Evaluating Privacy-Preserving Generative Models in the Wild
Technical Report

Bristena Oprisanu, Adria Gascon, Emiliano De Cristofaro

Abstract

The ability to share data among different entities can enable a number of compelling applications and analytics. However, useful datasets contain, more often than not, information of sensitive nature, which can endanger the privacy of users within the dataset. A possible alternative is to use machine learning algorithms known as generative models: these learn the data-generating distribution and allow to generate artificial data resembling the data they are trained on. In other words, entities can train and publish the model, instead of the original data; however, access to such model could still allow an adversary to reconstruct (possibly sensitive) training data.

To overcome these issues, researchers have recently proposed a number of techniques that use carefully crafted random noise to provide strong privacy guarantees – specifically, guaranteeing Differential Privacy – so that the privacy leakage of synthetic data generation can be quantified by rigorous, statistical means. A common approach in several of the proposed techniques involves training a generative machine learning model using a differentially private version of stochatic gradient descent, the so-called *moments accountant* method (Abadi et al. 2016).

In this project, 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 an privacy for private data synthesis are achievable in specific settings. The most encouraging results correspond to image and financial data as, for good privacy parameters, synthetic training datasets led to 5-8% accuracy loss with respect to training on the *non-private* original data, for simple linear models.

1 Introduction

Every day over 2.5 quintillion bytes of data are created [22]; 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 results, 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, recently 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 [6], 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) [28].

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" [15] is guaranteed—we describe differential privacy in Section 3. 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.

Privacy-Preserving Generative Models. 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.

Although these approaches have been evaluated in research papers for certain use cases, we set to investigate their real-world usability and limitations. Specifically, we select four models: 1) PrivBayes [33], an ϵ-differentially private algorithm, which constructs a generative model based on Bayesian networks, 2) DP-SYN [3] and 3) PrivVAE [4], which use differentially private generative models relying on the moments accountant [2], and 4) SynLoc [7], a seed-based generative model for location data, building on the concept of plausible deniability.

Overview of the Results

Our experimental evaluation is performed on five datasets, grouped into four categories: financial data, images, cyber threat logs, and location data. All the code and data con-

sidered in this report is available in a github repository, and easily deployable by means of a Docker container, as part of the project deliverables.

Financial data. We use two datasets from the UCI Machine Learning repository [12] and evaluate accuracy for classification tasks on the synthetic data (e.g., classifying individuals as having good/bad credit risks, or income above/below $50K). We find that PrivBayes performs best under a high-privacy regime for large datasets, but overall synthetic data generated from smaller datasets yields poor accuracy, even lower than random guessing. Overall, DP-SYN reaches the highest accuracy, but accuracy does not improve with less noise. Furthermore, even though the synthetic data generated by Priv-VAE achieves similar accuracy as DP-SYN, it performs poorly on datasets with imbalanced classes.

Images. For images, we use the MNIST datasets of handwritten digits [20], evaluating the performance of digit recognition, DP-SYN achieves highest accuracy for classification, and it also seems to generate clearer samples – from a human evaluation perspective.

Network activity logs. We evaluate the data synthesis on cyber threat logs obtained from Dshield [1] and find it to be the most sensitive to perturbation, to the point that no model is able to classify or predict threats with better accuracy than random guessing. However, we do not rule out that a dataset of this nature richer than Dshield could yield better results.

Location data. Finally, for location data, we use the San Francisco cabs dataset [26], evaluating performance on a clustering task, which can be used for detecting areas of interest for locations. We confirm that the model obtaining the most similar distribution of clusters to the original data is Syn-Loc, which specializes on this particular task.

Moments Accountant. Several privacy-preserving generative models are instantiations of the moments accountant method [2]. This technique allows to keep track of the overall privacy budget throughout a differentially private stochastic gradient descent optimization. The moments accountant offers a concrete regime of privacy guarantees that involves a complex dependency between the parameters of the gradient descent procedure, the available privacy budget, and the size of the input dataset. We provide a description of the technique and show empirically what range of privacy guarantees it can provide. Our goal is to shed light on the relationship between the moments accountant and the synthetic data generation techniques that rely on it, a point often ommitted in the description of approaches that use differentially private gradient descent as a subprocedure.

Report Organization

In Section 2, we review background information on privacy-preserving data synthesis, then, in Section 3, we provide an overview of differential privacy, followed by a discussion on the moments accountant in Section 4. We then present the models used in our evaluation in Section 5, while, in Section 6, the testing methodology and the experimental results results, followed by a discussion in Section 7.

2 Privacy-Preserving Data Release

The need to reconcile the need to release data and that to protect sensitive information has encouraged researchers to consider different approaches; we review them in this section.

2.1 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. [6] 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. [16] 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 [5].

Aggregation. Another approach is to share aggregate statistics about a dataset. For example, one can 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 [27]. However, this is also ineffective, due to susceptibility to membership inference attacks. For instance, Pyrgelis et al. [28] 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 [19, 32].

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 3, 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 [11].

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 pri-

vate 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 [29]. 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 [10], 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 [33]. 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 [7], 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 Differential Privacy

In this section, we discuss Differential Privacy (DP) as well as the Moments Accountant method presented in [2] in the context of privacy-preserving deep learning. Readers familiar with these concepts can skip this section without loss of continuity.

3.1 Definitions and Properties

DP addresses the paradox of learning nothing about an individual while learning useful information about a population [15]. 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 ϵ, δ, a randomized algorithm $\mathcal{A}$ satisfies (ϵ, δ)-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 $S \subseteq Range(\mathcal{A})$, the following formula holds:

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

The ϵ, δ parameters. Differential privacy analyses allow for some information leakage specific to individual data subjects, controlled by the privacy parameter ϵ. This measures the effect on each individual's information on the output of the analysis. With smaller values of ϵ, 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 [24].

If $\delta = 0$, we say that the mechanism is ϵ-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 [15], the values of δ are usually computed as an inverse function of a polynomial in the size of the dataset. In particular, any values of δ 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 δ 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 ϵ, δ, with $\epsilon = \infty$ 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 $\mathcal{A}$ be a randomized algorithm that is (ϵ, δ)-differentially private, and f be an arbitrary mapping. Then, $f \circ \mathcal{A}$ is also (ϵ, δ)-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 $\mathcal{M}_1, \ldots \mathcal{M}_n$, where each $\mathcal{M}_i$ is a (ϵ_i, δ_i)- differentially private algorithm, we have that $\mathcal{M}_{[n]}(x) = (\mathcal{M}_1(x), \ldots \mathcal{M}_n(x))$ is $(\sum_{i=1}^{n} \epsilon_i, \sum_{i=1}^{n} \delta_i)$-diffentially private.

Strong Composition. Dinur and Nissim [13] and Dwork and Nissim [14] showed that, under k-fold adaptive composition on a single database, the privacy parameter deteriorates less if a negligible loss in δ can be tolerated. This yields the Strong Composition Theorem:

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

$$\epsilon' = \sqrt{2k \ln \tfrac{1}{\delta'}} \cdot \epsilon + k \cdot \epsilon\epsilon_0,$$

where $\epsilon_0 = e^\epsilon - 1$. 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 (ϵ, δ)-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 $\sigma = S_2(F)\sqrt{2\ln(2/\delta)}/\epsilon$, and $\mathcal{N}(0, \sigma^2)$ are independent and identically distributed Gaussian random variables, the mechanism $\mathcal{M}$ provides (ϵ, δ)-differential privacy when:

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

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

Differentially Private Data Generation. As discussed earlier, the main focus of our work is differentially private synthetic data generation. The main idea is that, 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) has recently launched a differential privacy synthetic data challenge [31], aiming to find synthetic data generation algorithms that protect individual privacy but provide a high utility of the overall dataset.

3.2 Moments Accountant

In [2], 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 θ by minimizing the loss function $\mathcal{L}(\theta)$; 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, [2] provides a stronger accounting method, namely the *moments accountant*, which saves a factor $\sqrt{\frac{T}{\delta}}$ in the asymptotic bound.

Algorithm 1 Differentially Private SGD

1: **Input:** Examples $\{x_1, \ldots, x_N\}$, loss function $\mathcal{L}(\theta = \frac{1}{N}\sum_i \mathcal{L}(\theta, x_i)$. Parameters: learning rate η_t, noise scale σ, group size L, gradient norm bound C, number of steps T.
2: Initialize θ_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: $g_t(x_i) = \nabla_{\theta_t}\mathcal{L}(\theta_t, x_i)$
7: $\overline{g_t}(x_i) = \frac{g_t(x_i)}{\max(1, \frac{||g_t(x_i)||_2}{C})}$
8: $\widetilde{g_t} = \frac{1}{L}(\sum_i \overline{g_t}(x_i) + \mathcal{N}(0, \sigma^2 C^2 I))$
9: $\theta_{t+1} = \theta_t - \eta_t \widetilde{g_t}$
return θ_T and compute the overall privacy cost (ϵ, δ) using a privacy accounting method.

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 [2]. For proofs and details we reader to the supplementary material of [2].

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

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

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

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

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

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

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

Then, α achieves the following properties:

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

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

Algorithm 2 Epsilon Computation

Input q, σ, δ, number of epochs ep, number of orders λ
Initialize $l = -\infty$, $\alpha_{list}, \epsilon_{list} = \emptyset$
for $i = 2$ to $\lambda + 1$ **do**
    for $j = 2$ to $1 + 1$ **do**
        $l_1 = \log(\frac{i!}{j!(i-j)!} + j \cdot \log q + (i-j)\log(1-q)$
        $s = l_1 + (j^2 - j)/(s\sigma^2)$
        if l is $-\infty$ **then**
            $l = s$
        else:
            $l = \log(e^{\log(l-s)}) + s$
    $\alpha = l \cdot ep \cdot q$
    Append α to α_{list}
$i = 2$
for each α in α_{list} **do**:
    Append $(\alpha - \log(\delta))/(i-1)$ to ϵ_{list}
    $i = i + 1$
$\epsilon = \min(\epsilon_{list})$
return ϵ

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

- *Tail Bound:* For any $\epsilon > 0$, the mechanism $\mathcal{M}$ is (ϵ, δ)-differentially private for

$$\delta = \min_\lambda \exp(\alpha_{\mathcal{M}}(\lambda) - \lambda\epsilon)$$

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

$$\epsilon = \min_\lambda \frac{\alpha_{\mathcal{M}}(\lambda) - \log\delta}{\lambda}$$

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 $\mathcal{N}(0, \sigma^2)$ and $\mathcal{N}(1, \sigma^2)$, denoted as μ_0 and μ_1 respectively. If $\mu = (1-q)\mu_0 + q\mu_1$, we have $\alpha(\lambda) = \log\max(E_1, E_2)$, where:

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

4 A Discussion of the the Moments Accountant Method

In Section 3.2, we have defined the moments accountant method proposed in [2]. Here, we discuss and study its 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 ϵ, given δ, the sampling ratio q, the noise multiplier σ, the number of epochs ep, and the number of orders for the moment accountant λ are shown in Algorithm 2. We evaluate the effect of different parameters on the minimum value of ϵ, 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 [2]. Please note that Abadi et al.'s original analysis aimed to find the best possible (ϵ, δ) 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.

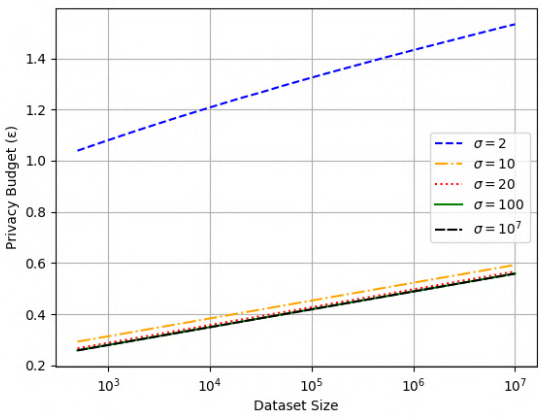

Figure 1: The minimum value of ϵ, calculated by the moments accountant, for different dataset sizes, with $\delta = \frac{1}{10*n}$, where n is the dataset size, for 20 training epochs, with sampling ratio $q = 0.01$.

Effect of the noise multiplier. We begin by evaluating the moments accountant with respect to σ, 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 1, we observe that with $\sigma = 2$, the minimum required value for the privacy budget is outside the so-called privacy regime ($\epsilon < 1$) even for the smallest dataset tested ($n = 500$). However, when looking at higher values of σ 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 2 illustrates the corresponding value for the moments α. The correlation between the values of α and the minimum values for ϵ is remarkable, as α quickly increases for $\sigma = 2$, and for higher values of σ there is little to no variation between the values of α.

Effect of δ. We then evaluate the effect of δ on the overall privacy budget. Recall from Section 3.1 that δ allows for a small probability of failure in the differential privacy definition, i.e. with a probability of δ 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. Hence $\delta \cdot n$ needs to be small, as to not let any users at risk, δ needs to be chosen based on the size of the dataset.

Given that ϵ is computed as a function of δ in the moments accountant, the privacy budget is highly dependent

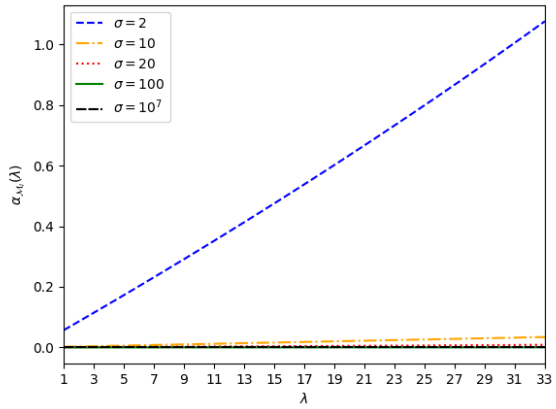

Figure 2: The values of the moments accountant $\alpha_{\mathcal{M}_i}(\lambda)$, for the minimum values of ϵ, with a noise multiplier of $\sigma = 20$.

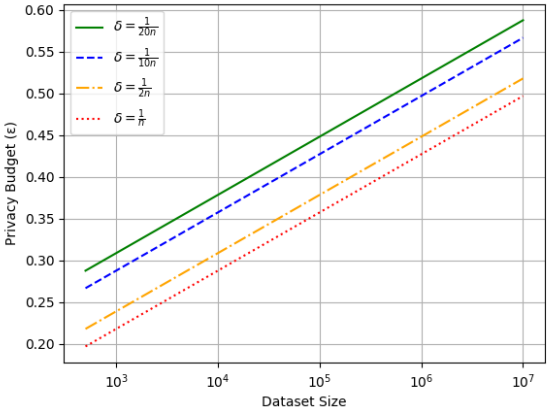

Figure 3: The minimum value of ϵ, calculated by the moments accountant, for different dataset sizes, where n is the dataset size, for 20 training epochs, with the sampling ratio $q = 0.01$, and the noise multiplier $\sigma = 20$.

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 3, we observe that the minimum value for ϵ increases with lower values for δ. We also illustrate the corresponding values of the moments, α, corresponding to the minimum values of ϵ over 32 moments in Figure 4. These values are independent of the dataset size, hence independent from δ.

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 5, we keep the δ parameter fixed at $\frac{1}{10 \cdot n}$, the noise multiplier $\sigma = 20$, and plot the minimum value of ϵ 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 the

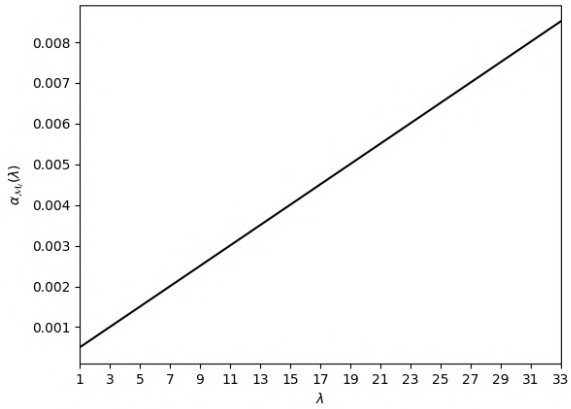

Figure 4: The values of the moments accountant $\alpha_{\mathcal{M}_i}(\lambda)$, for the minimum values of ϵ, with a noise multiplier $\sigma = 20$ and sampling ratio $q = 0.01$.

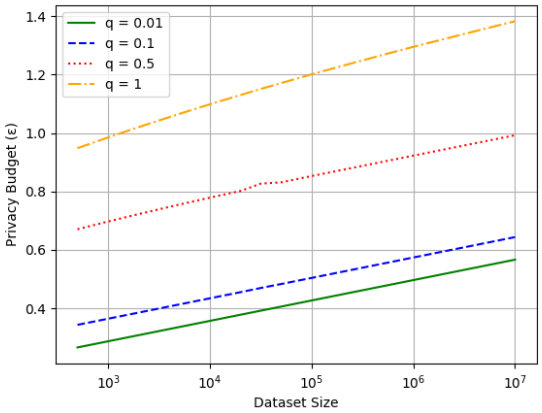

Figure 5: The minimum value of ϵ, calculated by the moments accountant, for different dataset sizes, with $\delta = \frac{1}{10*n}$, where n is the dataset size, for 20 training epochs, with noise multiplier $\sigma = 20$.

sampling ratio also affects the value of the moments accountant, as clear from Figure 6. For a small sampling ratio (i.e. $q = 0.01$ to $q = 0.1$), we find that the values of the moments α are close to 0, however, with increasing batch size, these values increase quickly, making the minimum value of the privacy parameter ϵ close to 1 even for small dataset sizes ($n \leq 10^3$).

Remarks. Overall, our analysis shows 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.

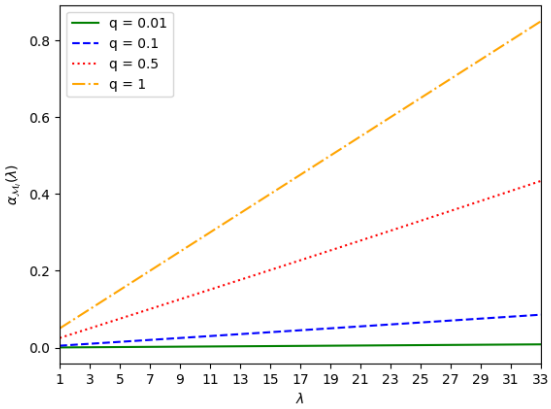

Figure 6: The values of the moments accountant $\alpha_{\mathcal{M}_i}(\lambda)$, for the minimum values of ϵ, with a noise multiplier of 20.

5 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.

5.1 PrivBayes

PrivBayes [33] is a solution for releasing a high-dimensional dataset D in an ϵ-differentially private manner. It involves three phrases:

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

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

3. The Bayesian network $\mathcal{N}$ 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 $\mathcal{N}$ on a dataset D, containing a set of $\mathcal{A}$ attributes. The mutual information between two variable is denoted as :

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

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

The proposed non-private algorithm "GreedyBayes" – see Algorithm 3 – extends the Chow and Liu algorithm [9]

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, Π_i) from Ω with maximal mutual information $I(X_i, \Pi_i)$
8: Add (X_i, Π_i) to $\mathcal{N}$; add X_i to V
return $\mathcal{N}$

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 Π is calculated. After that, an AP pair is sampled from Ω such that the sampling probability of any pair (X, Π) is proportional to $exp(\frac{I(X,\Pi)}{2\Delta})$, where Δ is the scaling factor. In order for the Bayesian network to satisfy $\frac{\epsilon}{2}$-differential privacy, Δ is set to

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

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, Π) 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, Π), a maximum joint distribution $\Pr^\diamond[X, \Pi]$ for X and Π is one that maximizes the mutual information between X and Π. 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, Π) be an AP pair and $\mathcal{P}^\diamond[X, \Pi]$ be the set of all maximum joint distributions for X and Π. Then F is defined as:

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

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, Π) be an AP pair and $|\Pi| = k$. Then, the joint distribution $\Pr[X, \Pi]$ can be represented as a 2×2^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 $\mathcal{P}^\diamond$ 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 ϵ-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 https://github.com/JiaMingLin/privbayes.

5.2 Approaches based on Generative Models and the Moments Accountant

5.2.1 Priv-VAE

In [4], 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 [21]). Then, to training 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 thy 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 [17].

Finally, note that the implementation of this approach is available upon request from the authors of [4].

5.2.2 DP-SYN

Abay et al. [3] present a framework combining private convolutional neural network with the private version of the iterative expectation maximization algorithm Dp-Em [25]. The dataset is first partitioned according to every instance's label, and then an $(\frac{\epsilon}{2}, \frac{\delta}{2})$-differentially private autoencoder is built for each label group. The private latent representation is then injected into an $(\frac{\epsilon}{2}, \frac{\delta}{2})$-differentially private expectation maximization function (Dp-Em [25]). 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 [33] and Priv-VAE [4]. 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.

5.3 Synthesis of Location Traces (Syn-Loc)

Finally, we look at the work by Bindschaedler and Shokri [7], which aims to generate fake, yet semantically plausible privacy-preserving location traces.

5.3.1 Plausible Deniabilty

Bindschaedler et al. [8] 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 models M such that $y = M(d_1)$ for $d_1 \in D$, we state that y is releaseable with (k, γ) -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 if the that 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$(\frac{1}{\epsilon_0})$ is added to k, then, their system satisfies (ϵ, δ) differential privacy, with $\epsilon = \epsilon_0 + \log \frac{\gamma}{t}$ and $\delta = e^{-\epsilon_0(k-t)}$, 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.

5.3.2 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 properties of the original data.

The protocol was evaluated on the Nokia Lausanne Dataset [26], 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 Experimental Evaluation

6.1 Datasets & Tasks

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 [12], 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 [20], a public image dataset, which includes 28×28-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 [1] 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 [30], i.e., forecasting which IPs will attack a target.

4. *Location data.* We use the San Francisco cabs dataset [26], containing mobility traces recorded by San Francisco taxis. This has often been used to predict next locations, identifying points of interests, etc. [28].

6.2 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 basis. For each of the datasets, we test multiple values for ϵ, while keeping δ 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. [23] 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 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.

6.3 Financial Data

German Credit. In order to evaluate the categorical attributes of the German Credit dataset, we use the numeric encoding provided in [12]. 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 7. The accuracy of the models improves with less noise added to the model (i.e. higher values for ϵ), and, among the tested models, DP-SYN 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

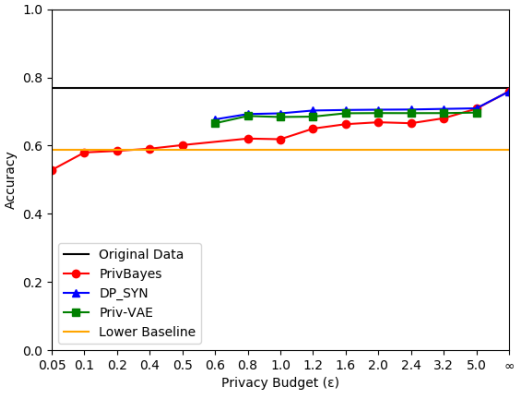

Figure 7: SVM accuracy results for German Credit Dataset with δ fixed at $\frac{1}{10 \cdot n}$, for varying values of ϵ.

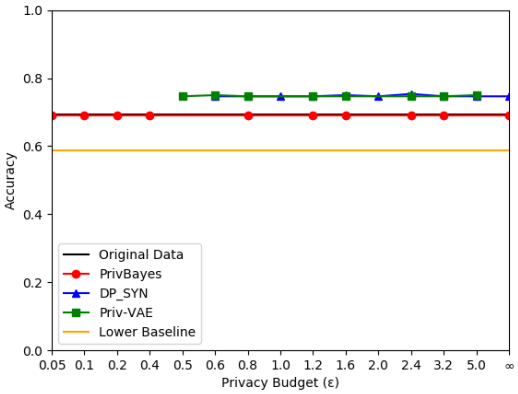

Figure 8: Linear regression accuracy results for German Credit Dataset with δ fixed at $\frac{1}{10 \cdot n}$, for varying values of ϵ.

within one class (in this case it classifies most customers as having bad credit score). For $\epsilon \leq 0.5$, the synthetic data obtained from PrivBayes reports less accuracy than our lower baseline.

In Figure 7, we also report a setting for $\epsilon = \infty$. 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 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.

Adult Dataset. We first encode the adult dataset, using One-Hot Encoding [18], 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 Fig-

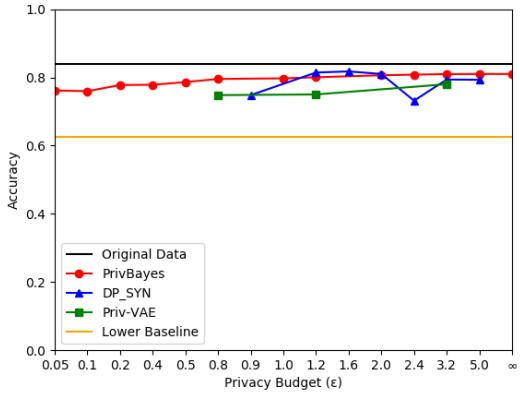

Figure 9: SVM accuracy results for Adult Dataset with δ fixed at $\frac{1}{10 \cdot n}$, for varying values of ϵ.

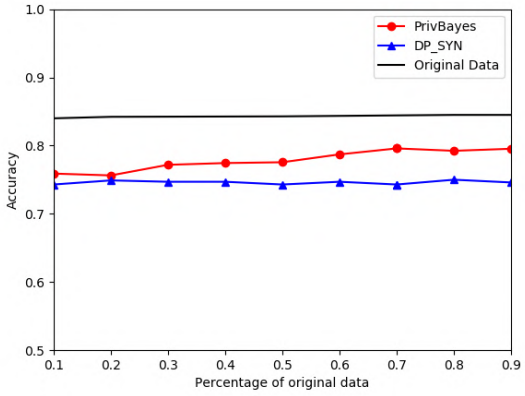

Figure 10: SVM accuracy results, for training models at a percentage of the original datasets. For DP-SYN, we have $\epsilon = 0.9$ (the minimum value for ϵ for this dataset, when $\delta = \frac{1}{10 \cdot n}$), and for PrivBayes $\epsilon = 0.8$.

ure 9. Note that PrivBayes has better accuracy with higher noise levels than DP-SYN ($\epsilon \leq 1$). Additionally, when ϵ 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 ϵ.

To observe if an increase in accuracy can be correlated with with increasing dataset size, we also train the SVM model on partial data, while keeping the same test dataset. In Figure 10, we plot the accuracy for $\epsilon = 0.9$ for DP-SYN (the minimum accepted value of ϵ for this dataset, when $\delta = \frac{1}{10 \cdot n}$), and $\epsilon = 0.8$ 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 11, we increase the value of ϵ 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 10, in this case we observe an improvement in accuracy for both models with increasing dataset sizes.

In Figure 12, we report the accuracy for increasing dataset sizes for $\epsilon = 3.2$. In this case DP-SYN outperforms PrivBayes, however, neither of the models' improvement in accuracy can be correlated with increasing dataset sizes.

6.4 Images

We then evaluate the models on the MNIST dataset. We use the the usual split for training and testing purposes, i.e. 60,000 samples for training and 10,000 samples for testing.

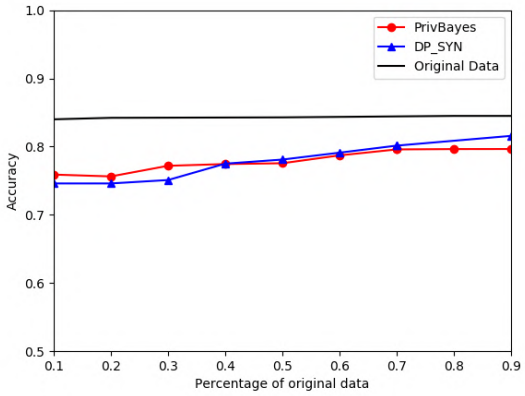

Figure 11: SVM accuracy results for training models at a percentage of the original datasets, with $\epsilon = 1.2$.

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 ϵ 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 13, all the classes seem similar, even in the lowest privacy case (i.e. $\epsilon = 10^7$). Therefore, we split the data into separate classes and trained on each class separately for generating the synthetic data.

In Figure 14, after splitting, with increasing values of epsilon, we start to distinguish between different classes. For DP-SYN (see Figure 15), the reconstruction of each of the digit classes is much clear even for small ϵ, and close to the average class reconstruction for the original data. Priv-VAE (see 16) outputs a less noisier reconstruction than PrivBayes, but not as clear as DP-SYN. Finally, in Figure 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 regres-

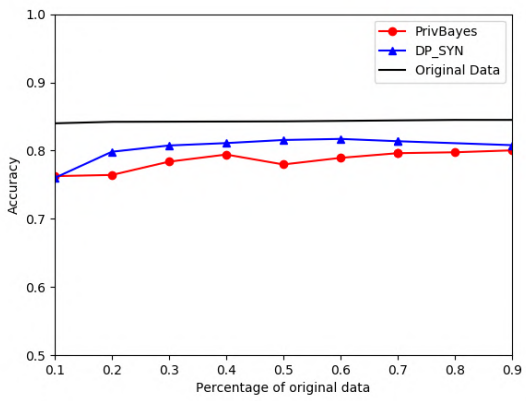

Figure 12: SVM accuracy results for training models at a percentage of the original datasets, with $\epsilon = 3.2$.

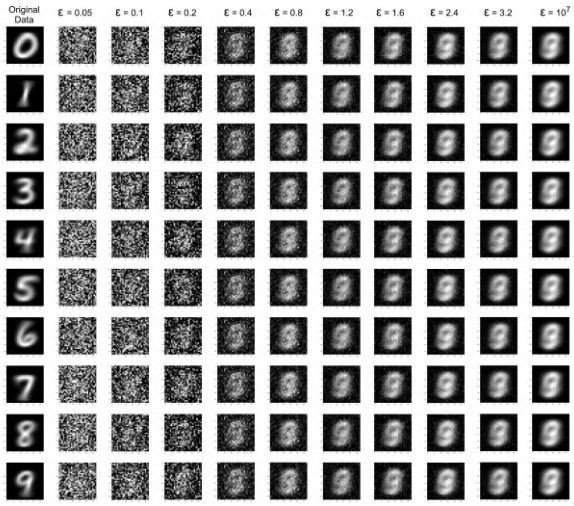

Figure 13: Average class reconstruction for PrivBayes, with no splitting before training.

sion 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 generates has a better accuracy than PrivBayes as the value of ϵ 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.

6.5 Cyber Threat Logs

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 of them achieved an 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

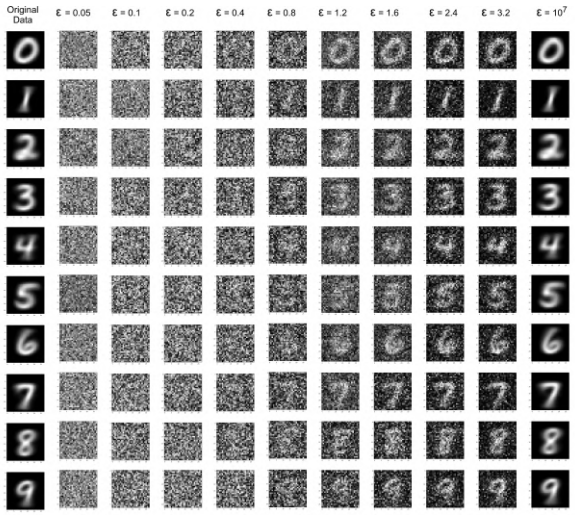

Figure 14: Average class reconstruction for PrivBayes, with splitting before training.

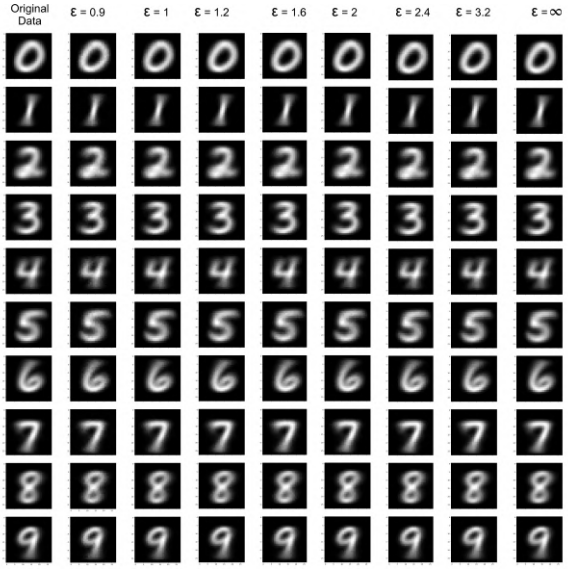

Figure 15: Average class reconstruction for DP-SYN.

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 [23], it failed to produce any samples that would correlate with the testing data. Hence, we 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 model.

6.6 Location Data

We generate the synthetic datasets for each of the models and plot the distribution of locations (Figure 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

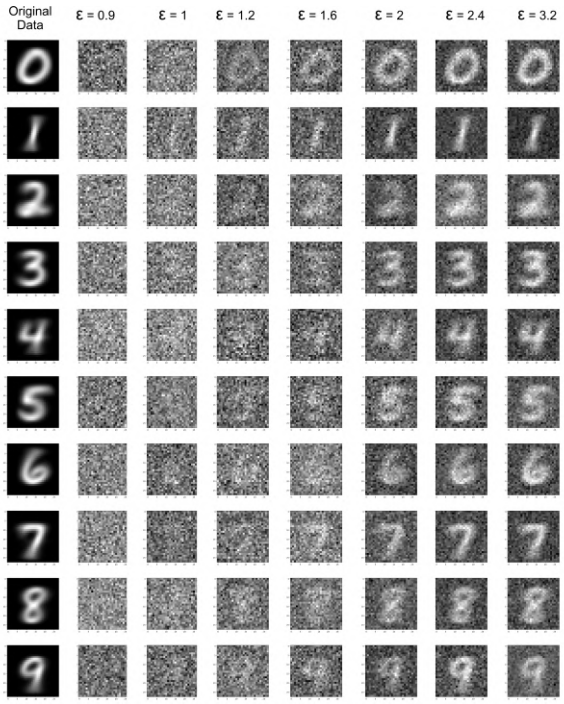

Figure 16: Average class reconstruction for Priv-VAE.

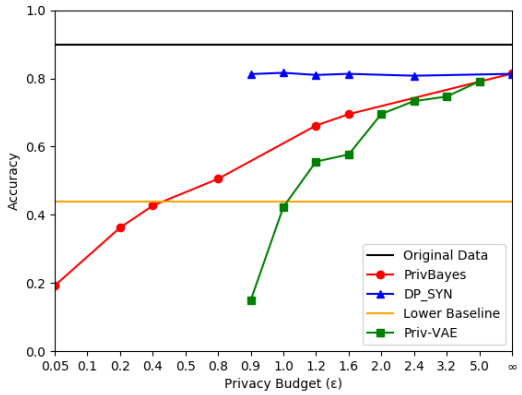

Figure 17: Linear Regression accuracy results for MNIST Dataset with δ fixed at $\frac{1}{10 \cdot n}$, for varying values of ϵ.

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 19. As expected, the synthetic data generated from DP-SYN is grouped within a single 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 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.

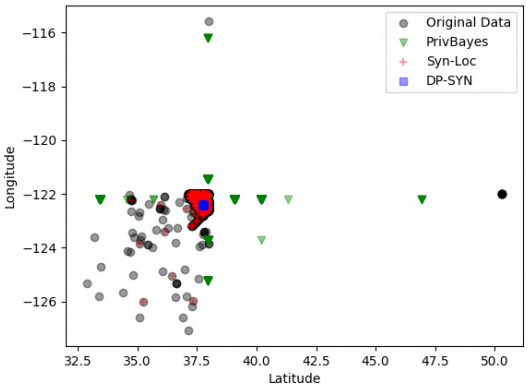

Figure 18: Distribution of locations for the San Francisco Cabs dataset.

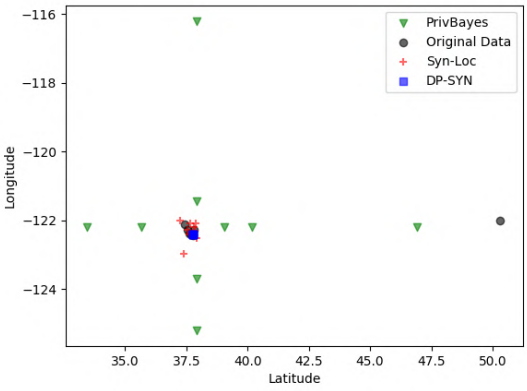

Figure 19: 10 Clusters of locations for the San Francisco Cabs dataset.

7 Discussion

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

Our analysis also showed that one of the most important notions to be thoroughly understood before attempting to generate differentially private synthetic datasets is the moments accountant [2]. This represents a complex accounting method for keeping track of the privacy budget where each parameter can have a great influence on the quality of the synthetic data.

We evaluated three of the privacy-preserving synthetic data generation models on binary classification tasks on two financial data datasets. The first model, PrivBayes [33], reported accuracy lower than randomly as-

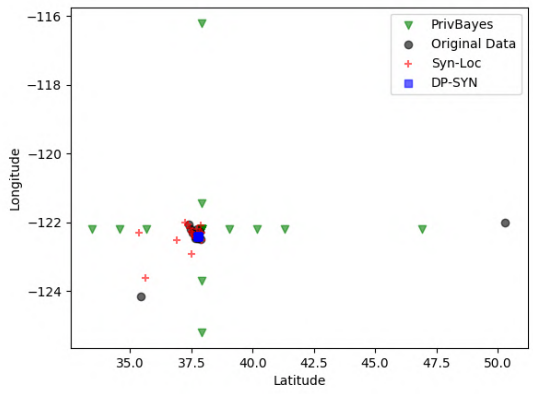

Figure 20: 20 Clusters of locations for the San Francisco Cabs dataset.

signing labels based on original distributions of the data when trained on noisy synthetic data (i.e., $\epsilon \leq 0.5$) 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 ($\epsilon < 1$). In fact, the average performance actually decreases for some of the less noisy synthetic datasets ($\epsilon > 2$). Priv-VAE generates synthetic data which even though reports testing accuracy comparable to DP-SYN, but, from closer inspection, it classifies most datapoints within 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 when evaluated for 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 $\epsilon = 0.9$ 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 reports 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 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 model used in this model. DP-SYN fails to generate meaningful synthetic data, generating 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.

In conclusion, our 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. It remains for the data provider to decide if the offset in utility associated with privacy-preserving synthetic data satisfies its needs. Overall, there is no "best" model among the ones we tested, as they all exhibit different performances on different datasets. 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. However, privacy-preserving synthetic data, even though not perfect, offers a better alternative then anonymization techniques which fail to provide useful privacy guarantees, or differential privacy which can greatly affect the utility of the data.

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

These results are encouraging, and leave open several venues for further work that could potentially lead to better trade-offs between privacy and utility, as one can (i) consider synthesis procedures that are more specialized to concrete types of data, (ii) evaluation under more powerful domain-specific models, and (iii) consider large datasets, which is a realistic consideration given that the datasets we covered in our evaluation are of moderate size.

References

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

[2] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang. Deep Learning with Differential Privacy. In *ACM CCS*, 2016.

[3] N. C. Abay, Y. Zhou, M. Kantarcioglu, B. Thuraisingham, and L. Sweeney. Privacy preserving synthetic data release using deep learning. In *ECML PKDD*, 2018.

[4] G. Acs, L. Melis, C. Castelluccia, and E. De Cristofaro. Differentially private mixture of generative neural networks. *IEEE Transactions on Knowledge and Data Engineering*, 2018.

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

[6] M. Archie, S. Gershon, A. Katcoff, and A. Zeng. De-anonymization of Netflix Reviews using Amazon Reviews. https://courses.csail.mit.edu/6.857/2018/project/Archie-Gershon-Katchoff-Zeng-Netflix.pdf.

[7] V. Bindschaedler and R. Shokri. Synthesizing plausible privacy-preserving location traces. In *IEEE Symposium on Security and Privacy*, 2016.

[8] V. Bindschaedler, R. Shokri, and C. A. Gunter. Plausible deniability for privacy-preserving data synthesis. *VLDB*, 2017.

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

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

[11] F. K. Dankar and K. El Emam. The Application of Differential Privacy to Health Data. In *Joint EDBT/ICDT Workshops*, 2012.

[12] D. Dheeru and E. Karra Taniskidou. UCI Machine Learning Repository. http://mlr.cs.umass.edu/ml/datasets.html, 2017.

[13] I. Dinur and K. Nissim. Revealing information while preserving privacy. In *ACM PODS*, 2003.

[14] C. Dwork and K. Nissim. Privacy-preserving datamining on vertically partitioned databases. In *IACR CRYPTO*, 2004.

[15] C. Dwork and A. Roth. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends in Theoretical Computer Science*, 9, 2013.

[16] M. Gymrek, A. L. McGuire, D. Golan, E. Halperin, and Y. Erlich. Identifying personal genomes by surname inference. *Science*, 2013.

[17] M. Hardt, K. Ligett, and F. McSherry. A simple and practical algorithm for differentially private data release. In *NeurIPS*, 2012.

[18] D. Harris and S. Harris. *Digital Design and Computer Architecture*. Morgan Kaufmann, 2010.

[19] N. Homer, S. Szelinger, M. Redman, D. Duggan, W. Tembe, J. Muehling, J. V. Pearson, D. A. Stephan, S. F. Nelson, and D. W. Craig. Resolving Individuals Contributing Trace Amounts of DNA to Highly Complex Mixtures Using High-Density SNP Genotyping Microarrays. *PLoS Genetics*, 2008.

[20] Y. LeCun, C. Cortes, and C. Burges. MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/.

[21] L. v. d. Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of machine learning research*, 2008.

[22] B. Marr. Forbes – How Much Data Do We Create Every Day? https://bit.ly/2Iz6Hbv, 2018.

[23] L. Melis, A. Pyrgelis, and E. De Cristofaro. On collaborative predictive blacklisting. *ACM SIGCOMM CCR*, 2019.

[24] K. Nissim, T. Steinke, A. Wood, M. Altman, A. Bembenek, M. Bun, M. Gaboardi, D. R. OBrien, and S. Vadhan. Differential privacy: A primer for a non-technical audience. In *Privacy Law Scholars Conference*, 2017.

[25] M. Park, J. Foulds, K. Chaudhuri, and M. Welling. DP-EM: Differentially Private Expectation Maximization. *arXiv preprint arXiv:1605.06995*, 2016.

[26] M. Piorkowski, N. Sarafijanovic-Djukic, and M. Grossglauser. CRAWDAD dataset epfl/mobility (v. 2009-02-24). https://crawdad.org/epfl/mobility/20090224/.

[27] R. A. Popa, A. J. Blumberg, H. Balakrishnan, and F. H. Li. Privacy and accountability for location-based aggregate statistics. In *ACM CCS*, 2011.

[28] A. Pyrgelis, C. Troncoso, and E. De Cristofaro. Knock Knock, Who's There? Membership Inference on Aggregate Location Data. In *NDSS*, 2018.

[29] D. B. Rubin. Statistical disclosure limitation. *Journal of official Statistics*, 1993.

[30] F. Soldo, A. Le, and A. Markopoulou. Predictive blacklisting as an implicit recommendation system. In *INFOCOM*, 2010.

[31] B. Vendetti. 2018 Differential Privacy Synthetic Data Challenge. https://www.nist.gov/ctl/pscr/funding-opportunities/prizes-challenges/2018-differential-privacy-synthetic-data-challenge.

[32] R. Wang, Y. F. Li, X. Wang, H. Tang, and X. Zhou. Learning Your Identity and Disease from Research Papers: Information Leaks in Genome Wide Association Study. In *ACM CCS*, 2009.

[33] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao. PrivBayes: Private Data Release via Bayesian Networks. *ACM Transactions on Database Systems*, 2017.