

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

[120] Xi He, Ashwin Machanavajjhala, Cheryl Flynn, and Divesh Srivastava. Composing Differential Privacy and Secure Computation: A Case Study on Scaling Private Record Linkage. In *ACM SIGSAC*, pages 1389–1406, 2017. doi:10.1145/3133956.3134030.

[121] Henry VIII. Henry VIII: March 1538, 11–15. In James Gairdner, editor, *Letters and Papers, Foreign and Domestic, Henry VIII, Volume 13 Part 1, January–July 1538*, pages 176–192. Her Majesty's Stationery Office, London, 1892. URL http://www.british-history.ac.uk/letters-papers-hen8/vol13/no1/pp176-192.

[122] Daeyoung Hong, Woohwan Jung, and Kyuseok Shim. Collecting Geospatial Data under Local Differential Privacy with Improving Frequency Estimation. *IEEE TKDE*, 2022. doi:10.1109/TKDE.2022.3181049. In Press.

[123] Mengdi Huai, Di Wang, Chenglin Miao, Jinhui Xu, and Aidong Zhang. Privacy-Aware Synthesizing for Crowdsourced Data. In *IJCAI*, pages 2542–2548, 2019. doi:10.24963/ijcai.2019/353.

[124] Chong Huang, Peter Kairouz, Xiao Chen, Lalitha Sankar, and Ram Rajagopal. Context-Aware Generative Adversarial Privacy. *Entropy*, 19(12), 2017. doi:10.3390/e19120656.

[125] Zhiqi Huang, Ryan McKenna, George Bissias, Gerome Miklau, Michael Hay, and Ashwin Machanavajjhala. PSynDB: Accurate and Accessible Private Data Generation. *PVLDB*, 12(12):1918–1921, 2019. doi:10.14778/3352063.3352099.

[126] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial Transformer Networks. In *NeurIPS*, pages 2017–2025, 2015. doi:10.5555/2969442.2969465.

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

[128] Bo Jiang, Mohamed Seif, Ravi Tandon, and Ming Li. Context-Aware Local Information Privacy. *IEEE Transactions on Information Forensics and Security*, 16:3694–3708, 2021. doi:10.1109/TIFS.2021.3087350.

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

[130] Noah Johnson, Joseph P. Near, and Dawn Song. Towards Practical Differential Privacy for SQL Queries. *PVLDB*, 11(5):526–539, 2018. doi:10.1145/3177732.3177733.

[131] Zach Jorgensen, Ting Yu, and Graham Cormode. Conservative or Liberal? Personalized Differential Privacy. In *IEEE ICDE*, pages 1023–1034, 2015. doi:10.1109/ICDE.2015.7113353.

[132] Matthew Joseph, Aaron Roth, Jonathan Ullman, and Bo Waggoner. Local differential privacy for evolving data. In *NeurIPS*, pages 2381–2390, 2018. doi:10.5555/3327144.3327164.

[133] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. Secure Multi-Party Differential Privacy. In *NeurIPS*, pages 2008–2016, 2015. doi:10.5555/2969442.2969464.

[134] Manohar Kaul, Bin Yang, and Christian S. Jensen. Building Accurate 3D Spatial Networks to Enable Next Generation Intelligent Transportation Systems. In *IEEE MDM*, pages 137–146, 2013. doi:10.1109/MDM.2013.24.

[135] Michael Kechinov. eCommerce Behavior Data from Multi Category Store, 2020. URL https://www.kaggle.com/mkechinov/ecommerce-behavior-data-from-multi-category-store.

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

[137] Daniel Kifer and Ashwin Machanavajjhala. A Rigorous and Customizable Framework for Privacy. In *ACM PODS*, pages 77–88, 2012. doi:10.1145/2213556.2213571.

[138] Jong Seon Kim, Yon Dohn Chung, and Jong Wook Kim. Differentially Private and Skew-Aware Spatial Decompositions for Mobile Crowdsensing. *Sensors*, 18(11), 2018. doi:10.3390/s18113696.

[139] Jong Wook Kim, Dae-Ho Kim, and Beakcheol Jang. Application of Local Differential Privacy to Collection of Indoor Positioning Data. *IEEE Access*, 6:4276–4286, 2018. doi:10.1109/ACCESS.2018.2791588.

[140] Konstantin Klemmer and Daniel B. Neill. Auxiliary-Task Learning for Geographic Data with Autoregressive Embeddings. In *ACM SIGSPATIAL*, pages 141–144, 2021. doi:10.1145/3474717.3483922.

[141] Konstantin Klemmer, Adriano Koshiyama, and Sebastian Flennerhag. Augmenting Correlation Structures in Spatial Data Using Deep Generative Models. 2019. doi:10.48550/ARXIV.1905.09796.

[142] Konstantin Klemmer, Tianlin Xu, Beatrice Acciaio, and Daniel B. Neill. SPATE-GAN: Improved Generative Modeling of Dynamic Spatio-Temporal Patterns with an Autoregressive Embedding Loss. *AAAI*, 36(4):4523–4531, 2022. doi:10.1609/aaai.v36i4.20375.

[143] Aleksandra Korolova, Krishnaram Kenthapadi, Nina Mishra, and Alexandros Ntoulas. Releasing Search Queries and Clicks Privately. In *ACM WWW*, pages 171–180, 2009. doi:10.1145/1526709.1526733.

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

[145] Szilvia Lestyán, Gergely Ács, and Gergely Biczók. In Search of Lost Utility: Private Location Data. *PETS*, 2022(3):352–372, 2022. doi:10.2478/popets-2022-0076.

[146] Bai Li, Vishesh Karwa, Aleksandra Slavković, and Rebecca Carter Steorts. A Privacy Preserving Algorithm to Release Sparse High-Dimensional Histograms. *Journal of Privacy and Confidentiality*, 8(1), 2018. doi:10.29012/jpc.657.

[147] Bing Li, Hong Zhu, and Meiyi Xie. Releasing Differentially Private Trajectories with Optimized Data Utility. *Applied Sciences*, 12(5), 2022. doi:10.3390/app12052406.

[148] Chao Li, Balaji Palanisamy, and James Joshi. Differentially Private Trajectory Analysis for Points-of-Interest Recommendation. In *IEEE Big Data Congress*, pages 49–56, 2017. doi:10.1109/BigDataCongress.2017.16.

[149] Chun-Liang Li, Manzil Zaheer, Yang Zhang, Barnabas Poczos, and Ruslan Salakhutdinov. Point Cloud GAN. 2018. doi:10.48550/ARXIV.1810.05795.

[150] Meng Li, Liehuang Zhu, Zijian Zhang, and Rixin Xu. Achieving Differential Privacy of Trajectory Data Publishing in Participatory Sensing. *Information Sciences*, 400-401:1–13, 2017. doi:10.1016/j.ins.2017.03.015.

[151] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. t-closeness: Privacy Beyond k-anonymity and l-diversity. In *IEEE ICDE*, pages 106–115, 2007. doi:10.1109/ICDE.2007.367856.

[152] Ruihui Li, Xianzhi Li, Chi Wing Fu, Daniel Cohen-Or, and Pheng Ann Heng. PU-GAN: A Point Cloud Upsampling Adversarial Network. In *IEEE ICCV*, 2019. doi:10.1109/ICCV.2019.00730.

[153] Wanjie Li, Xing Zhang, Xiaohui Li, Guanghui Cao, and Qingyun Zhang. PPDP-PCAO: An Efficient High-Dimensional Data Releasing Method with Differential Privacy Protection. *IEEE Access*, 7:176429–176437, 2019. doi:10.1109/ACCESS.2019.2957858.

[154] Changchang Liu, Supriyo Chakraborty, and Prateek Mittal. Dependence Makes You Vulnberable: Differential Privacy under Dependent Tuples. In *Network and Distributed System Security Symposium*, 2016. doi:10.14722/ndss.2016.23279.

[155] Kun Liu, Chris Giannella, and Hillol Kargupta. A Survey of Attack Techniques on Privacy-Preserving Data Perturbation Methods. In *Privacy-Preserving Data Mining: Models and Algorithms*, chapter 15, pages 359–381. Springer US, 2008. doi:10.1007/978-0-387-70992-5_15.

[156] Martin Lnenicka and Anastasija Nikiforova. Transparency-by-Design: What is the Role of Open Data Portals? *Telematics and Informatics*, 61, 2021. doi:10.1016/j.tele.2021.101605.

[157] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. 2019. doi:10.48550/ARXIV.1711.05101.

[158] Ashwin Machanavajjhala, Johannes Gehrke, Daniel Kifer, and Muthuramakrishnan Venkitasubramaniam. l-diversity: Privacy Beyond k-anonymity. In *IEEE ICDE*, 2006. doi:10.1109/ICDE.2006.1.

[159] Ashwin Machanavajjhala, Daniel Kifer, John Abowd, Johannes Gehrke, and Lars Vilhuber. Privacy: Theory Meets Practice on the Map. In *IEEE ICDE*, pages 277–286, 2008. doi:10.1109/ICDE.2008.4497436.

[160] Laassri Majid, Tatiana Zagorodnyaya, Ewan P. Plant, Svetlana Petrovskaya, Bella Bidzhieva, Zhiping Ye, Vahan Simonyan, and Konstantin Chumakov. Deep Sequencing for Evaluation of Genetic Stability of Influenza A/California/07/2009 (H1N1) Vaccine Viruses. *PLOS ONE*, 10(9), 2015. doi:10.1371/journal.pone.0138650.

[161] Mani Malek, Ilya Mironov, Karthik Prasad, Igor Shilov, and Florian Tramèr. Antipodes of Label Differential Privacy: PATE and ALIBI. 2021. doi:10.48550/ARXIV.2106.03408.

[162] Brian W. Matthews. Comparison of the Predicted and Observed Secondary Structure of T4 Phage Lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure*, 405(2):442–451, 1975. doi:10.1016/0005-2795(75)90109-9.

[163] Matthew S. Mayernik. Open Data: Accountability and Transparency. *Big Data & Society*, 4(2): 1–5, 2017. doi:10.1177/2053951717718853.

[164] Ryan McKenna and Daniel R. Sheldon. Permute-and-Flip: A New Mechanism for Differentially Private Selection. In *NeurIPS*, pages 193–203, 2020. doi:10.5555/3495724.3495741.

[165] Ryan McKenna, Gerome Miklau, Michael Hay, and Ashwin Machanavajjhala. Optimizing Error of High-Dimensional Statistical Queries under Differential Privacy. *PVLDB*, 11(10):1206–1219, 2018. doi:10.14778/3231751.3231769.

[166] Ryan McKenna, Gerome Miklau, Michael Hay, and Ashwin Machanavajjhala. HDMM: Optimizing Error of High-Dimensional Statistical Queries under Differential Privacy. 2021. doi:10.48550/ARXIV.2106.12118.

[167] Ryan McKenna, Gerome Miklau, and Daniel Sheldon. Winning the NIST Contest: A Scalable and General Approach to Differentially Private Synthetic Data. *Journal of Privacy and Confidentiality*, 11(3), 2021. doi:10.29012/jpc.778.

[168] Frank McSherry and Kunal Talwar. Mechanism Design via Differential Privacy. In *IEEE FOCS*, pages 94–103, 2007. doi:10.1109/FOCS.2007.41.

[169] Frank D. McSherry. Privacy Integrated Queries: An Extensible Platform for Privacy-Preserving Data Analysis. In *ACM SIGMOD*, pages 19–30, 2009. doi:10.1145/1559845.1559850.

[170] Ministry of Health NZ. COVID-19: Contact Tracing Locations of Interest, 2021. URL https://www.health.govt.nz/our-work/diseases-and-conditions/covid-19-novel-coronavirus/covid-19-health-advice-public/contact-tracing-covid-19/covid-19-contact-tracing-locations-interest.

[171] Nomad Mohammed, Dima Alhadidi, Benjamin C. M. Fung, and Mourad Debbabi. Secure Two-Party Differentially Private Data Release for Vertically Partitioned Data. *IEEE Transactions on Dependable and Secure Computing*, 11(1):59–71, 2014. doi:10.1109/TDSC.2013.22.

[172] Prashanth Mohan, Abhradeep Thakurta, Elaine Shi, Dawn Song, and David Culler. GUPT: Privacy Preserving Data Analysis Made Easy. In *ACM SIGMOD*, pages 349–360, 2012. doi:10.1145/2213836.2213876.

[173] Christopher Mühl and Franziska Boenisch. Personalized PATE: Differential Privacy for Machine Learning with Individual Privacy Guarantees. 2022. doi:10.48550/ARXIV.2202.10517.

[174] Elham Naghizade, Lars Kulik, Egemen Tanin, and James Bailey. Privacy- and Context-Aware Release of Trajectory Data. *ACM Transactions on Spatial Algorithms and Systems*, 6(1), 2020. doi:10.1145/3363449.

[175] Arvind Narayanan and Vitaly Shmatikov. Robust De-Anonymization of Large Sparse Datasets. In *IEEE Security and Privacy*, pages 111–125, 2008. doi:10.1109/SP.2008.33.

[176] Boel Nelson and Jenni Reuben. SoK: Chasing Accuracy and Privacy, and Catching Both in Differentially Private Histogram Publication. 2019. doi:10.48550/ARXIV.1910.14028.

[177] David F. Nettleton and Julián Salas. A Data Driven Anonymization System for Information Rich Online Social Network Graphs. *Expert Systems with Applications*, 55:87–105, 2016. doi:10.1016/j.eswa.2016.02.004.

[178] New York City Open Data. 311 Service Requests from 2010 to Present, 2020. URL https://data.cityofnewyork.us/browse?q=311.

[179] New York City Taxi and Limousine Commission. TLC Trip Record Data, 2013. URL https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page.

[180] Duc Tam Nguyen, Chaithanya Kumar Mummadi, Thi Phuong Nhung Ngo, Thi Hoai Phuong Nguyen, Laura Beggel, and Thomas Brox. SELF: Learning to Filter Noisy Labels with Self-Ensembling. 2019. doi:10.48550/ARXIV.1910.01842.

[181] Nadia Niknami, Mahdi Abadi, and Fatemeh Deldar. SpatialPDP: A Personalized Differentially Private Mechanism for Range Counting Queries over Spatial Databases. In *International Conference on Computer and Knowledge Engineering*, pages 709–715, 2014. doi:10.1109/ICCKE.2014.6993414.

[182] Ben Niu, Yahong Chen, Boyang Wang, Jin Cao, and Fenghua Li. Utility-Aware Exponential Mechanism for Personalized Differential Privacy. In *IEEE Wireless Communications and Networking Conference*, pages 1–6, 2020. doi:10.1109/WCNC45663.2020.9120532.

[183] Citibike NYC. Citibike NYC System Data, 2020. URL https://ride.citibikenyc.com/system-data.

[184] OpenStreetMap Contributors. OpenStreetMap, 2022. URL https://www.openstreetmap.org/.

[185] Inkit Padhi, Yair Schiff, Igor Melnyk, Mattia Rigotti, Youssef Mroueh, Pierre Dognin, Jerret Ross, Ravi Nair, and Erik Altman. Tabular Transformers for Modeling Multivariate Time Series. In *ICASSP*, pages 3565–3569, 2021. doi:10.1109/ICASSP39728.2021.9414142.

[186] Aditya Pal, Abhilash Barigidad, and Abhijit Mustafi. Identifying Movie Genre Compositions Using Neural Networks and Introducing GenRec – a Recommender System Based on Audience Genre Perception. In *International Conference on Computing, Communication and Security*, pages 1–7, 2020. doi:10.1109/ICCCS49678.2020.9276893.

[187] Manas Pathak, Shantanu Rane, and Bhiksha Raj. Multiparty Differential Privacy via Aggregation of Locally Trained Classifiers. In *NeurIPS*, pages 1876–1884, 2010. doi:10.5555/2997046.2997105.

[188] Charles A. Phillips, Kai Wang, Erich J. Baker, Jason A. Bubier, Elissa J. Chesler, and Michael A. Langston. On Finding and Enumerating Maximal and Maximum k-Partite Cliques in k-Partite Graphs. *Algorithms*, 12(1), 2019. doi:10.3390/a12010023.

[189] Michal Piorkowski, Natasa Sarafijanovic-Djukic, and Matthias Grossglauser. CRAWDAD Dataset epfl/mobility (v. 2009-02-24). Downloaded from https://crawdad.org/epfl/mobility/20090224, 2009.

[190] David M. W. Powers. Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness and Correlation. 2020. doi:10.48550/ARXIV.2010.16061.

[191] Wahbeh Qardaji, Weining Yang, and Ninghui Li. Differentially Private Grids for Geospatial Data. In *IEEE ICDE*, pages 757–768, 2013. doi:10.1109/ICDE.2013.6544872.

[192] Wahbeh Qardaji, Weining Yang, and Ninghui Li. Understanding Hierarchical Methods for Differentially Private Histograms. *PVLDB*, 6(14):1954–1965, 2013. doi:10.14778/2556549.2556576.

[193] Wahbeh Qardaji, Weining Yang, and Ninghui Li. PriView: Practical Differentially Private Release of Marginal Contingency Tables. In *ACM SIGMOD*, pages 1435–1446, 2014. doi:10.1145/2588555.2588575.

[194] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *IEEE CVPR*, pages 77–85, 2017. doi:10.1109/CVPR.2017.16.

[195] Fang-Yu Rao, Jianneng Cao, Elisa Bertino, and Murat Kantarcioglu. Hybrid Private Record Linkage: Separating Differentially Private Synopses from Matching Records. *ACM Transactions on Privacy and Security*, 22(3), 2019. doi:10.1145/3318462.

[196] Xuebin Ren, Chia-Mu Yu, Weiren Yu, Shusen Yang, Xinyu Yang, Julie A. McCann, and Philip S. Yu. LoPub: High-Dimensional Crowdsourced Data Publication with Local Differential Privacy. *IEEE Transactions on Information Forensics and Security*, 13(9):2151–2166, 2018. doi:10.1109/TIFS.2018.2812146.

[197] Xuebin Ren, Liang Shi, Weiren Yu, Shusen Yang, Cong Zhao, and Zongben Xu. LDP-IDS: Local Differential Privacy for Infinite Data Streams. In *ACM SIGMOD*, pages 1064–1077, 2022. doi:10.1145/3514221.3526190.

[198] Yanbing Ren, Xinghua Li, Yinbin Miao, Robert Deng, Jian Weng, Siqi Ma, and Jianfeng Ma. DistPreserv: Maintaining User Distribution for Privacy-Preserving Location-Based Services. *IEEE Transactions on Mobile Computing*, 2022. doi:10.1109/TMC.2022.3141398. In Press.

[199] Safegraph. Safegraph Places Schema, 2020. URL https://docs.safegraph.com/v4.0/docs/places-schema.

[200] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved Techniques for Training GANs. In *NeurIPS*, pages 2234–2242, 2016. doi:10.5555/3157096.3157346.

[201] Muhammad Sarmad, Hyunjoo Jenny Lee, and Young Min Kim. RL-GAN-Net: A Reinforcement Learning Agent Controlled GAN Network for Real-Time Point Cloud Shape Completion. In *IEEE CVPR*, pages 5891–5900, 2019. doi:10.1109/CVPR.2019.00605.

[202] Sina Shaham, Gabriel Ghinita, Ritesh Ahuja, John Krumm, and Cyrus Shahabi. HTF: Homogeneous Tree Framework for Differentially-Private Release of Location Data. In *ACM SIGSPATIAL*, pages 184–194, 2021. doi:10.1145/3474717.3483943.

[203] Sina Shaham, Gabriel Ghinita, and Cyrus Shahabi. Differentially-Private Publication of Origin-Destination Matrices with Intermediate Stops. In *EDBT*, pages 131–142, 2022. doi:10.48786/edbt.2022.04.

[204] Dongwook Shu, Sung Woo Park, and Junseok Kwon. 3D Point Cloud Generative Adversarial Network Based on Tree Structured Graph Convolutions. In *IEEE ICCV*, pages 3858–3867, 2019. doi:10.1109/ICCV.2019.00396.

[205] Theresa Stadler, Bristena Oprisanu, and Carmela Troncoso. Synthetic Data – Anonymisation Groundhog Day. In *USENIX Security Symposium*, pages 1451–1468, 2022. URL https://www.usenix.org/system/files/sec22summer_stadler.pdf.

[206] Dong Su, Jianneng Cao, Ninghui Li, Elisa Bertino, and Hongxia Jin. Differentially Private K-Means Clustering. In *ACM CODASPY*, pages 26–37, 2016. doi:10.1145/2857705.2857708.

[207] Dong Su, Jianneng Cao, Ninghui Li, Elisa Bertino, Min Lyu, and Hongxia Jin. Differentially Private K-Means Clustering and a Hybrid Approach to Private Optimization. *ACM Transactions on Privacy and Security*, 20(4), 2017. doi:10.1145/3133201.

[208] Sen Su, Peng Tang, Xiang Cheng, Rui Chen, and Zequn Wu. Differentially Private Multi-Party High-Dimensional Data Publishing. In *IEEE ICDE*, pages 205–216, 2016. doi:10.1109/ICDE.2016.7498241.

[209] Lichao Sun, Jianwei Qian, Xun Chen, and Philip S. Yu. LDP-FL: Practical Private Aggregation in Federated Learning with Local Differential Privacy. 2020. doi:10.48550/ARXIV.2007.15789.

[210] Latanya Sweeney. Weaving Technology and Policy Together to Maintain Confidentiality. *Journal of Law, Medicine & Ethics*, 25(2-3):98–110, 1997. doi:10.1111/j.1748-720X.1997.tb01885.x.

[211] Latanya Sweeney. k-anonymity: A Model for Protecting Privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, 2002. doi:10.1142/S0218488502001648.

[212] Rong Tan, Yuan Tao, Wen Si, and Yuan-Yuan Zhang. Privacy Preserving Semantic Trajectory Data Publishing for Mobile Location-Based Services. *Wireless Networks*, 26(8):5551–5560, 2020. doi:10.1007/s11276-019-02058-8.

[213] Peng Tang, Rui Chen, Sen Su, Shanqing Guo, Lei Ju, and Gaoyuan Liu. Differentially Private Publication of Multi-Party Sequential Data. In *IEEE ICDE*, pages 145–156, 2021. doi:10.1109/ICDE51399.2021.00020.

[214] Peng Tang, Xiang Cheng, Sen Su, Rui Chen, and Huaxi Shao. Differentially Private Publication of Vertically Partitioned Data. *IEEE Transactions on Dependable and Secure Computing*, 18(2): 780–795, 2021. doi:10.1109/TDSC.2019.2905237.

[215] Stuart A. Thompson and Charlie Warzel. They Stormed the Capitol. Their Apps Tracked Them., 2021. URL https://www.nytimes.com/2021/02/05/opinion/capitol-attack-cellphone-data.html.

[216] Cenxi Tian, Hongyun Xu, Tao Lu, Rui Jiang, and Yong Kuang. Semantic and Trade-Off Aware Location Privacy Protection in Road Networks via Improved Multi-Objective Particle Swarm Optimization. *IEEE Access*, 9:54264–54275, 2021. doi:10.1109/ACCESS.2021.3071407.

[217] Hien To, Gabriel Ghinita, and Cyrus Shahabi. A Framework for Protecting Worker Location Privacy in Spatial Crowdsourcing. *PVLDB*, 7(10):919–930, 2014. doi:10.14778/2732951.2732966.

[218] Hien To, Liyue Fan, and Cyrus Shahabi. Differentially Private *h*-Tree. In *ACM Workshop on Privacy in Geographic Information Collection and Analysis*, 2015. doi:10.1145/2830834.2830837.

[219] Amirsina Torfi, Edward A. Fox, and Chandan K. Reddy. Differentially Private Synthetic Medical Data Generation using Convolutional GANs. *Information Sciences*, 586:485–500, 2022. doi:10.1016/j.ins.2021.12.018.

[220] Reihaneh Torkzadehmahani, Peter Kairouz, and Benedict Paten. DP-CGAN: Differentially Private Synthetic Data and Label Generation. In *IEEE CVPR*, pages 98–104, 2019. doi:10.1109/CVPRW.2019.00018.

[221] Emina Torlak. Scalable Test Data Generation from Multidimensional Models. In *ACM SIGSOFT International Symposium on the Foundations of Software Engineering*, 2012. doi:10.1145/2393596.2393637.

[222] Carmela Troncoso, Mathias Payer, Jean-Pierre Hubaux, Marcel Salathé, James Larus, Edouard Bugnion, Wouter Lueks, Theresa Stadler, Apostolos Pyrgelis, Daniele Antonioli, Ludovic Barman, Sylvain Chatel, Kenneth Paterson, Srdjan Čapkun, David Basin, Jan Beutel, Dennis Jackson, Marc Roeschlin, Patrick Leu, Bart Preneel, Nigel Smart, Aysajan Abidin, Seda Gürses, Michael Veale, Cas Cremers, Michael Backes, Nils Ole Tippenhauer, Reuben Binns, Ciro Cattuto, Alain Barrat, Dario Fiore, Manuel Barbosa, Rui Oliveira, and José Pereira. Decentralized Privacy-Preserving Proximity Tracing. 2020. doi:10.48550/ARXIV.2005.12273.

[223] Keenan Trotter. Public NYC Taxicab Database Lets You See How Celebrities Tip, 2014. URL https://www.gawker.com/the-public-nyc-taxicab-database-that-accidentally-track-1646724546.

[224] Stacey Truex, Ling Liu, Ka-Ho Chow, Mehmet Emre Gursoy, and Wenqi Wei. LDP-Fed: Federated Learning with Local Differential Privacy. In *ACM International Workshop on Edge Systems, Analytics and Networking*, pages 61–66, 2020. doi:10.1145/3378679.3394533.

[225] University of British Columbia. ubcv-buildings, 2016. URL https://github.com/UBCGeodata/ubcv-buildings.

[226] US Census Bureau. North American Industry Classification System, 2020. URL https://www.census.gov/naics/.

[227] US Census Bureau. Hierarchy Diagrams, 2021. URL https://www.census.gov/programs-surveys/geography/guidance/hierarchy.html.

[228] US Census Bureau. 2020 Decennial Census: Processing the Count: Disclosure Avoidance Modernization, 2022. URL https://www.census.gov/programs-surveys/decennial-census/decade/2020/planning-management/process/disclosure-avoidance.html.

[229] Jan van den Brand. A Deterministic Linear Program Solver in Current Matrix Multiplication Time. In *ACM-SIAM SODA*, pages 259–278, 2020. doi:10.5555/3381089.3381105.

[230] Eduardo Velázquez, Isabel Martínez, Stephan Getzin, Kirk A. Moloney, and Thorsten Wiegand. An Evaluation of the State of Spatial Point Pattern Analysis in Ecology. *Ecography*, 39(11): 1042–1055, 2016. doi:10.1111/ecog.01579.

[231] Di Wang and Jinhui Xu. On Sparse Linear Regression in the Local Differential Privacy Model. *IEEE Transactions on Information Theory*, 67(2):1182–1200, 2021. doi:10.1109/TIT.2020.3040406.

[232] Guozhang Wang, Johannes Gehrke, and Xiaokui Xiao. Differential Privacy via Wavelet Transforms. *IEEE TKDE*, 23(8):1200–1214, 2011. doi:10.1109/TKDE.2010.247.

[233] Hao Wang, Zhengquan Xu, Shan Jia, Ying Xia, and Xu Zhang. Why Current Differential Privacy Schemes Are Inapplicable For Correlated Data Publishing? *World Wide Web*, 24(1):1–23, 2021. doi:10.1007/s11280-020-00825-8.

[234] Jian Wang, Yanli Wang, Guosheng Zhao, and Zhongnan Zhao. Location Protection Method for Mobile Crowd Sensing Based on Local Differential Privacy Preference. *Peer-to-Peer Networking and Applications*, 12(5):1097–1109, 2019. doi:10.1007/s12083-019-00774-8.

[235] Leye Wang, Daqing Zhang, Dingqi Yang, Brian Y. Lim, and Xiaojuan Ma. Differential Location Privacy for Sparse Mobile Crowdsensing. In *IEEE ICDM*, pages 1257–1262, 2016. doi:10.1109/ICDM.2016.0169.

[236] Leye Wang, Dingqi Yang, Xiao Han, Tianben Wang, Daqing Zhang, and Xiaojuan Ma. Location Privacy-Preserving Task Allocation for Mobile Crowdsensing with Differential Geo-Obfuscation. In *ACM WWW*, pages 627–636, 2017. doi:10.1145/3038912.3052696.

[237] Leye Wang, Daqing Zhang, Dingqi Yang, Brian Y. Lim, Xiao Han, and Xiaojuan Ma. Sparse Mobile Crowdsensing with Differential and Distortion Location Privacy. *IEEE Transactions on Information Forensics and Security*, 15:2735–2749, 2020. doi:10.1109/TIFS.2020.2975925.

[238] Ning Wang, Xiaokui Xiao, Yin Yang, Ta Duy Hoang, Hyejin Shin, Junbum Shin, and Ge Yu. PrivTrie: Effective Frequent Term Discovery under Local Differential Privacy. In *IEEE ICDE*, pages 821–832, 2018. doi:10.1109/ICDE.2018.00079.

[239] Shuo Wang, Carsten Rudolph, Surya Nepal, Marthie Grobler, and Shangyu Chen. PART-GAN: Privacy-Preserving Time-Series Sharing. In *Artificial Neural Networks and Machine Learning*, pages 578–593, 2020. doi:10.1007/978-3-030-61609-0_46.

[240] Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha. Locally Differentially Private Protocols for Frequency Estimation. In *USENIX Security Symposium*, pages 729–745, 2017. doi:10.5555/3241189.3241247.

[241] Tianhao Wang, Ninghui Li, and Somesh Jha. Locally Differentially Private Frequent Itemset Mining. In *IEEE Security and Privacy*, pages 127–143, 2018. doi:10.1109/SP.2018.00035.

[242] Tianhao Wang, Bolin Ding, Jingren Zhou, Cheng Hong, Zhicong Huang, Ninghui Li, and Somesh Jha. Answering Multi-Dimensional Analytical Queries under Local Differential Privacy. In *ACM SIGMOD*, pages 159–176, 2019. doi:10.1145/3299869.3319891.

[243] Tianhao Wang, Joann Qiongna Chen, Zhikun Zhang, Dong Su, Yueqiang Cheng, Zhou Li, Ninghui Li, and Somesh Jha. Continuous Release of Data Streams under Both Centralized and Local Differential Privacy. In *ACM SIGSAC*, pages 1237–1253, 2021. doi:10.1145/3460120.3484750.

[244] Weina Wang, Lei Ying, and Junshan Zhang. On the Relation Between Identifiability, Differential Privacy, and Mutual-Information Privacy. *IEEE Transactions on Information Theory*, 62(9):5018–5029, 2016. doi:10.1109/TIT.2016.2584610.

[245] Stanley L. Warner. Randomized Response: A Survey Technique for Eliminating Evasive Answer Bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965. doi:10.1080/01621459.1965.10480775.

[246] Eric W. Weisstein. Triangle Point Picking, 2021. URL https://mathworld.wolfram.com/TrianglePointPicking.html.

[247] Jin-Long Wu, Karthik Kashinath, Adrian Albert, Dragos Chirila, Prabhat, and Heng Xiao. Enforcing Statistical Constraints in Generative Adversarial Networks for Modeling Chaotic Dynamical Systems. *Journal of Computational Physics*, 406, 2020. doi:10.1016/j.jcp.2019.109209.

[248] Yonghui Xiao and Li Xiong. Protecting Locations with Differential Privacy under Temporal Correlations. In *ACM SIGSAC*, pages 1298–1309, 2015. doi:10.1145/2810103.2813640.

[249] Yonghui Xiao, Li Xiong, and Chun Yuan. Differentially Private Data Release Through Multidimensional Partitioning. In *VLDB Conference on Secure Data Management*, pages 150–168, 2010.

[250] Yonghui Xiao, Li Xiong, Si Zhang, and Yang Cao. LocLok: Location Cloaking with Differential Privacy via Hidden Markov Model. *PVLDB*, 10(12):1901–1904, 2017. doi:10.14778/3137765.3137804.

[251] Lingxi Xie, Jingdong Wang, Zhen Wei, Meng Wang, and Qi Tian. DisturbLabel: Regularizing CNN on the Loss Layer. In *IEEE CVPR*, pages 4753–4762, 2016. doi:10.1109/CVPR.2016.514.

[252] Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially Private Generative Adversarial Network. 2018. doi:10.48550/ARXIV.1802.06739.

[253] Xingxing Xiong, Shubo Liu, Dan Li, Jun Wang, and Xiaoguang Niu. Locally Differentially Private Continuous Location Sharing with Randomized Response. *International Journal of Distributed Sensor Networks*, 15(8), 2019. doi:10.1177/1550147719870379.

[254] Xingxing Xiong, Shubo Liu, Dan Li, Zhaohui Cai, and Xiaoguang Niu. Real-Time and Private Spatio-Temporal Data Aggregation with Local Differential Privacy. *Journal of Information Security and Applications*, 55, 2020. doi:10.1016/j.jisa.2020.102633.

[255] Chugui Xu, Ju Ren, Yaoxue Zhang, Zhan Qin, and Kui Ren. DPPro: Differentially Private High-Dimensional Data Release via Random Projection. *IEEE Transactions on Information Forensics and Security*, 12(12):3081–3093, 2017. doi:10.1109/TIFS.2017.2737966.

[256] Jia Xu, Zhenjie Zhang, Xiaokui Xiao, Yin Yang, and Ge Yu. Differentially Private Histogram Publication. In *IEEE ICDE*, pages 32–43, 2012. doi:10.1109/ICDE.2012.48.

[257] Min Xu, Bolin Ding, Tianhao Wang, and Jingren Zhou. Collecting and Analyzing Data Jointly from Multiple Services under Local Differential Privacy. *PVLDB*, 13(12):2760–2772, 2020. doi:10.14778/3407790.3407859.

[258] Tianlin Xu, Li K. Wenliang, Michael Munn, and Beatrice Acciaio. COT-GAN: Generating Sequential Data via Causal Optimal Transport. 2020. doi:10.48550/ARXIV.2006.08571.

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

[260] Jianyu Yang, Tianhao Wang, Ninghui Li, Xiang Cheng, and Sen Su. Answering Multi-Dimensional Range Queries under Local Differential Privacy. *PVLDB*, 14(3):378–390, 2020. doi:10.14778/3430915.3430927.

[261] Mengmeng Yang, Lingjuan Lyu, Jun Zhao, Tianqing Zhu, and Kwok-Yan Lam. Local Differential Privacy and Its Applications: A Comprehensive Survey. 2020. doi:10.48550/ARXIV.2008.03686.

[262] Qifan Yang, Yuanfang Chen, Mohsen Guizani, and Gyu Myoung Lee. Spatiotemporal Location Differential Privacy for Sparse Mobile Crowdsensing. In *International Wireless Communications and Mobile Computing*, pages 1734–1741, 2021. doi:10.1109/IWCMC51323.2021.9498951.

[263] Zeng Yang, Jin-Long Wu, and Heng Xiao. Enforcing Deterministic Constraints on Generative Adversarial Networks for Emulating Physical Systems. 2019. doi:10.48550/ARXIV.1911.06671.

[264] Lin Yao, Zhenyu Chen, Haibo Hu, Guowei Wu, and Bin Wu. Privacy Preservation for Trajectory Publication Based on Differential Privacy. *ACM Transactions on Intelligent Systems and Technology*, 13(3), 2022. doi:10.1145/3474839.

[265] Yutong Ye, Min Zhang, and Dengguo Feng. Collecting Spatial Data under Local Differential Privacy. In *International Conference on Mobility, Sensing and Networking*, pages 120–127, 2021. doi:10.1109/MSN53354.2021.00032.

[266] Emre Yilmaz, Sanem Elbasi, and Hakan Ferhatosmanoglu. Predicting Optimal Facility Location Without Customer Locations. In *ACM SIGKDD*, pages 2121–2130, 2017. doi:10.1145/3097983.3098198.

[267] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. PATE-GAN: Generating Synthetic Data with Differential Privacy Guarantees. In *ICLR*, 2019. URL https://openreview.net/forum?id=S1zk9iRqF7.

[268] Jing Yuan, Yu Zheng, Chengyang Zhang, Wenlei Xie, Xing Xie, Guangzhong Sun, and Yan Huang. T-Drive: Driving Directions Based on Taxi Trajectories. In *ACM SIGSPATIAL*, pages 99–108, 2010. doi:10.1145/1869790.1869807.

[269] Jing Yuan, Yu Zheng, Xing Xie, and Guangzhong Sun. Driving with Knowledge from the Physical World. In *ACM SIGKDD*, pages 316–324, 2011. doi:10.1145/2020408.2020462.

[270] Sen Yuan, Milan Shen, Ilya Mironov, and Anderson Nascimento. Label Private Deep Learning Training Based on Secure Multiparty Computation and Differential Privacy. In *NeurIPS Workshop on Privacy in Machine Learning*, 2021. URL https://openreview.net/forum?id=tg9W8YAJVO6.

[271] Sepanta Zeighami, Ritesh Ahuja, Gabriel Ghinita, and Cyrus Shahabi. A Neural Database for Differentially Private Spatial Range Queries. *PVLDB*, 15(5):1066–1078, 2022. doi:10.14778/3510397.3510404.

[272] Jun Zhang, Xiaokui Xiao, Yin Yang, Zhenjie Zhang, and Marianne Winslett. PrivGene: Differentially Private Model Fitting Using Genetic Algorithms. In *ACM SIGMOD*, pages 665–676, 2013. doi:10.1145/2463676.2465330.

[273] Jun Zhang, Xiaokui Xiao, and Xing Xie. PrivTree: A Differentially Private Algorithm for Hierarchical Decompositions. In *ACM SIGMOD*, pages 155–170, 2016. doi:10.1145/2882903.2882928.

[274] Jun Zhang, Graham Cormode, Cecilia M. Procopiuc, Divesh Srivastava, and Xiaokui Xiao. PrivBayes: Private Data Release via Bayesian Networks. *ACM Transactions on Database Systems*, 42(4), 2017. doi:10.1145/3134428.

[275] Mengxiao Zhang, Fernando Beltran, and Jiamou Liu. Selling Data at an Auction under Privacy Constraints. In *Conference on Uncertainty in Artificial Intelligence*, pages 669–678. PMLR, 2020. URL https://proceedings.mlr.press/v124/zhang20b.html.

[276] Yingxue Zhang, Yanhua Li, Xun Zhou, Xiangnan Kong, and Jun Luo. Curb-GAN: Conditional Urban Traffic Estimation Through Spatio-Temporal Generative Adversarial Networks. In *ACM SIGKDD*, pages 842–852, 2020. doi:10.1145/3394486.3403127.

[277] Xiangguo Zhao, Yanhui Li, Ye Yuan, Xin Bi, and Guoren Wang. LDPart: Effective Location-Record Data Publication via Local Differential Privacy. *IEEE Access*, 7:31435–31445, 2019. doi:10.1109/ACCESS.2019.2899099.

[278] Hang Zhou, Dongdong Chen, Jing Liao, Kejiang Chen, Xiaoyi Dong, Kunlin Liu, Weiming Zhang, Gang Hua, and Nenghai Yu. LG-GAN: Label Guided Adversarial Network for Flexible Targeted Attack of Point Cloud Based Deep Networks. In *IEEE CVPR*, pages 10353–10362, 2020. doi:10.1109/CVPR42600.2020.01037.

[279] Renguang Zuo, Frederik P. Agterberg, Qiuming Cheng, and Lingqing Yao. Fractal Characterization of the Spatial Distribution of Geological Point Processes. *International Journal of Applied Earth Observation and Geoinformation*, pages 394–402, 2009. doi:10.1016/j.jag.2009.07.001.