PRIVACY-PRESERVING SEEDBASED DATA SYNTHESIS

BY

VINCENT BINDSCHADLER

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2018

Urbana, Illinois

Abstract

How can we share sensitive datasets in such a way as to maximize utility while simultaneously safeguarding privacy? This thesis proposes an answer to this question by re-framing the problem of sharing sensitive datasets as a data synthesis task. Specifically, we propose a framework to synthesize full data records in a privacy-preserving way so that they can be shared *instead* of the original sensitive data.

The core the framework is a technique called *seedbased data synthesis*. Seedbased data synthesis produces data records by conditioning the output of a generative model on some input data record called the *seed*. This technique produces synthetic records that are similar to their seeds, which results in high quality outputs. But it simultaneously introduces statistical dependence between synthetic records and their seeds, which may compromise privacy. As a countermeasure, we introduce a new class of techniques that can achieve strong privacy notions in this setting: *privacy tests*. Privacy tests are algorithms that probabilistically reject candidate synthetics records which are determined to leak sensitive information. Synthetic records that fail the test are simply discarded, whereas those that pass the test are deemed safe and included in the synthetic dataset to be shared. We design two privacy tests that provably yield differential privacy.

We analyze the quality of synthetic datasets based on a cryptography-inspired definition of distinguishability: if synthetic data records are indistinguishable from real records, then they are (by definition) as useful as real data. On the theory front, we characterize the utility-privacy trade-off of seedbased data synthesis. On the experimental front, we design an efficient procedure to experimentally quantify distinguishability.

We experimentally validate the seedbased data synthesis framework using five probabilistic generative models. Specifically, using real-world datasets as input, we produce synthetic data records for four different application scenarios and data types: location trajectories, census microdata, medical data, and facial images. We evaluate the quality of the produced synthetic records using both application-dependent utility metrics and distinguishability, and show that the framework is capable of producing highly realistic synthetic data records while providing differential privacy for conservative parameters.

To my parents, for their love and support.

Acknowledgments

First, I want to express my deepest gratitude to my Ph.D. advisor, Carl Gunter, for his continuous guidance and mentorship. I am especially grateful to Carl for encouraging me to strive to be an independent researcher and for the resulting opportunities that this has afforded me. I have found Carl to be an invaluable resource for advice regarding the execution of research and career advancement.

Second, I am indebted to my committee members: Carl, Chengxiang Zhai, Nikita Borisov, and Adam Smith, for their valuable feedback. More than anything else, their comments and suggestions on earlier versions of this work have led me to change my perspective and thinking about key aspects of this work.

Third, I want to thank the brilliant researchers that I have had the pleasure to work with as a graduate student, including Carl, Reza Shokri, XiaoFeng Wang, Vitaly Shmatikov, Tom Ristenpart, Paul Grubbs, Yunhui Long, Yangyi Chen, Yan Huang, Xiaorui Pan, Wenhao Wang, Haixu Tang, and Shantanu Rane. I have learned most of what I know about conducting research from them and with them. In particular, I wish to thank Reza who is my longest-standing collaborator, Vitaly for hosting me at Cornell Tech for a summer internship, and XiaoFeng for working with me on a diverse set of projects over the years. I also owe a special thank you to my M.Sc. advisor Jean-Pierre Hubaux and my colleagues during my time at EPFL, including Reza, Nevena Vratonjic, and Kévin Huguenin, for introducing me to security and privacy research and encouraging me to pursue a Ph.D.

Fourth, I am thankful for my many Illinois Security Lab labmates over the years for creating and maintaining a congenial working environment. In particular, I would like to thank Muhammad Naveed, Soteris Demetriou, Avesta Hojjati, Yunhui Long, Ravinder Shankesi, Aston Zhang, Güliz Tuncay, Ting Wu, Gaurav Lahoti, and Siddharth Gupta, for inspiring discussions about computer science research and life beyond it.

Finally, I wish to thank my wonderful girlfriend Winnie for her patience.

Table of Contents

Chapter 1: Introduction

A major challenge in the big data era is sharing sensitive data. Suppose we wish to share a database of electronic medical records in a way that safeguards patients' privacy. Why not simply remove individually identifiable information from the database before sharing the rest? This procedure is called anonymization or de-identification and it is mandated by regulations such as the Health Insurance Portability and Accountability Act (HIPAA) and the more recent European General Data Protection Regulation (GDPR).

Unfortunately, proper anonymization of data is difficult to achieve. In 2001, Latanya Sweeney demonstrated the recovery of the medical record of William Weld, governor of the state of Massachusetts at the time, from "anonymized" medical data released by the Group Insurance Commission [1]. Since then, catastrophic incidents have occurred to companies such as AOL [2], and Netflix [3, 4] who attempted analogous releases of purportedly anonymized data. Similar issues have arisen with location data [5, 6], genomic data [7, 8], and medical data [9, 10]. To this day, anonymizing data in a way that guarantees privacy but does not overly degrade its quality has not yet found a satisfying solution.

This thesis proposes a new approach to the problem of sharing sensitive datasets. This new approach re-frames the problem as a data synthesis task: create a synthetic dataset that is similar to the original sensitive dataset (but not the same) and share it instead. We develop a framework to perform data synthesis in a privacy-preserving way using a probabilistic generative model given as a black-box. The idea of data synthesis for privacy is not new. It dates back to 1993 when it was proposed by Rubin [11]. What is new is that the framework performs a particular kind of data synthesis which we call *seedbased* and is inspired and compatible with state-of-the-art generative models based on neural-network architectures. This is in contrast to all prior techniques for privacy-preserving data synthesis which we call *seedless* because they essentially reduce to sampling from a probability distribution with parameters learned in a privacy-preserving way. Seedbased data synthesis produces synthetic records from generative models by constraining their output on some *seed* record given as input. As a result, synthesis is not performed by sampling from the distribution induced by the generative model but rather by sampling from its distribution conditioned on the seed (in the sense of conditional probability). This gives greater control on the output of the model and thus better outputs, but the dependence on the seed induces a privacy leak. To address this leak, we propose a novel class of techniques that yield provable privacy guarantees: privacy tests. Instead of noising the generative model, as is typical in data privacy, we add a privacy test that achieves privacy through rejection sampling.

1.1 OVERVIEW

This thesis proposes an answer to the following question.

> *How can we share sensitive datasets in such a way as to maximize utility while simultaneously safeguarding privacy?*

By re-framing the data sharing problem as a data synthesis task, our framework aims to produce a synthetic dataset such that the synthetic records:

(a) do not reveal sensitive information about the data from which they are generated, and
(b) preserve the utility, i.e., statistical properties, of the original data.

***Re-framing data sharing as data synthesis*.** Instead of trying to anonymize a sensitive dataset directly, we can synthesize *full* data records similar to those in the sensitive dataset. The challenge in synthesizing data, especially high-dimensional data, is to provide both privacy and utility guarantees. That is: certain sensitive facts about the sensitive data should not be revealed by the synthetic data (no matter what), and certain data analysis tasks should perform (almost) as well on the synthetic data as on the sensitive dataset.

***The benefits of synthetic data*.** In the early days of theoretically sound data privacy research the idea of synthetic data fell out of favor. Instead, the focus shifted to an interactive paradigm: analysts ask queries and get answers about a dataset without ever having direct access to data records [12]. Today, this paradigm is increasingly at odds with modern workflows of data analysis because it assumes that the analyst has precise questions to ask *before* looking at the data. For centuries, this was true of the scientific method: a specific hypothesis was formulated, an experiment designed, and an answer was produced after collecting and analyzing the data. In the age of big data, questions often emerge only after looking at the data. In their 2013 book on Big Data [13], Viktor Mayer-Schönberger and Kenneth Cukier compare the process of a big data investigation to a fishing expedition: "it is unclear at the outset not only whether one will catch anything but what one may catch." Data analysts often need the flexibility of accessing full data records before they can formulate a question. As a result, tasks such as early-stage data exploration benefit tremendously from having a synthetic dataset of full data records with an identical format to the original (sensitive) dataset, instead of query-based access to a statistical database.

***A new data synthesis technique*.** Seedbased data synthesis is a new idea introduced and formalized in this thesis. It is inspired by recent development of generative models based on neural networks such as [14–19] and the realization that such models can be used to produce outputs similar to a given input by conditioning the output distribution on the input [20–23]. This is the essence of seedbased synthesis: one takes a point or record as input (called the *seed*) and use it to produce a similar *synthetic* point or record as output. The intuition is that given an example of anything, it is easy (for humans at least) to recognize a pattern and then generate a similar example with the same or a similar pattern. Unfortunately, the dependence of the synthetic outputs on their seeds is a double-edged sword. From a data quality perspective, the process produces realistic synthetic outputs that are highly similar to their seeds (by design). But from a privacy perspective, synthetics records produced this way leak information about their seeds. This makes privacy harder to achieve than for *seedless* techniques, which all prior work fall under.

Testing privacy: a new strategy. We propose a new methodology to design privacy-preserving algorithms that applies to seedbased data synthesis and may also apply beyond data synthesis. We take a generative model as a black-box and use a *privacy test* as a form of rejection sampling to filter its output. Outputs that are determined to leak too much about their seeds fail the privacy test and are discarded. Others, i.e., outputs that pass the test, are considered part of the produced synthetic dataset. A key feature of this idea is that one may design any privacy test that can be expressed as an algorithm looking at candidate synthetic records. In particular, we show that simple privacy tests can yield strong privacy guarantees such as differential privacy. As a result, the framework provides provable privacy guarantees for several classes of generative models, including state-of-the-art generative models based on deep learning, where no alternative exists to safeguard the privacy of seeds.

A tale of two indistinguishabilities. Though the framework is not inherently tied to any particular notions of privacy or utility, part of this work focuses on framing the utility–privacy trade-off as a trade-off between two kinds of indistinguishabilities. Specifically, privacy is formalized in the sense of differential privacy viewed as a form of indistinguishability of neighboring datasets. Utility is framed as indistinguishability between real and synthetic data: if one cannot distinguish a synthetic record from a real record, then the synthetic data is useful. This framing enables us to characterize the utility-privacy trade-off in two ways: (1) we define relative sensitivity, a property of a generative model with respect to a dataset, and use it to show the impossibility of achieving certain trade-offs; and (2) we propose a methodology to analyze the quality of synthetic data, in the sense of indistinguishability, both theoretically and experimentally.

1.2 MOTIVATING NARRATIVE

You are the chief technology officer of a small Midwest hospital. You would like to share patient electronic medical records with biomedical researchers (external to the hospital) who are working on a study. You could produce summary statistics of the medical records. But you do not know what information the researchers need. In fact, the biomedical researchers confess to you that they do not know precisely what they are looking for and probably will not know until they have a look at the data.

You want to share full data records, with the same format as the originals but you are concerned about patients' privacy and do not know how to properly anonymize the medical records. You decide that a possible solution could be privacy-preserving data synthesis. That is, you want to synthesize, in a privacy-preserving way, a few dozen medical records similar to those of the hospital's patients and send them to the biomedical researchers. You hope that this small sample of synthetic records will be representative of the hospital's patients and sufficient for early-stage data exploration or maybe even formulate a hypothesis.

For the generative model, you decide to train a neural network. And to safeguard privacy, you want to apply the result of a recent paper showing how to train a neural network with differential privacy guarantees. But, your hospital being a small regional hospital, you only have a few hundreds medical records which is not enough to obtain an accurate model. Indeed, state-of-the-art deep-learning models often require hundreds of thousands if not millions of training examples to produce useful outputs [24, 25].

However, you can train an accurate model if the training set includes a large subset of the medical records of all US hospitals. But sampling from the generative model, i.e., seedless synthesis, will yield synthetic records reflecting the distribution of medical records across the US, not that of your hospital. So you decide to sample from the generative model by conditioning its output. Specifically, you use one of the hospital's medical records as a seed for each synthetic record you produce. This is seedbased synthesis. The produced synthetic records are highly similar to those of the hospital's patients. But, there is a problem. What if a synthetic record is so similar to its seed that this compromises patient privacy?

You think of a simple countermeasure: if a synthetic record is too similar to its seed, then discard it; otherwise keep it. You have invented privacy tests. But what does "too similar" mean? What would be a systematic methodology to ensure that the entire procedure provably guarantees privacy for some meaningful notion such as differential privacy? In fact, it is not obvious that there exists such an algorithm. And even if there exists one, what about utility? Could the filtering of the privacy test distort the distribution of synthetics so much that the resulting output is not meaningful?

1.3 CONTRIBUTIONS

This thesis introduces seedbased data synthesis and describes a framework for it.

(1) We formalize seedbased synthesis and propose a mechanism for privacy-preserving seedbased synthesis. This is the first mechanism of its kind.

(2) We introduce privacy tests as a technique to achieve strong privacy guarantees. We believe this technique is of independent interest. We design two privacy tests that combined with our seedbased synthesis mechanism yield (ε, δ)-differential privacy.

(3) We propose an efficient methodology to evaluate the quality of synthetic data both theoretically and experimentally. The methodology is based on cryptography-like indistinguishability game definitions.

(4) We demonstrate the construction of seedbased generative models through examples and propose techniques to use existing models. Specifically, we propose a new generative model for location trajectories and construct a seedbased model inspired by Bayesian networks. We also show how a large class of existing models, including state-of-the-art neural-network models, can be used as seedbased generative models.

(5) We perform extensive experimental validation of the framework using real-world datasets. Specifically, we produce and analyze the quality of synthetic datasets of: location trajectories, census microdata, medical discharge records, and facial images.

1.4 STRUCTURE

This thesis is composed of two parts. The first part (Chapters 2 to 5) develops a framework for sharing sensitive datasets through seedbased data synthesis. The second part (Chapters 6 to 10) validates the framework experimentally using location trajectories, census microdata, electronic medical records, and facial images from real-world datasets.

1. Theory
    - Chapter 2 introduces notation and background concepts, including relevant results on differential privacy.
    - Chapter 3 describes the seedbased synthesis framework, its main mechanism, and how it uses privacy tests. We explain the difference between seedbased synthesis and seedless synthesis. We define the notion of relative sensitivity that captures

the stability of the generative model and the input dataset. Finally, we derive a triangle lemma showing the impossibility of certain utility-privacy trade-offs.

- Chapter 4 describes two privacy tests that yield differential privacy. The first is based on an intuitive criterion we call plausible deniability, whereas the second generalizes the idea and provides a superior privacy guarantee.
- Chapter 5 proposes a methodology to evaluate the utility of synthetic data both theoretically and experimentally. We advocate for the use of a distinguishability-based metric and show that we can, in certain cases, derive utility bounds on the quality of synthetics.

2. Validation

- Chapter 6 describes several generative models that we use for validation. Specifically, we describe: (1) a similarity-based generative model to synthesize location trajectories, (2) a Bayesian networks inspired model to capture statistical relationships between data record attributes, and (3) a class of models, latent-space models, that includes prominent state-of-the-art generative models.
- Chapter 7 evaluates the generative model for location trajectories in two scenarios using real-world location data.
- Chapter 8 applies the framework to generate census microdata using our Bayesian network model.
- Chapter 9 applies the framework to produce synthetic medical records using two latent-space models: principal component analysis and variational autoencoders.
- Chapter 10 applies the framework to synthesize facial images using an off-the-shelf autoencoder generative adversarial network [26].

The thesis is concluded with a brief survey of the relevant literature (Chapter 11) and a discussion of limitations and future work (Chapter 12).

Chapter 2: Background

This chapter introduces notation and background concepts.

2.1 NOTATION

Datasets and data records. A dataset D is a (non-empty) multiset of data records. Each data record is an element of a data universe U which is a large but finite set. The data universe depends on the type of data considered. For example, in a medical context, the data universe U may represent the set of all medical records with a given format (e.g., a specific set of attributes). In a different context, the data universe U may be the set of all bit-strings of length $l \geq 1$, i.e., $U = \{0, 1\}^l$.

The cardinality or size of dataset D, denoted $|D|$ is the number of records in D. Two datasets D_1, D_2 are said to be neighboring, denoted $\|D_1 - D_2\| = 1$, if one dataset can be obtained from the other by adding a single data record $d \in U$.

Models and mechanisms. A *seedbased* generative model $\mathcal{M}$ is a randomized algorithm taking a data record $d \in U$ as input and producing another record $y \in U$ as output. The probability that a generative model $\mathcal{M}$ produces a fixed $y \in U$ as output given input data record $d \in U$ is denoted as: $\Pr\{y \leftarrow \mathcal{M}(d)\}$.

A mechanism $\mathcal{F}$ is a randomized algorithm that takes a dataset as input and produces a single data record $y \in U$ as output. The probability that a generative model $\mathcal{F}$ produces a fixed $y \in U$ as output given input dataset D is denoted as: $\Pr\{y \leftarrow \mathcal{F}(D)\}$. For both generative models and mechanisms, the probability space is over the random choices of the algorithm. We refer to the probability that a generative model or a mechanism produces a given output as the *synthesis probability*.

Probability distributions and expectation. If P denotes a discrete probability distribution over a set X then we let p_x denote the probability of outcome $x \in X$.

The expectation of a real-valued function f of a random variable $\mathbf{X}$ taking values over a set X according to a probability distribution P is denoted by $\mathbb{E}_P[f(\mathbf{X})]$ (or $\mathbb{E}[f(\mathbf{X})]$ when the probability distribution is clear from the context) and is defined as:

$$\mathbb{E}_P[f(\mathbf{X})] = \sum_{x \in X} f(x) \cdot \Pr\{\mathbf{X} = x\} = \sum_{x \in X} f(x) \cdot p_x \ .$$

For an event or statement A, we write $\mathbb{1}_A$ which equals 1 if A occurs and 0 otherwise. It is often convenient to use the fact that: $\Pr\{A\} = \mathbb{E}[\mathbb{1}_A]$.

This thesis references several probability distributions both discrete and continuous.

- The Bernoulli distribution with parameter p is denoted $\mathrm{Bern}(p)$ and has probability mass $p^x(1-p)^{1-x}$ for $x \in \{0,1\}$.
- The Gaussian distribution with mean μ and standard deviation σ is denoted $\mathrm{Gaus}(\mu,\sigma)$.
- The Laplace distribution with shape parameter $b > 0$ has density $\frac{1}{2b}\exp\left(-\frac{|x|}{b}\right)$ and is denoted $\mathrm{Lap}(b)$.
- The symmetric geometric distribution with parameter $0 < \alpha < 1$ has probability mass $\frac{1-\alpha}{1+\alpha}\alpha^{|i|}$, for $i \in \mathbb{Z}$, and is denoted $\mathrm{Geom}(\alpha)$.

Distances and distance metrics. The hamming distance (or weight) of a bit-string x of l bits, denoted $\mathrm{HAMM}(x)$, is the total number of 1 bits it contains.

We consider several distance metrics over discrete probability distributions P and Q with the same support X. The Kullback–Leibler divergence (KL divergence) from Q to P is:

$$\mathrm{KL}(P,Q) = \sum_{x \in X} p_x \ln \frac{p_x}{q_x} \ . \tag{2.1}$$

The statistical distance (SD), also called total variation distance, between P and Q is:

$$\mathrm{SD}(P,Q) = \frac{1}{2}\sum_{x \in X} |p_x - q_x| \ . \tag{2.2}$$

Pinkser's inequality [27] connects the KL divergence to the statistical distance: $\mathrm{SD}(P,Q) \leq \sqrt{\frac{1}{2}\mathrm{KL}(P,Q)}$. The Hellinger distance (HEL) between P and Q is:

$$\mathrm{HEL}(P,Q) = \frac{1}{\sqrt{2}}\sqrt{\sum_{x \in X}(\sqrt{p_x} - \sqrt{q_x})^2} \ . \tag{2.3}$$

Vectors and matrices. When it is unclear from the context, we use bold-type face to denote vectors and matrices. Insofar as convenient and clear, we use lower-case for vectors and upper-case for matrices, so that $\mathbf{x}$ denotes a vector whereas $\mathbf{X}$ denotes a matrix. In particular, we let $\mathbf{I}$ denote the identity matrix where the dimension is either specified or can be inferred from the context. Also, we denote the transpose of a matrix $\mathbf{X}$ as $\mathbf{X}^T$.

2.2 DIFFERENTIAL PRIVACY

Informally, differential privacy [28] is a condition on the output distribution resulting from a computation: specifically that the distribution be multiplicatively bounded by a factor e^{ε} (for some $\varepsilon > 0$ called the *privacy budget*) for any two neighboring input datasets.

Definition 2.1 (Differential Privacy [29])**.**
Mechanism $\mathcal{F}$ satisfies (ε, δ)-differential privacy *if for any neighboring datasets D, D', and any output $S \subseteq \text{Range}(\mathcal{F})$:*

$$\Pr\{\mathcal{F}(D') \in S\} \leq e^{\varepsilon}\Pr\{\mathcal{F}(D) \in S\} + \delta \ .$$

This definition is often called approximate differential privacy because it allows for some small probability $\delta > 0$ of a bad event (i.e., an event such that the multiplicative e^{ε} bound does not hold). In this thesis, we want $\delta \leq 2^{-\lambda}$ where λ is a security parameter.

Differential privacy falls within the category of membership privacy [30] which defines privacy by the extent to which the membership status of an individual record in a dataset influences the output distribution of a mechanism.

There are several prominent mechanisms that provide differential privacy guarantees. We briefly describe the Laplace mechanism [29] and the Exponential mechanism [31].

An important concept is the sensitivity of a function which is the maximum possible change in the function's output for any two neighboring inputs. Specifically, the sensitivity of a function f with outputs over the reals, denoted Δf, is: $\max_{D,D'} ||f(D) - f(D')||_1$, where D and D' are neighboring datasets.

Definition 2.2 (Laplace Mechanism – 3.3 [29])**.**
Let f be any function that takes a dataset D as input and outputs a real number. The Laplace mechanism for $\varepsilon > 0$ outputs $f(D) + z$, where z is drawn from $\text{Lap}(\frac{\Delta f}{\varepsilon})$.

It is shown in [29] that the Laplace mechanism satisfies $(\varepsilon, 0)$-differential privacy.

For computations with outputs ranging over an arbitrary domain R, the Exponential mechanism [31] can be used. Informally, the mechanism works by sampling from its output domain R such that a sample $r \in R$ is returned with probability proportional to $e^{\varepsilon q(D,r)}$, where $q(D, \cdot)$ is a quality function which assigns a score to each element of R. The guarantee (for one output) is $(2\varepsilon\Delta q, 0)$-differential privacy, where Δq is the sensitivity of q.

A crucial property of any meaningful privacy definition is that it provides some way to understand and obtain a guarantee when invoking a mechanism more than once. There are several composition results for differential privacy. For ease of reference, we reproduce here the results for sequential composition and advanced composition. The reader is referred to [29] for a more extensive discussion.

Theorem 2.1 (Sequential Composition – 3.16 [29])**.** *Let $\mathcal{F}_i$ be a $(\varepsilon_i, \delta_i)$-differentially private mechanism for $i = 1, 2, \ldots, m$. Then, for any dataset D, the mechanism which releases outputs $(\mathcal{F}_1(D), \mathcal{F}_2(D), \ldots, \mathcal{F}_m(D))$ is a (ε, δ)-differentially private mechanism for $\varepsilon = \sum_{i=1}^{m} \varepsilon_i$ and $\delta = \sum_{i=1}^{m} \delta_i$.*

Theorem 2.2 (Advanced Composition – 3.20 [29])**.** *Let $\mathcal{F}_i$ be a (ε, δ)-differentially private mechanism for $i = 1, 2, \ldots, m$. Then the mechanism represented by the sequence of k queries over $\mathcal{F}_1(\cdot), \mathcal{F}_2(\cdot), \ldots, \mathcal{F}_m(\cdot)$* with potentially different inputs *is a (ε', δ')-differentially private mechanism for:*

$$\varepsilon' = \varepsilon\sqrt{2k \ln \frac{1}{\delta''}} + k\varepsilon(e^{\varepsilon} - 1) \qquad \textit{and} \qquad \delta' = k\delta + \delta'' \ .$$

A related result, sometimes referred to as parallel composition, is that if we release $(\mathcal{F}_1(D_1), \mathcal{F}_2(D_2), \ldots, \mathcal{F}_m(D_m))$ for *disjoint* datasets $D_1, D_2, \ldots, D_m$, where each $\mathcal{F}_i$ is a $(\varepsilon_i, \delta_i)$-differentially private mechanism, then the guarantee is (ε, δ)-differential privacy, for $\varepsilon = \max_i(\varepsilon_i)$ and $\delta = \max_i(\delta_i)$.

2.3 SYNTHETIC DATA

Synthetic data is nebulous term which often means different things to different people. It is often implicitly or imprecisely defined as any data that was not produced or derived from direct measurement or observation. Throughout this thesis, synthetic data refers to data records with the same format (and belonging to the same data universe) as the real data records they are produced from or inspired by. We call the synthetic data records produced by the techniques proposed in the thesis *full data records* to emphasize that the data format (and dimension) is fully preserved.

The idea of using synthetic data for privacy dates back to 1993 when it was proposed by Rubin [11]. What is attractive about this proposition is that synthetic data (in the sense of full data records) can be used *in lieu* of the real data for many applications. This makes synthetic data ideal for early-stage discovery and data analysis processes that would otherwise need to use real sensitive data.

Chapter 3: Framework

This chapter describes the seedbased synthesis framework. Elements of the framework were originally introduced in [32] and [33]. Specifically, we describe the seedbased mechanism and formalize the notion of privacy tests. We also introduce the concept of relative sensitivity and use it to derive a triangle constraint on achievable utility-privacy trade-offs which applies to any seedbased synthesis process.

3.1 SEEDBASED SYNTHESIS

We are given a probabilistic generative model and a dataset and tasked with producing a synthetic dataset.

3.1.1 Generative Models

A probabilistic generative model is a randomized algorithm that produces a record, which we call *synthetic* as output, where the probability space is over random choices made by the algorithm. In this work, we consider the special case of *seedbased* generative models which (probabilistically) maps the data records universe onto itself, i.e.: $\mathcal{M} : U \to U$. We denote the probability that a generative model $\mathcal{M}$ produces output $y \in U$ given input $x \in U$ (called the *seed*) as: $\Pr\{y \leftarrow \mathcal{M}(x)\}$. This expression, called the *synthesis probability*, is a conditional probability which can be interpreted as follows. Given input $x \in U$: $\mathcal{M}(x)$ describes a (discrete) probability distribution (of the output) over U. That is for all $x \in U$, we have: $\sum_{y \in U} \Pr\{y \leftarrow \mathcal{M}(x)\} = 1$.

The framework requires generative models to satisfy the following properties.

- (Seedbased) the output probability distribution depends on the seed. That is, there exists distinct $x_1, x_2 \in U$ and $y \in U$ such that: $\Pr\{y \leftarrow \mathcal{M}(x_1)\} \neq \Pr\{y \leftarrow \mathcal{M}(x_2)\}$.
- (Computability) for any $x, y \in U$: there exists a procedure to compute $\Pr\{y \leftarrow \mathcal{M}(x)\}$ exactly.

It is important that the procedure to compute $\Pr\{y \leftarrow \mathcal{M}(x)\}$ be efficient enough to be used in practice.[1]

[1] We do not impose any formal efficiency constraints, but remark that in most practical scenarios we desire the time complexity of computing $\Pr\{y \leftarrow \mathcal{M}(x)\}$ to depend on $|x| + |y|$ and not on $|U|$ which can be very large for high-dimensional data.

To provide intuition, we consider two examples of seedbased generative models.

Example 3.1 (Identity model)**.**

The identity generative model is defined (by its probability distribution) as:

$$\Pr\{y \leftarrow \mathcal{M}(x)\} = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{otherwise} \end{cases} .$$

This model is seedbased.

The identity model is not very interesting; its output is always the same as its input. But, for the same reason, it preserves utility.

Example 3.2 (Hamming similarity model)**.** Take as data universe the set of all bit-strings of length $l \geq 1$, i.e., $U = \{0,1\}^l$.

$$\Pr\{y \leftarrow \mathcal{M}(x)\} = \begin{cases} (c_{\mathrm{HAMM}(x)})^{-1} & \text{if } \mathrm{HAMM}(x) = \mathrm{HAMM}(y) \\ 0 & \text{otherwise} \end{cases} ,$$

where $c_{\mathrm{HAMM}(x)} = |\{y \in U : \mathrm{HAMM}(x) = \mathrm{HAMM}(y)\}|$.

In other words, the model outputs a uniformly random string with the same hamming weight as its input. Thus, by construction, this model preserves the hamming weight distribution of the input dataset.

Seedless Synthesis. A generative model is called *seedless* if its output is independent of its input. That is if for all y: the value of $\Pr\{y \leftarrow \mathcal{M}(x)\}$ is the same regardless of $x \in U$.

Example 3.3 (Uniform model)**.** The uniform model $\mathcal{M}_{\mathrm{U}}$ is such that for any $x, y \in U$:

$$\Pr\{y \leftarrow \mathcal{M}_{\mathrm{U}}(x)\} = \frac{1}{|U|} .$$

This model is seedless.

Note that seedless synthesis is a special case of seedbased synthesis.

Generative models as distributions. Generative models can equivalently be viewed as randomized algorithms or distributions. This also illustrates the difference between seedless and seedbased generative models. Let P denote the distribution of a generative model. Seedless synthesis is equivalent to drawing y as

$$y \sim P(y) \ .$$

In contrast, seedbased synthesis is equivalent to drawing y as

$$y \sim P(y \mid d) \ ,$$

where $d \in D$ is the seed.

Training. Real-world generative models typically have parameters to drive their behavior. The parameters are usually learned from data through a training process. It is important to note that if training is performed using sensitive data, the learned parameters may leak sensitive information that is not captured by what the model's output reveals about its input seed. This issue can sometimes be bypassed by learning model parameters from public data. Alternatively, we can use a training process that satisfies differential privacy.

3.1.2 Basic Synthesis

To produce synthetic records from an entire input dataset (as opposed to from a single seed record) we use the following process.

Definition 3.1 (Basic synthesis)**.**
Input: dataset D, generative model $\mathcal{M}$.
Output: synthetic $y \in U$.

1. *Select a* seed *record $d \in D$ uniformly at random.*
2. *Sample a* synthetic *record $y \leftarrow \mathcal{M}(d)$.*
3. *Output y.*

We denote this procedure as $\mathcal{G}$ and call it basic synthesis*. It produces a single synthetic record $y \in U$ for every invocation with probability:*

$$\Pr\{y \leftarrow \mathcal{G}(D)\} = |D|^{-1} \sum_{d \in D} \Pr\{y \leftarrow \mathcal{M}(d)\} \ .$$

In other words, basic synthesis simply picks a random input record as seed and feeds it into the generative model to obtain an output.

3.2 MECHANISM

We construct a *seedbased synthesis mechanism* which uses a privacy test in addition to basic synthesis. The privacy test rejects synthetic candidates that do not satisfy some privacy criterion in order to achieve a privacy guarantee such as differential privacy.

Given a generative model $\mathcal{M}$, and a dataset D, the mechanism runs a privacy test for any released data. The mechanism produces data records by using $\mathcal{M}$ on dataset D.

Mechanism 3.1 (The mechanism)**.**
Input: generative model $\mathcal{M}$, dataset D.
Output: synthetic record y or nothing ($\perp$).

1. *Select a* seed *record $d \in D$ uniformly at random.*
2. *Generate a* candidate *synthetic record $y \leftarrow \mathcal{M}(d)$.*
3. *Invoke the* **privacy test** *on $(\mathcal{M}, D, d, y)$.*
4. *If the tuple passes the test, then output y.*
   Otherwise, output $\perp$.

Note that basic synthesis can be viewed as an instance of Mechanism 3.1 with a privacy test which always returns *pass*.

Privacy Tests. The privacy test is a form of rejection sampling; by rejecting some synthetics and not others, the test can shape the output distribution to ensure that the mechanism satisfies some privacy guarantee such as differential privacy. We formalize the class of privacy tests as tuples (κ, δ), where κ is a privacy score function (the higher the score the higher the privacy) and δ is function mapping scores to probabilities.

The privacy score function κ takes as input the generative model $\mathcal{M}$, the dataset D, the seed record d, and the synthetic y and produces a non-negative real number interpreted as a score. The larger the score, the more desirable it is that y passes the test. The probability of passing the privacy test is: $\delta(x)$, where $x = \kappa(\mathcal{M}, D, d, y)$. This thesis considers functions δ that are well-behaved in the following sense.

Definition 3.2. *A privacy test function $\delta : [0, \infty) \to [0, 1]$ is* well-behaved *if the following conditions hold.*

(i) Non-decreasing*: for any $x > y$: $\delta(x) \geq \delta(y)$.*

(ii) Bounded increments*: there exist constants $c_0 > 0$, and $\beta_0 \geq 1$ such that for all $x \geq 0$:*

$$\frac{\delta(x + c_0)}{\delta(x)} \leq \beta_0 \quad \textit{and} \quad \frac{1 - \delta(x)}{1 - \delta(x + c_0)} \leq \beta_0 \ .$$

Informally, the first condition says that the higher the score the higher the probability of passing the test, whereas the second condition says that the increase in probability (of passing and of failing the test) with the score itself is bounded. As we will see in Chapter 4, the second condition is necessary to ensure that the test itself does not leak sensitive information.

Algorithmically, we describe a generic test as follows.

Privacy Test 3.1 (Generic test $\mathcal{T}_{\kappa,\delta}$).
Input: generative model $\mathcal{M}$, dataset D, data records d and y, privacy score function κ, and privacy test function δ
Output: pass *to allow releasing y, otherwise output* fail.

```
1. Compute the score $x = \kappa(\mathcal{M}, D, d, y)$.
2. With probability $\delta(x)$ return pass
   Otherwise return fail.
```

Example 3.4. The simplest form of privacy test is a threshold test on the score: if the score exceeds some pre-defined threshold k, then the test returns *pass*, otherwise it returns *fail*. This test implicitly defines δ such that for $x < k$: $\delta(x) = 0$, and $\delta(x) = 1$ otherwise.

Deterministic tests do not provide differential privacy (and are not well-behaved). But we can turn a deterministic threshold test into a well-behaved (randomized) test by adding noise to the threshold k. We call this the *noisy threshold technique*.

Privacy Test 3.2 (Noisy threshold test $\mathcal{T}_{\kappa,Z,k}$).
Input: generative model $\mathcal{M}$, dataset D, records d and y, privacy score function κ, privacy parameter k, and noise distribution Z
Output: pass *to allow releasing y, otherwise output* fail.

```
1. Randomize $k$ by adding fresh noise: $\tilde{k} = k + z$.
2. Compute the score $x = \kappa(\mathcal{M}, D, d, y)$.
3. If $x \geq \tilde{k}$ then return pass
   Otherwise return fail.
```

Where z is a sample drawn from the noise distribution Z, i.e., $z \sim Z$.

Note that when the noise is symmetric, it may be useful to think of it as being added to the score rather than to the threshold. Appendix A.1 describes properties of the symmetric geometric distribution and the Laplace distribution as they relate to Definition 3.2.

The probability of synthesizing any fixed $y \in U$ using Mechanism 3.1 (denoted by $\mathcal{F}$) with privacy test (κ, δ) given input dataset D is:

$$\Pr\{y \leftarrow \mathcal{F}(D)\} = |D|^{-1} \sum_{d \in D} \Pr\{y \leftarrow \mathcal{M}(d)\} \cdot \delta(\mathcal{M}, D, d, y) \ ,$$

where for conciseness we overload δ as $\delta(\mathcal{M}, D, d, y) = \delta(x)$ for $x = \kappa(\mathcal{M}, D, d, y)$.

In Chapter 4 we describe two instances of privacy tests that lead to differential privacy.

3.3 CHARACTERIZING THE TRADE-OFF

Suppose we seek to design a mechanism $\mathcal{F}$ that satisfies ε-differential privacy. Further, suppose we would like $\mathcal{F}$ to produce synthetics with similar probability as basic synthesis. Given $y \in U$ this demand can be formalized by requiring that for some small $\alpha \geq 1$: $\alpha^{-1}\Pr\{y \leftarrow \mathcal{G}(D)\} \leq \Pr\{y \leftarrow \mathcal{F}(D)\} \leq \alpha \Pr\{y \leftarrow \mathcal{G}(D)\}$. Given that utility and privacy may be in conflict, a natural question is: what characterizes the trade-off between α and ε? In this section, we explore this question through the notion of *relative sensitivity*.

3.3.1 Relative Sensitivity

A key idea in the theory of differential privacy is to calibrate noise to the sensitivity of the function one wishes to privatize [28]. For example, using the Laplace mechanism for a real-valued function f, the noise needed to achieve ε-differential privacy is $\mathrm{Lap}(\Delta f/\varepsilon)$, where Δf is the global sensitivity of f and $\Delta f = \sup_{D,D':\|D-D'\|=1} \|f(D) - f(D')\|_1$. In this sense, we can think of the sensitivity of f as defining a trade-off between the privacy level (ε) and the utility, i.e., inaccuracy of the response or noise magnitude $(\Delta f/\varepsilon)$.

To understand the sensitivity of basic synthesis $\mathcal{G}$, we look towards the synthesis probabilities $\Pr\{y \leftarrow \mathcal{G}(D)\}$ and $\Pr\{y \leftarrow \mathcal{G}(D')\}$ for neighboring D, D' and some fixed arbitrary $y \in U$ and observe that we can find $\mathcal{M}$ and neighboring D, D' such that:

$$|\Pr\{y \leftarrow \mathcal{G}(D)\} - \Pr\{y \leftarrow \mathcal{G}(D')\}| = \frac{1}{|D|} \ .$$

Example 3.5. Let $D = D' \cup \{d^\star\}$ for some $d^\star \in U$ and take $\mathcal{M}$ such that $\Pr\{y \leftarrow \mathcal{M}(d)\} = 0$ for $d \in D'$ and $\Pr\{y \leftarrow \mathcal{M}(d^\star)\} = 1$.

This is a significant challenge towards achieving privacy because for high-dimensional data universes $(|U| \gg |D|)$ we expect the synthesis probabilities to be orders of magnitude smaller than $\frac{1}{|D|}$. Under what circumstances could we expect the (local) sensitivity to be roughly the same order of magnitude as the synthesis probabilities? What would have to hold of $\mathcal{G}$ (i.e., $\mathcal{M}$) and D? A bound on the following, which we call the relative sensitivity of $\mathcal{G}$ at D.

Definition 3.3. *The* relative sensitivity *of $\mathcal{G}$ at D, y is:*

$$\tilde{\Delta}\mathcal{G}(D, y) = \max_{d^\star \in D} \left\{ \frac{\Pr\{y \leftarrow \mathcal{G}(D)\}}{\Pr\{y \leftarrow \mathcal{G}(D')\}}, \frac{\Pr\{y \leftarrow \mathcal{G}(D')\}}{\Pr\{y \leftarrow \mathcal{G}(D)\}} \right\} , \tag{3.1}$$

where $D' = D \setminus \{d^\star\}$ and we interpret the ratio to be unbounded ($\tilde{\Delta}\mathcal{G}(D, y) = \infty$) if there exists $d^\star$ such that $\Pr\{y \leftarrow \mathcal{G}(D \setminus \{d^\star\})\} = 0$.

Note that the relative sensitivity is actually a condition on the pair $(\mathcal{M}, D)$, since:

$$\frac{\Pr\{y \leftarrow \mathcal{G}(D)\}}{\Pr\{y \leftarrow \mathcal{G}(D')\}} = \frac{|D| - 1}{|D|} \left[1 + \frac{\Pr\{y \leftarrow \mathcal{M}(d^\star)\}}{\sum_{d \in D'} \Pr\{y \leftarrow \mathcal{M}(d)\}} \right] .$$

For example, observe that the identity model has unbounded relative sensitivity for arbitrary input datasets. As another example, consider as data universe the set of all bit-strings of length $l \geq 1$, i.e., $U = \{0, 1\}^l$. The relative sensitivity of the model in Example 3.2 is unbounded: if a dataset D' does not contain the bit-string 0^l, then the probability of producing $y = 0^l$ under $D = D' \cup \{0^l\}$ is non-zero whereas it is zero under D'. That said, remark that for some datasets the relative sensitivity may be bounded. For example, take any D that contains two or more record with every hamming weight between 0 and l.

In contrast, any seedless model when applied to bit-strings of length l, has relative sensitivity 1 (which is minimal). For example, the model of Example 3.2 and that of Example 3.3 can be combined to produce a hybrid model as follows.

Example 3.6. Let $\mathcal{M}_{\text{HAMM}}$ denote the model of Example 3.2 and $\mathcal{M}_{\text{U}}$ denote the model of Example 3.3. Let $0 < b < 1$ be a parameter. The hybrid model $\mathcal{M}_b$ is:

$$\Pr\{y \leftarrow \mathcal{M}_b(x)\} = b \cdot \Pr\{y \leftarrow \mathcal{M}_{\text{HAMM}}(x)\} + (1 - b) \cdot \Pr\{y \leftarrow \mathcal{M}_{\text{U}}(x)\} .$$

It can be seen that the value of the parameter b directly controls the relative sensitivity of the model of Example 3.6. The closer b is to 1 the smaller the relative sensitivity and (simultaneously) the less the model's output depends on the input.

3.3.2 The Triangle Lemma

We introduce a key result towards characterizing the utility-privacy trade-off, which we call the *Triangle Lemma* by analogy to the triangle inequality. Informally, the lemma says that given a dataset D and a generative model $\mathcal{M}$, the utility-privacy trade-off is constrained by the relative sensitivity of $\mathcal{G}$ at D. A consequence is the impossibility of achieving certain utility-privacy trade-offs (no matter the mechanism $\mathcal{F}$).

Lemma 3.1 (Triangle Lemma)**.** *Let $\mathcal{G}$ denote basic synthesis with some generative model $\mathcal{M}$, let $\mathcal{F}$ be any mechanism with outputs over U. Also, let D be a dataset and take $D' = D\backslash\{d^\star\}$ for some $d^\star \in D$. Finally, let $y \in U$, If:*

- $\alpha = \max_{D^\star\in\{D,D'\}} \max\left\{\frac{\Pr\{y\leftarrow\mathcal{G}(D^\star)\}}{\Pr\{y\leftarrow\mathcal{F}(D^\star)\}}, \frac{\Pr\{y\leftarrow\mathcal{F}(D^\star)\}}{\Pr\{y\leftarrow\mathcal{G}(D^\star)\}}\right\}$,
- $\varepsilon = \max_{D_1,D_2:\|D_1-D_2\|=1} \max\left\{\ln\frac{\Pr\{y\leftarrow\mathcal{F}(D_1)\}}{\Pr\{y\leftarrow\mathcal{F}(D_2)\}}, \ln\frac{\Pr\{y\leftarrow\mathcal{F}(D_2)\}}{\Pr\{y\leftarrow\mathcal{F}(D_1)\}}\right\}$,
- $\eta = \frac{\Pr\{y\leftarrow\mathcal{G}(D)\}}{\Pr\{y\leftarrow\mathcal{G}(D')\}} \geq 1$,

with $\alpha, \epsilon, \eta < \infty$.

Then:

$$\eta \leq \alpha^2 e^{\varepsilon} \ . \tag{3.2}$$

The first condition can be viewed as a utility demand as it relates to the closeness condition on synthesis probabilities of mechanism $\mathcal{F}$ and those of basic synthesis. As such it, α is related to utility in a sense which we make precise in Chapter 5. For now, we ask the reader to conceive of α as related to utility in the following sense: the closer α is to 1, the higher the utility. Remark that the second condition says that $\mathcal{F}$ satisfies $(\varepsilon, 0)$-differential privacy (if it holds for all $y \in U$).

Proof. Take $\mathcal{G}$, $\mathcal{F}$, y, D, D', α, ε, and η as in the lemma. Then:

$$\eta = \frac{\Pr\{y \leftarrow \mathcal{G}(D)\}}{\Pr\{y \leftarrow \mathcal{G}(D')\}} \leq \alpha\frac{\Pr\{y \leftarrow \mathcal{F}(D)\}}{\Pr\{y \leftarrow \mathcal{G}(D')\}} \leq \alpha e^{\varepsilon}\frac{\Pr\{y \leftarrow \mathcal{F}(D')\}}{\Pr\{y \leftarrow \mathcal{G}(D')\}} \leq \alpha^2 e^{\varepsilon} \ .$$

□

We call Lemma 3.1 the *Triangle Lemma* by analogy to the triangle inequality. The indistinguishability demand of ε-differential privacy coupled with the α-closeness condition of the synthesis probability of $\mathcal{F}$ and $\mathcal{G}$ constraint the relative sensitivity of $\mathcal{G}$ at D. Equivalently, if the relative sensitivity of $\mathcal{G}$ at D is known, then certain pairs (α, ε) characterizing points on the utility-privacy trade-off curve cannot be reached by any mechanism $\mathcal{F}$. Fig. 3.1 shows the feasible regions for different values of η in terms of α and ε.

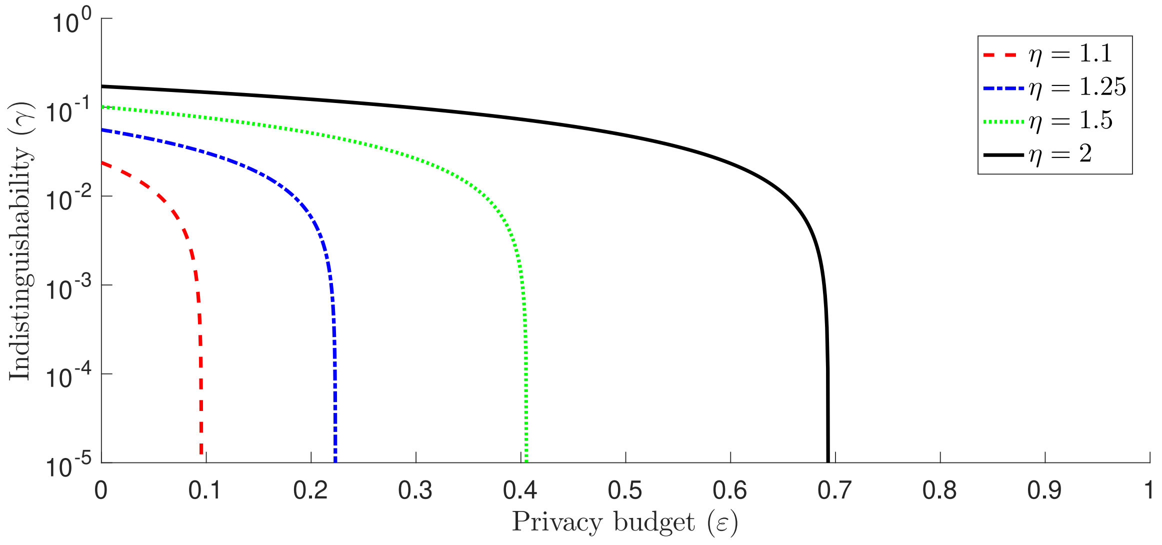


Figure 3.1: Visual representation of the Triangle Lemma (Lemma 3.1) for different values of η in terms of $\gamma = \frac{\alpha-1}{\alpha+1}$ (which we call indistinguishability) and ε. The area to the top-right of each curve is the feasible region according to Eq. (3.2). No mechanism can offer a utility-privacy trade-off that lies to the bottom left of the corresponding curve.

Example 3.7. For $\eta = 3$ no mechanism that satisfies differential privacy for $\varepsilon = \ln 2$ can have $\alpha < \sqrt{3/2} \approx 1.22$.

A further (and related) consequence of Lemma 3.1 is that if we desire to have α arbitrarily close to 1 (i.e., maximize utility), then we cannot achieve better privacy than $\varepsilon \approx \ln \eta$. Informally, if the relative sensitivity is large there is no hope of achieving good privacy without a significant decrease in utility!

Chapter 4: Testing Privacy

This chapter presents two generic privacy tests which when used with the seedbased synthesis mechanism (Chapter 3) yield (ε, δ)-differential privacy. The first is based on an intuitive criterion called *plausible deniability* (Section 4.1). The second test (Section 4.2) generalizes the idea using insights from the notion of relative sensitivity and provides better privacy.

Before describing the two privacy tests, we provide an illustrating example.

Example 4.1 (Understanding Privacy Testing)**.**

Imagine running a survey asking individuals whether they have engaged in some illicit behavior; they must respond by yes (1) or no (0). This can be modeled in the data universe $U = \{0, 1\}$ where each record is one individual's response. Assuming $n \geq 1$ respondents, survey responses form a dataset D of n records.

One way to get an estimate of the number of respondents that engaged in the illicit behavior is to repeatedly pick a record $d \in D$ uniformly at random and examine it. Let $n_1(D)$ denotes the proportion of 1 records in D. The probability of picking a 1 record is $\frac{n_1(D)}{n}$. This process is exactly modeled by the identity generative model $\mathcal{M}$: $\mathcal{M}(d) = d$ for $d \in \{0, 1\}$. The probability of output $x \in \{0, 1\}$ from basic synthesis ($\mathcal{G}$) is:

$$\Pr\{x \leftarrow \mathcal{G}(D)\} = \begin{cases} p(D) & \text{for } x = 1 \\ 1 - p(D) & \text{for } x = 0 \end{cases},$$

where $p(D) = \frac{n_1(D)}{n}$.

Observe that $\mathcal{G}$ does not satisfy ε-differential privacy (for any $\varepsilon > 0$). For example, take $D_1 = \{0, 0, \ldots, 0\}$ and $D_2 = D_1 \cup \{1\}$. Then $\Pr\{1 \leftarrow \mathcal{G}(D_2)\} = |D_2|^{-1}$ but $\Pr\{1 \leftarrow \mathcal{G}(D_1)\} = 0$. In other words, from the output of $\mathcal{G}$ one may be able to determine whether a target individual has engaged in the illicit behavior (if it is known that no other individual in the dataset has).

Consider the following generative model $\mathcal{M}$ as an alternative:

$$\mathcal{M}(d) = \begin{cases} d & \text{with probability } \frac{2}{3} \\ 0 & \text{with probability } \frac{1}{6} \\ 1 & \text{with probability } \frac{1}{6} \end{cases}, \tag{4.1}$$

which is an instance of randomized response [34]. Specifically, with probability 2/3 the output is identical to the input, otherwise it is equally likely to be 0 or 1.

The probability of outputting $x \in \{0, 1\}$ from basic synthesis with this alternative generative model is:

$$\Pr\{x \leftarrow \mathcal{G}(D)\} = \begin{cases} \frac{2}{3}\left[p(D) + \frac{1}{4}\right] & \text{for } x = 1 \\ \frac{2}{3}\left[\frac{5}{4} - p(D)\right] & \text{for } x = 0 \end{cases},$$

with $p(D) = \frac{n_1(D)}{n}$. Remark that the quantity $p(D)$ can be estimated by multiplying by $3/2$ the proportion of 1s observed and removing $1/4$.

It can be seen that $\mathcal{G}$ satisfies $(\ln 3, 0)$-differential privacy. Given a generative model we want to use, it is often possible to modify it into (e.g., by adding noise) one that leads to differential privacy. This chapter shows a different way to obtain privacy guarantees. Instead of modifying the generative model, we use a privacy test that will probabilistic reject some outputs. Observe that we can trivially achieve differential privacy using rejection sampling: simply reject all outputs. If no output is released, there is no privacy leakage. The challenge is, of course, to design tests that achieve privacy with minimal impact on the utility which often involves minimizing the rejection rate.

Take the identity generative model and consider the following privacy test parameterized by an integer $k \geq 2$:

$$\delta(D, \cdot, x) = \left(\frac{2}{3}\right)^{\max\{k - c_x,\ 1\}},$$

where $c_x = |\{d \in D : d = x\}|$ is number of records in D with value x. The mechanism with this privacy test satisfies $(\ln 3, (3/2)^{-(k-1)})$-differential privacy for any dataset with at least k records. Up to the δ, this is equivalent to the guarantee offered by the randomized response generative model. However, for any input dataset D that contains at least k 1s and k 0s (i.e., $\min\{n_1(D), n - n_1(D)\} \geq k$) the observed proportion of 1s is exactly $p(D)$ (when the test fails, there is no output) which results in less distortion than that of the randomized response generative model. Basic synthesis with the identity generative model does not leak much when the input data contains at least a few 0s and 1s. In contrast, techniques which modify the generative model often introduce noise or distortion to safeguard the worst-case (in this case, few 0s and 1s) but end up affecting all the cases.

4.1 PLAUSIBLE DENIABILITY

In this section, we show how to design a privacy test based on an intuitive criterion called *plausible deniability*. This criterion was first proposed in [33] and formalized in [32]. We show that a privacy test based on plausible deniability as a privacy score function yields differential privacy provided an appropriate choice of the privacy test function δ. For example, one can make use of the noisy threshold technique.

The intuition behind the plausible deniability criterion is that given a synthetic record, the larger the set of input records which could have plausible generated the synthetic, the lower the leakage about the seed and so the higher the privacy (and the score). We formalize this idea as follows. As before $\mathcal{M}$ denotes an arbitrary seedbased probabilistic generative model that given any data record d generates a synthetic record y with probability $\Pr\{y \leftarrow \mathcal{M}(d)\}$. Let $k \geq 1$ be an integer and $r \geq 1$ be a real number.

Definition 4.1 (Plausible Deniability Criterion).
For any dataset D with $|D| \geq k$, and any record y generated by a probabilistic generative model $\mathcal{M}$ such that $y \leftarrow \mathcal{M}(d_1)$ for $d_1 \in D$, we state that y is releasable with (k, r)-plausible deniability*, if there exist at least $k-1$ distinct records $d_2, ..., d_k \in D \setminus \{d_1\}$ such that*

$$r^{-1} \leq \frac{\Pr\{y \leftarrow \mathcal{M}(d_i)\}}{\Pr\{y \leftarrow \mathcal{M}(d_j)\}} \leq r, \tag{4.2}$$

for any $i, j \in \{1, 2, \ldots, k\}$.

The larger privacy parameter k is, the larger the indistinguishability set for the input data record. Also, the closer to 1 privacy parameter r is, the stronger the indistinguishability of the input record among other plausible records.

Based on Definition 4.1, we propose the following privacy score function:

$$\kappa(\mathcal{M}, D, s, y) = \left|\{d \in D : \lfloor -\log_r \Pr\{y \leftarrow \mathcal{M}(s)\}\rfloor = \lfloor -\log_r \Pr\{y \leftarrow \mathcal{M}(d)\}\rfloor\}\right| , \tag{4.3}$$

where we assume that $\Pr\{y \leftarrow \mathcal{M}(s)\} > 0$ since otherwise it could not be the seed. (Alternatively, define $\kappa(\mathcal{M}, D, s, y) = 0$ whenever $\Pr\{y \leftarrow \mathcal{M}(s)\} = 0$.)

In other words, the privacy score is the number of *plausible seeds* which we define as the number of input data records $d \in D$ such that:

$$r^{-i-1} < \Pr\{y \leftarrow \mathcal{M}(d)\} \leq r^{-i} ,$$

where i is the unique positive integer that satisfies:

$$r^{-i-1} < \Pr\{y \leftarrow \mathcal{M}(s)\} \leq r^{-i} \ .$$

The following is a deterministic test using Eq. (4.3) as privacy score.

Privacy Test 4.1 (Deterministic test $\mathcal{T}_k$).
Input: generative model $\mathcal{M}$, dataset D, data records s, y, and privacy parameters k and r.
Output: pass *to allow releasing y,* fail *otherwise.*

1. *Let $i \geq 0$ be the (only) integer such that: $r^{-i-1} < \Pr\{y \leftarrow \mathcal{M}(s)\} \leq r^{-i}$.*
2. *Let k' be the number of records $d \in D$ such that: $r^{-i-1} < \Pr\{y \leftarrow \mathcal{M}(d)\} \leq r^{-i}$.*
3. *If $k' \geq k$ then return* pass, *otherwise return* fail.

Remark that this privacy score function enforces a stringent condition that the probability of generating a candidate synthetic y given the seed s and the probability of generating the same record given another plausible seed d both fall into a geometric range $[r^{-i-1}, r^{-i}]$, for some integer $i \geq 0$, assuming $r > 1$. Notice that, under this test, the set of $k-1$ different ds plus s satisfies the plausible deniability condition Eq. (4.2).

Informally, the threshold k prevents releasing the *implausible* synthetics records y. As k increases the number of plausible records which could have produced y also increases. Thus, an adversary with only partial knowledge of the input dataset cannot readily determine whether a particular input record d was the seed of any released record y. This is because there are at least $k-1$ other records $d_i \neq d$ in the input dataset which could *plausibly* have been the seed.

4.1.1 Differential Privacy

A problem with the deterministic test is that passing it with some y inherently reveals something about the number of plausible seeds, which could potentially reveal whether a particular d is included in the input data. This problem can be solved if we appropriately randomize the threshold k. Indeed, we show that with the randomized test based on the noisy threshold technique for an appropriate noise distribution, the mechanism (Mechanism 3.1) denoted by $\mathcal{F}$ satisfies (ε, δ)-differential privacy.

Specifically, we consider the following privacy test which is an instance of the noisy threshold technique with Eq. (4.3) as privacy score.

Privacy Test 4.2 (Randomized test $\mathcal{T}_{\epsilon_0,k,r}$)**.**
Input: generative model $\mathcal{M}$*, dataset* D*, data records* s*,* y*, and parameters* k*,* r*,* ϵ_0*.*
Output: pass *to allow releasing* y*,* fail *otherwise.*

1. *Randomize* k *by adding fresh noise:* $\tilde{k} = k + \mathrm{Lap}(\frac{1}{\epsilon_0})$*.*
2. *Let* $i \geq 0$ *be the (only) integer such that:* $r^{-i-1} < \Pr\{y \leftarrow \mathcal{M}(s)\} \leq r^{-i}$*.*
3. *Let* k' *be the number of records* $d \in D$ *such that:* $r^{-i-1} < \Pr\{y \leftarrow \mathcal{M}(d)\} \leq r^{-i}$*.*
4. *If* $k' \geq \tilde{k}$ *then return* pass*, otherwise return* fail*.*

Theorem 4.1. *Let* $\mathcal{F}$ *denote Mechanism 3.1 with Privacy Test 4.2 and parameters* $k \geq 1$*,* $r > 1$*, and* $\varepsilon_0 > 0$*. For any neighboring datasets* D *and* D' *such that* $|D|, |D'| \geq k$*, any set of outcomes* $Y \subseteq U$*, and any integer* $1 \leq t < k$*, we have:*

$$\Pr\{\mathcal{F}(D') \in Y\} \leq e^{\varepsilon} \Pr\{\mathcal{F}(D) \in Y\} + \delta \ ,$$

for $\delta = e^{-\varepsilon_0(k-t)}$ *and* $\varepsilon = \varepsilon_0 + \ln(1 + \frac{r}{t})$*.*

The proof of Theorem 4.1 can be found in Appendix A.3. Roughly speaking, the theorem says that, except with some small probability δ, adding a record to a dataset cannot change the probability that any synthetic record y is produced by more than a small multiplicative factor. The intuition behind this is the following.

Fix an arbitrary synthetic record y produced by the mechanism on some dataset. Remark that given y, records are partitioned into disjoint sets according to their probabilities of generating y (with respect to $\mathcal{M}$). That is, partition i for $i = 0, 1, 2 \ldots$, contains those records d such that $r^{-(i+1)} < \Pr\{y \leftarrow \mathcal{M}(d)\} \leq r^{-i}$.

Now suppose we add an arbitrary record d' to the dataset. The probability of producing y changes in two ways: (1) the probability that y is generated increases because d' may be chosen as seed, and (2) the probability that y passes the privacy test increases because d' is an additional plausible seed. Remark that this change only impacts whichever partition d' falls into because the probability of passing the privacy test depends only on the number of plausible seeds in the partition of the seed.

Thus, we focus on the partition in which d' falls. If that partition contains a small number of records compared to k then introducing d' could increase the probability of generating y significantly, but the probability of passing the privacy test is very small. (The probability of passing the privacy test decreases exponentially the fewer plausible seeds are available compared to k.) In contrast, if the partition contains a number of records comparable to k or larger, then the probability of generating y increases only slightly (because there are already a large number of plausible seeds with similar probability of generating y as d'). And, the probability of passing the privacy test increases by a multiplicative factor of at most e^{ε_0} due to adding Laplacian noise. In both cases, the increase to the probability of producing y due to adding d' is small and bounded.

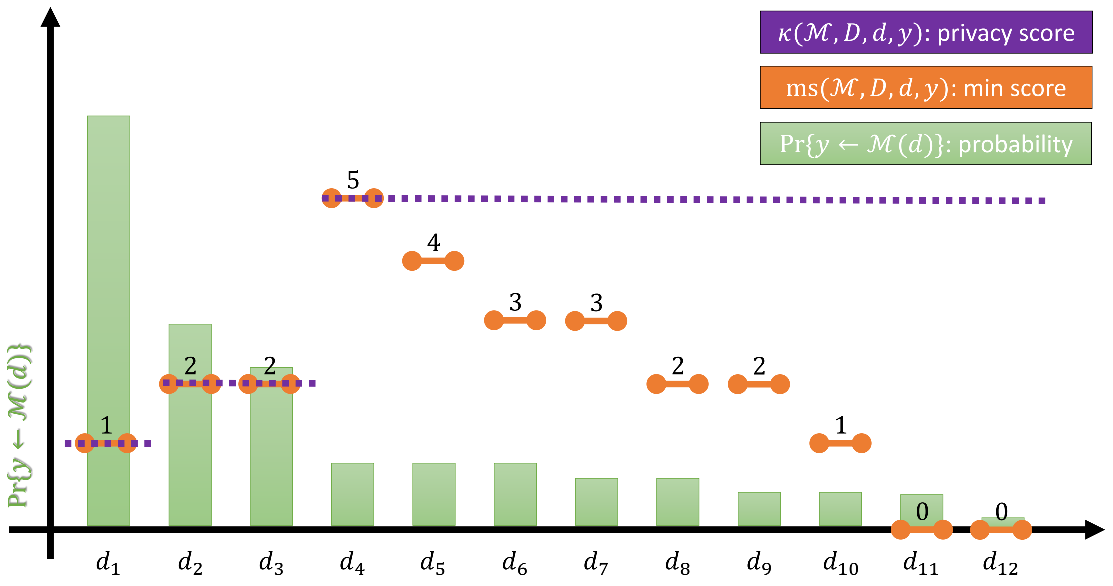


Figure 4.1: Improving the test: illustration of the relationship between the synthesis probabilities, the min score, and the privacy score κ. (For simplicity of illustration, we take the floor of the min score to make it integer valued.)

4.2 IMPROVING THE TEST

The plausible deniability criterion has two downsides: (1) it is overly conservative so that tests based on it sometimes fail even in situations where privacy is readily achieved, and (2) its rigid partitioning of records in plausible seeds makes analyzing utility difficult. In this section, we use insights from the concept of relative sensitivity (Section 3.3.1) to construct a privacy test which achieves tighter bounds than Theorem 4.1 and which facilitates analysis of the utility of the produced synthetics (Chapter 5).

We propose to use a privacy score function κ based on relative sensitivity (Definition 3.3). Given a synthetic candidate y, the higher the relative sensitivity $\tilde{\Delta}\mathcal{G}(D, y)$, the more y leaks about D and therefore the smaller the privacy score (and thus the probability of passing the test) should be.

Define the *min score* of a dataset D with respect to seed $d \in D$ and $y \in U$ as follows:

$$\mathrm{ms}(\mathcal{M}, D, d, y) = \frac{\sum_{s \in S_d \setminus \{d\}} \Pr\{y \leftarrow \mathcal{M}(s)\}}{\Pr\{y \leftarrow \mathcal{M}(d)\}} \ , \tag{4.4}$$

where $S_d = \{s \in D : \Pr\{y \leftarrow \mathcal{M}(d)\} \geq \Pr\{y \leftarrow \mathcal{M}(s)\}\}$. For convenience, we define $\mathrm{ms}(\mathcal{M}, D, d, y) = 0$ whenever $\Pr\{y \leftarrow \mathcal{M}(d)\} = 0$, which is consistent with the fact that d being the seed guarantees that $\Pr\{y \leftarrow \mathcal{M}(d)\} > 0$. Remark that:

$$0 \leq \mathrm{ms}(\mathcal{M}, D, d, y) \leq |S_d| - 1 \ .$$

We define the privacy score in terms of the min score. We do so recursively based on the rank ordering of the records of D with respect to their probability of producing y when selected as seed.

Definition 4.2 (Privacy score)**.** *Given dataset D of n records, generative model $\mathcal{M}$, seed d, and synthetic $y \in U$. Label the records in D according to their rank such that for $D = \{d_1, d_2, \ldots, d_n\}$:*

$$\Pr\{y \leftarrow \mathcal{M}(d_1)\} \geq \Pr\{y \leftarrow \mathcal{M}(d_2)\} \geq \ldots \geq \Pr\{y \leftarrow \mathcal{M}(d_n)\} \ .$$

Then:

$$\kappa(\mathcal{M}, D, d, y) = \begin{cases} \mathrm{ms}(\mathcal{M}, D, d, y) & \textit{if } i = 1 \\ \max\{\kappa(\mathcal{M}, D, d_{i-1}, y), \mathrm{ms}(\mathcal{M}, D, d, y)\} & \textit{for } i > 1 \ , \end{cases}$$

where i is such that $d = d_i$.

Observation 4.1. *Let $d, d' \in D$. If $\Pr\{y \leftarrow \mathcal{M}(d)\} \geq \Pr\{y \leftarrow \mathcal{M}(d')\}$ then:*

$$\kappa(\mathcal{M}, D, d, y) \leq \kappa(\mathcal{M}, D, d', y) \ .$$

In the remainder of this section, we consider a single generative model $\mathcal{M}$ and omit it whenever possible for conciseness. We write $\mathrm{ms}(D, d, y)$ and $\kappa(D, d, y)$ to denote the min score and privacy score respectively.

The privacy score quantifies the influence of a single record on the overall synthesis probabilities in the following sense.

Observation 4.2. *Let $d \in D$, $y \in U$. If $\kappa(D, d, y) \geq t$ for some $t > 0$, then:*

$$\frac{\Pr\{y \leftarrow \mathcal{M}(d)\}}{\sum_{s \in D} \Pr\{y \leftarrow \mathcal{M}(s)\}} \leq \frac{1}{t} \ .$$

Definition 4.2 has sensitivity 1, which makes it attractive to use for a privacy test. Fig. 4.1 illustrates the relationship between the synthesis probabilities, the min score (Eq. (4.4)), and the privacy score Definition 4.2.

Theorem 4.2. *For any two neighboring datasets D_1, D_2 of at least $n \geq 1$ records, any $y \in U$, if $\mathcal{F}$ denotes Mechanism 3.1 with Definition 4.2 as privacy score function, and any well-behaved privacy test function δ (Definition 3.2) for $c_0 = 1$, $\beta_0 \geq 1$, then for any integer $1 \leq t \leq n$:*

$$\Pr\{y \leftarrow \mathcal{F}(D_1)\} \leq e^{\varepsilon} \Pr\{y \leftarrow \mathcal{F}(D_2)\} + \delta(t) \ ,$$

with $\varepsilon = \ln \beta_0 (1 + \frac{1}{t})$.

A proof of Theorem 4.2 can be found in Appendix A.4. Informally, the theorem says that the change in the probability that the mechanism produces a synthetic y is multiplicatively bounded by $(1 + t^{-1}) \cdot \beta_0$, except with probability $\delta(t)$.[1] Consider adding a record d' to dataset D and the resulting change in the synthesis probability of producing some y. If $\kappa(D', d', y) < t$, then the probability $\Pr\{y \leftarrow \mathcal{M}(d')\}$ is (relatively) large compared to other records in d. But the probability of passing the privacy with d' as seed is at most $\delta(t)$. In contrast, if $\kappa(D', d', y) \geq t$, so that $\Pr\{y \leftarrow \mathcal{M}(d')\}$ is not large compared to other records in d, then by definition of κ, the increase in synthesis probability is bounded because $\Pr\{y \leftarrow \mathcal{M}(d')\} \leq t^{-1} \sum_{d \in D} \Pr\{y \leftarrow \mathcal{M}(d)\}$. In all cases, since the sensitivity of κ is 1, then the probability of passing the privacy test (for any seed) increases by at most β_0 (Definition 3.2) when d' is added to the dataset.

If we use the noisy threshold technique (Privacy Test 3.2) with geometric noise parameter $\alpha = e^{-\varepsilon_0}$ then we have the following corollary.

Corollary 4.1. *For any two non-empty neighboring datasets D_1, D_2, any $y \in U$, if $\mathcal{F}$ denotes Mechanism 3.1 with Definition 4.2 as privacy score function and Privacy Test 3.2 with parameter $k \geq 2$ and geometric noise parameter $\alpha = e^{-\varepsilon_0}$ for $\varepsilon_0 > 0$. Then for any integer $1 \leq t < k$:*

$$\Pr\{y \leftarrow \mathcal{F}(D_1)\} \leq e^{\varepsilon} \Pr\{y \leftarrow \mathcal{F}(D_2)\} + \delta \ ,$$

with $\varepsilon = \varepsilon_0 + \ln(1 + \frac{1}{t})$ and $\delta < e^{-\varepsilon_0(k-t)}$.

[1]This probability be made small by an appropriate choice of t and privacy test function δ. For example, it is a decreasing exponential in $k - t$ in the case of Corollary 4.1.

By tuning k and t, we can make δ arbitrarily small as it is bounded by an exponentially decreasing function of $k - t$. Given their dependence on ε_0, there is an inherent trade-off between ε and δ (as expected).

A key insight to understand Theorem 4.2 is that that the sensitivity of the privacy score function, κ (Definition 4.2), is 1 in the following sense.

Lemma 4.1. *For any dataset D with $|D| \geq 1$ and $D' = D \cup \{d'\}$ for some $d' \in U$, and any $y \in U$:*

$$\kappa(D, d, y) \leq \kappa(D', d, y) \leq \kappa(D, d, y) + 1 \ , \tag{4.5}$$

for any $d \in D$.

We prove Lemma 4.1 in Appendix A.4.

4.3 COMPOSITION

Suppose we want to produce a synthetic dataset of m records. To understand the privacy guarantee we obtain for the entire synthetic dataset, we use composition results for differential privacy. For example, if we use sequential composition then we achieve $(m\varepsilon, m\delta)$-differential privacy. However, if m is large enough, then using advanced composition, the results of Kairouz et al. [35], or the ratio bucket technique [36] yields better bounds.

Suppose we want $(1, 2^{-\lambda})$-differential privacy for a synthetic dataset with m records where $\lambda > 0$ is a security parameter. As a concrete example, we take Corollary 4.1 and advanced composition. How should we set the privacy parameters k, t, and ε_0? We propose the following strategy.

To start, we incorporate the constraint on δ. To satisfy the demand that $\delta \leq 2^{-\lambda}$, we let $\delta'' = 2^{-\lambda}/(m+1)$ and set k, t, and ε_0 such that: $\delta = \exp\{-\varepsilon_0(k-t)\} = 2^{-\lambda}/(m+1)$. In other words, we set: $\varepsilon_0 = \frac{\tilde{\lambda}}{k-t}$ where $\tilde{\lambda} = \lambda \ln 2 + \ln(m+1)$.

With this, the privacy budget for a single invocation of the mechanism (Corollary 4.1) is:

$$\varepsilon_{k,t} = \varepsilon_0 + \ln\left(1 + \frac{1}{t}\right) = \frac{\tilde{\lambda}}{k-t} + \ln\left(1 + \frac{1}{t}\right)$$

where any pair (k, t) such that $1 \leq t < k$ is valid. We propose to take $k = 2t$, so that:

$$\varepsilon_{k,t} = \frac{\tilde{\lambda}}{t} + \ln\left(1 + \frac{1}{t}\right) \leq \frac{\tilde{\lambda} + 1}{t} = \varepsilon_t \ .$$

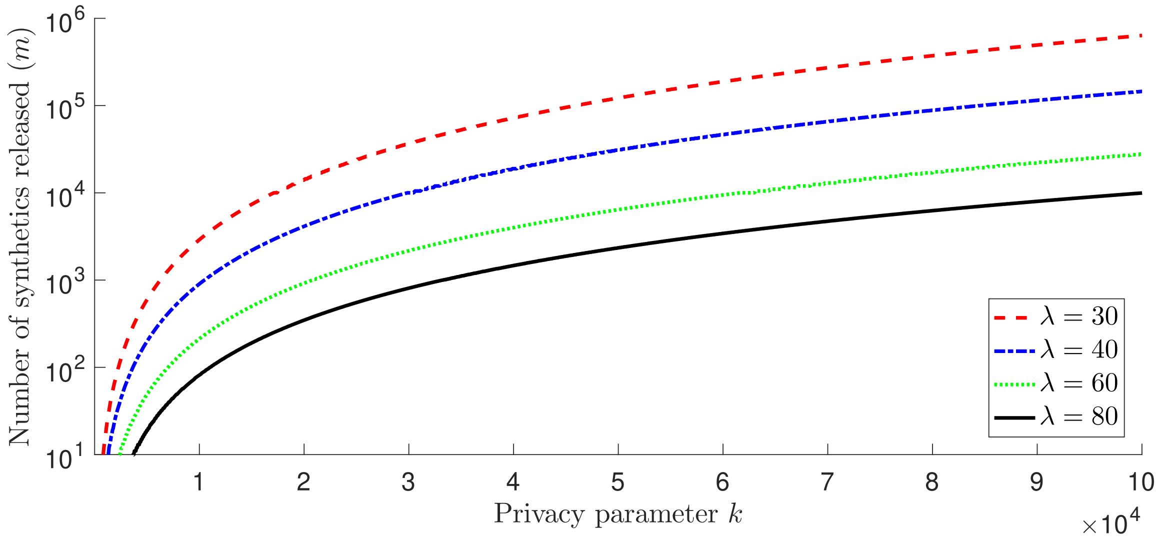


Figure 4.2: Composition: trade-off between the number of synthetics released m and privacy parameter k, where $t = k/2$. Each curve plots a different value λ with respect to the privacy guarantee: $(1, 2^{-\lambda})$-differential privacy.

Then it suffices to set t such that:

$$1 \geq \varepsilon_t \sqrt{2m \ln \frac{1}{\delta''}} + m\varepsilon_t(e^{\varepsilon_t} - 1) = \varepsilon_t \sqrt{2m\tilde{\lambda}} + m\varepsilon_t(e^{\varepsilon_t} - 1) \ . \tag{4.6}$$

In this example, we started with a number of synthetic records m to produce and then derived the parameters to reach the level of privacy desired. Remark that Eq. (4.6) suggests an alternative strategy. Instead of considering the number of synthetics records m fixed and setting t accordingly, we can set t and then determine m, the number if synthetics that can be produced, given the desired level of privacy.

To illustrate composition, Fig. 4.2 shows the trade-off between the number of synthetics produced and the parameter k (with $k = 2t$) to obtain $(1, 2^{-\lambda})$-differential privacy overall for different values of λ (according to Corollary 4.1).

Partitioning. An alternative is partitioning which is also called parallel composition. Instead of using the entire dataset D as input to Mechanism 3.1 we can randomly partition D into m disjoint subsets $D_1, \ldots, D_m$ of (roughly) equal size. This allows us to produce up to m synthetic records by invoking Mechanism 3.1 independently on each of $D_1, \ldots, D_m$. This yields the same privacy guarantee as producing a single synthetic from D. However, given a fixed threshold k, the smaller sub-datasets D_i are, the less likely it is that candidate synthetics will pass the privacy test.

Chapter 5: Utility

This chapter outlines a systematic approach to evaluate the quality of synthetics produced through the framework. In Section 5.1, we propose to evaluate utility loss through a distinguishability-based definition and game. In Section 5.2, we use the distinguishability-based approach to characterize the utility-privacy trade-off offered by the framework and derive bounds on the utility loss. The key insight is that the quality degradation to achieve the privacy guarantee can be expressed in terms of the expected proportion of synthetics that pass the privacy test, which in turns depends on the relative sensitivity of the generative model and the input. Given that precise measurements of relative sensitivity are typically unavailable, we describe an efficient experimental methodology to estimate distinguishability and utility bounds in practice.

A Tale of Two Losses. The quality of privacy-preserving synthetic data can be understood in terms of two losses: (1) the loss from real (input) data to synthetic data produced by the generative model, which we call the *model loss*, and (2) the loss of filtering synthetics candidates into privacy-preserving synthetics that pass the privacy test, which we call the *filtering loss*. The term *utility* refers to the quality degradation between reals and privacy-preserving synthetics which includes both the model loss and the filtering loss.

The crucial difference here, compared to differential privacy for interactive database queries, is that the generative model itself induces loss. This is inherent to data synthesis and applies equally to our framework as to other data synthesis techniques. When given a generative model to use, we can do nothing about the model loss: if the model is bad, its output will be bad. However, we can measure the model loss experimentally using the notion of distinguishability (Section 5.1). Additionally, our notion of distinguishability is transitive so that given the model loss and the filtering loss we can obtain a utility guarantee.

5.1 DISTINGUISHABILITY

If synthetics cannot be distinguished from real records, then surely synthetics can be used instead of real records. This is the intuition behind *distinguishability*, the utility metric we propose in this thesis. An advantage of this metric is that it is application independent and thus it can be used no matter what type of data and the ultimate use of the synthetics.

We formalize the notion of distinguishability as a game.[1] An adversary, modeled as a (possibly randomized) algorithm, is given a record $y \in U$ and has to decide whether y comes from distribution P or Q.

[1] The definition is similar in essence to cryptographic definitions of indistinguishability games [37].

Definition 5.1 (Distinguishing Game)**.**
Input: P and Q probability distributions over U, Adversary $\mathcal{A}$.
Output: success *or* failure*.*

1. *Pick a random bit $b \in \{0, 1\}$ uniformly at random.*
2. *If $b = 0$: $y \leftarrow P$. Otherwise: $y \leftarrow Q$.*
3. *Invoke the adversary to obtain a guess $b' = \mathcal{A}(P, Q, y)$.*
4. *If $b = b'$, then output* success*.*
   Otherwise, output failure*.*

If there exists an adversary that wins the game (for distributions P and Q) with probability $\frac{1+\gamma}{2}$, we say that P's distinguishability from Q is γ, or that P and Q are γ-distinguishable.
Definition 5.1 can capture utility loss, model loss, and filtering loss by appropriately choosing P and Q. If P is the distribution of real data and Q is the distribution our privacy-preserving synthetics (Section 3.2), then the distinguishing game measures the utility loss. In contrast, if P is the distribution of real data and Q is the distribution of basic synthesis on some dataset D, then the distinguishing game measures the model loss. Similarly, if P is the distribution of basic synthesis on D and Q the distribution our privacy-preserving mechanism (Section 3.2) on D, then the distinguishing game measures the filtering loss. Remark that distinguishability (with Definition 5.1) depends on the adversary. So, we would like to measure distinguishability with respect to the adversary with the highest success rate.

We define the best adversary and its advantage for arbitrary discrete distributions. Let P and Q be discrete probability distributions over a set of events X such that p_x (resp. q_x) denotes the probability of event x according to P (resp. Q). Given a sample $x \in X$, we can optimally distinguish P from Q through the likelihood ratio p_x/q_x. The advantage in distinguishing P from Q with respect to sample x is $A(P, Q, x) = 2\frac{\max\{p_x, q_x\}}{p_x + q_x} - 1$. We have:

- If $p_x = q_x$, then $A(P, Q, x) = 0$.
- If $p_x > 0$, $q_x = 0$ (or $p_x = 0$, $q_x > 0$): $A(P, Q, x) = 1$.
- If $A(P, Q, x) \leq \gamma$ for $0 \leq \gamma < 1$: $\max\{\frac{p_x}{q_x}, \frac{q_x}{p_x}\} \leq \frac{1+\gamma}{1-\gamma}$.

Definition 5.2 (max advantage)**.** *Let P and Q denote two discrete probability distributions with support X. The* max advantage *in distinguishing P and Q from a single sample is:*

$$A(P, Q) = \max_{x \in X} A(P, Q, x) = \max_{x \in X} \frac{|p_x - q_x|}{p_x + q_x} . \tag{5.1}$$

If $A(P, Q) \leq \gamma$ for $0 \leq \gamma < 1$, we say that P and Q are *max γ-indistinguishable.* In contrast, if $A(P, Q) \geq \gamma$, then we say that P and Q are *γ-distinguishable.*

Definition 5.3 (mean advantage)**.** *Let P and Q denote two discrete probability distributions with support X. The* mean advantage *in distinguishing P and Q from a single sample is:*

$$\bar{A}(P,Q) = \sum_{x \in X} \Pr\{x; P, Q\} \cdot A(P,Q,x) \ . \tag{5.2}$$

If $\bar{A}(P,Q) \leq \gamma$ for $0 \leq \gamma < 1$, we say that P and Q are *mean γ-indistinguishable* or simply *γ-indistinguishable*. In contrast, if $\bar{A}(P,Q) \geq \gamma$, we say that P and Q are *γ-distinguishable*. Observe that if $A(P,Q) \leq \gamma$ then: $\bar{A}(P,Q) \leq \gamma$.

Since the sample is equally likely to come from P or Q, we have $\Pr\{x; P, Q\} = \frac{1}{2}(p_x + q_x)$. Thus, $\bar{A}(P,Q)$ is exactly the statistical distance between P and Q:

$$\bar{A}(P,Q) = \sum_{x \in X} \frac{p_x + q_x}{2} \cdot \frac{|p_x - q_x|}{p_x + q_x} = \mathrm{SD}(P,Q) \ . \tag{5.3}$$

Using Eq. (5.3) and the standard triangle inequality, it can be seen that the mean advantage (and thus the mean distinguishability) is transitive in the following sense. If P and R are γ_1-indistinguishable, and R and Q are γ_2-indistinguishable, then P and Q are γ-indistinguishable for $\gamma = \gamma_1 + \gamma_2$. This property allows us to make end-to-end indistinguishability statements: i.e., if the model loss is γ_1 and the filter loss is γ_2, then the utility loss is at most $\gamma_1 + \gamma_2$.

Experimentally. In practice, we can (typically) only measure distinguishability using Definition 5.1 instead of Eqs. (5.1) to (5.3) due to the dimension of the data which makes computation intractable. Thus, we choose an adversary $\mathcal{A}$ to play the distinguishing game and measure its success rate. For this, we can use machine learning techniques and train a classifier to play the distinguishing game. We give the classifier a labeled dataset of records from P and Q, and use its classification accuracy on a disjoint test set to estimate distinguishability. For example, this is the approach we take in Chapters 8 and 9, where decision trees and random forest classifiers are found to perform well.

5.2 UTILITY-PRIVACY TRADE-OFF

We apply the distinguishability-based methodology of the previous section to the mechanism with privacy tests introduced in Section 4.2 and characterize the utility-privacy trade-off in terms of (ε, δ)-differential privacy and filtering loss (distinguishability). The key observation is that if the relative sensitivity is (upper-)bounded then the probability of passing the privacy test is (lower-)bounded. This results in allowing us to characterize the filtering loss, measured using distinguishability, in terms of relative sensitivity.

In Section 5.2.1, we tackle the case where the relative sensitivity is bounded everywhere (i.e., for all synthetics). In Section 5.2.2 we relax this assumption and derive bounds on the filtering loss. Finally, given that precise measurements of the relative sensitivity are not always available, we propose an efficient methodology to derive bounds based on empirical measurements.

Preliminaries. To understand the trade-off we need to compare the synthesis probability of the mechanism, $\Pr\{y \leftarrow \mathcal{F}(D)\}$, with those of basic synthesis, $\Pr\{y \leftarrow \mathcal{G}(D)\}$. For this, recall (from Chapter 3) that:

$$\Pr\{y \leftarrow \mathcal{G}(D)\} = |D|^{-1} \sum_{d \in D} \Pr\{y \leftarrow \mathcal{M}(d)\} \text{ , and}$$

$$\Pr\{y \leftarrow \mathcal{F}(D)\} = |D|^{-1} \sum_{d \in D} \Pr\{y \leftarrow \mathcal{M}(d)\} \cdot \delta(D, d, y) \ .$$

For conciseness, we write $P_D^{\mathcal{H}}(y) = \Pr\{y \leftarrow \mathcal{H}(D)\}$ for any mechanism $\mathcal{H}$.

A difficulty is that we cannot compare the synthesis probabilities $P_D^{\mathcal{G}}(y)$ and $P_D^{\mathcal{F}}(y)$ directly because the two distributions have different support. Indeed, $\mathcal{F}$ produces $\perp$ as output when the privacy test fails. Instead, since we only release $y \in U$, we have to compare it to:

$$\tilde{P}_D^{\mathcal{F}}(y) = \frac{P_D^{\mathcal{F}}(y)}{\sum_{y \in U} P_D^{\mathcal{F}}(y)} = c^{-1} P_D^{\mathcal{F}}(y) \ ,$$

where $c = \sum_{y \in U} P_D^{\mathcal{F}}(y) = 1 - P_D^{\mathcal{F}}(\perp)$ is a normalization constant.[2]

The following is the key observation relating the ratio of the synthesis probabilities $P_D^{\mathcal{G}}(y)$ and $P_D^{\mathcal{F}}(y)$ with the probability of passing the privacy test.

Observation 5.1. *Let $\mathcal{F}$ denote Mechanism 3.1 with any well-behaved privacy test function δ. For any $y \in U$, we have the following.*

- *If $P_D^{\mathcal{G}}(y) = 0$, then: $P_D^{\mathcal{F}}(y) = 0$.*
- *If $P_D^{\mathcal{G}}(y) > 0$:*

$$\frac{\bar{\delta}_y}{c} \leq \frac{\tilde{P}_D^{\mathcal{F}}(y)}{P_D^{\mathcal{G}}(y)} \leq \frac{1}{c} \ , \tag{5.4}$$

where $\bar{\delta}_y = \min_{d \in D} \delta(D, d, y)$.

[2]Note that $c > 0$ for the privacy tests we consider because they always assign a non-zero probability of passing the test.

Further, observe that $c = \sum_{y \in U} P_D^{\mathcal{F}}(y) \geq \bar{\delta}$ for $\bar{\delta} = \min_{y \in U} \bar{\delta}_y = \min_{y \in U, d \in D} \delta(D, d, y)$. As a result, if $P_D^{\mathcal{G}}(y) > 0$, then it follows from Eq. (5.4) that:

$$\max \left\{ \frac{\tilde{P}_D^{\mathcal{F}}(y)}{P_D^{\mathcal{G}}(y)}, \frac{P_D^{\mathcal{G}}(y)}{\tilde{P}_D^{\mathcal{F}}(y)} \right\} \leq \bar{\delta}^{-1} .$$

In other words, a bound on the ratio of synthesis probabilities $P_D^{\mathcal{G}}(y)$ and $\tilde{P}_D^{\mathcal{F}}(y)$ depends only on the minimum probability of passing of the privacy test, which for the test of Section 4.2, depends on relative sensitivity.

5.2.1 Bounded Relative Sensitivity

The Triangle Lemma (Lemma 3.1) tells us that no mechanism achieves a utility-privacy trade-off beyond the constraint given by the relative sensitivity of $\mathcal{G}$ at D, y.

Suppose that the relative sensitivity of $\mathcal{G}$ at D is bounded by some constant $\eta \geq 1$ for all $y \in U$, i.e.: $1 \leq \tilde{\Delta}\mathcal{G}(D, y) \leq \eta$. This assumption immediately results in a lower bound on the value of κ (Section 4.2) from which we can fully characterize the utility-privacy trade-off offered by our mechanism in terms of ε, δ, and γ.

Lemma 5.1. *If for all $y \in U$: $1 \leq \tilde{\Delta}\mathcal{G}(D, y) \leq \eta < \infty$ and $\mathcal{F}$ denotes Mechanism 3.1 with privacy test (κ, δ) (Definition 4.2) for any well-behaved δ with $\beta_0 = e^{\varepsilon_0}$, $c_0 = 1$.*

Then, for any integer $1 \leq t \leq n$:

- *$\mathcal{F}$ satisfies differential privacy for $\varepsilon = \varepsilon_0 + \ln(1 + t^{-1})$ and $\delta = \delta(t)$.*
- *The output distribution of $\mathcal{F}(D)$ over U is max γ-indistinguishable from that of $\mathcal{G}(D)$ for $\gamma = \frac{1-\delta(\tilde{\eta})}{1+\delta(\tilde{\eta})}$, where $\tilde{\eta} = ((1 + |D|^{-1})\eta - 1)^{-1}$.*

Example 5.1. Take the mechanism and privacy test of Corollary 4.1, and suppose we want (ε, δ)-differential privacy for $\varepsilon = 1$ and $\delta = 2^{-40}$. If $|D| = 10000$ and $\eta = 1.02$ is a bound on the relative sensitivity as in Lemma 5.1, then it suffices to take $k = 40$, $t = 5$, and $\varepsilon_0 = 0.8$ to ensure γ-indistinguishability for $\gamma \approx 0.0001$.

The following example shows that our wish to minimize δ is what keeps us from achieving a utility-privacy trade-off close to that of the Triangle Lemma (Lemma 3.1).

Example 5.2. Recall from the Triangle Lemma that if η is a bound on the relatively sensitivity, then no mechanism can achieves ε-differential privacy and γ-indistinguishability (for $\gamma = \frac{\alpha-1}{\alpha+1}$) unless $\eta \leq \alpha^2 e^{\varepsilon}$. Equivalently, if we want to have α arbitrarily close to 1, then we must have $\varepsilon \geq \ln \eta$.

From Lemma 5.1, we know that we can achieve γ-indistinguishability with our mechanism for γ arbitrarily close to 0 (i.e., α arbitrarily close to 1) by choosing the privacy test function such that $\delta(\tilde{\eta})$ is arbitrarily close to 1.[3] If we further set $t = \tilde{\eta}$, then the mechanism achieves $(\varepsilon, \delta(t))$-differential privacy for $\varepsilon = \ln \eta + \ln \beta_0(1 + |D|^{-1})$. Given that the dataset D could be arbitrarily large and that we can choose β_0 arbitrarily close to 1, this is bound is arbitrarily close to that of the Triangle Lemma!

Unfortunately, if δ is set such that $\delta(\tilde{\eta})$ is arbitrarily close to 1, then so is $\delta(t)$. Thus, if we need to minimize $\delta(t)$ (as we should) then we must set $t < \tilde{\eta}$ (or perhaps even $t \ll \tilde{\eta}$) which results in a privacy guarantee clearly worse than optimal.

5.2.2 Bounds for the Unbounded Case

Lemma 5.1 says that the loss (in the sense of γ-indistinguishability) introduced by $\mathcal{F}$ depends on the relative sensitivity of $\mathcal{G}$ at D. What if the relative sensitivity is not bounded everywhere? For example, there may exist $y \in U$ such that $\tilde{\Delta}\mathcal{G}(D, y) = \infty$.

The following generalizes Observation 5.1 to deal with the case where the relative sensitivity is unbounded for some outputs.

Observation 5.2. *Let $\mathcal{F}$ denote Mechanism 3.1 with privacy test (κ, δ) (Definition 4.2). Let $t > 0$, $Y_{t^-} = \{y \in U : \min_{d \in D} \kappa(D, d, y) < t\}$ and define $\beta_t = \sum_{y \in Y_{t^-}} P^{\mathcal{G}}_D(y)$. For any $y \in U$ such that $P^{\mathcal{G}}_D(y) > 0$, we have the following.*

- *If $y \in Y_{t^-}$:*
$$0 \leq \frac{\tilde{P}^{\mathcal{F}}_D(y)}{P^{\mathcal{G}}_D(y)} \leq [\delta(t)(1 - \beta_t)]^{-1} ,$$
- *If $y \in U \setminus Y_{t^-}$:*
$$\delta(t)(1 - \beta_t) \leq \frac{\tilde{P}^{\mathcal{F}}_D(y)}{P^{\mathcal{G}}_D(y)} \leq [\delta(t)(1 - \beta_t)]^{-1} .$$

Recall that the statistical distance is the same as the mean distinguishability (Eq. (5.3)). The following shows a bound on the filtering loss using Observation 5.2.

Lemma 5.2. *Let $\mathcal{F}$ denote Mechanism 3.1 with privacy test (κ, δ) (Definition 4.2). Then, for any $t > 0$:*
$$\mathrm{SD}(P^{\mathcal{G}}_D, \tilde{P}^{\mathcal{F}}_D) \leq \begin{cases} \beta_t + \frac{1-\delta(t)}{2\delta(t)} & \text{if } \alpha_t \leq 2 \\ \frac{\alpha_t - 1}{2} & \text{if } \alpha_t > 2 \end{cases}$$
where $\beta_t = \sum_{y \in Y_{t^-}} P^{\mathcal{G}}_D(y)$ and $\alpha_t = [\delta(t)(1 - \beta_t)]^{-1}$.

[3]For well-behaved tests (Definition 3.2), we can never have $\delta(x) = 1$ for any x, but we can (in principle) have $\delta(x)$ arbitrarily close to 1 for some x, while keep β_0 small.

Example 5.3. Suppose we pick t such that $\delta(t) = 0.9$, and that $\beta_t = 0.01$. Then $\alpha_t \approx 1.122$ and the statistical distance is $\mathrm{SD}(P_D^{\mathcal{G}}, \tilde{P}_D^{\mathcal{F}}) \approx 0.066$, or equivalently the output distribution of $\mathcal{F}$ on D is 0.066-indistinguishable from that of basic synthesis.

Experimental Quantification. Lemma 5.2 enables us to predict the quality of the synthetics for various tasks. The problem is that β_t is unknown and intractable to compute in general. Experimentally, we propose to estimate β_t as follows. Invoke $\mathcal{G}(D)$ to obtain a synthetic dataset $Y = \{y_1, y_2, \ldots, y_m\}$. Then compute:

$$\hat{\beta}_t = \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}_{\min_{d \in D} \kappa(D,d,y_i) < t} \ . \tag{5.5}$$

This is equivalent to flipping a biased coin (with probability of heads β_t) m times and computing the proportion of heads.

We can then use Lemma 5.2 by plugging in our estimate for β_t (and computing α_t accordingly). However, for example, there is no guarantee that the obtained bound on $\mathrm{SD}(P_D^{\mathcal{G}}, \tilde{P}_D^{\mathcal{F}})$ holds. It could happen that our estimate of β_t is less than its true value. A remedy is to use the upper edge of a $1 - \alpha$ confidence interval instead.

There are several ways to compute a confidence interval on β_t given $\hat{\beta}_t$ and m. For example, a $1 - \alpha$ approximate confidence interval for β_t is given by:

$$\hat{\beta}_t \pm z \sqrt{\frac{\hat{\beta}_t(1 - \hat{\beta}_t)}{m}} \ ,$$

where z is the $1 - \frac{\alpha}{2}$-quantile of the standard normal distribution [38].

Chapter 6: Generative Models

All models are wrong, but some are useful.

George E. P. Box

This chapter presents several probabilistic generative models.

6.1 SYNTHESIZING LOCATION TRAJECTORIES

This section describes a seedbased model to generate synthetic location trajectories. For a complete description of this model and how to use it, we refer the reader to [33]. The model is evaluated experimentally in Chapter 7.

6.1.1 Generative Model

We assume that time and space are discrete, so that a location trace is a sequence $r_1, r_2, \dots, r_t$ of visited locations over time. We denote by R the set of locations and each $r_i \in R$ is represents a location visit. Given a real (seed) location trace we describe a generative model to synthesize a similar location trace. The generative model is based on the idea that a trace can be decomposed into a geographic component and a semantic component. The geographic dimension is what is sensitive because it reveals where a person lives and the location visited. In contrast, the semantic dimension is what is useful from a human mobility perspective; it captures the person's lifestyle and the kinds of places visited. Fig. 6.1 illustrates this idea.

At a high-level, the model transforms a seed trace into the semantic space (discarding the geographic information) and then probabilistically transforms it back to the geographic space, thereby obtaining a (new) semantically plausible synthetic trace. The geographic and semantic domains are defined implicitly by two similarity metrics defined based on Markovian mobility models (Section 6.1.2). We first describe the generative model abstracting away the specific of the mobility model and similarity metrics.

The generative model has a training step as pre-processing before it can be used to synthesize trajectories. Let D_S be the input dataset of traces used for synthesis. The training step uses a separate dataset of traces D_T over the same location space R which are public or disjoint from D_S. The training step creates an aggregate mobility model and a location semantic graph which captures the semantic of locations (according to our semantic similarity metric).

The transition probability matrix of user u is denoted as $p(u)$ and the corresponding visiting probability vector is written $\pi(u)$. The mobility profile of user u is denoted $\langle p(u), \pi(u) \rangle$. The aggregate mobility model is $\langle \bar{p}, \bar{\pi} \rangle$. T denotes the number of time periods. We use $\mathsf{sim_G}(u, v)$ and $\mathsf{sim_S}(u, v)$ to denote the geographic and semantic similarity between the mobility of u and v, respectively. For a trace s, $s(t)$ denotes the user's location at time t.

The pre-processing step is described in Algorithms 6.1 to 6.3. To simplify the presentation, we assume a single time period.

Algorithm 6.1 (Pre-process).
Input: location set R, training dataset D_T, and desired number of clusters k.
Output: semantic clustering C, and aggregate mobility $\langle \bar{p}, \bar{\pi} \rangle$.

```
1. Set aggregate mobility model $\langle \bar{p}, \bar{\pi} \rangle$ be average of $\langle p(u), \pi(u) \rangle$ over all $u \in D_T$
2. Set $C = \mathsf{SemanticClustering}(R, D_T, k)$
3. Return $C, \langle \bar{p}, \bar{\pi} \rangle$.
```

Algorithm 6.2 (Semantic Clustering).
Input: location set R, training dataset D_T, and desired number of clusters k.
Output: semantic clustering C.

```
1. Create weighted graph $G$ with locations $R$ as vertices
2. For all pairs of trajectories $u, v \in D_T, u \neq v$, do:
    (a) Set $s_u^v, \sigma_u^v = \mathsf{SemanticSimilarity}(u, v)$
    (b) For all locations $r, r' \in R$ such that $r' = \sigma_u^v(r)$:
        • Set edge weight $w_G(r, r') = w_G(r, r') + s_u^v$
3. Set $C = \mathsf{K\text{-}Means}(G, k)$
4. Return $C$
```

Algorithm 6.3 (Semantic Similarity).
Input: trajectories u, v.
Output: semantic similarity score $\mathsf{sim_S}(u, v)$, and optimal mapping σ_u^v.

```
1. Compute mobility models $\langle p(u), \pi(u) \rangle$ and $\langle p(v), \pi(v) \rangle$
2. Compute optimal mapping $\sigma_u^v$ from Eq. (6.8)
3. Compute semantic similarity $\mathsf{sim_S}(u, v)$ from Eq. (6.9)
4. Return $\mathsf{sim_S}(u, v), \sigma_u^v$.
```

Learning an aggregate mobility model. We construct the aggregate mobility model by averaging the mobility models of all traces in dataset D_T, as well as giving a small probability to the possible movements between locations according to their distance and connectivity. More precisely, we compute the aggregate transition probability $\bar{p}_r^{r'}$ as $z_r^{-1} \cdot \sum_{u \in D_T} p_r^{r'}(u) + \epsilon \cdot \max(1, d(r, r'))^{-2}$, where ϵ is a small constant, $d(r, r')$ is the distance between locations r and r', and z_r is a normalization constant. We compute the aggregate visiting probability $\bar{\pi}^r$ as the average of $\pi^r(u)$ for $u \in D_T$.

Learning a location semantic graph. We analyze and discover the semantic relation between different locations using traces in the training data. To this end, we propose a *semantic similarity metric* (Section 6.1.2). Intuitively, we assign a higher similarity value to a pair of locations if multiple individuals have similar spatiotemporal activities in them. We find the optimal way to map the visited locations in a pair of traces such that the mapping maximizes the statistical similarity between their mobility models. The semantic similarity metric is therefore the statistical similarity between mobility models under the optimal semantic mapping between locations. This means that if we were to translate the locations visited by two individuals according to the discovered best mapping, they would follow the same mobility model when their semantic similarity is high (i.e., have similar life styles). For example, if Alice and Bob spend all day at their respective work locations w_A and w_B, and all night at their respective home locations h_A and h_B their mobility models are highly semantically similar, though it may be that $h_A \neq h_B$ and $w_A \neq w_B$. In this example, the best semantic mapping between locations is $w_A \leftrightarrow w_B$ and $h_A \leftrightarrow h_B$. The proposed semantic similarity metric goes beyond home and work; it is over all locations, so that that Alice's favorite bar could be mapped to Bob's favorite nightclub, if Alice and Bob visit those places in a similar way.

For each pair of mobility models of traces in the training dataset D_T, we compute their semantic similarity and the best semantic mapping between their locations. Note that the semantic similarity is quantifying the similarity of two mobility models, not that of two location traces. We then aggregate all the location matchings across all trace pairs, with weights based on the semantic similarity between mobility models, and construct a *location semantic graph*, where the nodes are locations and the weight of the edges is the average semantic similarity between the locations over the dataset.

The location semantic graph enables us to infer which locations have similar meanings (or purpose) for different people. The locations that have higher semantic similarity can be grouped together to represent one *location semantic class*. We run a clustering algorithm on the location semantic graph to partition locations into distinct classes. Regardless of their geographic positions, the locations that fall into the same class are visited in the same way by different people. In other words, their visit probability, time of visit, and the probabilities of transition from/to them to/from other locations with the same type is similar. Thus, we can consider them as being semantically equivalent. So, using the notation of our previous example, w_A and w_B should belong to the same cluster that can represent "workplace" locations, and h_A and h_B should be grouped into another cluster representing residential or "home" locations.

We compute the semantic similarity between all locations in R, and create a location semantic graph $G\langle R, E, w\rangle$ such that the vertices are in R and the weight $w_G(r, r')$ on the edge between locations r and r' is the weighted sum of the number of pairs of users u and v for whom r and r' are semantically mapped (i.e., $r = \sigma_u^v(r')$), weighted according to their similarity. Then, we create the equivalent semantic classes C by running a clustering algorithm on this graph. We make use of the k-means clustering algorithm, and we choose the number of clusters such that it optimizes the clustering objective.

Transforming a trace into the semantic domain. We transform the seed trace s into its corresponding semantic trace s_{sem} by simply replacing each location in the trace with all its semantically equivalent locations (according to the semantic classes C). Fig. 6.2 depicts an example of such a semantic seed. Intuitively, this composite trace encompasses all possible geographic traces that are semantically similar to the original seed trace. To be flexible with respect to traces we can generate, we add randomness to the semantic seed trace. During the transformation process of the seed trace into the semantic trace, we sub-sample locations from the semantic classes (instead of using them all). For each cluster, we also remove each location independently with probability p_c, resulting in a new cluster. Further, we allow locations of different classes to merge into each other around time instants where the user moves from one class to the other: we add a location from one time step to another with a Δt gap with a geometric probability ${p_m}^{\Delta t}$.

Sampling a trace given the semantic seed. Any random walk on the semantic seed trace that travels through the available locations at each time is a valid location trace that is semantically similar to the seed trace. However, synthetic traces need to be geographically consistent with the general mobility of people in the considered area. We cast the problem of sampling such traces as a decoding problem in Hidden Markov Models (HMMs) [39]. The hidden states are the visited locations, the observed output are the semantic classes (which are the set of semantically equivalent locations in the same class), and the transition probability matrix follows the aggregate mobility model.

By decoding the semantic trace into geographic traces, we generate traces that are plau-

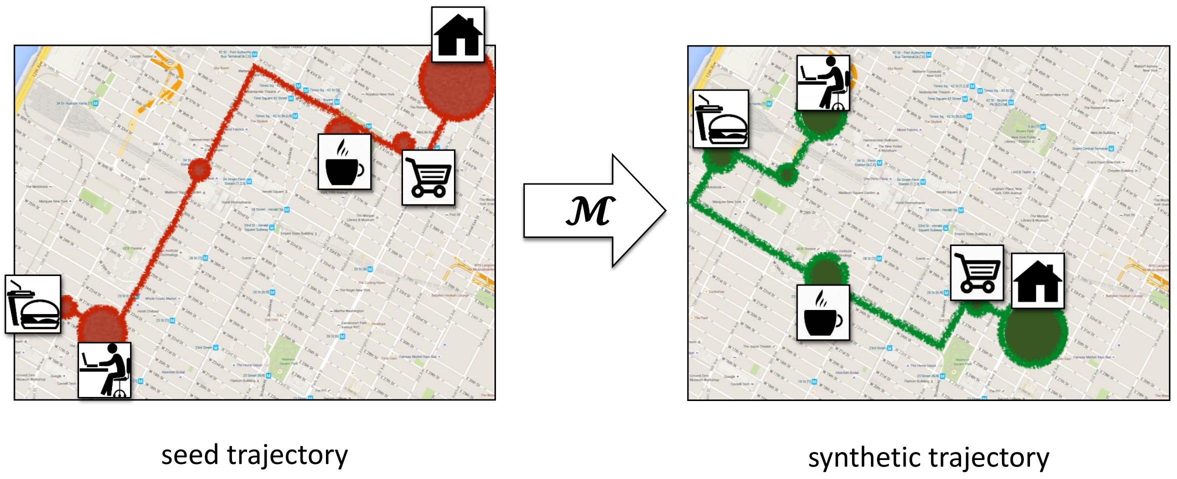


Figure 6.1: Illustration of the generative model for location trajectories.

sible according to aggregate mobility models, i.e., there could be an individual who could have made that trace. Among existing HMM decoding algorithms, we use the Viterbi algorithm [40] that finds

$$\arg\max_{y} \Pr\{y | s_{\text{sem}}(t), \langle \bar{p}, \bar{\pi} \rangle\}$$

assuming that $y(t)$ can only choose from locations in $s_{\text{sem}}(t)$. Finding the most likely synthetic trace is equivalent to finding the shortest path in an edge-weighted directed graph where each location at time t is linked to all locations at time $t+1$ in the semantic seed trace.

This ensures that the sampled trace is consistent with the general mobility of the area. However, the Viterbi algorithm produces a single trace: the most probable one. To ensure that the generative model is capable of producing different synthetic traces for each seed, we randomize the trace reconstruction of Viterbi as follows. We perturb the probabilities so that at each step the algorithm randomly selects one location among a set of locations that are highly probable (instead of always selecting the most probable location as the original algorithm dictates). We implement this idea by multiplying the transition probabilities of moving from one location to the next with a random number between 1 and p_v, where $p_v \geq 1$ is a parameter.

The synthesis process to transform a seed trajectory into a synthetic trajectory (after pre-processing) is described in Algorithm 6.4. The algorithm uses the semantic clustering C obtained through Algorithm 6.2. We let C_i represent the set of locations belonging to the cluster i. Also, the procedure HMMDecode() refers to the aforementioned randomized Viterbi procedure with parameter p_v.

Algorithm 6.4 (Location Trace Synthesizer).
Input: seed trajectory s, aggregate mobility $\langle \bar{p}, \bar{\pi} \rangle$, and parameters p_c, p_l, p_m.
Output: synthetic trajectory y.

```
1. Set $C' = C$
2. Update $C'$ by removing locations in any partition with probability $p_c$
3. Set semantic seed $s_{\text{sem}} = s$
4. Update $s_{\text{sem}}$ by replacing locations $r$ with $C'_i$ where $r \in C'_i$
5. Update $s_{\text{sem}}$ by removing the location $r = s(t)$ from time $t$ with probability $p_l$
6. Update $s_{\text{sem}}$ by merging locations within time $\Delta t$ with probability ${p_m}^{\Delta t}$
7. Let $y \leftarrow \mathsf{HMMDecode}(s_{\text{sem}}, \langle \bar{p}, \bar{\pi} \rangle)$
8. Return $y$.
```

6.1.2 Mobility Model

We model the user mobility as a time-dependent first-order Markov chain on the set of regions (locations). As users have different activities and mobility patterns during different periods of time, we assume that time is partitioned into time periods, e.g., morning - afternoon - evening - night. So, the mobility profile $\langle p(u), \pi(u) \rangle$ of a given user u is a *transition* probability matrix of the Markov chain associated with the user's mobility (from a region to another), and the user's *visiting* probability distribution over the regions, respectively. Note that these probabilities are dependent on each other, and together they constitute the joint probability of two regions that are subsequently visited by the user. The entry $p^{r'}_{r,\tau,\tau'}(u)$ of $p(u)$ is the probability that user u will move to region r' in the next time instant (which will be in time period τ'), given that she is now (in time period τ) in region r. The entry $\pi^r_\tau(u)$ is the probability that user u is in region r in time period τ. We can compute $\pi(u)$ from traces or directly from $p(u)$ (in some circumstances). Let the random variable $\mathbf{A}^t_u$ represent the actual location of user u at time t, and τ^t be the time period associated with $\mathbf{A}^t_u$. So, the mobility profile of a given user u consists of the following probabilities:

$$\begin{aligned} &p^{r'}_{r,\tau,\tau'}(u) = \Pr\{\mathbf{A}^{t+1}_u = r' \mid \mathbf{A}^t_u = r; \tau^{t+1} = \tau', \tau^t = \tau\}, \\ &\pi^r_\tau(u) = \Pr\{\mathbf{A}^t_u = r; \tau^t = \tau\} \end{aligned} \tag{6.1}$$

This Markovian model can predict the location of an individual to a great extent, as it takes both location and time aspects into account. It can become even more precise, by increasing its order, or by enriching its state. We can incorporate new dimensions similar to

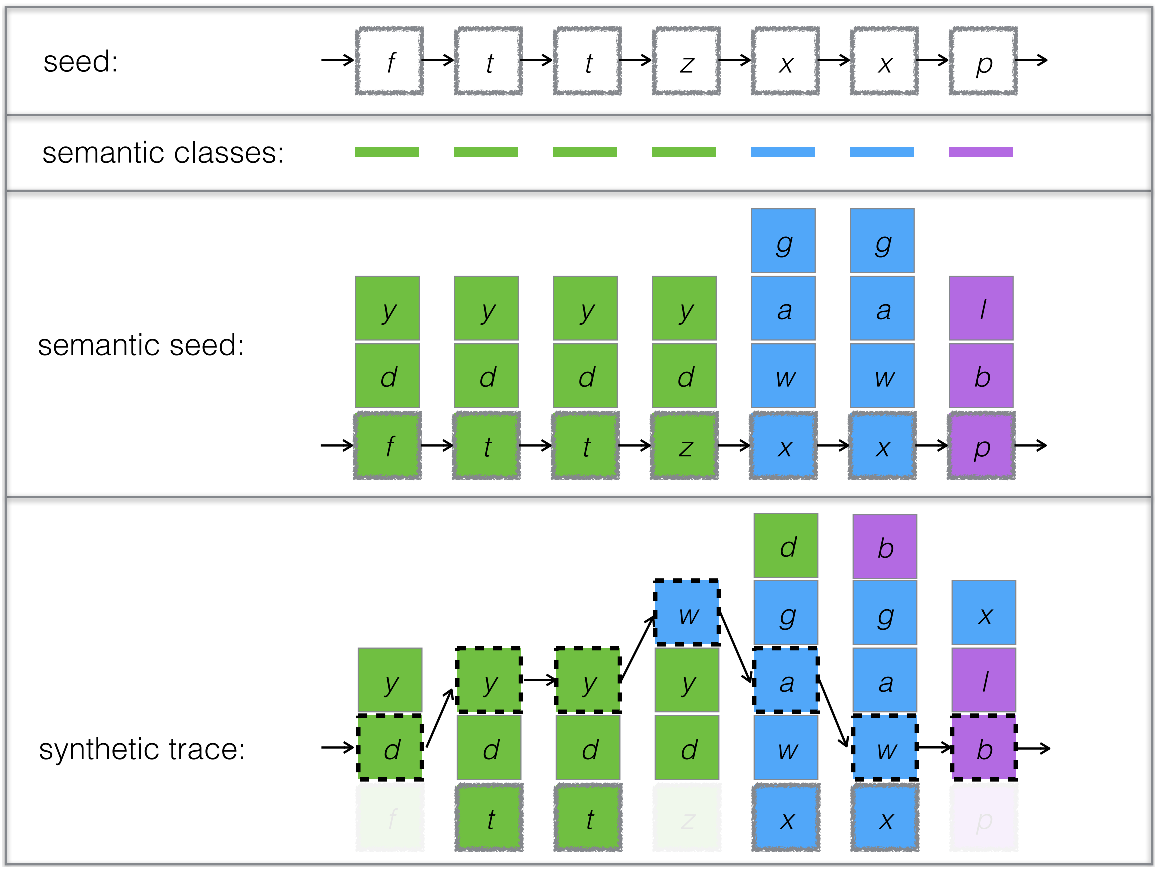


Figure 6.2: Generating a synthetic trace from a seed. Each location is represented by an English letter in a box. The semantic class associated with each location is represented by a different color. The semantic seed trace includes the locations in the seed along with other locations in the same cluster at each time instant. Here, locations are clustered as $\{y, d, f, t, z\}, \{g, a, w, x\}, \{l, b, p\}$. To generate a synthetic trace, we first probabilistically remove the seed location and probabilistically merge subsequent classes. In this example, f, z, p are removed, and w, d, b, x are merged into their neighboring visited clusters. We then run a decoder to generate a probable trace given the possibility of choosing from all available locations at each time instant. The resulting synthetic trace is shown with connected dashed boxes.

the way we model the time periods. To learn the probabilities of the mobility profile Eq. (6.1), from location traces, we can use maximum likelihood estimation (if the traces are complete) or make use of algorithms such as Gibbs sampling (if the traces have missing locations or are noisy) [41].

Mobility Similarity Metrics. We propose two metrics to compare the mobility of two users and compute their similarities: *geographic* and *semantic* similarity. The *geographic similarity* metric captures the correlation between location traces that are generated by two mobility profiles. It reflects if two users visit similar locations over time with similar probabilities and if they move between those locations also with similar probabilities. Using this metric, for example, two individuals who live in the same region A and their workplace is in the same region B potentially have very similar mobility, as they spend their work hours in B and most of their evenings in A.

The geographic similarity between the mobility models of two random individuals is usually low. However, if we ignore their exact visited locations, they tend to share similar patterns for visiting locations with similar semantics (locations therein they have similar activities). Consider the semantic dimension of locations as a coloring of them on the map. Besides the geographic correlation between location traces, we can compute their correlation at the semantic level too (by reducing the set of locations to colors and computing the similarity of colored traces). This is the intuition behind the *semantic similarity* metric. If the pair of locations that two individuals visit over time have the same semantic, their mobility models are also semantically similar (even if they do not intersect geographically). For example, if we transform trace X by replacing its locations with their corresponding semantically similar locations in trace Y, the transformed trace becomes statistically similar to Y. So, two traces are semantically similar if their locations can be mapped (translated) to each other in this way.

Geographic Similarity. We define this similarity metric based on the Earth Mover's Distance (EMD) for probability distributions. The EMD is widely used in a range of applications [42, 43], and can be understood by thinking of the two distributions as piles of dirt where it represents the minimum amount of work needed to turn one pile of dirt (i.e., one distribution) into the other; the cost of moving dirt being proportional to both the amount of dirt and the distance to the destination. The special case of EMD for probability distributions has been shown to be equivalent to the Mallows distance [44].

Let $\mathbf{X}$ and $\mathbf{Y}$ be discrete random variables with probability distributions p and q, such that $\Pr\{\mathbf{X} = x_i\} = p_i$ and $\Pr\{\mathbf{Y} = y_i\} = q_i$, respectively, for $i = 1, 2, \dots, n$. We also have $\sum_i p_i = 1$ and $\sum_i q_i = 1$.

Definition 6.1. *(From [44]) Let $d(\cdot)$ be an arbitrary distance function between $\mathbf{X}$ and $\mathbf{Y}$. The* Mallows distance *$M_d(p, q)$ is defined as the minimum expected distance between $\mathbf{X}$ and $\mathbf{Y}$ with respect to $d(\cdot)$ and to any joint distribution function f for $(\mathbf{X}, \mathbf{Y})$ such that p and q are the marginal distributions of $\mathbf{X}$ and $\mathbf{Y}$, respectively.*

$$M_d(p, q) = \min_f \{\mathbb{E}_f[d(\mathbf{X}, \mathbf{Y})] : (\mathbf{X}, \mathbf{Y}) \sim f, \mathbf{X} \sim p, \mathbf{Y} \sim q\}, \tag{6.2}$$

where the expectation, minimized under f, is

$$\mathbb{E}_f[d(X,Y)] = \sum_{i=1}^{n}\sum_{j=1}^{n} f_{ij}\ d(x_i, y_j). \tag{6.3}$$

In addition to the two constraints $\sum_{i=1}^{n}\sum_{j=1}^{n} f_{ij} = 1$ and $f_{ij} \geq 0$, for all i, j, the joint probability distribution function f must also satisfy $\sum_{i=1}^{n} f_{ij} = q_j$ and $\sum_{j=1}^{n} f_{ij} = p_i$.

Note that, for given p and q, the minimum f is easily computed by expressing the optimization problem as a linear program.

Using the previous definition, we define the geographic similarity metric based on the Mallows distance.

Definition 6.2. *Let $d(\cdot)$ be an arbitrary distance function. The* dissimilarity *between two mobility profiles $\langle p(u), \pi(u)\rangle$ and $\langle p(v), \pi(v)\rangle$ (belonging to individuals u and v), is defined as the expected Mallows distance of the next random locations $\mathbf{r}'$ and $\mathbf{r}''$ according to the mobility profiles of u and v, respectively. More formally, it is*

$$\mathbb{E}_{(u)}[M_d(p^{\mathbf{r}'}_{\mathbf{r},\tau,\tau'}(u), p^{\mathbf{r}''}_{\mathbf{r},\tau,\tau'}(v))], \tag{6.4}$$

where $p^{\mathbf{r}'}_{r,\tau,\tau'}(u)$ and $p^{\mathbf{r}''}_{r,\tau,\tau'}(v)$ denote the conditional probability distributions of the next location, given the current location and the current and next time periods. The Mallows distance is computed over random variables $\mathbf{r}'$ and $\mathbf{r}''$, and the expectation is computed over random variable $\mathbf{r}$ and time periods τ and τ'.

We define the geographic similarity *between mobility patterns of u and v as*

$$\mathsf{sim}_{\mathsf{G}}(u,v) = 1 - \frac{\mathbb{E}[M_d(p^{\mathbf{r}'}_{\mathbf{r},\tau,\tau'}(u), p^{\mathbf{r}''}_{\mathbf{r},\tau,\tau'}(v))]}{z_g}, \tag{6.5}$$

where z_g is a normalization constant equal to the maximum value of (the expectation of) the Mallows distance given $d(\cdot)$, ensuring that the geographic similarity always is in the $[0,1]$ range.

We compute the geographic *dissimilarity* using the law of total expectation. This also clarifies its meaning by showing more directly the role of the random variables.

$$\mathbb{E}[M_d(p^{\mathbf{r}'}_{\mathbf{r},\tau,\tau'}(u), p^{\mathbf{r}''}_{\mathbf{r},\tau,\tau'}(v))] = \sum_{r,\tau,\tau'} M_d(p^{\mathbf{r}'}_{r,\tau,\tau'}(u), p^{\mathbf{r}''}_{r,\tau,\tau'}(v)) \cdot p^{r,\tau,\tau'}(u). \tag{6.6}$$

This is simply the average, for each time and location, of the EMD between the distributions of the next location of u and v. So, for each current location (and time), we use the

EMD to compute the dissimilarity between the distributions representing the next locations of users u and v, respectively. The current location is taken according to user u's mobility profile, making this metric asymmetric.

For a particular distance function $d(\cdot)$, the definition can be expanded and previous expressions can be further simplified. This is the case for $d(i, j) = \mathbb{1}_{i \neq j}$, for which $M_d(p, q)$, for arbitrary probability distributions p and q, has closed form $1 - \sum_i \min\{p_i, q_i\}$.

Using the dissimilarity metric, we can compute the *geographic similarity* between the mobility profiles $\langle p(u), \pi(u)\rangle$ and $\langle p(v), \pi(v)\rangle$, for any distance function (e.g., hamming distance, Euclidean distance). For example, considering hamming distance $d(r, r') = \mathbb{1}_{r \neq r'}$, the geographic similarity is:

$$\sum_{r,r',\tau,\tau'} p^{\tau'}_{r,\tau}(u) \pi^{r,\tau}(u) \min\{p^{r'}_{r,\tau,\tau'}(u), p^{r'}_{r,\tau,\tau'}(v)\}. \tag{6.7}$$

We emphasize that this definition leads to an asymmetrical similarity measure, i.e., the similarity of u to v need not be the same as the similarity of v to u. In principle, this metric can also be computed using measures other than EMD. For example, one can use Kullback-Leibler divergence measure [45] to compute the difference between two probability distributions, ignoring the distance between the locations. We emphasize that we use EMD, in our geographic similarity metric, as we also want to include the distance function $d(\cdot)$ between locations in computing the difference between two mobility models.

Consider now the computation of the geographic similarity. For the case, $d(r, r') = \mathbb{1}_{r \neq r'}$, the computation according to closed-form of Eq. (6.7) takes $O(T^2 \cdot |R|^2)$ operations, where T is the number of time periods . For arbitrary $d(\cdot)$ with no closed-form expressions, the geographic similarity is obtained through $T^2 \cdot |R|$ EMD computations. Each of these EMD computations involves minimizing the Mallows distance, that is equivalent to solving the linear program given by Eq. (6.2).

Semantic Similarity. The semantic similarity metric builds upon the basic assumption that for two individuals u and v there exists an (unknown) semantics mapping σ of locations R onto itself (i.e. a permutation) such that R for u, and $\sigma^{-1}(R)$ for v semantically match. It is important to note that assuming such a mapping does not commit us to trying to learn it based on modeling location semantics directly. Instead, we define the *hidden* semantic similarity between u and v as the maximum geographic similarity taken over all possible mappings σ. We define semantic similarity metric as follows.

Definition 6.3. *Let σ be the mapping of locations of u to locations of v. Let $\mathbf{r}$, $\mathbf{r}'$, and $\mathbf{r}''$ be random variables for locations, and τ and τ' be two time periods. We define the semantic* dissimilarity *between u and v for moving in the sequence of time periods $\{\tau, \tau'\}$ as*

$$\mathcal{D}_u^v(\{\tau, \tau'\}) = \min_{\sigma} \mathbb{E}\left[M_d(p^{\mathbf{r}'}_{\mathbf{r},\tau,\tau'}(u), p^{\sigma(\mathbf{r}'')}_{\sigma(\mathbf{r}),\tau,\tau'}(v)) \right], \tag{6.8}$$

where the Mallows distance $M_d(\cdot)$ is computed over the random variable $\mathbf{r}'$ and the expectation is computed over the random variable $\mathbf{r}$ given time periods τ and τ'.

Now, we define the semantic similarity *between u and v over any sequences of time periods as*

$$\mathsf{sim}_\mathsf{S}(u, v) = 1 - \frac{\mathbb{E}\left[\mathcal{D}_u^v(\{\tau, \tau'\})\right]}{z_s}, \tag{6.9}$$

where z_s is a normalization constant equal to the maximum value of (the expectation of) the Mallows distance given $d(\cdot)$, ensuring that the semantic similarity always is in the $[0, 1]$ *range.*

What we compute in Eq. (6.8) is the minimum geographic mobility dissimilarity between u and v where the locations of v are relabeled and mapped to locations of u according to the permutation function σ_u^v (which is the σ that minimizes 6.8). Consider two individuals u and v are at $\mathbf{r}$ and $\sigma(\mathbf{r})$, respectively, at time period τ. The Mallows distance M_d computes how dissimilar their movement will be to the next location which are represented with random variables $\mathbf{r}'$ for u and $\sigma(\mathbf{r}'')$ for v. If, according to a mapping, the way that they move between these locations is similar, they behave similarly with respect to those locations. If this is true across different time periods and location pairs, their mobilities are similar. So, the semantic similarity between two individuals is determined by σ_u^v.

We compute this metric at two different levels of accuracy of the mobility model. If we only consider the visiting probability π part of each individual's mobility profile, we compute sim_S as follows: Let us consider the hamming distance function $d(r, r') = 1_{r \neq \sigma^{-1}(r')}$. In this case, we can compute the semantic similarity metric as

$$1 - \sum_{\tau} \Pr\{\tau\} \max_{\sigma} \sum_{r} \min\{\pi_\tau^r(u), \pi_\tau^{\sigma(r)}(v)\}. \tag{6.10}$$

Note that the computation of Eq. (6.10) requires finding the mapping σ which maximizes the inner term for each time period τ. Since there are $|R|!$ possible candidates for the maximum mapping σ, a brute-force approach is inefficient. However, the problem's structure resembles that of a linear assignment. Focusing on the inner sum, we see that each term (each r) can be associated with $|R|$ values of $\sigma(r)$ independently of the other components of σ. To recast the problem as a linear assignment, we construct a bipartite graph where the nodes represent R and $\sigma(R)$, and each edge represents the association (through σ) of r

with $\sigma(r)$. The maximum weight assignment of the constructed bipartite graph gives the permutation σ. The running time of this procedure is $O(T \cdot |R|^3)$ using the Hungarian algorithm [46].

In the case where we consider the more accurate mobility profile $\langle p, \pi \rangle$, it can be computed as follows:

$$1 - \sum_{\tau,\tau'} \max_{\sigma} \sum_{r,r'} \pi^{r,\tau}(u) p^{\tau'}_{r,\tau}(u) \min\{p^{r'}_{r,\tau,\tau'}(u), p^{\sigma(r')}_{\sigma(r),\tau,\tau'}(u)\} \ . \tag{6.11}$$

It is not known whether there is an efficient algorithm to compute the semantic similarity according to Eq. (6.11). The difficulty comes from having to consider assignments of pairs: (r, r') to $(\sigma(r), \sigma(r'))$, which makes this computation look similar to the Quadratic Assignment Problem (QAP) [47], known to be NP-Hard and APX-Hard. But, Eq. (6.11) can be approximated using the Metropolis-Hastings algorithm [48] or Simulated Annealing [49].

6.2 A BAYESIAN NETWORKS MODEL

In this section, we construct a seedbased generative model to capture statistical relationships between attributes of a record. This model, which was proposed in [32], uses techniques inspired from Bayesian networks. We evaluate it with a census dataset in Chapter 8.

We discuss training the model on a training set (disjoint from the input data for synthesis) and achieving differential privacy for the model itself (with respect to its training set). Specifically, the training set is composed of two disjoint subsets D_T used for structure learning, and D_P used for parameters learning.

6.2.1 Model

We consider a universe of records with m attributes, and we let $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_m\}$ be the set of random variables associated with the attributes of the data records in U. Let G be a directed acyclic graph (DAG), where the nodes are the random variables, and the edges represent the probabilistic dependency between them. A directed edge from $\mathbf{x}_j$ to $\mathbf{x}_i$ indicates the probabilistic dependence of attribute i to attribute j. Let $\mathrm{par}_G(i)$ be the set of parents of random variable i according to the dependency graph G. The following model represents the joint probability of data attributes.

$$\Pr\{\mathbf{x}_1, ..., \mathbf{x}_m\} = \prod_{i=1}^{m} \Pr\{\mathbf{x}_i \,|\, \{\mathbf{x}_j\}_{\forall j \in \mathrm{par}_G(i)}\} \tag{6.12}$$

This model is based on a structure between random variables, captured by the DAG G, and a set of parameters that defines the conditional probabilities. In Section 6.2.3 and Section 6.2.4, we present differentially private algorithms to learn the structure and parameters of the model, respectively.

6.2.2 Synthesis

We probabilistically transform a real data record (the seed) into a synthetic data record, by updating its attributes. Let $\{x_1, x_2, ..., x_m\}$ be the values for the set of data attributes for a randomly selected record in the input dataset D. Let ω be the number of attributes for which we generate new values. Thus, we keep (i.e., copy over) the values of $m-\omega$ attributes from the seed to the synthetic data. Let σ be a permutation over $\{1, 2, ..., m\}$ to determine the re-sampling order of attributes.

We set the re-sampling order σ to be the dependency order between random variables. More precisely, $\forall j \in \text{par}_G(i)$: $\sigma(j) < \sigma(i)$. We fix the values of the first $m - \omega$ attributes according to σ (i.e., the synthetic record and the seed overlap on their $\{\sigma(1), ..., \sigma(m-\omega)\}$ attributes). We then generate a new value for each of the remaining ω attributes, using the conditional probabilities Eq. (6.12). As we update the record while we re-sample, each new value can depend on attributes with updated values as well as the ones with original (seed) values.

Specifically, we re-sample attribute $\sigma(i)$, for $i > m - \omega$, as

$$\begin{aligned} x'_{\sigma(i)} \sim \Pr\{\mathbf{x}_{\sigma(i)} \,|\, &\{\mathbf{x}_{\sigma(j)} = x_{\sigma(j)}\}_{\forall j \in \text{par}_G(i), j \le m-\omega}, \\ &\{\mathbf{x}_{\sigma(j)} = x'_{\sigma(j)}\}_{\forall j \in \text{par}_G(i), j > m-\omega}\} \end{aligned} \tag{6.13}$$

Seedless synthesis. Observe that when $\omega = m$, no attributes are copied from the seed and thus the re-sampling of all m attributes does not depend on the seed. As a result, this limiting case is an example of seedless synthesis. More generally, the smaller ω is the higher the dependence on the seed. In particular, if $\omega = 0$, no attributes are re-sampled and the generative model is equivalent to the identity model (Section 3.1.1).

Marginal synthesis. As a baseline, we consider a (seedless) generative model that (independently from any seed record) samples a value for an attribute from its marginal distribution. That is, for all attributes i, we generate $x_i \sim \Pr\{\mathbf{x}_i\}$. This is based on an assumption of independence between attributes: $\Pr\{\mathbf{x}_1, ..., \mathbf{x}_m\} = \prod_{i=1}^{m} \Pr\{\mathbf{x}_i\}$.

6.2.3 Privacy-Preserving Structure Learning

The generative model depends on the dependency structure between random variables that represent data attributes which is captured by the DAG G. We show how to learn G from real (training) data. We also explain how to do this in a privacy-preserving way.

The algorithm is based on maximizing a scoring function that reflects how correlated the attributes are according to the training data. There are multiple approaches to this problem in the literature [50]. We propose to use a method based on a well-studied machine learning problem: *feature selection*. For each attribute, we want to find the best set of features (among all attributes) to predict it, and add them as the attribute's parents, under the condition that the dependency graph remains acyclic.

The machine learning literature proposes different ways to rank features by their predictive power over a particular attribute. For example, we could calculate the information gain of each feature with the target attribute, but this approach ignores the redundant information between features. Instead, we propose to use Correlation-based Feature Selection (CFS) [51] which determines the best subset of predictive features according to some correlation measure. This is an optimization problem: select a subset of features that have high correlation with the target attribute and simultaneously low correlation among themselves. The task is to find the best subset of features which maximizes a merit score that captures our objective.

We follow [51] to compute the merit score for a parent set $\mathrm{par}_G(i)$ for attribute i as

$$\mathrm{score}(\mathrm{par}_G(i)) = \frac{\sum_{j \in \mathrm{par}_G(i)} \mathrm{corr}(\mathbf{x}_i, \mathbf{x}_j)}{\sqrt{|\mathrm{par}_G(i)| + \sum_{j,k \in \mathrm{par}_G(i)} \mathrm{corr}(\mathbf{x}_j, \mathbf{x}_k)}}, \tag{6.14}$$

where $|\mathrm{par}_G(i)|$ is the size of the parent set, and $\mathrm{corr}()$ is the correlation between two random variables associated with two attributes. The numerator rewards correlation between parent attributes and the target attribute, and the denominator penalizes the inner-correlation among parent attributes. The suggested correlation metric in [51], which we use, is the symmetrical uncertainty coefficient:

$$\mathrm{corr}(\mathbf{x}_i, \mathbf{x}_j) = 2 - 2\frac{\mathrm{H}(\mathbf{x}_i, \mathbf{x}_j)}{\mathrm{H}(\mathbf{x}_i) + \mathrm{H}(\mathbf{x}_j)}, \tag{6.15}$$

where $\mathrm{H}()$ is the entropy function.

The optimization objective in constructing G is to maximize the total $\mathrm{score}(\mathrm{par}_G(i))$ for all attributes i. Unfortunately, the number of possible solutions to search for is exponential in the number of attributes, making it impractical to find the optimal solution. The greedy algorithm, suggested in [51], is to start with an empty parent set for a target attribute and always add the attribute (feature) that maximizes the score.

There are two constraints in our optimization problem. First, the resulting dependency graph obtained from the set of best predictive features (i.e., parent attributes) for all attributes should be acyclic. This would allow us to decompose and compute the joint distribution over attributes as represented in Eq. (6.12).

Second, we enforce a maximum allowable complexity cost for the set of parents for each attribute. The cost is proportional to the number of possible joint value assignments (configurations) for the parent attributes. So, for each attribute i, the constraint is

$$\mathrm{cost}(\mathrm{par}_G(i)) = \prod_{j \in \mathrm{par}_G(i)} |\mathbf{x}_j| \leq \mathtt{maxcost} \tag{6.16}$$

where $|\mathbf{x}_j|$ is the total number of possible values that the attribute j takes. This constraint prevents selecting too many parent attribute combinations. The larger the joint cardinality of attribute i's parents is, the fewer data points to estimate the conditional probability $\Pr\{\mathbf{x}_i \,|\, \{\mathbf{x}_j\}_{\forall j \in \mathrm{par}_G(i)}\}$ can be found. This would cause overfitting the conditional probabilities on the data, that results in low confidence parameter estimation in Section 6.2.4.

To compute the score and cost functions, we discretize the parent attributes. Let bkt() be a discretizing function that partitions an attribute's values into buckets. If the attribute is continuous, it becomes discrete, and if it is already discrete, bkt() might reduce the number of its bins. Thus, we update conditional probabilities as follows:

$$\Pr\{\mathbf{x}_i \,|\, \{\mathbf{x}_j\}_{\forall_{j \in \mathrm{par}_G(i)}}\} \approx \Pr\{\mathbf{x}_i \,|\, \{\mathrm{bkt}(\mathbf{x}_j)\}_{\forall_{j \in \mathrm{par}_G(i)}}\} \;, \tag{6.17}$$

where the discretization varies for each attribute. We update Eq. (6.14) and Eq. (6.16) according to Eq. (6.17). This approximation itself decreases the cost complexity of a parent set, and further prevents overfitting on the data.

Differential privacy.

We show how to safeguard the privacy of individuals whose records are in the training data, and could influence the model structure (which might leak about their data).

All computations to learn the DAG depend on computing the correlation metric Eq. (6.15). Thus, we can achieve differential privacy by simply adding appropriate noise to the metric. Observe that the correlation metric is based on the entropy of a single or a pair of random variables. Thus, we only need to compute the entropy functions in a differentially-private way and ensure that the result remains in the $[0, 1]$ range.

Let $\tilde{H}(\mathbf{z})$ be the noisy version of the entropy of a random variable $\mathbf{z}$, where $\mathbf{z}$ could be a single or pair of random variables associated with the attributes and their discretized version (as presented in Eq. (6.17)). We need to compute the noisy entropy $\tilde{H}(\mathbf{x}_i)$, $\tilde{H}(\text{bkt}(\mathbf{x}_i))$, $\tilde{H}(\mathbf{x}_i, \mathbf{x}_j)$, and $\tilde{H}(\mathbf{x}_i, \text{bkt}(\mathbf{x}_j))$, for all attributes i and j. For each of these, we make use of the Laplace mechanism, i.e., we generate fresh noise drawn from the Laplace distribution calibrated to Δ_H, the sensitivity of the entropy function. That is:

$$\tilde{H}(\mathbf{z}) = \text{H}(\mathbf{z}) + \text{Lap}(\frac{\Delta_H}{\varepsilon_H}) \ . \tag{6.18}$$

where ε_H is the privacy budget for this step.

We show in Appendix A.2 that if $\mathbf{z}$ is a random variable with a probability distribution, estimated from $n_T = |D_T|$ data records, an upper bound for the entropy sensitivity is:

$$\Delta_H \leq \frac{1}{n_T}[2 + \frac{1}{\ln(2)} + 2\log_2 n_T] = O(\frac{\log_2 n_T}{n_T}) \ . \tag{6.19}$$

Remark that Δ_H is a function of n_T (the number of records in D_T) which also needs to be protected. So, we compute Δ_H with the Laplace mechanism, where instead of n_T we use:

$$\tilde{n}_T = n_T + \text{Lap}(\frac{1}{\varepsilon_{n_T}}) \ . \tag{6.20}$$

By randomizing the entropy values, according to Eq. (6.18), the dependency graph, which we denote as $\tilde{G}$, is obtained with differential privacy.

6.2.4 Privacy-Preserving Parameter Learning

Given the DAG $\tilde{G}$, we still need to compute the conditional probabilities for predicting each of the attributes given its parent set (see Eq. (6.12)). This is a well-known problem in statistics. In this section, we show how to learn the parameters that represent such conditional probabilities, from the training data D_P. We also show how this can be done with differential privacy.

The problem to be solved is to first learn a prior distribution over the parameters of the conditional probabilities. To do so, we learn the hyper-parameters (the parameters of the prior distribution over the model's parameters) from data. Only then, can we compute the parameters that form the conditional probabilities from the prior distribution.

Let us take the example of computing the parameters for predicting discrete/categorical attributes. In this case, we assume a multinomial distribution over the attribute's values (that fall into different bins). The conjugate prior for multinomials comes from a Dirichlet family. The Dirichlet distribution assigns probabilities to all possible multinomial distributions, according to the statistics obtained from a set of data records.

Let $|\mathbf{x}_i|$ be the number of distinct values that attribute i can take. The probability of some multinomial distribution parameters $p_i^c = p_{i,1}^c, p_{i,2}^c, ..., p_{i,|\mathbf{x}_i|}^c$ to predict attribute i, under configuration c for $\text{par}_G(i)$, is

$$\Pr\{p_i^c \,|\, \tilde{G}, D_P\} = \text{Dir}(\alpha_i^c + n_i^c) \ . \tag{6.21}$$

where α_i^c is the vector of default hyper-parameters for the Dirichlet distribution, and n_i^c is the vector for the number of data records in D_P with $\text{par}_{\tilde{G}}(i)$ configuration c with different values for attribute i (i.e., element $n_{i,l}^c$ is the number of records for which $x_i = l$ and $\text{par}_{\tilde{G}}(i)$ configuration is c). The Dirichlet distribution is computed as

$$\text{Dir}(\alpha_i^c + n_i^c) = \Gamma(\alpha_i^c + n_i^c) \prod_{l=1}^{|\mathbf{x}_i|} \frac{(p_{i,l}^c)^{\alpha_{i,l}^c + n_{i,l}^c - 1}}{\Gamma(\alpha_{i,l}^c + n_{i,l}^c)} \ , \tag{6.22}$$

where $\alpha_i^c = \sum_l \alpha_{i,l}^c$, and $n_i^c = \sum_l n_{i,l}^c$. The number of configurations $\#c$ is $\prod_{j \in \text{par}_{\tilde{G}}(i)} |\text{bkt}(\mathbf{x_j})|$, which according to constraint Eq. (6.16) can at most be `maxcost`.

Learning the parameters of the model, in the case of a Dirichlet prior for the multinomial distribution, is simply computing n_i^c from the data records in D_P. Given the probability distribution Eq. (6.21) over the multinomial parameters, we can compute the most likely set of parameters as

$$p_{i,l}^c = \frac{\alpha_{i,l}^c + n_{i,l}^c}{\alpha_i^c + n_i^c} \ , \tag{6.23}$$

or we can sample a set of multinomial parameters according to Eq. (6.22). This increases the variety of data samples that we can generate.

Note that for computing the marginal distributions (needed for the baseline) we perform the same computations but simply set the parent sets to be empty.

Differential privacy.

The parameters of the conditional probabilities depend on the data records in D_P, thus they can leak sensitive information about individuals who contributed to the real dataset. We show how to learn parameters of the attribute conditional probabilities (i.e., p_i^c values) in a differentially private way for the discrete case. Observe that in Eq. (6.21), the only computations that are dependent on D_P are the n_i^c counts (for all c and i). To find the variance of the noise to be added to these counts, to achieve differential privacy, we need to compute their sensitivity.

Suppose we are computing the parameters associated with predicting a given attribute i given its parent set $\mathrm{par}_{\tilde{G}}(i)$. Observe that adding a record to D_P increases exactly a single component $n^c_{i,l}$, for which it matches value l for attribute i and configuration c for its parent set. So, only one single element among all $\#c \times |\mathbf{x}_i|$ elements of $n_i = n^1_i, n^2_i, ..., n^{\#c}_i$ changes. This implies that the L1 sensitivity of n_i is 1. Consequently, random noise drawn from $\mathrm{Lap}(\frac{1}{\varepsilon_p})$ can be added to each component of n_i independently. More precisely, for any attribute i, value l, and configuration c, we randomize counts as

$$\tilde{n}^c_{i,l} = \max(0, n^c_{i,l} + \mathrm{Lap}(\frac{1}{\varepsilon_p})) \tag{6.24}$$

and use them to compute Eq. (6.21).

6.2.5 Differential Privacy Analysis

In this section, we compute the differential privacy guarantee we obtained for the entire training process. This includes learning the structure and the parameters of the model. We compute the overall ε and δ by composing the differentially-private mechanisms in Section 6.2.3 and Section 6.2.4.

The $m(m+1)$ entropy values, $\tilde{H}(\mathbf{z})$, needed for structure learning are obtained in an a way that satisfies ε_H-differential privacy. This is also the case for the number of records n_T, i.e., it satisfies ε_{n_T}-differential privacy. Similarly, the counts $n^c_{i,l}$ parameters learned for each configuration satisfy ε_p-differential privacy.

For structure learning, we use advanced composition for the $m(m+1)$ entropy values and sequential composition with the number of records. So that the overall privacy achieved is $(\varepsilon_L, \delta_L)$-differential privacy for a fixed $\delta_L \ll \frac{1}{n_T}$ and $\varepsilon_L = \varepsilon_{n_T} + \varepsilon_H\sqrt{2m(m+1)\ln(\delta_L^{-1})} + m(m+1)\varepsilon_H(e^{\varepsilon_H} - 1)$.

For the parameter learning the privacy achieved is $(\varepsilon_P, \delta_P)$-differential privacy using advanced composition over the m attributes. Here, $\delta_P \ll \frac{1}{n_p}$ (where n_p is the number of records in D_P) and $\varepsilon_P = \varepsilon_p\sqrt{2m\ln(\delta_P^{-1})} + m\varepsilon_p(e^{\varepsilon_p} - 1)$.

Given that D_T and D_P are non-overlapping, the guarantee obtained for the generative model is differential privacy with parameters $(\max\{\varepsilon_L, \varepsilon_P\}, \max\{\delta_L, \delta_P\})$.

6.3 LATENT-SPACE SEEDBASED SYNTHESIS

We characterize a class of models that we call *latent-space models*. We evaluate instances of latent-space models using medical data and images in Chapter 9 and Chapter 10, respectively.

Latent-Space Models. A *latent-space model* is a pair of two deterministic algorithms Encode, Decode that operate on elements of some data universe U and map it to and from $\mathbb{R}^w$ for some positive integer w. We typically think of the latent space $\mathbb{R}^w$ as much smaller than U in the sense that $w \ll \log |U|$. Concretely:

- The *encoder* transforms a data record $d \in U$ (back) into its latent-space representation, point or code, $z = \mathsf{Encode}(d)$, with $z \in \mathbb{R}^w$.
- The *decoder* takes a latent-space point $z \in \mathbb{R}^w$ and transforms it into a data record $y \in U$ such that $y = \mathsf{Decode}(z)$.

Starting from a record $d \in U$ and encoding it to obtain $z = \mathsf{Encode}(d)$, we can feed z into the decoder to obtain a *reconstructed* record $\tilde{d} = \mathsf{Decode}(z)$. Remark that there is no guarantee that $d = \tilde{d}$. Ideally, however, models are accurate in the sense that $\tilde{d}$ is likely to be similar to d for all $d \in U$.

Note that because Encode and Decode are deterministic functions, the pair always produces the same output given the same input. Consequently, encoding a data record d and producing a synthetic as $y = \mathsf{Decode}(d)$ is a degenerate case of a generative model.

As an alternative, we propose two techniques to construct a probabilistic seedbased generative model from a latent-space model.

6.3.1 Latent-Space Noise Adding

A generic technique is to add noise to the latent space.

Definition 6.4 (Noise-Adding Latent-Space Generative Model)**.**
Input: data record (seed) d, noise distribution Z (over $\mathbb{R}^w$).
Output: synthetic $y \in U$.

1. *Compute the code of the seed $z = \mathsf{Encode}(d)$.*
2. *Sample noise vector $\zeta \sim Z$.*
3. *Decode $y = \mathsf{Decode}(z + \zeta)$.*
4. *Output y.*

For example, the noise distribution may be independent Gaussians with mean zero and variance β, i.e., $Z = \mathrm{Gaus}(\mathbf{0}, \beta\mathbf{I})$.

To use this technique within our framework (Section 3.1), it is necessary to compute the synthesis probabilities $\Pr\{y \leftarrow \mathcal{M}(d)\}$. This is a challenge because even if the noise distribution is known, we may not know the pre-image(s) of some y through Decode.

To overcome this obstacle, we propose to apply the framework directly on the latent space. Given input dataset D, we first use Encode on each record to obtain a dataset of codes. Then we simply apply Mechanism 3.1 with the dataset of codes as input, with a generative model that simply adds noise from Z to its input. Synthetics that pass the privacy test are points (in $\mathbb{R}^w$) which we then reconstruct into records in the original data universe using Decode.[1]

In this case, the generative model operates over the latent space $\mathbb{R}^w$ directly, so the synthesis probabilities, which only depend on the noise distribution Z, can be computed easily using the probability density/mass function of Z.

Additional latent-space processing. A salient feature of this technique is that because it occurs entirely over the latent space, any additional operations on the produced synthetics (e.g., in the latent space) are considered post-processing operations from the point-of-view of differential privacy. That is, we can manipulate a synthetic $z \in \mathbb{R}^w$ arbitrarily (including making copies) before decoding it without compromising the privacy guarantee obtained. We call this technique *additional latent-space processing* and illustrate it experimentally in Chapter 10.

6.3.2 Probabilistic Projection

Some latent-space models produce outputs that do not lie strictly in the data universe, i.e., for $d \in U$, it is possible that $\tilde{d} = \mathsf{Decode}(\mathsf{Encode}(d)) \notin U$. For example, this may occur when the input data records are binary strings (which can be interpreted as a sequence of $\{0, 1\}$ values) but where $\tilde{d}$ is a sequence of real numbers either each close to 0 or 1. Such occurrences are often dealt with by forcing the elements of $\tilde{d}$ to be either 0 or 1 (e.g., through rounding) as part of the decoding process.

An alternative to deterministic rounding, which we call *probabilistic projection*, is to sample a (synthetic) data record $y \in U$ from $\tilde{d}$ using some probabilistic function. This is a seedbased generative model (Section 3.1.1) where the seed is $\tilde{d}$ and we can use the framework on reconstructed seeds.

[1]Since Decode does not depend on D, it is a post-processing operation so that it does not compromise the privacy guarantee obtained.

Example 6.1. Let U be the set of all medical diagnosis codes of some kind (e.g., ICD-9) so that each code is a binary variable coding whether a patient has been diagnosed with the disease. A patient medical record $d \in U$ is a sequence of $\{0, 1\}$ values where if the i^{th} value is 1, the patient suffers from the i^{th} disease.

Suppose we want to synthesize medical records of this type using a latent-space model based on a neural network. Further suppose that the decoder produces outputs as sequences of real numbers $r_i \in [0, 1]$. To obtain $y \in U$, we could therefore simply set y_i, the i^{th} value of y, to $y_i = \mathbb{1}_{r_i > 0.5}$. This is a deterministic projection.

Alternatively, we may interpret r_i as the probability that the (synthetic) patient suffers from the i^{th} disease. To obtain $y \in U$, we simply sample y_i as a Bernoulli random variable with parameter r_i, i.e., $y_i \sim \text{Bern}(r_i)$. This exact probabilistic projection technique is evaluated on real-world medical records in Chapter 9.

6.3.3 Seedless

As a seedless baseline, we make use of the following technique. To produce synthetic $y \in U$: sample some $z \in \mathbb{R}^w$ and set $y = \mathsf{Decode}(z)$. Given that many practical models impose a Gaussian prior on the latent-space distribution, we propose to draw z from $\text{Gaus}(\mathbf{0}, \mathbf{I})$.

6.3.4 Examples

Many prominent techniques that were designed for purposes other than generating synthetic data, are latent-space models compatible with the framework.

PCA. Principal Component Analysis is a well-established technique used in a variety of applications [52–54]. In particular, it can be used as a dimensionality reduction technique: a dataset, expressed as a set of records composed of m real attributes, is summarized with respect to its w principal components (for some positive integer $w < m$).

Encoding and decoding of records with m attributes is performed using the $m \times w$ transformation matrix $\mathbf{W}$, which is obtained through "training" as follows. Given a dataset D of n records each with m attributes, we center its $n \times m$ data matrix $\mathbf{X}$ by subtracting its mean $\bar{\mathbf{X}}$. Columns of the transformation matrix $\mathbf{W}$ are the w eigenvectors associated with the w largest eigenvalues of the $m \times m$ covariance matrix $(\mathbf{X} - \bar{\mathbf{X}})^T(\mathbf{X} - \bar{\mathbf{X}})$.

Given a record $d \in U$ represented as $1 \times m$ row vector $\mathbf{x_d}$, we compute its latent-space representation as: $\mathbf{z_d} = (\mathbf{x_d} - \bar{\mathbf{X}})\mathbf{W}$. We can reconstruct or decode the $1 \times w$ row vector $\mathbf{z}$ as: $\tilde{\mathbf{x}} = \mathbf{z}\mathbf{W}^T + \bar{\mathbf{X}}$.

It should be noted that PCA has recently been proposed as a technique to produce synthetic data by Chanyaswad et al. [55]; their proposal is seedless.

Auto-Encoders. Neural-network based auto-encoders are composed of an encoder network and a decoder network trained simultaneously. The input and output layers have m neurons for m-dimensional input data. In contrast, the encoder part has w neurons in the output layer which coincides with the input layer of the decoder part.

The network is trained by feeding in data records and iteratively updating the weights so as to minimize the reconstruction error according to some loss function. For example, if the input data records attributes are in $[0,1]$, one may use the Bernoulli negative log-likelihood as loss function. That is, if $x_j \in \{0,1\}$ denotes the j^{th} input and $\tilde{x_j} \in [0,1]$ denotes the reconstructed output, then the loss is:

$$-\sum_{i=1}^{m} [x_i \log \tilde{x_i} + (1-x_i)\log(1-\tilde{x_i})] \ .$$

We add a regularization term to the loss function to constrain the latent-space distribution. For example, if the mean and standard deviation of a mini-batch are μ and σ, respectively, we add the following term to the loss function: $C \cdot \mathrm{KL}(\mathrm{Gaus}(\mu,\sigma), \mathrm{Gaus}(\mathbf{0},\mathbf{I}))$ for some constant $C > 0$. This has the effect of imposing a Gaussian prior on the latent-space distribution.

Auto-Encoders GANs. A popular way to train neural networks is to use the adversarial training paradigm. Train a generative network simultaneously with a distinguisher network: the former's job is to produce realistic samples whereas the latter is to distinguish generated samples from training samples. This is called a Generative Adversarial Network (GAN).

Typically, the generator is given random noise as input which makes the technique seedless. However, GANs can be combined with auto-encoders [26] by collapsing the generator and the decoder into a single network. This results in an architecture suitable for seedbased synthesis. We evaluate an AEGAN model, proposed in [26], in Chapter 10.

Chapter 7: Application: Location Data

In this chapter we evaluate the generative model presented in Section 6.1 using a real-world dataset of fine-grained location trajectories. We produce synthetic trajectories and evaluate their quality in two scenarios: sharing locations with Location-Based Services (LBS), and releasing synthetic location datasets. This evaluation study originally appeared in [33].

Note that we sometimes refer to the synthetic trajectories as *fakes* because this term is often used in the location privacy literature.

7.1 DATASET

The dataset we use for the evaluation is collected through the Nokia Lausanne Data Collection Campaign (see [56]). We pre-process the data in two phases, filling gaps in the traces and discretizing the time and location.

The raw dataset contains events of three types: GPS coordinates, WLAN and GSM identifiers. We construct valid traces (out of partial traces) by filling gaps. We interpolate along the path of consecutive GPS points, using the WLAN and GSM information.

We then extract two days of traces for each user, such that each trace (of one day) contains a sequence of 72 locations, with one location reported every 20 minutes. Some locations are visited very rarely only by very few users. So, we reduce the number of locations from 1491 to 400 by clustering close-by locations together. We use a hierarchical clustering algorithm for this purpose, and place the locations that are geographically close or have very few visits in one cluster. The geographical distribution of visits of all users over the locations in the considered area is shown in Fig. 7.1-(left).

7.2 SETUP

From all traces, we then sub-sample 30 user traces. The 1st day of traces for these users is used as input dataset D_S, whereas the 2nd day of traces will be used as baseline (testing set) during the evaluation. Using the input dataset, we compute the mobility profiles of all 30 users, and then the semantic location graph by calculating a similarity score for each pair of locations, averaged across all users. After clustering this semantic location graph, we obtain 20 location clusters. We choose this number of clusters as it provides optimal clustering i.e., it maximizes the ratio of inter-cluster similarity over intra-cluster similarity. This clustering is illustrated in Fig. 7.1-(right), where each location is drawn with the color of the cluster it

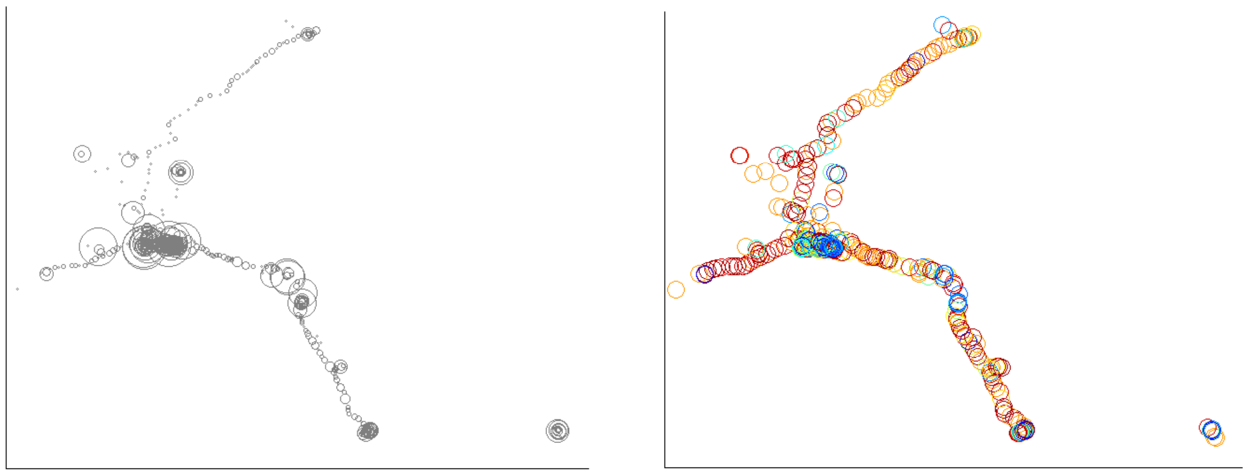


Figure 7.1: 400 locations visited around Lausanne and nearby towns by the 30 users. Some users commute between two towns whereas the majority of them live and work in the same city of Lausanne (the area with higher concentration).

belongs to. The figure allows us to distinguish some patterns, for example locations at the center of cities are mostly in blue, while many locations representing roads and highways are colored in red. Also notice that the semantic clustering does not seem to depend on the geographical distance of locations.

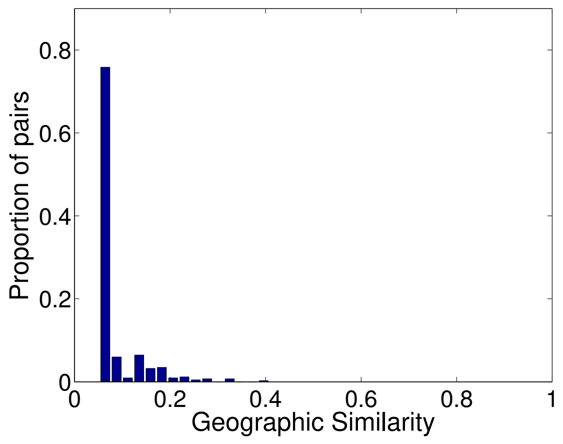
To illustrate our geographic and semantic similarity metrics, we compute those metrics pairwise over all 30 users.[1] The result is shown in Fig. 7.2. The first histogram shows that the 30 users are not strongly geographically similar to each other, except for a few pairs of users. This is expected given the range of locations they explore overall, as seen in Fig. 7.1-(left). On the other hand, the distribution of the semantic similarity across all distinct pairs of users has a larger variance, and a large number of users are highly similar.

Simulation setup. Recall that from the input dataset, we sub-sampled 30 user traces (day 1) that we use for the input seed dataset D_S. As for the parameters of the Synthesizer() algorithm (Section 6.1.1), we set $p_c = 0.25$, $p_m = 0.75$, $p_l = 1.0$, and $p_v = 4$.

For each of the 30 input (seed) traces, we generated about 500 synthetic traces. We then select and use these traces according to the scenario evaluated. For example, for the LBS scenario, we sampled traces (for each user) according to the synthetic traces likelihoods.

Evaluation metrics. We evaluate the use of synthetic traces in two popular scenarios: (i) using fake locations along with real locations when accessing location-based services (Section 7.4), and (ii) releasing synthetic location datasets to be used for various geo-data analysis tasks (Section 7.3). There are some differences in terms of the adversary model

[1]We exclude the geographic/semantic similarity of a user with herself because it is 1.0.



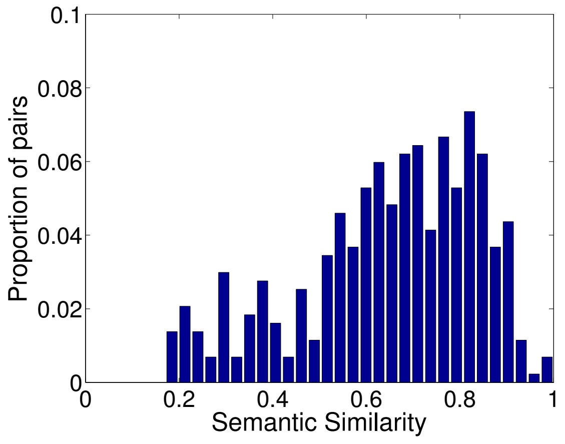


Figure 7.2: Normalized histogram of the geographic similarity and semantic similarity between all distinct pairs of 30 users in the input dataset. Mobility models of different individuals is geographically very specific to themselves, i.e., they are unique. This is well reflected in the skewed distribution of geographic similarity towards very small values. As hypothesized in Section 6.1, most of individuals have high semantic similarities between their mobility models.

between different scenarios. Therefore, there are additional considerations regarding the privacy of users in location-based services, e.g., their privacy against inference attacks, that we discuss in the corresponding section. The utility metric is also dependent on the scenario and is measured differently in each case.

7.3 EVALUATION: SYNTHETIC DATA RELEASE

In this scenario, the synthetic traces form a location dataset that is meant to be used for various geo-data analysis tasks *in place of* real location traces.

Setup. We generate a large number of synthetic traces out of which we ultimately select: 10 datasets each containing 30 traces. This is done in order to have each synthetic dataset of the same size and format as the input (seed) dataset.

Utility. Because we release a set of synthetic traces to be used instead of a real location dataset, to evaluate utility we must take into account how the released traces are to be used. Specifically, we must determine to what extent the key features and statistics, which are relevant for the considered applications, are preserved. Clearly, we cannot expect all statistics (of real traces) to be preserved in the synthetic dataset. An example of property that we do not preserve is the relationship in the mobility of input traces, e.g., individuals commuting to work together. Indeed, if two individuals carpool to work, the corresponding synthetic traces will not exhibit analogous co-traveling behavior (since synthetic traces are generated independently). However, we their common semantic features should be preserved.

From the literature, we identify the following prominent geo-data analysis tasks.

(1) Points of Interests (PoIs) extraction. The goal is to discover locations that are frequently visited and are prominently of interest to the public. PoIs can be used to provide travel recommendations. In particular, [57] proposes techniques to mine the top n interesting locations in a given region. A key feature to preserve is the distribution of visits among locations, specifically the most visited (i.e., popular) locations.

(2) Semantic annotation / labeling of locations. The goal is to automatically assign labels to locations according to their semantics (e.g., restaurant, bar, shopping mall). For example, [58] proposes an SVM classifier to assign multiple labels to location-based social network check-ins. In contrast, [59] proposes to do automatic labeling of locations into 10 semantic categories using smartphone recorded GPS, WiFi, and cell-tower data. In all cases, the distribution of visitors (and unique visitors) per location are key features of the input data. In addition, [58, 59] use users' temporal behavioral data, such as the amount of time a user spends in a location.

(3) Map inference. [60] evaluates the two main approaches to infer road maps from a large scale GPS traces: using the sample coordinates themselves, or using the transitions between samples. A related task is the discovery of semantic regions in a city [61]. In both cases, key features of the input data include the distribution of visited locations, and transitions, particularly the popular ones.

(4) Modeling human mobility. [62] proposes to learn a multi-layer spatial density model from social network check-ins. In this case, temporal features of location data are largely overlooked. Rather the focus is on features such as the spatial location distribution in aggregate and at individual level.

(5) Determining optimal locations for retail stores. The goal is to find ideal geographic placement for a retail store, or a new business. In particular, [63] proposes to mine online location-based services to evaluate the retail quality of a geographic area. Specifically, the focus is on a combination of mobility features such as popularity of an area, and semantic features such as visits to semantically similar venues (e.g., coffee shops of the same franchise) or transitions between venues.

Based on the features that prominent geo-data analysis tasks require, we identify six statistics that need to be preserved to ensure that such tasks can reasonably be performed on a set of synthetic traces *instead of* a real dataset. As baseline, we use the value of the statistic on the testing (day 2) dataset, which consists of location traces of the same users as the input (day 1) dataset. When appropriate, we also use uniformly random location traces as a baseline.

(a) Distribution of the number of visits. Tasks such as (2) and (3) exploit the fact that some locations are more frequently visited than others. In fact, [59] explicitly mention "how often places are visited" as a major feature.

In order to evaluate this, we do the following. For each dataset, we compute the spatial allocation, i.e., for each location (from least to most popular, for that dataset), we calculate the number of visits spent in that location across all traces in the dataset. We then normalize this quantity to obtain a probability distribution over locations (sorted by popularity), i.e., for each location we have the probability of a random visit to that location. From these distributions, we compute the KL-divergence of the real (day 1) dataset to each of our synthetic datasets, and to a variety of baselines.[2] Some related work such as [64] uses the relative error of counting queries as a metric instead of the KL-divergence. Therefore, we additionally calculate the relative error by interpreting the number of visits to each location as the answer to a counting query. That is, if the number of visits to location x is n_1 for dataset 1 and n_2 for dataset 2, then we calculate the relative error as $\frac{|n_1 - n_2|}{\max(n_1, 0.001 \cdot N)}$, where N is the total number of visits to any location (the same for all the datasets) [64]. We report the average relative error over all locations. The results of both metrics are shown in Section 7.3. The results suggest that a lot of information is preserved in this case: while the error for the synthetic datasets is greater than that for the real (testing, day 2) dataset, the error is significantly lower than the other baselines.

Table 7.1: KL-divergence and relative error of the location visiting probabilities of the real (day 1) dataset against the 10 synthetic datasets, and various baselines. "Real" is the testing (day 2) portion of the real dataset (see Section 7.2). "Uniform" is the uniform distribution over all locations. "Single" is the distribution where all users always visit the same location.

	Real	Synthetic	Uniform	Single
KL-divergence	0.037	0.384 ± 0.043	1.191	4.666
Relative error [64]	0.144	0.370 ± 0.010	1.621	0.542

(b) Distribution of number of visits for top* 50 *locations. For most tasks, features of the most popular locations (i.e., the most frequently visited locations) are the most important ones to preserve. In particular, this is consistent with the results provided in [59] for automatic labeling.

To evaluate this, we use the same procedure as for (a), except that we only consider the top 50 locations, and plot the distribution instead of calculating the KL-divergence. Fig. 7.3

[2] We set zero probabilities to 0.1, before normalizing, for the sake of computing KL-divergence (that requires nonzero probabilities). This is required because there may be locations which are visited in the synthetics but not in the real traces, or vice-versa.

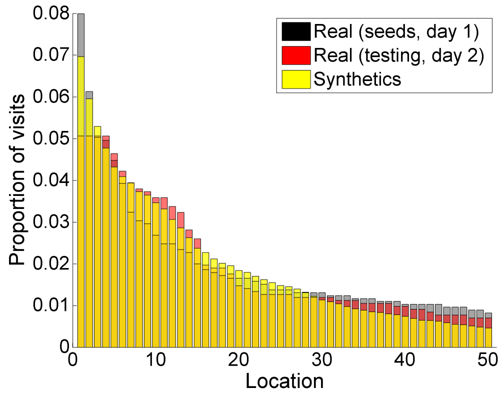


Figure 7.3: Distribution of visiting probability for top-50 locations; real versus synthetic datasets. We overlay the histograms of the three datasets (i.e., day 1, in black; real day 2, in red; synthetics, in yellow).

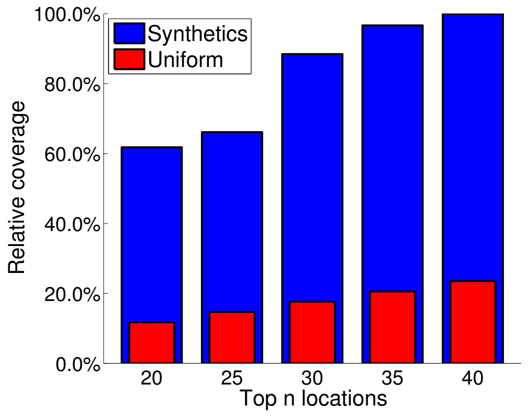


Figure 7.4: Relative coverage of top n (most frequently visited) locations. The coverage is reported relative to the real (testing, day 2) dataset. Uniform visiting of all locations (400 in total) is used as comparative baseline.

shows the results for this case, which plots a histogram where the distributions for different datasets are overlayed (with some transparency). The error (of the synthetic dataset) for this case (i.e., top 50) is significantly lower than that for the entire distribution. This strongly indicates that the information about the popular locations (i.e., the most important ones) is largely preserved.

(c) Top n coverage of locations. For tasks such as (1), (3), and (5), it may not be sufficient to ensure that the distribution of visits is preserved. Indeed, it may be required to ensure that if a location is in the top n most frequently visited locations in the real dataset, it is also in the top n most frequently visited locations in the released (synthetic) dataset. Therefore, we measure across two datasets (e.g., one real and one synthetic), how many locations in their respective top n they have in common, for various values of n.

Specifically, we take the n most frequently visited locations of the real (day 1) dataset. For each of the other datasets, we then compute how many top n locations (from the input dataset) are also in the top n most frequently visited locations of that dataset. For the synthetic datasets and the uniform baseline, we report the relative coverage as the ratio of the coverage of that dataset and of the testing (day 2) dataset. That is, if the coverage of the real (testing, day 2) dataset is y (of the top n locations of the input dataset), and the average coverage of the synthetic datasets is x, then we report the relative coverage as $\min\left(\frac{x}{y}, 1.0\right)$. The results are shown in Fig. 7.4. The relative coverage of the synthetic traces is in the $[61\%, 100\%]$ range, whereas for the uniform baseline it is in the $[11\%, 24\%]$ range, indicating a high-level of preservation.

(d) Users' time allocation. For semantic labeling (2) and other tasks, the users' temporal behavior cannot be ignored. Indeed [58, 59] use the amount of time spent per location for each user as a major feature.

In order to evaluate this, we do the following: for each dataset and each user, we calculate the time spent at each location, among the locations visited. That is, we calculate, for the three most popular locations of that user, what proportion of the time is spent in each. We perform this calculation across all 30 users and normalize the result. We compare this distribution for the real and synthetic datasets. Section 7.3 shows the KL-divergence of the real (seed, day 1) dataset to the synthetic datasets and baselines: real (testing, day 2) dataset; uniform time allocation (each user spends $1/k$ proportion of time at each of the k locations); random time allocation (each user spends a uniformly random proportion of time at the location). To visualize those results further, Fig. 7.5 shows the distribution across all 30 users (for each dataset) for the most popular location (only). The statistic is highly preserved in the synthetic traces; sometimes the synthetics' distribution is closer to that of the real (day 1) dataset, than the distribution of the real (testing, day 2) dataset is.

Table 7.2: KL-divergence of the distribution of users' time allocation among the three most popular locations (of each user) of the real (day 1) dataset against synthetic datasets and baselines.

	Real (day 2)	Synthetics	Uniform	Random
1st	0.0189	0.0125 ± 0.0022	0.1652	0.6794
2nd	0.0026	0.0092 ± 0.0031	0.0778	0.5360
3rd	0.0114	0.0089 ± 0.0036	0.0779	0.5092

(e) Spatiotemporal mobility features. When constructing mobility models from location data (4), the overall geographic and temporal behavior of users' mobility is used.

To evaluate this, we compare the basic mobility statistics obtained from the real and synthetic datasets. We compute the aggregate mobility model for each synthetic dataset, and compare its geographic similarity with the real (day 1) dataset. More precisely, for a synthetic dataset Y, we compute $\langle \bar{p}_Y, \bar{\pi}_Y \rangle$ and compute its similarity to $\langle \bar{p}, \bar{\pi} \rangle$. The statistical similarity of $\bar{p}_Y$ with $\bar{p}$ over all synthetic datasets is

$$[0.8061\,(\text{average}), \quad 0.8073\,(\text{median}), \quad 0.0060\,(\text{std})],$$

and the results for the statistical similarity of $\bar{\pi}_Y$ with $\bar{\pi}$ is

$$[0.7856\,(\text{average}), \quad 0.7867\,(\text{median}), \quad 0.0152\,(\text{std})].$$

Both these results show a strong correlation between average/aggregate mobility information of real and synthetic datasets.

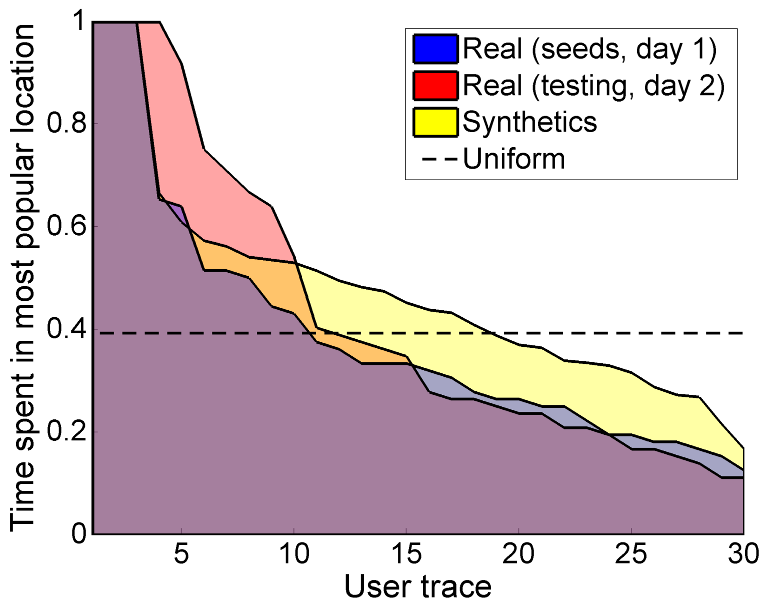


Figure 7.5: Distribution of the proportion of time spent in the most popular location (of each user) of the real datasets against synthetic datasets. The information is presented as an area plot, where the distribution for each dataset is plotted as surface of a different color (i.e., day 1, in blue; real day 2, in red; synthetics, in yellow). The areas are overlayed on top of one another. Therefore, the distance between the distribution of two datasets is represented by their non-overlapping area. For example, the yellow and orange regions represent areas where the synthetics' distribution is either non-overlapping (yellow) or overlaps with the real day 2 dataset's distribution, but not with the input (day 1) dataset. The majority of the colored area is a region where the real and synthetics distributions overlap (i.e., purple region). This indicates a high-level of preservation.

(f) Semantic mobility features. In contrast to other applications, identifying areas for new businesses, i.e., task (5) explicitly takes into account semantic features of the input location data. Specifically, it takes into account visits to semantically similar venues and transitions between different types of venues. Consequently, it is meaningful to measure the extent to which semantic features of a real dataset are preserved in a synthetic dataset.

To evaluate this, we proceed in two steps. We first compute the semantic similarity of each synthetic trace with its own seed trace to check if the semantic features of the original traces are indeed preserved. Fig. 7.7 illustrates the distribution of this value over all synthetic traces. Clearly, the distribution is biased towards higher similarity values. So, the synthetic traces considerably preserve the semantic features of the real traces.

In the second step, we look at whether the set of synthetic traces preserves the inner similarity between the set of traces. In Fig. 7.8, we present the correlation between two distributions: semantic similarity among real traces, and semantic similarity among synthetic traces. The Q-Q plot shows a significant correlation between these two distributions; they

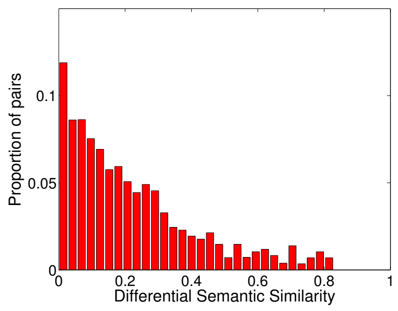


Figure 7.6: Histogram of the differential semantic similarity between synthetic and real traces. The distribution of the absolute difference $|\mathsf{sim}_{\mathsf{S}}(s, y) - \mathsf{sim}_{\mathsf{S}}(s', y)|$, for all pairs of y (plus its seed s) and s'.

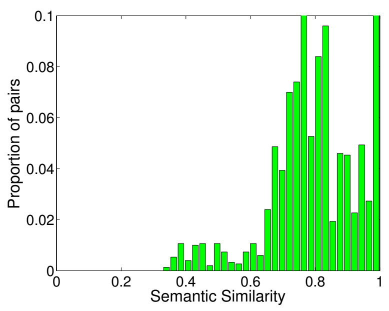


Figure 7.7: Normalized histogram of the semantic similarity of all distinct pairs of: each of the 30 real traces, along with their associated synthetic traces.

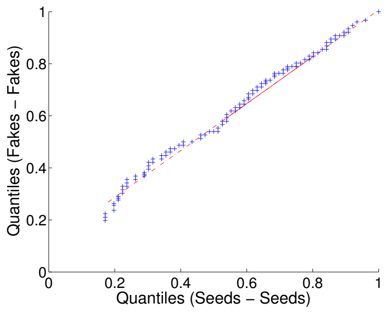


Figure 7.8: Q-Q plot comparing two distributions: semantic similarity among all real seed traces, and semantic similarity among all synthetic traces. The two distributions are strongly correlated.

are strongly linearly related. This reflects that in addition to maintaining the information about each seed, we also preserve the statistical relation among the traces.

7.4 EVALUATION: LOCATION-BASED SERVICES

In this section, we consider the use of synthetic trajectories (*fakes*) in accessing location-based services (LBS). Specifically, we compare our proposed technique with existing methods. We evaluate the utility and privacy according to well-established metrics for this scenario and measure how well our fakes perform against state-of-the-art inference attacks.

7.4.1 Setup

In this setting, a user shares her current location with a location-based service. The service provider, in return, provides contextual information about the shared locations (e.g., list of nearby restaurants, current traffic information on the road). The user makes such queries over time whenever she wishes to obtain contextual information.

In order to protect her location privacy, i.e., hiding her location at the time of access to the LBS and also preventing the inference of the full trajectory, the user's device sends a number of fake locations along with her true location. For example, if two fake locations are used, then every time the user makes a query, the device sends locations x, y, z to the service provider. Out of $\{x, y, z\}$, one location is the user's actual (i.e., true) location and the other two are fakes. The service provider does not know which of x, y, z is the true location, but may be able to filter out fake locations over time (i.e., over multiple queries over time), if the

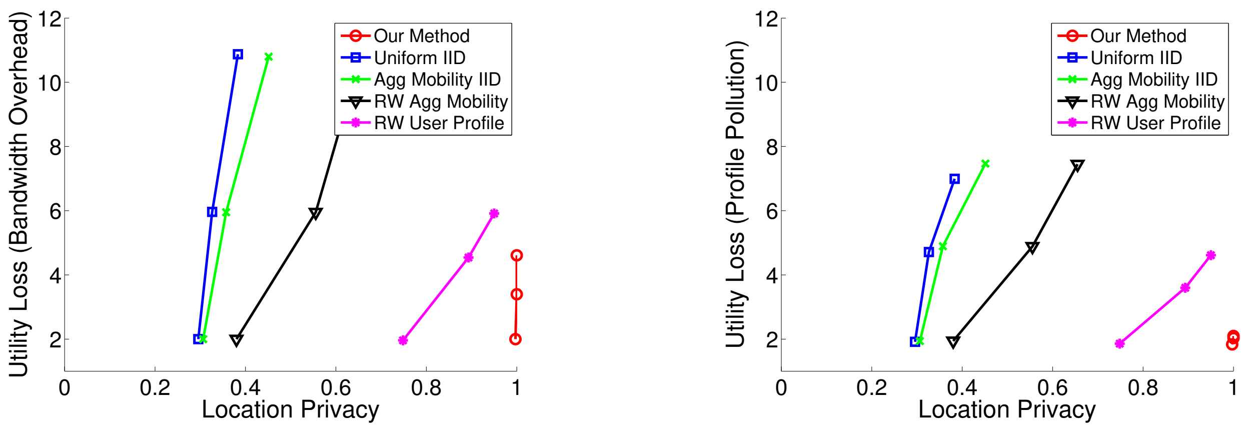


Figure 7.9: Location privacy versus utility loss for different fake generation algorithms. The privacy is measured as probability of error of adversary in guessing the correct location of users ([41, 65]). We plot the median location privacy across all LBS users. A user makes, on average, an LBS query every 40 minutes. We evaluate the use of $1, 5, 10$ fake traces, hence three dots for each algorithm. (We repeat the experiment 20 times and take the average: 4 times with a different selection of fake traces, and for each of such selection, 5 times to eliminate the randomness.) The utility loss is (left) the bandwidth overhead ([66, 67]), i.e., number of distinct locations sent to the server; and (right) the profile pollution, i.e., the number of distinct semantic classes exposed for each LBS access.

fake locations are not believable (i.e., plausible). This is why it is crucial to use synthetic *traces* as opposed to independent fake *locations*.

The fake locations are obtained as followed. First, we generate a collection of synthetic traces. The users can select from these traces and store them in their devices. Then, when a user makes an LBS query, say at time t, she picks the i^{th} fake location (reported to the LBS) as the location which is visited at time t in fake trace i. All existing fake location generation methods (i.e., [65–71]) work this way; but the techniques differ in how the fake locations are generated. Existing fake locations generation methods can be classified into four categories.

Uniform IID [65]: Generate each fake location independently and identically distributed from uniform probability distribution.

Aggregate Mobility IID [65]: Generate each fake location independently and identically distributed from the aggregate mobility probability distribution $\bar{\pi}$.

Random Walk on Aggregate Mobility [66–68, 70]: Generate a fake trace by doing a random walk on the set of locations following the probability distribution $\bar{p}$.

Random Walk on User's Mobility [69, 71] [3]: Do a random walk on the set of locations following the probability distribution $p(u)$ to generate a fake trace.

[3] [69, 71] make fakes dependent on the user's location over time (used to establish the position of dummies). We make this probabilistic and so assume usage of the user's mobility profile instead. This leads to overestimating the privacy gain of the original algorithm.

For Uniform IID and Aggregate Mobility IID, we evaluate exactly the method described in [65]. For the other two we evaluate a representative method in each case. In addition to this, we evaluate our proposed technique, which only differs from these in that the fake traces are generated using the method described in Section 6.1.1. In all cases, when a user makes use of a location-based service, both a query for her real location *and* queries for the fake locations are sent to the LBS. Because of this, the user's device must, upon receiving the responses from the service provider, filter out information not related to her true query.

7.4.2 Privacy Metric

The adversary (e.g., the service provider) who observes the LBS queries made by the user's device wants to find the true sequence of locations visited by the user. To do this, the adversary runs an inference attack which (if successful) results in filtering out the fake locations. For this, he makes use of the aggregate mobility model $\langle \bar{p}, \bar{\pi} \rangle$ and uses state-of-the art location inference attack [41]. The attack is a localization attack which consists in finding the user's (true) location at each time, given the observation (i.e., the sequence of locations queried to the LBS). This is a well-known inference problem for Hidden Markov models which can be solved efficiently using dynamic programming.

The metric to quantify the privacy is the *probability of error* of inference attack on guessing the correct location. This is the metric predominantly used in the literature, in works such as [41,65]. To put it simply, this metric consists in calculating the fraction of true locations that are missed by the adversary. For example, if the user queries LBS on three different occasions, but the adversary only correctly infers the true location once (i.e., the inference attack correctly filters out the fake locations) then the user's location privacy is 2/3.

7.4.3 Utility Metric

With all synthetic generation techniques (i.e., ours and [65–71]) for the LBS scenario, the user's real location will always be among the locations queried to the LBS. Therefore, as identified by related work, there is no utility loss in terms of quality of service degradation. That is, the user will always obtain an accurate response to her query (after filtering out responses corresponding to fake location queries).

Therefore, we measure the utility loss as the *bandwidth overhead* which is used in the literature on fake generation techniques (e.g., [66,67,69]). The bandwidth overhead is calculated as the number of locations (i.e., real plus fakes) sent to the LBS for each user query.

Beyond traditional location-based services, some service providers (e.g., Google Now) profile the user's interest over time based on the type of locations she visits. This is to provide recommendations or reminders. In such cases, queries that are sent to the server can "pollute" the user's profile, hence reduce the predictability power of the service provider to provide useful recommendations. To further evaluate utility for such location-based recommender services, we calculate the number of (distinct) semantic clusters among the locations sent by the user at each time. We call this metric: *profile pollution*.

7.4.4 Results

Fig. 7.9-(left) shows the tradeoff between location privacy and utility for various methods of generating fake traces. We evaluate the utility loss in terms of two metrics: bandwidth overhead (Fig. 7.9-(left)) which is predominantly used in the literature, and also the profile pollution (Fig. 7.9-(right)). We evaluate the privacy for three different number of fake traces: $1, 5, 10$. Although the number of fake traces are the same, across different algorithms, the average number of distinct locations sent to the LBS is not the same. This is because of the potential overlap between fake traces available to the user. Methods such as *Uniform IID*, *Agg Mobility IID*, and *RW Agg Mobility* have a high randomness in selecting fake traces from all possible locations. Thus, the chance of overlap is small. Our method and the *RW User Profile* method have both lower bandwidth overhead.

Results show that our method clearly outperforms all the existing techniques, especially the random strategies. For the case of *RW User Profile* method, the privacy level against the tracking attack gets closer to what we achieve (which is almost maximum), due to the fact that the fake traces generated by *RW User Profile* are semantically very similar to the user's locations, and hence creates high confusion, hence error, for the adversary. However, it is very important to note that the *RW User Profile* is never a privacy-preserving fake injection method as the adversary can easily de-anonymize the user, no matter if he makes mistakes on exactly tracking the user at each access time (as shown here).

Overall, the plot shows that our method is the strongest fake generating algorithm. Note that the absolute privacy levels changes if the adversary knowledge changes. But, what we are interested in is the relative gain of our method to others.

Chapter 8: Application: Census Data

In this chapter, we apply the framework to synthesize census data records in a privacy-preserving way. This case study on census data originally appeared in [32]. We use Mechanism 3.1 with the privacy test based on plausible deniability (Section 4.1) and the generative model described in Section 6.2.

8.1 DATASET

The 2013 American Community Survey (ACS) [72] contains upwards of 3 million of individual records. Each record includes a variety of demographics attributes such as age, sex, race, as well as attributes related to the individual's income such as yearly income in USD.

The ACS dataset has been used for various purposes ranging from examining the relationship between education and earnings [73] to looking at current language use patterns of children of immigrants [74]. Furthermore, the prominent UCI Adult dataset, which provides a well-established benchmark for machine learning tasks, was extracted from the 1994 Census database. The 2013 ACS dataset contains similar attributes so we process it in a manner similar to how the Adult dataset was extracted. In particular, we extract the same attributes whenever possible.

As pre-processing, we discard records with missing or invalid values for the considered attributes (Table 8.1). Table 8.2 shows some statistics of the data cleaning and extracted dataset. This is a highly dimensional dataset despite having only 11 attributes, there are more than half a trillion *possible* records and out of the roughly 1.5 million records obtained after cleaning, approximately two third are unique.

We aggregate (Section 6.2.3) values of the age attribute in bins (i.e., buckets) of 10, i.e., 17 to 26, 27 to 36, etc. (Following the rules used to extract the Adult dataset, we only consider individuals older than 16.) We also aggregate the values of: hours worked per week (HPW), in bins of 15 hours; education, to aggregate education level below a high-school diploma in a single bin, and high-school diploma but not college into (another) single-bin. This aggregation step is performed based on the data format and the semantics of attributes (and thus is privacy-preserving). It is done only for structured learning (Section 6.2.3); both the input and output data format remain the same.

Table 8.1: Pre-processed 2013 ACS dataset attributes.

Name	Type	Cardinality (Values)
Age (AGEP)	Numerical	80 (17 to 96)
Workclass (COW)	Categorical	8
Education (SCHL)	Categorical	24
Martial Status (MAR)	Categorical	5
Occupation (OCCP)	Categorical	25
Relationship (RELP)	Categorical	18
Race (RAC1P)	Categorical	5
Sex (SEX)	Categorical	2 (male or female)
Hours Worked per Week (WKHP)	Numerical	100 (0 to 99+)
World Area of Birth (WAOB)	Categorical	8
Income Class (WAGP)	Categorical	2 ($\leq$ 50K, $>$ 50K)[USD]

Table 8.2: 2013 ACS data extraction and cleaning statistics.

Records	$3,132,796$ (clean: $1,494,974$)
Attributes	11 (numerical: 2, categorical: 9)
Possible Records	$540,587,520,000$ ($\approx 2^{39}$)
Unique Records	$1,022,718$ (68.4%)
Classification Task	Income class

8.2 SETUP

We split the 2013 ACS dataset (Section 8.1) into three disjoint parts D, D_T and D_P such that each of D_T and D_P contain about $280,000$ records while D contains the rest ($735,000$ records). We use D_T and D_P as training data to learn the model's structure and parameters. We follow Section 6.2 to train the generative model in a way that satisfies differential privacy for $\varepsilon = 1$ (though we also give some results for $\varepsilon = 0.1$) and $\delta = 2^{-30} \approx 10^{-9}$.

We want to compare the quality of our generated synthetics with real records (coming from the input dataset) and privacy-preserving marginals (Section 6.2.2) which we refer to as *reals* and *marginals*, respectively. The synthetics we generate are referred by their generation parameters (e.g., $\omega = 10$). Unless otherwise stated, we set $k = 50$, $\varepsilon_0 = 1$, $r = 4$, and ω is set to vary between 5 and 11.

We maintain an independent test set of roughly $100,000$ records. Evaluation of classifiers (in this chapter) uses at least $100,000$ records for training and a (disjoint) testing set of size at least 30% of the size of the aforementioned training set.

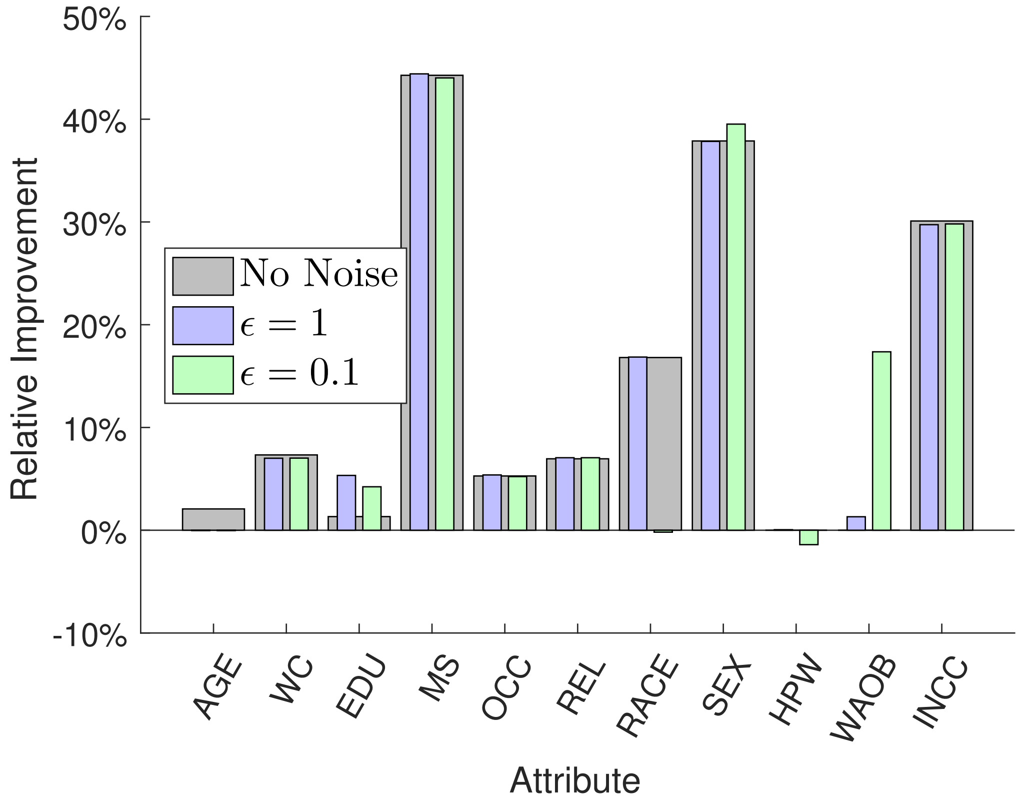


Figure 8.1: Relative improvement of model accuracy of the un-noised, $\varepsilon = 1$, and $\varepsilon = 0.1$ models, with respect to the baseline (marginals). Overall, the improvement for $\varepsilon = 1$ or $\varepsilon = 0.1$ is comparable to that for the un-noised version. Adding noise to achieve differential privacy for structure learning (Section 6.2.3) can lead to a different acyclic graph of the model. (This is why there is a significant difference in improvement for attributes RACE and WAOB between $\varepsilon = 1$ and $\varepsilon = 0.1$.)

8.3 EVALUATION

The evaluation is divided into four parts: (Section 8.3.1) statistical measures (how good are the synthetics according to well-established statistical metrics); (Section 8.3.2) machine learning measures (how good are the synthetics for machine learning tasks, specifically classification); (Section 8.3.3) distinguishing game (how successful is an adversary at distinguishing between a real record and a synthetic one); and (Section 8.3.4) performance measures (how computationally complex it is to generate synthetics).

8.3.1 Statistical Measures

We evaluate the quality of the synthetics in terms of their statistical utility, i.e., the extent to which they preserve the statistical properties of the original (input) dataset. We can

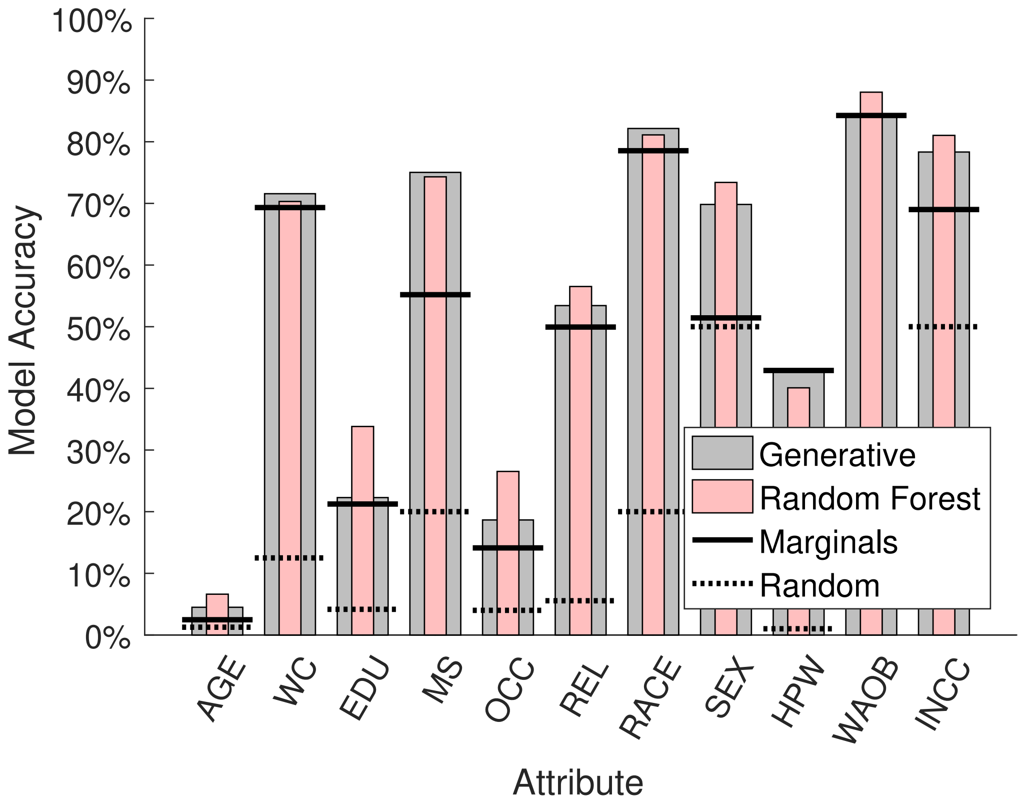


Figure 8.2: Model accuracy. The difference between the random forest accuracy and the marginals accuracy indicates how informative the data is about each attribute.

do this at the level of the generative model (Section 6.2.1) itself. Concretely, we directly quantify the error of the privacy-preserving generative model before any synthetic record is generated. We do this for each attribute by repeatedly selecting a record from the input dataset (uniformly at random) and using the generative model to find the most likely attribute value (of that attribute) given the other attributes. The generative model error is then measured as the proportion of times that the most likely attribute value is *not* the correct (i.e., original) one. We repeat this procedure millions of times to quantify the average error of the model for each attribute. Because the generative model is made differentially private by adding noise (Section 6.2.5) we additionally repeat the whole procedure 20 times (learning a different private model each time) and take the average.

The results are shown in Figs. 8.1 and 8.2. Fig. 8.1 shows the relative decrease in model error (i.e., improvement of model accuracy) over the (privacy-preserving) marginals; it shows this improvement for the un-noised, $\varepsilon = 1$-differential privacy, and $\varepsilon = 0.1$-differential privacy generative models. There is a clear accuracy improvement over marginals, in addition to a low decrease in improvement between the un-noised model and the $\varepsilon = 1$ and $\varepsilon = 0.1$ cases.

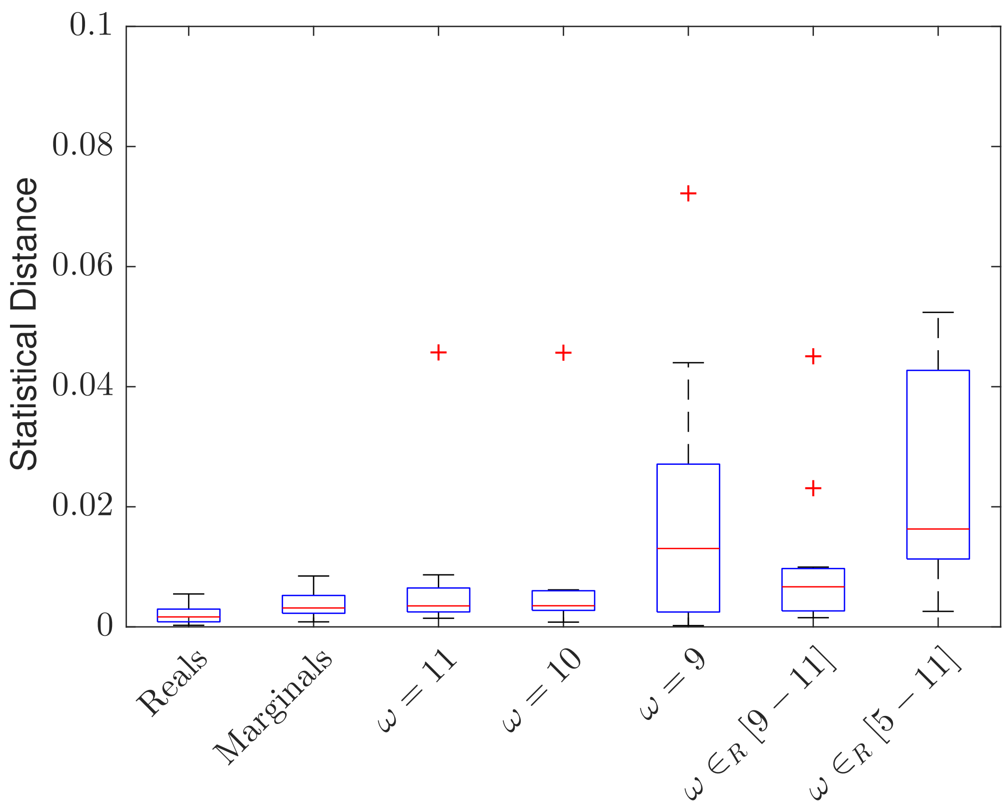


Figure 8.3: Statistical distance for individual attributes of two distributions: reals and (other) reals; reals and marginals; reals and synthetics (for varying ω). The smaller the statistical distance the more information is preserved. The distance of reals and $\omega = 11$ and $\omega = 10$ synthetics is similar to that of reals and marginals.

Fig. 8.2 shows the accuracy of the un-noised generative model against the (un-noised) marginals, random guessing (baseline), and the best classifier we could find (trained on as many records as the generative model), the random forest (RF). While RF's accuracy is sometimes higher than that of the generative model, the accuracy of the latter is in many cases significantly higher than that of marginals and random guessing. We conclude that while the proposed generative model does not perform as well as RF (though making RF differentially private would certainly lower its performance) it does perform significantly better than marginals (or random guessing).

In addition to the model loss, we directly evaluate whether the generated synthetics preserve statistical properties by comparing the probability distributions of the synthetics with the reals and marginals. Specifically, for reals, marginals and synthetics datasets, we compute the distribution of each attribute and of each pair of attributes. We compare each of these distributions to those computed on (other) reals and quantify their statistical distance.

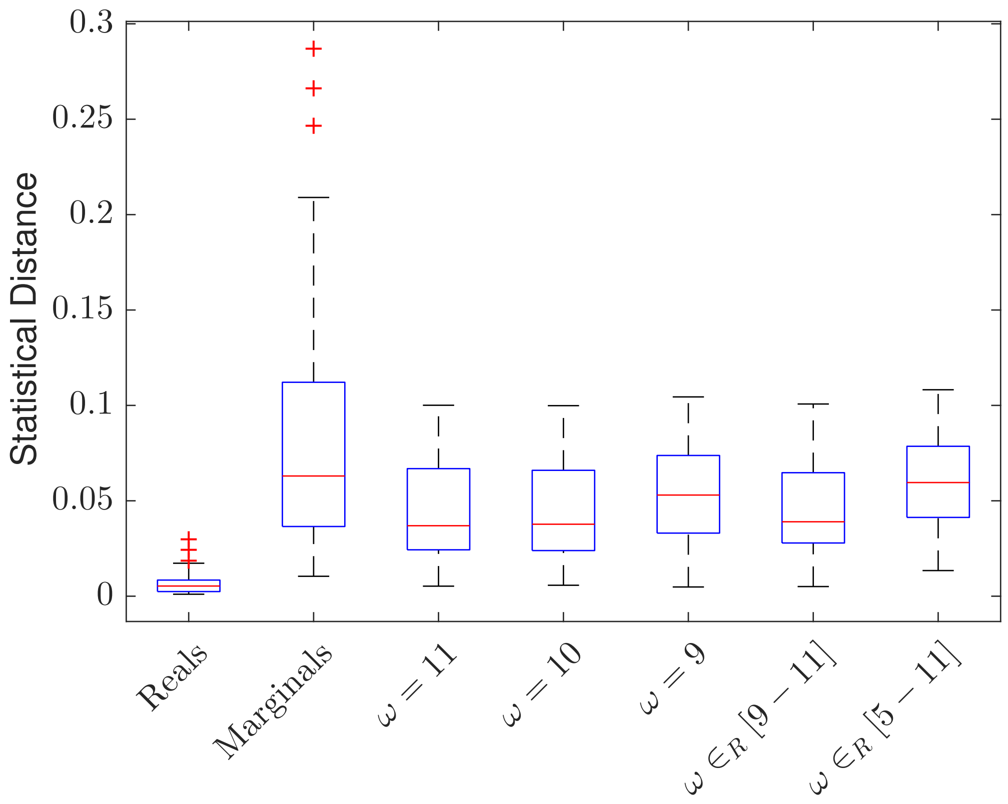


Figure 8.4: Statistical distance for pairs of attributes of two distributions: reals and (other) reals; reals and marginals; reals and synthetics (for varying ω). The smaller the statistical distance the more information is preserved. The distance of reals and synthetics is significantly smaller than that of reals and marginals.

The results are shown in Figs. 8.3 and 8.4, where Fig. 8.3 shows box-and-whisker plots for the distance of the distributions of each attribute separately, and Fig. 8.4 shows box-and-whisker plots for the distance of the distributions of all pairs of attributes. While marginals do well for single attribute and sometimes outperform our synthetics (though the statistical distance for all datasets is small), synthetics clearly outperform marginals for pairs of attributes. We conclude that the generated synthetics preserve significantly more statistical information than marginals.

8.3.2 Machine Learning Measures

In addition to preserving statistical properties of the original (input) dataset, the synthetics should also be suitable to various machine learning tasks. In particular, given a learning task, we can evaluate the extent to which synthetics are suitable replacements for a real dataset. For this kind of census data, a natural classification task is to predict a person's income class (i.e., $\geq$ 50K or $<$ 50K) using the other attributes as features.

Table 8.3: Classifier comparisons. The agreement rate is the proportion of times that the classifier makes the same prediction as a classifier trained on real data.

	Accuracy			Agreement Rate		
	Tree	RF	Ada	Tree	RF	Ada
Reals	77.8%	80.4%	79.3%	80.2%	86.4%	92.4%
Marginals	57.9%	63.8%	69.2%	58.5%	65.4%	75.6%
$\omega = 11$	72.4%	75.3%	78.0%	73.9%	79.0%	83.0%
$\omega = 10$	72.3%	75.2%	78.1%	73.8%	78.9%	83.6%
$\omega = 9$	72.4%	75.2%	77.5%	73.9%	79.2%	82.4%
$\omega \in_R [9-11]$	72.3%	75.2%	78.1%	73.7%	79.0%	83.9%
$\omega \in_R [5-11]$	72.1%	75.2%	78.1%	73.6%	79.2%	83.3%

We train various classifiers on the synthetic datasets and on the real (input) dataset. We then compare: the classification accuracy obtained, and the agreement rate of the learned classifiers. Specifically, for two classifiers trained on different datasets (but with the same classification task), we define the *agreement rate* to be the percentage of records for which the two classifiers make the same prediction (regardless of whether the prediction is correct). Given that we look at the agreement rate of classifiers trained on reals and synthetics, the agreement rate reveals the extent to which the classifier trained on synthetic data has learned the *same* model as the classifier trained on real data.

Table 8.3 shows the obtained results for three (best) classifiers: Classification Tree (Tree), Random Forest (RF), and AdaBoostM1 (Ada). The accuracy and agreement rate are calculated as the average over 5 independent runs, that is, for each run, we use different (randomly sampled) training and testing datasets. Overall, both the accuracy and the agreement rates of the synthetics are significantly closer to that of the reals than the marginals are.

In addition to comparing the best classifiers trained on real data versus those trained on synthetic data, we can also compare privacy-preserving classifiers trained on real data versus non-private classifiers trained on (privacy-preserving) synthetic data. In particular, Chaudhuri et al. [75] propose two techniques based on empirical risk minimization to train logistic regression (LR) and support vector machines (SVM) binary classifiers: output perturbation (noise is added to the learned model), and objective perturbation (noise is added to the objective function of the minimization problem). To train such classifiers, we first pre-process our datasets following the instructions in [75]: we transform each categorical attribute into an equivalent set of binary attributes, and normalize features so that each feature takes values in $[0, 1]$ and subsequently further normalize each training example such that its norm is at most 1. The target attribute for classification is again the person's income class. The method proposed in [75] has two parameters: the privacy budget ε which we set to 1 (the same as for our generative model), and λ which is a regularization parameter. We use the code of [75], which we obtain courtesy of the authors, to train the LR and SVM classifiers. Because the classification models vary greatly depending on λ, we vary its value in the set $\{10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}\}$ and (optimistically) pick whichever value maximizes the accuracy of the non-private classification model.

We report the accuracy obtained in each case in Table 8.4, where we compare non-private, output perturbation differential privacy, and objective perturbation differential privacy classifiers trained on real data with non-private classifiers trained on our synthetic datasets (for various values of ω). Non-private LR and SVM classifiers trained on our (privacy-preserving) synthetic datasets are competitive with differentially-private LR and SVM classifiers trained on real data. The classifiers trained on our privacy-preserving synthetics outperforms ε-differential privacy LR classifier and only achieves 1% lower accuracy than the objective-perturbation ε-differential privacy SVM. This is significant because the technique to train the ε-differential privacy LR and SVM is specifically optimized for that task. In contrast, our synthetics are *not* specifically generated to optimize any particular classification task; instead the general objective is to preserve the statistical properties of real data.

Table 8.4: Privacy-preserving classifiers comparison.

	LR	SVM
Non Private	79.9%	78.5%
Output Perturbation	69.7%	76.2%
Objective Perturbation	76.3%	78.2%
Marginals	68.9%	68.9%
$\omega = 11$	77.6%	77.2%
$\omega = 10$	77.7%	77.1%
$\omega = 9$	77.5%	77.1%
$\omega \in_R [9-11]$	77.5%	76.9%
$\omega \in_R [5-11]$	77.7%	77.3%

Table 8.5: Distinguishing game. Random forest (RF) and classification tree (Tree) easily distinguish marginals from reals but perform significantly less well in distinguishing synthetics from reals.

	Reals	Marginals	$\omega =$ or $\in_R$				
			11	10	9	$[9-11]$	$[5-11]$
RF	50%	79.8%	62.3%	61.8%	63.0%	60.1%	61.4%
Tree	50%	73.2%	58.9%	58.6%	59.8%	57.9%	58.4%

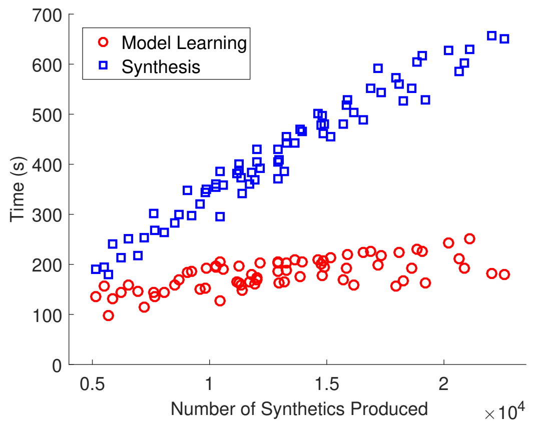


Figure 8.5: Synthetic generation performance. The parameters are: $\omega = 9$, $k = 50$, $r = 4$. The time to generate 10,000 synthetic records on a single-core is less than 10 minutes.

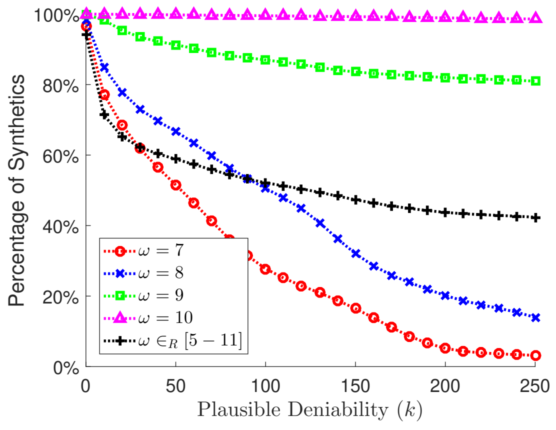


Figure 8.6: Percentage of candidates which pass the privacy test for various values of k and ω ($r = 2$). The percentage decreases for higher privacy (i.e., larger k) but remains significant.

8.3.3 Distinguishing Game

We evaluate the quality of synthetic datasets by the extent to which the synthetics can be distinguished from real records. For this, we use the methodology from the distinguishing game (Definition 5.1) introduced in Section 5.1. The role of the adversary is played by the two best classifiers (those that best distinguish synthetics from reals): Random Forest (RF) and Classification Tree (Tree). Specifically, we provide 50,000 records from both a real dataset and a synthetic dataset (i.e., 100,000 total) as training examples to the (binary) classifier. We then evaluate the accuracy on a 50% mix of real and synthetic records which were not part the training set. Table 8.5 shows the results: both classifiers achieve reasonably high (79.8% and 73.2%) accuracy in distinguishing marginals from real records. However, both classifiers achieve lower accuracy (63%) in distinguishing synthetics from reals.

8.3.4 Performance Measures

In addition to how much utility they preserve, synthetics also need to be easy to generate. The generation is a parallel process, so we measure the time taken for learning the privacy-preserving generative model (model learning) and synthetics generation (synthesis). Fig. 8.5 shows the time taken to produce various number of synthetics records (totaling over 1 million). The machine used for the experiment runs Scientific Linux and is equipped with an Intel Xeon E5-2670 processor (2.60GHz) with 32 processing units and 128GB of RAM. We ran 96 instances (16 in parallel at a time) and picked a random maximum runtime for each instance between 3 and 15 minutes.

The generator outputs all synthetics produced regardless of whether they pass the privacy test. Naturally, only those which pass the test should to be released. Thus, the extent to which we can synthesize large (privacy-preserving) datasets depends on how easy it is to find synthetics that pass the privacy-test (Section 4.1). To evaluate this, we set $r = 2$ but vary k and ω. We measure the proportion of synthetics which pass the privacy test. The results are shown in Fig. 8.6: even for stringent privacy parameters (e.g., $k = 100$) a significant proportion (i.e., over 50% for $\omega \in_R [5 - 11]$) of synthetics pass the test.

Chapter 9: Application: Medical Data

In this chapter, we synthesize medical records.

9.1 DATASET

We use the 2013 HCUP Nationwide Inpatient Sample (NIS) [76] which contains millions of fine-grained medical discharge records. As pre-processing, we clean the data by removing records with missing/invalid attribute values and extract cancer diagnosis (i.e., the corresponding CCS categories [77]). Table 9.6 shows the attributes selected.

The processed data contains over 6 millions records of moderately high dimension, i.e., $|U| = 2^{42}$. We split the dataset into three disjoint subsets of records. We use the first for training the generative models, the second as independent (holdout) test set, and the last as input data to the mechanism.

9.2 SETUP

We use two different latent-space models architectures (Section 6.3): PCA and VAE.

PCA-5. This generative model is based on principal components analysis with $w = 5$ principal components. To "train" the model, we follow the procedure described in Section 6.3.4 using the training set. This process results in obtaining a mean vector and a projection matrix. We remark that the procedure can also be performed in a way that satisfies differential privacy using the technique from [55].

To perform synthesis, we use the input dataset and follow the procedure from Section 6.3.4 using the mean vector and projection matrix obtained from training.

VAE-3. This generative model is a variational auto-encoder with latent width $w = 3$. The encoder and decoder networks each have a single hidden layer with 6 neurons. Unless otherwise stated, the Gaussian KL-divergence regularization constant is $C = 0.5$. We train the model for 200 epochs with a batch size of 500, and the Adam optimizer with a learning rate of 0.004. The loss function is the Bernoulli negative log-likelihood (Section 6.3.4).

For both PCA-5 and VAE-3, we use the probabilistic projection technique (Section 6.3.2).

Seedless baseline. As a baseline, we define seedless generative models for both PCA-5 and VAE-3. The seedless variants consists in sampling Gaussian samples from the latent-space and applying the decoder, followed by probabilistic projection. Specifically, we draw samples from $\mathcal{N}(\mu, \beta\mathbf{I})$, where μ and β are estimated by encoding records from the test set. Due to regularization, we typically have $\mu \approx \mathbf{0}$ and $\beta \approx 1$.

9.3 EVALUATION

For PCA and VAE, we produce privacy-preserving synthetic datasets using Mechanism 3.1 with Privacy Test 3.2 (geometric noise), denoted as PP PCA-5 and PP-VAE-3, respectively.

Unless otherwise stated experiments are performed on datasets of at least $100'000$ records. We evaluate our synthetics with regards to the following aspects.

- In Section 9.3.1, we evaluate the quality of synthetics produced by the PCA-5 and VAE-3 generative models compared to the holdout test dataset across various distance metrics and using our distinguishability metric (Section 5.1).

- In Section 9.3.2, we explore the relationship between privacy guarantee, synthetic dataset size, and filtering loss. Unless otherwise stated, we guarantee $(1, 2^{-40})$-differential privacy for our privacy-preserving synthetics.

- In Section 9.3.3, we investigate the seedless variants of our PCA-5 and VAE-3 generative models with the goal of understanding the trade-offs in using seedbased synthesis versus seedless synthesis.

- In Section 9.3.4, we apply the theory developed in Chapter 5 to quantify experimentally the filtering loss with respect to the PCA-5 and VAE-3 generative models, as measured by distinguishability and statistical distance.

9.3.1 Quality

We perform comparisons on various metrics between our seedbased synthetics (PCA-5 and VAE-3), our (seedbased) privacy-preserving synthetics (PP PCA-5 and PP VAE-3), and the (holdout) test dataset. For the privacy-preserving synthetics we set the parameters of the privacy test so that the *entire* synthetic dataset is obtained with $(1, 2^{-40})$-differential privacy. (See Section 9.3.2 for a discussion on setting the privacy parameters.)

Probability Distance Metrics. We evaluate the extent to which the datasets preserve the statistical properties through computation of the one-way (and two-ways) marginals distribution over attributes (and pairs of attributes). In each case, then marginals are compared to that of the test dataset across three probability distance metrics: (i) the statistical distance (SD), (ii) the Hellinger distance (HEL), and (iii) the KL-divergence (KL). Because the KL-divergence is undefined for 0-probability events, we add a (very) small smoothing factor $\epsilon = 0.0001$ to all of the probabilities before calculating the KL-divergence.

Table 9.1: Probability distribution metrics (statistical distance, Hellinger distance, and KL-divergence) of various datasets compared to real records. For each dataset, we compute the mean and standard deviation of each metric for both one-way and two-way marginals.

Dataset	Marginals	Statistical (SD)	Hellinger (HEL)	KL-Divergence (KL)
		Mean (± Standard Deviation)		
Reals	1-way	0.000 (±0.000)	0.001 (±0.001)	0.000 (±0.000)
	2-way	0.001 (±0.001)	0.002 (±0.001)	0.000 (±0.000)
PCA-5	1-way	0.001 (±0.001)	0.003 (±0.004)	0.000 (±0.000)
	2-way	0.002 (±0.002)	0.006 (±0.004)	0.000 (±0.000)
PP PCA-5	1-way	0.002 (±0.003)	0.006 (±0.010)	0.001 (±0.002)
	2-way	0.003 (±0.005)	0.012 (±0.013)	0.001 (±0.003)
VAE-3	1-way	0.001 (±0.001)	0.003 (±0.002)	0.000 (±0.000)
	2-way	0.001 (±0.001)	0.006 (±0.003)	0.000 (±0.000)
PP VAE-3	1-way	0.003 (±0.005)	0.014 (±0.020)	0.004 (±0.014)
	2-way	0.007 (±0.008)	0.026 (±0.028)	0.010 (±0.022)

Results are shown in Table 9.1 where we give both the mean and standard deviation (of the three distance metrics) for both one-way and two-way marginals. It can be seen that for all three distance metrics, the synthetics have higher distance to the test data than real data records, and that achieving differential privacy degrades slightly the quality of the synthetics.

Distinguishability. We evaluate the extent to which synthetics can be distinguished from real records. For this, we use the theory developed in Section 5.1. Specifically, we let various classification algorithms play the role of the adversary in the distinguishing game (Definition 5.1). We train the classifiers using a training set composed of 50% real records (from the test dataset) and 50% "synthetic" records. We calculate the classification accuracy over a test set both containing real and "synthetic" records with the same 50% split as the training set but that does not overlap. We compute *distinguishability* as twice the absolute difference between the classification accuracy and random chance (50%). Fig. 9.1 shows the results for five popular classifiers. We observe that, for all classifiers, our synthetics and our privacy-preserving synthetics all have distinguishability in the 0.05 to 0.1 range. Albeit this is higher than reals which have near 0 distinguishability, this is low and suggests that the synthetics are close in distribution to real records. We also observe that the distinguishability is lower for PCA-5 than VAE-3 which indicates a lower model loss. We conjecture that this is the result of larger latent width: $w = 5$ for PCA and $w = 3$ for VAE. Finally, we remark that decision trees and random forest classifiers outperform others, which suggests that they are a better choice when it comes to experimentally evaluating distinguishability.

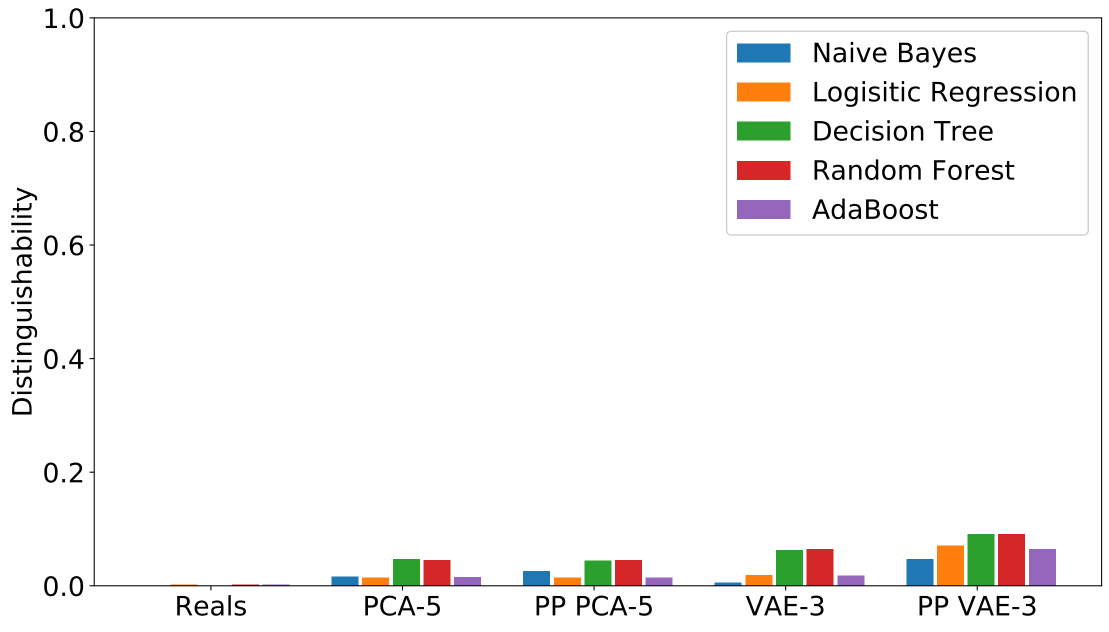


Figure 9.1: Distinguishability (for various classifiers as distinguishers) of the five datasets with respect to the real (input) data.

9.3.2 Privacy and Size

We explore the relationship between privacy, synthetic dataset size, and distinguishability. If we target a fixed privacy level, then there is a trade-off between privacy parameters (i.e., k, t, ε_0) and the number of synthetics (m) we can produce. Both the privacy parameters and the number of synthetics impact the quality of the produced synthetic dataset.

We follow the composition methodology described in Section 4.3 to set the parameters and partition the input data into 4 disjoint sets which we subsequently use to produce synthetic datasets of sizes ranging from 1000 to 50000 synthetics with the mechanism (using both PCA-5 and VAE-3). The parameters chosen are shown in Table 9.2. For this experiment, the overall privacy budget if we release *the entire synthetic dataset* is $\varepsilon = 1$ and $\delta = 2^{-40}$.

Table 9.2: Privacy vs. synthetic dataset size: examples of parameters. See Section 4.3 for details on the composition methodology.

	Synthetic Dataset Size						
	1000	5000	10000	20000	30000	40000	50000
m	250	1250	2500	5000	7500	10000	12500
k	5500	12800	18400	26500	32700	38000	42700
t	900	1700	2400	3300	5000	5800	5600
ε_0	0.0072	0.0031	0.0022	0.0016	0.0013	0.0011	0.0010

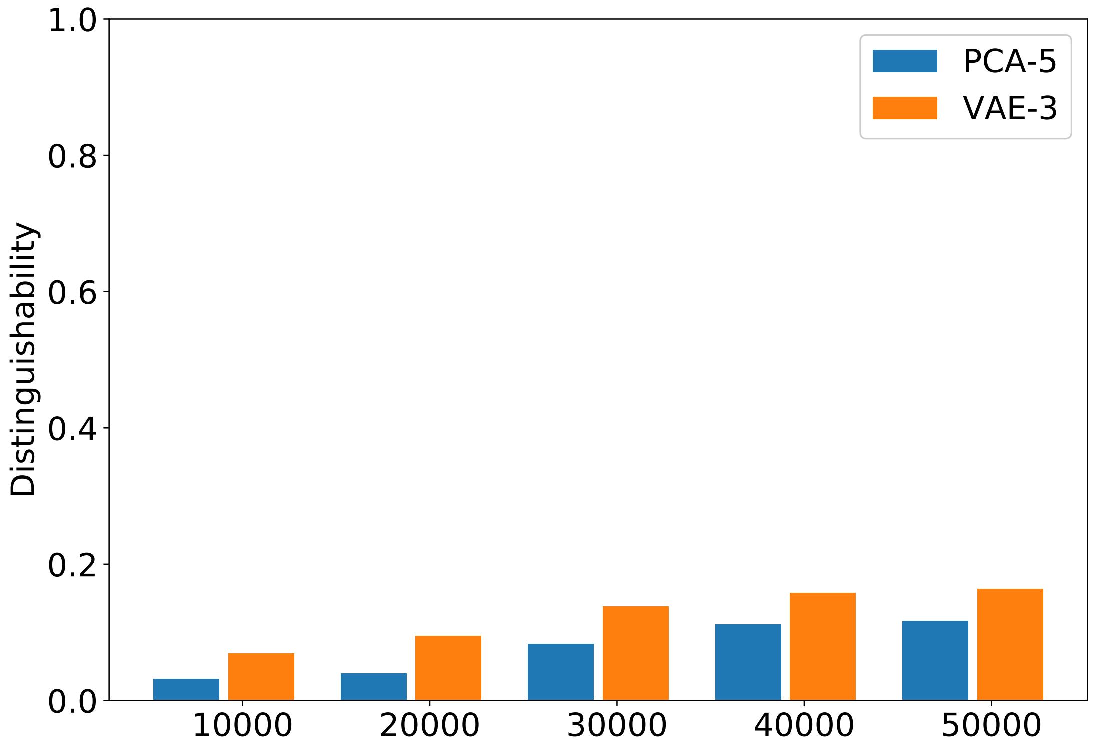


Figure 9.2: Synthetic dataset size vs distinguishability. The privacy parameters (k, t, ε_0) are set as in Table 9.2. (We exclude the 1000 and 5000 cases which do not contain enough samples to train and evaluate robust classifiers.) The overall privacy budget if we release *the entire synthetic dataset* is $\varepsilon = 1$ and $\delta = 2^{-40}$.

We compare the synthetic datasets across both generative models using the distinguishing game methodology described in Section 9.3.1. But use only the two best distinguishers, i.e., random forest and decision trees. The results are shown in Fig. 9.2. It can be seen that the distinguishability, which is (relatively) low in all cases, increases with the size of the synthetic data. This is expected as the larger the synthetic dataset size, the larger the privacy parameters k and t must be, which makes passing the privacy test more difficult and increases the filtering loss (whereas the model loss is the same).

To separately investigate the model loss and filtering loss, we use the distinguishing game methodology to measure the model loss and then vary the privacy budget (by varying the privacy parameters) and measuring the filtering loss each time. Specifically, we measure the model loss by comparing the distribution of real records with that of seedbased synthetics with no filtering (i.e., no privacy test). The results are shown in Table 9.3. It can be seen that the filtering loss is overall lower for PCA-5 than VAE-3, which is consistent with earlier results. Also, for values of $\varepsilon \geq 1$, we see that for both VAE-3 and PCA-5, the model loss is actually the main component of the utility loss. This indicates that the quality degradation of synthetics is mostly due to the model as opposed to the filtering (of the privacy tests) that is needed to achieve privacy. Finally, we observe that as ε increases, the filtering loss decreases towards 0, which is what we expect.

Table 9.3: Model loss and filtering loss. For both VAE-3 and PCA-5, we measure experimentally the model loss and filtering losses (for varying ε) using the distinguishing game methodology (Section 5.1). For the filtering loss, we use a dataset of $10'000$ synthetics produce to satisfy $(\varepsilon, 2^{-40})$-differential privacy. The privacy parameters are: $k = 39000$, $t = 3000$, $\varepsilon_0 = 0.001$ for $\varepsilon = 0.5$; $k = 18400$, $t = 2400$, $\varepsilon_0 = 0.0022$ for $\varepsilon = 1.0$; $k = 12460$, $t = 460$, $\varepsilon_0 = 0.003$ for $\varepsilon = 2.0$; $k = 7400$, $t = 200$, $\varepsilon_0 = 0.005$ for $\varepsilon = 4.0$; and $k = 4100$, $t = 100$, $\varepsilon_0 = 0.009$ for $\varepsilon = 8.0$.

	Model loss	Filtering loss				
		$\varepsilon = 0.5$	$\varepsilon = 1.0$	$\varepsilon = 2.0$	$\varepsilon = 4.0$	$\varepsilon = 8.0$
VAE-3	0.064	0.136	0.053	0.039	0.035	0.026
PCA-5	0.047	0.099	0.036	0.022	0.017	0.012

9.3.3 Seedfulness

We investigate the quality of seedless compared to seedbased synthetics. Experimentally, we observe that seedless synthetics are of lower quality than seedbased. For example, using random forests classifiers: our privacy-preserving PCA-5 synthetics have a distinguishability of 0.05 whereas the seedless PCA-5 synthetics have a distinguishability of 0.24.

Latent-space sampling. We conjecture that a reason for the poor performance of the seedless technique is that sampling from the latent-space distribution is difficult. Specifically, since the seedless technique consists of sampling from a Gaussian distribution, if encoded real records do not follow a similar distribution, we should not expect that the produced synthetics will be of high quality. The question is therefore what is the latent-space distribution of real records?

For PCA, if the input data is not Gaussian distributed, there is no guarantee that the components will be independent. As a result, it is difficult to characterize the latent-space distribution. In contrast, for VAE, the latent space distribution is constrained to be approximately Gaussian through the regularization term added to the loss function (Section 6.3.4). Thus, we expect that the larger the regularization constant C is, the closer the performance of seedbased and seedless synthetics will be. At the same time, if C is too large then the reconstruction performance of the VAE model may be poor (due to high model loss) which in turns decrease the quality of the produced synthetics.

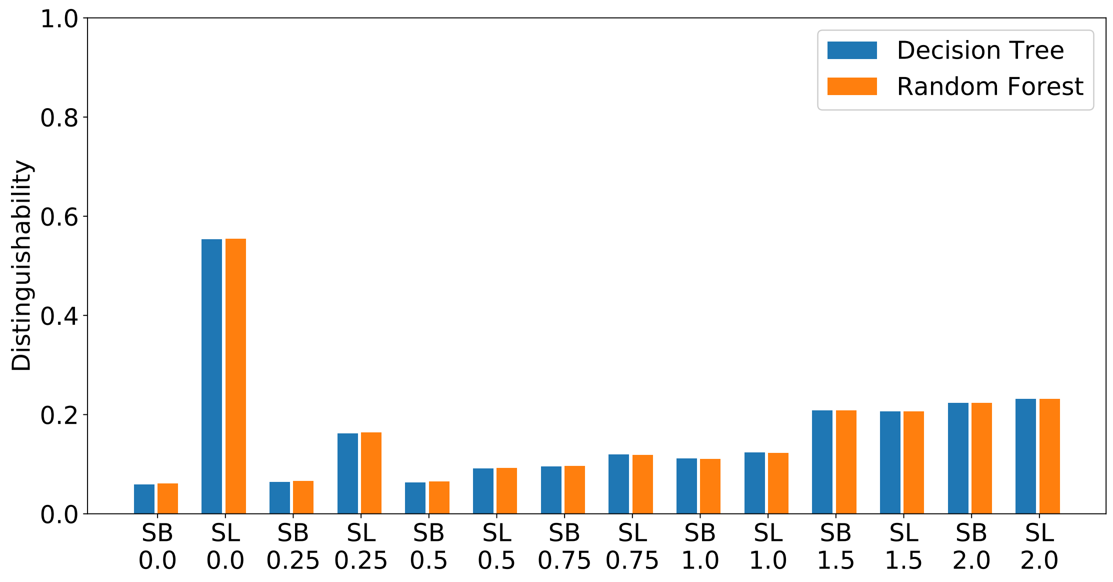


Figure 9.3: Distinguishability of VAE-3 seedbased (SB) and seedless (SL) for varying C (for the two best classifiers as distinguishers) with respect to the real (input) data.

To test this conjecture we train VAE-3 models with $C \in \{0, 0.25, 0.5, 0.75, 1, 1.5, 2\}$. In all cases we produce synthetics both through seedless and seedbased synthesis and compare the two in terms of distinguishability. For this we use the two best distinguishers, i.e., random forest and decision trees. The results are shown in Fig. 9.3. Given that increasing C increases the regularization of the model, we expect the model loss to increase with C (which is what we observe experimentally). We see that the larger C is the lower the distinguishability of seedless synthetics; simultaneously, the larger C is the higher the distinguishability of seedbased synthetics. This can be explained as follows. When C is low, the constraint on the latent-space distribution being Gaussian is weak which results in poor performance for seedless synthesis. In this case, seedbased synthesis performs well because the model loss is low. In contrast, when C is large, the constraint on the latent-space distribution being Gaussian is strong which improves seedless synthesis but increases model loss.

Sub-population synthesis. We illustrate an advantage of seedbased synthesis compared to seedless synthesis: it can be used produce synthetics from a sub-population (or a different population) than that used to train the model. As an example, suppose we want to synthesize records using input data with a different distribution than the training data. This is possible with seedbased synthesis. To capture this experimentally, we use a subset of the HCUP NIS discharge medical records that have *alcohol-related disorders* (CCS category `cat660`) as input data to the mechanism.[1] In this case, we expect the produced synthetics records to be similar to the input data records (as opposed to the model's training data records).

Using this input data, we produce $10'000$ synthetics using the mechanism with the VAE-3 model. The distinguishability of the produced synthetics compared to the mechanism's input data is 0.026. In contrast, the distinguishability of the input data compared to model's training data is 0.365, showing that the two distributions are very different. To further illustrate this scenario, we evaluate the following four queries on the synthetics, the model's training data ($10'000$ records), and the input data ($10'000$ records).

1. Number of records for female patients.
2. Number of records with patient outcome: death.
3. Number of records with hospital birth.
4. Number of records with elective procedure.

The results are shown in Table 9.4. We observe a significant difference between the training data (representative of the overall HCUP NIS dataset) and the input data (the sub-population of patients with alcohol-related disorders). For example, in the HCUP NIS data is about 57% of patients are female whereas, in the sub-population, only about 28% of patients are female. Further, we see that the query results for the synthetics (VAE-3) are very close to that of the sub-population. This is expected as similarity between input data records and synthetics is the point of seedbased synthesis!

Table 9.4: Queries over synthetics produced from a sub-population. In all cases, we use a sample of $10'000$ records.

	Q1 (`female`)	Q2 (`died`)	Q3 (`hospbirth`)	Q4 (`elective`)
Training	5766	191	2205	1007
Sub-pop	2787	204	979	2
VAE-3	2896	206	922	9

9.3.4 Utility Bounds

We explore utility in the sense of filtering loss using the theory developed in Chapter 5. Specifically, we follow the approach from Section 5.2.2 to quantify experimentally β_t and eventually derived bounds such as Lemma 5.2.

Recall from Chapter 5 that β_t is a quantity which is reflective of the relative sensitivity of the model. Informally, the smaller β_t, the higher the utility we can expect. We follow the methodology outlined in Section 5.2.2 and measure $\hat{\beta}_t$ according to Eq. (5.5) for varying values of t for both VAE-3 and PCA-5. We plot the results as a graph (Fig. 9.4).

[1] Note that this attribute is not one of our selected attributes (Table 9.6).

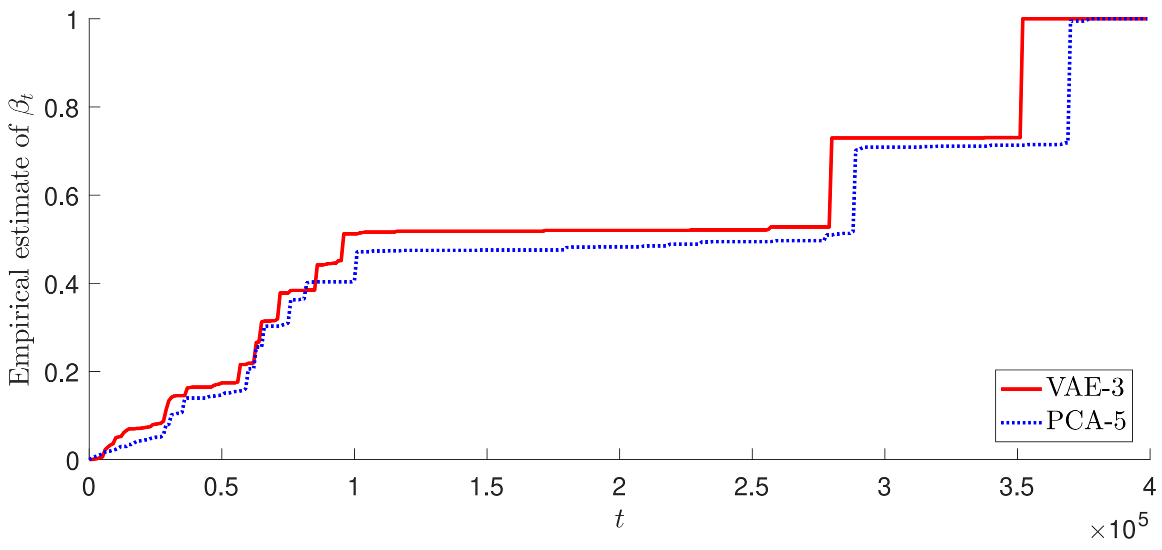


Figure 9.4: Empirical measurement of β_t for varying values of t and both VAE-3 and PCA-5. It can be seen that if we choose $t \leq 50'000$, then β_t is (relatively) small.

Table 9.5: SD bound (Lemma 5.2) for varying p1arameters t and k based on empirical measurements of β_t. We also show $\delta(t)$ and $\hat{\alpha}_t$ (Section 5.2.2). In all cases, we set $\varepsilon_0 = 0.001$.

	k	t	$\hat{\beta}_t$	$\delta(t)$	$\hat{\alpha}_t$	SD bound (Lemma 5.2)
VAE-3	1000	5000	0.006	0.991	1.015	0.0103
	5000	10000	0.049	0.997	1.055	0.0509
	10000	15000	0.070	0.997	1.079	0.0713
	20000	25000	0.080	0.997	1.091	0.0819
	50000	55000	0.174	0.997	1.214	0.1754
PCA-5	1000	5000	0.013	0.991	1.022	0.0171
	5000	10000	0.023	0.997	1.027	0.0246
	10000	15000	0.034	0.997	1.039	0.0359
	20000	25000	0.050	0.997	1.057	0.0522
	50000	55000	0.154	0.997	1.186	0.1558

Using $\hat{\beta}_t$, we can set values of k and ε_0 to obtain α_t (Section 5.2.2) and thus in turn obtain bounds on the statistical distance (Lemma 5.2). We set $\varepsilon_0 = 0.001$ and choose varying values of $t > k$. The results are shown in Table 9.5 where it can be seen that for some choices of parameters, the bounds are quite low.

Table 9.6: Selected HCUP-NIS data attributes. All attributes are binary valued.

Attribute	Description
`FEMALE`	Female indicator
`DIED`	Patient death indicator
`ELECTIVE`	Elective admission
`HOSPBRTH`	In-hospital birth
`ORPROC`	Major operating room procedure
`CAT11`	Cancer of head and neck
`CAT12`	Cancer of esophagus
`CAT13`	Cancer of stomach
`CAT14`	Cancer of colon
`CAT15`	Cancer of rectum and anus
`CAT16`	Cancer of liver and intrahepatic bile duct
`CAT17`	Cancer of pancreas
`CAT18`	Cancer of other GI organs; peritoneum
`CAT19`	Cancer of bronchus; lung
`CAT20`	Cancer; other respiratory and intrathoracic
`CAT21`	Cancer of bone and connective tissue
`CAT22`	Melanomas of skin
`CAT23`	Other non-epithelial cancer of skin
`CAT24`	Cancer of breast
`CAT25`	Cancer of uterus
`CAT26`	Cancer of cervix
`CAT27`	Cancer of ovary
`CAT28`	Cancer of other female genital organs
`CAT29`	Cancer of prostate
`CAT30`	Cancer of testis
`CAT31`	Cancer of other male genital organs
`CAT32`	Cancer of bladder
`CAT33`	Cancer of kidney and renal pelvis
`CAT34`	Cancer of other urinary organs
`CAT35`	Cancer of brain and nervous system
`CAT36`	Cancer of thyroid
`CAT37`	Hodgkins disease
`CAT38`	Non-Hodgkins lymphoma
`CAT39`	Leukemias
`CAT40`	Multiple myeloma
`CAT41`	Cancer; other and unspecified primary
`CAT42`	Secondary malignancies
`CAT43`	Malignant neoplasm without specification of site
`CAT44`	Neoplasms of unspecified nature or uncertain behavior
`CAT45`	Maintenance chemotherapy; radiotherapy
`CAT46`	Benign neoplasm of uterus
`CAT47`	Other and unspecified benign neoplasm

Chapter 10: Application: Images

In this chapter, we show how to synthesize realistic images in a privacy-preserving way.

10.1 DATASET

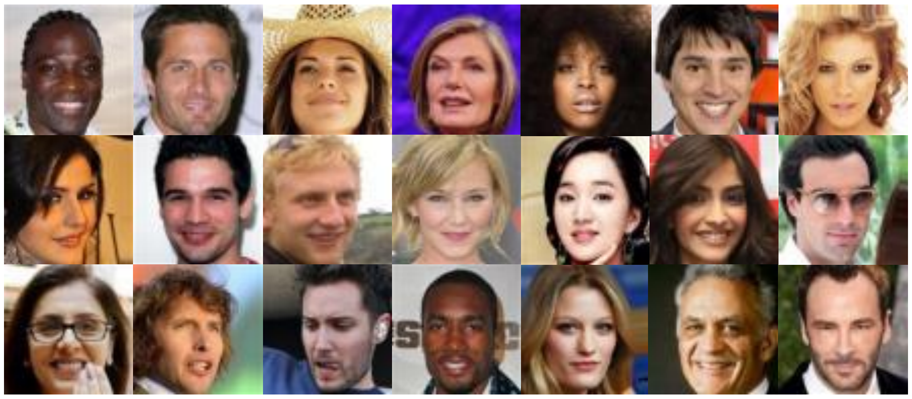


Figure 10.1: Samples from the CelebA dataset [78].

The CelebFaces Attributes Dataset (CelebA) [78] contains over 200′000 images of celebrity faces. Fig. 10.1 shows sample images from the dataset. As a pre-processing step, we ensure that all images are 64 by 64 pixels through a sequence of cropping and scaling operations. We split the resulting collection of images into two disjoint subsets of roughly equal size, and use the first part for model training and the second part as input data to the mechanism.

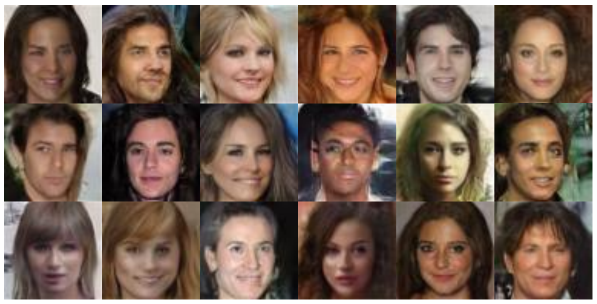
10.2 SETUP

As generative model, we use the VAE/GAN model from Larsen et al. [26] which combines a VAE with a GAN by collapsing the generator and decoder. As an implementation of the generative model, we use of the authors' code (as a black-box) which is available online [79]. We train the model using the training set for 250 epochs using the (default) parameters suggested by the authors, and use a latent width of $w = 32$.

To generate synthetics from the model, we use the latent-space noise adding technique presented in Section 6.3.1. Specifically, we add noise drawn from $\text{Gaus}(\mathbf{0}, \beta\mathbf{I})$, and we set $\beta = 0.8$. We set the parameters of the privacy test (k, t, and ε_0) in order to guarantee $(1, 2^{-40})$-differential privacy for each privacy-preserving synthetic image released. As a baseline, we also consider a seedless synthesizer based on the same model. For this we simply draw a latent space point z over its latent space distribution $\text{Gaus}(\mathbf{0}, \mathbf{I})$ and feed it to the decoder to obtain its corresponding image.

10.3 EVALUATION

To evaluate whether we can generate realistic images of faces with differential privacy, we produce a few synthetics using the mechanism. We also produce a few synthetics using the seedless baseline. The two sets are shown side-by-side in Figs. 10.2(a) and 10.2(b).


(a) Seedless

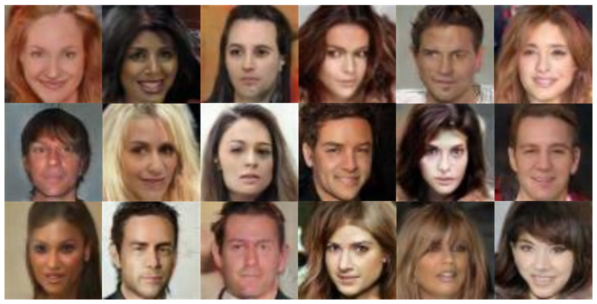
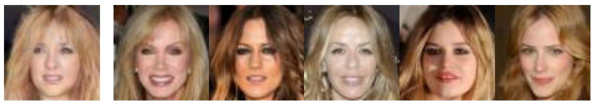

(b) Privacy-Preserving Seedbased

Figure 10.2: Synthetic faces produced using the VAE/GAN model of Larsen et al. [26].

As an example, we pick out two synthetic faces produced through the seedbased mechanism (shown in Fig. 10.2(b)). For each, we select a set of four input faces among the plausible seeds according to their synthesis probability. We then reconstruct each of these faces using the model and show them alongside their input and the synthetic in Figs. 10.3(a) and 10.3(b). It can be seen that the synthetics are similar (in terms of their facial features) to the plausible seeds (as expected). This illustrates a use case of seedbased synthesis: synthetics are similar to their seeds, but not so similar that this would violate privacy!


(a) Example 1

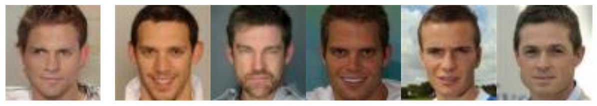

(b) Example 2

Figure 10.3: Synthesizing faces. Process explained by examples. Leftmost image: (privacy-preserving) synthetic face y. Rightmost images: (some) plausible seeds, i.e., input images with the highest probability of synthesizing the synthetic y.

One advantage of the latent-space noise adding technique over other latent-space synthesis techniques is that it allows us to further manipulate the synthetic over the latent space before decoding (Section 6.3.1). To illustrate this, we give two examples of additional latent-space processing in Fig. 10.4. The first (top row of Fig. 10.4) is sampling around the synthetic z by adding low-magnitude noise, i.e., $\text{Gaus}(\mathbf{0}, 0.25 \cdot \mathbf{I})$, to obtain different samples of the same (synthetic) face. For the second (bottom row of Fig. 10.4), we use the attributes annotation of the CelebA dataset to add the corresponding features to the (synthetic) face using latent-space vector addition.

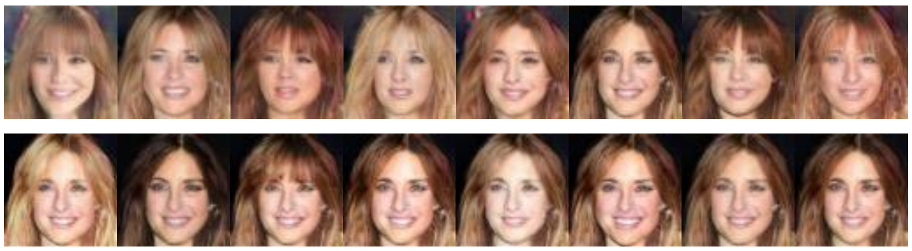


Figure 10.4: Examples of additional latent-space processing on seedbased privacy-preserving faces. Synthetic face from Fig. 10.3(a). Top row: additional samples through low-magnitude noise adding. Bottom row: vector-space operations to add attributes; (from left to right) blond hair; black hair; bangs; heavy makeup; pale skin; rosy cheeks; straight hair; young.

Chapter 11: Related Work

This chapter briefly surveys the relevant literature.

11.1 DATA ANONYMIZATION

The first systematic attempt at privacy-preserving data publishing came about with k-anonymity [1, 80, 81]. The idea behind k-anonymity is to prevent an adversary from re-identifying an individual in anonymized published data (i.e., point to that individual's record) by ensuring that identifiers present in the dataset be identical for at least k records (so that an adversary does not know which among k is the target record). The key concept is quasi-identifiers, attributes such as age, gender, or ZIP code which do not on their own uniquely identify an individual but may serve as unique identifiers when used in combination with other quasi-identifiers. The definition of k-anonymity states that there must be at least k records in the dataset for each combination of quasi-identifiers values. To address drawbacks of k-anonymity, l-diversity [82], t-closeness [83], and many other metrics [84–86] were proposed. For example, l-diversity stipulates that there must be l distinct sensitive attribute values for each combination of quasi-identifiers values. Moreover, k-anonymity, l-diversity, and t-closeness are called syntactic metrics because they declare that a dataset is safe to publish if it satisfies a specific syntactic condition.

Syntactic metrics are popular due to their ease of use. But privacy researchers have expressed concerns [84, 86–90]. A particular concern is that syntactic metrics are not robust to adversarial background knowledge [91]. Nevertheless, syntactic metrics are often used for many applications that prioritize utility and practicality. For example, [92–96] aim to achieve k-anonymity in the context of location privacy. Similarly, k-anonymity has also been used for privacy-preserving data mining of query logs [97, 98], to anonymize social network data [99], and for data publishing of medical data [100–103].

11.2 DIFFERENTIAL PRIVACY

Though there has been work on privacy in statistical databases for decades [104], a series of work [12, 28, 105, 106] in the early 2000s led to the definition of differential privacy [28]. Since then, the case for differential privacy has been strengthen [29, 31, 107–112] and there is now a vast surrounding literature [75, 113–117]. Though it is has been criticized [118, 119], a major advantage of differential privacy over other privacy notions is that is makes (almost) no assumption on adversary background knowledge.

There has been several attempts at relaxing and/or generalizing differential privacy. This has led to notions such as crowd-blending privacy [120], local privacy [121], adversarial privacy [122], random differential privacy [123], computational differential privacy [124], and outlier privacy [125]. In addition, some related work propose full-fledged data privacy frameworks of which differential privacy is an instance under some conditions. This is the case for example for pufferfish [126], coupled-worlds privacy [127], and membership privacy [30].

Substantial efforts have been made to connect popular privacy notions. For example, Li et al. [128] show a connection between k-anonymity and differential privacy through the sub-sampling of the input data records. This technique can be used to reduce the privacy budget of a mechanism that already satisfies differential privacy [129]. In addition, it allows mechanisms guaranteeing weaker notions of privacy to satisfy differential privacy [120, 127].

Differential privacy is most commonly achieved through a mechanism which adds noise to the output of a query on a database. In this case, the noise distribution must be calibrated to the sensitivity of the query [28]. As a result, the concept of sensitivity which measures the query's output worst-case change for any two neighboring databases, is a central concept in the design of mechanisms. But the sensitivity of a query for databases pairs that are neighbors of the input database, called the local sensitivity, is often much lower than the global sensitivity. Unfortunately, calibrating the noise to the local sensitivity is insufficient to achieve differential privacy, as the sensitivity itself may leak information. This observation was made by Nissim, Raskhodnikova, and Smith [108] and led the authors to propose the notion of smooth sensitivity.

If noise is properly calibrated to the global sensitivity or the smooth sensitivity of the query, many distributions of noise are acceptable. For example, instead of using the popular Laplace distribution, adding Gaussian noise with zero mean and appropriate variance leads to (ε, δ)-differential privacy [29, 108]. For queries which guarantee integer-valued responses, adding noise from the symmetric geometric distribution is possible [130]. Another example is the staircase mechanism which uses a geometric mixture of uniform random variables [131, 132].

There is a broad range of techniques that yield differential privacy (not only noise adding). For example, [129] points out that posterior sampling achieves differential privacy "for free" if the log-likelihood is bounded. More generally, a variety of different techniques such as the sparse-vector technique [29, 133, 134] or the exponential mechanism of McSherry and Talwar [31] can be used to achieve differential privacy. In fact, the exponential mechanism is often used as a building block in other mechanisms that seek to achieve differential privacy.

There has been work on composition of differential privacy showing that the privacy budget need not increase linearly with the number of compositions [135]. More recent work such as Kairouz et al. [35] has further improved the composition results. Unfortunately, as pointed out by Murtagh and Vadhan in [136] computing the privacy budget under the best composition result is difficult in practice. That said, Meiser and Esfandiar [36] recently proposed an efficient technique to numerically approximate the privacy budget of differential privacy under composition.

The closest idea in the literature to the idea of privacy tests (Chapter 4) is the Propose-Test-Release (PTR) framework of Dwork and Lei [137] which guarantee differential privacy by testing candidate bounds for the sensitivity. If a suitable bound is found, then a noisy answer with noise calibrated to it is produced. Otherwise, the output is "no answer" ($\perp$). For example, PTR can be used to calculate the median of a dataset when the scale of the data is not known a priori. Though not designated as such, an early example of privacy test using the noisy threshold technique (Chapter 3) is used by Korolova et al. [138] to produce web search logs with differential privacy.

11.3 SYNTHETIC DATA

Synthetic data is used in various applications as a substitute for real data when the latter is hard to obtain. For example, it has been used for software testing [139, 140], testing an OCR system [141], and performance evaluation in the context of database systems [142]. In particular, database systems benchmarks sometimes rely on synthetic data [143–145].

As Rubin [11] proposed in 1993, synthetic data can also be used for privacy. The idea of Rubin is to use multiple imputation to create synthetic datasets. Multiple imputation is the repeated use of a function that proposes values for missing fields in a record. This and subsequent work in releasing synthetic data such as [146–149] have attracted significant interest from practitioners. For example, some of the techniques have been adopted by the U.S. Census Bureau [150–152].

Despite some negative hardness results such as [153, 154], there is a large body of work on releasing datasets privately. Blum et al. [155] show that it is possible (in theory) to create synthetic datasets that are useful for some queries. There are also more practical techniques achieving differential privacy. Abowd and Vilhuber [156] propose a technique for count data, Machanavajjhala et al. [157] sample from a dirichlet-multinomial distribution, Wasserman and Zhou [158] explore different techniques that sample from histogram in a differentially-private way. These and similar mechanisms have been analyzed in an effort to understand their utility [159, 160]. Li et al. [116] propose a generative technique based on Copula functions. Xu et al. [161] use a random projection technique to synthesize data. Chanyaswad et al. [55] use a similar idea based on principal component analysis. Liu [162] proposes a technique called model-based data synthesis. All of these are example of private model learning: privacy is achieved by learning a model from data privately. A notable example is PrivBayes [163] which constructs a differentially-private generative model based on Bayesian networks. Their construction is similar to that of Section 6.2, except seedless.

A notable exception to this private model learning paradigm is the multiplicative weights algorithm [164] which describes a way to efficiently release synthetic data with differential privacy, provided one has a set of linear queries. The algorithm starts with a uniform sample as synthetic dataset and iteratively improves it by scaling up or down the contribution of each synthetic record according to that record's influence on the set queries. The exponential mechanism is used internally for each iteration as a way to select a query among the set, in a differential private way.

There is also a large collection of work [117, 165–168] which shows how to release histograms and contingency tables in a differentially-private way. Note that techniques producing contingency tables, histograms or other low-dimension synopses do not necessarily fall under our definition of data synthesis as they may not be equivalent to producing *full* data records or preserving the input data format. For example, an algorithm producing 2D-histograms does not qualify if the input data records have more than two attributes.

The rise in popularity of neural networks and deep learning has led to the design of techniques such as Variational Auto Encoders (VAE) [14] and Generative Adversarial Networks (GANs) [15]. These and other techniques have been used to generate synthetic data of many types such as speech [16, 169], music [17], text and handwriting [20], textures [21], images, and 3D-shapes [170], as well as for many applications such as face rotation [18], super resolution [171–173], scene completion [174], generating time series for medical applications [175], forecasting from a static image [19], creating images from a label [22, 176, 177], and translating one image into another [23].

A key feature distinguishing these generative models from graphical models and more traditional probabilistic models is *not* only that they are based on a neural-network architecture. Rather, it is that their output is typically obtained through some form of conditioning: the output is obtained through sampling the model conditional on some input. In particular, Mirza and Osindero introduced the concept of Conditional Generative Adversarial Networks [178]. Conditioning can be viewed as a form of seedbased synthesis.

There is a collection of papers describing techniques to train neural networks (e.g., using gradient descent) with differential privacy [179–182]. These techniques can make neural networks private, but not the application of such neural networks to synthesize data with conditioning. In addition, there has been recent attention on making generative neural networks differentially private [183–185]. These techniques fall within the seedless category. In contrast, this thesis proposes a framework for seedbased data synthesis.

Chapter 12: Discussion

This chapter discusses limitations of the framework and avenues for future research.

12.1 LIMITATIONS AND FUTURE RESEARCH

Synthesizing multiple records per invocation. The seedbased mechanism proposed in this thesis has the special form of selecting a single seed to produce a single synthetic record. But one could take a broader view on seedbased synthesis. For example, one could design a mechanism that uses multiple seeds to produce multiples synthetic records (per invocation). Ultimately, synthesizing $m > 1$ records at once may yield superior privacy guarantees than synthesizing a single record (per invocation) and composing the process m times.

Dependent data records. A synthetic dataset is obtained by taking i.i.d. samples from the seedbased mechanism which ignores statistical relationships between data records. As a result, some properties of the input data are not preserved. For example, in the case of facial images, if the input data contains a certain number of identical twins, this will not (in general) be reflected in the produced synthetic dataset. Preserving such properties is typically not an important requirement for many applications and data types. But, it may be a desirable goal when synthesizing complex data types such as graphs.

Distinguishing game. When quantifying distinguishability experimentally, the difficulty of the distinguishing game depends on the amount of training data given to the classifier. Thus, the classifier should be trained using as much data as possible to reduce the chance of underestimating distinguishability. The limitation is that while one can produce as many synthetics as desired for the purpose of estimating distinguishability, the number of real records available is finite. Further research could explore the relationship between training dataset size and accuracy of the resulting distinguishability estimate.

Validating tentative conclusions. A major obstacle to the deployment of privacy-preserving data analysis techniques, including data synthesis, is providing ways to validate tentative conclusions. Indeed, whether one uses a statistical database with queries performed through a differential privacy process, or privacy-preserving seedbased data synthesis; an analyst will make tentative conclusions on the basis of noisy or distorted data or queries answers. Future work could explore ways to validate such tentative conclusions, for example through a privacy-preserving protocol which would check whether the same conclusions hold of the real (input) dataset.

Navigating the landscape of generative models. The framework presented in this thesis assumes we are given a generative model as a black box. How can we instantiate accurate seedbased generative models or customize a model to a specific application? This is a question that future research should investigate.

A possible starting point is to leverage the distinguishing game methodology to iteratively improve a generative model through a feedback loop: given a generative model, one uses the distinguishing game to experimentally quantify its model loss; then, based on the kind of synthetic records that are easily distinguished from real records, one refines the model by tweaking its parameters or hyperparameters. Application-specific utility requirements may be incorporated in this process if they can be formulated as a set of adversaries capable of playing the distinguishing game.

Privacy testing. The idea of using a privacy test as a technique to achieve provable privacy guarantees is worthy of further exploration. And it may be of independent interest for data privacy outside of the context of data synthesis. We conjecture that privacy tests which act as probabilistic filters could be used in place of noise-adding in other contexts such as for statistical queries or aggregation.

12.2 Q&A

This section addresses, in a question-and-answer format, common concerns and criticism about the framework and the idea of seedbased synthesis.

- *Can one really analyze data that is noisy or synthetic? What if this yields a wrong answer?*

Real-world datasets are messy. It is a characteristic of big data to have to deal with messy and incomplete datasets [13]. We are used to dealing with uncertainty when analyzing data. This is why statistical tools like confidence intervals have been developed. What matters is that we understand how the synthetic data is generated and how it differs from the original (real) data.

Given the distortion induced by all of the techniques that provide strong privacy guarantees (including ours) we do not advocate that synthetic datasets be used in a context where accuracy is critical such as for clinical decision making. Nevertheless, we believe that synthetic data can be an asset for scientific research or data analysis tasks, especially early-stage data exploration. We envision our technique to be used in scenarios where synthetic data is easy to obtain so that it can be used by an analyst to make tentative findings which can later be verified on real data (perhaps in a privacy-preserving way).

- *How do you know that the produced synthetic data will be useful?*

  We cannot provide utility guarantees that hold regardless of the generative model and of the input data because there are pairs of generative models and datasets for which no privacy-preserving technique can also provide utility. This does not mean that we can say nothing about utility. Indeed, as we show in Chapter 5, the framework allows us to derive utility bounds in certain situations and make predictions about the utility of privacy-preserving synthetics produced from a given generative model and input dataset.

  For data sharing, in general, even without any privacy constraints, there are no guarantees that a particular dataset will be useful for a specific analysis task.

- *Why not use traditional techniques to achieve differential privacy like the Laplace mechanism or Exponential mechanism?*

  These mechanisms simply do not apply to synthetic data. For instance, the Laplace mechanism works for real-valued output, not arbitrary data records. The Exponential mechanism can be used to generate synthetic data but the process is intractable when the data universe is large. Real-world data is often complex, sparse, and high dimension.

  There is prior work which proposes techniques to produce synthetic datasets with privacy guarantees (see Section 11.3). All of them are seedless. This thesis proposes the first framework for privacy-preserving seedbased data synthesis.

- *Why is seedbased data synthesis useful? Why not do seedless instead?*

  Seedless is a special case of seedbased synthesis. We propose seedbased data synthesis to bridge a gap between the kinds of techniques for which we know how to achieve privacy (and differential privacy in particular) and the state-of-the-art applications which produce synthetic data using generative models based on neural networks through conditioning.

  Seedbased synthesis is particularly useful if one wishes to sample records from a population different than that used to train a model as we illustrate experimentally in Chapter 9.

- *With seedbased data synthesis, you can only generate a finite set of synthetic records before you exhaust the privacy budget. What if that's not enough?*

  The number of synthetics required is dependent on the application. For some applications, only a few synthetic records may be needed. Each synthetic produced through a seedbased process increases the privacy budget because it bring new information (from its seed). In contrast, one can sample an arbitrarily large synthetic dataset from a seedless generative model, but the information is finite and bounded to whatever is captured in the model itself.

Chapter 13: Conclusions

This thesis proposes a new approach to sharing sensitive datasets through data synthesis. We describe, discuss, and experimentally evaluate a framework for privacy-preserving seedbased synthesis. In contrast to prior work, the framework has two distinctive novel features: (1) it produces synthetic records from probabilistic generative models through a form of conditioning, and (2) it achieves meaningful privacy guarantees such as differential privacy using privacy tests which are algorithms that probabilistically reject outputs deemed to leak sensitive information. We validate the framework experimentally use five different generative models to synthesize records of four different data types: location traces, census microdata, medical records, and facial images. We show that the framework can produce highly realistic synthetic output and we believe the experimental results provide compelling preliminary evidence of the viability of seedbased data synthesis in certain scenarios.

This work is a step towards the design of a practical technique for privacy-preserving data synthesis. We hope that this work will spur further research in this direction. And that the idea of privacy tests will be of interest beyond data synthesis.

Appendix A: Proofs

A.1 PROPERTIES OF THE PRIVACY TEST

The noisy threshold test (Privacy Test 3.2) is well-behaved (Definition 3.2) for the Laplace distribution and the symmetric geometric distribution. Indeed, let δ denote Privacy Test 3.2 function, and let $k > 0$ and $\varepsilon_0 > 0$ be parameters. Then: $\delta(x) = \Pr\{\mathbf{Z} \geq k - x\}$, where $\mathbf{Z}$ has distribution $\text{Lap}(\frac{1}{\varepsilon_0})$ for the Laplace case, or distribution $\text{Geom}(\alpha)$ with $\alpha = e^{-\varepsilon_0}$ for the symmetric geometric case.

In both cases, observe that:

$$1 \leq \frac{\Pr\{\mathbf{Z} < k - x)\}}{\Pr\{\mathbf{Z} < k - (x+1)\}} \leq e^{\varepsilon_0} \quad \text{and} \quad 1 \leq \frac{\Pr\{\mathbf{Z} \geq k - (x+1)\}}{\Pr\{\mathbf{Z} \geq k - x\}} \leq e^{\varepsilon_0} \tag{A.1}$$

In other words: $1 \leq \frac{\delta(x+1)}{\delta(x)} \leq e^{\varepsilon_0}$ and $1 \leq \frac{1-\delta(x)}{1-\delta(x+1)} \leq e^{\varepsilon_0}$ which shows that the test is well-behaved for $c_0 = 1$ and $\beta_0 = e^{\varepsilon_0}$.

This observation has previously been used in related work such as Korolova et al. [138].

A.2 SENSITIVITY OF ENTROPY

Lemma A.1 (Sensitivity of H).
Let $\mathbf{z}$ *be a discrete random variable with a probability distribution estimated from* $n \geq 1$ *data records. The sensitivity of* $\mathrm{H}(\mathbf{z})$ *is:*

$$\Delta_H \leq \frac{1}{n}[2 + \frac{1}{\ln(2)} + 2\log_2 n] \ .$$

Proof. Let $\mathbf{z}$ and $\mathbf{z}'$ be the random variables associated with two histograms (of dimension m) computed from two neighboring datasets D and D', respectively. Both datasets have n records but differ in exactly one record.

Let $z_c = (c_1, c_2, \ldots, c_m)$ and $z'_c = (c'_1, c'_2, \ldots, c'_m)$ represent the histograms over the m values of the attribute for $\mathbf{z}$ and $\mathbf{z}'$, respectively. The entropy is computed over the probability distribution represented by a histogram.

Remark that the histograms of the considered neighboring datasets D and D' can only differ in two positions. If they do not differ in any position, then the Lemma trivially holds ($\Delta_H = 0$). That is, without loss of generality, there exists j_1 and j_2 (with $j_1 \neq j_2$) such that

$c'_{j_1} = c_{j_1} + 1$ and $c'_{j_2} = c_{j_2} - 1$. Also, $n - 1 \geq c_{j_1} \geq 0$ which means that $n \geq c'_{j_1} \geq 1$, and $n \geq c_{j_2} \geq 1$ which means that $n - 1 \geq c'_{j_2} \geq 0$. Furthermore for $i \neq j_1, j_2$, we have $c'_i = c_i$, and also:

$$\sum_{i=1}^{m} c_i = \sum_{i=1}^{m} c'_i = n \ .$$

Now:

$$\begin{aligned}
\mathrm{H}(\mathbf{z}) &= -\sum_{i=1}^{m} \frac{c_i}{n} \log_2 \frac{c_i}{n} \\
&= -\frac{1}{n} \left[\sum_{i=1}^{m} c_i \log_2 c_i - n \log_2 n \right] \\
&= \log_2 n - \frac{1}{n} \left(c_{j_1} \log_2 c_{j_1} + c_{j_2} \log_2 c_{j_2} + \sum_{i \neq j_1, j_2} c_i \log_2 c_i \right) \ .
\end{aligned}$$

Similarly,

$$\begin{aligned}
\mathrm{H}(\mathbf{z}') = \log_2 n &- \frac{1}{n} \sum_{i \neq j_1, j_2} c_i \log_2 c_i \\
&- \frac{1}{n} \left[(c_{j_1} + 1) \log_2 (c_{j_1} + 1) + (c_{j_2} - 1) \log_2 (c_{j_2} - 1) \right] \ .
\end{aligned}$$

We have that $\Delta_H = \max_{c_{j_1}, c_{j_2}} |\mathrm{H}(\mathbf{z}) - \mathrm{H}(\mathbf{z}')|$. We analyze this quantity with respect to the values of c_{j_1} and c_{j_2} to show that the lemma holds in each case.

Observe that:

$$\begin{aligned}
\Delta_H &= |\mathrm{H}(\mathbf{z}) - \mathrm{H}(\mathbf{z}')| \\
&= \frac{1}{n} |c_{j_1} \log_2 c_{j_1} - (c_{j_1} + 1) \log_2 (c_{j_1} + 1) \\
&+ c_{j_2} \log_2 c_{j_2} - (c_{j_2} - 1) \log_2 (c_{j_2} - 1)| \ .
\end{aligned}$$

- Case 1: $c_{j_1} = 0$. We have

$$\Delta_H = \frac{1}{n} |c_{j_2} \log_2 c_{j_2} - (c_{j_2} - 1) \log_2 (c_{j_2} - 1)| \ .$$

Clearly, if $c_{j_2} = 1$ then $\Delta_H = 0$ and the Lemma trivially holds. So assume $c_{j_2} > 1$. We have:

$$\begin{aligned}
\Delta_H &= \frac{1}{n}|c_{j_2}\log_2 c_{j_2} - (c_{j_2}-1)\log_2(c_{j_2}-1)| \\
&= \frac{1}{n}\left|c_{j_2}\log_2\left(\frac{c_{j_2}}{c_{j_2}-1}\right) + \log_2(c_{j_2}-1)\right| \\
&\leq \frac{1}{n}\left|c_{j_2}\log_2\left(\frac{c_{j_2}}{c_{j_2}-1}\right)\right| + \frac{1}{n}\log_2(c_{j_2}-1) \\
&\leq \frac{1}{n}\log_2(n-1) + \frac{1}{n}\left|(a+1)\log_2\left(1+\frac{1}{a}\right)\right| \ ,
\end{aligned}$$

where $a = c_{j_2} - 1 \geq 1$. It is easy to see that $(a+1)\log_2(1+\frac{1}{a}) \leq 2$. We conclude that: $\Delta_H \leq \frac{1}{n}(2 + \log_2 n)$.

- Case 2: $c_{j_2} = 1$. We have

$$\Delta_H = \frac{1}{n}|c_{j_1}\log_2 c_{j_1} - (c_{j_1}+1)\log_2(c_{j_1}+1)| \ .$$

Again if $c_{j_1} = 0$, then the Lemma trivially holds. So assume $c_{j_1} > 0$. We have:

$$\begin{aligned}
\Delta_H &= \frac{1}{n}|c_{j_1}\log_2 c_{j_1} - (c_{j_1}+1)\log_2(c_{j_1}+1)| \\
&= \frac{1}{n}\left|c_{j_1}\log_2(\frac{c_{j_1}}{c_{j_1}+1}) - \log_2(c_{j_1}+1)\right| \\
&\leq \frac{\log_2 n}{n} + \frac{1}{n}\left|c_{j_1}\log_2(\frac{c_{j_1}}{c_{j_1}+1})\right| \\
&= \frac{\log_2 n}{n} + \frac{1}{n}c_{j_1}\log_2(\frac{c_{j_1}+1}{c_{j_1}}) \\
&= \frac{\log_2 n}{n} + \frac{1}{n}c_{j_1}\log_2\left(1+\frac{1}{c_{j_1}}\right) \ .
\end{aligned}$$

Using L'Hopital's rule we have $c_{j_1}\log_2\left(1+\frac{1}{c_{j_1}}\right) \leq \frac{1}{\ln 2}$. We conclude that: $\Delta_H \leq \frac{1}{n}(\frac{1}{\ln 2} + \log_2 n)$.

- Case 3: $c_{j_1} \geq 1$, $c_{j_2} \geq 2$. We have:

$$\begin{aligned}\Delta_H &= \frac{1}{n}|c_{j_1}\log_2 c_{j_1} - (c_{j_1}+1)\log_2(c_{j_1}+1) \\ &+ c_{j_2}\log_2 c_{j_2} - (c_{j_2}-1)\log_2(c_{j_2}-1)| \\ &\leq \frac{1}{n}|c_{j_1}\log_2 c_{j_1} - (c_{j_1}+1)\log_2(c_{j_1}+1)| \\ &+ \frac{1}{n}|c_{j_2}\log_2 c_{j_2} - (c_{j_2}-1)\log_2(c_{j_2}-1)| \ ,\end{aligned}$$

  where it is seen that the two terms have been bounded for cases 1 and 2. Thus, putting it all together, we conclude that $\Delta_H \leq \frac{1}{n}\left(2 + \frac{1}{\ln 2} + 2\log_2 n\right)$.

□

A.3 PLAUSIBLE DENIABILITY AND DIFFERENTIAL PRIVACY

In this section, we prove Theorem 4.1.

Theorem 4.1. *Let $\mathcal{F}$ denote Mechanism 3.1 with Privacy Test 4.2 and parameters $k \geq 1$, $r > 1$, and $\varepsilon_0 > 0$. For any neighboring datasets D and D' such that $|D|, |D'| \geq k$, any set of outcomes $Y \subseteq U$, and any integer $1 \leq t < k$, we have:*

$$\Pr\{\mathcal{F}(D') \in Y\} \leq e^{\varepsilon}\Pr\{\mathcal{F}(D) \in Y\} + \delta \ ,$$

for $\delta = e^{-\varepsilon_0(k-t)}$ and $\varepsilon = \varepsilon_0 + \ln(1 + \frac{r}{t})$.

Let D and D' denote two neighboring datasets, i.e., either $D = D' \cup \{d\}$ for some $d \in U$, or $D' = D \cup \{d'\}$ for some $d' \in U$. Assume that D and D' have at least k records, and we have parameters $k \geq 1$, $r > 1$, $\varepsilon_0 > 0$. For convenience we write $p_d(y) = \Pr\{y \leftarrow \mathcal{M}(d)\}$, (referring to $\mathcal{M}$ only implicitly).

Given a dataset $D^\star$, we want to reason about the probability that synthetic record y is released: $\Pr\{y \leftarrow \mathcal{F}(D^\star)\}$. Observe that given synthetic record $y \in U$, the records of $D^\star$ can be partitioned (into disjoint sets) by the privacy criterion. Concretely, let $I_d(y)$ be the partition number of a record $d \in D^\star$ with respect to y. The *partition number* $I_d(y)$ is the unique non-negative integer such that $r^{-(I_d(y)+1)} < p_d(y) \leq r^{-I_d(y)}$. In other words, $I_d(y) = \lfloor -\log_r p_d(y) \rfloor$. If $p_d(y) = 0$ then the partition number is undefined. Similarly, we define the *partition* (or partition set) for $i \geq 0$ as $C_i(D^\star, y) = \{d : d \in D^\star, I_d(y) = i\}$. That is, partition i is the set of records with partition number i.

A key step is to express $\Pr\{y \leftarrow \mathcal{F}(D^\star)\}$ in terms of: (1) the probability of generating y from a specific partition (i.e., the seed is in that partition) and (2) the probability of passing the test. For (2) remark that the probability of passing the privacy test depends only on the partition of the seed (see Privacy Test 4.2).

Observation A.1. *For any dataset $D^\star$, if the seed is in partition i, the probability of passing the privacy test is given by: $\delta(D^\star, i, y) = \Pr\{\mathbf{Z} \geq k - |C_i(D^\star, y)|\}$, where $\mathbf{Z} \sim \mathrm{Lap}(\frac{1}{\varepsilon_0})$.*

Observation A.2. *For any dataset $D^\star$, the probability of producing y from partition i is:*

$$q(D^\star, i, y) = \delta(D^\star, i, y) \sum_{s \in C_i(D^\star, y)} p_s(y) \ .$$

The following expresses $\Pr\{y \leftarrow \mathcal{F}(D^\star)\}$ in terms of (1) and (2).

Lemma A.2. *For any dataset $D^\star$ and any synthetic record $y \in U$ we have:*

$$\Pr\{y \leftarrow \mathcal{F}(D^\star)\} = \frac{1}{|D^\star|} \sum_{i \geq 0} q(D^\star, i, y) \ . \tag{A.2}$$

In other words, the probability of releasing y (from $D^\star$) can be expressed as the sum, over all partitions, of the probability of generating y from a given partition and then releasing it.

Proof of Lemma A.2. Fixing a y and following the description of Mechanism 3.1, we have:

$$\Pr\{y \leftarrow \mathcal{F}(D^\star)\} = \frac{1}{|D^\star|} \sum_{s \in D^\star} p_s(y)\ \delta(D^\star, s, y) \ ,$$

Further, observe that terms of the sum for which $p_s(y) = 0$ can be omitted and that we can partition the set $\{d : d \in D^\star, p_d(y) > 0\}$ with respect to the partition number of its elements. That is:

$$\{d : d \in D^\star, p_d(y) > 0\} = \cup_{i \geq 0} C_i(D^\star, y) \ ,$$

where for each $d \in D^\star$ such that $p_d(y) > 0$, there exists a unique non-negative integer j such that $d \in C_j(D^\star, y)$.

Thus:

$$\begin{aligned}
&\Pr\{y \leftarrow \mathcal{F}(D^\star)\} \\
&= \frac{1}{|D^\star|} \sum_{s \in D^\star} p_s(y)\ \delta(D^\star, s, y) \\
&= \frac{1}{|D^\star|} \sum_{i \geq 0} \sum_{s \in C_i(D^\star, y)} p_s(y)\ \delta(D^\star, s, y) \\
&= \frac{1}{|D^\star|} \sum_{i \geq 0} \Pr\{\mathbf{Z} \geq k - |C_i(D^\star, y)|\} \sum_{s \in C_i(D^\star, y)} p_s(y) \\
&= \frac{1}{|D^\star|} \sum_{i \geq 0} q(D^\star, i, y)\ ,
\end{aligned}$$

given that the privacy score depends only on the partition of the seed (and not on the seed itself). (See the description of Privacy Test 4.2 with $\mathbf{Z}$ drawn from $\text{Lap}(\frac{1}{\varepsilon_0})$.) $\square$

Note that $q(D^\star, i, y) = 0$ if and only if $C_i(D^\star, y) = \emptyset$. Also, observe that if a record is added to or subtracted from $D^\star$ then only one partition changes. As a result, we can analyze case-by-case the change in the probability of releasing y from partition i, when adding or removing a record to partition i.

The following is a consequence of the fact that the privacy test is well-behaved. See Appendix A.1 and Eq. (A.1).

Lemma A.3. *Given any $y \in U$, any neighboring datasets D and D' such that $D' = D \cup \{d'\}$. For any partition i we have:*

$$\delta(D, i, y) \leq \delta(D', i, y) \leq e^{\varepsilon_0} \delta(D, i, y)\ .$$

Proof of Lemma A.3. There are two cases: $i = I_{d'}(y)$ or $i \neq I_{d'}(y)$. If $i = I_{d'}(y)$ then d' falls into partition i and so $C_i(D', y) = C_i(D, y) \cup \{d'\}$. We have:

$$\begin{aligned}
\delta(D', i, y) &= \Pr\{\mathbf{Z} \geq k - |C_i(D', y)|\} \\
&\leq e^{\varepsilon_0} \Pr\{\mathbf{Z} \geq k - |C_i(D', y)| + 1\} \\
&= e^{\varepsilon_0} \Pr\{\mathbf{Z} \geq k - |C_i(D, y)|\} = e^{\varepsilon_0} \delta(D, i, y)\ ,
\end{aligned}$$

Also, we have that: $\delta(D', i, y) > \delta(D, i, y)$.

Otherwise, if $i \neq I_{d'}(y)$ then $C_i(D', y) = C_i(D, y)$, and so: $\delta(D', i, y) = \delta(D, i, y)$. $\square$

To quantify the change in $q(D^\star, i, y)$ due to adding a record to partition i we need to separate two cases: (1) the partition is initially empty (or more generally has initially less than t records) and (2) the partition is not empty (or more generally has at least t records).

Lemma A.4. *For any $y \in U$ and any dataset D. Let $D' = D \cup \{d'\}$ for some $d' \in U$. Let j be the partition number of d' (i.e., $I_d(y) = j$). The following holds.*

(a) For all $i \neq j$, we have $q(D', i, y) = q(D, i, y)$.

(b) If $|C_j(D, y)| < t$:

$$q(D, j, y) < q(D', j, y) \ ,$$

and

$$q(D', j, y) \leq e^{-\varepsilon_0(k-t)} \sum_{s \in C_j(D', y)} p_s(y) \leq t \ e^{-\varepsilon_0(k-t)} \ .$$

If $|C_j(D, y)| \geq t$:

$$\frac{q(D', j, y)}{q(D, j, y)} \leq e^{\varepsilon_0} \left[1 + \frac{r}{t}\right] \ .$$

Corollary A.1 (of Lemma A.4)**.** *For any $y \in U$ and any dataset D. Let $D' = D \cup \{d'\}$ for some $d' \in U$. We have $q(D, i, y) \leq q(D', i, y)$, for all $i \geq 0$.*

Proof of Lemma A.4. Fix y and let j be the partition that d' falls into.

For part (a), note that for $i \neq j$: $C_i(D, y) = C_i(D', y)$. So $q(D, i, y) = q(D', i, y)$.

For part (b), we have that:

$$\begin{aligned}
q(D, j, y) &= \delta(D, j, y) \sum_{s \in C_j(D, y)} p_s(y) \\
&< \delta(D, j, y) \left[\sum_{s \in C_j(D, y)} p_s(y) + p_{d'}(y) \right] \\
&= \delta(D, j, y) \sum_{s \in C_j(D', y)} p_s(y) \\
&\leq \delta(D', j, y) \sum_{s \in C_j(D', y)} p_s(y) = q(D', j, y) \ ,
\end{aligned}$$

given that $p_{d'}(y) > 0$ and $\delta(D', j, y) \geq \delta(D, j, y)$ (Lemma A.3).

Now, if $|C_j(D,y)| < t$, then:

$$\begin{aligned} q(D',j,y) &= \delta(D',j,y) \sum_{s \in C_j(D',y)} p_s(y) \\ &\leq e^{-\varepsilon_0(k-t)} \sum_{s \in C_j(D',y)} p_s(y) \\ &\leq t\ e^{-\varepsilon_0(k-t)}\ , \end{aligned}$$

given that $|C_j(D',y)| \leq t$ and $p_d(y) \leq 1$ for any d. Here, the first inequality follows from the fact that $\delta(D',j,y) = \Pr\{\mathbf{Z} \geq k - |C_j(D',y)|\} \leq \Pr\{\mathbf{Z} \geq k-t\} = \frac{1}{2}e^{-\varepsilon_0(k-t)}$.

If $|C_j(D,y)| \geq t$, we have:

$$\begin{aligned} q(D',j,y) &= \delta(D',j,y) \sum_{s \in C_j(D',y)} p_s(y) \\ &= \delta(D',j,y)\ [\sum_{s \in C_j(D,y)} p_s(y) + p_{d'}(y)] \\ &\leq e^{\varepsilon_0}\ \delta(D,j,y)\ [\sum_{s \in C_j(D,y)} p_s(y) + p_{d'}(y)] \\ &\leq e^{\varepsilon_0}\ [1 + \frac{r}{t}]\ \delta(D,j,y) \sum_{s \in C_j(D,y)} p_s(y) \\ &= e^{\varepsilon_0}\ [1 + \frac{r}{t}]\ q(D,j,y)\ , \end{aligned}$$

given Lemma A.3 and the fact that $p_{d'}(y) \leq r\ p_s(y)$ for any $s \in C_j(D,y)$ and so $p_{d'}(y) \leq \frac{r}{t} \sum_{s \in C_j(D,y)} p_s(y)$. □

The following is the core result underlying Theorem 4.1.

Lemma A.5. *Let $\mathcal{F}$ denote Mechanism 3.1 with Privacy Test 4.2 and parameters $k \geq 1$, $r > 1$, and $\varepsilon_0 > 0$. Take any dataset D with $|D| \geq k$ and let $D' = D \cup \{d'\}$ for any $d' \in U$. Then for any integer $1 \leq t < k$ and synthetic record $y \in U$, we have:*

$$\Pr\{y \leftarrow \mathcal{F}(D)\} \leq e^{\varepsilon} \Pr\{y \leftarrow \mathcal{F}(D')\}\ ,$$

and

$$\Pr\{y \leftarrow \mathcal{F}(D')\} \leq e^{\varepsilon} \Pr\{y \leftarrow \mathcal{F}(D)\} + \delta\ ,$$

where $\delta = \delta(D', d', y) \leq e^{-\varepsilon_0(k-t)}$ *and* $\varepsilon = \varepsilon_0 + \ln(1 + \frac{r}{t})$*. Here,*

$$\delta(D', d', y) = e^{-\varepsilon_0(k-t)}|D'|^{-1} \sum_{s \in C_j(D', y)} p_s(y) \ ,$$

with $j = I_{d'}(y)$*.*

Proof of Lemma A.5. Fix an arbitrary synthetic record $y \in U$ and an arbitrary dataset D with $|D| \geq k$. Let $D' = D \cup \{d'\}$ for some arbitrary $d' \in U$. Applying Lemma A.2 to D we have: $\Pr\{y \leftarrow \mathcal{F}(D)\} = \frac{1}{|D|}\sum_{i \geq 0} q(D, i, y)$. Also, from Corollary A.1 we have $q(D, i, y) \leq q(D', i, y)$ for all i. Thus:

$$\begin{aligned}
\Pr\{y \leftarrow \mathcal{F}(D)\} &= \frac{1}{|D|}\sum_{i \geq 0} q(D, i, y) \\
&\leq \frac{1}{|D|}\sum_{i \geq 0} q(D', i, y) = \frac{|D'|}{|D|}\Pr\{y \leftarrow \mathcal{F}(D')\} \\
&\leq \left(1 + \frac{1}{k}\right) \Pr\{y \leftarrow \mathcal{F}(D')\} \ .
\end{aligned}$$

Observe that since (by assumption) $r > 1$ and $1 \leq t \leq k$, we have: $\frac{1}{k} \leq \frac{1}{t}$, and so $1 + \frac{1}{k} < 1 + \frac{r}{t} \leq e^{\varepsilon_0}(1 + \frac{r}{t}) = e^{\varepsilon}$. This shows the first part.

For the second part, apply Lemma A.2 to D', and let j be the partition number of d', i.e., $j = I_{d'}(y)$. We have:

$$\begin{aligned}
\Pr\{y \leftarrow \mathcal{F}(D')\} &= \frac{1}{|D'|}\sum_{i \geq 0} q(D', i, y) \\
&= \frac{1}{|D'|}\left[\sum_{i \geq 0 : i \neq j} q(D', i, y) + q(D', j, y)\right] \\
&= \frac{1}{|D'|}\sum_{i \geq 0 : i \neq j} q(D, i, y) + \frac{q(D', j, y)}{|D'|} \ .
\end{aligned}$$

The last equality follows from Lemma A.4 part (a).

Applying Lemma A.4 part (b), we obtain two cases.

- Case 1: $|C_j(D,y)| < t$. We have:

$$\begin{aligned}
\Pr\{y \leftarrow \mathcal{F}(D')\} &= \frac{1}{|D'|}\sum_{i\geq 0: i\neq j} q(D,i,y) + \frac{q(D',j,y)}{|D'|} \\
&\leq \frac{1}{|D'|}\sum_{i\geq 0: i\neq j} q(D,i,y) + \delta(D',j,y) \\
&\leq \frac{1}{|D'|}\sum_{i\geq 0} q(D,i,y) + \delta(D',j,y) \\
&= \frac{|D|}{|D'|}\Pr\{y \leftarrow \mathcal{F}(D)\} + \delta(D',j,y) \\
&\leq \Pr\{y \leftarrow \mathcal{F}(D)\} + \delta\ ,
\end{aligned}$$

where $\delta(D',j,y) = \frac{1}{|D'|}e^{-\varepsilon_0(k-t)}\sum_{s\in C_j(D',y)} p_s(y)$.

- Case 2: $|C_j(D,y)| \geq t$. We have:

$$\begin{aligned}
\Pr\{y \leftarrow \mathcal{F}(D')\} &= \frac{1}{|D'|}\left[\sum_{i\geq 0: i\neq j} q(D,i,y) + q(D',j,y)\right] \\
&\leq \frac{1}{|D'|}\left[\sum_{i\geq 0: i\neq j} q(D,i,y) + e^{\varepsilon_0}[1+\frac{r}{t}]q(D,j,y)\right] \\
&\leq e^{\varepsilon_0}[1+\frac{r}{t}]\frac{1}{|D'|}\sum_{i\geq 0} q(D,i,y) \\
&= e^{\varepsilon_0}[1+\frac{r}{t}]\frac{|D|}{|D'|}\Pr\{y \leftarrow \mathcal{F}(D)\} \\
&\leq e^{\varepsilon}\Pr\{y \leftarrow \mathcal{F}(D)\}\ ,
\end{aligned}$$

for $\varepsilon = \varepsilon_0 + \ln(1+\frac{r}{t})$ given that $\frac{|D|}{|D'|} < 1$.

Letting $\delta(D',d',y) = \delta(D', I_{d'}(y), y)$ finishes the proof. □

With this, we are in a position to prove Theorem 4.1.

Proof of Theorem 4.1. Fix dataset D with $|D| \geq k$ and any record $d' \in U$. Let $D' = D \cup \{d'\}$. The range of $\mathcal{F}$ is U and so any outcome Y is a non-empty subset of U. Fix an arbitrary $Y \subseteq U$ with $Y \neq \emptyset$.

We will show that $\Pr\{F(D_1) \in Y\} \leq e^{\varepsilon}\Pr\{F(D_2) \in Y\} + \delta$, whether $D_1 = D$ and $D_2 = D'$, or $D_1 = D'$ and $D_2 = D$.

Consider first the case $D_1 = D$ and $D_2 = D'$. Applying Lemma A.5, we obtain:

$$\begin{aligned}
\Pr\{\mathcal{F}(D) \in Y\} &= \sum_{y\in Y} \Pr\{y \leftarrow \mathcal{F}(D)\} \\
&\leq \sum_{y\in Y} e^{\varepsilon} \Pr\{y \leftarrow \mathcal{F}(D')\} \\
&= e^{\varepsilon} \Pr\{\mathcal{F}(D') \in Y\} \ .
\end{aligned}$$

Now, consider the case $D_1 = D'$ and $D_2 = D$. Define $c(d', y) = |C_{I_{d'}(y)}(D, y)|$. Given d', we can partition Y between those $y \in Y$ such that the partition in which d' falls has at least t and those such that the partition has less than t. That is, $Y = Y_{t-} \cup Y_{t+}$, with $Y_{t-} = \{y : y \in Y, c(d', y) < t\}$ and $Y_{t+} = \{y : y \in Y, c(d', y) \geq t\}$. We have:

$$\begin{aligned}
\Pr\{\mathcal{F}(D') \in Y\} &= \sum_{y\in Y} \Pr\{y \leftarrow \mathcal{F}(D')\} \\
&\leq \sum_{y\in Y_{t+}} e^{\varepsilon} \Pr\{y \leftarrow \mathcal{F}(D)\} \\
&+ \sum_{y\in Y_{t-}} [e^{\varepsilon} \Pr\{y \leftarrow \mathcal{F}(D)\} + \delta(D', I_{d'}(y), y)] \\
&= e^{\varepsilon} \sum_{y\in Y} \Pr\{y \leftarrow \mathcal{F}(D)\} + \sum_{y\in Y_{t-}} \delta(D', d', y) \\
&= e^{\varepsilon} \Pr\{y \leftarrow \mathcal{F}(D)\} + \sum_{y\in Y_{t-}} \delta(D', d', y) \ ,
\end{aligned}$$

where the inequality applies cases of Lemmas A.4 and A.5 separately to each y depending on whether $y \in Y_{t-}$ (case 1 in the proof of Lemma A.5) and $y \in Y_{t+}$ (case 2 in the proof of Lemma A.5).

It remains to show that $\sum_{y\in Y_{t-}} \delta(D', d', y) \leq e^{-\varepsilon_0(k-t)}$. For this define $C(D', d', y) = C_{I_{d'}(y)}(D', y)$. We have:

$$\begin{aligned}
\sum_{y\in Y_{t-}} \delta(D', d', y) &= \sum_{y\in Y_{t-}} \frac{e^{-\varepsilon_0(k-t)}}{|D'|} \sum_{s\in C(D',d',y)} p_s(y) \\
&= \frac{e^{-\varepsilon_0(k-t)}}{|D'|} \sum_{s\in D'} \sum_{y\in Y_{t-}} \mathbb{1}_{I_{d'}(y)=I_s(y)}\ p_s(y) \\
&\leq e^{-\varepsilon_0(k-t)} \ ,
\end{aligned}$$

given that $\sum_{y\in Y_{t-}} \mathbb{1}_{I_{d'}(y)=I_s(y)}\ p_s(y) \leq \sum_{y\in U} p_s(y) \leq 1$ for all $s \in D'$.

□

A.4 IMPROVED TEST DIFFERENTIAL PRIVACY

In this section, we prove Theorem 4.2.

Lemma 4.1. *For any dataset D with $|D| \geq 1$ and $D' = D \cup \{d'\}$ for some $d' \in U$, and any $y \in U$:*

$$\kappa(D, d, y) \leq \kappa(D', d, y) \leq \kappa(D, d, y) + 1 \ , \tag{4.5}$$

for any $d \in D$.

Proof of Lemma 4.1. Fix arbitrary D and D' as in the Lemma.

By definition: $\kappa(D, d, y) \leq \kappa(D', d, y)$.

Assume towards a contradiction that there exists $d \in D$ such that $\kappa(D', d, y) > \kappa(D, d, y) + 1$. By the definition of κ this can only be true if there exists $\hat{d}$ such that $\kappa(D', d, y) = \mathrm{ms}(D', \hat{d}, y)$ with $\Pr\{y \leftarrow \mathcal{M}(\hat{d})\} \geq \Pr\{y \leftarrow \mathcal{M}(d)\}$. There are two cases:

- If $\hat{d} = d'$ then there must exist some $\tilde{d} \in D$ with the largest probability that is less than $\Pr\{y \leftarrow \mathcal{M}(d')\}$. We have:

  $$\mathrm{ms}(D', d', y) = \frac{\sum_{s \in S_{d'} \setminus \{d'\}} \Pr\{y \leftarrow \mathcal{M}(s)\}}{\Pr\{y \leftarrow \mathcal{M}(d')\}} \leq \mathrm{ms}(D, \tilde{d}, y) + 1 \ ,$$

  where $\tilde{d}$ is such that $\Pr\{y \leftarrow \mathcal{M}(\tilde{d})\} \geq \Pr\{y \leftarrow \mathcal{M}(d)\}$.

- If $\hat{d} \in D$ then: either $\mathrm{ms}(D', \hat{d}, y) = \mathrm{ms}(D, \hat{d}, y) \leq \kappa(D, d, y)$ if $\Pr\{y \leftarrow \mathcal{M}(d')\} > \Pr\{y \leftarrow \mathcal{M}(\hat{d})\}$ which leads to an immediate contradiction, or $\Pr\{y \leftarrow \mathcal{M}(d')\} \leq \Pr\{y \leftarrow \mathcal{M}(\hat{d})\}$ so that:

  $$\mathrm{ms}(D', \hat{d}, y) = \mathrm{ms}(D, \hat{d}, y) + \frac{\Pr\{y \leftarrow \mathcal{M}(d')\}}{\Pr\{y \leftarrow \mathcal{M}(\hat{d})\}} \leq \mathrm{ms}(D, \hat{d}, y) + 1 \ .$$

Both cases lead to a contradiction. □

For clarity, we restate Theorem 4.2 here.

Theorem 4.2. *For any two neighboring datasets D_1, D_2 of at least $n \geq 1$ records, any $y \in U$, if $\mathcal{F}$ denotes Mechanism 3.1 with Definition 4.2 as privacy score function, and any well-behaved privacy test function δ (Definition 3.2) for $c_0 = 1$, $\beta_0 \geq 1$, then for any integer $1 \leq t \leq n$:*

$$\Pr\{y \leftarrow \mathcal{F}(D_1)\} \leq e^{\varepsilon} \Pr\{y \leftarrow \mathcal{F}(D_2)\} + \delta(t) \ ,$$

with $\varepsilon = \ln \beta_0 (1 + \frac{1}{t})$.

Proof of Theorem 4.2. Fix an arbitrary privacy test function δ that is well-behaved for $c_0 = 1$, $\beta_0 \geq 1$, synthetic $y \in U$ and dataset D' with $|D'| \geq 1$. Also, fix an arbitrary $d^\star \in U$ and let $D = D' \cup \{d^\star\}$.

We use the shorthand: $p_d(y) = \Pr\{y \leftarrow \mathcal{M}(d)\}$. We have:

$$\begin{aligned} |D'|\Pr\{y \leftarrow \mathcal{F}(D')\} &= \sum_{d \in D'} p_d(y)\ \delta(D', d, y) \\ &\leq \sum_{d \in D'} p_d(y)\ \delta(D, d, y) \\ &\leq \sum_{d \in D} p_d(y)\ \delta(D, d, y)\ . \end{aligned}$$

The first inequality holds because the test is non-decreasing, i.e., for all $d \in D'$: $\delta(D', d, y) \leq \delta(D, d, y)$ since $\kappa(D', d, y) \leq \kappa(D, d, y)$ (Lemma 4.1). Therefore:

$$\Pr\{y \leftarrow \mathcal{F}(D')\} \leq \frac{|D|}{|D'|}\Pr\{y \leftarrow \mathcal{F}(D)\}\ .$$

Now consider $\Pr\{y \leftarrow \mathcal{F}(D)\}$. To partition the sum depending on whether κ exceeds t, define: $D_{t^-} = \{d \in D : \kappa(D, d, y) < t\}$ and $D_{t^+} = \{d \in D : \kappa(D, d, y) \geq t\}$. We have:

$$\begin{aligned} |D|\Pr\{y \leftarrow \mathcal{F}(D)\} &= \sum_{d \in D} p_d(y)\ \delta(D, d, y) \\ &\leq \delta(t)\sum_{d \in D_{t-}} p_d(y) + \sum_{d \in D_{t+}} p_d(y)\ \delta(D, d, y) \\ &\leq \delta(t)|D| + \sum_{d \in D_{t+}} p_d(y)\ \delta(D, d, y) \end{aligned}$$

For $d \in D' \cap D_{t^+}$ we can apply Lemma 4.1 and the bounded increment property of the privacy test so that:

$$\delta(D, d, y) \leq \beta_0 \cdot \delta(D', d, y) \tag{A.3}$$

There are two cases depending on the value of $\kappa(D, d^\star, y)$:

- Case (i): $d^\star \in D_{t-}$. In this case:

$$\begin{aligned}
|D|\Pr\{y \leftarrow \mathcal{F}(D)\} &\leq \delta(t)|D| + \beta_0 \sum_{d \in D_{t+}} p_d(y)\ \delta(D', d, y) \\
&\leq \delta(t)|D| + \beta_0 \sum_{d \in D'} p_d(y)\ \delta(D', d, y) \\
&= \delta(t)|D| + \beta_0 |D'| \Pr\{y \leftarrow \mathcal{F}(D')\}
\end{aligned}$$

- Case (ii): $d^\star \in D_{t^+}$. Move the $d^\star$ term out of the sum before using Eq. (A.3). Let $D'_{t^+} = D' \cap D_{t^+}$.

$$\sum_{d \in D_{t+}} p_d(y)\ \delta(D, d, y) \leq \beta_0 \sum_{d \in D'_{t+}} p_d(y)\ \delta(D', d, y) + p_{d^\star}(y)\ \delta(D, d^\star, y)$$

Let $D_\star = \{d \in D' : \kappa(D, d, y) \geq \kappa(D, d^\star, y)\}$. Because $\kappa(D, d^\star, y) \geq t$ then it must be the case that:

$$p_{d^\star}(y) \leq \frac{1}{t} \sum_{d \in D_\star} p_d(y)\ ,$$

so that because the test is non-decreasing:

$$p_{d^\star}(y)\ \delta(D, d^\star, y) \leq t^{-1} \sum_{d \in D_\star} p_d(y)\ \delta(D, d, y)\ .$$

Since $D_\star \subseteq D'$ and $\delta(D, d, y) \leq \beta_0\ \delta(D', d, y)$ by bounded increments:

$$\begin{aligned}
|D|\left[\Pr\{y \leftarrow \mathcal{F}(D)\} - \delta(t)\right] &\leq \beta_0 \sum_{d \in D'_t} p_d(y)\ \delta(D', d, y) + p_{d^\star}(y)\ \delta(D, d^\star, y) \\
&\leq \beta_0\ (1 + t^{-1}) \sum_{d \in D'} p_d(y)\ \delta(D', d, y) \\
&= \beta_0\ (1 + t^{-1})\ |D'|\ \Pr\{y \leftarrow \mathcal{F}(D')\}\ .
\end{aligned}$$

Given that $|D'| = |D| - 1$, we conclude that for both $D_1 = D$, $D_2 = D'$ and $D_2 = D$, $D_1 = D'$:

$$\Pr\{y \leftarrow \mathcal{F}(D_1)\} \leq e^\varepsilon \Pr\{y \leftarrow \mathcal{F}(D_2)\} + \delta\ ,$$

with $\varepsilon = \ln\left[\beta_0(1 + t^{-1})\right]$ and $\delta = \delta(t)$. □

References

[1] L. Sweeney, “Weaving technology and policy together to maintain confidentiality,” *The Journal of Law, Medicine & Ethics*, vol. 25, no. 2-3, pp. 98–110, 1997.

[2] M. Barbaro and T. Z. Jr., “A face is exposed for aol searcher no. 4417749,” The New York Times, 2006.

[3] A. Narayanan and V. Shmatikov, “Robust de-anonymization of large sparse datasets,” in *Security and Privacy, 2008. IEEE Symposium on*. IEEE, 2008, pp. 111–125.

[4] R. Singel, “Netflix cancels recommendation contest after privacy lawsuit,” https://www.wired.com/2010/03/netflix-cancels-contest/, Wired, 2010.

[5] P. Golle and K. Partridge, “On the anonymity of home/work location pairs,” in *International Conference on Pervasive Computing*. Springer, 2009, pp. 390–397.

[6] A. Tockar, “Riding with the stars: Passenger privacy in the nyc taxicab dataset,” http://research.neustar.biz/2014/09/15/riding-with-the-stars-passenger-privacy-in-the-nyc-taxicab-dataset/, 2014.

[7] R. Wang, Y. F. Li, X. Wang, H. Tang, and X. Zhou, “Learning your identity and disease from research papers: information leaks in genome wide association study,” in *Proceedings of the 16th ACM conference on Computer and communications security*. ACM, 2009, pp. 534–544.

[8] M. Humbert, E. Ayday, J.-P. Hubaux, and A. Telenti, “Addressing the concerns of the lacks family: quantification of kin genomic privacy,” in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM, 2013, pp. 1141–1152.

[9] C. Culnane, B. I. Rubinstein, and V. Teague, “Health data in an open world,” *arXiv preprint arXiv:1712.05627*, 2017.

[10] E. Han, “Guilty: Health department breached privacy laws publishing data of 2.5m people,” https://www.smh.com.au/national/guilty-health-department-breached-privacy-laws-publishing-data-of-2-5m-people-20180329-p4z6wf.html, The Sydney Morning Herald, 2018.

[11] D. B. Rubin, “Statistical disclosure limitation,” *Journal of official Statistics*, vol. 9, no. 2, pp. 461–468, 1993.

[12] I. Dinur and K. Nissim, “Revealing information while preserving privacy,” in *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM, 2003, pp. 202–210.

[13] V. Mayer-Schönberger and K. Cukier, *Big Data: A revolution that transforms how we work, live, and think*. Houghton Mifflin Harcourt Boston, 2013.

[14] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*, 2013.

[15] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[16] Z. Wu, C. Valentini-Botinhao, O. Watts, and S. King, “Deep neural networks employing multi-task learning and stacked bottleneck features for speech synthesis,” in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on.* IEEE, 2015, pp. 4460–4464.

[17] A. Huang and R. Wu, “Deep learning for music,” *arXiv preprint arXiv:1606.04930*, 2016.

[18] R. Huang, S. Zhang, T. Li, and R. He, “Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2439–2448.

[19] J. Walker, C. Doersch, A. Gupta, and M. Hebert, “An uncertain future: Forecasting from static images using variational autoencoders,” in *European Conference on Computer Vision.* Springer, 2016, pp. 835–851.

[20] A. Graves, “Generating sequences with recurrent neural networks,” *arXiv preprint arXiv:1308.0850*, 2013.

[21] L. Gatys, A. S. Ecker, and M. Bethge, “Texture synthesis using convolutional neural networks,” in *Advances in Neural Information Processing Systems*, 2015, pp. 262–270.

[22] A. Odena, C. Olah, and J. Shlens, “Conditional image synthesis with auxiliary classifier gans,” in *International Conference on Machine Learning*, 2017, pp. 2642–2651.

[23] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” *arXiv preprint*, 2017.

[24] A. Halevy, P. Norvig, and F. Pereira, “The unreasonable effectiveness of data,” *IEEE Intelligent Systems*, vol. 24, no. 2, pp. 8–12, 2009.

[25] A. Rajaraman, “More data usually beats better algorithms,” *Datawocky Blog*, 2008.

[26] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther, “Autoencoding beyond pixels using a learned similarity metric,” in *International Conference on Machine Learning*, 2016, pp. 1558–1566.

[27] I. Csiszar and J. Körner, *Information theory: coding theorems for discrete memoryless systems.* Cambridge University Press, 2011.

[28] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of Cryptography Conference*, 2006, pp. 265–284.

[29] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Foundations and Trends in Theoretical Computer Science*, vol. 9, no. 3-4, pp. 211–407, 2014.

[30] N. Li, W. Qardaji, D. Su, Y. Wu, and W. Yang, "Membership privacy: a unifying framework for privacy definitions," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security.* ACM, 2013, pp. 889–900.

[31] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *Foundations of Computer Science, 2007. FOCS'07. 48th Annual IEEE Symposium on.* IEEE, 2007, pp. 94–103.

[32] V. Bindschaedler, R. Shokri, and C. A. Gunter, "Plausible deniability for privacy-preserving data synthesis," *Proceedings of the VLDB Endowment*, vol. 10, no. 5, 2017.

[33] V. Bindschaedler and R. Shokri, "Synthesizing plausible privacy-preserving location traces," in *Security and Privacy.* IEEE, 2016.

[34] J. A. Fox and P. E. Tracy, "Randomized response: A method for sensitive surveys." 1986.

[35] P. Kairouz, S. Oh, and P. Viswanath, "The composition theorem for differential privacy," in *Proceedings of The 32nd International Conference on Machine Learning*, 2015, pp. 1376–1385.

[36] M. Sebastian and E. Mohammadi, "Ratio buckets: A numeric method for k-fold tight differential privacy," *Cryptology ePrint Archive: Report*, 2017.

[37] M. Bellare, A. Desai, E. Jokipii, and P. Rogaway, "A concrete security treatment of symmetric encryption," in *Foundations of Computer Science, 1997. Proceedings., 38th Annual Symposium on.* IEEE, 1997, pp. 394–403.

[38] L. D. Brown, T. T. Cai, and A. DasGupta, "Interval estimation for a binomial proportion," *Statistical science*, pp. 101–117, 2001.

[39] L. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[40] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *Information Theory, IEEE Transactions on*, vol. 13, no. 2, pp. 260–269, 1967.

[41] R. Shokri, G. Theodorakopoulos, J.-Y. Le Boudec, and J.-P. Hubaux, "Quantifying location privacy," in *Proceedings of the 2011 IEEE Symposium on Security and Privacy*, ser. SP '11. Washington, DC, USA: IEEE Computer Society, 2011. [Online]. Available: http://dx.doi.org/10.1109/SP.2011.18 pp. 247–262.

[42] Y. Rubner, C. Tomasi, and L. Guibas, "A metric for distributions with applications to image databases," in *Computer Vision, 1998. Sixth International Conference on*, jan 1998, pp. 59 –66.

[43] Y. Rubner, C. Tomasi, and L. J. Guibas, "The Earth Mover's Distance as a Metric for Image Retrieval," *International Journal of Computer Vision*, vol. 40, pp. 99–121, 2000, 10.1023/A:1026543900054. [Online]. Available: http://dx.doi.org/10.1023/A:1026543900054

[44] E. Levina and P. Bickel, "The Earth Mover's distance is the Mallows distance: some insights from statistics," in *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, vol. 2, 2001, pp. 251 –256 vol.2.

[45] T. M. Cover and J. A. Thomas, *Elements of information theory.* John Wiley & Sons, 1994.

[46] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the Society for Industrial & Applied Mathematics*, vol. 5, no. 1, pp. 32–38, 1957.

[47] P. M. Pardalos, H. Wolkowicz et al., *Quadratic Assignment and Related Problems: DIMACS Workshop, May 20-21, 1993.* American Mathematical Soc., 1994, vol. 16.

[48] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The journal of chemical physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[49] S. Brooks and B. Morgan, "Optimization using simulated annealing," *The Statistician*, pp. 241–257, 1995.

[50] D. Margaritis, "Learning bayesian network model structure from data," Ph.D. dissertation, US Army, 2003.

[51] M. A. Hall, "Correlation-based feature selection for machine learning," Ph.D. dissertation, The University of Waikato, 1999.

[52] I. T. Jolliffe, "Principal component analysis and factor analysis," in *Principal component analysis.* Springer, 1986, pp. 115–128.

[53] C. Ding and X. He, "K-means clustering via principal component analysis," in *Proceedings of the twenty-first international conference on Machine learning.* ACM, 2004, p. 29.

[54] G. Saporta and N. Niang, "Principal component analysis: application to statistical process control," *Data analysis*, pp. 1–23, 2009.

[55] T. Chanyaswad, C. Liu, and P. Mittal, "Coupling dimensionality reduction with generative model for non-interactive private data release," *arXiv preprint arXiv:1709.00054*, 2017.

[56] N. Kiukkonen, J. Blom, O. Dousse, D. Gatica-Perez, and J. Laurila, "Towards rich mobile phone datasets: Lausanne data collection campaign," *Proc. ICPS, Berlin*, 2010.

[57] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from gps trajectories," in *Proceedings of the 18th International Conference on World Wide Web*, ser. WWW '09. New York, NY, USA: ACM, 2009. [Online]. Available: http://doi.acm.org/10.1145/1526709.1526816 pp. 791–800.

[58] M. Ye, D. Shou, W.-C. Lee, P. Yin, and K. Janowicz, "On the semantic annotation of places in location-based social networks," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 2011, pp. 520–528.

[59] T. M. T. Do and D. Gatica-Perez, "The places of our lives: Visiting patterns and automatic labeling from longitudinal smartphone data," *Mobile Computing, IEEE Transactions on*, vol. 13, no. 3, pp. 638–648, 2014.

[60] X. Liu, J. Biagioni, J. Eriksson, Y. Wang, G. Forman, and Y. Zhu, "Mining large-scale, sparse gps traces for map inference: comparison of approaches," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 2012, pp. 669–677.

[61] J. Yuan, Y. Zheng, and X. Xie, "Discovering regions of different functions in a city using human mobility and pois," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 2012, pp. 186–194.

[62] M. Lichman and P. Smyth, "Modeling human location data with mixtures of kernel densities," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 2014, pp. 35–44.

[63] D. Karamshuk, A. Noulas, S. Scellato, V. Nicosia, and C. Mascolo, "Geo-spotting: Mining online location-based services for optimal retail store placement," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 2013, pp. 793–801.

[64] R. Chen, G. Acs, and C. Castelluccia, "Differentially private sequential data publication via variable-length n-grams," in *Proceedings of the 2012 ACM conference on Computer and communications security.* ACM, 2012, pp. 638–649.

[65] R. Shokri, G. Theodorakopoulos, G. Danezis, J.-P. Hubaux, and J.-Y. Le Boudec, "Quantifying location privacy: the case of sporadic location exposure," in *Proceedings of the 11th international conference on Privacy enhancing technologies*, ser. PETS'11. Berlin, Heidelberg: Springer-Verlag, 2011. [Online]. Available: http://dl.acm.org/citation.cfm?id=2032162.2032166 pp. 57–76.

[66] T.-H. You, W.-C. Peng, and W.-C. Lee, "Protecting moving trajectories with dummies," in *Mobile Data Management, 2007 International Conference on*, May 2007, pp. 278–282.

[67] H. Kido, Y. Yanagisawa, and T. Satoh, "An anonymous communication technique using dummies for location-based services," in *Pervasive Services, 2005. ICPS '05. Proceedings. International Conference on*, July 2005, pp. 88–97.

[68] R. Chow and P. Golle, "Faking contextual data for fun, profit, and privacy," in *WPES '09: Proceedings of the 8th ACM workshop on Privacy in the electronic society.* New York, NY, USA: ACM, 2009, pp. 105–108.

[69] R. Kato, M. Iwata, T. Hara, A. Suzuki, X. Xie, Y. Arase, and S. Nishio, "A dummy-based anonymization method based on user trajectory with pauses," in *Proceedings of the 20th International Conference on Advances in Geographic Information Systems.* ACM, 2012, pp. 249–258.

[70] J. Krumm, "Realistic driving trips for location privacy," in *Pervasive '09: Proceedings of the 7th International Conference on Pervasive Computing.* Berlin, Heidelberg: Springer-Verlag, 2009, pp. 25–41.

[71] A. Suzuki, M. Iwata, Y. Arase, T. Hara, X. Xie, and S. Nishio, "A user location anonymization method for location based services in a real environment," in *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, ser. GIS '10. New York, NY, USA: ACM, 2010. [Online]. Available: http://doi.acm.org/10.1145/1869790.1869846 pp. 398–401.

[72] US Census Bureau, "American community survey (acs)," http://www.census.gov/programs-surveys/acs/.

[73] T. Julian and R. Kominski, "Education and synthetic work-life earnings estimates. american community survey reports. acs-14." *US Census Bureau*, 2011.

[74] M. T. Mora, D. J. Villa, and A. Dávila, "Language maintenance among the children of immigrants: A comparison of border states with other regions of the us," *Southwest Journal of Linguistics*, vol. 24, no. 1-2, pp. 127–145, 2005.

[75] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially private empirical risk minimization," *Journal of Machine Learning Research*, vol. 12, no. Mar, pp. 1069–1109, 2011.

[76] Healthcare Cost and Utilization Project (HCUP), "Hcup nationwide inpatient sample (nis)," https://www.hcup-us.ahrq.gov/nisoverview.jsp, 2008.

[77] Healthcare Cost and Utilization Project (HCUP), "Hcup clinical classifications software (ccs) for icd-9-cm," https://www.hcup-us.ahrq.gov/toolssoftware/ccs/ccs.jsp, Retrieved March, 2016.

[78] X. W. Ziwei Liu, Ping Luo and X. Tang, "Large-scale celebfaces attributes (celeba) dataset," http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html.

[79] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther, "Autoencoding beyond pixels using a learned similarity metric (code)," http://github.com/andersbll/autoencoding_beyond_pixels.

[80] P. Samarati and L. Sweeney, "Generalizing data to provide anonymity when disclosing information," in *PODS*, vol. 98, 1998, p. 188.

[81] L. Sweeney, "k-anonymity: A model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 05, pp. 557–570, 2002.

[82] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "l-diversity: Privacy beyond k-anonymity," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 1, no. 1, p. 3, 2007.

[83] N. Li, T. Li, and S. Venkatasubramanian, "t-closeness: Privacy beyond k-anonymity and l-diversity," in *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on.* IEEE, 2007, pp. 106–115.

[84] M. E. Nergiz, M. Atzori, and C. Clifton, "Hiding the presence of individuals from shared databases," in *Proceedings of the 2007 ACM SIGMOD international conference on Management of data.* ACM, 2007, pp. 665–676.

[85] K. Wang, B. C. Fung, and P. S. Yu, "Template-based privacy preservation in classification problems," in *Data Mining, Fifth IEEE International Conference on.* IEEE, 2005, pp. 8–pp.

[86] J. Brickell and V. Shmatikov, "The cost of privacy: destruction of data-mining utility in anonymized data publishing," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 2008, pp. 70–78.

[87] C. C. Aggarwal, "On k-anonymity and the curse of dimensionality," in *Proceedings of the 31st international conference on Very large data bases.* VLDB Endowment, 2005, pp. 901–909.

[88] J. Domingo-Ferrer and V. Torra, "A critique of k-anonymity and some of its enhancements," in *Availability, Reliability and Security, 2008. ARES 08. Third International Conference on.* IEEE, 2008, pp. 990–993.

[89] V. Rastogi, D. Suciu, and S. Hong, "The boundary between privacy and utility in data publishing," in *Proceedings of the 33rd international conference on Very large data bases.* VLDB Endowment, 2007, pp. 531–542.

[90] R. Shokri, C. Troncoso, C. Diaz, J. Freudiger, and J.-P. Hubaux, "Unraveling an old cloak: k-anonymity for location privacy," in *Proceedings of the 9th annual ACM workshop on Privacy in the electronic society.* ACM, 2010, pp. 115–118.

[91] D. J. Martin, D. Kifer, A. Machanavajjhala, J. Gehrke, and J. Y. Halpern, "Worst-case background knowledge for privacy-preserving data publishing," in *2007 IEEE 23rd International Conference on Data Engineering*. IEEE, 2007, pp. 126–135.

[92] B. Gedik and L. Liu, "Protecting location privacy with personalized k-anonymity: Architecture and algorithms," *IEEE Transactions on Mobile Computing*, vol. 7, no. 1, pp. 1–18, 2008.

[93] B. Niu, Q. Li, X. Zhu, G. Cao, and H. Li, "Achieving k-anonymity in privacy-aware location-based services," in *INFOCOM, 2014 Proceedings IEEE*. IEEE, 2014, pp. 754–762.

[94] K. Vu, R. Zheng, and J. Gao, "Efficient algorithms for k-anonymous location privacy in participatory sensing," in *INFOCOM, 2012 Proceedings IEEE*. IEEE, 2012, pp. 2399–2407.

[95] D. Yang, X. Fang, and G. Xue, "Truthful incentive mechanisms for k-anonymity location privacy," in *INFOCOM, 2013 Proceedings IEEE*. IEEE, 2013, pp. 2994–3002.

[96] Y. Zhang, W. Tong, and S. Zhong, "On designing satisfaction-ratio-aware truthful incentive mechanisms for k-anonymity location privacy," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 11, pp. 2528–2541, 2016.

[97] G. Navarro-Arribas, V. Torra, A. Erola, and J. Castellà-Roca, "User k-anonymity for privacy preserving data mining of query logs," *Information Processing & Management*, vol. 48, no. 3, pp. 476–487, 2012.

[98] J. Liu and K. Wang, "Anonymizing bag-valued sparse data by semantic similarity-based clustering," *Knowledge and information systems*, vol. 35, no. 2, pp. 435–461, 2013.

[99] B. K. Tripathy and A. Mitra, "An algorithm to achieve k-anonymity and l-diversity anonymisation in social networks," in *Computational Aspects of Social Networks (CASoN), 2012 Fourth International Conference on*. IEEE, 2012, pp. 126–131.

[100] K. El Emam and F. K. Dankar, "Protecting privacy using k-anonymity," *Journal of the American Medical Informatics Association*, vol. 15, no. 5, pp. 627–637, 2008.

[101] K. El Emam, F. K. Dankar, R. Issa, E. Jonker, D. Amyot, E. Cogo, J.-P. Corriveau, M. Walker, S. Chowdhury, R. Vaillancourt et al., "A globally optimal k-anonymity method for the de-identification of health data," *Journal of the American Medical Informatics Association*, vol. 16, no. 5, pp. 670–682, 2009.

[102] F. Kohlmayer, F. Prasser, C. Eckert, A. Kemper, and K. A. Kuhn, "Highly efficient optimal k-anonymity for biomedical datasets," in *Computer-Based Medical Systems (CBMS), 2012 25th International Symposium on*. IEEE, 2012, pp. 1–6.

[103] F. Prasser, F. Kohlmayer, and K. A. Kuhn, “A benchmark of globally-optimal anonymization methods for biomedical data,” in *Computer-Based Medical Systems (CBMS), 2014 IEEE 27th International Symposium on*. IEEE, 2014, pp. 66–71.

[104] N. R. Adam and J. C. Worthmann, “Security-control methods for statistical databases: a comparative study,” *ACM Computing Surveys (CSUR)*, vol. 21, no. 4, pp. 515–556, 1989.

[105] C. Dwork and K. Nissim, “Privacy-preserving datamining on vertically partitioned databases,” in *Annual International Cryptology Conference*. Springer, 2004, pp. 528–544.

[106] A. Blum, C. Dwork, F. McSherry, and K. Nissim, “Practical privacy: the sulq framework,” in *Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM, 2005, pp. 128–138.

[107] B. Barak, K. Chaudhuri, C. Dwork, S. Kale, F. McSherry, and K. Talwar, “Privacy, accuracy, and consistency too: a holistic solution to contingency table release,” in *Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM, 2007, pp. 273–282.

[108] K. Nissim, S. Raskhodnikova, and A. Smith, “Smooth sensitivity and sampling in private data analysis,” in *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*. ACM, 2007, pp. 75–84.

[109] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, “Our data, ourselves: Privacy via distributed noise generation,” in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2006, pp. 486–503.

[110] C. Dwork, “Differential privacy: A survey of results,” in *International Conference on Theory and Applications of Models of Computation*, 2008.

[111] C. Dwork, “Differential privacy,” in *Encyclopedia of Cryptography and Security*. Springer, 2011, pp. 338–340.

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

[113] J. Blocki, A. Datta, and J. Bonneau, “Differentially private password frequency lists,” in *NDSS*, 2016.

[114] G. Cormode, M. Procopiuc, D. Srivastava, and T. Tran, “Differentially private publication of sparse data,” in *International Conference on Database Theory (ICDT)*, 2012.

[115] E. Lantz, K. Boyd, and D. Page, “Subsampled exponential mechanism: Differential privacy in large output spaces,” in *Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security*, ser. AISec ’15. New York, NY, USA: ACM, 2015. [Online]. Available: http://doi.acm.org/10.1145/2808769.2808776 pp. 25–33.

[116] H. Li, L. Xiong, and X. Jiang, "Differentially private synthesization of multi-dimensional data using copula functions," in *Advances in database technology: proceedings. International Conference on Extending Database Technology*, vol. 2014. NIH Public Access, 2014, p. 475.

[117] J. Xu, Z. Zhang, X. Xiao, Y. Yang, G. Yu, and M. Winslett, "Differentially private histogram publication," *The VLDB Journal*, vol. 22, no. 6, pp. 797–822, 2013.

[118] A. Haeberlen, B. C. Pierce, and A. Narayan, "Differential privacy under fire." in *USENIX Security Symposium*, 2011.

[119] D. Kifer and A. Machanavajjhala, "No free lunch in data privacy," in *ACM SIGMOD*. ACM, 2011.

[120] J. Gehrke, M. Hay, E. Lui, and R. Pass, "Crowd-blending privacy," in *Advances in Cryptology–CRYPTO 2012*. Springer, 2012, pp. 479–496.

[121] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Local privacy and statistical minimax rates," in *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*. IEEE, 2013, pp. 429–438.

[122] V. Rastogi, M. Hay, G. Miklau, and D. Suciu, "Relationship privacy: output perturbation for queries with joins," in *Proceedings of the twenty-eighth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM, 2009, pp. 107–116.

[123] R. Hall, A. Rinaldo, and L. Wasserman, "Random differential privacy," *arXiv preprint arXiv:1112.2680*, 2011.

[124] I. Mironov, O. Pandey, O. Reingold, and S. Vadhan, "Computational differential privacy," in *Advances in Cryptology-CRYPTO 2009*. Springer, 2009, pp. 126–142.

[125] E. Lui and R. Pass, "Outlier privacy," in *Theory of Cryptography Conference*. Springer, 2015, pp. 277–305.

[126] D. Kifer and A. Machanavajjhala, "Pufferfish: A framework for mathematical privacy definitions," *ACM Transactions on Database Systems (TODS)*, vol. 39, no. 1, p. 3, 2014.

[127] R. Bassily, A. Groce, J. Katz, and A. Smith, "Coupled-worlds privacy: Exploiting adversarial uncertainty in statistical data privacy," in *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*. IEEE, 2013, pp. 439–448.

[128] N. Li, W. Qardaji, and D. Su, "On sampling, anonymization, and differential privacy or, k-anonymization meets differential privacy," in *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*. ACM, 2012, pp. 32–33.

[129] Y.-X. Wang, S. Fienberg, and A. Smola, "Privacy for free: Posterior sampling and stochastic gradient monte carlo," in *International Conference on Machine Learning*, 2015, pp. 2493–2502.

[130] A. Ghosh, T. Roughgarden, and M. Sundararajan, "Universally utility-maximizing privacy mechanisms," *SIAM Journal on Computing*, vol. 41, no. 6, pp. 1673–1693, 2012.

[131] Q. Geng, P. Kairouz, S. Oh, and P. Viswanath, "The staircase mechanism in differential privacy," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 7, pp. 1176–1184, 2015.

[132] Q. Geng and P. Viswanath, "The optimal mechanism in differential privacy," in *Information Theory (ISIT), 2014 IEEE International Symposium on.* IEEE, 2014, pp. 2371–2375.

[133] J. Lee and C. W. Clifton, "Top-k frequent itemsets via differentially private fp-trees," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 2014, pp. 931–940.

[134] M. Lyu, D. Su, and N. Li, "Understanding the sparse vector technique for differential privacy," *Proceedings of the VLDB Endowment*, vol. 10, no. 6, pp. 637–648, 2017.

[135] C. Dwork, G. N. Rothblum, and S. Vadhan, "Boosting and differential privacy," in *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on.* IEEE, 2010, pp. 51–60.

[136] J. Murtagh and S. Vadhan, "The complexity of computing the optimal composition of differential privacy," in *Theory of Cryptography Conference.* Springer, 2016, pp. 157–175.

[137] C. Dwork and J. Lei, "Differential privacy and robust statistics," in *Proceedings of the forty-first annual ACM symposium on Theory of computing.* ACM, 2009, pp. 371–380.

[138] A. Korolova, K. Kenthapadi, N. Mishra, and A. Ntoulas, "Releasing search queries and clicks privately," in *Proceedings of the 18th international conference on World wide web.* ACM, 2009, pp. 171–180.

[139] M. A. Whiting, J. Haack, and C. Varley, "Creating realistic, scenario-based synthetic data for test and evaluation of information analytics software," in *Proceedings of the 2008 Workshop on BEyond time and errors: novel evaLuation methods for Information Visualization.* ACM, 2008, p. 8.

[140] K. H. Pedersen, K. Torp, and R. Wind, "Simple and realistic data generation," in *Proceedings of the 32nd International Conference on Very Large Data Bases.* Association for Computing Machinery, 2006.

[141] T. K. Ho and H. S. Baird, "Evaluation of ocr accuracy using synthetic data," in *Proceedings of the 4th Annual Symposium on Document Analysis and Information Retrieval.* Citeseer, 1995.

[142] J. Gray, P. Sundaresan, S. Englert, K. Baclawski, and P. J. Weinberger, "Quickly generating billion-record synthetic databases," in *ACM SIGMOD Record*, vol. 23, no. 2. ACM, 1994, pp. 243–252.

[143] D. Bitton, D. J. DeWitt, and C. Turbyfill, *Benchmarking database systems: A systematic approach.* University of Wisconsin-Madison. Computer Sciences Department, 1983.

[144] A. Aboulnaga, J. F. Naughton, and C. Zhang, "Generating synthetic complex-structured xml data." in *WebDB*, vol. 1, 2001, pp. 79–84.

[145] T. G. Armstrong, V. Ponnekanti, D. Borthakur, and M. Callaghan, "Linkbench: a database benchmark based on the facebook social graph," in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data.* ACM, 2013, pp. 1185–1196.

[146] J. P. Reiter, "Releasing multiply imputed, synthetic public use microdata: An illustration and empirical study," *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, vol. 168, no. 1, pp. 185–205, 2005.

[147] J. Drechsler, *Synthetic datasets for statistical disclosure control: theory and implementation.* Springer Science & Business Media, 2011, vol. 201.

[148] H. Wang and J. P. Reiter, "Multiple imputation for sharing precise geographies in public use data," *The annals of applied statistics*, vol. 6, no. 1, p. 229, 2012.

[149] S. A. Keller, S. Shipp, and A. Schroeder, "Does big data change the privacy landscape? a review of the issues," *Annual Review of Statistics and Its Application*, vol. 3, pp. 161–180, 2016.

[150] S. Hawala, "Producing partially synthetic data to avoid disclosure," in *Proceedings of the Joint Statistical Meetings. Alexandria, VA: American Statistical Association*, 2008.

[151] S. K. Kinney, J. P. Reiter, A. P. Reznek, J. Miranda, R. S. Jarmin, and J. M. Abowd, "Towards unrestricted public use business microdata: The synthetic longitudinal business database," *International Statistical Review*, vol. 79, no. 3, pp. 362–384, 2011.

[152] S. K. Kinney, J. P. Reiter, and J. Miranda, "Synlbd 2.0: improving the synthetic longitudinal business database," *Statistical Journal of the IAOS*, vol. 30, no. 2, pp. 129–135, 2014.

[153] C. Dwork, M. Naor, O. Reingold, G. N. Rothblum, and S. Vadhan, "On the complexity of differentially private data release: efficient algorithms and hardness results," in *Proceedings of the forty-first annual ACM symposium on Theory of computing.* ACM, 2009, pp. 381–390.

[154] J. Ullman and S. Vadhan, "Pcps and the hardness of generating private synthetic data," in *Theory of Cryptography Conference*. Springer, 2011, pp. 400–416.

[155] A. Blum, K. Ligett, and A. Roth, "A learning theory approach to noninteractive database privacy," *Journal of the ACM (JACM)*, vol. 60, no. 2, p. 12, 2013.

[156] J. M. Abowd and L. Vilhuber, "How protective are synthetic data?" in *International Conference on Privacy in Statistical Databases*. Springer, 2008, pp. 239–246.

[157] A. Machanavajjhala, D. Kifer, J. Abowd, J. Gehrke, and L. Vilhuber, "Privacy: Theory meets practice on the map," in *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering*. IEEE Computer Society, 2008, pp. 277–286.

[158] L. Wasserman and S. Zhou, "A statistical framework for differential privacy," *Journal of the American Statistical Association*, vol. 105, no. 489, pp. 375–389, 2010.

[159] A.-S. Charest, "How can we analyze differentially-private synthetic datasets?" *Journal of Privacy and Confidentiality*, vol. 2, no. 2, p. 3, 2011.

[160] D. McClure and J. P. Reiter, "Differential privacy and statistical disclosure risk measures: An investigation with binary synthetic data." *Transactions on Data Privacy*, vol. 5, no. 3, pp. 535–552, 2012.

[161] C. Xu, J. Ren, Y. Zhang, Z. Qin, and K. Ren, "Dppro: Differentially private high-dimensional data release via random projection," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 12, pp. 3081–3093, 2017.

[162] F. Liu, "Model-based differential private data synthesis," *arXiv preprint arXiv:1606.08052*, 2016.

[163] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao, "Privbayes: Private data release via bayesian networks," in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. ACM, 2014, pp. 1423–1434.

[164] M. Hardt, K. Ligett, and F. McSherry, "A simple and practical algorithm for differentially private data release," in *Advances in Neural Information Processing Systems*, 2012, pp. 2339–2347.

[165] Y. Xiao, L. Xiong, and C. Yuan, "Differentially private data release through multidimensional partitioning," in *Workshop on Secure Data Management*. Springer, 2010, pp. 150–168.

[166] G. Acs, C. Castelluccia, and R. Chen, "Differentially private histogram publishing through lossy compression," in *Data Mining (ICDM), 2012 IEEE 12th International Conference on*. IEEE, 2012, pp. 1–10.

[167] Y. Xiao, J. Gardner, and L. Xiong, "Dpcube: Releasing differentially private data cubes for health information," in *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*. IEEE, 2012, pp. 1305–1308.

[168] W. Qardaji, W. Yang, and N. Li, "Differentially private grids for geospatial data," in *Data Engineering (ICDE), 2013 IEEE 29th International Conference on*. IEEE, 2013, pp. 757–768.

[169] H. Ze, A. Senior, and M. Schuster, "Statistical parametric speech synthesis using deep neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 7962–7966.

[170] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum, "Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling," in *Advances in Neural Information Processing Systems*, 2016, pp. 82–90.

[171] R. Dahl, M. Norouzi, and J. Shlens, "Pixel recursive super resolution," *arXiv preprint arXiv:1702.00783*, 2017.

[172] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European Conference on Computer Vision*. Springer, 2016, pp. 694–711.

[173] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang et al., "Photo-realistic single image super-resolution using a generative adversarial network," *arXiv preprint*, 2016.

[174] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*. IEEE, 2016, pp. 2536–2544.

[175] C. Esteban, S. L. Hyland, and G. Rätsch, "Real-valued (medical) time series generation with recurrent conditional gans," *arXiv preprint arXiv:1706.02633*, 2017.

[176] A. van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves et al., "Conditional image generation with pixelcnn decoders," in *Advances in Neural Information Processing Systems*, 2016, pp. 4790–4798.

[177] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," in *33rd International Conference on Machine Learning*, 2016, pp. 1060–1069.

[178] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[179] S. Song, K. Chaudhuri, and A. D. Sarwate, "Stochastic gradient descent with differentially private updates," in *Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE*. IEEE, 2013, pp. 245–248.

[180] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. ACM, 2015, pp. 1310–1321.

[181] A. Rajkumar and S. Agarwal, “A differentially private stochastic gradient descent algorithm for multiparty classification,” in *Artificial Intelligence and Statistics*, 2012, pp. 933–941.

[182] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, “Deep learning with differential privacy,” in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security.* ACM, 2016, pp. 308–318.

[183] G. Acs, L. Melis, C. Castelluccia, and E. De Cristofaro, “Differentially private mixture of generative neural networks,” *arXiv preprint arXiv:1709.04514*, 2017.

[184] X. Zhang, S. Ji, and T. Wang, “Differentially private releasing via deep generative model,” *arXiv preprint arXiv:1801.01594*, 2018.

[185] A. Triastcyn and B. Faltings, “Generating differentially private datasets using gans,” *arXiv preprint arXiv:1803.03148*, 2018.