PrivSyn: Differentially Private Data Synthesis

Report: Sukrit Jindal (22125037, O10)

PrivSyn (Authors of the Original Paper) : Zhikun Zhang, Tianhao Wang, Ninghui Li, Jean Honorio, Michael Backes, Shibo He, Jiming Chen, Yang Zhang

***Abstract*—Differential Privacy is an approach which aims to ensure privacy for individuals while preserving the pattern within the collective sample. One technique in DP is to generate synthetic datasets that capture the information of the dataset without compromising the privacy. These methods ensure certifiable differential privacy without using other data processing techniques or affecting the AI/ML model being applied on the dataset. PrivSyn, as introduced by Zhang, Zhikun, et al., is one such technique which improves the efficiency of previous synthetic data generation techniques since it is able to guarantee differential privacy even with large datasets, in terms of their domain size, attributes and number of records.**

***Index Terms*—Differential Privacy, Synthetic Data Generation, Information Security, Algorithms**

I. Introduction

As big data becomes more and more prevalent, more users are concerned for their privacy. Further, AI models are often trained on data without checking for the privacy of the individuals whose data is being used. As a result, there is a need to formalise the notion of privacy. This is done using differential privacy (DP). One DP technique is to create a synthetic dataset based on another dataset, which captures the patterns and informations of the original dataset. This allows one to train models on the synthetic dataset without altering them. Generally, most DP techniques used for synthetic data generation so far have used graphical models. Graph models are based on probabilistic graph models and they aim to capture the joint distribution of the entire data in terms of marginals of attribute sets. However, the data in the graph is generally sparse, while the marginals grow exponentially with each output. As a result, these models are not only inefficient in storage, but also in computation as the dataset grows in size. PrivSyn, on the other hand aims to capture general trends in the data using just one-way and two-way marginals. Even within the two-way marginals, it applies a greedy selection to choose only some of those two-way marginals under a privacy budget, trying to achieve a balance for the tradeoff between dependency and noise errors which arise due to using a lesser or larger number of marginals, respectively. They also choose, based on those selected two-way marginals, certain multi-way marginals but this ultimately results in a lower computational overhead as even then, not all multi-way combinations are selected. They back their claims by relying on certain theorems and results which will be discussed in this paper. Thereafter, they introduce a marginal combination method which gradually converges to the full joint distribution of the original dataset using a method they term as the gradual update method (GUM). This part of the paper is not related to differential privacy but is introduced as an efficient algorithm for synthetic data generation using marginals.

II. Mathematical Background

Differential Privacy is a mathematical codification of the sense of the privacy, i.e., intuitively, the DP notion requires that any single element in a dataset has only a limited impact on the output or that no specific record can be identified from the dataset alone. There are many mathematical notions (or definitions) of differential privacy used across literature.

(ϵ, δ)-Differential Privacy

For an algorithm $\mathscr{A}$ operating on a dataset D, (ϵ, δ) -Differential Privacy [5] is defined as,
$\forall T \subseteq \text{Range}(\mathscr{A}) : \Pr\left[\mathscr{A}(D) \in T\right] \leq e^{\epsilon} \Pr\left[\mathscr{A}(D') \in T\right] + \delta$
where $D' \simeq D$, $\epsilon > 0$ and $\delta \geq 0$

ρ-Zero Concentrated Differential Privacy

An algorithm $\mathscr{A}$ operating on a dataset D satisfies ρ-zero concentrated differential privacy [4] if for $D' \simeq D$
$\forall \alpha \in (1,\infty) : \mathscr{D}_{\alpha}\left(\mathscr{A}(D) \| \mathscr{A}(D')\right) \triangleq \frac{1}{\alpha - 1} \log\left(\mathbb{E}\left[e^{(\alpha-1)L^{o}}\right]\right) \leq \rho\alpha$
where L^{o} has the following PDF,

$$f(x) = \log\left(\frac{\Pr\left[\mathscr{A}(D)\right] = x}{\Pr\left[\mathscr{A}(D')\right] = x}\right)$$

Neighbouring datasets, $D \simeq D'$, are defined when $D' = D + r$ or $D = D' + r$ for some record r.
These two definitions of differential privacy are related and one can be expressed terms of the other, as follows,

if $\mathscr{A}$ satisfies ρ-zCDP, then it is $\left(\rho + 2\sqrt{\rho \log\left(\frac{1}{\delta}\right)}, \delta\right)$-DP for any $\delta > 0$.

Composition of Differentially Private Algorithms

For algorithms $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_k$ which satisfy (ϵ_i, δ_i)-differential privacy, their sequential composition satisfies $\left(\sum_{i=1}^{k} \epsilon_i, \sum_{i=1}^{k} \delta_i\right)$-differential privacy [3]. Further, for algorithms $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_k$ which satisfy ρ_i-zero concentrated differential privacy, their sequential composition satisfies $\sum_{i=1}^{k} \rho_i$-zero concentrated differential privacy [4].

III. Other Preliminaries and Results

Given a public dataset D_p, the intention of synthetic data generation is to create, as the name suggests, a synthetic dataset D_s. The authors have identified four basic steps which are involved in the process of synthetic data generation. These are marginal selection, noise addition, post-processing and data synthesis. In all four steps, the authors have used a novel approach in PrivSyn as opposed to existing methods. In the first step, their algorithm uses a greedy approach to optimally select marginals that capture the most information. The noise addition step then applies weighted budget allocation, and instead of using Laplacian noise, as is common with other DP methods, they use Gaussian noise. To ensure marginal consistency, they make use of ℓ_1-norm based consistency techniques and finally, for data synthesis, they introduce a method termed as the Gradual Update Method. Since the authors make use of ρ-zero concentrated differential privacy, for the rest of this report, it will be assumed that the a net privacy budget of ρ is used overall.

Gaussian Mechanism

The Gaussian mechanism is a differentially private method way to compute a function f on a dataset D by adding Gaussian noise which is dependent on its ℓ_2 sensitivity (or global sensitivity,

$\mathcal{A}(D) = f(D) + \mathcal{N}\left(0, \Delta_f^2 \sigma^2 \mathbf{I}\right)$ where Δ_f is defined as the global sensitivity of the function f applied on the dataset,

$\Delta_f = \max_{(D,D'):D \simeq D'} ||f(D) - f(D')||$

Here, the noise is a multi-dimensional Gaussian variable with mean 0 and standard deviation $\Delta_f \sigma$ such that $\sigma = \frac{\sqrt{2 \ln \frac{1.25}{\delta}}}{\epsilon}$. Interestingly enough, the choice of the Gaussian mechanism for noising is a relatively rare approach, as most techniques of differential privacy prefer the usage of the Laplacian mechanism, which adds Laplacian-sampled noise based on the ℓ_1 sensitivity of the function or the exponential mechanism which is based on exponentially-distributed noise.

An important theorem related to ρ-zCDP is that the Gaussian mechanism which answers $f(D)$ with noise $\mathcal{N}\left(0, \Delta_f^2 \sigma^2 \mathbf{I}\right)$ satisfies $\left(\frac{1}{2\sigma^2}\right)$-zCDP.

From here, we notice that given the value of (ϵ, δ), we can calculate the value of ρ based on the equivalence between (ϵ, δ)-Differential Privacy and ρ-zCDP, and then use the Gaussian mechanism to noise a function and obtain a differentially private version of the same.

Marginals and Independent Difference

v	$\mathsf{M}_{\text{gender}}(v)$
⟨male,∗⟩	0.40
⟨female,∗⟩	0.60

(a) 1-way marginal for gender.

v	$\mathsf{M}_{\text{age}}(v)$
⟨∗,teenager⟩	0.20
⟨∗,adult ⟩	0.30
⟨∗,elderly⟩	0.50

(b) 1-way marginal for age.

v	
⟨male, teenager⟩	0.08
⟨male, adult⟩	0.12
⟨male, elderly⟩	0.20
⟨female, teenager⟩	0.12
⟨female, adult⟩	0.18
⟨female, elderly⟩	0.30

(c) 2-way marginal assume indepent

v	
⟨male, teenager⟩	0.10
⟨male, adult⟩	0.10
⟨male, elderly⟩	0.20
⟨female, teenager⟩	0.10
⟨female, adult⟩	0.20
⟨female, elderly⟩	0.30

(d) Actual 2-way marginal

A 1-way marginal is the distribution of records over values of each attribute. For example, the marginal of 'Gender' would be the fraction of records belonging to 'Male' and 'Female', respectively, in a tabular form. Similarly, a 2-way marginal is calculated over all pairs of values for a pair of attributes. For example, the marginal of 'Gender', 'Age' can be 'Teenager, Male', 'Teenager, Female', 'Adult, Male', 'Adult, Female', 'Elderly, Male' and 'Elderly, Female', assuming 'Age' takes values from 'Teenager', 'Adult' and 'Elderly' while 'Gender' takes values from 'Male' and 'Female'. Similarly, multi-way marginals are made over two or more than two attributes. Marginals capture the distribution of data over the attributes for which they are calculated. E.g. in Fig. 1, (a) describes the 1-way marginal for gender which shows that 40% of the records are male and 60% female, while the 2-way marginal between age and gender in (d) shows that 30% of the records are of elderly females. Any marginal M has a sensitivity of $\Delta_M = 1$ based on the removal or addition of one record.

To measure the correlation between two attributes, the authors proposed a metric termed as Independent Difference (InDif), which is calculated as s the ℓ_1 distance between the 2-way marginal and the 2-way marginal which is generated assuming independence.

$\text{Indif}_{a,b} = |M_a \times M_b - M_{a,b}|$

They prove in the paper and its supplementary material that $\Delta_{\text{InDif}} = 4$ and compare it with metrics like mutual information and symmetrical uncertainty coefficient introduced in other papers to show the better applicability of InDif to measure correlation.

The sensitivity of a marginal and of the InDif metric is $\Delta_M = 1$ and $\Delta_{InDif} = 4$. Publishing marginal M with noise $\mathcal{N}\left(0, \frac{1}{2\rho}\mathbf{I}\right)$ satisfies ρ-zCDP. Further, publishing an InDif metric with noise $\mathcal{N}\left(0, \frac{8}{\rho}\mathbf{I}\right)$ also satisfies ρ-zCDP. Based on this, the privacy budget can be allocated for each task separately, using the composition theorem for ρ-zCDP. The authors use this to allocate unequal noise between different tasks, This again differs from previously existing methods which use equal-weighted budget allocation schemes.

IV. PrivSyn

So far, the preliminaries and some basic terms relevant to PrivSyn have been introduces. As mentioned previously, the authors have improved on all four steps of the process of synthetic data generation. The algorithm of PrivSyn is as follows,

Algorithm 1: PrivSyn

Input: Private dataset D_p, privacy budget ρ
Output: Synthetic dataset D_s

1. Publish 1-way marginals using the Gaussian Mechanism with $\rho_1 = 0.1\rho$
2. Select 2-way marginals using Algorithm 2 with $\rho_2 = 0.1\rho$
3. Combine marginals using Algorithm 3
4. Publish combined marginals using the Gaussian Mechanism with $\rho_3 = 0.8\rho$
5. Make noisy marginals consistent
6. Construct D_s using the Gradually Update Method

The algorithm illustrates the overall workflow of PrivSyn. We split the total privacy budget into three parts. The first part is used for publishing all 1-way marginals using the Gaussian mechanism, with each marginal getting an equal fraction of the privacy budget. The second part is used to differentially privately select two-way marginals. Here again a fraction of the privacy budget is consumed. The marginal selection method DenseMarg consists of two components, i.e., 2-way marginal selection (Algorithm 2) and marginal combine (Algorithm 3), where marginal combine is used to capture multi-way marginals based on graphical relations between the selected two-way marginals. Finally, the third part is used to obtain the noisy combined marginals. After obtaining the noisy combined marginals, we can use them to construct synthetic dataset D_{syn} without consuming privacy budget, since this is a post processing procedure. The authors term this method as the Gradually Update Method. PrivSyn as a method is only applicable to categorical attributes for which the domain size, i.e. number of categories per attribute, is a finite integer. Continuous features can be handled by binning the attributes into ranges and treating each range as a different category while the algorithm is being applied.

1-Way Marginal Gaussian Mechanism

The first step, viz., 1-way marginal publishing is fairly trivial. It involves the addition of noise in accordance with the Gaussian mechanism. Assuming their are d attributes, the noise added to each would be of the form $\mathcal{N}\left(0, \frac{d}{2\rho'}\right)$ where $\rho' = 0.1\rho$.

DenseMarg Marginal Selection

Algorithm 2: Marginal Selection Algorithm

Input: Number of pairs m, privacy budget ρ, dependency error $\langle \phi_i \rangle$, marginal size $\langle c_i \rangle$
Output: Selected marginal set X

$X \leftarrow \varnothing$ $t \leftarrow 0$ $E_0 \leftarrow \sum_{i \in \bar{X}} \phi_i$
while *True* **do**
  foreach *marginal* $i \in \bar{X}$ **do**
    Allocate ρ to marginals $j \in X \cup \{i\}$;
    $E_t(i) = \sum_{j \in X \cup \{i\}} c_j \sqrt{\frac{1}{\pi \rho_j}} + \sum_{j \in \bar{X} \setminus \{i\}} \phi_j$
  $\ell \leftarrow \arg\min_{i \in \bar{X}} E_t(i)$
  $E_t \leftarrow E_t(\ell)$
  if $E_t \geq E_{t-1}$ **then**
    Break
  $X \leftarrow X \cup \{l\}$
  $t \leftarrow t + 1$

In the phase of obtaining marginals, there are two sources of errors. One is information loss when some marginals are missed and the other is the noise error incurred due to noising mechanisms used in DP. There is a delicate balance which needs to be maintained as choosing too few marginals leads to loss of information and choosing too many marginals leads to excessive noising. To deal with this, the authors have come up with DenseMarg which uses a greedy approach to select 2-way marginals, by framing the selection as an optimisation problem between the noise error due to dependency (information captured in the marginal) and noise error due to Gaussian noise (privacy error).

$$\text{minimize} \sum_{i=1}^{m} [\psi_i x_i + \phi_i (1 - x_i)]$$
$$\text{subject to } x_i \in \{0, 1\}$$

If there are a total of $m = \frac{d(d-1)}{2}$ 2-way marginals, we can represent their dependency and privacy error as ϕ_i and ψ_i respectively and formulate the following optimisation problem, where x_i denotes whether a marginal has been selected or not, and when it has, $x_1 = 1$, else $x_i = 0$.

Here, ψ_i is dependent on the Gaussian noise added, and can be approximated as the expected ℓ_1 error for the marginal M_i with c_i number of cells. Formulaically, $\psi_i = c_i \sqrt{\frac{1}{\pi \rho_i}}$. Further, the dependency error is independent of the privacy budget allocated to each marginal, however it is positively correlated to the InDif score for each attribute. Thus, approximating $\phi_i = \text{InDif}_i + \mathcal{N}\left(0, m^2 \rho'^2 \mathbf{I}\right)$ for a fixed value of ρ'. As the authors have not clearly mentioned what choice of ρ' is appropriate, other than $\rho' < \rho$. The authors then show, that given a fixed set of marginals that has been chosen (for which x_i is determined), the optimal privacy budget allocation can be reframed as,

$$\text{minimize} \sum_{i: x_i = 1} c_i \sqrt{\frac{1}{\rho_i}}$$
$$\text{subject to} \sum_{i: x_i = 1} \rho_i = \rho$$

where ρ_i is the budget allocated corresponding to each marginal. Overall, on solving, one obtains the budget allocation as,

$$\rho_i = \frac{c_i^{2/3}}{\sum_j c_j^{2/3}} \rho$$

The authors handle this algorithmically by using a greedy approach. They do so by iteratively including marginals that give the maximal utility improvement. In particular, in each iteration, one marginal is selected that brings the maximum improvement to the overall error. More specifically, each marginal i that is not yet included in X (i.e., $i \in \overline{X}$, where $\overline{X} = \{1,2,3,\ldots,m\} \backslash X$) is considered for budget allocation followed by error calculation and then greedily the marginal with maximum utility improvement is selected. Once the utility does not improve (i.e. error starts to increase), the algorithm is terminated, which is guaranteed to occur due to the noise error as more marginals are selected.

Marginal Combination

Algorithm 3: Marginal Combine Algorithm

Input: Selected pairwise marginals X, threshold γ
Output: Combined marginals $\mathcal{X}$

```
Convert X to a set of pairs of attributes
Construct graph G from the pairs
S ← ∅; 𝒳 ← ∅
foreach clique size s from m to 3 do
    C_s ← cliques of size s in G
    foreach clique c ∈ C_s do
        if |c ∩ S| ≤ 2 and domain size of c ≤ γ then
            Append c to 𝒳
            Append the attributes of c to S
```

Using two-way marginals captures only the correlation between two attributes. However, sometimes multi-way marginals can help capture more information when the domain size is restricted. Thus, the authors propose a marginal combine algorithm to return certain combined multi-way marginals from the selected two-way marginals. This is done by converting the selected 2-way marginals into a graph, followed by extracting all the cliques from said graph. The cliques are then sorted by the number of nodes in descending order and are selected if they contain not more than 2 attributes which have already been selected, and removing all the 2-way marginals contained within said clique.

2-Way and Multi-Way Marginal Gaussian Mechanism

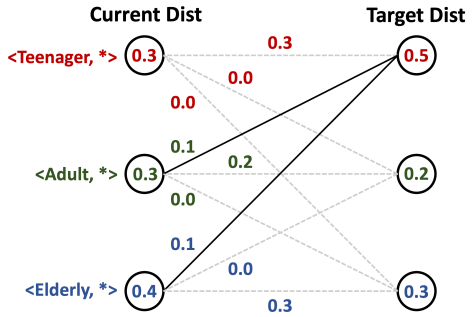
The next step is the addition of noise in accordance with the Gaussian mechanism for the selected two-way and multi-way marginals. Assuming their are m such marginals, the noise added to each would be of the form $\mathcal{N}\left(0, \frac{m}{2\rho'}\right)$ where $\rho' = 0.8\rho$. Here, the majority of privacy budget is allocated as these marginals capture not only the one-way marginals, but also important correlation information between attributes.

Ensuring Marginal Consistency

In terms of differential privacy, all of the privacy budget has been consumed by the last step. However, since the Gaussian mechanism involved the addition of Gaussian noise to the marginals, it may occur that the distribution obtained using the marginals may not be consistent. For instance, the two-way marginal on age and gender may disagree with the one-way marginal on gender regarding the ratio of males and females in the dataset. Further, as the Gaussian distribution has a suport over the entire real line, it may happen that the marginals (when viewed as a probability) may not sum to 1, their entries (or summation) may itself exceed 1, and/or they may contain negative values. This does not make sense intuitively and it is important to ensure that the marginals obtained are consistent. The authors used existing methods for the first part, where each marginal is made consistent to itself (i.e. it satisfies a summation to 1 with non-negative entries) like Norm-Cut and Norm-Sub [2] which minimise the ℓ_2 norm distance between the original and corrected marginals, followed by ensuring consistency between two or more marginals sharing a common attribute. For this, they utilise the fact that the marginal of each attribute may be calculated more than once over all the one-way, two-way and multi-way marginals, and the mean of all those entries represents a better estimate of the differentially private marginal than any individual choice.

Gradually Update Method (Dataset Synthesis)

The differential privacy portion of the paper was relevant till the calculation of and ensuring the consistency of the marginals. Thereafter, the next part of dataset generation is to finally use the marginals to generate a new dataset with records, and use it a differentially private dataset. For this, the authors used a method they term as the Gradually Update Method. Generally, the prevalent method for dataset generation based on marginals is to use minimum cost flow methods. In this method, the dataset is generated randomly and then is made consistent with each marginal. Graphically, a bipartite graph is created for the marginal to represent the flow of records from the original distribution to the marginal-consistent distribution. An illustration of the same for the age marginal is shown below.



This process iterates over all marginals to enforce consistency and is repeated multiple times to ensure that the dataset is consistent with almost all marginals. This method shows slow convergence as each step enforces strict consistency with each marginal which ends up invalidating previously enforced consistencies [15]. To deal with this, the authors use a multiplicative update method termed as Gradual Update Method (or Gradually Update the dataset on Marginals). Similar to MCF, GUM also makes use of a flow graph, but differs in other ways. Firstly, it doesn't make the dataset fully consistent with a marginal, rather it chooses multiplicative

parameters α and β to bound the number of records which change when the marginals have lower and higher values than required, respectively. α and β can be calculated and optimised to improve convergence performance, as the authors do by choosing α to be gradually decreasing.

When updating records, there are two ways to do so, one is to use a replace operation which modifies the value in a cell to the desired value. This affects other marginal values with common attributes. The other option is to use a duplication operation where a record is duplicated. This doesn't affect other marginals but also does not introduce new records, which may affect the dataset negatively if certain combinations do not already exist in the dataset. Empirically, the authors have shown that using a combined approach, where in every two iterations of the GUM method one uses a half-half approach with half records being replaced and the rest half being duplicated and the other uses solely the duplicate method, improves the convergence. Using a combined approach is important as only using the replace strategy significantly impacts the correlation information established by other marginals (similar to MCF) while only using the duplicate strategy will not introduce new records that can better reflect the overall joint distribution.

Some of the steps can be optimised to improve convergence and performance. For instance, while publishing 1-way marginals, one can apply a threshold and filters values with estimates smaller than that threshold by combining them. Further, when the DenseMarg marginal selection algorithm has a low privacy budget, it leads to disjoint subgraphs between the attributes when represented graphically. In this case, applying GUM to each subgraph and randomly shuffling the records before joining improves convergence. Another optimisation based on GUM is that attributes with a degree of one or less can be removed from the GUM method and updated later, as they are fully independent of (or can be determined completely using the pairwise distribution) by the other attributes, once they are consistent. Also, as mentioned previously, the choice of α (and β as well) and how its value changes across iterations also affects performance. However, some of these have not yet been implemented due to implementation constraints and otherwise satisfactory performance obtained without using them in the first place.

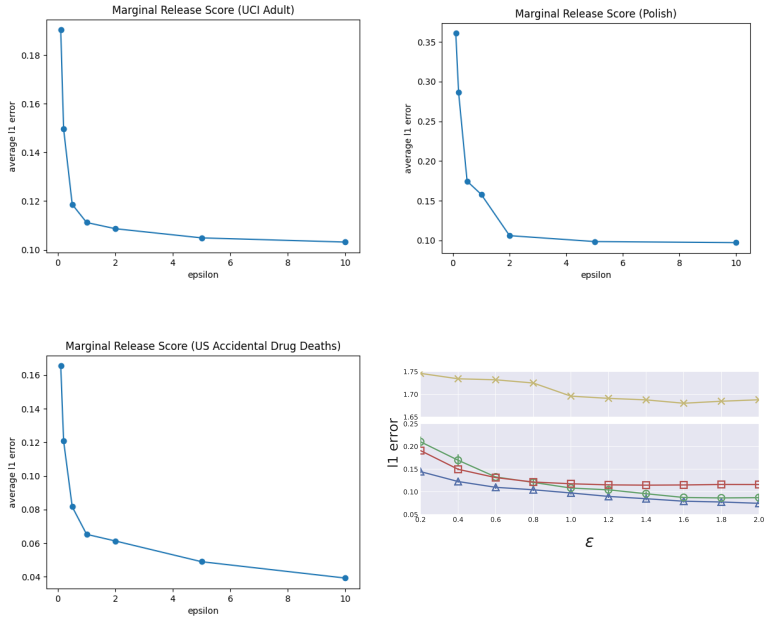
V. Experiments

The original paper evaluates the performance of PrivSyn based on 3 metrics, viz. Marginal Release, Range Query and Classification, across four datasets, due to constraints, the evaluations have only been conducted for Marginal Release and Classification for one of the datasets mentioned in the paper (UCI Adult) and another dataset (Polish) and Marginal Release for another dataset not mentioned in the paper (US Accidental Drug Deaths). In terms of datasets, there was an issue that two of the datasets mentioned in the paper (Colorado and Loan) cannot be found publicly while the other one (US Accident) shows some computational issues due to its size (more than 5 million records and around 50 attributes). Further, the implementation of Range Query was unclear as to what kind of queries were to be used for any three attributes. Also, the Classification metric was not evaluated for the US Accidental Drug Deaths dataset as it is not a dataset suited for classification tasks.

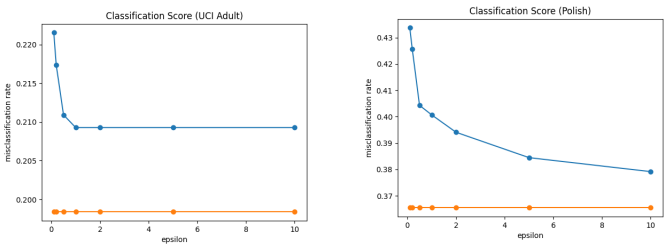
The Marginal Release metric is measured as the average ℓ_1 between marginals while the Classification metric is defined as the misclassification rate on an SVM.

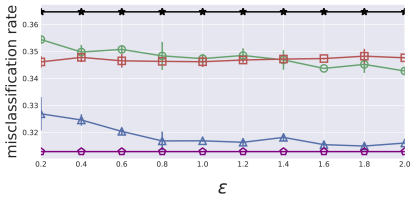
For evaluations, the value of δ is fixed at $3 \cdot 10^{-11}$ and the value of ρ' in DenseMarg is always passed as δ. Also, α uses exponential decay, as that was shown to be the best decay, empirically, by the authors. For training the SVM model, the sci-kit learn implementation was used without any hyper-parameter tuning or train-test split.

It can be seen that the results for Marginal Query closely matches the trend of the results of the paper, where the paper uses blue triangles for PrivSyn (and other symbols for other methods). For the same dataset (UCI Adult) as the papers, the values obtained are not exactly the same this can be attributed to stochasticity and implementation differences. However, the important thing is that the results closely show the same trend as that of PrivSyn and the order of magnitude of the metric is the same, differing only slightly. The same can be said for the US Accidental Drug Deaths dataset and Polish dataset which shows the same trend. The last graph is that of their evaluation on UCI Adult (PrivSyn is shown by blue triangles).



For the Classification Score metric, the misclassification rate using the private dataset for training an SVM (and evaluating on the same dataset) is 0.19839482412677611 and 0.36547497446373856 for UCI Adult and Polish, respectively. The misclassification rate increases on using the synthetic dataset, but is relatively close to the misclassification rate on the base SVM. Further, as shown in the paper, the misclassification rate is mostly invariant of the privacy budget used with a very weak downwards correlation as the privacy budget increases. The last graph here is of their evaluation on UCI Adult (PrivSyn is shown by blue triangles).





From here, it can be said that the results shown in the paper do match the trend shown during the evaluation, and that PrivSyn is a useful and fast method for differentially private synthetic data generation. The authors have also mathematically proved the that the time complexity of PrivSyn is lower than that of other methods.

VI. Comparison to Other Methods

Method \ Step	Marginal Selection	Noise Addition	Post Processing	Data Synthesis
PriView	Covering design	Equal budget + Laplace	Max-entropy Estimation	-
PrivBayes	Bayesian network + Info Gain (EM)	Equal budget + Laplace	-	Sampling
PGM	- (not dense)	Equal budget + Gaussian	Markov Random Field	Sampling
PrivSyn	Optimization + Greedy	Weighted budget + Gaussian	Consistency	GUM

This section specifically mentions how the PrivSyn method differs from existing methods. PrivSyn method is compared with other similar methods, i.e. those which use marginals, graphs and probabilistic settings. Other approaches are to use game-based formulations such as MWEM [10] and DualQuery [9] or deep generative models such as DP-GAN [7], G-PATE [6] and PATE-GAN [8]. Due to their approaches being fundamentally different from PrivSyn, they are not discussed further in this section. As mentioned previously, the authors identified four common steps in synthetic data generation, viz., marginal selection, noise addition, post processing and data synthesis. To this effect, the authors use novel approaches for all four. DenseMarg as the marginal selection algorithm (and the subsequent marginal combine algorithm) uses a greedily optimised approach. For noise addition, the authors use a weighted budget distribution instead of an equal budget allocation as is common otherwise. Further, they make use of the Gaussian mechanism instead of the Laplacian mechanism. Then, for post-processing they use marginal consistency techniques before generating the dataset using their novel GUM approach. This is different from methods like PriView [14], PrivBayes [11] and PGM [13], whose approaches are summarised in the above table, and also from other related methods like PrivMRF [12], which is based on Markov Random Fields like PGM. To show that their method is superior or equivalent to these methods, the authors conduct extensive ablation studies as well, which could not be replicated due to resource constraints. However, it can be said that their approach is much more scalable than the methods above in terms of increasing the number of attributes or domain size. One of the major drawbacks of PrivSyn (and other graphical/marginal based methods) is their inability to deal with continuous features, for which they rely on other strategies like binning to convert the numerical features to categorical ones.

VII. Conclusion

PrivSyn as a method provides theoretical guarantees for its ability to provide differentially private data synthesis. While implementing the code for the same, it was seen that there are some ambiguities in the algorithm which need to be made much clearer in the paper itself. For instance, the choice of ρ' for noising the dependency error is not clear. Further, even though the GUM method is defined within the paper, a separate algorithm for the same is not provided, which would have made its implementation easier. Rather, its description is mostly theoretical and no formal procedure has been provided and it is only defined with respect to the other dataset synthesis method which they used, viz. MCF. Apart from this, the results of the paper seem to mostly match the results obtained during the implementation, indicating that PrivSyn is useful for scalable data synthesis. However, even here, the Range Query metric is poorly defined. For the code implementation, the only important inputs needed are the data CSV file, and three configuration based files (YAML and JSON) to specify the data-loading steps. This is because before applying PrivSyn, it is important to deal with datatypes, missing values and continuous features.

References

1. Zhang, Zhikun, Tianhao Wang, Ninghui Li, Jean Honorio, Michael Backes, Shibo He, Jiming Chen, and Yang Zhang. "{PrivSyn}: Differentially Private Data Synthesis." In 30th USENIX Security Symposium (USENIX Security 21), pp. 929-946. 2021.
2. Wang, Tianhao, Milan Lopuhaä-Zwakenberg, Zitao Li, Boris Skoric, and Ninghui Li. "Locally differentially private frequency estimation with consistency." arXiv preprint arXiv:1905.08320 (2019).
3. Dwork, Cynthia, and Aaron Roth. "The algorithmic foundations of differential privacy." Foundations and Trends® in Theoretical Computer Science 9, no. 3–4 (2014): 211-407.
4. Bun, Mark, and Thomas Steinke. "Concentrated differential privacy: Simplifications, extensions, and lower bounds." In Theory of Cryptography Conference, pp. 635-658. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016.

5. Dwork, Cynthia, Frank McSherry, Kobbi Nissim, and Adam Smith. "Calibrating noise to sensitivity in private data analysis." In Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3, pp. 265-284. Springer Berlin Heidelberg, 2006.
6. Long, Yunhui, Boxin Wang, Zhuolin Yang, Bhavya Kailkhura, Aston Zhang, Carl Gunter, and Bo Li. "G-pate: Scalable differentially private data generator via private aggregation of teacher discriminators." Advances in Neural Information Processing Systems 34 (2021): 2965-2977.
7. Xie, Liyang, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. "Differentially private generative adversarial network." arXiv preprint arXiv:1802.06739 (2018).
8. Jordon, James, Jinsung Yoon, and Mihaela Van Der Schaar. "PATE-GAN: Generating synthetic data with differential privacy guarantees." In International conference on learning representations. 2018.
9. Gaboardi, Marco, Emilio Jesús Gallego Arias, Justin Hsu, Aaron Roth, and Zhiwei Steven Wu. "Dual query: Practical private query release for high dimensional data." In International Conference on Machine Learning, pp. 1170-1178. PMLR, 2014.
10. Hardt, Moritz, Katrina Ligett, and Frank McSherry. "A simple and practical algorithm for differentially private data release." Advances in neural information processing systems 25 (2012).
11. Zhang, Jun, Graham Cormode, Cecilia M. Procopiuc, Divesh Srivastava, and Xiaokui Xiao. "Privbayes: Private data release via bayesian networks." ACM Transactions on Database Systems (TODS) 42, no. 4 (2017): 1-41.
12. Cai, Kuntai, Xiaoyu Lei, Jianxin Wei, and Xiaokui Xiao. "Data synthesis via differentially private markov random fields." Proceedings of the VLDB Endowment 14, no. 11 (2021): 2190-2202.
13. McKenna, Ryan, Daniel Sheldon, and Gerome Miklau. "Graphical-model based estimation and inference for differential privacy." In International Conference on Machine Learning, pp. 4435-4444. PMLR, 2019.
14. Qardaji, Wahbeh, Weining Yang, and Ninghui Li. "Priview: practical differentially private release of marginal contingency tables." In Proceedings of the 2014 ACM SIGMOD international conference on Management of data, pp. 1435-1446. 2014.
15. Magnanti, Thomas L. "Ravindra K. Ahuja Thomas L. Magnanti and James B. Orlin."