# Evaluating Methods for Privacy-Preserving Data Sharing in Genomics

*Maria-Bristena Oprisanu*

A dissertation submitted in partial fulfillment

of the requirements for the degree of

**Doctor of Philosophy**

of

**University College London**.

Department of Computer Science

University College London

January 3, 2022

I, Maria-Bristena Oprisanu, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

# Abstract

The availability of genomic data is often essential to progress in biomedical research, personalized medicine, drug development, etc. However, its extreme sensitivity makes it problematic, if not outright impossible, to publish or share it.

In this dissertation, we study and build systems that are geared towards privacy preserving genomic data sharing. We first look at the Matchmaker Exchange, a platform that connects multiple distributed databases through an API and allows researchers to query for genetic variants in other databases through the network. However, queries are broadcast to all researchers that made a similar query in any of the the connected databases, which can lead to a reluctance to use the platform, due to loss of privacy or competitive advantage. In order to overcome this reluctance, we propose a framework to support anonymous querying on the platform.

Since genomic data's sensitivity does not degrade over time, we analyze the real-world guarantees provided by the only tool available for long term genomic data storage. We find that the system offers low security when the adversary has access to side information, and we support our claims by empirical evidence.

We also study the viability of synthetic data for privacy preserving data sharing. Since for genomic data research, the utility of the data provided is of the utmost importance, we first perform a utility evaluation on generative models for different types of datasets (i.e., financial data, images and locations). Then, we propose a privacy evaluation framework for synthetic data. We then perform a measurement study assessing state-of-the-art generative models specifically geared for human genomic data, looking at both utility and privacy perspectives. Overall, we find that there is no single approach for generating synthetic data that performs well across the board from both utility and privacy perspectives.

# Impact Statement

The research in this thesis focuses on analyzing existing methods that enable data sharing in genomics.

First, we look at the Matchmaker Exchange. Being aware that broadcasting queries might discourage researchers to use the platform due to possible loss of competitive advantage or privacy, we propose a framework that enables anonymous queries, AnoniMME. By using reverse private information retrieval as a building block, we enable queries to support public key encryption of contact details and add a response phase where users can anonymously reply to queries. Using an experimental evaluation, we show that AnoniMME is efficient and scalable, and can bring anonymity to the Matchmaker Exchange with low overhead. Thus, we are confident that it can be deployed in the wild and further encourage researchers to share genomic data.

Second, we look at GenoGuard, the only tool proposed for long term encryption of genomic data. We find that, under a low-entropy password setting, if the adversary obtains side information about the target sequence, there is a significant lower bound in their advantage. This shows that the system offers low security when the adversary has access to side information, and we support our claims by empirical evidence. In a high-entropy password setting, we quantify the privacy loss for a user using GenoGuard compared to state-of-the-art inference methods for genomic data, showing that it is non-negligible. This prompts the need for more research geared towards the design of long-term encryption tools for genomic data.

Finally, we study whether synthetic data is a viable solution for enabling privacy preserving data sharing. Since for genomic data research the utility of the data provided is of the utmost importance, we first perform a utility evaluation on generative models for different types of datasets (i.e., financial data, images and locations). We find that generative models that are purposely built for a specific type of data yield the best utility. Thus, a successful generic approach able to generate universally meaningful synthetic data might not be viable. We then propose a privacy evaluation framework for synthetic data. By treating the generative models as a black box, we quantify the privacy risk arising from releasing synthetic datasets under membership inference. Our framework is built as a modular Python library and is available for use to researchers and practitioners, which can also adapt the evaluation to any privacy concern specific to the data holder's case. Overall, we find that synthetic data is not the silver-bullet solution to privacy problems of microdata publishing. Last, but not least, we perform a measurement study assessing state-of-the-art generative models specifically geared for human genomic data, looking at both utility and privacy perspectives (membership inference). Moreover, we show that, even without access to the full sequence of a target individual, membership inference attacks are still a threat to individual privacy. Similar to the generic case, we find that no single approach for generating synthetic data performs well across the board, thus suggesting that synthetic data without explicit privacy protection might not be a viable option.

# Acknowledgements

First of all I want to thank my supervisor, Prof. Emiliano De Cristofaro. I am deeply grateful for his advice and guidance throughout my PhD journey. I would also like to thank my secondary supervisor Prof. Christophe Dessimoz for all the valuable feedback provided.

I would also like to thank everyone in the UCL Information Security Research Group for providing a thriving environment for my PhD. Thanks are also in order to all the people I had the chance to collaborate with: Dr. Adria Gascon, Theresa Stadler, Prof. Carmela Troncoso, and George Ganev.

The research presented here would not have been possible without the support from a Google Faculty Award on "Enabling Progress in Genomic Research via Privacy Preserving Data Sharing" and RCUK.

To my family, thank you for always supporting me with a special thanks to my mother for always being there for me and always encouraging me.

Last, but not least, I'd like to thank my husband, Marius, for his unconditional love and support, and for bearing my non-stop presence over the past year, as well as my faithful research assistant, Leia.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Advances in genome sequencing and genomics are enabling tremendous progress in medicine and healthcare, paving the way to making the prevention, diagnosis, and treatment of diseases tailored to the individual's specific genetic makeup, thus becoming cheaper and more effective. Researchers are also gaining a better understanding, and developing more successful treatments of rare genetic diseases. However, even though sequencing costs have plummeted from billions to thousands of dollars over the past 15 years [11], it is still hard for researchers to gain access to genomic data, especially those pertaining to rare conditions.

Therefore, seamless progress in genomics research hinges on the ability to collaborate and share data among different institutions. Numerous initiatives have been established to support and encourage genomic data sharing, and funding agencies like the National Institutes of Health (NIH) often make it a requirement to fund grant applications [129]. Successful data sharing programs include the International HapMap Project [122], which helped identify common genetic variations and study their involvement in human health and disease, the 1000 Genomes Project [12], and the 100,000 Genomes Project [63], both aiming to create a catalog of human variation and genotype data, as well as the European 1+ Million Genomes [157], which aims to allow for more personalized treatments and provide new impactful research.

More recently, programs like "All of Us"[3] or Genomics England [6] are in the process of sequencing the genomes of millions of individuals in the US and in the UK. Aiming to foster collaborations, the Global Alliance for Genomics and

Health (GA4GH) [73] was established, with core funding from NIH, Wellcome, and Canada's CanShare, with the explicit goal of making data sharing between institutes simple and effective. The GA4GH has developed several platforms, e.g., the Beacon Project [72], allowing researchers to search if a certain allele exists in a database of genomic data, as well as the Matchmaker Exchange (MME) [142], which facilitates rare disease discovery.

Alas, as more and more genomic data is generated, collected, and shared, serious privacy, security, and ethical concerns also become increasingly relevant. The genome contains very sensitive information related to, e.g., ethnic heritage, disease predispositions, and other phenotypic traits [24]. Furthermore, even though most published genomes have been anonymized, previous work has shown that anonymization does not provide an effective safeguard for genomic data [79]. While some individuals choose to donate their genome to science, or even publicly share it through initiatives like the Personal Genomes Project [141], others might be concerned about their privacy, or fear discrimination by employers, government agencies, insurance providers, etc. [36].

Worse yet, consequences of genomic data disclosure are not limited in time or to the data owner: due to its hereditary nature, access to one's sequenced genome inherently implies access to many features that are relevant to their progeny and their close relatives. A case in point is the story of Henrietta Lacks, a patient who died of cancer in 1951. Some of her cancerous cells were revealed to be useful for research because of their ability to keep on dividing. Unbeknownst to her family, the cells became the most commonly used "immortal cell line," and their genome was eventually sequenced and published [106]. This prompted serious privacy concerns among her family members, even 60 years later [39].

Motivated by these challenges, the research community has produced a large body of work aiming to protect genomic privacy and enable privacy-preserving sharing and testing of human genomes [118]. Available solutions mostly rely on cryptographic tools, including encryption as well as Secure Computation, Homomorphic Encryption, Oblivious RAM, etc. [28]. However, modern encryption algorithms provide security guarantees only against computationally bounded adversary; essentially, their security is assumed to last for 30 to 50 years [161]. While this timeframe is acceptable for most uses of encryption, it is not for genomic data. In fact, in a study published by Mittos et. al [118], the long term security of genomic data has been identified not only as one of the most important problems to solve, but also one of the most difficult.

More recently, genomics researchers have begun to investigate the possibility of releasing *synthetic* datasets, rather than real/anonymized data [155]. This follows a general trend in healthcare; for instance, the National Health Service (NHS) in England has recently concluded a project focused on releasing synthetic Emergency Room ("A&E") records [128]. The intuition is to use generative models to learn to generate samples with the same characteristics—more precisely, with the same distribution—of the real data. That is, rather than releasing data of actual individuals, entities share artificially generated data in such a way that the statistical properties of the original data are preserved, but minimizing the risk of malicious inference of sensitive information [61].

## 1.1 Research Questions and Contributions

Given the existing challenges and the great opportunities related to genomics, we set the broad goal for this dissertation to evaluate existing methods proposed for genomic data sharing. Such a goal entails addressing several open research questions, including:

RQ1. How can we improve existing frameworks that aim to support genomic data sharing and encourage researchers to collaborate?

RQ2. Can we rely on existing encryption techniques for long term encryption to enable sharing of encrypted genomic data?

RQ3. Is synthetic data a suitable alternative, both in terms of utility and privacy, for enabling genomic data sharing?

In this dissertation, we set to shed light on these research questions. More specifically, this thesis makes the following contributions:

C1. We propose a framework to support anonymous queries within the genomic data sharing platform the Matchmaker Exchange, without breaking any of its current security settings or functionality requirements. We use as our main building block Reverse Private Information Retrieval(PIR), extending queries to support public key encryption of contact details and adding a response phase so that users can also anonymously reply to queries. We show, experimentally, that our framework is efficient and scalable, and can bring anonymity to the Matchmaker Exchange with low overhead.

C2. We analyze the only system proposed to date that aims to provide long-term security for genomic data, GenoGuard. We show that under a low entropy password setting, if the adversary obtains side information about the target sequence, there is a significant lower bound in their advantage. In a high entropy password setting we quantify the privacy loss for a user as a result of using GenoGuard compared to state of the art inference methods for genomic data, showing that the privacy loss is non-negligible.

C3. We assess the suitability of synthetic data for enabling privacy-preserving data sharing in genomics. We begin our assessment with an utility evaluation of generative models on different types of datasets such as financial data, images and locations. We find that a generic approach might not be a viable option, and that models purposely built for a specific dataset yield the best utility. Then, we provide a privacy assessment in order to estimate the privacy risk associated with releasing a synthetic dataset instead of the real dataset. Finally, we look at state of the art generative models proposed for genomic data, and provide a measurement study focused on synthetic genomic data, assessing it from both utility and privacy perspectives.

## 1.2 Thesis Outline

The rest of the thesis is as follows. Chapter 2 introduces preliminary notions and tools used throughout the manuscript. Then, in Chapter 3, we review relevant prior work. Chapters 4 to 6 cover the contributions of this dissertation. More specifically, in Chapter 4, we introduce AnoniMME, a framework geared to bring anonymity to the Matchmaker Exchange (MME) platform. In Chapter 5, we perform a security analysis of the only tool aimed at long term encryption of genomic data, GenoGuard. Then, in Chapter 6 we explore the potential of using synthetic data to aid data sharing in genomics. Finally, Chapter 7 concludes the thesis with a discussion.

## 1.3 Collaboration Statement

The content presented in this thesis has been co-authored with other researchers and has been published (or submitted for publication) at Computer Science and Bioinformatics conferences and workshops.

The work presented in Chapter 4 has been conducted in collaboration with Prof. Emiliano De Cristofaro and was published at ISMB 2018 [133]. Specifically, the research required and initial writing of the paper was done by the author of this thesis, as part of MSc project while Prof. De Cristofaro had an advisory and editorial role.

The work of Chapter 5 has been conducted in collaboration with Prof. Cristophe Dessimoz and Prof. Emiliano De Cristofaro and was published at WPES 2019 [134]. The evaluation and initial writing of the paper was done by the author of this thesis, while Prof. Dessimoz and Prof. De Cristofaro had an advisory and editorial role.

Finally, the work of Chapter 6 involves multiple papers and collaborations. The utility evaluation of generative models has been performed in collaboration with Dr. Adria Gascon and Prof. De Cristofaro, and was published as a technical report [136]. The analysis and the writing of the technical report was done by the author of this thesis, while Dr. Gascon and Prof. De Cristofaro had an advisory and editorial role with respect to the writing of the paper and planning experiments. The privacy evaluation framework has been done in collaboration with Theresa Stadler and Prof. Carmela Troncoso from EPFL, and has been accepted for publication at USENIX Security 2022 [164]. My main contribution to this paper was the evaluation and analysis of results for the PATE-GAN model, as well as co-writing the sections of the paper related to this model. The measurement study of state-of-the-art generative models for human genomic data was done in collaboration with Georgi Ganev and Prof. Emiliano De Cristofaro, and has been accepted for publication at NDSS 2022 [135]. The analysis and writing of the paper was done by the author of the thesis, apart from the analysis of the WGAN model, which was performed by Georgi Ganev. Prof. De Cristofaro had an advisory and editorial role.

# Chapter 2

# Background

This chapter provides some relevant background information used throughout the paper.

## 2.1 Genomics Primer

**Genome.** In the nucleus of an organism's cell, double stranded deoxyribonucleic acid (DNA) molecules are packaged into thread-like structures called chromosomes. DNA molecules consist of two long and complementary polymer chains of four units called nucleotides, described with the letters A, C, G, and T. All chromosomes together make up the *genome*, which represents the entirety of the organism's hereditary information; in humans, the genome includes 3.2 billion nucleotides. A *gene* is a particular region of the genome that contain the information to produce functional molecules, in particular proteins. For instance, the BRCA2 [188] is a human tumor suppressor gene (it encodes a protein responsible for repairing the DNA), and a mutation in that gene increases significantly the risk for breast cancer [69]. *Alleles* are the different versions of genes, as organisms inherit two alleles for each gene, one from each parent. The set of genes is also called the *genotype.* Finally, the *haplotype* is a group of alleles in an organisms that are inherited together from a single parent [50].

**SNPs and SNVs.** Humans share about 99.5% of the genome, while the rest differs due to genetic variations. The most common type of variants are Single Nucleotide Polymorphisms (SNPs) [149], which occur at a single position and in at least 1%

of the population. More generally, variants at specific positions of a genome are referred to as Single-Nucleotide Variants (SNVs); they may be due to SNPs, to rare variants in the population, or to new mutations. Typically, SNPs and SNVs are encoded with a value in $\{0,1,2\}$, with 0 denoting the most common variant (allele) in the population, and 1 and 2 denoting alternative alleles.

**Allele Frequency (AF).** The frequency of an allele at a certain position in a given population is known as Allele Frequency (AF). More specifically, it is the ratio of the number of times the allele appears in the population over the total number of copies of the gene. In a nutshell, it shows the genetic diversity of a species' population.

**Linkage Disequilibrium (LD).** LD refers to the non-random association of alleles at two or more positions in the general population, defined as the difference between the frequency of a particular combination of alleles at different positions and the one expected by random association.

**Recombination Rate (RR).** The process of determining the frequency with which characteristics are inherited together is known as recombination. This is due to two chromosomes of similar composition coming together and performing a molecular crossover, thus, exchanging the genetic content. Because recombination can occur with small probability at any location along the chromosome, the frequency of recombination between two locations depends on the distance separating them. Therefore, for genes sufficiently distant on the same chromosome, the amount of crossover is high enough to destroy the correlation between alleles [112]. The recombination rate (RR), as defined in [143], is the probability that a transmitted haplotype constitutes a new combination of alleles different from that of either parental haplotype. An example of how a haplotype is created by copying parts from the other haplotypes is illustrated in Figure 2.1.

**Genome-Wide Association Studies (GWAS).** GWAS are hypothesis-free methods for identifying associations between genetic regions and traits. A typical GWAS looks for common variants in a number of individuals, both with and without a trait, using genome-wide SNP arrays [124, 62].

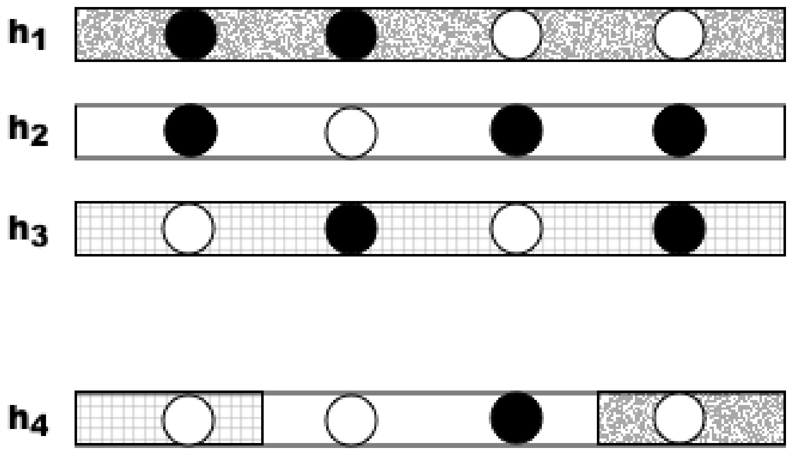


**Figure 2.1:** An example of a haplotype, $h_4$, built as an imperfect mosaic from $h_1, h_2, h_3$. $h_4$ is created by (imperfectly) "copying" parts from $h_1, h_2$, and $h_3$. Each column of circles represents a SNP locus, with the black and white circles denoting the two alleles – major and minor. (Adapted from [112]).

## 2.2 SNV Correlation Modeling

In order to model correlations between SNVs, and perform sequence inference (i.e. predicting the values of SNVs from a sequence), one can use a few different approaches (for more details on various SNV correlations, please refer to [156]). Throughout this thesis we focus on three models, all based on Markov chains; see next for a description of Markov chains and the three SNV correlation models.

**Markov Chains.** A Markov chain is a probabilistic model encoding a sequence of possible events: the probability of each one of them depends only on the state attained in the previous event [132].

In the context of genomes, a Markov chain can represent a series of SNVs ordered by their positions. In particular, a $k$-th order Markov chain, on genome sequences, can be used to encode a set of SNVs, where the value of each SNV$_i$ depends on the values of the $k$ preceding ones:

$$\Pr(SNV_i) = \Pr(SNV_i | SNV_{i-1}, \dots, SNV_{i-k}) \tag{2.1}$$

**Most likely genotype.** First, we use a correlation model based on the 1st order Markov chain model from AF and LD. Given allele frequencies (AF) and linkage disequilibrium (LD), we predict each SNV using the highest conditional probability of the SNV occurring. For each SNV, the joint probability matrix is computed taking into consideration the LD with the previous one and the AF. If a SNV is not in LD with the previous one, the probability is computed using only the allele frequency. When this model is used for inference, the highest value from the joint probability matrix or the highest probability given by the AF is chosen to predict the specific SNV.

**Sampled genotype.** The second model is built from the 1st order Markov chain model from AF and LD. For this model, the conditional probabilities are computed in a similar way as in the most likely genotype model. The main difference is in the choice of the value of the SNV, given the three computed probabilities for major homozygous $\mathrm{Pr}_0$, heterozygous $\mathrm{Pr}_1$, and minor homozygous allele $\mathrm{Pr}_2$. A seed $s$ is chosen uniformly at random from the interval $[0,1)$. If $s < \mathrm{Pr}_0$, then choose the SNV to be major homozygous; if $\mathrm{Pr}_0 \leq s < \mathrm{Pr}_1 + \mathrm{Pr}_0$, then the SNV is heterozygous; and minor homozygous otherwise.

**RR Model.** This is a high-order correlation model that relates LD patterns to the underlying recombination rate [112]. Given a set of $n$ sampled haplotypes, $\{h_1, h_2, ..., h_n\}$, the model relates their distribution to the underlying recombination rate. Given the recombination parameter, $\rho$, we have:

$$\mathrm{Pr}(h_1,...,h_n|\rho) = \mathrm{Pr}(h_1|\rho) \cdot \mathrm{Pr}(h_2|h_1;\rho) \cdot \ldots \cdot \mathrm{Pr}(h_n|h_1,\ldots,h_{n-1};\rho) \tag{2.2}$$

We use this model to determine the value of a SNP at a given position. At each SNP, $h_k$ is a possibly imperfect copy of one of $h_1, ..., h_{k-1}$. Let $H_i$ denote which haplotype is copied at a position $i$. For instance, in the example presented in Figure 2.1, for $h_4$, we have $(H_1, H_2, H_3, H_4) = (3,2,2,1)$. For a generic $h_k$, each $H_i$ can be modeled as a Markov chain on $\{1,\ldots,k-1\}$. Assuming that one part of $h_k$ comes from $h_i$, the next adjacent part can be copied from any of the $k-1$ haplotypes, and the probability depends on the recombination rates between these two parts. Overall, the probability of a particular haploid genotype $h_k$ can be computed as the sum over all possible event sequences of recombination and mutation that could lead to $h_k$. Let $h_{i,j+1}$ denote the allele found at position $j+1$ in haplotype $i$, and $h_{i,\leq j}$ denote the values of the first $j$ positions of haplotype $i$ (i.e. the prefix sequence of $h_{i,j+1}$). Then, we can compute the conditional probability of an allele $h_{k,j+1}$, given all preceding alleles as:

$$\Pr(h_{k,j+1}|h_{k,j},\dots h_{k,1}) = \frac{\Pr(h_{k,\leq j+1})}{\Pr(h_{k,\leq j})} \tag{2.3}$$

## 2.3 Cryptography Primer

In this subsection, we introduce the main cryptographic notions used throughout this thesis.

### 2.3.1 Private Information Retrieval (PIR)

Private Information Retrieval (PIR) is a cryptographic primitive geared to protect user privacy while querying a public database, i.e., it allows a user to retrieve an item from the database without revealing which item is being retrieved. A trivial PIR solution requires the user to download the entire database, and query it locally, but this obviously does not scale to large databases. There are two kinds of PIR protocols in literature: information theoretic [48, 74] and computational PIR [47, 137]: in the former, the database is split over multiple non-colluding servers, while, in the latter, there is a single server and the protocol relies on some computational assumptions (e.g., the quadratic residuosity problem).

In this thesis, we use information theoretical PIR (IT-PIR), as introduced by Chor et al. [48], which involves $n$ servers, at least one of which is considered to be honest. Without loss of generality, assume a user wants to retrieve a single bit $x_i$ from a string $x$ of length $l$. On input the index $i \in \{1,\dots,l\}$, and a random input $r$, the user produces $n$ queries, one per server, each of length $t_q$. The queries are built using secret sharing, in the form of sequences of subsets $A_1, A_2, \dots A_t \subseteq \{1,\dots,n\}$, such that no single query leaks any information on the index of interest. The servers send back replies, of length $t_a$, which depend on the contents of the database and the corresponding query. The replies consist of corresponding sequence of bits $\oplus_{j \in A_1} x_j, \ldots, \oplus_{j \in A_t} x_j$. Finally, the user can recover the desired item from the replies, using $i$ and $r$.

### 2.3.2 Reverse PIR

Reverse PIR allows a user to *write* into a database, without revealing at which row, thus complementing PIR, which lets users privately read a row from a database. It was introduced by Corrigan-Gibbs et al. [52] as part of an anonymous messaging system, called Riposte, which enables users to anonymously post messages to a shared "bulletin board," maintained at a small number of servers.

Specifically, let us consider a user, on input a private database index $i$, at which she wants to write a private value $y$. As in PIR, she uses secret sharing in order to split the value $y$ into $n$ shares, one for each server, so that no individual share leaks any information, and thus offering privacy in the context of all but one servers colluding with each other. However, all of the shares combined reveal $y$ at index $i$ of the database. In order to guarantee anonymity, multiple writes to the database are collected within an epoch (which can be based on either time or number of writes). At the end of the epoch, the servers aggregate the writes received during that epoch and update the database with all the processed writes. As a consequence, messages can be seamlessly posted and read, but there is no direct link between a user and a certain post.

### 2.3.3 Honey Encryption

Honey Encryption (HE) [100] is a cryptographic primitive used to provide confidentiality guarantees in the presence of possible brute-force attacks. It is a variant of Password-Based Encryption (PBE), in that it also uses an arbitrary string (password) to perform randomized encryption of a plaintext. Its main property is that all decryptions of a ciphertext will yield a plausible-looking plaintext, which is thus indistinguishable from the correct one.

The main building block of HE is the Distribution-Transforming Encoder (DTE). A DTE is a randomized encoding scheme `(encode, decode)` tailored on the target distribution. The `encode` algorithm takes as input a message $M$ from the message space $\mathscr{M}$, and outputs a value $S$ in a set $\mathscr{S}$, i.e., the seed space. Whereas, `decode` takes a seed $S \in \mathscr{S}$ and outputs a message $M \in \mathscr{M}$. A DTE scheme is *correct* if, for any $M \in \mathscr{M}$, $\Pr[\texttt{decode}(\texttt{encode}(M)) = M] = 1$. The DTE-then-encrypt scheme presented in [100] applies `encode` to a message, and then performs encryption using a secure symmetric encryption scheme (e.g., AES). Similarly, to decrypt a ciphertext, one first decrypts using the underlying cipher (e.g., AES), and then applies the `decode` algorithm.

## 2.4 Machine Learning Primer

In this section we evaluate machine learning concepts used throughout this dissertation.

### 2.4.1 Generative Models

Generative model are a class of statistical models that can generate new data instances. This model is typically used to estimate probabilities, modeling data points and distinguishing between classes based on these probabilities. Usually, a system's input features and output variables (as well as unobserved variables) are represented homogeneously by a joint probability distribution. These variables can be discrete or continuous and may also be multidimensional [97]. The types of generative models that we use in this thesis include the following models:

**Bayesian Networks.** Bayesian networks [140] are a widely-used class of probabilistic graphical models. They consist of two parts: a structure and parameters. The structure of a Bayesian network is a directed acyclic graph with a conditional probability distribution for each node. Each node in the network represents a domain variable and each arc between two nodes represents a probabilistic dependency. A Bayesian network is a compact, flexible and interpretable representation of a joint probability distribution. It is also a useful tool in knowledge discovery as directed acyclic graphs allow representing causal relations between variables. Typically, a Bayesian network is learned from data.

**Generative Adversarial Networks (GANs).** A GAN [76] is an unsupervised deep learning model consisting of two neural networks, a generator and a discriminator, which compete against each other in the form of a game setting. During training, the generator's goal is to produce synthetic data and the discriminator evaluates them against real data samples in order to distinguish the synthetic from the real samples. The training objective is to learn the data distribution so that the data samples produced by the generator cannot be distinguished from real data by the discriminator.

**Variational Autoencoders (VAEs).** A VAE [104, 151] comprises of two neural networks, namely an encoder and a decoder, as well as a loss function. The encoder is used to compress the data into a latent space, and then the decoder takes the latent representation output by the autoencoder and uses it to to reconstruct the data as close to the original input data as possible. The loss function penalizes the network for creating output data that differs from the input data.

**Restricted Boltzmann Machines (RBMs).** RBMs [162] are generative models geared to learn a probability distribution over a set of inputs. RBMs are shallow, two-layer neural nets: the first layer is known as the "visible" (on input) layer and the second as the hidden layer. The two layers are connected via a bipartite graph – i.e., every node in the visible layer is connected to every node in the hidden one, but no two nodes in the same group are connected to each other, allowing for more efficient training algorithms. The learning procedure consists of maximizing the likelihood function over the visible variables of the model. The RBM models recreate data in an unsupervised manner through many forward and backward passes between the two layers, corresponding to sampling from the learned distribution. The output of the hidden layer passes through an activation function, which then becomes the input for the hidden layer. RBMs are typically used for dimensionality reduction, classification, regression, collaborative filtering, topic modeling, etc.

### 2.4.2 Discriminative Models

In contrast to generative models that model the underlying distribution of the data and to generate samples based on the underlying probabilistic model, discriminative models learn the boundaries between classes or labels in a dataset conditional probability, thus optimizing mappings between the inputs to the desired outputs (e.g. a discrete class). They are usually used for classification and regression tasks. The following types of discriminative models are used throughout this thesis:

**Support Vector Machines (SVMs).** SVMs [131] are one of the classical machine learning techniques that is still used for classification or regression tasks. An SVM is an algorithm for maximizing a particular mathematical function with respect to a given collection of data. The task of an SVM is to determine which category a new data point belongs to.

**Linear Regression.** Linear regression is a commonly used type of predictive analysis. It studies the linear relationship between a continuous, dependent variable and one or more independent variables (which can be either continuous or categorical. It uses the mathematical equation $y = mx + c$ that describes the line of best fit for the relationship between $y$, the dependent variable and $x$, the independent variable. The method that is normally used for estimation of accuracy in for the linear regression algorithm is the least square estimation.

**Logistic Regression.** Logistic regression [55] is a classification algorithm commonly used for predicting a binary outcome based on a set of variables. In contrast to linear regression, it is used to predict values of *categorical* variables. For logistic regression, the weighted sum of input is passed through a sigmoid activation function and uses the maximum likelihood estimation for accuracy.

**k-Nearest Neighbors (KNN).** The KNN algorithm [54] is another standard machine learning method, based on the idea that observations with similar characteristics tend to have similar outcomes. KNN is a non-parametric algorithm, meaning that it can work for both continuous and categorical variables. This method only requires the choice of $k$, the number of neighbors to be considered when making the classification, which is normally chosen as the value which minimizes the classification error on some independent validation data or by cross-validation procedures, and the distance metric to be used (e.g. Euclidean distance).

**Random Forests.** A random forest classifier [35] is an ensemble classifier that produces multiple decision trees. Rather than depending on one decision tree, the random forest model combines the predictions of multiple decision trees to produce a more accurate prediction.

## 2.5 Differential Privacy

In this section, we discuss Differential Privacy (DP) as well as the Moments Accountant method presented in [13] in the context of privacy-preserving deep learning.

### 2.5.1 Definitions and Properties

DP addresses the paradox of learning nothing about an individual while learning useful information about a population [60]. Generally speaking, differential privacy aims to provide rigorous, statistical guarantees against what an adversary can infer from learning the result of some randomized algorithm. Typically, differentially private techniques protect the privacy of individual data subjects by adding random noise when producing statistics. In a nutshell, differential privacy guarantees that an individual will be exposed to the same privacy risk whether or not her data is included in a differentially private analysis.

**Definition.** Formally, for two non-negative numbers $\varepsilon, \delta$, a randomized algorithm $\mathscr{A}$ satisfies $(\varepsilon, \delta)$-differential privacy if and only if, for any neighboring datasets $D$ and $D'$ (i.e. differing at most one record), and for the possible output $S \subseteq Range(\mathscr{A})$, the following formula holds:

$$\Pr[\mathscr{A}(D) \in S] \leq e^{\varepsilon} \Pr[\mathscr{A}(D') \in S] + \delta$$

**The $\varepsilon, \delta$ parameters.** Differential privacy analyses allow for some information leakage specific to individual data subjects, controlled by the privacy parameter $\varepsilon$. This measures the effect on each individual's information on the output of the analysis. With smaller values of $\varepsilon$, the dataset is considered to have stronger privacy, but less accuracy, thus reducing its utility. An intuitive description of the privacy parameter, along with supporting examples, is available in [130].

If $\delta = 0$, we say that the mechanism is $\varepsilon$-differentially private. This is considered to be the *absolute* case, in which one cannot gain more than a small amount of probabilistic information about a single individual. By contrast, $\delta > 0$ allows for a small probability of failure, e.g., an output can occur with probability $\delta > 0$ if an individual is present in the dataset, and never happens otherwise.

As per [60], the values of $\delta$ are usually computed as an inverse function of a polynomial in the size of the dataset. In particular, any values of $\delta$ on the order of $\frac{1}{|D|}$, where $|D|$ represents the size of the dataset $D$, are considered to be very dangerous: even though this case is "privacy-preserving," it would still allow the publication of complete records for a small number of participants. In order to better understand why values of $\delta$ of the order of $\frac{1}{|D|}$ can be dangerous, consider an algorithm that simply releases an entry of the dataset $|D|$ uniformly at random. This algorithm is $\varepsilon, \delta$, with $\varepsilon = \infty$ and $\delta = \frac{1}{|D|}$, and yet obviously does not provide a meaningful privacy guarantee.

**Post-Processing.** Differential privacy is "immune" to post-processing, i.e., any function applied to the output of a differentially private algorithm cannot provide less privacy guarantees than the original mechanism. More formally, let $\mathscr{A}$ be a randomized algorithm that is $(\varepsilon, \delta)$-differentially private, and $f$ be an arbitrary mapping. Then, $f \circ \mathscr{A}$ is also $(\varepsilon, \delta)$-differentially private.

**Composition.** One of the most important properties of differential privacy is its robustness under composition. When combining multiple differentially private mechanisms, composition theorems can be used to account for the total differential privacy of the system. More precisely, for mechanisms $\mathscr{M}_1, \dots \mathscr{M}_n$, where each $\mathscr{M}_i$ is a $(\varepsilon_i, \delta_i)$- differentially private algorithm, we have that $\mathscr{M}_{[n]}(x) = (\mathscr{M}_1(x), \dots \mathscr{M}_n(x))$ is $(\sum_{i=1}^{n} \varepsilon_i, \sum_{i=1}^{n} \delta_i)$- diffentially private.

**Strong Composition.** Dinur and Nissim [58] and Dwork and Nissim [59] showed that, under $k$-fold adaptive composition on a single database, the privacy parameter deteriorates less if a negligible loss in $\delta$ can be tolerated. This yields the Strong Composition Theorem:

For every $\varepsilon > 0$, $\delta, \delta' > 0$ and $k \in \mathbb{N}$, the class of $(\varepsilon, \delta)$-differentially private mechanisms is $(\varepsilon', k\delta + \delta')$-differentially private under $k$-fold adaptive composition, for

$$\varepsilon' = \sqrt{2k \ln \tfrac{1}{\delta'}} \cdot \varepsilon + k \cdot \varepsilon\varepsilon_0,$$

where $\varepsilon_0 = e^{\varepsilon} - 1$. This theorem introduces a stronger bound on the expected privacy loss due to multiple mechanisms, which relaxes the worst-case result given from the composition theorem.

**Sensitivity.** The notion of the sensitivity of a function is very useful in the design of differentially private algorithms, and define the notion of sensitivity of a function with respect to a neighboring relationship. Given a query $F$ on a dataset $D$, the sensitivity is used to adjust the amount of noise required for $F(D)$. More formally, if $F$ is a function that maps a dataset (in matrix form) into a fixed-size vector of real numbers, we can define the $L_i$-sensitivity of $F$ as:

$$S_i(F) = \max_{D,D'} ||F(D) - F(D')||_i,$$

where $||\cdot||_i$ denotes the $L_i$ norm, $i \in \{1, 2\}$ and $D$ and $D'$ are any two neighboring datasets.

**The Gaussian Mechanism.** One of the most widely used methods to achieve $(\varepsilon, \delta)$-differential privacy is to add Gaussian noise to the result of a query. Given a function $F : D \to \mathbb{R}$ over a dataset $D$, if $\sigma = S_2(F)\sqrt{2\ln(2/\delta)}/\varepsilon$, and $\mathcal{N}(0, \sigma^2)$ are independent and identically distributed Gaussian random variables, the mechanism $\mathcal{M}$ provides $(\varepsilon, \delta)$-differential privacy when:

$$\mathcal{M}(D) = f(D) + \mathcal{N}(0, \sigma^2)$$

More specifically, the Gaussian Mechanism with parameter $\sigma$ adds noise scaled to $\mathcal{N}(0, \sigma^2)$ to each of the components of the output.

**Differentially Private Data Generation.** One of the applications of differential privacy is differentially private synthetic data generation. The main idea is that,

**Algorithm 1** Differentially Private SGD

```
1: Input: Examples $\{x_1,\dots,x_N\}$, loss function $\mathscr{L}(\theta = \frac{1}{N}\sum_i \mathscr{L}(\theta, x_i)$. Parameters: learning rate $\eta_t$, noise scale $\sigma$, group size $L$, gradient norm bound $C$, number of steps $T$.
2: Initialize $\theta_0$ randomly
3: for $t = 1$ to $T-1$ do
4:     Take a random sample $L_t$, with sampling probability $q = \frac{L}{N}$
5:     for each $i \in L_t$ do
6:         $g_t(x_i) = \nabla_{\theta_t}\mathscr{L}(\theta_t, x_i)$
7:         $\overline{g_t}(x_i) = \frac{g_t(x_i)}{\max(1, \frac{\|g_t(x_i)\|_2}{C})}$
8:     $\widetilde{g_t} = \frac{1}{L}(\sum_i \overline{g_t}(x_i) + \mathscr{N}(0, \sigma^2 C^2 I))$
9:     $\theta_{t+1} = \theta_t - \eta_t \widetilde{g_t}$
   return $\theta_T$ and compute the overall privacy cost $(\varepsilon, \delta)$ using a privacy accounting method.
```

once the data is generated, it can be used for multiple analyses, without the need to further increase the privacy budget. This is a consequence of the postprocessing property of differential privacy mentioned above. Incidentally, the National Institute of Standards and Technology (NIST) launched a differential privacy synthetic data challenge in 2018 [173], aiming to find synthetic data generation algorithms that protect individual privacy but provide a high utility of the overall dataset.

### 2.5.2 Moments Accountant

In [13], Abadi et al. show how to bound the privacy loss of gradient descent-like computations. They present a privacy-preserving stochastic gradient descent (SGD) algorithm for training a model with parameters $\theta$ by minimizing the loss function $\mathscr{L}(\theta)$; see Algorithm 1.

At each step of the algorithm, the gradient is computed for a small subset of examples (Line 6), then the $L_2$ norm of each gradient is clipped (Line 7) in order to bound the influence of each individual example. Noise is then added to the clipped gradient (Line 8) and a step is taken in the opposite direction of the average noisy gradient. When outputting the model (Line 9), the total privacy loss needs to be computed, using a privacy accounting method. The composition property of differential privacy allows to computes the privacy cost at each access to the training data, accumulating the cost over all training steps. An initial bound is given by the strong composition theorem, however, [13] provides a stronger accounting method, namely the *moments accountant*, which saves a factor $\sqrt{\frac{T}{\delta}}$ in the asymptotic bound.

**Formal Description.** Next, we provide a formal description of the main technical aspects of the moments accountant technique, mirroring the presentation in the original paper [13]. For proofs and details we direct the reader to the supplementary material of [13].

For any neighboring databases $D, D'$, a mechanism $\mathscr{M}$, an auxiliary input *aux* and a outcome $o$, the privacy loss at $o$ is defined as:

$$c(o;\mathscr{M},aux,D,D') = \log \frac{\Pr[\mathscr{M}(aux,D)=o]}{\Pr[\mathscr{M}(aux,D')=o]}$$

For a given mechanism $\mathscr{M}$, the $\lambda^{\text{th}}$ moment $\alpha_{\mathscr{M}}(\lambda;\mathscr{M},aux,D,D')$ is defined as:

$$\alpha_{\mathscr{M}}(\lambda;\mathscr{M},aux,D,D') = \log \mathbb{E}_{o\sim\mathscr{M}(aux,D)}[\exp(\lambda \mathrm{c}(\mathrm{o};\mathscr{M},aux,D,D'))]$$

In order to provide the guarantees of the mechanism, all possible $\alpha_{\mathscr{M}}(\lambda;\mathscr{M},aux,D,D')$ should be bounded, so $\alpha_{\mathscr{M}}(\lambda)$ is defined as:

$$\alpha_{\mathscr{M}}(\lambda) = \max_{aux,D,D'} \alpha_{\mathscr{M}}(\lambda;\mathscr{M},aux,D,D'),$$

where the maximum is taken over all possible *aux* and all neighboring datasets $D$ and $D'$.

Then, $\alpha$ achieves the following properties:

- *Composability:* Suppose that a mechanism $\mathscr{M}$ consists of a sequence of adaptive mechanisms $\mathscr{M}_1 \ldots \mathscr{M}_k$. Then, for any $\lambda$:

$$\alpha_{\mathscr{M}}(\lambda) \leq \textstyle\sum_{i=1}^{k} \alpha_{\mathscr{M}_i}(\lambda).$$

  Thus, in order to bound the mechanism overall, we need to bound each $\alpha_{\mathscr{M}_i}(\lambda)$ and sum them.

- *Tail Bound:* For any $\varepsilon > 0$, the mechanism $\mathscr{M}$ is $(\varepsilon,\delta)$-differentially private for

**Algorithm 2** Epsilon Computation

```
1:  Input $q, \sigma, \delta$, number of epochs $ep$, number of orders $\lambda$
2:  Initialize $l = -\infty$, $\alpha_{list}, \varepsilon_{list} = \emptyset$
3:  for $i = 2$ to $\lambda + 1$ do
4:      for $j = 2$ to $\imath + 1$ do
5:          $l_1 = \log(\frac{i!}{j!(i-j)!} + j \cdot \log q + (i-j)\log(1-q)$
6:          $s = l_1 + (j^2 - j)/(s\sigma^2)$
7:          if $l$ is $-\infty$ then
8:              $l = s$
9:          else:
10:             $l = \log(\mathrm{e}^{\log(l-s)}) + s$
11:     $\alpha = l \cdot ep \cdot q$
12:     Append $\alpha$ to $\alpha_{list}$
13: $i = 2$
14: for each $\alpha$ in $\alpha_{list}$ do:
15:     Append $(\alpha - \log(\delta))/(i-1)$ to $\varepsilon_{list}$
16:     $i = i + 1$
17: $\varepsilon = \min(\varepsilon_{list})$
18: return $\varepsilon$
```

$$\delta = \min_\lambda \exp(\alpha_{\mathscr{M}}(\lambda) - \lambda\varepsilon)$$

The tail bound converts the moments bound to the $(\varepsilon, \delta)$-differential privacy guarantee and gives us a way to compute $\varepsilon$, given a fixed $\delta$ as:

$$\varepsilon = \min_\lambda \frac{\alpha_{\mathscr{M}}(\lambda) - \log\delta}{\lambda}$$

Assuming that the training is done over multiple epochs, we can fix the sampling ratio $q = \frac{L}{N}$, where $N$ is the number of data points in the dataset, and $L$ is the number of datapoints within a lot. Then, for a Gaussian Mechanism with random sampling, it suffices to compute the probability density function for $\mathscr{N}(0, \sigma^2)$ and $\mathscr{N}(1, \sigma^2)$, denoted as $\mu_0$ and $\mu_1$ respectively. If $\mu = (1-q)\mu_0 + q\mu_1$, we have $\alpha(\lambda) = \log\max(E_1, E_2)$, where:

$$E_1 = \mathbb{E}_{z\sim\mu_0}[(\frac{\mu_0(z)}{\mu(z)})^\lambda]$$
$$E_2 = \mathbb{E}_{z\sim\mu}[(\frac{\mu(z)}{\mu_0(z)})^\lambda]$$

# Chapter 3

# Related Work

## 3.1 Approaches for Privacy and Data Analysis

**Anonymization.** In theory, one could try to anonymize data by stripping personally identifiable information before sharing it. The assumption is that sharing anonymized records can be done freely, since no one knows who the respective record belongs to. In practice, however, this assumption has been disproven on multiple occasions, for numerous datasets. Archie et al. [21] re-identify users from the Netflix Prize dataset by using publicly available IMDb data. This is an even bigger problem for more sensitive data, such as genomes, where re-identification of users has also been proven to be possible. Gymrek et al. [79] demonstrate that recovery of surnames from genomic data donors can be inferred using data publicly available from recreational genealogy databases. Additionally, not even $k$-anonymity, where generalization techniques are used to mask exact values of attributes, are safe against inference attacks [17].

**Aggregation.** Another approach is to share aggregate statistics about a dataset. For example, one can find the number of people in a certain location at a given time in order to determine if the location is considered a point of interest [147]. However, this is also ineffective, due to susceptibility to membership inference attacks. For instance, Pyrgelis et al. [148] show how to determine whether a user is part of aggregate location data. Membership inference attacks have also been demonstrated in other contexts, e.g., genomic data [89, 177].

**Differentially Private Data Release.** In order to provide stronger privacy guarantees, techniques that satisfy differential privacy have been more widely proposed as more effective solutions. Differential privacy, reviewed in Section 2.5, provides a formal mathematical definition that specifies requirements for controlling privacy risk, with several properties (e.g., composition, post-processing, etc.) that facilitate reasoning about privacy and the construction of differentially private algorithms. However, the tension between usability and privacy is inherently complex and application-dependent, and differentially privacy algorithms have often been regarded as providing low utility for researchers, as, e.g., in the case of health data [56].

**Privacy-Preserving Synthetic Data Generation.** To overcome these limitations, another approach is to generate realistic synthetic data using generative models. These yield new samples that follow the same probabilistic distribution of a given training dataset. The intuition is that entities can train and publish the model, in a differentially private way, so that anybody can generate a synthetic dataset resembling the data it was trained on, without exposing the training data itself.

*Imputation models.* One of the first approaches for generating fully synthetic data has been proposed by Rubin [153]. The idea is to treat all observations from the sampling frame as missing data and to input them using the multiple imputation method. Because the synthetic data has no functional link to the original data, it can preserve the confidentiality of participants. However, as discussed in [53], the synthetic data generated this way is subject to inferential disclosure risk when the model used to generate the data is too accurate.

*Statistical models.* Other approaches attempt to generate a statistical model based on the original data [190]. The main idea is to generate a low-dimensional distribution of the original data to help with the data generation process. This approach, combined with differential privacy, aims to provide privacy guarantees to the synthetic data.

*Generative Models.* More recently, generative machine learning models have attracted a lot of attention from the research community. A generative model is a way to learn any kind of data distribution using unsupervised learning, aiming to generate new samples that follow the same probabilistic distribution of a given dataset. Generative models based on neural networks work by optimizing the weights of the connections between neurons by back-propagation techniques. For complex networks, the optimization is usually done by the mini-batch stochastic gradient descent (SGD) algorithm. Generative models can be used in conjunction with differential privacy. This is usually done using a differentially private training procedure, which guarantees that the learned model is differentially private, and thus any synthetic dataset we can derive from it will also guarantee differential privacy.

We also look at seed-based generative models [31], which condition the output of the model based on input data, called the seed. This way, the model will produce synthetic records similar to the seed, which can increase the quality of the output. Because of the high correlation between the output and the seed, privacy tests which provide differential privacy guarantees are introduced.

## 3.2 Genome Privacy

**Re-identification.** Genomic data is hard to anonymize, due to the genome's uniqueness as well as correlations within different regions. For instance, Gymrek et al. [79] demonstrate that surnames of genomic data donors can be inferred using data publicly available from recreational genealogy databases. They also discuss how, through deep genealogical ties, publishing even a few markers can lead to the identification of another person who might have no acquaintance with the one who released their genetic data. In follow-up work, Erlich et al. [64] show that a genetic database which covers only 2% of the target population can be used to find a third-cousin of nearly any individual.

**Membership inference.** Homer et al. [89] present a membership inference attack (MIA) in which they infer the presence of an individual's genotype within a complex genomic DNA mixture. Wang et al. [177] improve on the attack using correlation statistics of just a few hundreds SNPs, while Im et al. [94] rely on regression coefficients. Shringarpure and Bustamante [159] perform membership inference against the Beacon network.[1] They use a likelihood-ratio test to predict whether an individual is present in the Beacon, detecting membership within a Beacon with 1,000 individuals using 5,000 queries. Also, Von Thenen et al. [174] reduce the number of queries to less than 0.5%. Their best performing attack uses a high-order Markov chain to model the SNP correlations, as described in [156]. Note that, as part of the attacks described in this paper, we use inference methods from [156] as our baseline inference methods.

**Data sharing.** Progress in genomics research is dependent on collaboration and data sharing among different institutions. Given the sensitive nature of the data, as well as regulatory and ethics constraints, this often proves to be a challenging task. Kamm et al. [102] propose the use of secret sharing to distribute data among several entities and, using secure multi-party computations, support privacy-friendly computations across multiple entities. Wang et al. [179] present GENSETS, a genome-wide, privacy-preserving similar patients querying system using genomic edit distance approximation and private set difference protocols. Then, Chen et al. [43] use Software Guard Extensions (SGX) to build a privacy-preserving international collaboration tool; this enables secure and distributed computations over encrypted data, thus supporting the analysis of rare disease genetic data across different continents.

**Privacy-friendly testing.** Another line of work focuses on protecting privacy in the context of personal genomic testing, i.e., computational tests run on sequenced genomes to assess, e.g., genetic susceptibility to diseases, determining the best course of treatment, etc. Baldi et al. [29] assume that each individual keeps a copy of their data and consents to tests done in such a way that only the outcome is disclosed. They present a few cryptographic protocols allowing researchers to pri-

[1]Beacons are web servers that answer questions e.g. "does your dataset include a genome that has a specific nucleotide at a specific genomic coordinate?" to which the Beacon responds yes or no, without referring to a specific individual; see: `https://github.com/ga4gh-beacon/specification`.

vately search mutations in specific genes. Ayday et al. [26] rely on a semi-trusted party to store an encrypted copy of the individual's genomic data: using additively homomorphic encryption and proxy re-encryption, they allow a Medical Center to privately perform disease susceptibility tests on patients' SNPs. Naveed et al. [126] introduce a new cryptographic primitive called Controlled Functional Encryption (CFE), which allows users to learn only certain functions of the (encrypted) data, using keys obtained from an authority; however, the client is required to send a fresh key request to the authority every time they want to evaluate a function on a ciphertext. Overall, for an overview of privacy-enhancing technologies applied to genetic testing, we refer the reader to [118].

**Long-term security.** As the sensitivity of genomic data does not degrade over time, access to an individual's genome poses a threat to her descendants, even years after she has deceased. To the best of our knowledge, GenoGuard [92] is the only attempt to provide long-term security. GenoGuard, reviewed in Section 5.1, relies on Honey Encryption [100], aiming to provide confidentiality in the presence of brute-force attacks; it only serves as a storage mechanism, i.e., it does not support selective retrieval or testing on encrypted data (as such, it is not "composable" with other techniques supporting privacy-preserving testing or data sharing). In this thesis, we provide a security analysis of GenoGuard. In parallel to our work, Cheng et al. [44] recently propose attacks against probability model transforming encoders, and also evaluate them on GenoGuard. Using machine learning, they train a classifier to distinguish between the real and the decoy sequences, and exclude all decoy data for approximately 48% of the individuals in the tested dataset.

## 3.3 Honey Encryption

Juels and Ristenpart [100] introduce Honey Encryption (HE) as a general approach to encrypt messages using low min-entropy keys such as passwords. HE, reviewed in Section 2.3.3, is designed to yield plausible-looking ciphertexts, called honey messages, even when decrypted with a wrong password. In a nutshell, it uses a distribution-transforming-encoder (DTE) to encode a-priori knowledge of the message distribution, aiming to provide *message recovery* security against computationally unbounded adversaries. It was originally designed to encrypt credit card information, RSA secret keys, etc. [172].

Message recovery security can be defined as follows [95]: given a message encrypted under a key whose maximum probability of taking on any particular value is at most $1/2^{\mu}$, an unbounded adversary's ability to guess the correct message, even given the ciphertext, is at most $1/2^{\mu}$ plus a negligible amount. However, Jaeger et al. [95] discuss deficiencies of message recovery security as per modern security goals. More specifically, not only they prove the impossibility of known-message attack security in the case of low-entropy keys, but they also mention that schemes meeting message recovery security might actually leak a significant amount of information about the plaintexts, even if the adversary cannot correctly recover the full message with non-negligible probability. Although this serves as an inspiration to our work, note that the context of our evaluation is different, as in the low-entropy setting, we show that a lower bound also applies to the adversary's advantage when partial information from the target sequence is available to the attacker, compared to having pairs of ciphertext and plaintext. Another work studying attacks against HE is that by Cheng et al. [44], which we have reviewed above.

**Honeywords.** Before Honey Encryption [100], Juels and Rivest [101] introduced the concept of "honeywords" to improve the security of password databases. They propose adding honeywords (false passwords) to a password database together with the actual password (hashed with salt) of each user. This way, an adversary who hacks into the password database and inverts the hash function cannot know whether she has found the password or a honeyword.

Wang et al. [176] present an evaluation of the honeyword system [101], finding it to be vulnerable to a number of attacks. More specifically, an adversary that wants to distinguish between real and decoy passwords can do so with a success rate of 30% compared to an expected 5%. In the case of a targeted attack, when the adversary is assumed to know some personal information about the user, they show that the adversary's success rate is further improved to about 60%. Our attacks differ from those in [176], first, as they target the honeywords system [101], while we focus on Honey Encryption [100], and in particular its application to GenoGuard [92]. Moreover, their attack only aims to identify the correct password from a given password pool, while we also examine the case when the correct password is not found within the tried passwords.

## 3.4 Machine Learning and Privacy

In this section, we review relevant related work on synthetic data and MIAs against machine learning models.

**Synthetic Data Initiatives.** In recent years, researchers have focused on the generation of synthetic electronic health records (EHR), aiming to facilitate research in and adoption of machine learning in medicine. Choi et al. [46] use a combination of an autoencoder with GAN model, called medGAN, to generate high-dimensional multi-label discrete data. ADS-GAN [187] uses a quantifiable definition for "identifiability" that is combined with the discriminator's loss to minimize the probability of patient's re-identification, while CorGAN [168] combines convolutional GANs and convolutional autoencoders to capture the correlations between adjacent medical features. Biswal et al. [34] use variational autoencoder to synthesize sequences of discrete EHR encounters and encounter features. Other initiatives focus on generating synthetic data modeled on primary care data [180, 171, 128, 9]. e.g., by utilizing resampling and probabilistic graphical models (Bayesian networks) with latent variables.

Researchers have also explored generating synthetic health patient data to detect cancer and other diseases, e.g., RDP-CGAN [169] combines convolutional GANs and convolutional autoencoders, both trained with Rényi differential privacy [117]. while Goncalves et al. [75] evaluate probabilistic models, classification-based imputation models, and GANs. Specific to genomics are the works we have introduced in Section 6.3.2 and evaluated, in terms of utility and privacy, throughout Section 6.3 [156, 185, 103].

NHS England is one of the first organizations that explored the potential of using synthetic data [128], in order to enable open data release. By using statistical models, they aim to retain the data characteristics while maintaining patient confidentiality. However, before the statistical models are applied, the data goes through a sanitization process to remove some granularity and disclosive information. More recently, the Medicines and Healthcare products Agency (MHRA) has announced the creation of two new synthetic datasets to support the development of medical technologies to fight coronavirus (COVID-19) and cardiovascular disease [9].

**MIAs against Machine Learning Models.** When it comes to the privacy of synthetic data, most models rely solely on differential privacy in order to ensure privacy preserving properties [111, 144, 184, 14]. MIAs have long been studied in the context of machine learning. Shokri et al. [158] present the first attack against discriminative models, aiming to identify whether a data record was used in training, using an approach based on shadow models. They train multiple shadow models, imitating the behavior of the target model, on datasets of similar distribution as the original training data. The inference model is then trained to recognize the model's prediction on data points that were included in the original training data versus data points that were not. Hayes et al. [83] present the first MIA against generative models like GANs; they use a discriminator to output the data with the highest confidence values as the original training data. The main difference between their membership inference attack and our current work is the assumption that, for LOGAN the attacker has knowledge about the size of the original training dataset , $n$, as well as access to a dataset, $D$, with datapoints suspected to be in the original training records. They then parse the dataset $D$ through the discriminator and output the top $n$ confidence values as their guess for the original training data. In contrast, our framework takes a target record and outputs a prediction on whether the target record was used in the synthetic data generation. Hilprecht et al. [85] study MIAs against both GANs and Variational AutoEncoders (VAEs), They evaluate membership inference under two settings: first, from an adversarial actor perspective, who aims to identify individual records used to train the model, and second from a regulatory actor perspective, where a regulator is given two datasets and needs to decide which of the two datasets is a subset of the original training data. Chen et al. [42] propose a generic MIA model against GANs where the attack is formulated as a binary classification task using the distance metric between a query sample and its reconstructed copy from the GAN.

Finally, measurement studies have focused on privacy in machine learning. Jayaraman and Evans [96] evaluate differential privacy to understand the impact that different choices for privacy parameters have on both utility and privacy. Long et al. [113] study membership inference attacks on discriminative models, in order to understand why and how they succeed. They quantify the risk of membership inference of a record with respect to a classifier and its training data using a measure called Differential Training Privacy (DTP). For a given classifier trained on a dataset, the membership leakage of a record is quantified by comparing that classifier's predictions to those of a classifier trained without that specific record. Yeom et al. [186] explore the relationship between privacy, overfitting and influence for machine learning models. More specifically, they formalize privacy leakage in machine learning models, looking at how much an adversary can infer from a model, for two types of attacks, namely membership inference and attribute inference attacks. Jayaraman and Evans [96] perform an evaluation of differential privacy mechanisms for machine learning models. They use the measures previously defined by Yeom et al. [186] and study the privacy leakage arising from relaxation of the differential privacy definitions. Finally, other attacks on machine learning models include memorization attacks [40], model inversion [67, 186], model stealing [170], hyperparameter stealing [175] or property inference attacks [23, 70].

# Chapter 4

# Enabling Anonymous Queries on Rare Disease Discovery Platform

Advances in genome sequencing and genomics are enabling tremendous progress in medicine and healthcare, paving the way to making the prevention, diagnosis, and treatment of diseases tailored to the individual's specific genetic makeup, thus becoming cheaper and more effective. Researchers are also gaining a better understanding, and developing more successful treatments of rare genetic diseases. However, even though sequencing costs have plummeted from billions to thousands of dollars over the past 15 years (see https://www.genome.gov/sequencingcosts/), it is still hard for researchers to gain access to genomic data, especially those pertaining to rare conditions.

Therefore, seamless progress in genomics research hinges on the ability to collaborate and share data among different institutions. Indeed, funding agencies often require that data sharing is considered in grant applications, and a number of initiatives have been announced to gather and share genomic data. For instance, the All Of Us Research Program (formerly known as the Precision Medicine initiative) was launched in the US in 2015, aiming to collect health and genetic data from one million citizens. Similar projects exist elsewhere, e.g., in the UK, Genomics England is sequencing the genomes of 100,000 patients, focusing on rare diseases and cancer. There are also initiatives specifically targeting data sharing, such as the NIH's Genomic Data Commons (GDC), which provides the cancer research community with a unified data repository across cancer genomic studies ([1]).

Aiming to foster collaborations, the Global Alliance for Genomics and Health (GA4GH) [73] was established, with core funding from NIH, Wellcome, and Canada's CanShare, with the explicit goal of making data sharing between institutes simple and effective. The GA4GH has developed several platforms, e.g., the Beacon Project [72], allowing researchers to search if a certain allele exists in a database of genomic data, as well as the Matchmaker Exchange (MME) [142], which facilitates rare disease discovery.

In this chapter, we focus on the latter; The MME platform connects multiple distributed databases through an API and allows researchers to query for genetic variants in other databases in the network. That is, MME acts as a portal supporting simultaneous querying over multiple databases that are members of the exchange. More specifically, MME allows a researcher to query a specific gene, e.g., "AP3B2" (a gene where rare mutations have been linked to early-onset epileptic encephalopathy). If a match is found, the researcher is notified of all matches within all databases in the MME, and can get in touch with the user that submitted the case on which a match is generated. Note that, querying a gene really implies querying a known rare variation of that gene.

However, researchers might be reluctant to use the platform since the queries they make are revealed to other researchers, and this exposes what they are working on and what kinds of patients they might have, ultimately resulting in loss of privacy and competitive advantage. Indeed, MME currently requires researchers to submit a registration application to be given access to the platform, with the goal of preventing misuse of the system, thus, queries made on this platform are not anonymous and are revealed to all other researchers with an interest in the same gene.

**Problem Statement.** This motivates the need to support *anonymous querying* on MME, so that a researcher's interest in a specific gene is not broadcast, but only communicated to relevant contacts, i.e., researchers with same interests or willing to collaborate. To this end, we present AnoniMME, a framework letting researchers anonymously query a gene within the MME, without violating any of MME's current functionalities and requirements. We build AnoniMME using a cryptographic primitive called Reverse Private Information Retrieval, using a model similar to that presented by the anonymous messaging system Riposte [52], while creating queries and implementing the same functionalities as in MME. In other words, researchers can perform anonymous queries to the federated platform, in a multi-server setting, by writing their query, along with a public encryption key, anonymously, in a public database. We also construct AnoniMME to support responses, so that other researchers can respond to queries by providing their encrypted contact details.

**Solution Intuition.** We build queries in regular epochs, where the length of each epoch is based on the number of write requests. In order to anonymously write to the database, the user selects a random row of the the database, and splits the query, containing the gene and her public key, into shares, one for each server (which we denote as *node* servers). This way, the node servers cannot learn anything about the write request, if at least one of the them is honest. Then, a *master* server can gather queries that have been collected during an epoch from the node servers and collate them together to recover and publish the actual queries. The MME matching system can then be used in order to generate matches for the queries, in the usual manner, and contact details of other researchers/clinicians can be exchanged, encrypted using the public key, and published in the same row as the queried gene, in an adjacent column.

To demonstrate the practicality of AnoniMME, we implement and evaluate our prototype experimentally in Section 4.2.3. We do so in two different settings, one involving two node servers and a master server, and another involving six node servers (and a master server). In both settings, the nodes collect write requests during an epoch, and then forward them to the master server which collates them and publishes the final database.

**Contributions.** In summary, this chapter makes several contributions:

1. We present AnoniMME, a framework enabling anonymous queries within the Matchmaker Exchange (MME), without breaking any of its current security and functionality requirements.

2. We build AnoniMME from Reverse PIR [52], using an information-theoretic approach, extending queries to support public key encryption of contact details, and adding a response phase so that users can also anonymously reply to queries.

3. We show, experimentally, that AnoniMME is efficient and scalable, and can bring anonymity to MME with low overhead. Therefore, we are confident that it can be deployed in the wild and further encouraging researchers to share genomic data.

## 4.1 Approach

In this section, we first introduce the Matchmaker Exchange together with its functionalities, and then we proceed in presenting our approach for developing an anonymity overlay to it.

### 4.1.1 Matchmaker Exchange

As mentioned, the Global Alliance for Genomics and Health (GA4GH) was established, in 2013, aiming to support simple mechanisms for sharing data between institutes. The GA4GH has developed and deployed various systems, including the Matchmaker Exchange (MME) [142], which facilitates rare disease gene discovery and constitutes the main focus of our work. MME is a federated platform that facilitates the identification of cases with similar phenotypic and genotypic profiles through a standardized Application Programming Interface (API). Essentially, it enables searches in multiple databases, without having to query all of them separately or deposit data in each of them. As of March 2018, it involves seven organizations with full member status (AGHA Patient Archive, DECIPHER, GeneMatcher, Matchbox, Monarch Initiative, MyGene2, and PhenomeCentral), and eight additional participant organizations.

The Matchmaker Exchange Application Programming Interface (MME API) [38] fully specifies the data format and the protocol for querying databases to identify individuals with similar phenotypic profiles and genetic variations. To ensure the accuracy of the patient comparison, similar phenotypes are determined by matching identical or ontologically similar with the Human Phenotype Ontology (HPO). The MME API also specifies the format of both the query, which is sent to participating databases (called "matchmaking service") and the response, which contains information about matching individuals in the remote database. It is implemented under a query-by-example methodology: a user can query a specific gene, e.g., "AP3B2," and she will be notified of all matches within all databases in the MME. Note that querying a gene really implies querying a known rare variation of that gene. If a match is found, the user receives a Case ID for the match, information about the user that submitted the case on which a match is generated, such as name, institution and email address, as well as the corresponding candidate gene or phenotype. In order to query the platform, users must be registered with one of the member databases, and have a clinician/researcher account. Some of the member databases allow for patient/family registrations as well, however, the submissions made by these type of users are excluded from matching via MME, due to the current MME rules.

The query protocol is illustrated in Figure 4.1. A user, Bob, sends the metadata (i.e., Case ID, submitter information) as well as the patient data (gene and/or phenotype) to Database B. Another user, Alice, submits a similar case to Database A; Database A then sends an MME API match request to Database B, which performs the match and returns a list of scored patients, along with relevant metadata, to Database A. After receiving the match results, Database A informs Alice, providing contact information for Bob. The result of querying MME yields a list of matches, where each match has a *patient* object, i.e., the information on the matched patient, consisting of the same information as described in the query, and a *score* object. The scoring of the patients is done according to how well the results patient matches the query patient, i.e., it is a numerical value in the range $[0,1]$, where 0.0 is a poor match and 1.0 a perfect match. Given the current functionality of MME, and the fact that queries are broadcast to all users which have made with an interest in the same gene, we propose the addition of functionality that would allow users to make

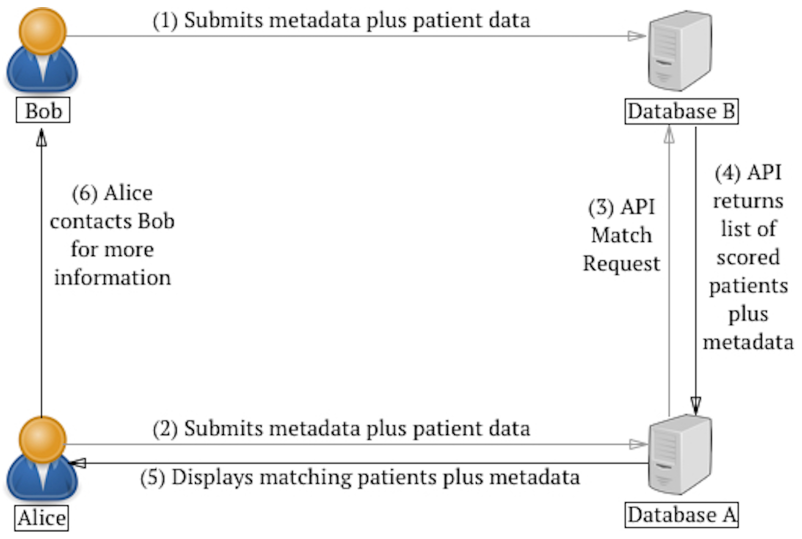

(1) Submits metadata plus patient data
Bob
Database B
(6) Alice contacts Bob for more information
(3) API Match Request
(4) API returns list of scored patients plus metadata
(2) Submits metadata plus patient data
(5) Displays matching patients plus metadata
Alice
Database A

**Figure 4.1:** Visual representation of a MME query sequence.

*anonymous* queries on the platform.

### 4.1.2 Entities and Operations

Our proposed framework involves the following entities:

*Querying Users:* researchers/clinicians who query the system to find other users that have patients with a rare mutation or an interest in the same gene. As discussed later, they generate a write request specifying the row at which their query, i.e., the gene of interest and their public key, will be processed.

*Responding Users:* researchers/clinicians replying to an existing query. They use the public key of a querying user to encrypt their contact details and generate a write request for the same row as the gene of interest including their (encrypted) contact details.

*Nodes:* the servers collecting write requests from the users. These are aggregated until the end of an epoch, based on the maximum number of write requests. Each node server can be run by one of the current MME members.

*Master Server:* a server that gathers the databases from each node at the end of an epoch, and publishes the database with all the write requests revealed. The master server role can also be assigned to one of the existing MME members, and can be reassigned to another member at the end of each epoch.

Overall, AnoniMME implements the following operations:

*Query Write Request:* On input row $i$, query gene $X$, and public key $PK$, a querying user generates $n$ write requests, one for each node. Each write request is generated by encoding the gene and the public key into $n$ vectors, so that all of them combined will write the gene/public key at index $i$.

*Query Response Request:* On input row $i$, encrypted contact details $c$, a responding user generates $n$ write requests, one for each node. Write requests are generated, once again, by encoding the encrypted contact details into $n$ vectors.

*Database Collation:* On input $n$ databases, the master server collates them into one final database, and publishes it.

### 4.1.3 Security Model

AnoniMME aims to guarantee the following three security goals:

*1. Correctness.* When all nodes execute the protocols correctly and send data to the master server at the end of an epoch, the resulting database contains all the write requests processed as if the requests were directly applied to the final database.

*2. Anonymous Write.* The probability that an adversary guesses at which particular row a user has written is only negligibly better than random guessing.

*3. Disruption Resistance.* An adversary controlling $n$ users can make at most $n$ write requests (i.e., there is a limit to the number of write requests each user can make during an epoch).

**Threat Model.** We assume that the users of the system are untrusted, and may collude with the nodes, the master server, or other users in order to violate the security properties of the system. Both the master server and the nodes are trusted for availability and to follow the protocol correctly, under the assumption that at least one of the nodes is honest (i.e., does not collude with other nodes). We do not consider external adversaries, since their actions can be mitigated via standard network security techniques (i.e., using a secure and authenticated communication channel). Finally, note that the security model of AnoniMME mirrors that of Riposte [52].

### 4.1.4 A First Attempt

We now present a first attempt at instantiating AnoniMME, and discuss its limitations, which we address in the actual construction of AnoniMME presented in Section 4.2.2.

**Intuition.** We start by attempting to build from a simple extension of Reverse Private Information Retrieval (Reverse PIR) [52]. More specifically, we implement the query phase using the same mechanism of Riposte, i.e., we let users anonymously submit the gene of interest, along with their public key, with a "write request." We then add a response phase, allowing users with an interest in the same gene to respond—specifically, by encrypting their contact information using the public key contained in the query, and adding it to another write request.

In the following, we present a construction assuming the presence of 2 servers ($S_1$ and $S_2$) and a database with $l$ rows.

**Query phase.** Assume user A wants to anonymously query gene $X_A$. She builds a write request, consisting of $(X_A, PK_A)$, where $PK_A$ is her public key, and chooses(randomly) to write this at row $i$ in the database. More specifically, she picks $2l$ random numbers, $r_1, r_2, \ldots, r_l$ and $s_1, s_2, \ldots, s_l$, where $l$ is the size of the database. The query write request vectors are constructed as follows:

$$\begin{aligned}
v_1 &= (r_1, r_2, \ldots, r_i + X_A, \ldots, r_l), \\
v_1' &= (s_1, s_2, \ldots, s_i + PK_A, \ldots, s_l), \\
v_2 &= (-r_1, -r_2, \ldots, -r_i, \ldots, -r_l), \\
v_2' &= (-s_1, -s_2, \ldots, -s_i, \ldots, -s_l).
\end{aligned}$$

Note that $v_1 + v_2 = X_A \cdot e_i$, and $v_1' + v_2' = PK_A \cdot e_i$, where $e_i$ denotes the unit vector with 0's at all positions except at position $i$, where it is equal to 1, and thus the construction is correct. Then, A sends $(v_1, v_1')$ to $S_1$, and $(v_2, v_2')$ to $S_2$. Due to the fact that the user needs to send vectors of the size of the database ($l$) to each of the servers, the bandwidth overhead of the querying phase for each user is $O(l)$.

Our construction also has similarities with trivial secret sharing where all shares are required in order to recover the secret. As for secret sharing, the random numbers chosen by the users could also be binary numbers of the same length as the binary representation of the query gene/ private key and use XOR as the linear operation, or the random numbers can random integers with well defined overflow semantics from any field and use any linear operator in that field.

Write requests are collected until the end of an epoch, when the servers combine their local states and publish the database with the queries. Note that the number of write requests collected during an epoch cannot be more than the size of the database. As long as the two servers do not collude, none of them can reconstruct what any given user has written, i.e., none of the servers can recover the gene or public key of the user sent in the write request. Also, in order to achieve disruption resistance, one can limit the number of queries to one per user for each phase of the epoch.

**Response phase.** After the database with the queries is published, the response phase begins. Here we can rely on MME's algorithm to generate matches on existing MME data, and simply extend it to encrypt the contact details of the relevant users with an interest in the same gene. This would be inline with the current privacy policy of the MME, as contact details of researchers with an interest in the same gene are already shared.

Users can also be given an option to voluntarily provide their contact details as follows. If user B notices that another researcher (user A) has an interest in the same gene X, say at row $i$ of the database, she gets A's public key $PK_A$, and encrypts her contact information ($C_B$) under $PK_A$ and generates a write request as a share of $Enc_{PK_A}(C_B)$, in a similar manner to the first epoch. More specifically, she chooses random $r'_1, \dots, r'_l$ and forms the following vectors:

$$u_1 = (r'_1, \dots, r'_i + Enc_{PK_A}(C_B), \dots, r'_l),$$
$$u_2 = (-r'_1, \dots, -r'_i, \dots, -r'_l)$$

User B then sends $u_1$ to server $S_1$ and $u_2$ to $S_2$. At the end of this epoch, the results are being published in a column adjacent to the queried gene and the public encryption key. The querying users can use the database to find the row of interest (in this case $i$), decrypt the contact details, and get in touch with the responding users. Similar to the query phase, the response phase has a bandwith overhead $O(l)$. Note that one of the key differences between the query and the response phase is the choice of where to write in the database. For querying users, they choose the row at which to write their query randomly from the rows of the database, while for the responding users, they choose the row at which to write their response based on where the message they want to respond to is in the database.

**Correctness and Security.** It is straightforward to see that the construction is correct, since, if all nodes execute the protocols correctly the result of combining all their local database states at the end of an epoch by the master server will result in revealing all the write requests processed. An adversary's advantage of guessing at which a certain user has written in the final database is the same as random guessing, hence, the construction guarantees anonymous writes. Disruption resistance can be also achieved in a straightforward manner since MME requires users to register on one of the databases, so they can allow maximum one write request per registered user per epoch.

**Limitations.** Alas, this construction has the following limitations:

1. *Collisions:* They might occur for writes generated by honest users, which all want to write at the same row;
2. *Maliciously-formed write requests:* A malicious user can easily send a malformed request to the servers, making all the data within the database non recoverable.

In this chapter we focus on methods for collision handling, and leave the maliciously-formed write requests as part of future work. However, previous work done on Reverse PIR [52] suggests that the maliciously-formed write requests can be handled by either introducing an extra party which acts as an audit server, and must not collude with the other servers, or by applying zero-knowledge techniques which would allow clients to prove that their write requests are correctly formed.

**Discussion.** We choose to base our construction on Reverse PIR, for several reasons. First, MME is described as a genomic discovery platform, which means that the querying user will not know the intended recipient of his query. Hence, anonymous systems such as mixes, which requires a recipient for the message sent would not be a suitable choice in this setting. Second, in comparison to Reverse PIR, anonymity systems such as Tor[165] do not protect against passive network adversaries who can correlate network traffic flows. However, it would be possible to combine both Reverse PIR and Tor (i.e., clients submit the queries to a platform using reverse PIR via the Tor network) , which would mean that even if all servers in the Reverse PIR setting colluded, the could not learn which user wrote which message without also breaking the anonymity of Tor.We leave the study and implementation of this system to future work. Third, even though Invertible Bloom Lookup Tables(IBLT) [77] might seem like a good candidate for finding the users which have an interest in the same gene, it would essentially have the similar functionality to the current mode of operation of MME, with the main difference being that two users could chose to compare their database records to look for similar records. While the users could choose to compare records only with other users they trust, and thus eliminate some of the privacy concerns, their queries could still be vulnerable to intersection attacks. Potentially combining IBLT with secret sharing could be a viable alternative for obtaining the desired security properties, however, we also leave this to future work. Finally, e-voting systems could be used as alternative building block for our system, offering similar privacy guarantees as Reverse PIR, but we favored the latter construction due to lower overhead compared to the zero-knowledge proofs used in e-voting.

## 4.2 Methods

In this section we provide methods for collision handling for our first attempt and use it to provide a description of the n-server protocol. We also evaluate the proposed method in terms of time and bandwidth required in order to asses the feasibility of the proposed construction.

### 4.2.1 Handling Collisions

As discussed previously, collisions might occur whenever multiple users want to write at the same row. Aiming to address them, we set the database size to be large enough to accommodate write requests at a 95% non-collision rate. In other words, 5% of the queries will likely fail due to collisions and will need to be re-submitted.

#### 4.2.1.1 Minimizing collisions

Our intuition is to follow a "balls and bins" approach, i.e., if we throw $m$ balls uniformly and randomly into the $l$ bins, we can estimate how many bins will contain exactly one ball. In our model, we can associate write requests to the $m$ balls and the rows of the database to the $l$ bins. Let $B_{ij}$ be the event that ball $i$ falls into bin $j$: for all $i$ and $j$, we have $\Pr[B_{ij}] = \frac{1}{l}$. Then, let $O_j^{(1)}$ be the event that exactly one ball falls in bin $j$. We have that:

$$\Pr[O_j^{(1)}] = \frac{m}{l}(1-\frac{1}{l})^{m-1} \approx \frac{m}{l} - \left(\frac{m}{l}\right)^2 + \frac{1}{2}\left(\frac{m}{l}\right)^3$$

using the binomial theorem and ignoring low order terms. Then, $l\Pr[O_j^{(1)}]$ is the expected number of bins with exactly one ball, i.e., the expected number of messages successfully received. Dividing by $m$, we get the expected success rate as

$$E[SuccesRate] = \frac{l}{m}\Pr[O_j^{(1)}] \approx 1 - \frac{m}{l} + \frac{1}{2}\left(\frac{m}{l}\right)^2$$

Thus, for a 95% expected success rate, we need $l \approx 19.5m$.

In AnoniMME, in order to set the size of the database, we need to estimate the expected number of write requests for each epoch. Looking at the three MME members which show statistics on the number of users, we find that GeneMatcher has $4,066$ registered users, MyGene2 345 registered families, and Decipher 247 registered projects (users have to be part of a project in order to join Decipher) as of November 2017. This yields an average of approximately $1,550$ users per database.

Assuming that this is representative of the number of users for all MME databases, we can approximate the total number of users to be in the order $10,000$. We also need to estimate how many users make queries in each epoch: assuming $5\%$ of users do so at each epoch, each epoch can run for 500 queries, yielding a database of size $l \approx 10,000$. Further, note that we design AnoniMME's write request so that the row number at which we write is determined at random, given the number of write requests in the epoch as well as the database size, in order to avoid biases in choosing rows. This method, however, does not provide any way to recover in the case where a collision occurs, in that case the queries are irrecoverable, and the users would need to resubmit their queries in a future epoch.

#### 4.2.1.2 Recovering from collisions

We also use a simple technique for recovering from collisions if/when these occur. Assume $\alpha$ messages have been written at row $i$, i.e., we have $a = m_1 + m_2 + \ldots + m_\alpha$. Inspired by [52], we can modify the way in which the queries are built to recover each of the individual message $m_j$, for $1 \le j \le \alpha$; specifically, we can use a system of $\alpha$ equations, which allows us to solve for each of the colliding messages. Without loss of generality, we consider the case $\alpha = 2$ and explain how to recover from collisions occurring for the gene name, but similar methods can be used for $\alpha > 2$ and to recover public key and/or encrypted contact details. When a collision occurs at row $i$, we have an entry $a = X_A + X_B$, where $X_A$ is the gene sent by user $A$, and $X_B$ is the gene sent by user $B$. If, rather than just sending the queried gene $X$, users send $(X, X^2)$, we can recover $X_A$ and $X_B$ by solving a system of two equations with two variables.

In this case, we also compute the size of the database needed for an expected success rate as follows:

$$E[SuccessRate] = \frac{l}{m}\Pr[O_j^{(1)}] + \frac{2l}{m}\Pr[O_j^{(2)}],$$

where $l\Pr[O_j^{(1)}]$ is the expected number of rows with exactly one write request applied to them, computed as before, and $2l\Pr[O_j^{(2)}]$ is the expected number of rows with exactly two write requests applied to them. Computing $\Pr[O_j^{(2)}] = \binom{m}{2}\frac{1}{l^2}(1-\frac{1}{l})^{m-2}$, we obtain the value of the expected success rate as:

$$E[SuccessRate] \approx 1 - \frac{1}{2}\left(\frac{m}{l}\right)^2 + \frac{1}{3}\left(\frac{m}{l}\right)^3.$$

In this case, for an epoch of $m$ write requests, with a 95% expected success rate, we need a database with $l' \approx 2.7\ m$ cells (two columns and $l = \frac{l'}{2}$ rows ). This implies that with 500 write requests per epoch, the database needs $l' \approx 2.7 \cdot 500 = 1{,}350$ cells for each vector.

We now generalize for any value of $\alpha$. Users submit $X, X^2, \ldots, X^\alpha$ for any gene $X$ to be queried. This allows us to recover from an $\alpha$-way collision as, in that case we obtain a system of $\alpha$ equations with $\alpha$ variables. The expected success rate is:

$$\begin{aligned} E[SuccessRate] =& \frac{l}{m}\Pr[O_j^{(1)}] + \frac{2l}{m}\Pr[O_j(2)] + \\ &+ \ldots + \frac{\alpha l}{m}\Pr[O_j^{\alpha}] \end{aligned}$$

where $l\Pr[O_j^{(k)}]$ is the expected number of rows with exactly $k$ write requests applied to them. Each $\Pr[O_j^{(k)}]$ is computed as $\Pr[O_j^{(k)}] = \binom{m}{k}\frac{1}{l^k}(1-\frac{1}{l})^{m-k}$. Hence, we obtain:

$$E[SuccessRate] \approx 1 + \frac{(-1)^{\alpha+1}}{\alpha!}(\frac{m}{l})^{\alpha} + \frac{(-1)^{\alpha+2}}{(\alpha+1)!}(\frac{m}{l})^{\alpha+1}$$

We solve this equation for $l$, given the expected success rate $E[SuccessRate]$, the collision recovery factor $\alpha$ and $m$ the number of write requests to be written in a certain epoch. If this method is used throughout both epochs, colliding requests from the query phase will have to be recovered before the response phase can begin.

Due to the nature of our query/response model, we can expect collisions to occur more often in the response phase. Hence, we will build the system using different collision recovery factors $\alpha_q$ for the query phase and $\alpha_r$ for the response phase, with $\alpha_r \geq \alpha_q$.

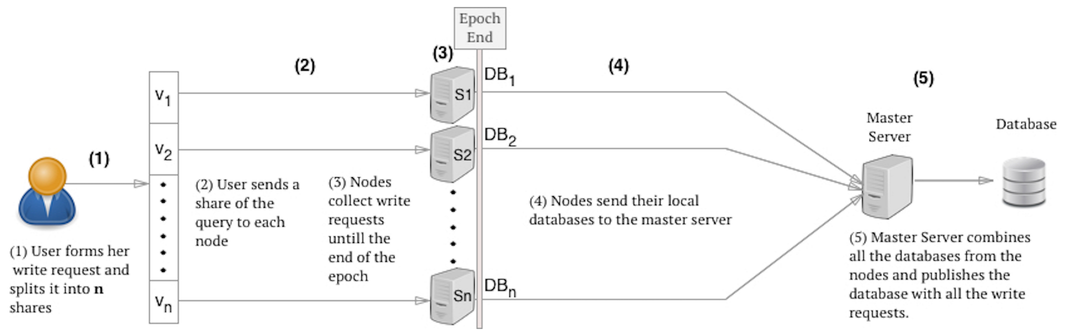

Epoch End
(1)
(2)
(3)
(4)
(5)
v1
v2
vn
S1
S2
Sn
DB1
DB2
DBn
Master Server
Database
(1) User forms her write request and splits it into n shares
(2) User sends a share of the query to each node
(3) Nodes collect write requests untill the end of the epoch
(4) Nodes send their local databases to the master server
(5) Master Server combines all the databases from the nodes and publishes the database with all the write requests.

**Figure 4.2:** *n*-server write request processing. At the end of the epoch the Master Server publishes the database with all the write requests and the nodes will be reset to hold an empty database.

### 4.2.2 N-server Construction

We now present the generalized model for the case with $n$ servers and a database with $l$ rows. We use collision parameters $\alpha_q$ and $\alpha_r$ for the query and response phase, respectively. The various steps of the construction are illustrated in Figure 4.2.

**Query phase.** Assume user A wants to query gene $X_A$, but does not want to reveal that she is the person querying it. As in the construction presented in Section 4.1.4, A builds her write request, consisting of $(X_A, PK_A)$, where $PK_A$ is her public key, aiming to write at row $i$ in the database. She picks random numbers $r_{1,1},\ldots,r_{1,l},r_{1,l+1},\ldots r_{1,l\alpha_q},r_{2,1},\ldots,r_{n,l\alpha_q}$ and $r'_{1,1},\ldots,r'_{1,l},r'_{1,l+1},\ldots,r'_{1,l\alpha_q},r'_{2,1},\ldots,r'_{n,l\alpha_q}$, where $l$ is the size of the database, $n$ the number of nodes the write request will be sent to, and $\alpha_q$ the number of allowed collisions. The query write request vectors are then constructed as follows:

$$
\begin{aligned}
v_{1,1} &= (r_{1,1}, r_{1,2}, \ldots, r_{1,i} + X_A, \ldots, r_{1,l}) \\
v'_{1,1} &= (r'_{11}, r'_{1,2}, \ldots, r'_{1,i} + PK_A, \ldots, r'_{1,l}) \\
v_{1,2} &= (r_{1,l+1}, \ldots, r_{1,l+i} + X_A^2, \ldots, r_{1,2l}) \\
v'_{1,2} &= (r'_{1,l+1}, \ldots, r'_{1,l+i} + PK_A^2, \ldots, r'_{1,2l}) \\
&\vdots \\
v_{1,\alpha_q} &= (r_{1,l(\alpha_q-1)+1}, \ldots, r_{1,l(\alpha_q-1)+i} + X_A^{\alpha_q}, \ldots, r_{1,l\alpha_q}) \\
v'_{1,\alpha_q} &= (r'_{1,l(\alpha_q-1)+1}, \ldots, r'_{1,l(\alpha_q-1)+i} + PK_A^{\alpha_q}, \ldots, r'_{1,l\alpha_q}) \\
&\vdots \\
v_{2,1} &= (r_{2,1}, r_{2,2}, \ldots, r_{2,i}, \ldots, r_{2,l}) \\
v'_{2,1} &= (r'_{2,1}, r'_{2,2}, \ldots, r'_{2,i}, \ldots, r'_{2,l}) \\
&\vdots \\
v_{n,1} &= -(r_{1,1}, r_{1,2}, \ldots, r_{1,i}, \ldots, r_{1,l}) - \sum_{j=2}^{n-1} v_{j,1} \\
v'_{n,1} &= -(r'_{11}, r'_{1,2}, \ldots, r'_{1,i}, \ldots, r'_{1,l}) - \sum_{j=2}^{n-1} v'_{j,1} \\
&\vdots \\
v_{n,\alpha_q} &= (r_{1,l(\alpha_q-1)+1}, \ldots, r_{1,l(\alpha_q-1)+i}, \ldots, r_{1,l\alpha_q}) - \sum_{j=2}^{n-1} v_{j,\alpha_q} \\
v'_{n,\alpha_q} &= (r'_{1,l(\alpha_q-1)+1}, \ldots, r'_{1,l(\alpha_q-1)+i}, \ldots, r'_{1,l\alpha_q}) - \sum_{j=2}^{n-1} v'_{j,\alpha_q}.
\end{aligned}
$$

The querying user A ends $(v_j, v'_j)$ to server $j$ for each $j$, $1 \leq j \leq n$, where $v_j = (v_{j,1}, \ldots, v_{j,\alpha_q})$, and $v'_j = (v'_{j,1} \ldots, v'_{j,\alpha_q})$ . We also consider the special case of $\alpha_q = 1$, when there is no recovery for collisions, but, instead, we adjust the database size according to the minimizing collisions case. The servers collect write requests until the end of the epoch and then send their local databases to the master server, which will combine them to reveal the database.

**Response Phase.** As the database with the queries is published, the response phase begins. As discussed in Section 4.1.4, we can rely on MME's algorithm to generate matches on existing data from the platform, encrypt the contact details of the relevant users with an interest in the same gene, and extend it to allow for voluntary responses. More specifically, user B can add their contact details $C_B$ by sending a write request as a share of $c = Enc_{PK_A}(C_B)$, in a similar manner to the first epoch. That is, first, she picks random $s_{1,1},\dots,s_{1,l},s_{1,l+1},\dots,s_{1,l\alpha_r},s_{2,1},\dots,s_{n,l\alpha_r}$ and forms the following vectors:

$$\begin{aligned}
u_{1,1} =& (s_{1,1},\dots,s_{1,i}+c,\dots,s_{1,l}),\\
u_{1,2} =& (s_{1,l+1},\dots,s_{1,l+i}+c^2,\dots,s_{1,2l}),\\
&\vdots\\
u_{1,\alpha_r} =& (s_{1,l(\alpha_r-1)+1},\dots,s_{1,l(\alpha_r-1)+i}+c^{\alpha_r},\dots,s_{1,l\alpha_r})\\
u_{2,1} =& (s_{2,1},\dots,r'_{2,i},\dots,s_{2,l})\\
&\vdots\\
u_{n,1} =& -(s_{1,1},s_{1,2},\dots,s_{1,i},\dots,s_{1,l}) - \sum_{j=2}^{n-1} u_{j,1},\\
&\vdots\\
u_{n,\alpha_r} =& -(s_{1,l(\alpha_r-1)+1},\dots,s_{1,l(\alpha_r-1)+i},\dots,s_{1,l\alpha_r}) + \sum_{j=2}^{n-1} u_{j,\alpha_r}
\end{aligned}$$

User B then sends $u_j = (u_{j,1},\dots u_{j,\alpha_q})$ to server $S_j$. At the end of this epoch, the results are being published in a column adjacent to the queried gene and the public encryption key. In case of collisions, the individual ciphertexts can be recovered up to $\alpha_r$ collisions. Finally, the querying users can use the database to find the row of interest (in this case $i$) and decrypt the contact details received and contact the person.

### 4.2.3 Experimental Evaluation

We now present an experimental evaluation of AnoniMME, aiming to demonstrate its practicality for real-world deployment.

We have implemented the $n$-server construction (Section 4.2.2) using Python

3.6 and evaluated our prototype on a Macbook Pro running MacOS Sierra 10.12.6 and equipped with a 2.7GHz Intel i5 processor, and 16GB of RAM. Experiments are performed in two different settings, with two and six node servers, respectively, and always averaged over 1,000 executions. We also use three different epoch sizes, namely, 100, 500, and 1,000 write requests per epoch during the query phase. For the response phase, we keep the database size fixed from the query phase. Overall, we evaluate running times needed to generate the write requests and the bandwidth overhead supporting the recovery of 2, 5, and 10 colliding messages, all on the client side (i.e. one request per epoch).

The servers run Flask with RESTful interface, so we use HTTP requests to send the messages, and the payload is built in JSON, therefore, we measure, in bytes, the size of the JSON payload (plus HTTP headers) to estimate the total bandwidth required for sending write requests.

On the client side, the cryptographic layer includes generating public/private keys (done only once) and building the vectors to be sent to the $n$ servers as part of the write request, which incurs $\mathrm{O}(n)$ complexity. Gene name and contact details are assumed to be no longer than 64 characters, while random numbers used for vector generation during query phase are up to 1,024 bits long, for $\alpha_q \in \{1,2\}$ and $\alpha_r = 2$. For the response phase, the length of the random values varies according to the collision recovery factor $\alpha_r$. For $\alpha_r = 5$, their length is 2,560 bits, while for $\alpha_r = 10$ it is 5,120.

Finally, note that plausible gene queries are generated using the set of gene symbols (e.g, "BRCA2") from http://gfuncpathdb.ucdenver.edu/iddrc/iddrc/data/officialGeneSymbol.html.

#### 4.2.3.1 Two Node Servers

We start with the setting involving two node servers and a master server, considering epochs of size 100, 500, and 1,000. As mentioned above, we evaluate bandwidth overhead and running times required for query and response write requests.

The database size required for each of the three test cases is calculated according to the method presented in Section 4.2.1 for minimizing collisions, thus,

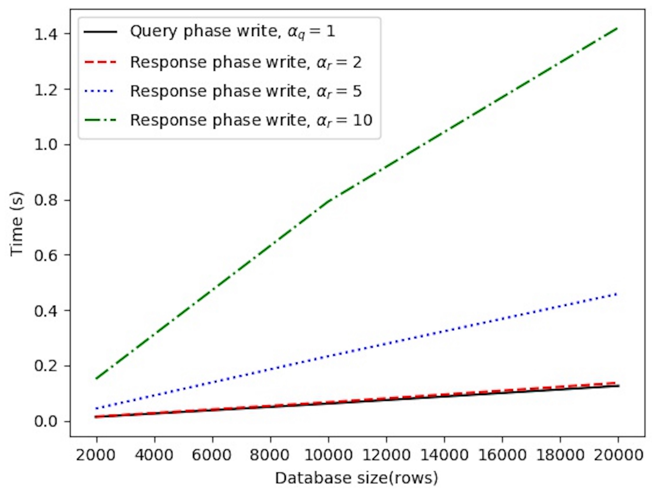


**Figure 4.3:** Two nodes running times for query write request, response write request with recovery from 2 collisions, response write request with recovery from 5 collisions, response write request with recovery from 10 collisions.

$l = 19.5m$, where $l$ denotes the number of rows required and $m$ is the number of write requests for the epoch. It follows that the $l$ amounts to 2,000, 10,000, and 20,000 rows for $m$ equal to 100, 500, and 1,000, respectively.

Running times for both the query write and the response (considering $\alpha_r \in \{2, 5, 10\}$) are shown in Figure 4.3. Overall, we find that, during the query phase, with a database size of 2,000 rows, it takes approximately 0.014s to generate vectors in our testbed. Running times scale linearly, i.e., it takes 0.062s with 10,000 rows and 0.126s with 20,000 rows. The bandwidth overhead, shown in Figure 4.4, ranges from 2.5MB for the smallest database size to 25MB for the largest case considered in our test cases, which can be considered an acceptable amount of traffic expected from the client side.

For the response phase, we find that, when $\alpha_r = 2$, the results are similar to the query phase since responding users need to generate two vectors in order to allow collision recovery, same as for the querying user. When $\alpha_r$ equals 5 or 10, we notice an increase in both running times and bandwidth. Nonetheless, computational complexity is still acceptable, since, even with the largest database size, write request generation takes less than 0.5s for $\alpha_r = 5$ and less than 1.5s for $\alpha_r = 10$. Communi-

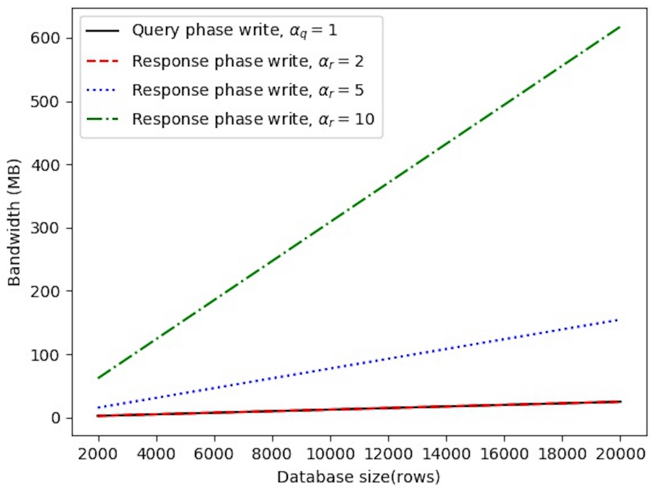


**Figure 4.4:** Two nodes bandwidth averages for query write request, response write request with recovery from 2 collisions, response write request with recovery from 5 collisions, response write request with recovery from 10 collisions.

cation overhead, on the other hand, increases to 160MB and 617MB, respectively, with the largest database size.

However, one can adjust the collision minimization parameter so that 10-way collision recovery is not needed.

#### 4.2.3.2 Six Node Servers

We also experiment with an instantiation of AnoniMME using six node servers, thus mirroring the current MME setting, which involves seven members. Once again, we consider three settings (100, 500, and 1,000 write requests per epoch), and obtain the resulting database size based on the recovery from collisions method discussed in Section 4.2.1. We support recovery from two colliding messages for the query phase, i.e. $\alpha_q = 2$. Therefore, the number of rows required is $l = \frac{2.7m}{2}$, where $m$ is the number of write requests for the epoch, thus, $l$ equals 135, 675, and 1,350 for $m$ = 100, 500, and 1,000, respectively. As per the response phase, we run tests with different values $\alpha_r \in \{2, 5, 10\}$, considering the database size fixed as for the query phase.

Once again, we estimate running times (see Figure 4.5) and the bandwidth overhead (see Figure 4.6). Even though this requires more vectors to be generated

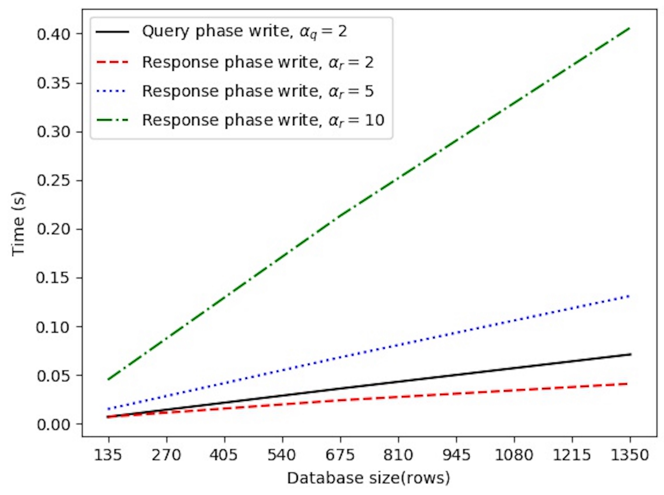


**Figure 4.5:** Six nodes running times for query write request, response write request with recovery from 2 collisions, response write request with recovery from 5 collisions, response write request with recovery from 10 collisions.

by the users compared to the two-node setting (cf. Section 4.2.3.1), we observe a considerable decrease in both running times and bandwidth overhead for the same epoch sizes due to the decreased number of rows in the database. Specifically, computational complexity is again linear over all test cases, but the write request generation taking less than half the time. There is also a big improvement in terms of communication complexity: even in the most bandwidth-heavy case (i.e., $\alpha_r = 10$), with 1,000 write requests per epoch, we observe a five-fold improvement, with bandwidth decreasing from 617MB to 125MB.

On the other hand, the query phase is less efficient than the response phase (with $\alpha_r = 2$), compared to the two-node setting, since the querying user now has to generate two vectors for each gene so that collision recovery is possible, hence, four vectors in total; whereas, the responding user only generates two vectors.

## 4.3 Discussion

This chapter presented AnoniMME, a framework geared to bring anonymity to Matchmaker Exchange (MME) platform. Specifically, AnoniMME supports anonymous queries, by relying on Reverse PIR, while mirroring the functionalities of

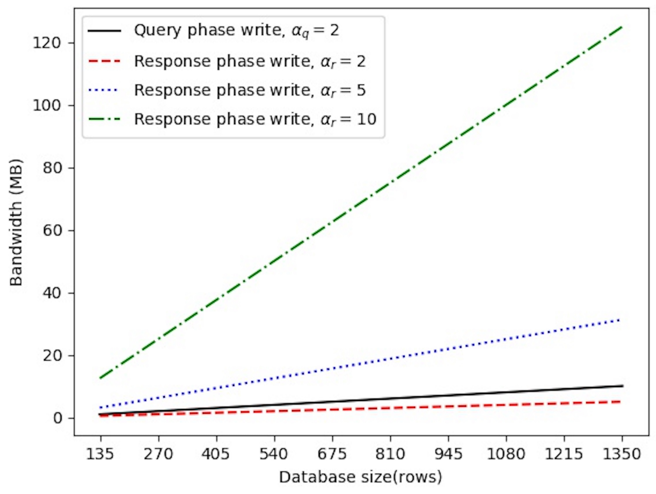


**Figure 4.6:** Six nodes bandwidth averages for query write request, response write request with recovery from 2 collisions, response write request with recovery from 5 collisions, response write request with recovery from 10 collisions.

MME. Queries include the gene name as in MME, but also the querying user's public key, and are collected during epochs, whose length is based on the number of write requests, therefore building anonymity sets for the epoch. By taking advantage of the underlying MME matching protocol, these queries can be seamlessly responded to, without publicly revealing the contact details of other researchers/clinicians which generated a match, by using the public key provided to encrypt the match. Also, other users can provide their (encrypted) contact details if they so wish.

Our experimental evaluation attests to the practicality of using AnoniMME to bring anonymity to the Matchmaker Exchange. Overall, using the method proposed in Section 4.2.3.1 to recover write requests in case of collisions yields better running times and bandwidth complexities, even when the number of nodes increases.

Since AnoniMME is based on Riposte [52], one might want to compare the two systems; however, Riposte focuses on experimental results from the server's perspective, while we evaluate performance from a user perspective.

Also note that the bandwidth overhead in our *n*-server construction is non-negligible, especially with a high collision recovery factor and increasing database sizes (as discussed in Section 4.2.3.1). A possible solution would be to use distributed point functions to reduce bandwidth complexity, similar to Riposte. However, we leave this to future work.

As the anonymity set size in AnoniMME corresponds to the number of users querying in a given epoch, one could increase it by requiring users to send empty queries to the system, following a certain probability distribution. The write requests would be formed as discussed in Section 4.2.2, although, instead of inputting a gene, the public key, or the contact details, the users just send an empty query. This is also used in Riposte, to minimize statistical disclosure attacks on their platform.

Finally, note that our implementation currently allows for 64 character messages, thus, queries can also include phenotypes from the Human Phenotype Ontology (as currently supported by MME), although, to ease of presentation we have discussed our experiments by only considering gene names. In future work, we plan to conduct a user study simulating a real-world deployment of AnoniMME with users of the MME, aiming to evaluate its usability with respect to anonymity protection, delays introduced by epochs, etc.

Chapter 5

# Empirical Analysis of Long-Term Security for Genomic Data

Over the past two decades, the cost of sequencing the human genome – i.e., determining a person's complete DNA sequence – has plummeted from millions to thousands of dollars, and continues to drop [123]. As a result, sequencing has not only become routine in biology and biomedics research, but is also increasingly used in clinical contexts, with treatments tailored to the patient's genetic makeup [22]. At the same time, the "direct-to-consumer" genetic testing market is booming [183] with companies like 23andMe and AncestryDNA attracting millions of customers, and providing them with easy access to reports on their ancestry or genetic predisposition to health-related conditions. Progress and investments in genomics have also enabled public initiatives to gather genomic data for research purposes. For instance, in 2015, the US launched the "All of Us" program [125], which aims to sequence one million people, while, in the UK, Genomics England is sequencing the genomes of 100,000 patients with rare diseases or cancer [6].

Alas, as more and more genomic data is generated, collected, and shared, serious privacy, security, and ethical concerns also become increasingly relevant. The genome contains very sensitive information related to, e.g., ethnic heritage, disease predispositions, and other phenotypic traits [24]. Furthermore, even though most published genomes have been anonymized, previous work has shown that anonymization does not provide an effective safeguard for genomic data [79]. While some individuals choose to donate their genome to science, or even publicly share it [141], others might be concerned about their privacy, or fear discrimination by employers, government agencies, insurance providers, etc. [36].

Worse yet, consequences of genomic data disclosure are not limited in time or to the data owner: due to its hereditary nature, access to one's sequenced genome inherently implies access to many features that are relevant to their progeny and their close relatives.

Motivated by these challenges, the research community has produced a large body of work aiming to protect genomic privacy and enable privacy-preserving sharing and testing of human genomes [118]. Available solutions mostly rely on cryptographic tools, including encryption as well as Secure Computation, Homomorphic Encryption, Oblivious RAM, etc. [28]. However, modern encryption algorithms provide security guarantees only against computationally bounded adversary; essentially, their security is assumed to last for 30 to 50 years [161]. While this timeframe is acceptable for most uses of encryption, it is not for genomic data.

**GenoGuard [92].** To address the problem of "long-term security," Huang et al. [92] introduce GenoGuard, a tool based on Honey Encryption (HE) [100] to provide confidentiality of genomic data even in the presence of an adversary who can brute force all possible encryption keys. GenoGuard uses a distribution transforming encoder (DTE) together with symmetric (password-based) encryption. In essence, whenever an attacker would try to decrypt a GenoGuard ciphertext using a wrong password, the decryption will give a wrong but plausible looking plaintext, which we denote as a *honey sequence.*

HE schemes based on DTE-then-encrypt constructions (as is the case for GenoGuard) only provide security in the message recovery context. That is, having access to the ciphertext only gives an unbounded adversary a negligible advantage in guessing the correct plaintext. However, as first discussed by Jaeger et al. [95], ciphertexts obtained from DTE-then-encrypt HE might still leak a significant amount of information about the plaintexts. As an example, the authors say that their honey encryption scheme would not be able to effectively protect normal HTTPS certificate keys due to the fact that the public key associated with the encrypted secret key being available to the attacker.

**Technical Roadmap.** We evaluate GenoGuard security by analyzing ciphertexts obtained using easily guessable (low-entropy) passwords as well as hard (high-entropy) ones. In other words, in both cases, we decrypt a GenoGuard ciphertext using a corpus of passwords and analyze the resulting decryptions (honey sequences). In the low-entropy setting, we consider an adversary who aims to identify the correct sequence among a pool of honey sequences, whereas, in the high-entropy case, one that uses the GenoGuard ciphertext in order to obtain more information about the target sequence when side information is available as opposed to the adversary using only inference methods for genomic data on that side information..

In our experimental evaluation, we show that, under a low-entropy password setting, an adversary who has access to side information about the target sequence can quickly eliminate the decoy sequences in order to have an increased advantage of guessing the correct sequence. This draws attention to the fact that if the attacker obtains a list of known passwords for a user (as passwords are often compromised and/or re-used), together with some side information about the user's sequence, she can have a significant advantage in guessing the correct sequence.

In the high-entropy setting, we not only observe that access to the GenoGuard ciphertext improves an adversary's accuracy in guessing SNVs from a target sequence when 10% or less of the target sequence is available to her as side information, but also draws attention to the fact that with enough side information, the adversary can predict a significant part of the target genome just by using state of the art inference methods for genomic sequences.

**Contributions.** In summary, this chapter makes two main contributions. First, under a low-entropy password setting, we formally show that, if the adversary obtains side information about the target sequence, there is a significant lower bound in her advantage. This highlights that the system offers low security when the adversary has access to side information, as supported by empirical evidence. Second, in the high-entropy password setting, we quantify the privacy loss for a user as a result of

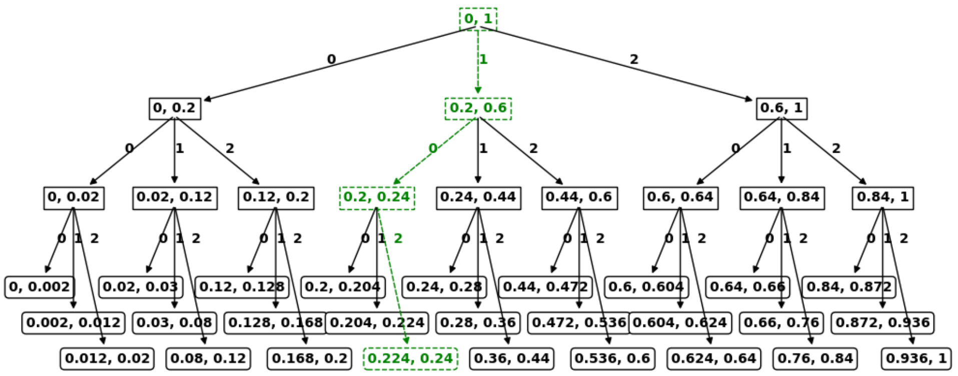


**Figure 5.1:** Toy example describing the encoding process for a sequence $(1,0,2)$. The green dashed line represents the correct encoding of the sequence. When the final leaf (interval $[0.224, 0.24]$) is reached, a seed is picked at random from this range.

using GenoGuard, compared to the best inference methods for genomic data; once again, showing that that it is non-negligible.

**Terminology.** In the rest of the chapter, to denote sequences decrypted from GenoGuard, we use the term *honey sequences*.

## 5.1 GenoGuard

In this section, we review GenoGuard [92], along with a security analysis of the framework.

### 5.1.1 Construction

GenoGuard is a framework providing long-term confidentiality for genomic data based on Honey Encryption [100]. More specifically, it allows to encode genomic data, encrypt it using a secret password, and store in a database, in such a way that its confidentiality is preserved even against an attacker that can brute-force all possible passwords. In GenoGuard, genomes are represented as a sequence of single-nucleotide variants (SNVs), i.e., values in $\{0,1,2\}$.

**Encoding.** The construction uses a DTE scheme optimized for genome sequences. It assigns subspaces of seed space $\mathscr{S}$ to the prefixes of a sequence $M$, i.e., all the subsequences in the set $\{M_{1,i} | 1 \le i \le n\}$, where $n$ is the length of the sequence. For example, the prefixes of the sequence 01102 are $\{0, 01, 011, 0110, 01102\}$. The seed space $\mathscr{S}$ is the interval $[0,1)$, with each seed being a real number in this interval.

Let $\mathscr{M}$ be the set of all possible sequences (the plaintext space). To calculate the cumulative distribution function (CDF) of each sequence, a total order $\mathscr{O}$ is assigned to all sequences in $\mathscr{M}$. For any two different sequences $M$ and $M'$, we assume that they start to differ at SNV$_i$ and SNV$'_i$. If the value of SNV$_i$ is smaller than that of SNV'$_i$, then, $\mathscr{O}(M) < \mathscr{O}(M')$, and $\mathscr{O}(M) > \mathscr{O}(M')$ otherwise. The CDF of a sequence $M$ is then calculated as:

$$CDF(M) = \sum_{M' \in \mathscr{M}, \mathscr{O}(M') \leq \mathscr{O}(M)} \Pr_{SNV}(M')$$

where $\Pr_{SNV}(M')$ is the probability of the sequence $M'$.

The encoding of a sequence can be performed using a perfect ternary tree, as depicted in Figure 5.1. (Note that the plot was generated using code obtained from GenoGuard's Github page.[1]) Each node in the tree represents a prefix of a sequence, and each leaf a complete sequence. Nodes have an interval $[L_i^j, U_i^j)$, where $i$ is the depth of the node in the tree and $j$ its order at a given depth $i$. The first node has the interval $[L_0^0, U_0^0) = [0,1)$. Depending on the value of the SNV at position $i{+}1$, the encoding proceeds from the node that represents $M_{1,i}$ with order $j$ at depth $i$ to depth $i+1$ as follows:

- If SNV$_{i+1} = 0$, go to the left branch and attach an interval $[L_{i+1}^{3j}, U_{i+1}^{3j}) = [L_i^j, L_i^j + (U_i^j - L_i^j) \times \Pr(SNV_{i+1} = 0 | M_{1,i}))$
- If SNV$_{i+1} = 1$, go to the middle branch and attach an interval $[L_{i+1}^{3j+1}, U_{i+1}^{3j+1}) = [L_i^j + (U_i^j - L_i^j) \times \Pr(SNV_{i+1} = 0 | M_{1,i}), L_i^j + (U_i^j - L_i^j) \times (\Pr(SNV_{i+1} = 0 | M_{1,i} + \Pr(SNV_{i+1} = 1 | M_{1,i})))$
- If SNV$_{i+1} = 2$, go to the right branch and attach an interval $[L_{i+1}^{3j+2}, U_{i+1}^{3j+2}) = [L_i^j + (U_i^j - L_i^j) \times (\Pr(SNV_{i+1} = 0 | M_{1,i} + \Pr(SNV_{i+1} = 1 | M_{1,i})), U_i^j)$.

In order to compute the conditional probabilities, Huang et al. [92] consider several models and compare their goodness of fit for real-world genome datasets. Specifically, they experiment with Linkage Disequilibrium (LD), Allele Frequencies (AF),

[1]https://github.com/acs6610987/GenoGuard

building $k$-th order Markov chains on the dataset and recombination rates (RR), and find the latter to perform best.

Finally, when a leaf is reached, a seed is picked uniformly from this range as the encoding of the corresponding sequence, and then fed into a Password-based Encryption (PBE) scheme to perform encryption, using a password chosen by the user.

**Decoding.** To decode an encoded-then-encrypted sequence, the ciphertext is first decrypted (as per the PBE scheme) using the user-chosen password; this recovers the seed. Then, the decoding process proceeds similar to the encoding one. That is, given the seed $S \in [0,1)$, at each step, the algorithm computes three intervals for the three branches, chooses the interval in which the seed $S$ falls, and moves down the tree. Once a leaf node is reached, the path from the root to the leaf is outputted as the decoded sequence.

**Finite Precision.** Note that the Honey Encryption encoding model, as described in Section 5.1.1, requires the seed space $\mathscr{S}$ to be a real number domain with infinite precision. In the case of DNA sequences, this would yield a very long floating-point representation, and thus a high storage overhead. Therefore, GenoGuard uses a modification of the DTE scheme for finite precision. Specifically, for a sequence of length $n$, where each SNV takes three possible values, at least $n \cdot \log_2 3$ bits are needed for storing the sequence. Hence, a storage overhead parameter $h > \log_2 3$ is selected, and each sequence is encoded over $h \cdot n$ bits. The algorithm works as before, by selecting intervals according to the values of the respective SNVs based on conditional probabilities. The root interval is $[0, 2^{hn}-1]$. At each branch at depth $i$, the algorithm will allocate a seed space of size $3^{n-i-1}$, and each following step will segment an input interval into three parts of equal size. Hence, any subinterval of the $j$-th node at depth $i$ will contain $3^{n-i-1}$ integers.

## 5.1.2 Security

Huang et al. [92] evaluate the security of GenoGuard vis-à-vis the probability of an unbounded adversary recovering the encrypted sequence. That is, given the encryption of a message, what is the probability of the adversary recovering the correct message, even if she can brute-force all possible encryption keys for the underlying PBE scheme?

**Upper Bound.** More formally, they prove an upper bound to the probability an adversary recovers the correct message to be:

$$Pr_{p_m,p_k} \leq w(1+\delta) + \frac{3^n + 1/w}{2^{(h-\log_2 3)n}} \tag{5.1}$$

where $p_m$ is the original sequence distribution with maximum sequence probability $\gamma$, $p_k$ is a key (password) distribution with maximum weight $w$ (i.e., the most probable password has probability $w$), $n$ is the length of the sequence, $h$ the overhead parameter, and $\delta$ a parameter depending on $w$ and $\gamma$.

Let $\Delta$ denote the fraction $\frac{3^n+1/w}{2^{(h-\log_2 3)n}}$ in Equation 5.1. Note that $\Delta$ is a security loss term, since the upper bound on plaintext recovery probability should be $w$, as an adversary who trivially decrypts the ciphertext with the most probable key and outputs the result can recover the original message with probability $w$. $\Delta$ is essentially the security lost due to DTE imperfectness when moving to finite precision, i.e., given by the difference between the original message distribution and the DTE distribution. As shown in [92], for $n = 20{,}000$, $h = 4$, $w = \frac{1}{100}$, and $\gamma = 2.89 \times 10^{-44}$, $\Delta$ is approximately $2^{-16600}$.

**Empirical Evaluation.** Huang et al. [92] also present an *empirical* security analysis based on two experiments. In both, the chromosome 22 of a victim is encrypted using a password pool consisting of numbers from 1 to 1000, with "539" assumed to be the correct one. Then, in order to rule out wrong passwords, the interval size of each of the decrypted sequences is computed. In the first experiment, a genome is encoded by assuming a uniform distribution (i.e., each branch has weight $1/3$ at all depths), and a PBE scheme is used to encrypt the seed. In the second experiment, GenoGuard is used to encrypt the victim's sequence. Hence, the size of the interval of a leaf in the ternary tree is proportional to the probability of the corresponding sequence. The results of their experiments, reported in Figure 10 in [92], show

| Symbol | Meaning |
|---|---|
| MR | Message recovery |
| SI | Side information |
| HEnc | Honey Encryption |
| HDec | Honey Decryption |
| $\mathscr{K}$ | Key space |
| $\mathscr{M}$ | Message space |
| $p_k$ | Key distribution |
| $p_m$ | Message distribution |
| $\mathscr{A}$ | Adversary |
| $\mathscr{A}^{SI}$ | Adversary with access to side information |

**Table 5.1:** Table of Notation.

that a simple classifier can distinguish the correct sequence in the first experiment, while, in the second one, it is "buried" among all the decrypted sequences.

## 5.2 Evaluation Methods

We now describe our evaluation methods, for both low and high-entropy password settings. Before doing so, we introduce the notation used in the rest of the paper in Table 5.1.

### 5.2.1 Low-Entropy vs High-Entropy Password

We use different approaches for evaluating GenoGuard under two different password types, namely low-entropy and high-entropy passwords. In other words, we encrypt a sequence with GenoGuard using either an easy to guess, low-entropy password ($\approx$7 bits), or using a harder password with a higher entropy ($\approx$72 bits).

The difference in the evaluation of the two approaches is given by the adversary's goal. Specifically, in the low-entropy password case, the adversary attempts to use the side information in order to distinguish the original encrypted sequence among a pool of honey sequences. By contrast, in the high-entropy setting, the adversary uses both the honey sequences and the side information in order to predict the value of each SNV at each position in the target sequence.

### 5.2.2 Threat Model

We use the same system and threat model presented in the GenoGuard paper [92], i.e., we assume a genomic sequence of a user is to be stored, encrypted, at a third-party database, e.g., a biobank. We consider an adversary that has access to the encrypted data (for instance, she breaks into the biobank and gets access to the encrypted database, or the biobank itself is adversarial) and has access to public knowledge as well as to some side information (as discussed below).

**Low-Entropy Password.** The main adversarial goal in this case is to identify the target sequence among a pool of honey sequences, using the side information available, i.e. "message recovery" with side information (**MR-SI**).

**High-Entropy Password.** The main adversarial goal is to obtain as much information as possible about the sequence that was encrypted. Note that this adversarial goal is different from "message recovery," according to which Huang et al. [92] evaluate GenoGuard's security (cf. Section 5.1.2). The main intuition is that, as also hypothesized by [95], using Honey Encryption might actually leak non-negligible information about the sequences encrypted using GenoGuard, even if the adversary cannot correctly recover the full plaintext with non-negligible probability.

### 5.2.3 Adversary's Side Information

As mentioned above, the adversary has access to the victim's encrypted sequence as well as to public information such as, Linkage Disequilibrium, Allele Frequencies, Recombination Rate (see Section 2.1). In addition, we assume that the adversary may have some side information about the victim.

When referring to side information, note that we do *not* consider knowledge of common traits from phenotype-genotype associations, e.g., gender, ancestry, or other information about the victim that could be obtained, e.g., from social media. In fact, this is covered by GenoGuard's guidelines, which state that the user should include as much side information about their genome as possible when performing the encoding. Whereas, even though assuming the user can knowingly enumerate all possible side information is quite a strong assumption, we actually consider the case where the victim undertakes some specific tests, and the adversary learns additional information about the victim from the outcome of those tests. Additionally, the victim might choose to re-encrypt their genome after obtaining the test results in order to incorporate them in the encoding, and the adversary could use the new ciphertext to extract information about the old ciphertext.

In the high-entropy password setting, we also evaluate the case where an adversary has no side information about the target sequence, in order to quantify the information leakage that might occur from using GenoGuard against baseline inference methods for genomic sequences. Overall, we consider different types of side information available to the adversary:

1. *No Side Information:* The adversary has access only to the encrypted sequence. (NB: this is only evaluated for the high-entropy password setting)
2. *Sparse SNVs:* The adversary has access to SNV values sparsely distributed in the target sequence.
3. *Consecutive SNVs:* The adversary has access to values from a cluster of consecutive SNVs in the target sequence.

### 5.2.4 Low-Entropy Password

We now formally provide a lower bound for the adversary's advantage in the case where she obtains side information about the target sequence and encryption is done using a low-entropy password.

We present a lower bound on the adversary's advantage when she has access to side information about the encrypted sequence and can exhaustively search the message space. We prove the bound formally, building on [95], which shows the impossibility of known-message attack (KMA) security with low-entropy passwords. However, instead of the adversary having access to message-ciphertext pairs, we assume that the adversary has access to (position, value) pairs from the encrypted sequence. The game defining message recovery security with side information is denoted as $\textbf{MR-SI}^{\mathscr{A}}_{HE,p_m,p_k}$ and illustrated in Figure 5.2.

```
$\text{MR-SI}^{\mathscr{A}}_{HE,p_m,p_k}$:
. $K^* \leftarrow_{p_k} \mathscr{K}$
. $M^* \leftarrow_{p_m} \mathscr{M}$
. $C^* \leftarrow_{\$} \text{HEnc}(K^*, M^*)$
. $M \leftarrow_{\$} \mathscr{A}^{SI}(C^*)$
. If $M = M^*$: Return 1
  Else: Return 0
```

**Figure 5.2:** Definition of Message Recovery Security with Side Information (MR-SI).

```
Adversary $\mathscr{A}^{SI}(C^*)$:
. Let $q$ be the number of known SNVs
. Let $SI$ be the set of $q$ pairs $(pos_i, val_i)$ from $M^*$
. $K_q \leftarrow \emptyset$
. For $K \in \mathscr{K}$:
    If($\forall i$ HDec$(K, C^*)[pos_i] = val_i$):
        $K_q \leftarrow K_q \cup \{K\}$
. $K \leftarrow_{\$} K_q$
. Return $HDec(K, C^*)$
```

**Figure 5.3:** Adversary strategy for MR-SI, having access to $q$ pairs of (position, value) from the original message.

Given a ciphertext $C^*$, an adversary $\mathscr{A}^{SI}$, with access to side information, is allowed to guess the message by brute force. The adversary $\mathscr{A}^{SI}$ wins the game if her output message is the same as the original message.

Our intuition is that the advantage of the adversary $\mathscr{A}^{SI}$ (Figure 5.3), for a number $q$ ($q \leq 2n$, where $n = [\log_2 |\mathscr{K}|]$) of positions and values, from the original sequence, is equal to the probability that a randomly chosen key that decrypts correctly all values at the given positions, will also decrypt the rest of the sequence, i.e., $\mathsf{Adv}^{\text{mr-si}}_{HE,p_m,p_k}(n) = \Pr[\text{MR}-\text{SI}^{\mathscr{A}}_{HE,p_m,p_k}]$. We denote by $K_q$ the number of keys consistent with the positions and values used as side information.

Hence, we use Lemma 4.2 from [95], as follows:

**Lemma 5.2.1.** *If $s_0, s_1, ..., s_n$ are positive integer-valued random variables such that $s_0 \leq 2^n$ and $s_{q+1} \leq s_q$, for $q \in \mathbb{Z}_n$, then* $\max_{q \in \mathbb{Z}_n} \mathbb{E}\left[s_{q+1}/s_q\right] \geq \frac{1}{2n}$.

*Proof. (Reproduced from [95]).* Let $\varepsilon = \max_{x \in \mathbb{Z}} \mathbb{E}\left[s_{q+1}/s_q\right]$. We will use an inductive argument to prove that $\Pr[s_q \geq 2^{n-q}] < 2q\varepsilon$ for $1 \leq q \leq n$. Then, considering when $q$ is $n$ and nothing that $s_n \geq 1$ always, we have $1 = \Pr[s_n \geq 1] \leq 2n\varepsilon$. Solving for $\varepsilon$ gives the desired bound.

We now give the inductive argument. First, Markov's inequality can be used to bound the probability that $s_{q+1}$ is at least half of $s_q$ by $\Pr[s_{q+1}/s_q \geq 1/2] \leq 2\,\mathbb{E}\left[s_{q+1}/s_q\right]$. Rewriting and bounding $\mathbb{E}\left[s_{q+1}/s_q\right]$ by $\varepsilon$ we get:

$$\Pr[s_{q+1} \geq (1/2)s_q] \leq 2\varepsilon \tag{5.2}$$

for all $q \in \mathbb{Z}_n$.

Recalling that $s_0 \leq 2^n$, the base case is easily derived by $\Pr[s_1 \geq 2^{n-1}] \leq \Pr[s_1 \geq (1/2)s_0] \leq 2\varepsilon/$ Now suppose $1 < q \leq n$ and $\Pr[s_{q-1} \geq 2^{n-(q-1)}] \leq 2(q-1)\varepsilon$. By definition, we have:

$$\begin{aligned}\Pr[s_q \geq 2^{n-q}] = {} & \Pr[s_q \geq 2^{n-q} | s_q - 1 < 2^{n-(q-1)}]\Pr[s_{q-1} \geq 2^{n-(q-1)}] \\ & + \Pr[s_q \geq 2^{n-q}]\Pr[s_q \geq 2^{n-q} | s_q - 1 < 2^{n-(q-1)}]\Pr[s_{q-1} < 2^{n-(q-1)}]\end{aligned} \tag{5.3}$$

The first part of the equation can be bounded using our inductive assumption:

$$\begin{aligned}\Pr[s_q \geq 2^{n-q} | s_q - 1 < 2^{n-(q-1)}]\Pr[s_{q-1} \geq 2^{n-(q-1)}] & \leq \Pr[s_{q-1} \geq 2^{k-(q-1)}] \\ & \leq 2(q-1)\varepsilon\end{aligned} \tag{5.4}$$

To bound the second part, note that, conditioned on the fact that $s_{q-1} < 2^{n-(q-1)}$, it can only hold that $s_q$ is greater than $2^{n-q}$ if $s_q$ is greater than $(1/2)s_{q-1}$. This gives us:

$$\Pr[s_q \geq 2^{n-q} | s_q - 1 < 2^{n-(q-1)}] \leq \Pr[s_q \geq (1/2)s_{q-1} | s_{q-1} < 2^{n-(q-1)}] \tag{5.5}$$

Then, form the definition of conditional probability and equation (5.2), we get that:

$$\begin{aligned}\Pr[s_q \geq 2^{n-q} | s_q - 1 < 2^{n-(q-1)}]\Pr[s_{q-1} < 2^{n-(q-1)}] & \leq \Pr[s_q \geq (1/2)s_{q-1}] \\ & \leq 2\varepsilon.\end{aligned} \tag{5.6}$$

```
Game 1:
. $K^* \leftarrow_{p_k} \mathscr{K}$
. $M^* \leftarrow_{p_m} \mathscr{M}$
. $C^* \leftarrow_{\$} \mathrm{HEnc}(K^*, M^*)$
. Let $q$ be the number of known SNVs
. Let $SI$ be the set of $q$ pairs $(pos_i, val_i)$ from $M^*$
. $K_q \leftarrow \emptyset$
. For $K \in \mathscr{K}$:
    If($\forall i$ $\mathrm{HDec}(K, C^*)[pos_i] = val_i$):
        $K_q \leftarrow K_q \cup \{K\}$
. $K \leftarrow_{\$} K_q$
. $M \leftarrow HDec(K, C^*)$
. If $M = M^*$: Return 1
  Else: Return 0
```

**Figure 5.4:** Game 1, used in the proof of Theorem 5.2.2.

Putting the above together, we get $\Pr[s_q \geq 2^{n-q}] \leq 2qe$, thus completing the proof.

□

Using Lemma 5.2.1, we can compute the adversary's advantage as follows:

**Theorem 5.2.2.** *Let HE be an encryption scheme and $n = [\log_2 |K|]$. Then, for any $p_m, p_k$, the adversary $\mathscr{A}^{SI}$ who obtains at most $n-1$ positions and values from the original sequence will have advantage:*

$$\mathsf{Adv}^{\mathrm{mr-si}}_{HE,p_m,p_k}(\mathscr{A}^{SI}) \geq \frac{1}{2n^2}$$

*Proof.* The advantage $\mathsf{Adv}^{\mathrm{mr-si}}_{HE,p_m,p_k}(\mathscr{A}^{SI})$, is equal to $\Pr[\text{Game 1 Retuns 1}]$ where Game 1 is defined in Figure 5.4. This is due to the fact that Game 1 is $\text{MR-SI}^{\mathscr{A}}_{HE,p_m,p_k}$ together with Adversary $\mathscr{A}^{SI}(C^*)$. By applying a few transformations to Game 1 (i.e. take the $(\forall i$ $\mathrm{HDec}(K, C^*)[pos_i] = val_i)$ check and changing it to a "for" loop), and changing the final check (i.e. instead of checking if $M = M^*$ before returning 0 or 1, it checks if the key $K$ is in the subset that decrypts $C^*$ to $M^*$), we obtain an equivalent game, Game 2 (Figure 5.5). Thus, $\Pr[\text{Game 1 Returns 1}] = \Pr[\text{Game 2 Returns 1}]$.

Since $K_{q+1} \subseteq K_q$, for fixed $q$, the probability that Game 2 will return 1 is $\mathbb{E}\left[\frac{|K_{q+1}|}{|K_q|}\right]$. So we have $\Pr[\text{Game 2 Returns 1}] = \sum_{q=0}^{n} \frac{1}{n} \mathbb{E}\left[\frac{|K_{q+1}|}{|K_q|}\right]$.

```
Game 2:
. $K^* \leftarrow_{p_k} \mathscr{K}$
. $M^* \leftarrow_{p_m} \mathscr{M}$
. $C^* \leftarrow_{\$} HEnc(K^*, M^*)$
. Let $q$ be the number of known SNVs
. Let $SI$ be the set of $q$ pairs $(pos_i, val_i)$ from $M^*$
. $K_0 \leftarrow \mathscr{K}; K_1, K_2, ...K_{q+1} \leftarrow \emptyset$
. For $(pos_i, val_i) \in SI$:
    For $K \in K_{i-1}$:
      If$(HDec(K, C^*)[pos_i] = val_i)$:
        $K_i \leftarrow K_i \cup \{K\}$
. For $K \in K_q$:
    If $HDec(K, C^*) = M^*$
      $K_{q+1} \leftarrow K_q \cup \{K\}$
. $K \leftarrow_{\$} K_q$
. If $K \in K_{q+1}$: Return 1
  Else: Return 0
```

**Figure 5.5:** Game 2, a transformed version of Game 1.

We then define Experiment 1 (Figure 5.6), which shows that the distribution of $K_{q+1}$ and $K_q$ for $q \in \mathbb{Z}_n$ is the same as the distribution in Game 1. Let $s_q$ denote $|K_q|$ and $\varepsilon = max_{q \in \mathbb{Z}_n} \mathbb{E}\left[\frac{s_{q+1}}{s_q}\right]$, where the expectation is taken in Experiment 1. Since all $K_q$ contain at least the key $K^*$, they all are positive. Thus, by applying Lemma 5.2.1 we have $\varepsilon \geq \frac{1}{2n}$. Then:

$$\begin{aligned}\mathsf{Adv}^{\mathrm{mr-si}}_{HE,p_m,p_k}(\mathscr{A}^{SI}) &= \Pr[\text{Game 2 Returns 1}] \\ &= \sum_{q=0}^{n} \frac{1}{n} \mathbb{E}\left[\frac{|K_{q+1}|}{|K_q|}\right] \geq \frac{1}{n} \cdot \varepsilon \geq \frac{1}{2n^2}\end{aligned}$$

□

This shows that the security of the systems is weak even with a small number of pairs (position, value) from the target sequence available to the attacker, as opposed to having multiple known ciphertext-plaintext pairs.

### 5.2.5 High-Entropy Password

We now give an overview of our inference strategy using the GenoGuard ciphertext and discuss the baseline inference methods we evaluate our strategy against. Note that both the baseline inferences as well as our inference strategy assume that the adversary has access to prior information about the target (i.e., AF, LD, recombina-

```
Experiment 1:
. $K_0 \leftarrow \mathscr{K}; K_1, K_2, \ldots K_n \leftarrow \emptyset$
. $\mathrm{K}^* \leftarrow_{p_k} \mathscr{K}$
. $\mathrm{M}^* \leftarrow_{p_m} \mathscr{M}$
. $\mathrm{C}^* \leftarrow_{\$} \mathrm{HEnc}(\mathrm{K}^*, \mathrm{M}^*)$
. Let $n$ be the number of known SNVs
. Let $SI$ be the set of $n$ pairs $(pos_i, val_i)$ from $\mathrm{M}^*$
. For $(pos_i, val_i) \in SI$:
    For $\mathrm{K} \in K_{i-1}$:
      If$(\mathrm{HDec}(\mathrm{K}, \mathrm{C}^*)[pos_i] = val_i)$:
        $K_i \leftarrow K_i \cup \{K\}$
```

**Figure 5.6:** Experiment 1, used in the proof of Theorem 5.2.2.

tion rate), which is available to them as the encoding tree is stored together with the encrypted sequence.

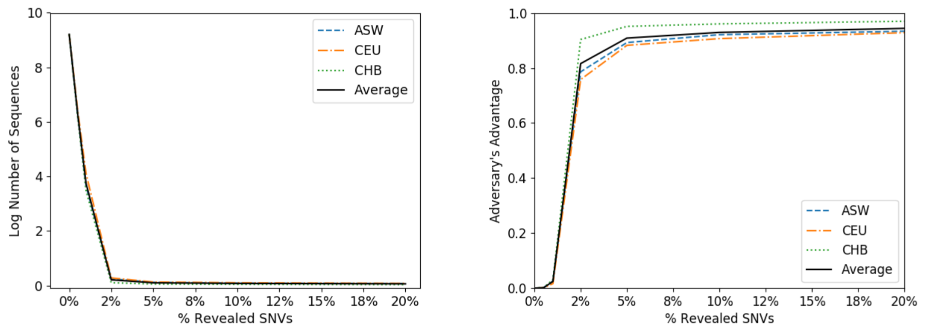
**Baseline Inferences.** We compare the performance of our inference strategy to baselines for genomic sequence inference. For these baselines, we assume that the adversary has access only to side information, as discussed in Section 5.2.3, but not the ciphertext resulting from GenoGuard's encode-then-encrypt method.

As done by Samani et al. [156], we set to infer the value of an unknown SNV$_i$, given a probabilistic modeling of genome sequences. More specifically, we use the following models for SNV correlation:

- *B1:* 1st order Markov chain model from AF and LD: most likely genotype.
- *B2:* 1st order Markov chain model from AF and LD: sampled genotype.
- *B3:* RR Model.

**GenoGuard Inference Methods.** Our method is based on exploiting the similarities between the honey sequences in order to obtain information about the target sequence. More specifically, we use two strategies:

1. *G1.* Side information-weighted SNVs: We assign a weight to each of the honey sequences based on the number of SNVs from the side information contained. We then consider only the sequences with the highest weight and output the most common SNVs among them as our candidate SNVs for the



**(a)** #Candidate sequences vs % revealed sparse SNVs from target sequence **(b)** Adv's advantage vs % revealed sparse SNVs from target sequence

**Figure 5.7:** Results of our evaluation in the low-entropy setting vis-à-vis an adversary with access to side information in the form of sparse SNVs from the target sequence.

target sequence. In the case of no side information, we consider the most common SNVs across all honey sequences.

2. *G2.* Interval and Side information-weighted SNVs: Similar to the previous method, however, we also adjust the weight of each sequence when considering the most common SNVs by the size of the interval that the seed of the respective sequence will fall into. In the case of no side information, we take the most common SNVs from all honey sequences, weighted by the previously mentioned interval size.

### 5.2.6 Experimental Evaluation

In this section, we present the datasets used for the experimental evaluation and the results obtained for both evaluation methods, i.e., low-entropy and high-entropy passwords.

#### 5.2.6.1 Dataset

We use the Phase III data from the HapMap dataset, i.e., the third release from the HapMap project.[2] HapMap was an international project [51], run between 2002 and 2009, aimed at developing a haplotype map of the human genome, and describe the common patterns of human genetic variation. The HapMap data has been made publicly available and used for various research purposes, e.g., to research genetic

[2]https://www.sanger.ac.uk/resources/downloads/human/hapmap3.html

variants affecting health, disease and responses to drugs and environmental factors, etc.

The Phase III release increased the number of DNA samples to 1,301 and included 11 different populations. In our experiments, we select data from three populations:

1. ASW (African ancestry in Southwest USA),
2. CEU (Utah residents with Northern and Western European ancestry from the CEPH collection),
3. CHB (Han Chinese in Beijing, China).

We sample 50 sequences at random from each of them, for a total of 150 sequences.

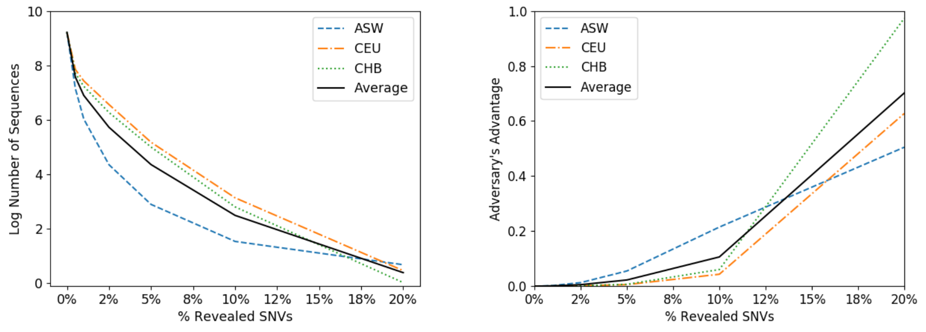
For all three populations presented above, we test the same SNVs positions.

#### 5.2.6.2 Low-Entropy Password

**Experiment Overview.** We use the following strategy for our evaluation:

1. Encrypt a sequence using GenoGuard's DTE-then-encrypt method: for each of the 150 sequences, we select and encrypt 1,000 positions from chromosome 13, with a storage overhead $h = 4$ (the same as in the experimental evaluation of GenoGuard), using a low-entropy password.
2. Decrypt the ciphertext, using the top 10,000 most common passwords released by Daniel Miessler[3] (with the encryption password in the set), to obtain plausible looking honey sequences;
3. Exclude the sequences which do not contain the side information.
4. Output the number of remaining sequences, given how many of the possible passwords match the side information.

[3]https://github.com/danielmiessler/SecLists/blob/master/Passwords/Common-Credentials/10k-most-common.txt



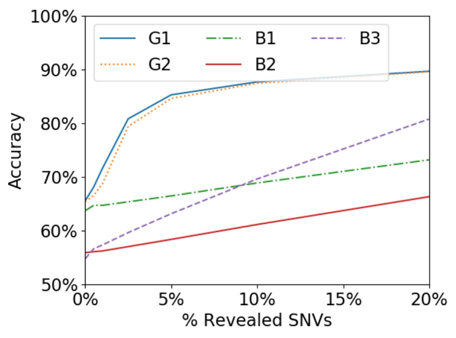
**(a)** #Candidate sequences vs % revealed consecutive SNVs from target sequence **(b)** Adv's advantage vs % revealed consecutive SNVs from target sequence

**Figure 5.8:** Results of our evaluation in the low-entropy setting vis-à-vis an adversary with access to side information in the form of a cluster of consecutive SNVs from the target sequence.

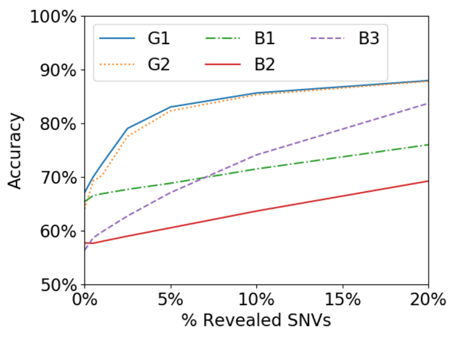
**Adversary's Advantage.** The performance of the adversary is calculated as the probability of the adversary guessing the target sequence within the remaining pool of honey sequences.

**Sparse SNVs from the Target Sequence.** Figure 5.7a illustrates how the log number of candidate sequences decreases with more side information available. With 1% side information (10 SNVs), the number of sequences that match the side information reduces to approximately 44 on average across the three populations. Figure 5.7b shows the increase of the adversary's advantage, averaged over 1000 rounds, vis-à-vis the number of SNVs available to her. 2.5% side information (25 SNVs) gives the adversary an advantage of approximately 80% on average for the ASW and CEU populations and close to 90% for the CHB population. With more side information, the adversary's advantage increases to over 90% for all populations.

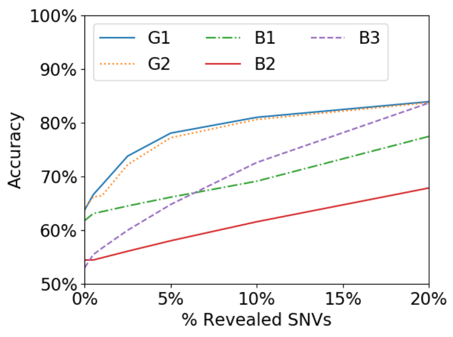
**Consecutive SNVs from the Target Sequence.** When the adversary has access to side information as a cluster of consecutive SNVs, she needs more side information to achieve comparable results to the Sparse SNVs case. Figure 5.8a shows the decrease of the log number of candidate sequences with increasing side information available. We observe the fastest decrease in the number of sequences with increasing side information available is for the ASW population when less than 10% of the sequence available. Figure 5.8b shows the increase of the adversary's advantage,



**(a)** ASW



**(b)** CEU



**(c)** CHB

**Figure 5.9:** Inference accuracy results in the high-entropy password setting for all three populations for side information available to the attacker in the form of sparse SNVs from the target sequence.

averaged over 1000 rounds, vis-à-vis the number of SNVs available to her. The increase in the adversary's advantage is slower as well, with an average of 70% across the three populations for 20% of the sequence available to the attacker.

#### 5.2.6.3 High-Entropy Password

**Experiment Overview.** The brute-force experiment presented in GenoGuard indicates that, when decrypting the same ciphertext with multiple passwords, the correct sequence would be "buried" among the incorrect ones. Hence, there is some similarity between the original sequence and the honey sequences.

As a result, we set to quantify the corresponding privacy loss, i.e. ***how much more information does an adversary obtain via access to ciphertext encrypted using GenoGuard obtains, compared to one who does not***.

Overall, we use the following evaluation strategy:

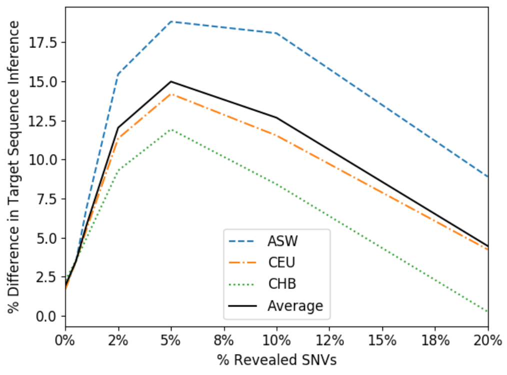


Figure 5.10: Difference in accuracy between the best performing GenoGuard and baseline inference methods, vis-à-vis an adversary with side information of sparse SNVs from the target sequence, in the high-entropy password setting.

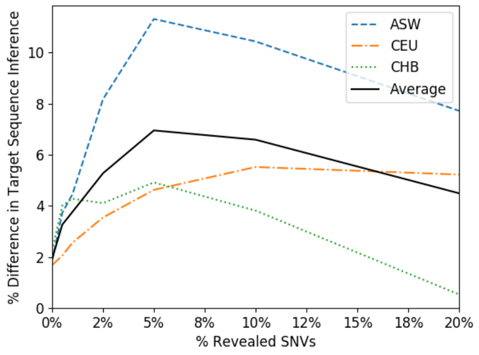


Figure 5.13: Difference in accuracy between the best performing GenoGuard and baseline inference methods, vis-à-vis an adversary with side information in the form of consecutive SNVs from the target sequence, in the high-entropy password setting.

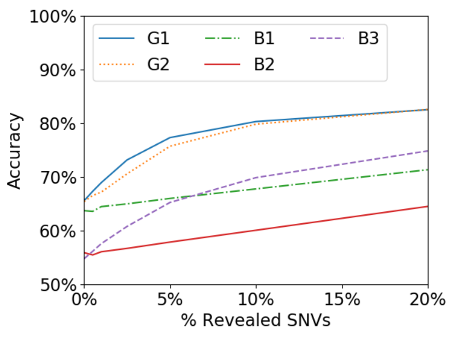
1. Encrypt a sequence using GenoGuard's DTE-then-encrypt method: for each of the 150 sequences, we select and encrypt 1,000 positions from chromosome 13, with a storage overhead $h = 4$, using a random, high-entropy password (approx. 72 bits).

2. Decrypt the ciphertext, using the top 10,000 most common passwords released by Daniel Miessler, to obtain plausible looking honey sequences;

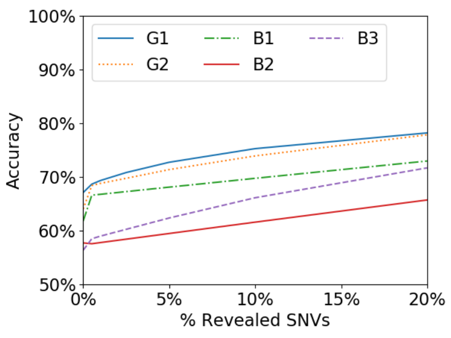
3. Infer the victim's sequence using the honey sequences.

**Accuracy.** To measure the performance and assess the potential leakage that access to the GenoGuard ciphertext might yield, we measure accuracy as the number of correctly guessed SNVs over the total number or SNVs guessed.

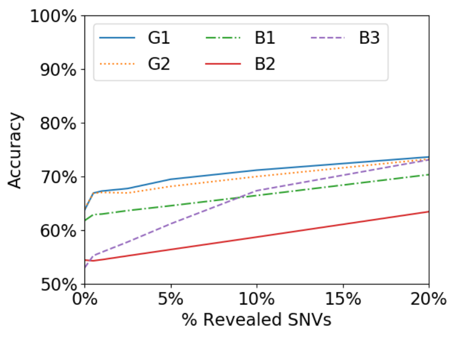
**Sparse SNVs from the Target Sequence.** Figure 5.9 shows the inference results in this case for the three population groups, averaged over 1,000 rounds. In the case where no side information is available to the attacker, for all populations, the attacker can infer approximately 2% more of the target sequence from the GenoGuard ciphertext than just by using baseline inferences based on the population. For the



**(a)** ASW



**(b)** CEU



**(c)** CHB

**Figure 5.12:** Inference accuracy results in the high-entropy password setting for all three populations for side information available to the attacker in the form of a cluster of consecutive SNVs from the target sequence.

ASW population (Figure 5.9a), over 80% of the target SNVs are guessed correctly with 2.5% (25 SNVs) or more of the target sequence available to the attacker. For the CEU population (Figure 5.9b), approximately 79% of the target SNVs are guessed correctly with 2.5% of the original sequence available to the attacker and over 83% of the target SNVs are guessed correctly with 5% (50 SNVs) or more are available. In the case of the CHB population (Figure 5.9c), the accuracy is of the GenoGuard inference is the lowest among the three populations, with over 73% accuracy in the cases where 2.5% SNVs are available to the attacker. The accuracy surpasses 80% for the CHB population when 10% or more of the target SNVs are available to the attacker.

In Figure 5.10, we illustrate the difference between the best performing inference method using the GenoGuard ciphertext and the best performing baseline inference method. On average, having access to the GenoGuard ciphertext improves the inference accuracy. The peak of the improvement in accuracy (approximately 15%) over the baseline models can be observed when the attacker has access to 5% sparse SNVs from the target sequence. After this, we can see a decline in this difference with increasing SNVs available for the attacker, as the baseline inference becomes more accurate with more information available. In fact, for the CHB population, the best performing baseline (B3) for the case when 20% of the target sequence is available to the attacker provides an accuracy comparable to the GenoGuard inferences ($\approx$83.8%).

**Consecutive SNVs from the Target Sequence.** In Figure 5.12, we illustrate the accuracy of the inference methods across the three populations when the adversary obtains, as side information, a cluster of consecutive SNVs, averaged over 1,000 rounds. For the ASW population (Figure 5.12a), the accuracy of inferred SNVs from the correct sequence using the GenoGuard ciphertext is over 73% for 2.5% or more of the target SNVs available as side information, and over 80% when 10% or more of the sequence is available to the attacker. The GenoGuard inference for the CEU population (Figure 5.12b) is over 70% when 2.5% or more of the target sequence is available to the attacker. For the CHB population (Figure 5.12c), the GenoGuard inferences have the lowest accuracy across the three populations, obtaining 70% or more accuracy only when 5% or more of the target sequence is available to the attacker.

Figure 5.13 shows the difference between the best performing GenoGuard inference method and the best performing baseline inference method. On average, the inference using the GenoGuard ciphertext gives better accuracy than the baseline methods, but overall less than the previous case where sparse SNVs are available as side information. In this case, the peak in the improvement of accuracy compared to the baseline methods is approximately 7%, on average, across the three populations, when 5% of the target SNVs are available to the attacker. For the CHB population, when 20% of the sequence is available as side information to the attacker, we observe, as in the case of sparse SNVs, that the best performing baseline inference method (B3) obtains a comparable accuracy to that of the GenoGuard inferences

(≈73%).

For this high entropy password case, while we do not truly break the security of GenoGuard in the cryptographic sense, recall that our goal was to determine *how much more information does an adversary with access to the ciphertext encrypted using GenoGuard obtains, compared to an adversary who does not.* We find that having access to just the GenoGuard-encrypted ciphertext does not significantly increase the adversary's inference power. The intuition behind the increase in inference accuracy in this case is that it is due to the finite precision of the encoding. However, with access to even as little as 2.5% increases the inference accuracy of the adversary by 12.5% on average for random SNVs as side information and by 5% on average for the consecutive SNVs case. By focusing on the honey sequences that contain the most side information, we leverage the fact that due to the high correlation between SNVs within the genomes, knowing one correct SNV will increase the accuracy of inferring the values of SNVs at close positions.

More recent work by Cheng et al. [45], also analyzes the security of GenoGuard, and shows that, under a random password, a hybrid PCA+SVM model trained on both real and decoy sequences succeeds achieves more accuracy(at least 76.54%) than the desired security, and manages to exclude the decoy sequences for 47.88% of the targets. They conclude that GenoGuard cannot resist attacks that exploit the difference between real message distribution and the message probability model (i.e., the decoy message distribution).

#### 5.2.6.4 Key Takeaways

Overall, our experimental evaluation shows that, when the adversary has access to some side information, access to a ciphertext encrypted using GenoGuard can help her recover a remarkably high percentage of the SNVs from the target sequence or significantly increase her advantage in recovering the correct sequence.

Therefore, users need to include as much side information that could be available to an adversary as possible when encrypting their genomic sequence. However, this prompts a parallel problem, with respect to how much that user is willing to publicly share (as this information is saved together with the ciphertext), considering that even without access to the GenoGuard ciphertext, it can enable attackers to correctly predict most of the target genome.

## 5.3 Discussion

Motivated by the decreasing cost of genomic sequencing and the related arising privacy challenges, the research community has produced a large body of work on genomic privacy. Most of the techniques focus on cryptographic tools, but fail to address the need for long term confidentiality for genomic data. In fact, GenoGuard [92] is the only tool available for ensuring the long term encryption needed for genomic data [118].

In this chapter, we set to determine whether GenoGuard can be safely used as an encryption tool, quantifying the additional privacy leakage arising from using it. We analyzed GenoGuard under two scenarios, based on the encryption password, for an adversary which has access to side information about the target sequence in the form of some values of SNVs from the target sequence. First, we assumed that the user encrypts his genomic sequence using a low-entropy, easily guessable password. In this case, we found that the adversary can easily exclude decoy passwords from the pool of possible passwords, and can guess the correct sequence with high probability by having access to 2.5% sparse SNVs or 20% or more consecutive SNVs from the target sequence.

Second, we assumed that the user encrypts his sequence using a high-entropy password. In this case, since elimination of decoy passwords might not yield any sequence, we use the honey sequences to obtain as much information as possible from the target sequence, exploiting the similarity between the original sequence and the honey sequences [92]. We then compared the sequence obtained from the honey sequences to state-of-the-art methods from genome sequence inference in order to observe the privacy leakage. Even with no side information available to the attacker, the sequence obtained from the honey sequences had a 2% improvement on average over all tested baseline methods. With side information in the form of sparse SNVs from the target sequence, the improvement in accuracy compared to the baseline inference models raises to up to 15% on average when 5% of the target sequence is available to the attacker, predicting more than 82% (on average) of the target sequence correctly. When the attacker obtains consecutive SNVs from the target sequence, the accuracy of the attacker decreases slightly from the previous case, yielding 73% accuracy when 5% of the target sequence is known, with an average improvement of 7% over the baseline methods.

In conclusion, we argue that the research community should invest more resources toward the design of long-term encryption tools for genomic data. Overall, GenoGuard could be a viable solution when the user incorporates *all* side information into the encryption. However, given the fact that all this information needs to be stored together with the ciphertext, it also prompts the question of how much is a user willing to disclose, considering that only the baseline methods can predict, with high accuracy, the correct sequence (e.g. with 20% sparse SNVs available to the attacker, her accuracy is, on average, over 82%). Users who have already used GenoGuard for long-term encryption purposes need to be aware that if further genomic information can be obtained by the attacker, it will severely diminish the security of the system.

Chapter 6

# Suitability of Generative Models for Genomic Data Applications

Every day over 2.5 quintillion bytes of data are created [115]; however, the value of this data is maximized only with the ability to analyze it and provide meaningful insight from it, ultimately facilitating research and meaningful analytics. In this context, data collection and data sharing often constitute necessary steps to enable progress of businesses and society. As a result, entities are often willing or compelled to provide access to their datasets, e.g., to enable analysis by third parties.

Alas, data sharing does not come without privacy risks. The GDPR, which was introduced in the EU, also puts pressure on businesses to be more responsible, more accountable, and more transparent with respect to personal information and privacy laws. In an attempt to mitigate these risks, several methods have been proposed, e.g., anonymizing datasets before sharing them. However, as pointed out on several occasions [21], in practice, anonymization fails to provide realistic privacy guarantees. Another approach has been to release aggregate statistics, but this is also vulnerable to a number of attacks, e.g., membership inference (where one could test for the presence of a given individual's data in the aggregates) [148].

More promising attempts come from providing access to statistics obtained from the data, while adding noise to the queries' response, in such a way that "differential privacy" [60] is guaranteed—we describe differential privacy in Section 2.5. However, this approach generally lowers the utility of the dataset, especially on high dimensional data. Additionally, by allowing unlimited non-trivial queries on a dataset can reveal the whole dataset, so this approach needs to keep track of the privacy budget over time.

An alternative approach for addressing these issues is to release realistic synthetic data using privacy-preserving generative models. Generative models yield new samples that follow the same probabilistic distribution of a given training dataset. The intuition is that entities can train and publish the model, but not the original data, so that anybody can generate a synthetic dataset resembling the data it was trained on. Crucially, the model should be published while also guaranteeing differential privacy.

In this chapter, our focus is on generative models and the synthetic data output by them. We first analyze state of the art generative models from an utility perspective on different types of datasets such as financial data, images and locations. Then, we propose a framework for quantifying the privacy risk arising from publishing synthetic datasets under membership inference. Finally, we perform a measurement study focused on state of the art generative models specifically geared for human genome data, from both utility and privacy perspectives

## 6.1 Utility of Generative Models

In this section, we investigate the real-world feasibility of four recently proposed approaches to private synthetic data generation. We evaluate such approaches extensively on different types of data and data analysis tasks. Moreover, we provide an empirical evaluation of the moments accountant method, highlighting the range of privacy guarantees that it yields. Overall, we show that a generic approach applicable across a wide range of datasets and tasks might be too much to ask. However, our experiments also suggest that satisfactory trade-offs between utility and privacy for private data synthesis are achievable in specific settings.

### 6.1.1 A Discussion of the the Moments Accountant Method

In Section 2.5.2, we have defined the moments accountant method proposed in [13]. Before we dive into our utility analysis, we discuss and study the moments accountant's effect on the privacy budget, aiming to provide a better understanding of how all variables used in the moment accountant influence the overall privacy budget of the dataset.

The steps for computing the value of $\varepsilon$, given $\delta$, the sampling ratio $q$, the noise multiplier $\sigma$, the number of epochs $ep$, and the number of orders for the moment accountant $\lambda$ are shown in Algorithm 2. We evaluate the effect of different parameters on the minimum value of $\varepsilon$, by varying the value of one of the parameters, and keeping all the others fixed, for different dataset sizes. For each of the cases, we also illustrate the values of 32 individual moments, similar to the analysis done in [13]. Please note that Abadi et al.'s original analysis aimed to find the best possible $(\varepsilon, \delta)$ for each of the datasets tested when testing accuracy on different datasets, whereas, we aim to provide a better understanding of the privacy budget limitations when using this method for different dataset sizes.

**Effect of the noise multiplier.** We begin by evaluating the moments accountant with respect to $\sigma$, the noise multiplier. We set the value of $\delta = \frac{1}{10 * n}$, where $n$ represents the size of the different datasets tested, the sampling ratio $q = 0.01$ and evaluate over 20 training epochs. From Figure 6.1, we observe that with $\sigma = 2$, the minimum required value for the privacy budget $\varepsilon$ is greater than 1 even for the smallest dataset tested ($n = 500$). However, when looking at higher values of $\sigma$ we can observe that the minimum value for the privacy budget decreases up to the case $\sigma = 20$, after which the epsilon values remain close to each other, even for high values for sigma ($\sigma = 10^7$). Figure 6.2 illustrates the corresponding value for the moments $\alpha$. The correlation between the values of $\alpha$ and the minimum values for $\varepsilon$ is remarkable, as $\alpha$ quickly increases for $\sigma = 2$, and for higher values of $\sigma$ there is little to no variation between the values of $\alpha$.

**Effect of $\delta$.** We then evaluate the effect of $\delta$ on the overall privacy budget. Recall from Section 2.5.1 that $\delta$ allows for a small probability of failure in the differential privacy definition, i.e. with a probability of $\delta$ a certain output of a query might be given if an individual in present in the dataset. As each individual record in the dataset has this probability of failure, this will happen, on average, $\delta \cdot n$ times.

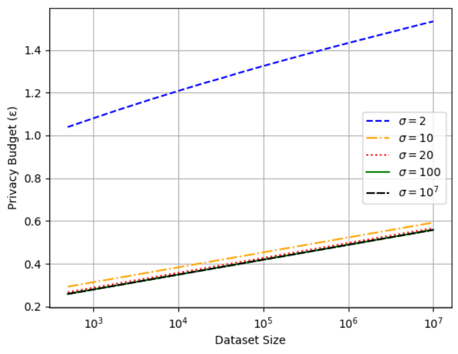


**Figure 6.1:** The minimum value of $\varepsilon$, calculated by the moments accountant, for different dataset sizes, with $\delta = \frac{1}{10*n}$, where $n$ is the dataset size, for 20 training epochs, with sampling ratio $q = 0.01$.

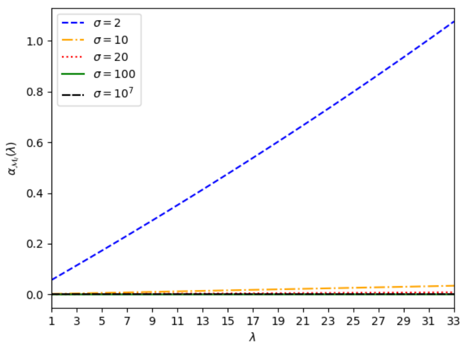


**Figure 6.2:** The values of the moments accountant $\alpha_{\mathscr{M}_i}(\lambda)$, for the minimum values of $\varepsilon$, with a noise multiplier of $\sigma = 20$.

Hence $\delta \cdot n$ needs to be small, as to not let any users at risk, and $\delta$ needs to be chosen based on the size of the dataset.

Given that $\varepsilon$ is computed as a function of $\delta$ in the moments accountant, the privacy budget is highly dependent on the size of the dataset. In this setting, we evaluate 20 training epochs, with a noise multiplier $\sigma = 20$ and a sampling ratio $q = 0.01$. From Figure 6.3, we observe that the minimum value for $\varepsilon$ increases with lower values for $\delta$. We also illustrate the corresponding values of the moments, $\alpha$, corresponding to the minimum values of $\varepsilon$ over 32 moments in Figure 6.4. These values are independent of the dataset size, hence independent from $\delta$.

**Effect of Number of Epochs.** We also studied the effect of the number of epochs on the privacy budget. However, even though there is a small variation on the privacy budget for different number of epochs, the effect of this parameter is not as prominent as the other parameters presented in this section.

**Effect of Sampling Ratio.** Finally, we look at the effects of different sampling ratios ($q$) on the overall privacy budget. In Figure 6.5, we keep the $\delta$ parameter fixed at $\frac{1}{10 \cdot n}$, the noise multiplier $\sigma = 20$, and plot the minimum value of $\varepsilon$ when varying the sampling ratio $q$. Here we see the increase in the minimum value of epsilon is also correlated with an increase in the sampling ratio $q$. The value of

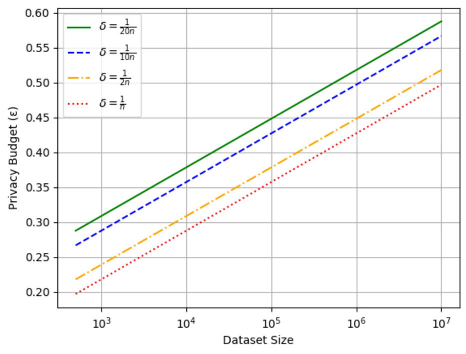


**Figure 6.3:** The minimum value of $\varepsilon$, calculated by the moments accountant, for different dataset sizes, where $n$ is the dataset size, for 20 training epochs, with the sampling ratio $q = 0.01$, and the noise multiplier $\sigma = 20$.

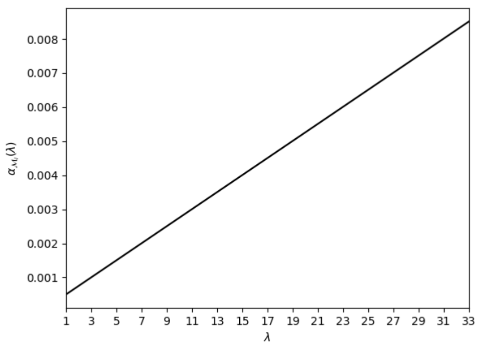


**Figure 6.4:** The values of the moments accountant $\alpha_{\mathcal{M}_i}(\lambda)$, for the minimum values of $\varepsilon$, with a noise multiplier $\sigma = 20$ and sampling ratio $q = 0.01$.

the sampling ratio also affects the value of the moments accountant, as clear from Figure 6.6. For a small sampling ratio (i.e. $q = 0.01$ to $q = 0.1$), we find that the values of the moments $\alpha$ are close to 0, however, with increasing batch size, these values increase quickly, making the minimum value of the privacy parameter $\varepsilon$ close to 1 even for small dataset sizes ($n \leq 10^3$).

**Remarks.** The question of how to properly set the privacy parameter has been present since the introduction of differential privacy and a vast amount of work has been dedicated to finding ways of setting $\varepsilon$ [120, 90, 109]. Overall, our analysis further emphasizes that differential privacy is not a purely "out-of-the-box" tool, but a complex process to satisfy a definition where different parameters affects the overall privacy budget. Therefore, before applying differential privacy to a dataset, one has to be aware the trade-offs between parameters *in practice* as well as the needs of the system with respect to privacy.

### 6.1.2 State-of-the-Art Approaches for Privacy-Preserving Data Synthesis

In this section, we describe the state-of-the-art approaches for privacy-preserving data synthesis, which we evaluate later in Section 6.1.5.

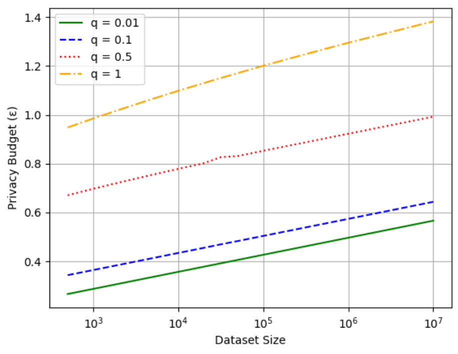


**Figure 6.5:** The minimum value of $\varepsilon$, calculated by the moments accountant, for different dataset sizes, with $\delta = \frac{1}{10*n}$, where $n$ is the dataset size, for 20 training epochs, with noise multiplier $\sigma = 20$.

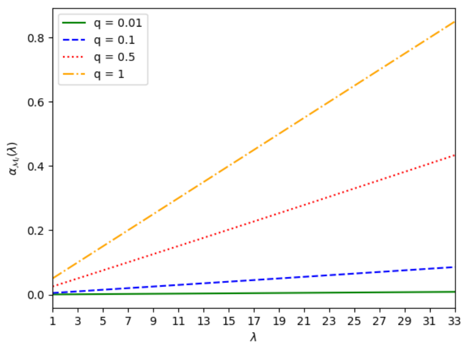


**Figure 6.6:** The values of the moments accountant $\alpha_{\mathscr{M}_i}(\lambda)$, for the minimum values of $\varepsilon$, with a noise multiplier of 20.

**PrivBayes.** PrivBayes [190] is a solution for releasing a high-dimensional dataset $D$ in an $\varepsilon$-differentially private manner. It involves three phrases:

1. A $k$-degree Bayesian network $\mathscr{N}$ is built over the attributes in $D$ using an $\frac{\varepsilon}{2}$-differentially private method ($k$ is a small value, chosen automatically by PrivBayes).

2. An $\frac{\varepsilon}{2}$-differentially private algorithm is used to generate a set of conditional distributions of $D$, such that for each attribute-parent (AP) pair $(X,\Pi)$ in $\mathscr{N}$, we have a noisy version of the conditional distribution $\Pr[X_i|\Pi_i]$.

3. The Bayesian network $\mathscr{N}$ and the $d$ noisy conditional distributions are used to derive an approximate distribution to generate a synthetic dataset $D^*$.

The model is first constructed in a non-private manner, using standard notions from information theory to construct the $k$-degree Bayesian network $\mathscr{N}$ on a dataset $D$, containing a set of $\mathscr{A}$ attributes. The mutual information between two variables is denoted as :

$$I(X,\Pi) = \sum_{x\in\text{dom}(X)} \sum_{\pi\in\text{dom}(\Pi)} \Pr[X = x, \Pi = \pi] log_2 \frac{\Pr[X=x,\Pi=\pi]}{\Pr[X=x]\Pr[\Pi=\pi]},$$

**Algorithm 3** Greedy Bayes

```
1: Initialize $\mathcal{N} = \emptyset$ and $V = \emptyset$
2: Randomly select an attribute $X_i$ from $\mathcal{A}$; add $(X_i, \emptyset)$ to $\mathcal{N}$; add $X_i$ to $V$
3: for $i = 2$ to $d$ do
4:     Initialize $\Omega = \emptyset$
5:     for each $X \in \mathcal{A} \setminus V$ and each $\Pi \in \binom{V}{k}$ do
6:         $\Omega = \Omega \cup (X, \Pi)$
7:     Select a pair $(X_i, \Pi_i)$ from $\Omega$ with maximal mutual information $I(X_i, \Pi_i)$
8:     Add $(X_i, \Pi_i)$ to $\mathcal{N}$; add $X_i$ to $V$
   return $\mathcal{N}$
```

where $\Pr[X,\Pi]$ is the joint distribution of $X$ and $\Pi$ and $\Pr[X]$ and $\Pr[\Pi]$ are the marginal distributions of $X$ and $\Pi$.

The proposed non-private algorithm "GreedyBayes" – see Algorithm 3 – extends the Chow and Liu algorithm [49] for higher values of $k$. The Bayesian network is built by greedingly picking the next edge based on the maximum mutual information. First, the network $\mathcal{N}$ is initialized to an empty set of AP pairs. The set of attributes whose parents were fixed in the previous step $(V)$ is also initialized to an empty set. Then an attribute is randomly chosen, and its parent is set to the empty set. The AP pair obtained is added to $\mathcal{N}$ and the attribute to $V$. For all the next $d-1$ steps, an AP pair is added to $\mathcal{N}$ based on the mutual information, i.e. the edge which maximizes $I$ is selected. Each AP-pair is chosen such that *i*) the size of the parent set is less than or equal to $k$ and *ii*) $\mathcal{N}$ contains no edge from the attribute at the current step to any of the attributes added at previous steps.

However, both $I$ and the best edge are data sensitive, so GreedyBayes is adapted to be differentially private. Each AP pair $(X,\Pi) \in \Omega$ is inspected and the mutual information between $X$ and $\Pi$ is calculated. After that, an AP pair is sampled from $\Omega$ such that the sampling probability of any pair $(X,\Pi)$ is proportional to $exp(\frac{I(X,\Pi)}{2\Delta})$, where $\Delta$ is the scaling factor. In order for the Bayesian network to satisfy $\frac{\varepsilon}{2}$-differential privacy, $\Delta$ is set to

$$\Delta = \frac{2(d-1)S(I)}{\varepsilon},$$

where $S_1(I)$ denotes the $L_1$ sensitivity of $I$. For the mutual information $I$, we have:

$$S_1(I(X,\Pi)) = \begin{cases} \frac{1}{n}\log_2 n + \frac{n-1}{n}\log_2 \frac{n}{n-1}, \text{if } X \text{ or } \Pi \text{ is binary} \\ \frac{2}{n}\log_2 \frac{n+1}{2} + \frac{n-1}{n}\log_2 \frac{n+1}{n-1}, \text{ otherwise} \end{cases}$$

However, since $S(I) > \frac{\log_2 n}{n}$, the range of $S$ can be quite large compared to the range of $I$. Hence, the authors propose the use of a novel function $F$ that maps each AP pair $(X,\Pi)$ to a score, such that *i)* $F$'s sensitivity is small (with respect to the range of $F$). and *ii)* if $F(X,\Pi)$ is large then $I(X,\Pi)$ tends to be large.

In order to define $F$, first the maximum joint distribution is defined. Given an AP pair $(X,\Pi)$, a maximum joint distribution $\Pr^\diamond[X,\Pi]$ for $X$ and $\Pi$ is one that maximizes the mutual information between $X$ and $\Pi$. In other words, if $|dom(X)| \leq |dom(\Pi)|$, $\Pr^\diamond[X,\Pi]$ is a maximum joint distribution if and only if *i)* $\Pr^\diamond[X=x] = \frac{1}{|dom(X)|}$ for all $x \in X$, and *ii)* for all $\pi \in dom(\Pi)$, there is at most one $x \in dom(X)$ with $\Pr[X=x,\Pi=\pi] > 0$.
Let $(X,\Pi)$ be an AP pair and $\mathscr{P}^\diamond[X,\Pi]$ be the set of all maximum joint distributions for $X$ and $\Pi$. Then $F$ is defined as:

$$F(X,\Pi) = \frac{1}{2}\min_{\Pr^\diamond \in \mathscr{P}^\diamond} ||\Pr[X,\Pi] - \Pr^\diamond[X,\Pi]||_1$$

The function $F$ defined this way has a smaller sensitivity than $I$, namely $S(F) = \frac{1}{n}$. In order to give a computation of $F$, let $(X,\Pi)$ be an AP pair and $|\Pi| = k$. Then, the joint distribution $\Pr[X,\Pi]$ can be represented as a $2\times 2^k$ matrix, where all elements sum up to 1. In order to identify the minimum $L_1$ distance between $\Pr[X,\Pi]$ and a maximum joint distribution $\Pr^\diamond[X,\Pi]$,the distributions in $\mathscr{P}^\diamond$ are partitioned into a number of equivalence classes and then $F(X,\Pi)$ is computed by processing each equivalence class individually.

In a nutshell, PrivBayes is an $\varepsilon$-differentially private method for high dimensional data using a Bayesian Network. The network is used to model the distribution between the attributes of the data. The distribution of the data is approximated using low dimensional marginals and then noise is added to satisfy differential privacy. Finally, the samples are drawn from the differentially private data for release. One of the considered drawbacks of this approach is the addition of too much noise during the network construction, which might make the approximation of the data distribution inaccurate.

The implementation for this method is available from [2].

**Priv-VAE.** In [16], Acs et al. present a technique for privately releasing generative models so that they can be used to generate an arbitrary number of synthetic datasets. It relies on generative neural networks to model the data distribution on various kinds of data, such as images or transit information. As the training procedure is differentially private with respect to the training data, any information derived from the generative models is also differentially private (by the post-processing property of differential privacy), including any dataset produced from them.

More concretely, the non-private version of the proposal works as follows: the dataset is partitioned into $k$ clusters $\widehat{D_1}, \widehat{D_2}, \ldots, \widehat{D_k}$, which are used to train $k$ distinct generative models, where the parameters of the resulting models are denoted $\theta_1, \ldots, \theta_k$. The data samples within a cluster are similar. $\theta_1, \ldots, \theta_k$ are learned using gradient descent. The generative models are then released, so that they can be used by third parties to generate synthetic data.

Note that both the clustering step and the training of the generative models inspect the data, and thus should be made differentially private. The clustering is performed using a differentially private kernel $k$-means algorithm which first transforms the data into a low-dimensional representation using the randomized Fourier feature map and the standard differentially private $k$-means is applied on these low dimensional features. To select the number of clusters $k$, Acs et al. rely on dimensionality reduction algorithms (e.g. t-SNE [114]). Then, to train a generative model for each cluster, they use the differentially private gradient descent procedure from Algorithm 1. In particular, authors experiment with Restricted Boltzmann Machines (RBMs) and Variational Autoencoders (VAEs). They evaluate the clustering accuracy of their algorithm on the MNIST dataset (for which they use $k = 10$) and their model on both a Call Data Records and a transit dataset, showing that the average relative error of their model outperforms MWEM [80].

Finally, note that the implementation of this approach is available upon request from the authors of [16].

**DP-SYN.** Abay et al. [14] present a framework combining private convolutional neural network with the private version of the iterative expectation maximization algorithm Dp-Em [139]. The dataset is first partitioned according to every instance's label, and then an $(\frac{\varepsilon}{2}, \frac{\delta}{2})$-differentially private autoencoder is built for each label group. The private latent representation is then injected into an $(\frac{\varepsilon}{2}, \frac{\delta}{2})$-differentially private expectation maximization function (Dp-Em [139]). The Dp-Em function detects different latent patterns in the encoded data and generates output data with similar patterns, which is then decoded to obtain the synthetic data.

DP-SYN uses a different approach to partitioning as opposed to Priv-VAE. The former partitions the data according to each of the label and the model is then trained on the labeled partitions, while the latter does so randomly, and uses differentially private k-means clustering for clustering the data samples.

The authors include an extensive experimental evaluation on nine datasets for binary classification tasks, comparing their results with four state-of-the-art techniques, including PrivBayes [190] and Priv-VAE [16]. All the experiments are run for 10 rounds, and only the best results for each algorithm are being recorded.

DP-SYN's implementation was originally available from https://github.com/ncabay/synthetic_generation, but as of March 2019 it is no longer so.

**Synthesis of Location Traces (Syn-Loc).** Finally, we look at the work by Bindschaedler and Shokri [31], which aims to generate fake, yet semantically plausible privacy-preserving location traces.

Bindschaedler et al. [32] formalize the concept of *plausible deniability* in the context of data synthesis, as a new privacy notion for releasing privacy-preserving datasets. More precisely, it is presented as a formal privacy guarantee such that an adversary (with no access to background knowledge) cannot deduce that a particular datapoint within a dataset was "more responsible" for a synthetic output than a collection of other datapoints.

More formally, plausible deniability is defined as follows. For any dataset $D$, with $|D| \geq k$, and any record $y$ generated by a probabilistic generative model $M$

such that $y = M(d_1)$ for $d_1 \in D$, we state that $y$ is releasable with $(k, \gamma)$ -plausible deniability if there exist at least $k-1$ distinct records $d_2, \ldots, d_k \in D \setminus \{d_1\}$ such that:

$$\gamma^{-1} \leq \frac{\Pr[y=M(d_i)]}{\Pr[y=M(d_j)]} \leq \gamma,$$

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

In short, a synthetic record provides plausible deniability that the synthetic data record could have been generated by a number of real data points.

Plausible deniability has been shown to satisfy differential privacy guarantees if randomness is added to the threshold $k$. In fact, the authors show that if Laplacian Noise Lap($\frac{1}{\varepsilon_0}$) is added to $k$, then, their system satisfies $(\varepsilon, \delta)$ differential privacy, with $\varepsilon = \varepsilon_0 + \log \frac{\gamma}{t}$ and $\delta = e^{-\varepsilon_0(k-t)}$, for any integer $t$, with $1 \leq t \leq k$. One of the main differences between plausible deniability and differential privacy for generative models is the lack of noise added to the generated data.

In order to describe the data synthesis algorithm, the paper first introduces two mobility metrics that capture how realistic a synthetic location trace is with respect to geographical semantic dimensions of human mobility. Then, it constructs a probabilistic generative model that produces synthetic, yet plausible traces. It is built using a dataset of real locations used as seeds, referred to as the *seed dataset*. For each set in the seed dataset, a probabilistic mobility model is computed, representing the visiting probability of each location and the transition probability between locations.

Generating the synthetic traces starts by transforming a real trace (taken as seed) to a semantic trace. The equivalent of semantic classes is created using the k-means clustering algorithm, where the number of clusters is chosen such that it optimizes the clustering objective. The seed is then converted to a semantic seed by replacing each location in the trace with all its semantically equivalent locations. Then, some randomness is injected into the semantic seed. Any random walk on the semantic seed trace that travels through the available locations at each time instant is a valid location trace that is semantically similar to the seed trace. Then, the semantic trace is decoded into a geographic trace in order to generate traces that are plausible according to aggregate mobility models. To generate multiple traces for each seed, the Viterbi algorithm with added randomness to the trace reconstruction is used.

Each of the generated traces is tested to ensure statistical dissimilarity and plausible deniability. Statistical dissimilarity ensures a maximum bounds on both the similarity between a fake trace and the seed from which it was generated, and between the intersection of all fake traces generated from a specific seed. Plausible deniability means that, for any fake trace generated from a seeds, there are at least a number of alternative seeds that could have generated it.

The most notable difference between this model and the other models presented in this section is that it is a seed-based model, where it takes a data record as a seed and generates synthetic data from that data record. In contrast, the other models model the synthetic data based on the statistical properties of the original data.

The protocol was evaluated on the Nokia Lausanne Dataset [146], containing a combination of GPS coordinates, WLAN and GSM identifiers for users, with the location being reported every 20 minutes. The implementation of the protocol is available from https://vbinds.ch/node/70.

### 6.1.3 Datasets

We use several datasets for our experimental evaluation, which we group into 4 categories. For each of the datasets, we also consider different tasks, following common experiments used in literature.

1. *Financial data.* We use two of the datasets from the UCI Machine Learning Datasets [57], namely: i) the German Credit dataset, with anonymized information of 1,000 customers, having 20 features and classifying customers as having good or bad credit risk; and ii) the Adult dataset, with information from 45,222 individuals, extracted from the 1994 US census, with 15 features, indicating whether the income of an individual exceeds 50,000 US dollars.

2. *Images.* We use the MNIST dataset [108], a public image dataset, which includes 28$\times$28-pixel images of handwritten digits, containing 70,000 samples.

The main classification task usually performed on this data set is dataset is to correctly classify the handwritten digit in the image.

3. *Cyber threat logs.* We rely on the DShield [8] data collected over 10 days, with approximately 5M entries collected each day. Each entry contains the Id, date, source IP, source port, target port, target IP of an attack, whenever an alert has been sounded by the firewall. This dataset has been often used in the context of predictive blacklisting [163], i.e., forecasting which IPs will attack a target.

4. *Location data.* We use the San Francisco cabs dataset [146], containing mobility traces recorded by San Francisco taxis. This has often been used to predict next locations, identifying points of interests, etc. [148].

### 6.1.4 Experimental Setup and Objectives

We aim to evaluate the performance of the synthetic datasets for different tasks to give a concrete overview of their usability in practice. Different datasets have different statistical requirements, hence we aim to provide an extensive analysis to cover multiple bases. For each of the datasets, we test multiple values for $\varepsilon$, while keeping $\delta$ fixed at $\frac{1}{10 \cdot n}$, allowing us to analyze the quality of the synthetic data under different levels of noise.

**Classification tasks.** We evaluate the synthetic data by training a discriminative model on it and evaluate its accuracy for classification on real test data. This should highlight whether the quality of the original data is preserved when using synthetic data. For this task, we compare our results to two baselines: 1) the original data, i.e., we evaluate how well the discriminative model performs when trained on the synthetic data as opposed to being trained on the original dataset; and 2) a model that would randomly assign a prediction based on the original distributions of the data. Any of the models that report an accuracy lower than this baseline are considered unsuitable for any classification task. In our evaluation, we will refer to the former as "Original Data" and to the latter as "Lower Baseline". We use this methodology to evaluate the financial data dataset and the cyber threats logs, using an SVM classifier.

**Linear regression.** We also use the synthetic data generated by each of the models and train a linear regression model on them. We evaluate the resulting model on a real testing data, vis-à-vis the accuracy of the predictions made. Similar to the classification tasks, we use the "Lower Baseline" and "Original Data" as baselines. We do this for the image and the German Credit dataset.

**Prediction.** Inspired by the work of Melis et al. [116] in the context of collaborative predictive blacklisting, we aim to analyze the performance of synthetic data in forecasting future attack sources for the cyber threat logs dataset. We use Melis et al.'s implementation from https://github.com/mex2meou/collsec.git to evaluate the synthetic data obtained from each of the models under their $k$-nearest neighbors approach. We evaluate the performance of the synthetic data by looking at the true positive rate for predicting future attacks.

**Clustering.** For location data, we evaluate if the synthetic data preserves the properties of the original data, specifically, extracting the points of interest and comparing the results to the points of interest from the original dataset.

By comparison, the NIST challenge [173] has similar criteria: the submitted algorithms must be able to preserve the balance of utility and privacy for regression, classification and clustering, but also for evaluation when the research question is unknown.

### 6.1.5 Experimental Evaluation

**Financial Data: German Credit.** In order to evaluate the categorical attributes of the German Credit dataset, we use the numeric encoding provided in [57]. We split the dataset into 70% training data and 30% testing data.

First, we train an SVM model on both synthetic data generated by each of the algorithms and on the original data, and report the accuracy results of the classification in Figure 6.7. The accuracy of the models improves with less noise added to the model (i.e. higher values for $\varepsilon$), and, among the tested models, DP-SYN

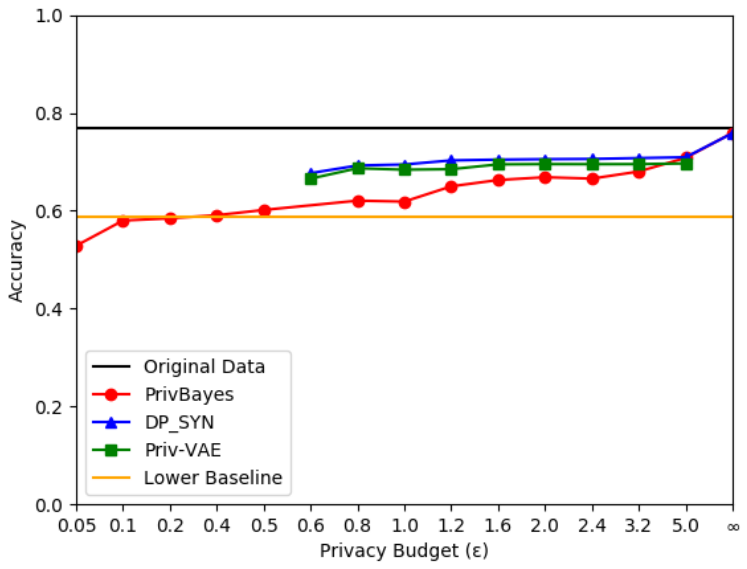


**Figure 6.7:** SVM accuracy results for German Credit Dataset with $\delta$ fixed at $\frac{1}{10 \cdot n}$, for varying values of $\varepsilon$.

obtains the highest accuracy. Priv-VAE has accuracy close to DP-SYN, however, when constructing the confusion matrices for Priv-VAE, we see that it fails to classify the German Credit dataset even in less noisy settings, and classifies most datapoints within one class (in this case it classifies most customers as having bad credit score). For $\varepsilon \leq 0.5$, the synthetic data obtained from PrivBayes reports less accuracy than our lower baseline. Recall from Figure 6.1, that, with our chosen value of $\delta$, the minimum value of $\varepsilon$ for this dataset will be above 0.5, so the we do not evaluate DP-SYN and Priv-VAE on $\varepsilon < 0.5$.

In Figure 6.7, we also report a setting for $\varepsilon = \infty$. This illustrates the models' accuracy with very little or no privacy. If the models generate synthetic data under this setting, the accuracy given is close to the accuracy of the SVM model trained on the original data.

Second, we tested a logistic regression model on the dataset, and observed the accuracy. From Figure 6.8, we see that this model fails to correctly classify even the original data. For the synthetic data, in fact, it reports a better accuracy than the SVM model, however, when looking at the confusion matrices, we notice that it fails to classify the dataset, reporting most datapoints as being within one class.

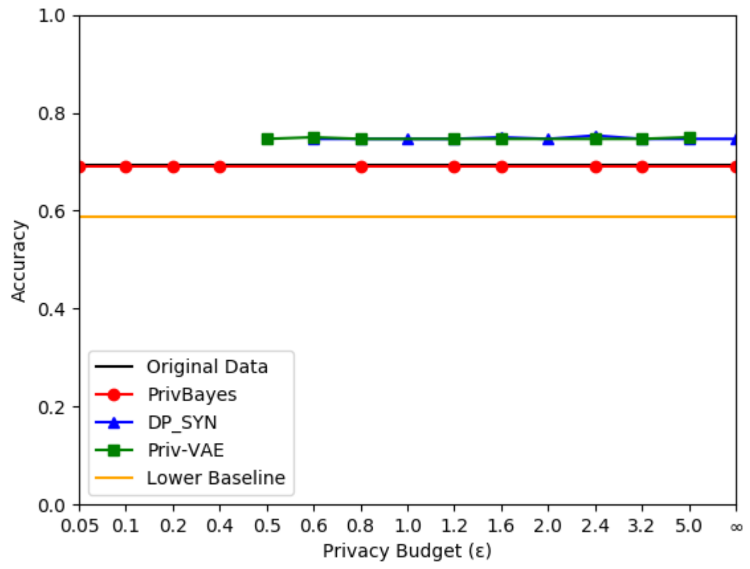


**Figure 6.8:** Linear regression accuracy results for German Credit Dataset with $\delta$ fixed at $\frac{1}{10 \cdot n}$, for varying values of $\varepsilon$.

**Financial Data: Adult Dataset.** We first encode the adult dataset, using One-Hot Encoding [81], to convert the categorical attributes of the dataset into numerical attributes. The encoding obtained has 57 attributes as opposed to 13 attributes in the original dataset. We split the dataset into 70% training data and 30% testing data.

We train an SVM model on the synthetic data obtained from each of the models and present the results in Figure 6.9. Note that PrivBayes has better accuracy with higher noise levels than DP-SYN ($\varepsilon \leq 1$). Additionally, when $\varepsilon$ is greater than 2, the accuracy of DP-SYN decreases, which is perhaps counter-intuitive, as it is expected for accuracy to increase when less noise is added to the model. Priv-VAE fails to cluster the data in this test case, often returning empty clusters during the differentially private $k$-means clustering. The accuracy of Priv-VAE, even though higher than the lower baseline used, is lower than the accuracy of the other two tested models for all tested values of $\varepsilon$.

To observe if an increase in accuracy can be correlated with increasing dataset size, we also train the SVM model on partial data, while keeping the same test dataset. In Figure 6.10, we plot the accuracy for $\varepsilon = 0.9$ for DP-SYN (the minimum

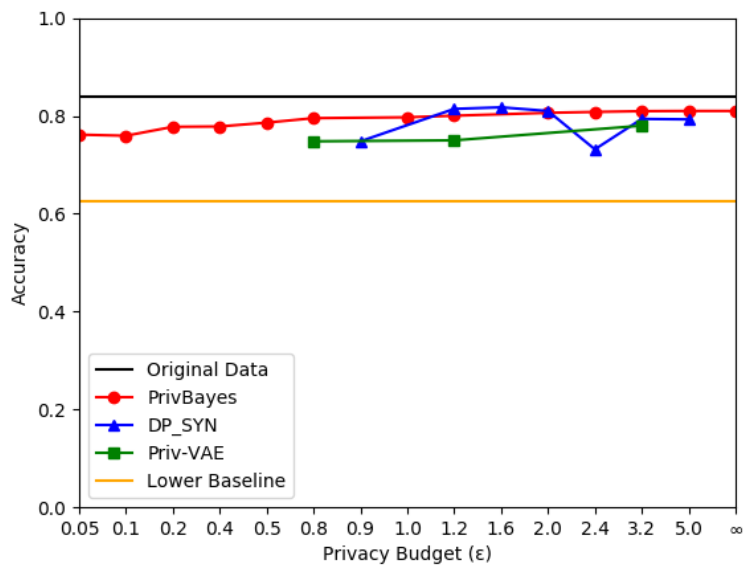


**Figure 6.9:** SVM accuracy results for Adult Dataset with $\delta$ fixed at $\frac{1}{10 \cdot n}$, for varying values of $\varepsilon$.

accepted value of $\varepsilon$ for this dataset, when $\delta = \frac{1}{10 \cdot n}$), and $\varepsilon = 0.8$ for PrivBayes. Even though in this case PrivBayes has more noise added to the model, it reports better accuracy than DP-SYN, for all partial datasets tested. The increase in accuracy for PrivBayes with larger dataset sizes is easily observed, from approximately 0.75 accuracy when 10% of the original data was used for generating the synthetic data to approximately 0.8 when 90% of the data was used. For DP-SYN, the same correlation cannot be observed, and in fact, the highest accuracy (0.75) is reported when 80% of the data was used for training.

In Figure 6.11, we increase the value of $\varepsilon$ to 1.2. We can observe that PrivBayes reports better accuracy when less than 40% of the original data was used for generating the synthetic data, and DP-SYN outperforms PrivBayes with increasing dataset size. In contrast to Figure 6.10, in this case we observe an improvement in accuracy for both models with increasing dataset sizes.

In Figure 6.12, we report the accuracy for increasing dataset sizes for $\varepsilon = 3.2$. In this case DP-SYN outperforms PrivBayes, however, neither of the models' improvement in accuracy can be correlated with increasing dataset sizes.

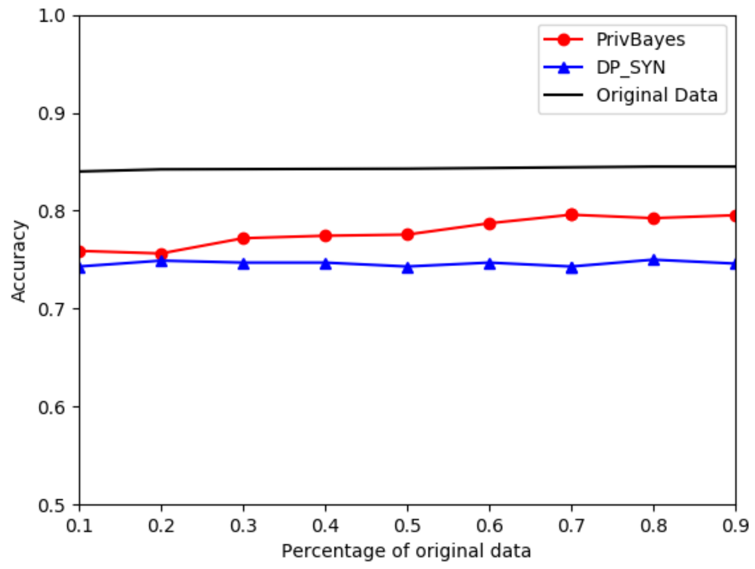


**Figure 6.10:** SVM accuracy results, for training models at a percentage of the original datasets. For DP-SYN, we have $\varepsilon = 0.9$ (the minimum value for $\varepsilon$ for this dataset, when $\delta = \frac{1}{10 \cdot n}$), and for PrivBayes $\varepsilon = 0.8$.

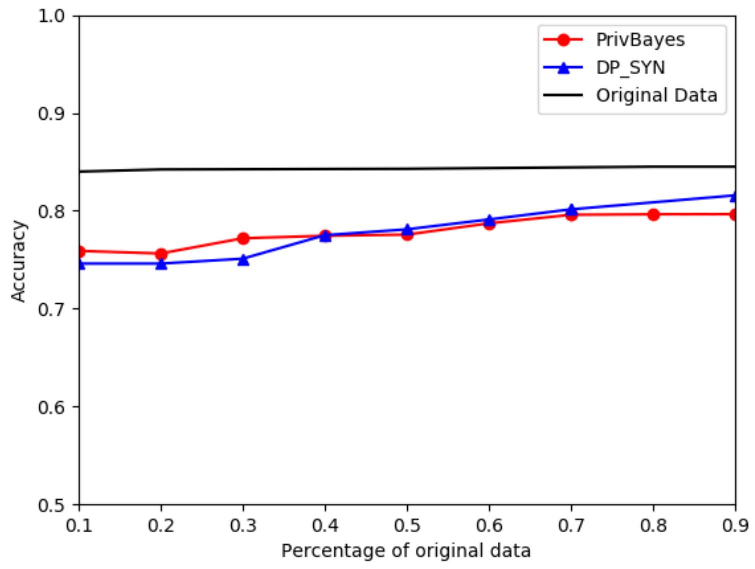


**Figure 6.11:** SVM accuracy results for training models at a percentage of the original datasets, with $\varepsilon = 1.2$.

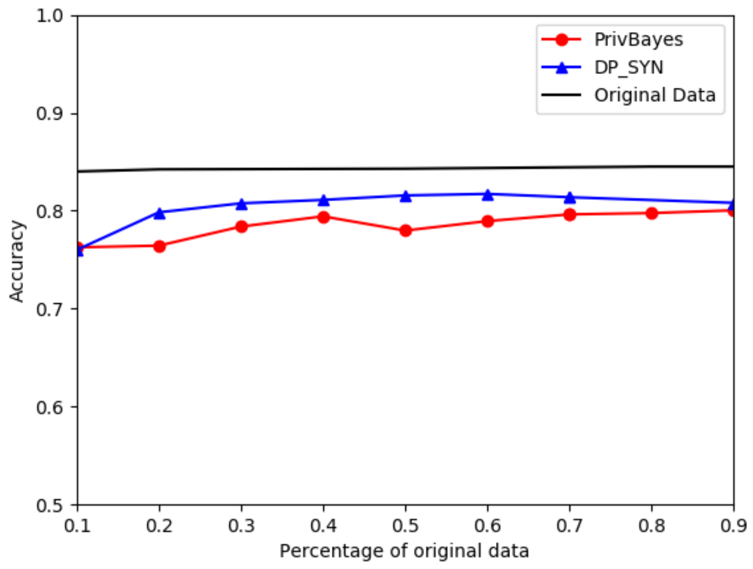


**Figure 6.12:** SVM accuracy results for training models at a percentage of the original datasets, with $\varepsilon = 3.2$.

**Images: MNIST Dataset.** We then evaluate the models on the MNIST dataset. We use the usual split for training and testing purposes, i.e. 60,000 samples for training and 10,000 samples for testing. We generate synthetic data with all three methods and then construct a linear regression model on the synthetic data for evaluating the accuracy of each of the models.

For each value of $\varepsilon$ tested we reconstruct the average image resulted in every class. When reconstructing the classes for PrivBayes, we can observe that the model did not correctly reconstruct separate class images.. As shown in Figure 6.13, all the classes seem similar, even in the lowest privacy case (i.e. $\varepsilon = 10^7$). Therefore, we split the data into separate classes and trained on each class separately for generating the synthetic data.

In Figure 6.14, after splitting, with increasing values of epsilon, we start to distinguish between different classes. For DP-SYN (see Figure 6.15), the reconstruction of each of the digit classes is clearer even for small $\varepsilon$, and close to the average class reconstruction for the original data. Priv-VAE (see 6.16) outputs a less noisier reconstruction than PrivBayes, but not as clear as DP-SYN. Finally, in Figure 6.17, we report the accuracy of the linear regression models. As expected from the reconstructed samples, the accuracy of PrivBayes is very low for the noisier samples. In fact, not even for the less noisy cases, when accuracy improves, it does not match the accuracy of the linear regression model when trained on the original data. Similarly, the accuracy of the synthetic samples generated from PrivVAE is correlated with the average class reconstruction, and, in fact, has a better accuracy than PrivBayes as the value of $\varepsilon$ increases. DP-SYN obtains a higher accuracy then both the other two models, however, it still reports lower accuracy that that of the original dataset.

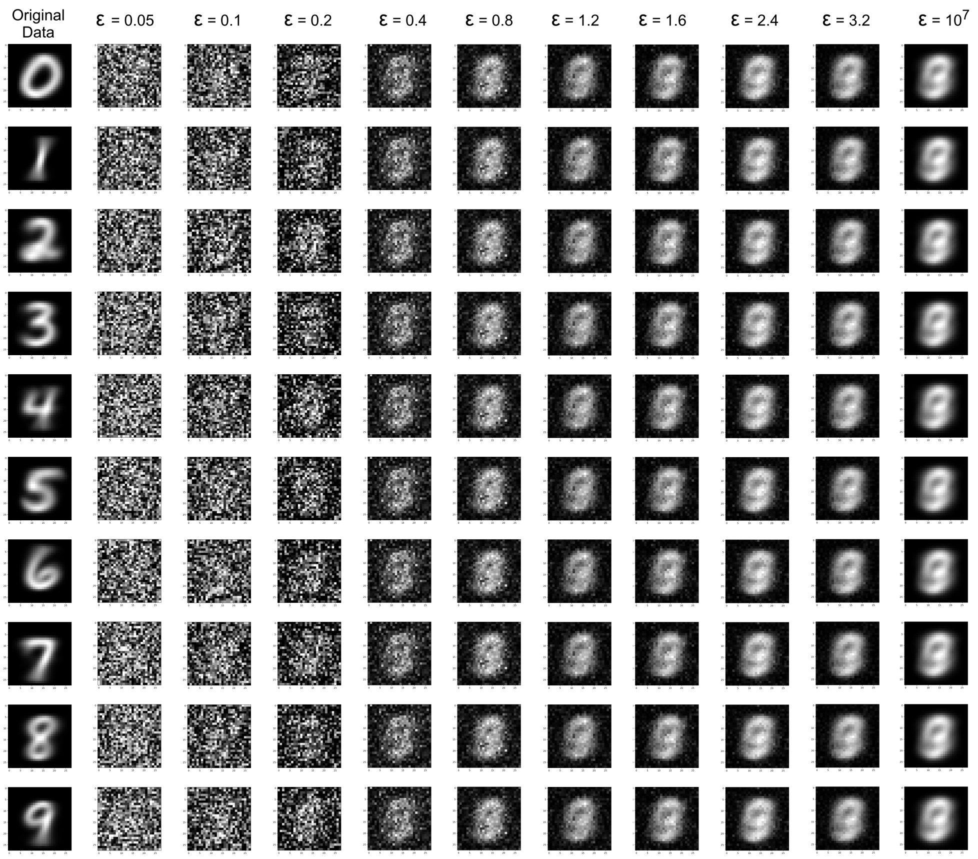


**Figure 6.13:** Average class reconstruction for PrivBayes, with no splitting before training.

**Cyber Threat Logs: DShield Log Dataset** First, we try a classification task for this dataset. We construct the dataset for this task by using the DShield logs and classifing the existing logs as threats, and adding as much non-threat traffic to the dataset. After randomizing, we split the data into 70% training and 30% testing. We train a discriminative model on the synthetic data for all models, however, none

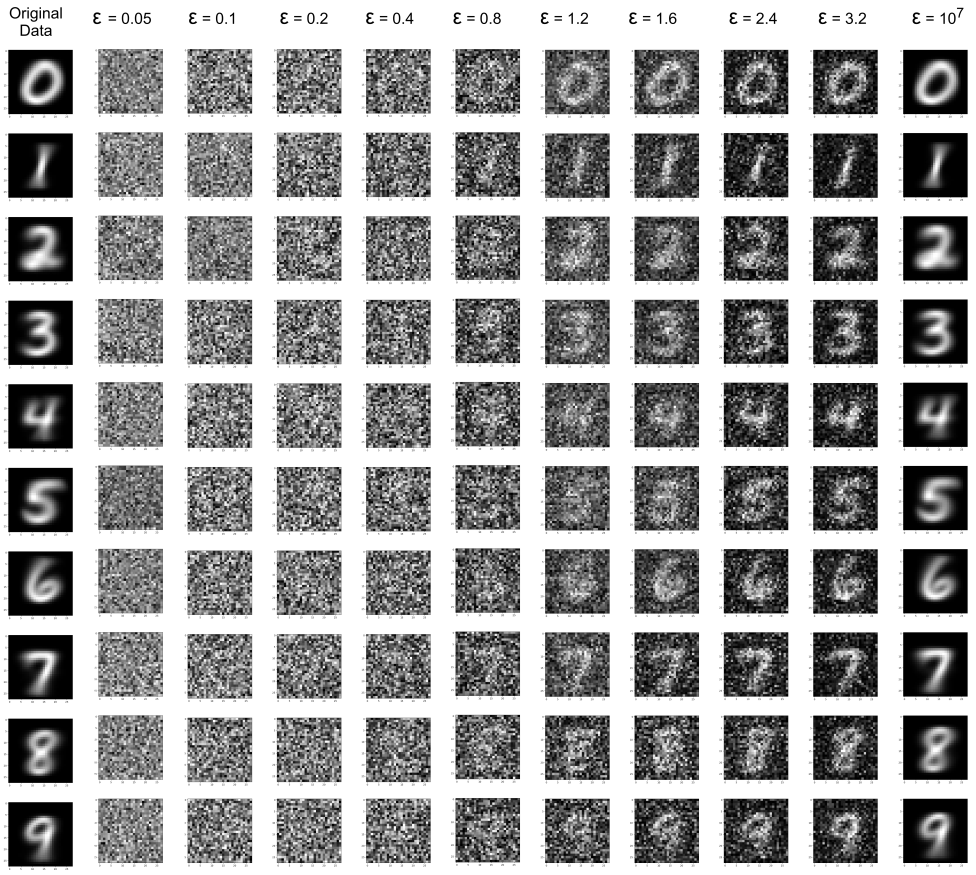


**Figure 6.14:** Average class reconstruction for PrivBayes, with splitting before training.

of them achieved accuracy better than the lower baseline.

Our second approach is to use this data set as both training and testing, using a 5 day sliding window for training the models and obtaining synthetic data, and we use the real data from the next date for testing. The aim of this is for a model trained on the synthetic data to generate new samples that could then be used to predict data found in the testing dataset.

When using the synthetic dataset obtained from PrivBayes to evaluate prediction using the k-NN approach from [116], it failed to produce any samples that would correlate with the testing data. Hence, we find that no meaningful predictions can be obtained on this synthetic dataset, regardless of noise level. DP-SYN managed to predict some of the future attacks, however, it reported a lower true positive rate (less than 50% true positive rate) than the original data. Even for this dataset, DP-SYN does not perform consistently better with less noise added to the

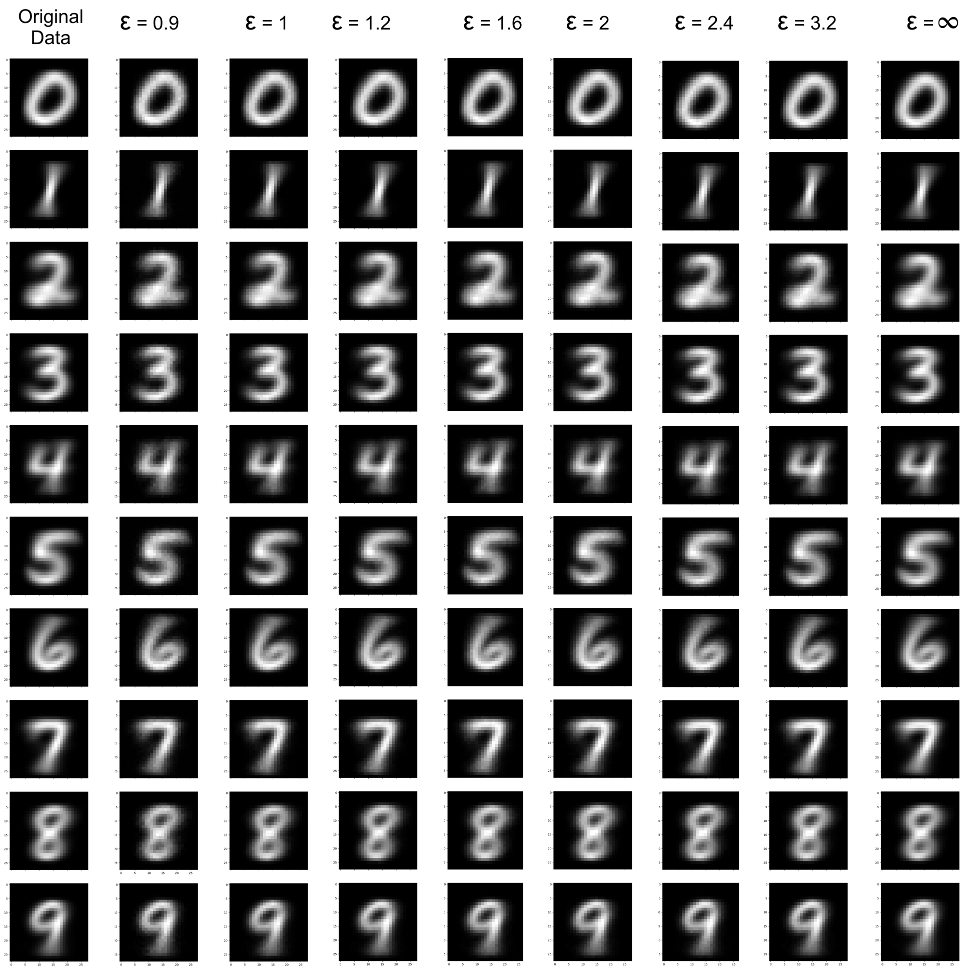


**Figure 6.15:** Average class reconstruction for DP-SYN.

model.

**Location Data: San Francisco Cabs Dataset** We generate the synthetic datasets for each of the models and plot the distribution of locations (Figure 6.18). DP-SYN fails to provide a meaningful distribution of locations, placing all locations on the same point on the map. The synthetic data generated by PrivBayes, even though more scattered on the map than DP-SYN, still fails to mimic the distribution of the original data.The synthetic data generated by Syn-Loc has a more similar distribution across the map to the original data. This is due to the more specialized model used for data generation.

We cluster the data using $k$-means clustering for extracting the points of interest on the map. We plot the clusters distribution on the map for $k = 10$ in Figure 6.19. As expected, the synthetic data generated from DP-SYN is grouped within a single

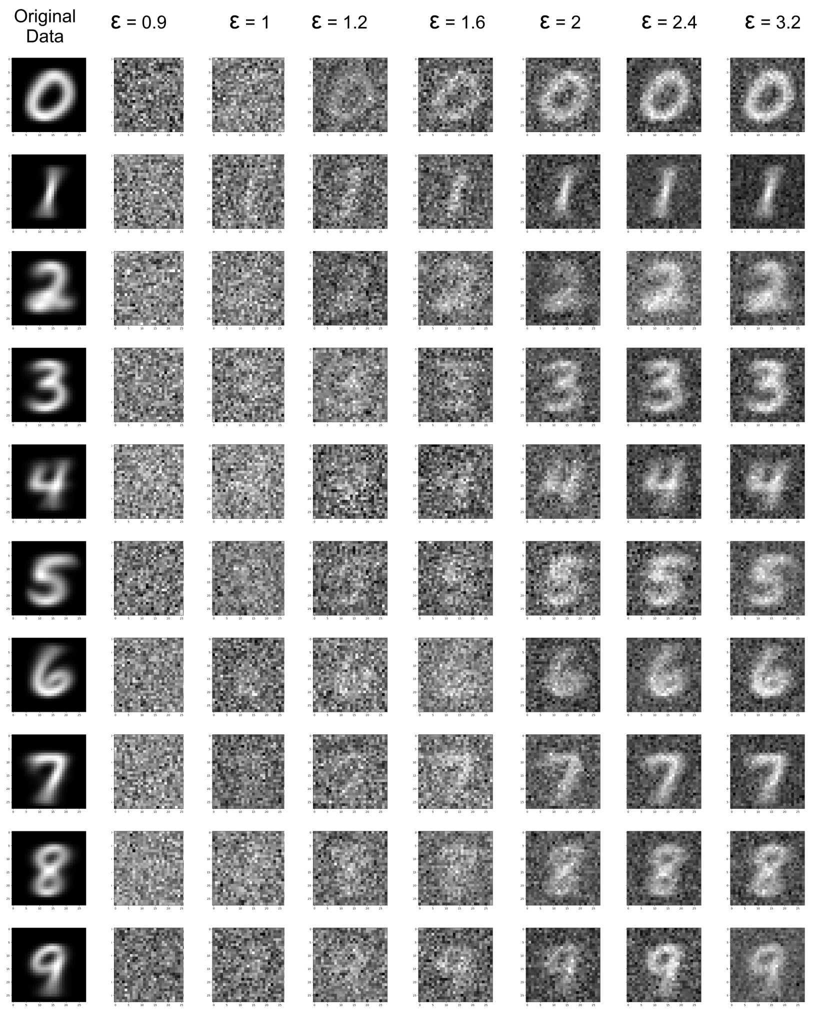


**Figure 6.16:** Average class reconstruction for Priv-VAE.

cluster. The synthetic data from PrivBayes does not generate empty clusters, but the distribution of clusters on the map does not resemble the distribution of clusters for the original dataset. The data generated by Syn-Loc provides the most similar clustering to the original dataset. In Figure 6.20, We plot the distribution of clusters for $k = 20$. In this case, not only the data from DP-SYN is grouped within one cluster, but clustering on the synthetic data from PrivBayes also return empty clusters. Again, Syn-Loc provides the most similar clustering to the original dataset.

### 6.1.6 Key Takeaways

The main goal of this section was to understand how to evaluate privacy-friendly synthetic data generation techniques. In other words, we set to determine whether

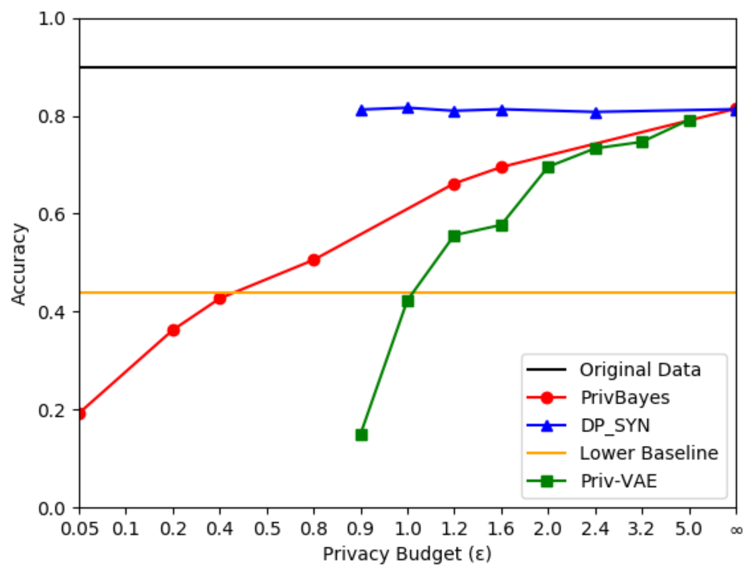


**Figure 6.17:** Linear Regression accuracy results for MNIST Dataset with $\delta$ fixed at $\frac{1}{10 \cdot n}$, for varying values of $\varepsilon$.

or not it is possible to privately synthesize data characteristics to yield translational findings to the original data. In short, the answer to this question depends on the task at hand, the size of the dataset, as well as the privacy requirements of the synthetic dataset.

We evaluated three of the privacy-preserving synthetic data generation models on binary classification tasks on two financial data datasets. The first model, PrivBayes [190], reported accuracy lower than randomly assigning labels based on original distributions of the data when trained on noisy synthetic data (i.e., $\varepsilon \leq 0.5$) for the German Credit dataset. However, when the same task is performed on a larger dataset, accuracy is better even for noisier synthetic datasets. Moreover, even on smaller samples of the adult dataset, the performance is still better than the lower baseline and reports better accuracy than the other models for noisier datasets. By contrast, DP-SYN yields better performance overall on the German Credit dataset, but lower accuracy on the larger dataset for noisy synthetic data ($\varepsilon < 1$). In fact, the average performance actually decreases for some of the less noisy synthetic datasets ($\varepsilon > 2$). Priv-VAE generates synthetic data which, even though reports testing accuracy comparable to DP-SYN, on closer inspection, classifies most datapoints within

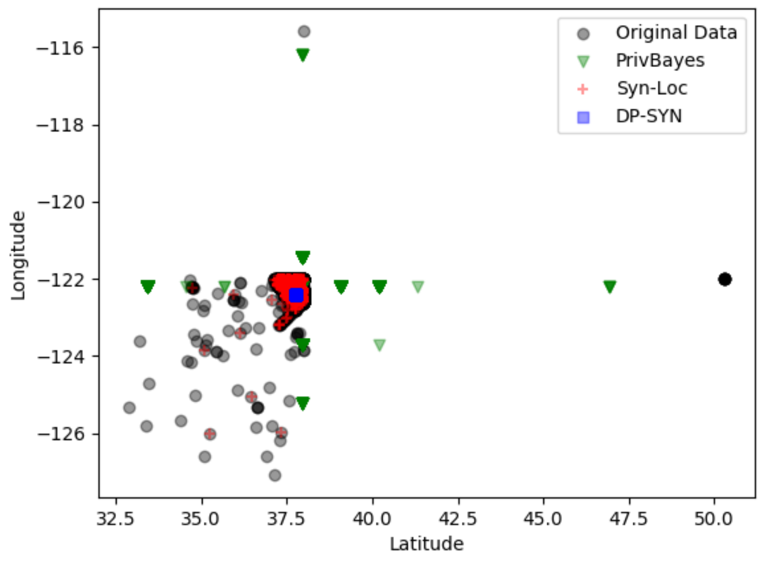


**Figure 6.18:** Distribution of locations for the San Francisco Cabs dataset.

one label.

The same models were then evaluated on the MNIST dataset, on a multi-class classification task. For this dataset, the synthetic data obtained from PrivBayes performs poorly under classification tasks. In fact, even from the synthetic data reconstruction of the average sample image for each class, it is easy to observe that the resulting images have a significant amount of noise. Moreover, we found that splitting the data into separate classes before training the model is necessary in order to be able to distinguish between the different classes. The synthetic data from DP-SYN returns better reconstructed samples, however, the minimum privacy budget for this dataset is $\varepsilon = 0.9$ for $\delta = \frac{1}{10 \cdot n}$, therefore not allowing too much noise to be added to the model.

For the cyber threat logs, none of the models tested performed well under the two tested tasks. The discriminative models trained on the synthetic datasets for classifying datapoints as threats or as benign all report a lower accuracy than the lower baseline. Even for the prediction task, none of the models achieve a meaningful true positive rate for predicting future attacks. We think this is due to the fact that the attack vectors given, based on IP and port, are quite sensitive to noise

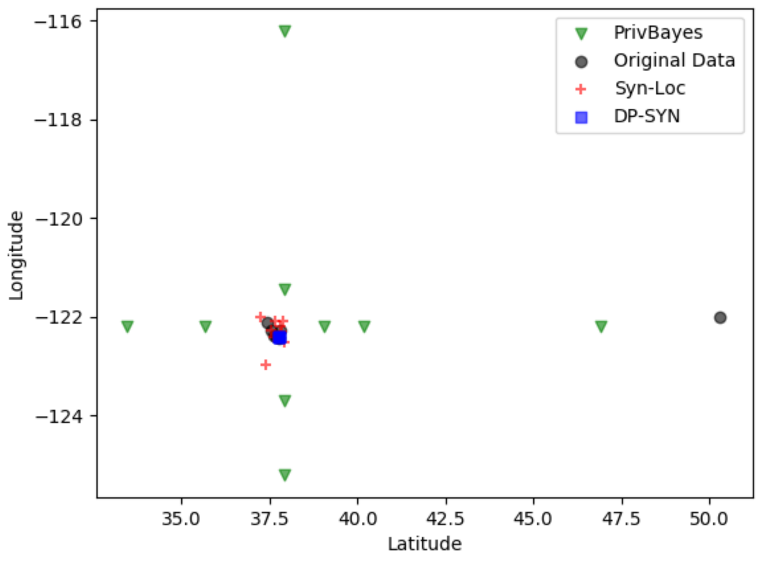


**Figure 6.19:** 10 Clusters of locations for the San Francisco Cabs dataset.

perturbations, and therefore noisy data can be unsuitable for such tasks.

For location data, the model that returns the distribution most similar to the original data is Syn-Loc. This is an expected outcome, because of the more specialized approach used for this model. DP-SYN fails to generate meaningful synthetic data, aggregating all synthetic data points within one location point. We believe that the performance of this model is worse on the location dataset compared to the other datasets because the model used to split the training data by class and generate synthetic samples for each class separately, whereas the same split is not possible in this case. PrivBayes managed to generate synthetic samples with a better distribution than DP-SYN, however not as good as Syn-Loc.

From our evaluation, we can conclude that DP-SYN can be used for image reconstructions, providing that the training can be done in classes, as it constructed the images in the MNIST dataset close to the original samples. PrivBayes provides high accuracy for binary classification tasks on large datasets, when a large noise perturbation is needed. Syn-Loc manages to simulate real location data distributions better than the other models due to its focus on location datasets.

Overall, the most encouraging results correspond to image and financial data

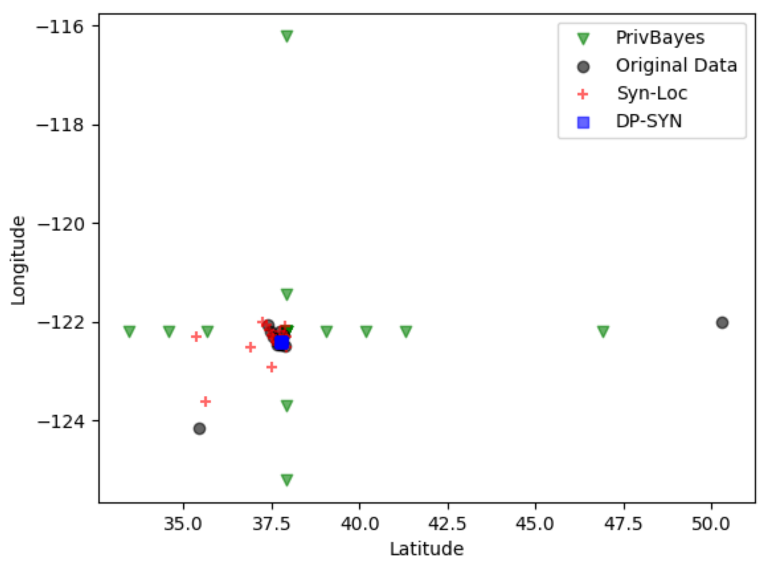


**Figure 6.20:** 20 Clusters of locations for the San Francisco Cabs dataset.

that, as for settings with good privacy guarantees – where the value of the epsilon and delta parameters of differential privacy are less than 1, and an order of magnitude smaller than the inverse of the size of the dataset, respectively – the evaluated approaches led to synthetic training datasets on which very basic models for prediction and classification incurred a 5-8 accuracy loss with respect to training on the *non-private* original data.

## 6.2 Privacy Evaluation of Synthetic Data

In this section, we propose a novel evaluation framework that addresses the shortcomings of previous evaluation approaches. It inherently models the risk of inference attacks, it can be applied to assess any type of generative model, with and without formal privacy guarantees, and measures the risk of releasing a synthetic dataset rather than a trained model. We *quantitatively* assess the privacy properties of three generative models and two differentially private variants on three realistic datasets. Our results challenge previous claims about the privacy benefits of synthetic data publishing and demonstrate that generative models often do not provide robust protection against privacy attacks on the generated datasets.

### 6.2.1 Evaluating the privacy gain of synthetic data

We introduce a novel evaluation framework to assess (claims about) the privacy properties of synthetic data publishing. The framework provides data holders and researchers with a tool to evaluate whether a synthetic dataset generated by a statistical model trained on a sensitive dataset protects the confidentiality of individual records in the raw data. In contrast to previous evaluation approaches, the framework allows to directly assess whether publishing a *single copy* of synthetic data in place of the raw data reduces the risk of private information leakage through *inference attacks*. Due to its modular nature, the framework is not limited to a unique threat model [150] but enables data holders to adapt the evaluation to any privacy concern specific to the data holder's use case. In contrast to model-specific evaluation techniques [91], our framework treats the data generating mechanism as a complete black-box and evaluates the privacy risks of synthetic data *publishing* rather than an adversary's inference power when given query access to a model [150, 82]. As the proposed evaluation method is independent of the generative model, it can be applied to synthetic data generated by models trained without any explicit privacy protection or trained under formal privacy guarantees [15, 33].

In this section, we provide a formal description of the proposed evaluation framework. In short, the framework measures the privacy gain of publishing a synthetic dataset in-place of the raw data with respect to a specific privacy concern. Each concern is modeled as a *privacy adversary* that targets an individual record and aims to infer a secret about this record. To evaluate *privacy gain*, the framework is instantiated under the pre-defined threat model and outputs an estimate about how much publishing the synthetic data instead of the raw data reduces the *privacy loss* of a chosen target record under this threat model.

Throughout this section, we use $X \sim \mathfrak{D}^n$ to refer to a generic dataset of size $n$ sampled from an unknown distribution $\mathfrak{D}$ over the data domain of the population $\mathfrak{R}$. $X$ could either be the raw data $R$, or a sanitized or synthetic version of $R$.

**Adversarial models.** In our framework, we measure the privacy gain of publishing $S$ instead of $R$ *with respect to a specific privacy concern* about the potential disclosure of a secret $t_s$ associated with an individual target record $\mathbf{t} \in \mathfrak{R}$. We model a privacy concern as an adversary $\mathscr{A}_{\mathbf{t}}(X) : X \to \hat{t}_s$ who, given access to a dataset $X$, optimizes her strategy $\mathscr{A}_{\mathbf{t}}()$ to produce an estimate $\hat{t}_s$ of the secret $t_s$ of a target record $\mathbf{t}$. We denote the adversary's training procedure that outputs the function $\mathscr{A}_{\mathbf{t}}()$ as $\mathtt{AdvTrain}(\mathbf{t},..)$. The internals of this procedure depend on the privacy concern and the specific threat model. We provide example implementations of this procedure to address concerns regarding the risk of linkability in Sections 6.2.5.

**Privacy loss.** We first define the *privacy loss* for an individual record $\mathbf{t}$ under an adversary $\mathscr{A}_{\mathbf{t}}()$ that is trying to infer secret $t_s$ linked to record $\mathbf{t}$ given access to dataset $X$.

**Definition 1** (Individual Record Privacy Loss). *Let $X \sim \mathfrak{D}^n$ be a dataset of size n,* $\mathbf{t}$ *a random record from $\mathfrak{R}$, and $t_s$ a secret associated with* $\mathbf{t}$. *We define the individual record privacy loss $\mathtt{PL}_{\mathbf{t}}(X)$ for target* $\mathbf{t}$ *under adversary $\mathscr{A}_{\mathbf{t}}(X)$ as*

$$\mathtt{PL}_{\mathbf{t}}(X) \triangleq P\left[\hat{t}_s = t_s | X\right] - P\left[t_s\right], \tag{6.1}$$

*where $\hat{t}_s$ is the guess output by $\mathscr{A}_{\mathbf{t}}(X)$, $P\left[\hat{t}_s = t_s | X\right]$ is the adversary's success probability given $X$, and $P\left[t_s\right]$ is her prior distribution over $t_s$. If $t_s$ is a continuous attribute and $P\left[t_s | X\right]$ defines a probability density function, we calculate $Pr[\hat{t}_s = t_s]$ as $Pr[\hat{t}_s \in [t_s - \varepsilon, t_s + \varepsilon]]$ for a small $\varepsilon$.*

In this definition, the adversary's prior and posterior probabilities of success take values in $[0,1]$, and thus $\mathtt{PL}_{\mathbf{t}}(X)$ ranges between $[-1,1]$. $\mathtt{PL}_{\mathbf{t}}(X) = 1$ implies that access to $X$ substantially improves the adversary's chance of making a correct guess with respect to her prior. Negative privacy loss values, $\mathtt{PL}_{\mathbf{t}}(X) < 0$, indicate that observing $X$ confuses the adversary, i.e., her success rate is smaller than her prior.

**Privacy gain.** We define the *privacy gain* for an individual record $\mathbf{t}$ when publishing dataset $X_1$ compared to $X_2$ as:

**Definition 2** (Individual Record Privacy Gain)**.** *Let $X_1 \sim \mathfrak{D}^n$ and $X_2 \sim \mathfrak{D}^m$ be two datasets from the same domain, and* **t** *a random record from $\mathfrak{R}$. The privacy gain for* **t** *of publishing $X_2$ compared to $X_1$ is defined as:*

$$\mathtt{PG}_{\mathbf{t}}(X_2, X_1) \triangleq \frac{\mathtt{PL}_{\mathbf{t}}(X_1) - \mathtt{PL}_{\mathbf{t}}(X_2)}{2}, \tag{6.2}$$

*where $\mathtt{PL}_{\mathbf{t}}(X_1)$ and $\mathtt{PL}_{\mathbf{t}}(X_2)$ denote* **t***'s privacy loss under $\mathscr{A}_{\mathbf{t}}()$ given $X_1$ and $X_2$, respectively. The multiplicative factor ½ normalizes* $\mathtt{PG}$ *to the interval* $[-1, 1]$.

Under most threat models, including our example instantiation of the linkability (6.2.5), the adversary's prior over the target's secret $\mathrm{P}[t_s]$ is independent of the published dataset $X$. In this case, the record privacy gain can directly be calculated as

$$\mathtt{PG}_{\mathbf{t}}(X_2, X_1) = \frac{\mathrm{P}[\hat{t}_s = t_s | X_1] - \mathrm{P}[\hat{t}_s = t_s | X_2]}{2}. \tag{6.3}$$

We use this definition to measure the privacy gain of publishing a synthetic dataset $S$ sampled from a generative model trained on $R$ instead of $R$ itself. $\mathtt{PG}_{\mathbf{t}}(S, R) > 0$ indicates that the synthetic dataset increases the target's privacy with respect to $R$, while $\mathtt{PG}_{\mathbf{t}}(S, R) < 0$ indicates that the synthetic dataset actually leaks more information about the target to the adversary than $R$. The extreme case of $\mathtt{PG}_{\mathbf{t}} = 1$ indicates that publishing $S$ in place of $R$ successfully prevents a breach of the target's privacy: While publishing $R$ would incur a privacy loss of $\mathtt{PL}_{\mathbf{t}} = 1$ to target record **t**, the synthetic dataset $S$ successfully mitigates the risk modeled by $\mathscr{A}_{\mathbf{t}}()$. Vice versa, $\mathtt{PG}_{\mathbf{t}} = -1$ implies that the synthetic dataset $S$ maximizes the target's privacy risk while $R$ provides maximum protection against this adversary. If $\mathtt{PG}_{\mathbf{t}} = 0$ publishing $S$ is equivalent to publishing $R$ in terms of privacy for record **t**.

**Framework implementation and practical evaluation.** We implemented the building blocks of the privacy evaluation framework as a Python library [10].

In Section 6.2.5, we use our implementation and instantiate the framework to assess the expected privacy gain for a set of randomly chosen target records under five generative model training algorithms (see 6.2.2). The expected record privacy gain provides a useful measure to not only assess the overall expected privacy gain for a dataset $R$ but also to analyze whether privacy gain is distributed uniformly across target populations.

In our experiments, we use the *expected record privacy gain* to compare the privacy protection awarded by different generative models across datasets and target populations.

**Definition 3** (Expected Record Privacy Gain)**.** *A target* $\mathbf{t}$*'s expected privacy gain under a fixed model training algorithm* $\mathtt{GM}()$ *can be estimated as*

$$\overline{\mathtt{PG}_{\mathbf{t}}} = \mathop{\mathbb{E}}_{\substack{R\sim\mathcal{D}^n_{\mathcal{R}} \\ f(R)\sim\mathtt{GM}(R) \\ S\sim\mathcal{D}^m_{f(R)}}} [\mathtt{PG}_{\mathbf{t}}(S,R)] \tag{6.4}$$

*where $R$ is the input dataset of size $n$,* $\mathtt{PG}_{\mathbf{t}}$ *is* $\mathbf{t}$*'s privacy gain from publishing $S$ instead of $R$, and $S$ a synthetic dataset generated by a generative model $f(R)$ that was obtained by running* $\mathtt{GM}(R)$*.*

### 6.2.2 Generative models

We implemented three generative models without explicit privacy protection and two models designed to offer DP. We chose models relevant to the tabular data sharing use case and to cover a wide range of model architectures. We also considered their computational feasibility and whether a working implementation was available.

`IndHist`. We adopted a simple independent histogram model in which each attribute in the tabular input data is modeled independently from Ping et al. [145]. The `IndHist` training algorithm extracts marginal frequency counts from each data attribute and generates synthetic datasets $S$ by sampling from each learned histogram independently. Continuous attributes are binned and the number of bins `nbins` is a model parameter.

`BayNet`. Bayesian networks capture correlations between attributes by factorizing the joint data distribution as a product of conditionals. The degree of the network model is a model parameter. The trained network provides an efficient way to sample synthetic records from the fitted distribution (see Zhang et al. [191] for details). We use the GreedyBayes implementation provided by Ping et al.'s DataSynthesizer [145].

`PrivBay`. PrivBayes [191] is a differentially private Bayesian network model. Both, the Bayesian network and the conditional distributions, are learned under $\varepsilon$-differentially private algorithms. The resulting $\varepsilon$-differentially private model enables to sample synthetic records without any additional privacy budget cost. We use the GreedyBayes procedure provided by Ping et al. [145] to train a differentially private version of `BayNet`.

`CTGAN`. CTGAN [182] uses mode-specific normalization of tabular data attributes to improve the approximation of complex distributions through GANs. CTGAN further uses a conditional generator and training-by-sampling to get better performance on imbalanced datasets.

`PATEGAN`. PATE-GAN builds on the Private Aggregation of Teacher Ensembles (PATE) framework [138] to achieve DP for GANs [99]. PATE-GAN replaces the discriminator's training procedure with the PATE mechanism. The trained model provides DP with respect to the discriminator's output.

### 6.2.3 Datasets

We include three tabular datasets, commonly used in the machine learning (ML) literature, in our experimental evaluation. Tabular datasets are the most relevant data type in the synthetic data publishing case. Two datasets contain financial data, and the third one contains health data:

`Adult` [105]. The Adult dataset contains information from 45,222 individuals extracted from the 1994 US Census database. Each entry consists of 15 attributes among which 6 are continuous attributes and 9 are categorical attributes.

`Texas` [167]. The Texas Hospital Discharge dataset is a large public use data file provided by the Texas Department of State Health Services. The dataset we use consists of 50,000 records uniformly sampled from a pre-processed data file that contains records from 2013. We retain 18 data attributes of which 11 are categorical and 7 continuous attributes.

`German` [87]. The original German Credit dataset contains data from 1000 individuals with 20 attributes. We use a pre-processed version of the dataset [7] which consists of 10 attributes for each individual of which 3 are continuous and 7 are categorical.

### 6.2.4 Experimental setup

We describe the experimental procedure we follow in Section 6.2.5 to empirically evaluate the privacy gain of the five generative models described above regarding the risks of membership inference using our framework implementation [10].

*Worst-case evaluation.* We evaluate the individual record privacy gain of a crafted target $\mathbf{t}_c \sim \mathscr{D}_R*$. This crafted target is designed to be an outlier, and that represents a very vulnerable target record due to its unusual distribution of attribute values. We craft this target by creating a distribution $\mathscr{D}_R*$ as follows: For each categorical attribute, we assign probability 1 to the least frequent category in the original dataset $R$. For each numerical attribute, we assign probability 1 to the maximum attribute value.

*Average case evaluation.* We run $n_R$ independent tests in which we evaluate the individual record privacy gain for a fixed set of $n_T$ target records: $T = (\mathbf{t}_1, \cdots, \mathbf{t}_{n_T})$. These records are randomly chosen from the dataset $R$. In each run, we repeat the following procedure: (1) We sample a new target training set $R^t \sim \mathscr{D}_R^n$ and use it as input to obtain five trained target models $f(R^t)$. (2) From each model $f(R^t)$, we sample $n_S$ synthetic test datasets $S_1^t, \cdots, S_{n_S}^t$ of size $m$. (3) For each $\mathbf{t} \in T$, we run an adversary $\mathscr{A}_\mathbf{t}()$ on each synthetic dataset to obtain $n_S$ independent estimates of the target record's privacy loss $\mathtt{PL}_\mathbf{t}(S_i)$. (4) Finally, we estimate a record's privacy gain $\mathtt{PG}_\mathbf{t}$ for dataset $R^t$ and trained model $f(R^t)$ as the average gain across all $n_S$ synthetic samples from $f(R^t)$: $\mathtt{PG}_\mathbf{t} = \frac{\mathtt{PL}_\mathbf{t}(R^t) - \overline{\mathtt{PL}_\mathbf{t}}(S)}{2}$.

Table 6.1 summarizes the parameters used in the evaluation. $n_A$ and $k$ are parameters specific to the adversary that implements the risk of linkability described in 6.2.5. We set these parameters based on a series of experiments in which we

**Table 6.1:** Experimental setup parameters

| Dataset | $n$ | $m$ | $n_T$ | $n_R$ | $n_S$ | $n_A$ | k |
|---|---|---|---|---|---|---|---|
| `Adult` | 1,000 | 1,000 | 50 | 25 | 100 | 10,000 | 10 |
| `Texas` | 1,000 | 1,000 | 50 | 25 | 100 | 10,000 | 10 |
| `German` | 300 | 300 | 50 | 25 | 100 | 500 | 10 |

varied each parameter to measure its impact on privacy gain and chose each value to present a worst-case bound on privacy.

### 6.2.5 Membership inference on synthetic datasets

We build on existing MIAs on predictive models [158] to develop a *generic* black-box attack that is independent of the model architecture. We use this attack model to evaluate the risk that an attacker can link a target record to a sensitive dataset given synthetic data sampled from a generative model trained on this sensitive data.

**Adversarial model.** In contrast to previous attacks on GANs and VAEs [82, 85, 42], we assume an adversary that, instead of query access, only has access to a *single synthetic dataset output by the target model.* We define a privacy adversary $MIA_{\mathbf{t}}()$ that implements a generative model membership inference attack. Given a single synthetic dataset output by a generative model, this adversary produces a binary label that predicts whether a target record belongs to the model's training set or not:

**Definition 4** (Membership Inference Attack). *Let $t_s = \mathbb{1}[\mathbf{t} \in R^t]$ be a random variable that indicates $\mathbf{t} \in R^t$ where $R^t$ is the training set of a generative target model $f(R^t)$. A membership inference attack on target $\mathbf{t}$ given a synthetic dataset $S \sim \mathcal{D}^m_{f(R^t)}$ is a mapping $MIA_{\mathbf{t}}(S) \rightarrow \hat{t_s}$, $\hat{t_s} \in \{0,1\}$.*

Previous MIAs on predictive models range from "trivial" approaches, that simply use a correct or incorrect prediction to assign a membership label [110], to Bayes-optimal adversaries, that assume perfect knowledge of the underlying probability distributions [154]. We cast membership inference as a supervised learning problem [158]. Concretely, our membership inference adversary $MIA_{\mathbf{t}}()$ outputs a guess according to

$$MIA_{\mathbf{t}}(S) \triangleq \underset{t_s \in \{0,1\}}{\operatorname{argmax}} P[t_s|S]. \tag{6.5}$$

We estimate the posterior probability $P[t_s|S]$ using shadow training: we produce a set of generative shadow models trained on samples from a reference dataset $R^a$ of size $n_A$ with and without the target $\mathbf{t}$, and train an attack model on synthetic datasets output by the shadow models labeled as *in* and *out*. As in [158], we assume that the adversary has access to the training algorithm $\texttt{GM}()$, and to a *reference dataset* $R^a \sim \mathscr{D}_{\mathscr{R}}^{n_A}$ that comes from the same distribution as the target model's training data $R^t \sim \mathscr{D}_{\mathscr{R}}^{n}$ and may or may not overlap with $R^t$.

**Algorithm 4** MIATrain

```
Input: $\texttt{GM}(), \mathbf{t}, R, n, m, n_S, k$
Output: $MIA_{\mathbf{t}}()$
1:  for $i = 1, \cdots, k$ do
2:      $R_i \sim R^n$
3:      $f_i \sim \texttt{GM}(R_i)$
4:      for $j = 1, \cdots, n_S$ do
5:          $S_j \sim f_i$
6:          $l_j \leftarrow 0$
7:          $S_{\texttt{train}} \overset{+}{\leftarrow} S_j$
8:          $l_{\texttt{train}} \overset{+}{\leftarrow} 0$
9:
10:     $R'_i \leftarrow R_i \cup \mathbf{t}$
11:     $f'_i \sim \texttt{GM}(R'_i)$
12:     for $j = 1, \cdots, n_S$ do
13:         $S_j \sim f'_i$
14:         $l_j \leftarrow 1$
15:         $S_{\texttt{train}} \overset{+}{\leftarrow} S_j$
16:         $l_{\texttt{train}} \overset{+}{\leftarrow} 1$
17: $MIA_{\mathbf{t}}() \leftarrow \texttt{Classifier}(\texttt{S}_{\texttt{train}}, \texttt{l}_{\texttt{train}})$
```

We describe the training procedure for the adversary $MIA_{\mathbf{t}}()$ in Algorithm 4. Given a reference dataset $R$, the adversary repeats the following steps $k$ times. First, the adversary samples a training set $R_i$ of size $n$ from $R$ (line 2) and trains a generative model (line 3). The adversary creates a set of $n_S$ synthetic datasets of size $m$ sampled from the trained model and assigns them the label 0 that indicates that the target record $\mathbf{t}$ was not present in the training data (lines 4–7). The adversary repeats the same procedure on the same training set, this time including the target, $R'_i = R_i \cup \mathbf{t}$, and assigns the label 1 to the synthetic datasets to indicate that the target was present (lines 8-13). Finally, the adversary trains a classifier $MIA_{\mathbf{t}}$ which, given a synthetic dataset $S$, predicts the membership of the target record $t$ in the training set of the generative model $f$ that produced $S$ (line 14).

**Privacy gain with respect to linkage risk.** As in previous work [186], we assume an adversary that has a uniform prior over the target's membership in $R$ ($\mathrm{P}[t_s] = 0.5$). As the adversary's prior is equal in both cases regardless of whether the real or the synthetic data is published, the record privacy gain can be directly calculated as the difference between the adversary's chances of success given access to $R$ and to $S$ (see 6.3). If the adversary has access to $R$, her probability to correctly guess whether the target record is present in $R$ is 1 ($MIA_{\mathbf{t}}(R) = 1$). Therefore, the record privacy gain for a target $\mathbf{t}$ with respect to linkability ranges between $\mathtt{PG}_{\mathbf{t}} = 0$, when publishing $S$ leads to the same privacy loss as publishing $R$, and $\mathtt{PG}_{\mathbf{t}} = 0.5$, when the synthetic data perfectly protects the target from linkage. When publishing $S$ does not hide nor give new information, the adversary's success probability to correctly infer membership is given by her prior knowledge. In this case $\mathtt{PG}_{\mathbf{t}} = \frac{1-MIA_{\mathbf{t}}(S)}{2} = \frac{1-0.5}{2} = 0.25$.

**Algorithm 5** `MIAGain`

```
Input: $\mathtt{GM}(), R^t_{out}, \mathbf{t}, n, m, n_S, R^a, k$
Output: $\mathtt{PG}_{\mathbf{t}}$
 1: $f_{out} \sim \mathtt{GM}(R^t_{out})$
 2: for $i = 1, \cdots, n_S$ do
 3:     $S_i \sim f^m_{out}$
 4:     $S_{test} \xleftarrow{+} S_i$
 5: $R^t_{in} \leftarrow R^t_{out} \cup \mathbf{t}$
 6: $f_{in} \sim \mathtt{GM}(R^t_{in})$
 7: for $i = 1, \cdots, n_S$ do
 8:     $S_i \sim f^m_{in}$
 9:     $S_{test} \xleftarrow{+} S_i$
10: $MIA_{\mathbf{t}}() \leftarrow \mathtt{MIATrain}(\mathtt{GM}(), \mathbf{t}, R^a, n, m, n_S, k)$
11: $\mathtt{PG}_{\mathbf{t}} \leftarrow \frac{1-\overline{MIA_{\mathbf{t}}(S_{test})}}{2}$
```

We first create an evaluation set of $2 * n_S$ synthetic datasets $S_{test}$. Half of the datasets are sampled from a generative model fitted to the raw dataset *without the* *target* $R^t_{out}$ (lines 1–4); half from a generative model fitted to the raw dataset *with the target record in* $R^t_{in}$ (lines 5–9). We then train a privacy adversary $MIA_{\mathbf{t}}()$ on target **t** using the model training procedure $\mathtt{GM}()$ and a reference dataset $R^a$ (line 10). We use this adversary to infer the presence of $t$ in each of the synthetic datasets in $S_{test}$ and we calculate the privacy gain for record **t** as the difference between the adversary's success probability under $R^t$ ($\mathrm{P}\left[MIA_{\mathbf{t}}(R^t) = t_s\right] = 1$) and the average probability of success under $S$ (line 11). In line 11, we write $MIA_{\mathbf{t}}(S)$ as a shorthand for $\mathrm{P}\left[MIA_{\mathbf{t}}(S) = t_s\right]$ and $\overline{MIA_{\mathbf{t}}}(S_{test})$ for $\sum_{S_i \in S_{test}} {}^{MIA_{\mathbf{t}}(S_i)}/_{2*n_S}$.

**Practical considerations and implementation.** Mounting a successful black-box MIA on a generative model is much more challenging than on a predictive model [82]. MIAs on predictive models leverage patterns in the confidence values that differ between two classes, members and non-members. MIAs on generative models need to identify the influence that *a single target record* has on the high-dimensional model's output distribution $\mathcal{D}_{f(R)}$. The non-deterministic output sampling process and the fact that the adversary only has access to a single output example from the trained model increases the difficulty of this task.

In our attack setup, the adversary needs to be able to distinguish between two distributions, $\mathcal{D}_{f(R)}$ and $\mathcal{D}_{f(R \cup \mathbf{t})}$, given a single observation $S \sim \mathcal{D}^m_{f(X)}$. Detecting the fine differences between $\mathcal{D}_{f(R)}$ and $\mathcal{D}_{f(R \cup \mathbf{t})}$, however, is next to impossible in the high-dimensional data domain of $S$. To reduce the effect of high-dimensionality and sampling uncertainty, the adversary can apply *feature extraction* techniques before training the classifier in line 14 of Algorithm 4. This means that, in practice, the adversary will not use 6.5 but the following attack:

$$MIA_{\mathbf{t}}(\mathtt{F}, S) = \underset{t_s \in \{0,1\}}{\operatorname{argmax}} P[t_s | \mathtt{F}(S)], \tag{6.6}$$

where $\mathtt{F}(S)$ denotes a function $\mathtt{F} : S \rightarrow F$ that extracts a feature vector $F$ of size $n_F$ from a synthetic dataset $S$. Under this attack, the adversary effectively maps $S$ back to a *lower dimensional feature space* and aims to detect the target's influence on these features. Whether the attack using feature set $\mathtt{F}$ is successful depends on two factors: First, whether the target's presence has a detectable impact on any of the features in $\mathtt{F}$, and second, whether the synthetic dataset has preserved these features from the raw data and hence preserved the target's signal.

In our evaluation we implement the membership inference adversary $MIA_{\mathbf{t}}()$ as an instantiation of our framework's `PrivacyAttack` class. We leverage the object-oriented structure of the library to create multiple attack versions that share the same training procedure `MIATrain` but use different attack models and feature extraction techniques. We implement the adversary's feature sets as feature extraction objects `FeatureSet`. Each `FeatureSet` takes in a synthetic dataset $S$ of size $m \times k$ and outputs a vector of size $n_{\mathtt{F}} \times 1$.

As feature extractors, we implemented a naive feature set with simple summary statistics $\mathtt{F}_{\mathtt{Naive}}$, a histogram feature set that contains the marginal frequency counts of each data attribute $\mathtt{F}_{\mathtt{Hist}}$, and a correlations feature set that encodes pairwise attribute correlations $\mathtt{F}_{\mathtt{Corr}}$.

As attack models, we implemented a Logistic Regression, Random Forests and K-Nearest Neighbors classifier. All attack models yielded similar results with Random Forests performing best across datasets, generative models, and feature sets. In the next section, we focus on results obtained using the best performing attack model, Random Forests, and the attack feature set with the overall highest success rate $\mathtt{F}_{\mathtt{Corr}}$.

### 6.2.6 Experimental results

We evaluate the expected privacy gain with respect to the risk of membership inference under all generative models described in 6.2.2, using the procedure described in 6.2.4. At the beginning of each experiment, we sampled a fixed reference dataset $R^a$ from $R$ and used it across all test runs and targets. The parameter $n_A$ in 6.1 describes the size of the adversary's reference dataset and $k$ the number of shadow models trained by the adversary.

**Worst-case privacy gain.** We show in 6.21 *left* the estimated record privacy gain for a record crafted to represent a worse-case target, i.e., a target likely to be vulnerable to membership inference due to its unusual combination of attribute values (see

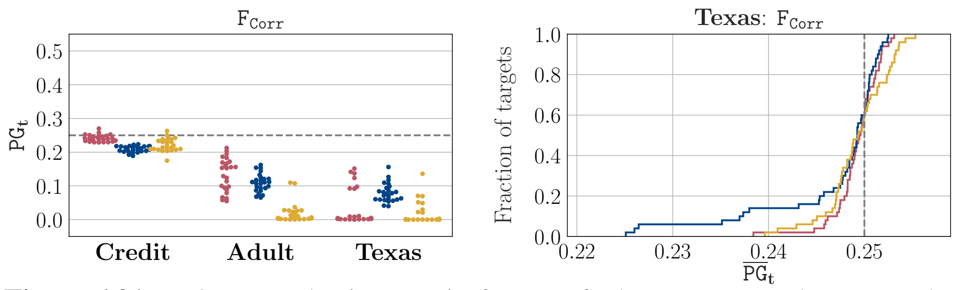


**Figure 6.21:** *Left*: Record privacy gain for a crafted target across datasets and generative models. *Right*: Cumulative distribution of record privacy gain across randomly sampled target records for the `Texas` dataset under three generative models. Generative models are colour-coded as ● `IndHist` ● `BayNet` ● `CTGAN`.

6.2.4). We plot the estimated record privacy gain of this target for $n_R = 25$ test runs. In each run, we sample a new target training set $R^t$ i.i.d from $\mathscr{R}$, and compute the estimated record privacy gain $\mathtt{PG_t}$ per target training set $R^t$ as specified in Algorithm 5.

For the two bigger datasets (`Adult` and `Texas`), the estimated record privacy gain for a worst-case target consistently lies *well below the expected gain* of $\mathtt{PG_t} = 0.25$ for all three generative models. For the `German` dataset, we observe that the estimated gain is much closer to the expected value of 0.25 which indicates that the adversary improves only slightly over her prior. This is likely due to the small size of this dataset. A generative model, when trained on such small amounts of data ($< 1000$ records per training set $R^t$) is unlikely to converge. Therefore, models (both targets and shadows) trained on small samples of the `German` dataset are not a robust representation of their input data. The attacker's strategy to use shadow models to mimic the target model's behavior is hence less likely to succeed than on the two bigger datasets.

More worryingly, there exist training sets $R^t$ for which the record privacy gain for the crafted target *is close to* 0. For those targets, publishing a synthetic dataset $S$ instead of the raw dataset $R^t$ would result in little to no additional privacy protection against linkage. In particular, for 68% of training sets from the `Texas` dataset the target's gain is smaller than 0.01 for synthetic data produced by `IndHist` and `CTGAN` models, respectively. `CTGAN`-trained models provide the smallest overall gain with an expected privacy gain of $\overline{\mathtt{PG_t}} = 0.02$. The low privacy gain in these cases indicates that the target's presence has a detectable influence on some of the features used in the attack ($\texttt{F}_{\texttt{Corr}}$) and that the generative model clearly preserves this signal in its output.

**Expected privacy gain across population.** The previous results focus on a worst-case scenario in which the target is a rare outlier, and thus highly vulnerable to membership inference. In 6.21 *right*, we compare the expected record privacy gain $\overline{\texttt{PG}_\mathbf{t}}$ (see 3) for 50 randomly chosen targets across 25 test runs on the `Texas` dataset (see 6.2.4).

As expected, the average gain for the randomly chosen targets is higher than the average gain of a worst-case target. However, we still find that for all three models the expected privacy gain is less than the random guess baseline ($\overline{\texttt{PG}_\mathbf{t}} = 0.25$) for $60\%$ of targets. These records are likely to be vulnerable to linkage independent of the distribution of records in the training set: Even when assessing the average gain across target training sets $R^t \sim \mathscr{D}^n_{\mathscr{R}}$ these records have a lower than expected gain. While in the worst-case `CTGAN`-produced synthetic data provides the least protection, `BayNet`-trained models are more likely to leave a randomly chosen target record vulnerable to linkage through an attack based on the correlations feature set. We obtained similar results on the `Adult` dataset.

**High variance, low predictability.** Other than the low minimum gain achieved for some target records, we observe a high variance in the record privacy gain across training sets for a fixed record (6.21 *left*) as well as a high variance in the average gain across target records for a fixed model training algorithm (6.21 *right*).

A high variance in the record privacy gain across training sets for a fixed target record demonstrates that it is the combination of training set $R^t$ and target record $\mathbf{t}$ that determines whether the target has a detectable influence on the data features preserved by the model. This means that it is not possible to predict privacy gain for a particular target record without fixing the entire training set.

A high variance in the average privacy gain across target records indicates that the model tends to amplify the signal of some records in the training set while it actually hides the presence of others, i.e., it gives disparate protection. Experiments with attack features other than the correlations set $\texttt{F}_{\texttt{Corr}}$ show that the variance in privacy gain provided by CTGAN is highest ranging from 0.04 to 0.48 across targets.

### 6.2.7 Differentially private model training does not increase privacy gain

In this section, we extend our evaluation to two model training algorithms, PrivBay and PATEGAN, that provide formal privacy guarantees and assess the expected privacy gain with respect to the adversary introduced in Subsection 6.2.5.

We used the same attacks and experimental setup as in previous sections to empirically evaluate whether models trained under DP increase the expected record privacy gain.

**Membership inference.** In 6.22, we show the estimated record privacy gain for a worst-case target under PrivBay and PATEGAN. As in previous experiments we observed that the German dataset does not provide enough data to obtain good model fits, we only present results for the Texas and Adult dataset. *Neither* differentially private training leads to an increase in average record privacy gain with respect to the risk of linkability. For instance, for 44% and 28% of training sets $R^t$ from the Adult dataset the target's estimated gain is close to 0 ($\mathtt{PG_t} < 0.01$) for synthetic data drawn from a PrivBay and PATEGAN model with $\varepsilon = 0.01$, respectively.

This implies that synthetic data drawn from either privacy-preserving model carries through enough information about the target's presence for an attacker to successfully identify the preserved features and infer the target's secret. This is especially surprising in the case of PrivBay under an attack focusing on the correlations feature set. The DP guarantee of the PrivBay procedure should ensure protection against linkage attacks on these features. Even more surprisingly, the expected gain of a worst-case target under this model even *decreases* as the privacy parameter decreases for both datasets (from $\overline{\mathtt{PG_t}} = 0.11$ for $\varepsilon = 1$ to $\overline{\mathtt{PG_t}} = 0.04$ for $\varepsilon = 0.01$ in the Adult dataset, statistically significant with $p < 0.0001$).

**No (free) gain.** We find that privacy guarantees for the lower-dimensional representation of the input data learned by differentially private models *do not necessarily*

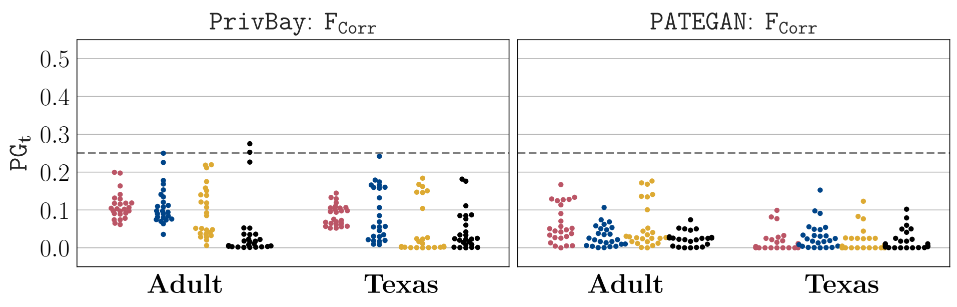


**Figure 6.22:** Record privacy gain for a crafted target under PrivBay- and PATEGAN-trained models for four different privacy parameter settings. The $\varepsilon$ values are colour-coded as ● $\varepsilon \to \infty$ ● $\varepsilon = 1$ ● $\varepsilon = 0.1$ ● $\varepsilon = 0.01$
$\varepsilon \to \infty$ indicates no noise addition during training. Data shown a Random Forests attack using the $F_{Corr}$ feature set.

*extend to all features of the high-dimensional synthetic data* sampled from these models. Thus, these models do not provide robust protection against linkage attacks on their outputs.

## 6.2.8 Key takeaways

Our evaluation framework enabled us to study the privacy gain provided by a wide variety of generative models, from simple statistical models to differentially private deep networks. Our results challenge the claim that synthetic data is a silver-bullet solution to the privacy problems of microdata publishing.

First, high-dimensional synthetic datasets often carry more information about their training data than what is captured by the generative model. Even synthetic data drawn from simple models often preserves a large number of raw data features. Adversaries are not constrained in their choice of feature set and therefore can use *any of the large number of data dimensions replicated in a model's output* to conduct their attack. This reminds of past failures of data anonymization: "high-dimensional data is inherently vulnerable to de-anonymization" [121]. As it is not possible to predict neither what information synthetic carries nor what features adversaries target, it impossible to predict the privacy gain of synthetic data publishing.

Second, our results show that DP provides little protection against ML-based inference attacks on high-dimensional data releases. However, this is more likely due to the model's implementation rather than to the soundness of the differential privacy definition, as they normally require additional metadata to operate (as, for example, the ranges of continuous attributes). In fact, this aids the adversary (and thus lowers the DP privacy guarantee) in practice, even though in theory the models fulfill the DP definitions.

## 6.3 Measuring Utility and Privacy of Genomic Data

As previously mentioned, data sharing in genomics is crucial to enable progress in Precision Medicine [71]. Unsurprisingly, however, this is inherently at odds with the need to protect individuals' privacy. Genomic data contains sensitive information related to heritage, predisposition to diseases, phenotype traits, etc., which makes it hard to anonymize [79]. Hiding "sensitive" portions of the genome is not effective either, as sensitive information can still be inferred via high-order correlation models [156]. For a thorough review of privacy threats in genomics, please see [25, 178, 127].

As a result, genomics researchers have begun to investigate the possibility of releasing *synthetic* datasets, rather than real/anonymized data [155]. This follows a general trend in healthcare; for instance, the National Health Service (NHS) in England has recently concluded a project focused on releasing synthetic Emergency Room ("A&E") records [128]. The intuition is to use generative models to learn to generate samples with the same characteristics—more precisely, with the same distribution—of the real data. That is, rather than releasing data of actual individuals, entities share artificially generated data in such a way that the statistical properties of the original data are preserved, but minimizing the risk of malicious inference of sensitive information [61].

**Generative Models and Genomics.** Specific to genomics, previous work has experimented with both statistical and machine learning generative models. Samani et al. [156] propose an inference model based on the recombination rate, which can also be used to generate new synthetic samples. Yelmen et al. [185] use Generative Adversarial Models (GANs) and Restricted Boltzmann Machines (RMBs) to mimic the distribution of real genomes and capture population structures. Finally, Killoran et al. [103] use ad-hoc training techniques for GANs and architectures for computer vision tasks.

**Motivation.** Prior work on generating synthetic data in genomics has thus far only scratched the surface with respect to assessing their utility, and more specifically their statistical fidelity. Moreover, we do not really know whether these approaches actually provide any meaningful privacy guarantees. To address this gap, we introduce a novel evaluation framework and perform a series of measurements geared to assess both the utility and the privacy of five state-of-the-art models used to generate human genomic synthetic data.

**Experimental Evaluation.** Our analysis unfolds along two main axes:

- *Utility.* We focus on a number of very common computational tasks on genomic data. We measure how well generative models preserve summary statistics (e.g., allele frequencies, population statistics), or linkage disequilibrium. We also assess how close are the distributions of synthetic data vs. real data for principal component analysis.
- *Privacy.* We mount *membership inference attacks* [89], having an attacker infer whether a target record was part of the real data used to train the model producing the synthetic dataset. More precisely, we quantify the privacy gained (recall from Section 6.2.1,) vis-à-vis this attack, from releasing synthetic data vs. releasing the real dataset. In the process, we also introduce a novel attack where the adversary only has *partial* information for a target individual.

**Main Findings.** Overall, our evaluation shows that there is no single approach for generating genomic synthetic data that performs well across the board, both in terms of utility and privacy. Among other things, we find that:

- A high-order correlation model specifically build for genomic data (Recomb) has the best utility metrics for small datasets but does so at the cost of privacy, even against weaker adversaries who only have partial information available.

- The RBM model has a better performance with increasing dataset sizes, both in terms of utility and privacy, as targets get stronger privacy protection when synthetic data is generated using a larger training set.

- Releasing synthetic datasets does not provide robust protection against membership inference attacks. We find cases where releasing the synthetic dataset sometimes offers better protection against membership inference attacks. However, because of the randomness introduced by the generative models, one cannot meaningfully predict a target's susceptibility to privacy attacks without fixing the training set and quantifying the respective privacy loss/gain for all targets in the set.

### 6.3.1 Datasets

In our evaluation, we use data from two projects: HapMap [122] and the 1000 Genome Project [12]. More specifically, we use 1,000 SNPs from chromosome 13 from the following three datasets:

1. *CEU Population (HapMap).* Samples from 117 Utah residents with Northern and Western European ancestry, released in phase 2 of the HapMap project.

2. *CHB Population (HapMap).* Samples from 120 Han Chinese individuals from Beijing, China.

3. *1,000 Genomes.* Samples from 2,504 individuals from 26 different populations released from phase 3 of the 1000 Genomes project.

### 6.3.2 Synthetic Data Approaches in Genomics

The ability to effectively train statistical models [20] from genomic data is very important in many applications. As many models suffer from the "curse of dimensionality" [18], i.e., models do not usually perform well on small datasets with high dimensionality, statistical and generative models have been proposed not only to mitigate possible privacy concerns of sharing genomic data but also to help "inflate" the size of the datasets for more meaningful analysis.

In this section, we provide an overview of the state-of-the-art models for generating synthetic genomic data. In particular, we discuss the Recombination model proposed by Samani et al. [156], the RBM and GAN models proposed by Yelmen et al. [185], and the WGAN model from Killoran et al. [103]. We also introduce and consider two other "hybrid" models.

**Recombination Model (Recomb).** Samani et al. [156] propose the use of a *recombination model* as an inference method for quantifying individuals' genomic privacy. This is a statistical model, based on high-order SNV correlation that relates linkage disequilibrium patterns to the underlying recombination rate. Given a set of sampled haplotypes, the model relates their distribution to the underlying recombination rate.

Also, [156] shows how to use this method to generate synthetic samples in order to perform Principal Component Analysis (PCA). The recombination model yields a distribution closer to the real data than models using only linkage disequilibrium and allele frequencies. In order to obtain the underlying recombination rate, the model uses a "genetic map," which includes the recombination rate. This is provided with the dataset for the HapMap datasets, but not for the 1000 genomes data. For the latter, we use the scripts from [5].

**Restricted Boltzmann Machines (RBMs).** In the context of genomic data, Yelmen et al. [185] use RBMs to generate synthetic genomic data. In our evaluation, we follow the same RBM settings as [185]. More specifically, we use a ReLu activation function, with the visible layer having the same size as the input we considered (1,000 features) and with the number of hidden nodes set to 100. The learning rate is set to 0.01, the batch size to 32, and we iterate over 2,000 epochs.

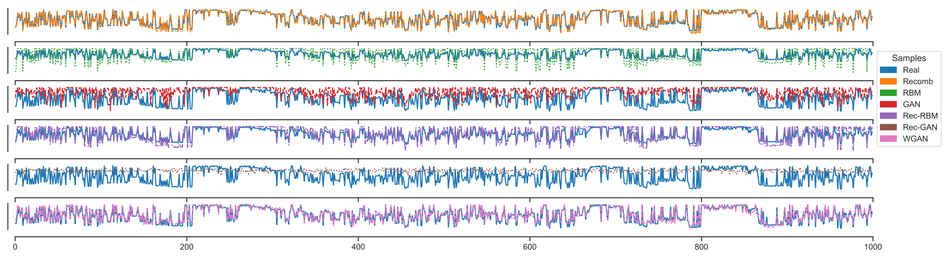
**Generative Adversarial Networks (GANs).** Again, we use the GAN approach proposed by Yelmen et al. [185], mirroring their experimental settings. More precisely, the generator model consists of an input layer with latent dimension set to 600, and two hidden layers, of sizes 512 and 1,024 respectively. The discriminator consists of an input layer with size equal to the number of SNPs evaluated (1,000), and two hidden layers of sizes 512 and 256 respectively, as well as an output layer of size 1. The output layer for the generator uses tanh as an activation function and the output layer for the discriminator uses the sigmoid activation function. For both the generator and discriminator, we compile them using the Adam optimization and binary cross-entropy as the loss function.

**Recombination RBM (Rec-RBM).** To overcome issues caused by low numbers of training samples, we propose a hybrid approach between the Recomb and the RBM models. In other words, we use the former to generate extra samples, which we then use, together with the real data samples, to train the RBM model with the same parameters as before. We do so to explore whether having more data points available to train the model improves the utility of the synthetic data.

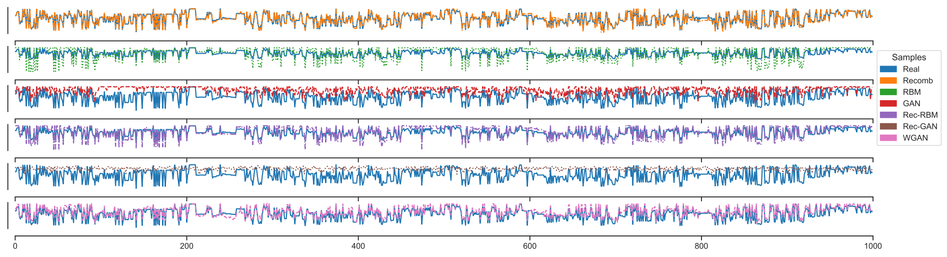
**Recombination GAN (Rec-GAN).** Similar to Rec-RBM, we use the Recomb model to generate extra training samples for the GAN model, using the same parameters as before. Again, we want to study whether having a larger dataset available for training the GAN improves the overall utility of the synthetic data output by it.

**Wasserstein GAN (WGAN).** Killoran et al. [103] propose an alternative GAN model by treating DNA sequences as a hybrid between natural language and computer vision data. The sequences are one-hot encoded, the GAN is based on a WGAN architecture trained with a gradient penalty [78], and both the generator and discriminator use convolutional neural networks [107] and a residual architecture [84], which includes skip connections that jump over some layers. The authors also propose a joint method combining the GAN model with an activation maximization design [160, 189, 119] in order to tune the sequences to have desired properties. We do not, however, include the joint model in our evaluation, as we focus on a range of statistics as opposed to a single desired property.

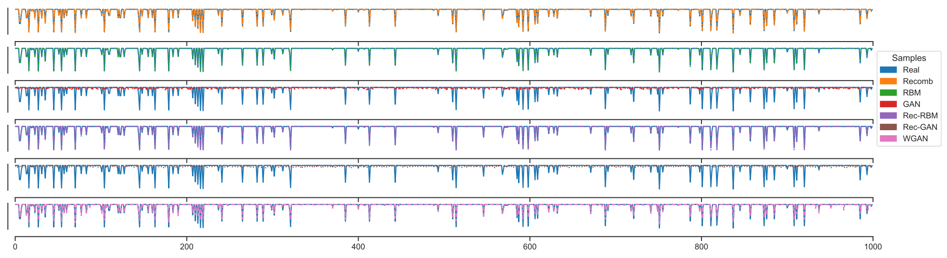
In our evaluation, we use the WGAN model with the default parameters from the implementation in [4]. The generator consists of an input layer with dimension of the latent space set to 100, followed by a hidden layer with size 100 times the length of the sequence (1,000), which is then reshaped to (length of the sequence, 100), followed by 5 resblocks. Finally, there is a 1-D convolutional layer followed by the output layer, which uses softmax. The discriminator has a very similar ar-



**(a)** CEU



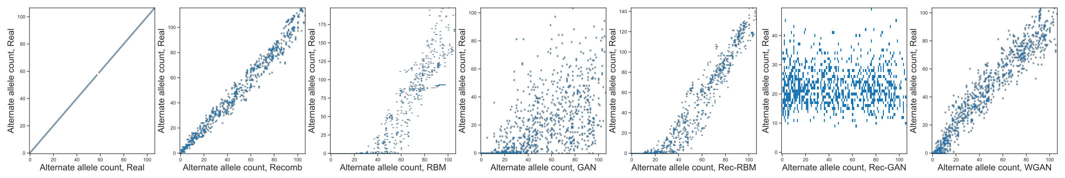
**(b)** CHB



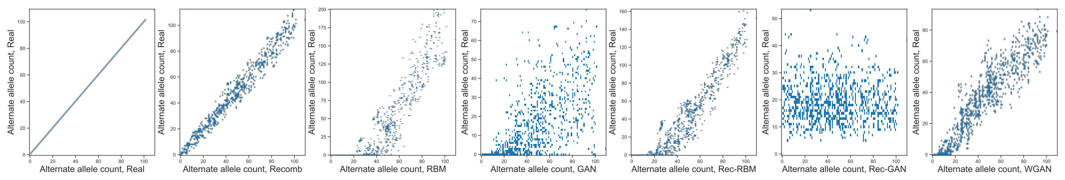
**(c)** 1000 Genomes

**Figure 6.23:** Major allele frequencies for synthetic data generated by the models, plotted against the real data, for the CEU population, the CHB population, and the 1000 Genomes dataset.

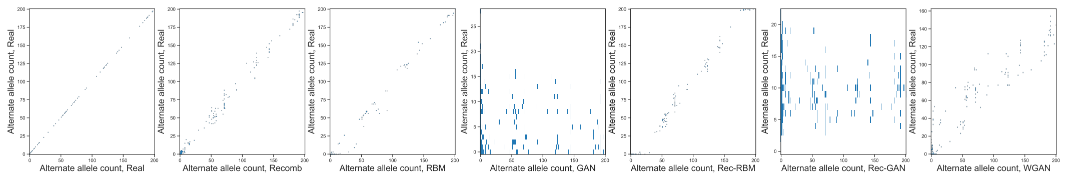
chitecture but in a different order – i.e., it starts with the input layer to which the one-hot sequences are fed, that is followed by the 1-D convolutional layer, then the 5 resblocks, followed by the reshape layer and the output layer of size 1. We perform 5 discriminator updates for every generator update. Both the generator and discriminator use Adam optimization and their learning rates are set to 0.0001, while the loss as mentioned is adjusted by a gradient penalty. We use a batch size of 64. In our experiments, the WGAN model converges after about 80 iterations; so, as opposed to the 100,000 proposed by the authors, we train the model for 100 iterations in our evaluation.



**(a)** CEU

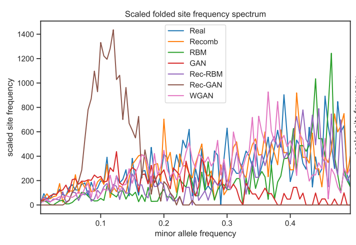


**(b)** CHB

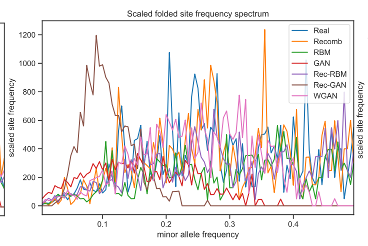


**(c)** 1000 Genomes

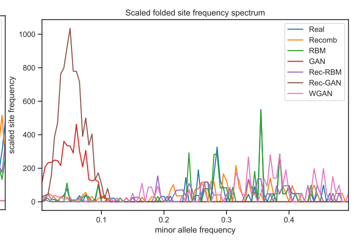
**Figure 6.24:** Alternate allele correlation for the CEU population, the CHB population, and the 1000 Genomes dataset.



**(a)** CEU



**(b)** CHB



**(c)** 1000 Genomes

**Figure 6.25:** Frequency spectrum analysis for the CEU population, the CHB population, and the 1000 Genomes dataset.

### 6.3.3 Utility Evaluation

We now perform a comprehensive utility evaluation of the synthetic data generated by the models presented in Section 6.3.2. We look at common summary statistics used in genome-wide association studies, aiming to assess the accuracy loss due to the use of synthetic datasets. More specifically, we analyze how well data generated by the generative models preserves allele frequencies, population statistics, and linkage disequilibrium, and how close the distribution of the synthetic data is to the real data for principal component analysis.

#### 6.3.3.1 Allele Statistics

**Major Allele Frequency (MAF).** In population genetics, the *major allele frequency* (MAF) is routinely used to provide helpful information to differentiate between common and rare variants in the population, as it quantifies the frequency at which the most common allele occurs in a given population. We start our utility analysis by comparing MAFs in the synthetic data vs. the real data.

In Fig. 6.23, we plot the MAF at each position for the real datasets and for the synthetic samples, over the CEU and CHB populations, and the 1000 Genomes dataset. For CEU/CHB (Fig. 6.23a–6.23b), we observe that Recomb and WGAN replicate best the allele frequencies in the real data. On the other hand, GAN and Rec-GAN fail to do so, and in fact, the generated samples seem random. The RBM model, even though not as close to the real frequencies as Recomb, performs better than the GAN and Rec-GAN models. In fact, RBM further improves when combined with Recomb (see Rec-RBM).

For 1000 Genomes (Fig. 6.23c), Recomb's MAF distribution is also similar to the real data's. However, RBM and Rec-RBM both display MAFs close to the real data, whereas, even with more training samples available, the GAN and Rec-GAN models still seem to produce random results. Moreover, WGAN does not match the MAF distribution for this population as closely. Overall, the difference in the MAF distributions across datasets is likely to be due to fewer samples available for the HapMap populations compared to the 1000 Genomes.

**Alternate Allele Correlation (AAC).** To evaluate whether the real and synthetic data are *genetically* different, in Fig. 6.24, we plot the *alternate allele correlation* (AAC). The more similar two populations are, the closer the SNPs should be to the diagonal, as in the leftmost plots, where we have the real data against itself. The strongest AAC is with the synthetic data generated by Recomb. On the opposite side of the spectrum, the synthetic data generated by GAN and Rec-GAN have weak correlations. For the CEU and CHB populations, we find Rec-RBM to yield stronger AACs than simple RBM and the WGAN. For the 1000 genomes dataset (Fig. 6.24c), there is a strong correlation between the alternate alleles for the real data and Recomb, RBM, Rec-RBM, and WGAN.

**Site Frequency Spectrum (SFS).** Another summary statistic that captures essential information about the underlying distribution of the allele frequencies of a given set of SNPs in a population or sample is the SFS [66, 65]. Basically, it provides a histogram whose size depends on the number of sequenced individuals. In Fig. 6.25, we plot the scaled folded SFS, which is the distribution of counts of minor alleles in a sample calculated over all segregating sites. We scale this value so that a constant value is expected across the spectrum for neutral variation and constant population size, which yields the best visual comparisons. If the distribution of allele frequencies for the synthetic samples matches that for the real data, we would expect to see the two spectra aligned.

With the HapMap populations (Fig. 6.25a– 6.25b), Rec-GAN suggests an excess of rare variants for a minor allele frequency around 0.1. Whereas GAN seems to generate data closer to a neutral expectation, i.e., the synthetic dataset describes a more stable population. Similarly, for the 1000 Genomes (Fig. 6.25c), Rec-GAN has an excess of rare variants for a minor allele frequency less than 0.1, and this is also displayed, at a lower scale, by the GAN-generated data.

We also compute the Kolmogorov-Smirnov (KS) two-sample test [86] for the goodness of fit on the SFS for each dataset vs. the synthetic data (see Table 6.2). The test compares the agreement between the cumulative distributions of two independent samples. For every two-samples test, the 95% critical value is approximately 0.195 (as we have 100 samples in each dataset), so we can reject the null hypothesis (that there is no difference between the distributions) for all synthetic data above this value. For both CEU and CHB, we cannot reject the null hypothesis only for the samples generated by the Recomb and the WGAN models. For the 1000 Genomes dataset, we reject the null hypothesis for synthetic data generated by the GAN and Rec-GAN.

#### 6.3.3.2 Population Statistics

Next, we look at population statistics to determine how close to the real dataset is the synthetic data. In particular, we look at the percentage of heterozygous variants for both real and synthetic samples, at the fixation index, and at the Euclidean Genetic Distance.

**Heterozygosity.** The condition of having two different alleles at a locus is denoted as heterozygosity. The percentage of heterozygous variants is commonly used in population studies, as a low percentage of heterozygous variants implies less diversity in the population. In Fig. 6.26a–6.26b, we plot the percentage of heterozygous variants in each sample for the CEU/CHB populations, comparing the real statistics (blue/leftmost bars) vs. those computed on the synthetic data. In both cases, Recomb and WGAN yield similar percentages to the real dataset. Whereas, with GAN and RBM, the percentage decreases, suggesting that both models produce more homozygous variants. Moreover, even though for the major allele frequencies Rec-RBM produces variants with statistics closer to the real data, the percentage of heterozygous variants turns out to be the lowest for both populations. By contrast, Rec-GAN produces a higher percentage of heterozygous variants than GAN, even though the major allele frequencies are not aligned with the original samples.

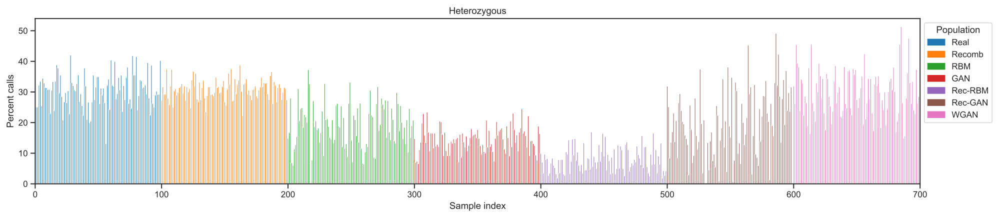
With the 1000 Genomes (Fig. 6.26c), the percentage of heterozygous samples in the real data is lower across all samples. Once again, and in line with previous results, GAN and the Rec-GAN significantly deviate from the percentages of heterozygous samples found in the real data.

We also run a Kolmogorov-Smirnov (KS) two-sample test [86] for the goodness of fit on the percentage of heterozygous samples for each dataset vs. the synthetic data (see Table 6.2). For both Recomb and WGAN, we do not reject the null

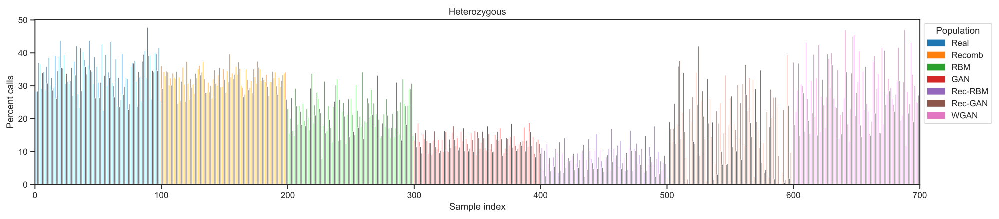
| | **SFS** | | | | | | **% Heterozygous Samples** | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **CEU** | | **CHB** | | **1000 Geno** | | **CEU** | | **CHB** | | **1000 Geno** | |
| **Models** | *D* | p-value | *D* | p-value | *D* | p-value | *D* | p-value | *D* | p-value | *D* | p-value |
| **Recomb** | 0.07 | 0.89 | 0.07 | 0.89 | 0.18 | <0.1 | 0.19 | 0.47 | 0.29 | <0.001 | 0.32 | <0.001 |
| **RBM** | 0.34 | <0.001 | 0.28 | <0.001 | 0.12 | 0.44 | 0.64 | <0.001 | 0.70 | <0.001 | 0.13 | 0.34 |
| **GAN** | 0.40 | <0.001 | 0.41 | <0.001 | 0.23 | <0.01 | 0.90 | <0.001 | 1.00 | <0.001 | 0.57 | <0.001 |
| **Rec-RBM** | 0.26 | <0.01 | 0.23 | <0.01 | 0.06 | 0.99 | 0.99 | <0.001 | 1.00 | <0.001 | 0.40 | <0.001 |
| **Rec-GAN** | 0.64 | <0.001 | 0.58 | <0.001 | 0.25 | <0.01 | 0.55 | <0.001 | 0.68 | <0.001 | 0.52 | <0.001 |
| **WGAN** | 0.09 | 0.79 | 0.18 | <0.1 | 0.19 | <0.1 | 0.17 | <0.1 | 0.39 | <0.001 | 0.45 | <0.001 |

**Table 6.2:** Two-sample (real vs. synthetic data) Kolmogorov-Smirnov test performed on the SFS and the percentage of heterozygous samples.

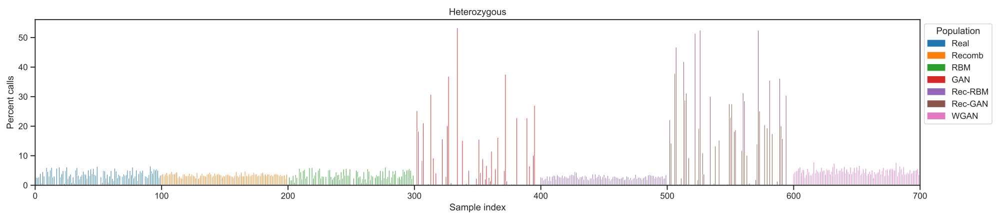
hypothesis for the CEU dataset, but we do for the CHB dataset. In fact, for all of



(**a**) CEU Population



(**b**) CHB Population



(**c**) 1000 Genomes Population

**Figure 6.26:** Percentage of heterozygous variants in each sample in the dataset for CEU, CHB populations, and the 1000 Genomes dataset.

the models trained on the CHB dataset, we reject the null hypothesis. For the 1000 Genomes dataset, we do not reject the null hypothesis only for RBM.

**Fixation Index (**$F_{ST}$**).** Another way to assess how *different* are groups of populations from each other is to use the fixation index [88]. This provides a comparison of differences in allele frequency, with values ranging from 0 (not different) to 1 (completely different/no alleles in common). In Fig. 6.27, we compare the fixation index values for the real data against the synthetic samples. For illustration purposes, we also include $F_{ST}$ of the real data against itself, which obviously yields $F_{ST} = 0$.

Recomb is once again the closest to the real data, which confirms the alignment from Fig. 6.23 of the allele frequencies of the synthetic recombination data with the real data. The $F_{ST}$ value for the synthetic data produced by RBM is, for both CEU and CHB populations, less than 0.10, however, the hybrid Rec-RBM model further reduces this value to less than 0.04, and so does WGAN. For both populations, data generated by GAN and Rec-GAN has the highest $F_{ST}$, although, for the CHB

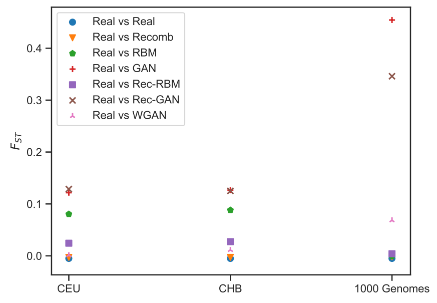


**Figure 6.27:** Fixation index values for the CEU and CHB populations, and the 1000 Genomes dataset.

population, the latter increases it and for the CEU population reduces it. Finally, for the 1000 genomes, Recomb, RBM, and Rec-RBM all have $F_{ST}$ close to the real data. While still having a low $F_{ST}$, WGAN has a slightly higher value. Whereas, with GAN and Rec-GAN, $F_{ST}$ significantly deviates from the real data, even with the increased number of samples of this dataset.

**Euclidean Genetic Distance (EGD).** Since the fixation index does not easily allow for pairwise comparisons among populations, in Fig. 6.28, we plot the Euclidean Genetic Distance (EGD) between the samples in each dataset. EGD is routinely used as a measure of divergence between populations, and shows the number of differences, or mutations, between two populations; a distance of 0 means that is no difference in the results, i.e., there is an exact match. From Fig. 6.28a–6.28b, where the EGD on the diagonal is 0, we observe that, for both CEU and CHB populations, the synthetic samples generated by GAN are closer to each other than by the other models. Rec-GAN generates samples with EGD close to 0, suggesting that there are very few differences between them, as well as samples with a distance of around 30. As for the other population statistics, Recomb generates samples that match the differences observed in the real data the closest, for both populations. For RBM, the samples generated have fewer differences than the real data. Perhaps more interestingly, Rec-RBM yields samples with a higher divergence than the real data; this can be a consequence of the low percentage of heterozygous samples found in the synthetic samples generated by this model (recall Fig. 6.26). The samples from WGAN match some of the differences observed in the real data, but the model also yields a few samples with a higher divergence.

Finally, for the 1000 Genomes (Fig. 6.28c), we find that all samples in the real data have closer EGDs between each other. In fact, the samples generated by RBM yield a similar pattern in the EGD distances. Although Recomb, Rec-RBM, and WGAN do too, they exhibit a lower distance, on average, between samples. As for CEU/CHB populations, GAN and Rec-GAN models overall fail to capture the differences between samples.

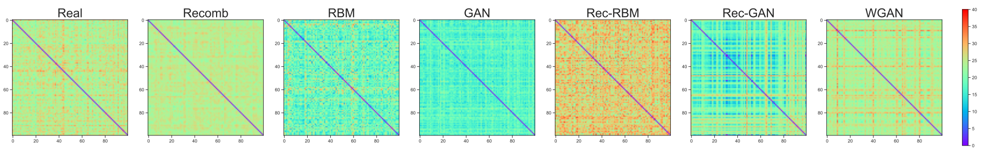
#### 6.3.3.3 Linkage Disequilibrium (LD) Analysis

Linkage disequilibrium (LD) captures the non-random association of alleles at two or more positions in a general population – i.e., those alleles do not occur randomly with respect to each other. In Genome-Wide Association Studies, LD allows researchers to optimize genetic studies, e.g., by preventing genotyping SNPs that provide redundant information [37]. In Fig. 6.29, we plot the $r^2$ value for LD based on the Rogers-Huff method [152]. This ranges from 0 (there is no LD between the 2 SNPs) to 1 (the SNPs are in complete LD, i.e., the two SNPs have not been separated by recombination and have the same allele frequencies).

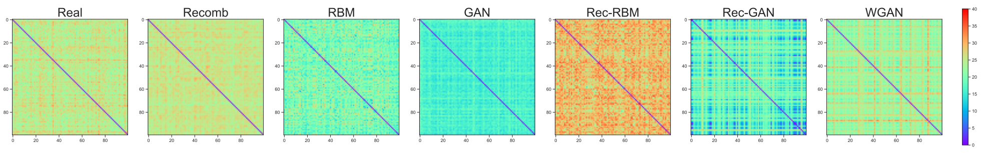
For CEU and CHB populations, RBM generates samples that display a stronger LD than the real data. With more training samples, Rec-RBM yields a weaker LD, but still stronger than the real data for both models. On the other side of the spectrum, for Rec-GAN, the LD for the synthetic data is the weakest. For the 1000 Genomes, we find a stronger LD between the real samples than with the other two datasets. RBM generates samples that are almost indistinguishable from the real data in terms of LD. The LD in the synthetic datasets generated by Recomb, Rec-RBM, and WGAN have lower correlations than RBM, with GAN and Rec-GAN both failing to preserve the LD.

#### 6.3.3.4 Principal Component Analysis (PCA)

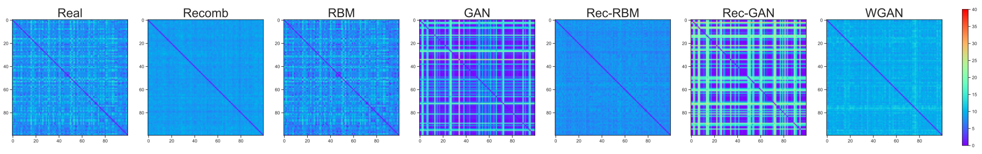
Finally, we further study the difference between synthetic and real data by performing a principal component analysis on the corresponding samples. We extract the first two principal components and project the real and synthetic datasets on these two components to show how the synthetic samples are distributed compared to the



**(a)** CEU



**(b)** CHB



**(c)** 1000 Genomes

**Figure 6.28:** Pairwise Euclidean Genetic Distance (EGD) between individuals.

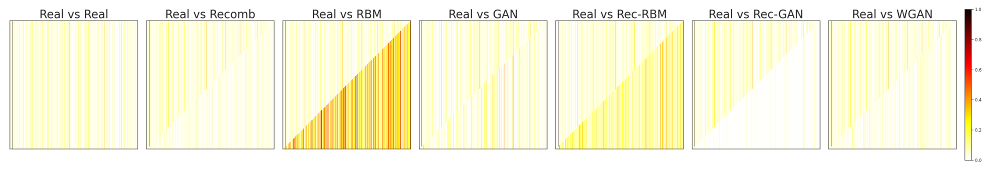
real data.

Fig. 6.30 presents this 2D visualization. Recomb has a close distribution to the real data for both HapMap populations, which, according to [156], is because the genetic recombination model considers all the correlations between SNPs and builds a higher-order model. For the 1000 Genomes, as for the other utility metrics studied in Section 6.3.3, the GAN and Rec-GAN models perform quite poorly, generating samples with a different distribution than the real samples. Therefore, in Figure 6.30d, we exclude them in order to take a closer look at the non-GAN-based models. Here, we can better observe that the RBM-generated samples have the closest distribution to the real data. In contrast to the HapMap populations, the samples from Recomb are all centered around 0 and fail to simulate the distribution given by the real data, and similar results are in the case of samples generated by Rec-RBM.

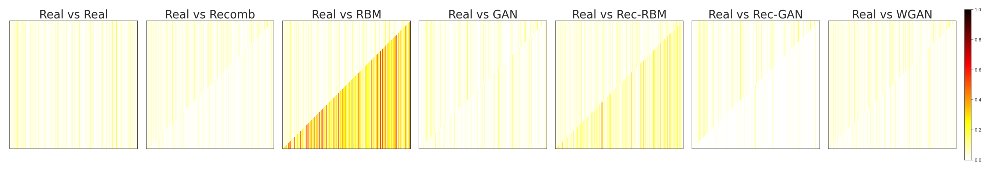
#### 6.3.3.5 Take-Aways

Our utility evaluation shows that there are only a handful of cases where generative models produce synthetic genomic data with high utility on popular tasks.

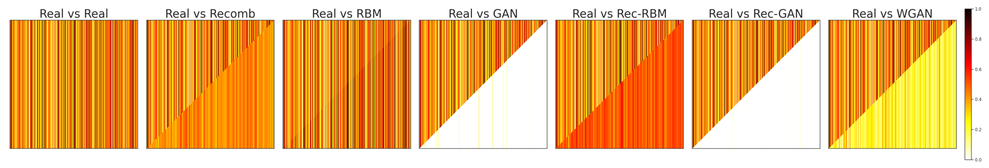
The Recomb model, which is based on high-order SNV correlations, generates synthetic data preserving most statistical properties displayed by the real data, even when few samples are available. We get better utility when the genetic map is



**(a)** CEU



**(b)** CHB



**(c)** 1000 Genomes

**Figure 6.29:** Pairwise Linkage Disequilibrium for Real vs. Synthetic Samples.

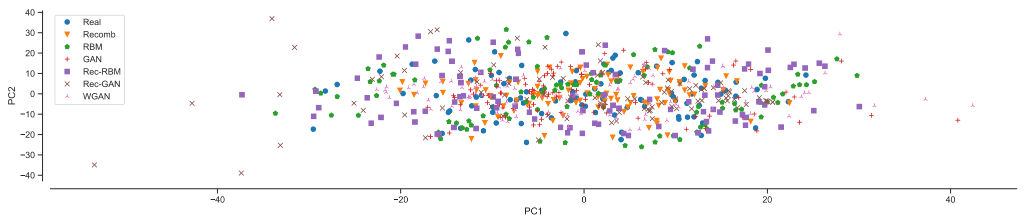
included with the data rather than generated from the existing data. With RBM, more training samples improve the quality of the synthetic data, as evidenced by the difference between the HapMap populations and the 1000 Genomes dataset.

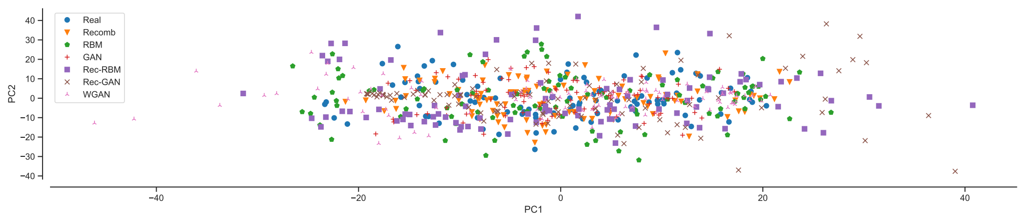
We also find that when few samples are available for training, the hybrid Rec-RBM model approach helps improve the quality of samples compared to just RBM. This is clear from the utility of the synthetic data on the two smaller HapMap datasets. For the 1000 genomes, it is not surprising that Rec-RBM's performance is worse than RBM since Recomb does not generate as "useful" samples as for the other two datasets. Finally, the GAN and the Rec-GAN models generate samples with the lowest utility, regardless of the number of samples available for training. However, the data generated by WGAN preserves most statistical properties of the real data.

Overall, our analysis exposes the limitations, in terms of statistical utility, of using generative machine learning models to produce synthetic genomic data.

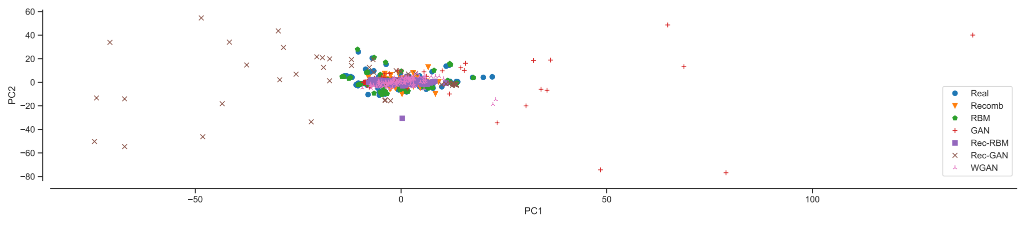
### 6.3.4 Privacy Evaluation

Next, we quantify the privacy provided by synthetic data by evaluating its vulnerability to Membership Inference Attacks (MIAs). More precisely, we compute the

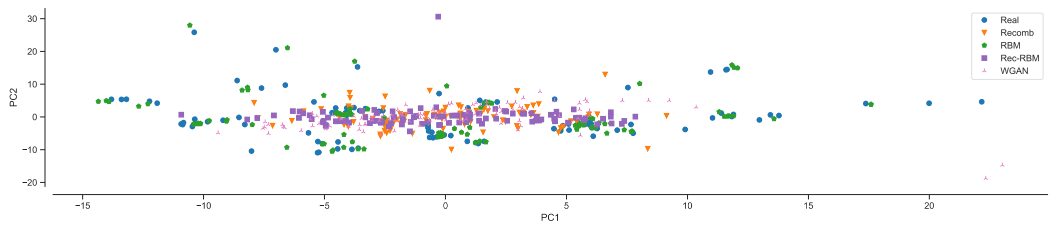




**(b)** CHB Population



**(c)** 1000 Genomes



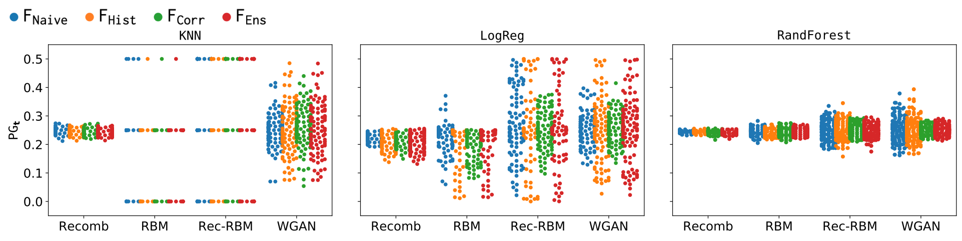
**(d)** 1000 Genomes, excluding the GAN and Rec-GAN models

**Figure 6.30:** 2D Principal Component Analysis (PCA) visualization of the real and synthetic sequences.

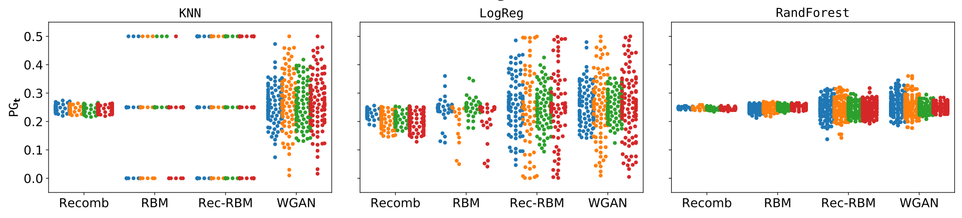
Privacy Gain PG (see Section 6.2.5) obtained by releasing a synthetic dataset instead of the real data. Recall that, if the synthetic data does not hide nor give any additional information to an MIA attacker, $PG_t$, for a target record $t$, should have a value of around 0.25.

We present experiments for both a "standard" MIA and a novel attack, which we denote as MIA with *partial information.* The latter essentially assumes that the adversary only has access to partial data from the target sequence. We exclude GAN and Rec-GAN from the evaluation since they yield poor utility performance, so there is not really any point in evaluating their privacy.

Throughout our evaluation, we randomly choose 10 targets from each dataset



**(a)** CEU Population



**(b)** CHB Population

**Figure 6.32:** Privacy Gain (PG) of different models over the two HapMap populations.

across 10 test runs. In each run, we fix the target and sample a new training cohort. We train the attack classifier using 5 shadow models, using 100 synthetic training sets for each of them. We then evaluate the privacy gain on 100 synthetic datasets, with a split of 50 sets generated from a training set including the target, and 50 sets generated without. Finally, we report the PG for each test and each target as the average PG across all synthetic datasets tested.

### 6.3.4.1 Privacy Gain Under Membership Inference Attack

We use three adversarial classifiers: K-Nearest Neighbor (KNN), Logistic Regression (LogReg), and Random Forest (RandForest). We use four feature sets, as described in Section 6.2.5: Naive (●$F_{Naive}$), Histogram (●$F_{Hist}$ ), Correlations (●$F_{Corr}$), and an Ensemble feature set (●$F_{Ens}$).

#### HapMap Populations

In Fig. 6.32, we report the PG value for targets randomly chosen from the two HapMap populations.

**KNN.** For CEU, using KNN (Fig. 6.32a, left), we find that over 74% of the targets in the synthetic dataset generated by Recomb have a PG lower than the random baseline (0.25) for the Ensemble, Correlations, and Histogram feature sets. With RBM, there are between 84% and 88% of the targets, depending on the feature set, that have a PG of 0.25; in other words, for these targets, the probability of the adversary inferring their presence in the training set is the same as random guessing. However, between 10% and 15% of the targets have no PG at all, whereas, there are between 1% and 4% of the targets, depending on the feature set, for which the synthetic data perfectly protects the target from MIA (PG=0.5). With Rec-RBM, at least 59% of the targets have a PG of 0.25 under the four feature sets. With WGAN, at least 48% of the targets have a PG of 0.25, depending on the feature set.

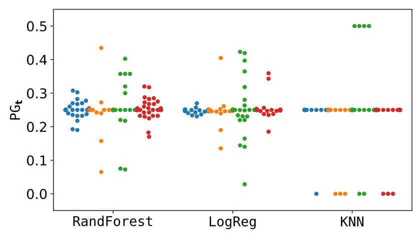
For the CHB population (Fig. 6.32b, left), we find that over 60% of the targets generated by the Recomb, with all features, have PG below the random guess baseline. With RBM, between 89% and 97% of targets have PG of exactly 0.25 across all feature sets, i.e. the synthetic dataset generated by the RBM for these targets does not hide nor give new information to the attacker about their membership to the synthetic dataset. Interestingly, for the Correlations feature set, there is no target that has PG lower than 0.25. As for the CEU population, about 50% of the targets across all feature sets have a PG of 0.25 for data from Rec-RBM, and at least 47% of targets from WGAN.

**LogReg.** Using LogReg, both Recomb and RBM have the lowest PG among all attack classifiers, for both HapMap populations. For CEU (Fig. 6.32a, middle), using the Histogram feature set, 94% (resp., 96%) of the targets from Recomb (resp., RBM) have PG below 0.25, which is the random guess baseline. Under Correlations, 99% (resp., 97%) of the targets in Recomb (resp., RBM) have PG below 0.25, while, for the Ensemble feature set, 96% (resp., 98%) of the targets from Recomb (resp., RBM) have PG below 0.25. With Rec-RBM, we find that between 52% and 56% of the targets across all feature sets have PG above 0.25, and with WGAN, between 50% and 57% of the targets across all feature sets have PG above 0.25. Moreover, for the Rec-RBM and WGAN-generated data, there is no target that consistently has a lower PG than the random guess baseline across all test runs.

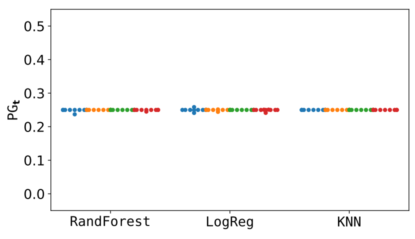
For CHB (Fig. 6.32b, middle), with synthetic data generated by Recomb, the average PG is below the random baseline (0.25) for 99% of the targets in the Histogram feature set, 97% for Correlations, and for 96% for Ensemble. For RBM, 79% of the targets in the Histogram feature set have a PG below 0.25. Under the Ensemble feature set, 84% of the targets have PG below 0.25. For synthetic data generated by Rec-RBM, we find that 54% of the targets from the Histogram and Ensemble feature sets have PG lower than 0.25 and 46% have PG over 0.25. For the Naive and Correlations feature sets, respectively, 45% and 47% of the targets have PG lower than the random guess baseline. For WGAN-generated data, Correlations feature set yields most targets (55%) with PG<0.25.

**RandForest.** When using RandForest as the attack classifier on data from the CEU population (Fig. 6.32a, right), with RBM-generated data, 73% of the targets from both Correlation and Histogram feature sets have lower PG than the random baseline. This is for 69% and 62% of the targets with, respectively, Ensemble and Naive feature sets. For the synthetic data generated by Rec-RBM, about 51% of targets have a PG of over 0.25, and 49% of the targets have PG of less than 0.25, across all feature sets. For WGAN, between 46% and 59% of the targets from all feature sets have a PG less than the random guess baseline (0.25), with the Correlations feature set having the least percentage of vulnerable targets (59%).

For CHB (Fig. 6.32b, right), the lowest privacy gain for the samples generated by Recomb: over 79% of all targets for each of the four feature sets have PG lower than the random baseline. For the synthetic samples from RBM, 71%, 61%, and 53% of the targets under the Naive, Histogram, respectively, Ensemble feature sets have PG lower than 0.25. However, for the Correlations feature sets, we find that 55% of the targets have a PG of 0.25, meaning that, those targets are protected from MIA. For the synthetic samples generated by Rec-RBM, 54% of the targets from the Histogram and Ensemble feature sets and 47 and 45% of the targets from the Correlations and Naive, respectively, have PG lower than 0.25. Finally, for the WGAN, we find that between 44% and 53% of the targets have PG>0.25 across all four feature sets.



**(a)** Random Target



**(b)** Outlier target

**Figure 6.33:** Privacy Gain (PG) of RBM over the 1000 Genomes dataset.

### 1000 Genomes Population

For the 1000 Genomes population, we focus our analysis on the RBM model, as it generated synthetic data closest to the real data across all utility metrics evaluated.

**Random Target.** In Figure 6.33a, we plot the PG for the synthetic data generated by RBM, for randomly chosen targets. With the RandForest MIA classifier, we observe that 56%, 75%, 92%, and 50% of the targets have PG higher than the random guess baseline (PG$\geq$0.25) for, respectively, Naive, Histogram, Correlations, and Ensemble feature sets. The high percentage of targets that have a PG$\geq$0.25 for the Correlations suggests that the impact of a single target in the training dataset of the RBM on the correlations of the synthetic dataset is minimal.

Then, with the LogReg classifier, we find a high variation in PG, similar to the HapMap populations in the case of Rec-RBM. Under the Naive feature set, half of the targets have PG gain below the random guess baseline. Similarly, 43%, 45%, and 43% of targets have a PG lower than 0.25 for, respectively, the Correlations, Histogram, and Ensemble feature sets.

Finally, for the KNN classifier, over 94% of the targets have a PG of 0.25 across all feature sets.

**Outlier Target.** To better understand whether, with more training data, a target's signal in the synthetic dataset is diluted, we also test an "extreme" outlier case. That is, we craft an outlier target that has only minor alleles at all positions. While we are aware that this case would be extremely rare in a real-world scenario, our goal is to observe whether, and how much, this impacts PG.

To this end, in Figure 6.33b, we plot the PG of this outlier case across 10 test runs.

With RandForest, we find that, under the Ensemble feature set, PG is below 0.25 for 8 of the 10 test runs. In fact, this is the only combination between attack classifier and feature set for which a greater percentage of the targets have a lower privacy gain than in the random target case. For the Naive feature set, in only 3 of the test runs, PG is below the random baseline. For the Correlations and Histogram feature sets, all test runs yield PG of 0.25 or above. while for the Histogram feature set the privacy gain is above 0.25 across all test runs.

With LogReg, 4 out of 10 of the test runs for the Naive, Histogram, and Correlations feature sets yield PG below 0.25. For Ensemble, this happens for 6 test runs. Finally, with KNN, across all feature sets, PG for all test runs is 0.25, i.e., the synthetic data does not disclose any membership information regarding the outlier.

While there are differences across classifiers and feature sets, the PG, for all test runs in this outlier target case, is centered around 0.25. This evident from Figure 6.33b, which implies that, across all test runs, the accuracy of the MIA is not much better than random guessing.

**Take-Aways** The different combinations of datasets, attack classifiers, and feature sets, yield varied results with respect to privacy. This is due to two main reasons: first, not all classifiers have the same accuracy on tasks for the same dataset, as shown in previous work [19]. Second, the features that the generative model preserves after training will “reflect” in the synthetic data; thus, this will impact the PG based on the feature extraction method.

On the HapMap populations, while the utility evaluation shows that Recomb-generated synthetic data is “closest” to the real data, it does so with a significant privacy loss in comparison to the other models. The RBM-generated synthetic data is the most vulnerable under the LogReg classifier, with at least 70% of the targets across both populations and all feature sets having PG below the random guess baseline. This suggests that, with few data samples available for training, the RBM model is likely to overfit and is thus susceptible to MIAs.

For Rec-RBM and WGAN, the attacker cannot reliably predict membership, i.e., the addition of extra samples from the Recomb model in the training of the Rec-RBM dilutes the target's signal in the training data. However, for both models, we still find combinations of targets and training sets for the attack classifier for which PG is significantly lower than the random guess baseline; i.e., the synthetic data will still expose membership information about the respective targets.

On the 1000 Genomes, PG values have a higher variation overall when the target is chosen randomly from the dataset than for the two smaller HapMap datasets. The results for RBM data confirm our hypothesis that the target's influence is diluted within larger datasets. However, once again, this does not mean that membership inference is not possible for both Rec-RBM and WGAN, depending on the combination of target, training set, attack classifier, and feature set.

However, in the case of an "extreme" outlier (i.e., a target which has minor alleles at all positions), the synthetic data generated by RBM does not have a big impact on PG. In this case, across all test runs, the PG is actually close to the random guess baseline across all test runs.

We find that targets get very different levels of privacy protection, based on the combination of target and training set used to generate the synthetic data. For a privacy mechanism to provide good privacy protection, it needs to be predictable, i.e., all targets should have a privacy gain above the random guess baseline, which our evaluation shows it is not the case for the data generated by the models we have evaluated.

#### 6.3.4.2 Privacy Gain under Membership Inference Attack with Partial Information

Next, we introduce a novel attack, which we denote as MIA with Partial Information (MIA-PI). Basically, we only give the attacker access to a fraction of SNVs from the target sequence, chosen at random. The attacker then uses the Recombination model from [156] as an inference method to predict the rest of the sequence. Compared to the previous attack, the adversary trains their (attack) classifier using the sequence *inferred* from the partial data. Thus, the PG formula also needs to be adjusted to account for how likely an adversary is to identify a target from partial information.

**PG for MIA-PI.** Assuming the attacker has partial information $t'$ as a fraction of the SNVs from $t$, they first use the Recomb model, as an inference algorithm, to predict the rest of the SNVs from the target sequence, which we denote by $t_p$. The privacy gain is computed as:

$$PG_t = \frac{\overline{MIA_{t_p}}(R_t) - \overline{MIA_{t_p}}(S_{test})}{2}, \text{ where}$$
$$\overline{MIA_{t_p}}(S_{test}) = \sum_{S_i \in S_{test}} \frac{\Pr[MIA_{t_p}(S_i) = 1]}{2 * n_s}, \text{ and}$$
$$\overline{MIA_{t_p}}(R_t) = \sum_{R_i \in R_t} \frac{\Pr[MIA_{t_p}(R_i) = 1]}{2 * n_s}.$$

That is, the privacy gain, in this case, is computed as the difference between the probability that the attacker, who has partial information about the target record, correctly identifies that the target is part of the real dataset versus the target being part of the training set used to generate the synthetic dataset.

As a result, PG now ranges between -0.5 and 0.5, where 0.5 means that having the real dataset $R$ and the partial information $t'$ about the target allows the adversary to infer the membership of $t$ in $R$, while the synthetic dataset reduces the adversary's chance of success (i.e. $\overline{MIA_{t_p}}(R_t) = 1$ and $\overline{MIA_{t_p}}(S_{test}) = 0$). A negative PG value means that publishing the synthetic data, instead of the real data, improves the adversary's chance to correctly infer membership of the target $t$ (i.e. $\overline{MIA_{t_p}}(R_t) < \overline{MIA_{t_p}}(S_{test})$). If publishing the synthetic data does not increase nor decrease the adversary's inference powers, we should have PG=0 (i.e. $\overline{MIA_{t_p}}(R_t) = \overline{MIA_{t_p}}(S_{test})$).

**Experiments.** From the experiments presented in Section 6.3.4.1, we find that the attack classifier that yields the lowest PG is Logistic Regression; thus, we only experiment with that one to ease presentation. In the following, we present the results of the MIA-PI experiments for the *CEU population*, focusing on the *Recomb and RBM models* (as mentioned, with a LogReg attack classifier). We do so as these two models yield the lowest PG in Section 6.3.4.1.

**Recomb.** In Figure 6.34, we plot the Cumulative Distribution Function (CDF) of the accuracy of the attack for Recomb when the adversary has access to the full sequence vs. partial information, specifically, a ratio of 0.05, 0.1, and 0.2 of the total SNVs from the target sequence. Interestingly, even when only 0.05 of the target SNVs are available to the attacker, for 90% and 91% of the targets from the Histogram and respectively Ensemble feature sets, the attacker's accuracy is still above the random guess baseline (50% accuracy). Our intuition is that many targets are vulnerable to the attack, even with little partial information, since we use the Recomb model not only for the attack but also as an inference method to predict the rest of the sequence.

To explore how much of the MIA-PI vulnerability is due to the release of synthetic datasets, and not only by how much information the attacker has available, in Figure 6.35, we plot the CDF of the PG with MIA-PI. In line with the accuracy results, we find that the PG is greater than 0 for at least 88% of the targets for all ratios of partial information tested in the case of the Correlations feature set. However, for the other three feature sets, releasing the synthetic dataset instead of the real data decreases the privacy gain (i.e., PG<0) for the majority of targets. When the adversary has access to just 5% of the SNVs from the target, there is a negative PG for 61% of the targets under the Histogram and 62% of the targets under the Ensemble feature sets. With 10% of the target sequence available, 54% and 60% of the targets under, respectively, the Histogram and the Ensemble feature sets have negative PG, and with 20%, these numbers go up to 64% and 67%. There are more targets with negative PG with increasing partial information available to the attacker for the Naive feature set, i.e., 59%, 70%, and 84% with, respectively, 5%, 10%, and 20% of the target sequence available. Overall, this shows that releasing the synthetic dataset instead of the real data does not fully mitigate privacy concerns, even when the attacker does not have access to the full sequence.

**RBM.** In Figure 6.36, we plot the CDF for the accuracy of the attack for the RBM model for both full and partial information about the target record available to the attacker. Across all feature sets, there is an increase in the accuracy of the attack

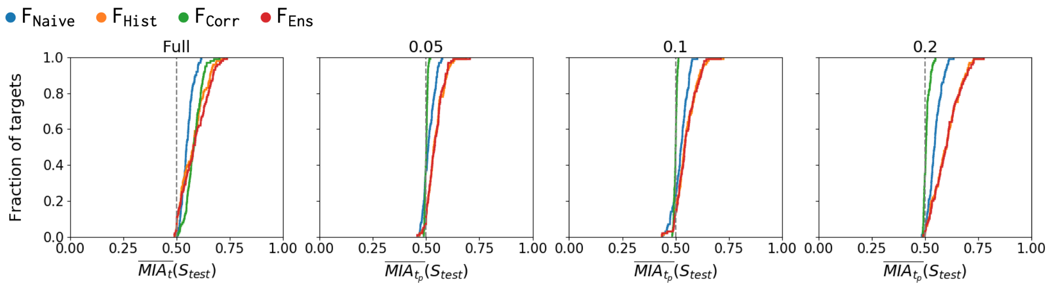


**Figure 6.34:** Accuracy of the Membership Inference Attack with access to full sequence and partial information (0.05, 0.1, and 0.2 ratio) for **Recomb**.

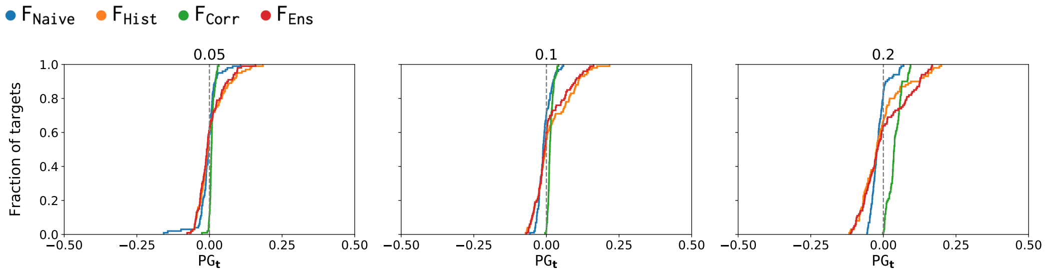


**Figure 6.35:** Privacy Gain (PG) for synthetic samples generated by **Recomb**.

with more information available to the attacker, as is expected. We also look at CDF for the PG in the case of partial information available to the attacker in Figure 6.37. Once again, under the Naive feature set, increasing the partial information available to the attacker negatively correlates with the percentage of targets with a negative PG. Under all other feature sets, for most targets, releasing the synthetic dataset instead of the real data yields a positive PG, meaning that releasing the synthetic dataset instead of the real dataset improves the PG.

**Takeaways.** We find that not even decreasing the attacker's power by only giving him partial information from the target sequence mitigates privacy for the Recomb-generated synthetic data. This is likely because, using the Recomb model as a generative model and an inference model, the adversary's inference power is increased since the feature set extracted from the synthetic data will be closer to the feature set for the predicted target.

However, in this case, for RBM, we see an increase in the privacy gained by releasing synthetic data instead of real data. This implies that, even if the RBM is likely to overfit when few samples are available for training, it does so on the predicted sequence of the target rather than on the full sequence and thus decreases

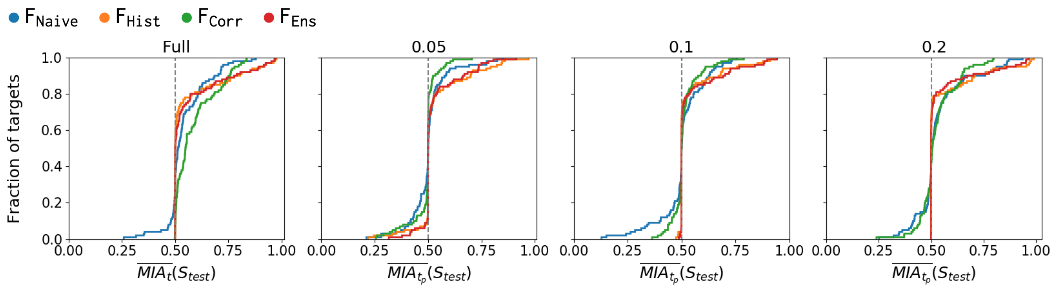


**Figure 6.36:** Accuracy of the Membership Inference Attack with access to full sequence and partial information (0.05, 0.1, and 0.2 ratio) for **RBM**.

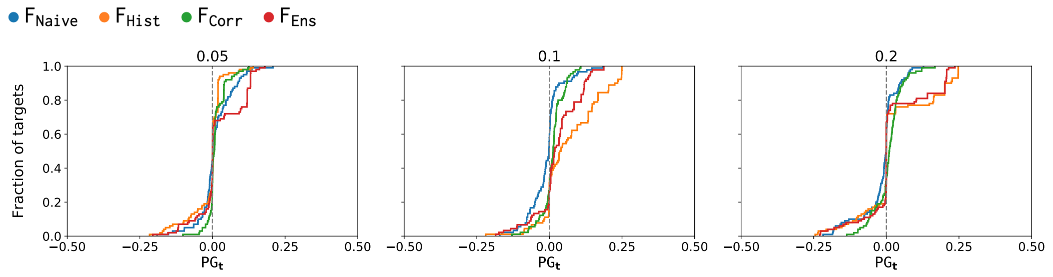


**Figure 6.37:** Privacy Gain (PG) for synthetic samples generated by **RBM**.

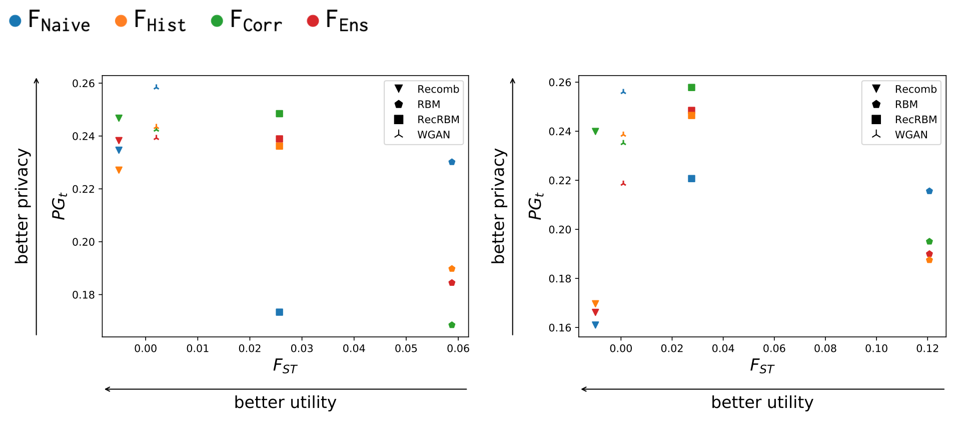
the accuracy of the MIA.

Overall, not even with partial data from the target sequence, we obtain privacy gain values constantly better than random guessing, which, as mentioned before, indicates that synthetic data is not really a reliable privacy defense.

#### 6.3.4.3 Privacy Gain vs. $F_{ST}$

To provide a quick visualization of the *trade-offs* between privacy and utility, we also plot the $F_{ST}$ (see Section 6.3.3.2) against the PG in Figure 6.38 for the two smaller HapMap Populations. For this set of experiments, we randomly sample 10 targets from each dataset. We train a Logistic Regression attack classifier using 5 shadow models, using 100 synthetic datasets for each of them. We then compute the PG and $F_{ST}$ on 100 synthetic datasets, with a split of 50 sets generated from a training set including the target and 50 sets generated without. We report the PG and $F_{ST}$ as the average across all targets and synthetic datasets.

Recall that the lower the value of the $F_{ST}$ the closer the synthetic samples are to the real data, and that the PG should be above the random guess baseline of 0.25 for the synthetic data to offer better privacy protection than releasing the real dataset. In other words, the ideal “place” in the plots in Figure 6.38 is the top left.



**(a)** CEU Population **(b)** CHB Population

**Figure 6.38:** Fixation index values vs Privacy gain for the HapMap populations.

For the CEU population (Figure 6.38a), the samples generated by the Recomb model have the best overall utility; however, the average privacy gain is below the random guess baseline. In line with our utility evaluation, we find that the hybrid model Rec-RBM model generates samples with a $F_{ST}$ value closer to the real data than the RBM model. The interesting point observation here is that while for Histogram, Correlation, and Ensemble feature sets, the PG for the Rec-RBM is close to 0.25, there is a significant privacy loss for the Naive feature set. In contrast, for the samples generated by the RBM, the Naive feature set yields the highest PG across all feature sets. The samples generated by the WGAN also have a $F_{ST}$ value close to 0; however, for the Histogram, Correlation and Ensemble, the PG values are below 0.25.

For the CHB population (Figure 6.38b), the utility results are similar to the CEU population. However, all values of PG apart from the ones generated by the RecRBM have a lower PG than the CEU population. This reiterates the fact that one cannot reliably use synthetic data as a good privacy mechanism, as the value of PG is unpredictable and can fluctuate based on target and training set combinations chosen.

### 6.3.5 Key Takeaways

This section presented an in-depth measurement study of state-of-the-art methods to generate synthetic genomic data. We did so vis-à-vis 1) their utility, with respect to a number of common analytical tasks performed by researchers, as well as 2) the privacy protection they provide compared to releasing real data.

High-quality synthetic data must accurately capture the relations between data points, however, this can enable attackers to infer sensitive information about the training data used to generate the synthetic data. This was illustrated by the performance of the Recomb model on the HapMap datasets: while it achieves the best utility, it does so at the cost of significantly reducing privacy.

Overall, there is no single method that outperforms the others for all metrics and all datasets. However, we did find that models based on a simple GAN architecture (i.e., GAN and Rec-GAN) are not a good fit to genomic data, as they provide the lowest utility across the board.

Our analysis revealed that the size of the training dataset matters, especially in the case of generative models. Not only we saw an improvement in utility with the addition of samples in the hybrid Rec-RBM approach for the smaller HapMap datasets, and for RBM and WGAN for the 1000 Genomes dataset, but we also measured a decrease in the number of targets exposed to membership inference.

Finally, we are confident that our techniques can be used by practitioners to assess the risks of deploying synthetic genomic data in the wild and serve as a benchmark for future work.

## 6.4 Discussion

In this chapter we took a critical look at synthetic data generation methods and provided an analysis both from an utility and a privacy perspective.

From a utility perspective, our initial experimental evaluation suggests that a generic approach which would successfully be able to generate universally meaningful synthetic datasets might not be viable. This is due to the complexity and varied nature of the datasets used in the wild. Hence, it remains up to the data provider to decide if the offset in utility associated with privacy-preserving synthetic data satisfies their needs. Overall, there is no “best” model among the ones we tested, as they all exhibit different performances on different datasets.

When it comes to privacy, we find that synthetic data is not the silver bullet solution it was assumed to be. In fact, synthetic data does not prevent membership inference, i.e., an adversary can infer whether a particular record was in the raw dataset. These attacks are (mostly) possible *regardless of the model used to generate the data*, and in some cases synthetic data even increases the inference power of the adversary. Protection from these attacks can only be achieved if the generative model does not accurately reproduce certain features of the raw data which implies a necessary loss in utility.

Finally, in our experimental evaluation on generative models purposely proposed for genomic data, again, we conclude that there is no single approach for generating genomic synthetic data that performs well across the board, both in terms of utility and privacy. As expected, the generative model that output the "best" utility metrics, did so at the cost of privacy, even against weaker adversaries who only have partial information available. Although there are cases where releasing the synthetic dataset sometimes offers better protection against membership inference attacks, because of the randomness introduced by the generative models, one cannot meaningfully predict a target's susceptibility to privacy attacks without fixing the training set and quantifying the respective privacy loss/gain for all targets in the set. Hence, releasing synthetic datasets does not provide robust protection against membership inference attacks.

**Chapter 7**

# Conclusion

Motivated by the quick progress in genomics sequencing, the research community has produced a large body of work on genomic data, to which this thesis aims to contribute.

First, we look at one of the existing genomic data sharing platforms, MME, and propose a framework geared to bring anonymity to the platform, AnoniMME (Chapter 4). By relying on reverse PIR, AnoniMME is compatible with the requirement of MME, but adds anonymous queries with a low overhead, as we demonstrate empirically. Thus, we are confident that AnoniMME can eventually be deployed in the wild and further encourage researcher to share genomic data, by minimizing the possibility of exposing confidential research when using MME.

Then, knowing that consequences of genomic data disclosure are not limited in time or to the data owner, we empirically evaluate the only tool proposed for long-term encryption of genomic data, GenoGuard, which is based on Honey Encryption (Chapter 5). We show that in the case of low entropy passwords, the attacker can easily exclude decoy passwords from the pool of possible passwords, and can guess the correct sequence with high probability. We also evaluate GenoGuard for high-entropy passwords, and find that with access to a GenoGuard encrypted ciphertext, the attacker has a non-negligible advantage compared to using state of the art inference methods.

Next, we study the feasibility of genomic synthetic data. Keeping in mind that, in the case of genomic data, high-quality synthetic data is needed in order to enable research, we start our study by looking at generic datasets to understand the utility of synthetic data from privacy preserving generative models. We find that there is no model that performs well for all types of datasets, and that in order to obtain better utility from the synthetic data, synthesis procedures that are more specialized to concrete data types are preferred. We then proceed to a privacy assessment of the same models, quantifying the privacy loss associated with publishing the synthetic datasets. Last but not least, present an in-depth measurement study of state-of- the-art methods to generate synthetic genomic data. We do so vis-à-vis their utility, with respect to several common analytical tasks performed by researchers and the privacy protection they provide compared to releasing real data. Overall, we find that there is no single generative method that yields both high utility and strong privacy across the board.

Finally, we hope that the research presented in this thesis will inspire further work on privacy-preserving techniques for genomic data sharing, thus enabling researchers to further progress in biomedical research, personalized medicine and drug development.

**Limitations.** Before we conclude this thesis we present some limitations of our current work.

The work presented in Chapter 4 proposes a framework to support anonymous queries within the genomic data sharing platform Matchmaker Exchange. While we experimentally show that our framework is scalable and efficient, it would benefit from a user-study simulating a real-world deployment of AnoniMME with users of the MME, aiming to evaluate its usability with respect to anonymity protection, delays introduced by epochs. Additionally, our framework could benefit by an extension to allow the execution of the response phase over multiple query epochs, and further reduction in bandwidth.

In Chapter 5, we analyze the only system designed for long-term encryption of genomic data. While we show that there exists leakage of information arising from the GenoGuard ciphertext, we did not explore any methods that would help mitigate this risk. Additional work can be done in exploring the privacy leakage of

GenoGuard for side information arising from kinship associations.

In Chapter 6, our evaluation focuses on existing generative methods for synthetic genomic data; thus, we have not engaged in fine-tuning the (hyper-)parameters of the models evaluated. Moreover, one might argue that the techniques we evaluate were not designed with privacy in mind, unlike previous work on differentially private generative models for images or clinical data [16, 30, 41]. That is, it is not entirely surprising that they yield small privacy gains. However, to the best of our knowledge, *no differentially private generative model has been proposed for genomic data*, which is the focus of our study. In fact, prior work has shown that, for precision medicine applications, the high dimensionality of the data tends to be a major limitation, resulting in poor utility for differentially private mechanisms [27, 68, 99, 184, 181]. Differentially private techniques for GWAS are also known to yield poor accuracy as the number of features is large, relative to the number of patients in a study [98].

**Future work.** Finally, we conclude this thesis by highlighting open research problems and items for future work.

**Long-term security of genomic data.** As also identified by Mittos et al. [118], genomic privacy literature has not sufficiently dealt with long term security. Even though Huang et al. [92] made some steps in the right direction, our evaluation shows that the problem of long term security for genomic data is far from being solved. However, this is not an easy problem to solve as pointed out by the vast array of answers from the experts interviewed in [118]: from using post-quantum cryptography or information theoretic solutions to "maybe cryptography is not the answer. Perhaps setting up an environment with different ways of controlling how the data is managed in order to provide more transparency" .

**Privacy-Preserving Synthetic Genomic Data.** Even though at the time of writing this thesis, no privacy-preserving generative models for genomic data have been proposed, it would be interesting to experiment with the possible adaptation of differentially private models to genomic data and evaluate them in future work. Finally, another interesting research topic would be to extend our privacy evaluation to understand how much the privacy loss stemming from releasing (synthetic) genomic datasets affects the relatives of those included in the training set of the corresponding generative models [93, 166].

# Bibliography

[1] https://gdc.cancer.gov/.

[2] https://github.com/JiaMingLin/privbayes.

[3] All of Us Research Program - National Institutes of Health (NIH). https://allofus.nih.gov/future-health-begins-all-us.

[4] Generating and designing DNA. https://github.com/co9olguy/Generating-and-designing-DNA.

[5] Genetic maps for the 1000 Genomes Project variants. https://github.com/joepickrell/1000-genomes-genetic-maps.

[6] Genomics England. https://www.genomicsengland.co.uk/.

[7] German Credit Risk. https://www.kaggle.com/uciml/german-credit. Accessed 2020-03-03.

[8] Internet Storm Center | DShield. http://www.dshield.org.

[9] New synthetic datasets to assist COVID-19 and cardiovascular research. https://www.gov.uk/government/news/new-synthetic-datasets-to-assist-covid-19-and-cardiovascular-research.

[10] Synthetic data privacy evaluation framework. https://figshare.com/s/2d0f6718476e9c07f8e2.

[11] The Cost of Sequencing a Human Genome. https://www.genome.gov/sequencingcosts/. Accessed.

[12] 1000 Genomes Project Consortium. A global reference for human genetic variation. *Nature*, 526(7571), 2015.

[13] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya

Mironov, Kunal Talwar, and Li Zhang. Deep Learning with Differential Privacy. In *ACM CCS*, 2016.

[14] Nazmiye Ceren Abay, Yan Zhou, Murat Kantarcioglu, Bhavani Thuraisingham, and Latanya Sweeney. Privacy preserving synthetic data release using deep learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2018.

[15] John M Abowd and Lars Vilhuber. How protective are synthetic data? In *International Conference on Privacy in Statistical Databases*, pages 239–246. Springer, 2008.

[16] Gergely Acs, Luca Melis, Claude Castelluccia, and Emiliano De Cristofaro. Differentially private mixture of generative neural networks. *IEEE Transactions on Knowledge and Data Engineering*, 2018.

[17] Charu C Aggarwal. On k-anonymity and the curse of dimensionality. In *VLDB*, 2005.

[18] Naomi Altman and Martin Krzywinski. The curse(s) of dimensionality. *Nature Methods*, 15(6), 2018.

[19] Diego Raphael Amancio, Cesar Henrique Comin, Dalcimar Casanova, Gonzalo Travieso, Odemir Martinez Bruno, Francisco Aparecido Rodrigues, and Luciano da Fontoura Costa. A systematic comparison of supervised classifiers. *PloS one*, 9(4):e94137, 2014.

[20] Christof Angermueller, Tanel Pärnamaa, Leopold Parts, and Oliver Stegle. Deep learning for computational biology. *Molecular Systems Biology*, 12(7), 2016.

[21] Maryam Archie, Sophie Gershon, Abigail Katcoff, and Aaron Zeng. De-anonymization of Netflix Reviews using Amazon Reviews. https://courses.csail.mit.edu/6.857/2018/project/Archie-Gershon-Katchoff-Zeng-Netflix.pdf.

[22] Euan A Ashley. Towards precision medicine. *Nature Reviews Genetics*, 17(9):507–522, 2016.

[23] Giuseppe Ateniese, Luigi V Mancini, Angelo Spognardi, Antonio Villani, Domenico Vitali, and Giovanni Felici. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *International Journal of Security and Networks*, 10(3), 2015.

[24] Erman Ayday, Emiliano De Cristofaro, Jean-Pierre Hubaux, and Gene Tsudik. The Chills and Thrills of Whole Genome Sequencing. *IEEE Computer*, 2015.

[25] Erman Ayday, Emiliano De Cristofaro, Jean-Pierre Hubaux, and Gene Tsudik. Whole genome sequencing: Revolutionary medicine or privacy nightmare? *Computer*, 48(2), 2015.

[26] Erman Ayday, Jean Louis Raisaro, Jean-Pierre Hubaux, and Jacques Rougemont. Protecting and Evaluating Genomic Privacy in Medical Tests and Personalized Medicine. In *ACM Workshop on Privacy in the Electronic Society*, 2013.

[27] C-A Azencott. Machine learning and genomics: precision medicine versus patient privacy. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 376(2128), 2018.

[28] Md Momin Al Aziz, Md Nazmus Sadat, Dima Alhadidi, Shuang Wang, Xiaoqian Jiang, Cheryl L Brown, and Noman Mohammed. Privacy-Preserving Techniques of Genomic Data – A Survey. *Briefings in Bioinformatics*, 2017.

[29] Pierre Baldi, Roberta Baronio, Emiliano De Cristofaro, Paolo Gasti, and Gene Tsudik. Countering Gattaca: Efficient and Secure Testing of Fully-Sequenced Human Genomes. In *ACM Conference on Computer and Communications Security*, 2011.

[30] Brett K Beaulieu-Jones, Zhiwei Steven Wu, Chris Williams, Ran Lee, Sanjeev P Bhavnani, James Brian Byrd, and Casey S Greene. Privacy-preserving generative deep neural networks support clinical data sharing. *Circulation: Cardiovascular Quality and Outcomes*, 12(7):e005122, 2019.

[31] Vincent Bindschaedler and Reza Shokri. Synthesizing plausible privacy-

preserving location traces. In *IEEE Symposium on Security and Privacy*, 2016.

[32] Vincent Bindschaedler, Reza Shokri, and Carl A Gunter. Plausible deniability for privacy-preserving data synthesis. *VLDB*, 2017.

[33] Vincent Bindschaedler, Reza Shokri, and Carl A Gunter. Plausible deniability for privacy-preserving data synthesis. *arXiv preprint arXiv:1708.07975*, 2017.

[34] Siddharth Biswal, Soumya Ghosh, Jon Duke, Bradley Malin, Walter Stewart, and Jimeng Sun. EVA: Generating Longitudinal Electronic Health Records Using Conditional Variational Autoencoders. *arXiv:2012.10020*, 2020.

[35] Leo Breiman. Random forests. *Machine learning*, 45, 2001.

[36] Janet Burns. GOP Bill Could Force Employees To Undergo DNA Tests Or Pay Huge Fines. https://www.forbes.com/sites/janetwburns/2017/03/14/gop-bill-could-force-employees-to-undergo-dna-testing-or-pay-thousands/, 2017.

[37] William S Bush and Jason H Moore. Genome-wide association studies. *PLoS Computational Biology*, 8(12), 2012.

[38] Orion J. Buske et al. The Matchmaker Exchange API: Automating patient matching through the exchange of structured phenotypic and genotypic profiles. *Human mutation*, 36:922–927, 2015.

[39] Ewen Callaway. HeLa publication brews bioethical storm. *Nature*, 2013.

[40] Nicholas Carlini, Chang Liu, Jernej Kos, Úlfar Erlingsson, and Dawn Song. The secret sharer: Measuring unintended neural network memorization & extracting secrets. *arXiv:1802.08232*, 2018.

[41] Dingfan Chen, Tribhuvanesh Orekondy, and Mario Fritz. Gs-wgan: A gradient-sanitized approach for learning differentially private generators. arXiv preprint arXiv:2006.08265, 2020.

[42] Dingfan Chen, Ning Yu, Yang Zhang, and Mario Fritz. GAN-Leaks: A taxonomy of membership inference attacks against GANs. In *ACM Conference*

*on Computer and Communications Security*, 2020.

[43] Feng Chen et al. PRINCESS: Privacy-protecting Rare disease International Network Collaboration via Encryption through Software guard extensionS. *Bioinformatics*, 33:871–878, 2017.

[44] Haibo Cheng, Zhixiong Zheng, Wenting Li, Ping Wang, and Chao-Hsien Chu. Probability model transforming encoders against encoding attacks. In *USENIX Security*, 2019.

[45] Haibo Cheng, Zhixiong Zheng, Wenting Li, Ping Wang, and Chao-Hsien Chu. Probability model transforming encoders against encoding attacks. In *28th {USENIX} Security Symposium ({USENIX} Security 19)*, pages 1573–1590, 2019.

[46] Edward Choi, Siddharth Biswal, Bradley Malin, Jon Duke, Walter F Stewart, and Jimeng Sun. Generating multi-label discrete patient records using generative adversarial networks. In *Machine learning for Healthcare Conference*, 2017.

[47] Benny Chor and Niv Gilboa. Computationally Private Information Retrieval (Extended Abstract). In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, pages 304–313, 1997.

[48] Benny Chor, Eyal Kushilevitz, Oded Goldreich, and Madhu Sudan. Private Information Retrieval. *Journal of the ACM*, 45(6):965–981, 1998.

[49] C Chow and Cong Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 1968.

[50] Geraldine M. Clarke and Lon R. Cardon. Disentangling Linkage Disequilibrium and Linkage From Dense Single-Nucleotide Polymorphism Trio Data. *Genetics*, 171(4), 2005.

[51] International HapMap Consortium et al. The international HapMap project. *Nature*, 426(6968):789, 2003.

[52] Henry Corrigan-Gibbs et al. Riposte: An Anonymous Messaging System Handling Millions of Users. In *Proceedings of the 2015 IEEE Symposium on*

*Security and Privacy*, pages 321–338, 2015.

[53] National Research Council et al. *Putting people on the map: Protecting confidentiality with linked social-spatial data.* National Academies Press, 2007.

[54] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27, 1967.

[55] David R Cox. The regression analysis of binary sequences. *Journal of the Royal Statistical Society: Series B (Methodological)*, 20(2):215–232, 1958.

[56] Fida Kamal Dankar and Khaled El Emam. The Application of Differential Privacy to Health Data. In *Joint EDBT/ICDT Workshops*, 2012.

[57] Dua Dheeru and Efi Karra Taniskidou. UCI Machine Learning Repository. http://mlr.cs.umass.edu/ml/datasets.html, 2017.

[58] Irit Dinur and Kobbi Nissim. Revealing information while preserving privacy. In *ACM PODS*, 2003.

[59] Cynthia Dwork and Kobbi Nissim. Privacy-preserving datamining on vertically partitioned databases. In *IACR CRYPTO*, 2004.

[60] Cynthia Dwork and Aaron Roth. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends in Theoretical Computer Science*, 9, 2013.

[61] Haris Elias. Synthetic data - all the perks without the risk? https://techhq.com/2020/10/synthetic-data-all-the-perks-without-the-risk/, 2020.

[62] EMBL-EBI Training. What are genome wide association studies (GWAS)? https://www.ebi.ac.uk/training-beta/online/courses/gwas-catalogue-exploring-snp-trait-associations/what-is-gwas-catalog/what-are-genome-wide-association-studies-gwas/, 2021.

[63] Genomics England. The 100,000 genomes project. *The*, 100:0–2, 2016.

[64] Yaniv Erlich, Tal Shor, Itsik Pe'er, and Shai Carmi. Identity inference of genomic data using long-range familial searches. *Science*, 362(6415):690–694, 2018.

[65] Steven N Evans, Yelena Shvets, and Montgomery Slatkin. Non-equilibrium theory of the allele frequency spectrum. *Theoretical Population Biology*, 71(1), 2007.

[66] Ronald Aylmer Fisher. The distribution of gene ratios for rare mutations. *Proceedings of the Royal Society of Edinburgh*, 50, 1931.

[67] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *ACM Conference on Computer and Communications Security*, 2015.

[68] Matthew Fredrikson, Eric Lantz, Somesh Jha, Simon Lin, David Page, and Thomas Ristenpart. Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing. In *USENIX Security*, 2014.

[69] Bernard Friedenson. The BRCA1/2 pathway prevents hematologic cancers in addition to breast and ovarian cancers. *BMC cancer*, 7(1):152, 2007.

[70] Karan Ganju, Qi Wang, Wei Yang, Carl A Gunter, and Nikita Borisov. Property inference attacks on fully connected neural networks using permutation invariant representations. In *ACM Conference on Computer and Communications Security*, 2018.

[71] Genetics Home Reference. What is Precision Medicine? https://medlineplus.gov/genetics/understanding/precisionmedicine/definition/, 2020.

[72] Global Alliance for Genomics and Health. A federated ecosystem for sharing genomic, clinical data. *Science*, 352:1278–1280, 2016.

[73] Global Alliance for Genomics and Health. https://www.ga4gh.org/, 2017.

[74] I. Goldberg. Improving the Robustness of Private Information Retrieval. In *IEEE Symposium on Security and Privacy*, pages 131–148, 2007.

[75] Andre Goncalves, Priyadip Ray, Braden Soper, Jennifer Stevens, Linda Coyle, and Ana Paula Sales. Generation and evaluation of synthetic patient data. *BMC Medical Research Methodology*, 20, 2020.

[76] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Gener-

ative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

[77] Michael T Goodrich and Michael Mitzenmacher. Invertible bloom lookup tables. In *2011 49th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 792–799. IEEE, 2011.

[78] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved Training of Wasserstein GANs. In *Advances in Neural Information Processing Systems*, 2017.

[79] Melissa Gymrek, Amy L McGuire, David Golan, Eran Halperin, and Yaniv Erlich. Identifying personal genomes by surname inference. *Science*, 339(6117), 2013.

[80] Moritz Hardt, Katrina Ligett, and Frank McSherry. A simple and practical algorithm for differentially private data release. In *NeurIPS*, 2012.

[81] David Harris and Sarah Harris. *Digital Design and Computer Architecture*. Morgan Kaufmann, 2010.

[82] Jamie Hayes, Luca Melis, George Danezis, and Emiliano De Cristofaro. LOGAN: Membership Inference Attacks Against Generative Models. *Proceedings on Privacy Enhancing Technologies*, 2019(1), January 2019.

[83] Jamie Hayes, Luca Melis, George Danezis, and Emiliano De Cristofaro. LOGAN: Membership inference attacks against generative models. *Proceedings on Privacy Enhancing Technologies*, 2019.

[84] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[85] Benjamin Hilprecht, Martin Härterich, and Daniel Bernau. Monte Carlo and Reconstruction Membership Inference Attacks against Generative Models. *Proceedings on Privacy Enhancing Technologies*, 2019.

[86] John L Hodges. The significance probability of the Smirnov two-sample test. *Arkiv för Matematik*, 3(5), 1958.

[87] Hans Hofmann. UCI Machine Learning Repository, 1998.

[88] Kent E. Holsinger and Bruce S. Weir. Genetics in geographically structured populations: defining, estimating and interpreting FST. *Nature Reviews Genetics*, 10(9), 2009.

[89] Nils Homer, Szabolcs Szelinger, Margot Redman, David Duggan, Waibhav Tembe, Jill Muehling, John V Pearson, Dietrich A Stephan, Stanley F Nelson, and David W Craig. Resolving individuals contributing trace amounts of DNA to highly complex mixtures using high-density SNP genotyping microarrays. *PLoS Genetics*, 2008.

[90] Justin Hsu, Marco Gaboardi, Andreas Haeberlen, Sanjeev Khanna, Arjun Narayan, Benjamin C Pierce, and Aaron Roth. Differential privacy: An economic method for choosing epsilon. In *2014 IEEE 27th Computer Security Foundations Symposium*, pages 398–410. IEEE, 2014.

[91] Jingchen Hu, Jerome P Reiter, and Quanli Wang. Disclosure risk evaluation for fully synthetic categorical data. In *International conference on privacy in statistical databases*, pages 185–199. Springer, 2014.

[92] Z. Huang, E. Ayday, J. Fellay, J. P. Hubaux, and A. Juels. GenoGuard: Protecting Genomic Data against Brute-Force Attacks. In *IEEE Symposium on Security and Privacy*, 2015.

[93] Mathias Humbert, Erman Ayday, Jean-Pierre Hubaux, and Amalio 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*, pages 1141–1152, 2013.

[94] Hae Kyung Im, Eric R Gamazon, Dan L Nicolae, and Nancy J Cox. On Sharing Quantitative Trait GWAS Results in an Era of Multiple-Omics Data and the Limits of Genomic Privacy. *The American Journal of Human Genetics*, 90(4), 2012.

[95] Joseph Jaeger, Thomas Ristenpart, and Qiang Tang. Honey encryption beyond message recovery security. In *EUROCRYPT*, 2016.

[96] Bargav Jayaraman and David Evans. Evaluating differentially private ma-

chine learning in practice. In *USENIX Security*, 2019.

[97] Tony Jebara. *Discriminative, generative and imitative learning*. PhD thesis, PhD thesis, Media laboratory, MIT, 2001.

[98] Aaron Johnson and Vitaly Shmatikov. Privacy-preserving data exploration in genome-wide association studies. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1079–1087, 2013.

[99] James Jordon, Jinsung Yoon, and Mihaela van der Schaar. PATE-GAN: Generating synthetic data with Differential Privacy guarantees. In *International Conference on Learning Representations, ICLR*, 2019.

[100] Ari Juels, Thomas Ristenpart, and Elisabeth Oswald. Honey Encryption: Security Beyond the Brute-Force Bound. In *EUROCRYPT*, 2014.

[101] Ari Juels and Ronald L Rivest. Honeywords: Making password-cracking detectable. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 145–160, 2013.

[102] Liina Kamm et al. A new way to protect privacy in large-scale genome-wide association studies. *Bioinformatics*, 29:886–893, 2013.

[103] Nathan Killoran, Leo Lee, Andrew Delong, David Duvenaud, and Brendan Frey. Generating and designing DNA with deep generative models. arXiv:1712.06148, 2017.

[104] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv:1312.6114*, 2013.

[105] Ronny Kohavi and Barry Becker. UCI Machine Learning Repository, 2013.

[106] Jonathan JM Landry, Paul Theodor Pyl, Tobias Rausch, Thomas Zichner, Manu M Tekkedil, Adrian M Stütz, Anna Jauch, Raeka S Aiyar, Gregoire Pau, Nicolas Delhomme, et al. The genomic and transcriptomic landscape of a HeLa cell line. *G3: Genes, Genomes, Genetics*, pages g3–113, 2013.

[107] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied

to handwritten zip code recognition. *Neural computation*, 1(4), 1989.

[108] Yann LeCun, Corina Cortes, and Chris Burges. MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/.

[109] Jaewoo Lee and Chris Clifton. How much is enough? choosing $\varepsilon$ for differential privacy. In *International Conference on Information Security*, pages 325–340. Springer, 2011.

[110] Klas Leino and Matt Fredrikson. Stolen memories: Leveraging model memorization for calibrated white-box membership inference. *arXiv preprint arXiv:1906.11798*, 2019.

[111] Haoran Li, Li Xiong, Lifan Zhang, and Xiaoqian Jiang. DPSynthesizer: differentially private data synthesizer for privacy preserving data sharing. In *Proceedings of the VLDB Endowment International Conference on Very Large Data Bases*, volume 7, 2014.

[112] Na Li and Matthew Stephens. Modeling linkage disequilibrium and identifying recombination hotspots using single-nucleotide polymorphism data. *Genetics*, 165, 2003.

[113] Yunhui Long, Vincent Bindschaedler, and Carl A Gunter. Towards measuring membership privacy. *arXiv:1712.09136*, 2017.

[114] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of machine learning research*, 2008.

[115] Bernard Marr. Forbes – How Much Data Do We Create Every Day? https://bit.ly/2Iz6Hbv, 2018.

[116] Luca Melis, Apostolos Pyrgelis, and Emiliano De Cristofaro. On collaborative predictive blacklisting. *ACM SIGCOMM CCR*, 2019.

[117] Ilya Mironov. Rényi differential privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 263–275. IEEE, 2017.

[118] Alexandros Mittos, Bradley Malin, and Emiliano De Cristofaro. Systematizing Genome Privacy Research: A Privacy-Enhancing Technologies Perspective. *Proceedings on Privacy Enhancing Technologies*, 1, 2019.

[119] Alexander Mordvintsev, Christopher Olah, and Mike Tyka. Inceptionism: Going Deeper into Neural Networks. https://ai.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html, 2015.

[120] Maurizio Naldi and Giuseppe D'Acquisto. Differential privacy: An estimation theory-based method for choosing epsilon. *arXiv preprint arXiv:1510.00917*, 2015.

[121] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets: a decade later. *May*, 21:2019, 2019.

[122] National Human Genome Research Institute. International HapMap Project. https://www.genome.gov/10001688/international-hapmap-project, 2012.

[123] National Human Genome Research Institute. The cost of sequencing a human genome. https://www.genome.gov/sequencingcosts/, 2017.

[124] National Human Genome Research Institute. Genome-Wide Association Studies Fact Sheet. https://www.genome.gov/about-genomics/fact-sheets/Genome-Wide-Association-Studies-Fact-Sheet, 2021.

[125] National Institute of Health. The All of Us Research Program. https://allofus.nih.gov/, 2017.

[126] Muhammad Naveed, Shashank Agrawal, Manoj Prabhakaran, XiaoFeng Wang, Erman Ayday, Jean-Pierre Hubaux, and Carl Gunter. Controlled Functional Encryption. In *ACM Conference on Computer and Communications Security*, 2014.

[127] Muhammad Naveed, Erman Ayday, Ellen W Clayton, Jacques Fellay, Carl A Gunter, Jean-Pierre Hubaux, Bradley A Malin, and XiaoFeng Wang. Privacy In The Genomic Era. *ACM Computing Surveys*, 48(1), 2015.

[128] NHS England. A&E Synthetic Data. https://data.england.nhs.uk/dataset/a-e-synthetic-data, 2021.

[129] NIH. Genomic Data Sharing. https://osp.od.nih.gov/scientific-sharing/genomic-data-sharing-faqs/, 2021.

[130] Kobbi Nissim, Thomas Steinke, Alexandra Wood, Micah Altman, Aaron Be-

mbenek, Mark Bun, Marco Gaboardi, David R O'Brien, and Salil Vadhan. Differential privacy: A primer for a non-technical audience. In *Privacy Law Scholars Conference*, 2017.

[131] William S Noble. What is a support vector machine? *Nature biotechnology*, 24(12):1565–1567, 2006.

[132] J. R. Norris. *Markov Chains*. Cambridge University Press, 1998.

[133] Bristena Oprisanu and Emiliano De Cristofaro. AnoniMME: bringing anonymity to the Matchmaker Exchange platform for rare disease gene discovery. *Bioinformatics*, 34(13), 2018.

[134] Bristena Oprisanu, Christophe Dessimoz, and Emiliano De Cristofaro. How much does genoguard really" guard"? an empirical analysis of long-term security for genomic data. In *Proceedings of the 18th ACM Workshop on Privacy in the Electronic Society*, pages 93–105, 2019.

[135] Bristena Oprisanu, Georgi Ganev, and Emiliano De Cristofaro. Measuring utility and privacy of synthetic genomic data. *arXiv preprint arXiv:2102.03314*, 2021.

[136] Bristena Oprisanu, Adria Gascon, and Emiliano De Cristofaro. Evaluating privacy-preserving generative models in the wild technical report.

[137] Rafail Ostrovsky and William E. Skeith. A Survey of Single-Database Private Information Retrieval: Techniques and Applications. In *Public Key Cryptography*, pages 393–411, 2007.

[138] Nicolas Papernot, Martín Abadi, Úlfar Erlingsson, Ian J. Goodfellow, and Kunal Talwar. Semi-supervised knowledge transfer for deep learning from private training data. In *International Conference on Learning Representations, ICLR*, 2017.

[139] Mijung Park, Jimmy Foulds, Kamalika Chaudhuri, and Max Welling. DP-EM: Differentially Private Expectation Maximization. *arXiv preprint arXiv:1605.06995*, 2016.

[140] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plau-*

*sible inference*. Elsevier, 2014.

[141] Personal Genome Project. https://www.personalgenomes.org/, 2018.

[142] Anthony A Philippakis et al. The Matchmaker Exchange: A platform for rare disease gene discovery. *Human mutation*, 36:915–921, 2015.

[143] Ron Milo Philips. What is the rate of recombination? http://book.bionumbers.org/what-is-the-rate-of-recombination/, 2018.

[144] Haoyue Ping, Julia Stoyanovich, and Bill Howe. Datasynthesizer: Privacy-preserving synthetic datasets. In *Proceedings of the 29th International Conference on Scientific and Statistical Database Management*, 2017.

[145] Haoyue Ping, Julia Stoyanovich, and Bill Howe. Datasynthesizer: Privacy-preserving synthetic datasets. In *Proceedings of the 29th International Conference on Scientific and Statistical Database Management*, 2017.

[146] Michal Piorkowski, Natasa Sarafijanovic-Djukic, and Matthias Grossglauser. CRAWDAD dataset epfl/mobility (v. 2009-02-24). https://crawdad.org/epfl/mobility/20090224/.

[147] Raluca Ada Popa, Andrew J Blumberg, Hari Balakrishnan, and Frank H Li. Privacy and accountability for location-based aggregate statistics. In *ACM CCS*, 2011.

[148] Apostolos Pyrgelis, Carmela Troncoso, and Emiliano De Cristofaro. Knock Knock, Who's There? Membership Inference on Aggregate Location Data. In *NDSS*, 2018.

[149] Genetics Home Reference. What are single nucleotide polymorphisms (SNPs)?

[150] Jerome P Reiter, Quanli Wang, and Biyuan Zhang. Bayesian estimation of disclosure risks for multiply imputed, synthetic data. *Journal of Privacy and Confidentiality*, 6(1), 2014.

[151] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR,

2014.

[152] Alan R Rogers and Chad Huff. Linkage disequilibrium between loci with unknown phase. *Genetics*, 182, 2009.

[153] Donald B Rubin. Statistical disclosure limitation. *Journal of official Statistics*, 9(2), 1993.

[154] Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, Yann Ollivier, and Hervé Jégou. White-box vs black-box: Bayes optimal strategies for membership inference. In *International Conference on Machine Learning*, pages 5558–5567, 2019.

[155] Pierre Sadrach. How Artificial Intelligence Can Revolutionize Healthcare. https://builtin.com/healthcare-technology/how-ai-revolutionize-healthcare, 2020.

[156] S. S. Samani, Z. Huang, E. Ayday, M. Elliot, J. Fellay, J. P. Hubaux, and Z. Kutalik. Quantifying Genomic Privacy via Inference Attack with High-Order SNV Correlations. In *IEEE Security and Privacy Workshops*, 2015.

[157] Gary Saunders, Michael Baudis, Regina Becker, Sergi Beltran, Christophe Béroud, Ewan Birney, Cath Brooksbank, Søren Brunak, Marc Van den Bulcke, Rachel Drysdale, et al. Leveraging european infrastructures to access 1 million human genomes by 2022. *Nature Reviews Genetics*, 2019.

[158] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *IEEE Symposium on Security and Privacy*, 2017.

[159] Suyash S. Shringarpure and Carlos D. Bustamante. Privacy Risks from Genomic Data-Sharing Beacons. *The American Journal of Human Genetics*, 97:631–646, 2015.

[160] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. arXiv:1312.6034, 2013.

[161] Nigel P. Smart, Vincent Rijmen, Benedikt Gierlichs, Kenneth G. Paterson,

Martijn Stam, Bogdan Warinschi, and Gaven Watson. Algorithms, Key Size and Parameters Report. https://www.enisa.europa.eu/publications/algorithms-key-size-and-parameters-report-2014/at_download/fullReport, 2014.

[162] Paul Smolensky. Information processing in dynamical systems: Foundations of harmony theory. Technical report, Colorado Univ at Boulder Dept of Computer Science, 1986.

[163] Fabio Soldo, Anh Le, and Athina Markopoulou. Predictive blacklisting as an implicit recommendation system. In *INFOCOM*, 2010.

[164] Theresa Stadler, Bristena Oprisanu, and Carmela Troncoso. Synthetic Data – A Privacy Mirage. arXiv:2011.07018, 2020.

[165] Paul Syverson, Roger Dingledine, and Nick Mathewson. Tor: The second-generation onion router. In *Usenix Security*, pages 303–320, 2004.

[166] Amalio Telenti, Erman Ayday, and Jean Pierre Hubaux. On genomics, kin, and privacy. *F1000Research*, 3, 2014.

[167] Texas Department of State Health Services, Austin, Texas. Texas Hospital Inpatient Discharge Public Use Data File 2013 Q1-Q4. https://www.dshs.texas.gov/THCIC/Hospitals/Download.shtm, 2013. Accessed 2020-06-01.

[168] Amirsina Torfi and Edward A Fox. CorGAN: Correlation-capturing convolutional generative adversarial networks for generating synthetic healthcare records. arXiv:2001.09346, 2020.

[169] Amirsina Torfi, Edward A Fox, and Chandan K Reddy. Differentially Private Synthetic Medical Data Generation using Convolutional GANs. *arXiv:2012.11774*, 2020.

[170] Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. Stealing Machine Learning Models via Prediction APIs. In *USENIX Security*, 2016.

[171] Allan Tucker, Zhenchen Wang, Ylenia Rotalinti, and Puja Myles. Generating high-fidelity synthetic patient data for assessing machine learning healthcare

software. *NPJ Digital Medicine*, 3(1), 2020.

[172] N Tyagi, J Wang, K Wen, and D Zuo. Honey Encryption Applications. 6.857. *Computer and Network Security, Massachusetts Institute of Technology*, 2015.

[173] Brianna Vendetti. 2018 Differential Privacy Synthetic Data Challenge. https://www.nist.gov/ctl/pscr/funding-opportunities/prizes-challenges/2018-differential-privacy-synthetic-data-challenge.

[174] Nora von Thenen, Erman Ayday, and A. Ercument Cicek. Re-Identification of Individuals in Genomic Data-Sharing Beacons via Allele Inference. *bioRxiv*, 2017.

[175] Binghui Wang and Neil Zhenqiang Gong. Stealing hyperparameters in machine learning. In *IEEE Symposium on Security and Privacy*, 2018.

[176] Ding Wang, Haibo Cheng, Ping Wang, Jeff Yan, and Xinyi Huang. A Security Analysis of Honeywords. In *NDSS*, 2018.

[177] Rui Wang, Yong Fuga Li, XiaoFeng Wang, Haixu Tang, and Xiaoyong Zhou. Learning your identity and disease from research papers: information leaks in genome wide association study. In *ACM Conference on Computer and Communications Security*, 2009.

[178] Shuang Wang, Xiaoqian Jiang, Siddharth Singh, Rebecca Marmor, Luca Bonomi, Dov Fox, Michelle Dow, and Lucila Ohno-Machado. Genome Privacy: Challenges, Technical Approaches to Mitigate Risk, and Ethical Considerations in the United States. *Annals of the New York Academy of Sciences*, 1387(1), 2017.

[179] Xiao Shaun Wang et al. Efficient Genome-Wide, Privacy-Preserving Similar Patient Query Based on Private Edit Distance. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 492–503, 2015.

[180] Zhenchen Wang, Puja Myles, and Allan Tucker. Generating and Evaluating Synthetic UK Primary Care Data: Preserving Data Utility & Patient Pri-

vacy. In *IEEE International Symposium on Computer-Based Medical Systems*, 2019.

[181] Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially private generative adversarial network. arXiv preprint arXiv:1802.06739, 2018.

[182] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional gan. In *Advances in Neural Information Processing Systems*, 2019.

[183] Yahoo Finance. Global Direct-to-Consumer Genetic Testing Market 2019-2023 — 16CAGR Projection Over the Next Five Years. https://uk.finance.yahoo.com/news/global-direct-consumer-genetic-testing-114500396.html, 2019.

[184] Andrew Yale, Saloni Dash, Ritik Dutta, Isabelle Guyon, Adrien Pavao, and Kristin Bennett. Privacy preserving synthetic health data. 2019.

[185] Burak Yelmen, Aurélien Decelle, Linda Ongaro, Davide Marnetto, Francesco Montinaro, Cyril Furtlehner, Luca Pagani, and Flora Jay. Creating Artificial Human Genomes Using Generative Models. https://www.biorxiv.org/content/10.1101/769091v2, 2019.

[186] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *IEEE Computer Security Foundations Symposium*, 2018.

[187] Jinsung Yoon, Lydia Drumright, and Mihaela Schaar. Anonymization Through Data Synthesis Using Generative Adversarial Networks (ADS-GAN). *IEEE Journal of Biomedical and Health Informatics*, PP, 2020.

[188] Kiyotsugu Yoshida and Yoshio Miki. Role of BRCA1 and BRCA2 as regulators of DNA repair, transcription, and cell cycle in response to DNA damage. *Cancer Science*, 95(11), 2004.

[189] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding Neural Networks Through Deep Visualization.

arXiv:1506.06579, 2015.

[190] Jun Zhang, Graham Cormode, Cecilia M. Procopiuc, Divesh Srivastava, and Xiaokui Xiao. PrivBayes: Private Data Release via Bayesian Networks. *ACM Transactions on Database Systems*, 2017.

[191] Jun Zhang, Graham Cormode, Cecilia M. Procopiuc, Divesh Srivastava, and Xiaokui Xiao. Privbayes: Private data release via bayesian networks. *ACM Trans. Database Syst.*, 2017.