POSTER: A Unified Framework of Differentially Private Synthetic Data Release with Generative Adversarial Network

Pei-Hsuan Lu and Chia-Mu Yu
National Chung Hsing University, Taiwan

ABSTRACT

Many differentially private data release solutions have been proposed for different types of data with the sacrifice of inherent correlation structure. Here, we propose a unified framework of releasing differentially private data. In particular, our proposed generative adversarial network (GAN)-based framework learns the input distribution, irrespective of tabular data and graphs, and generates synthetic data in a differentially private manner. Our preliminary results show the acceptable utility of the synthetic dataset.

1 INTRODUCTION

Privacy is of paramount importance particularly for machine learning and data analysis tasks on datasets with individual information. For example, the Netflix challenge releases the anonymized data, seeking for the performance improvement. Despite a success for crowdsourcing, Netflix challenge is also a classic example of how the wrongly anonymized data expose the individual information.

The above scenario motivates the problem of *private data release*, where sensitive dataset needs to be sanitized before released. A straightforward idea is to release the anonymized data (e.g., k-anonymity). Unfortunately, these *ad hoc* anonymization techniques are designed without the consideration of attacker's knowledge, and are subject to linkage attack and homogeneity attack etc.

1.1 Differential Privacy

Differential privacy (DP) is a provable privacy notion. With the sensitive dataset D to be released, (non-interactive) DP requires that only the sanitized dataset $A(D)$ can be released, where A is a randomized algorithm such that the output of A reveals limited information about any particular record in D. Formally, a randomized algorithm A satisfies (ϵ, δ)-differential privacy (ϵ, δ)-DP, if for any two datasets D_1 and D_2 differing only in one record, and for any possible output O of A, we have $\Pr[A(D_1) = O] \leq e^{\epsilon} \Pr[A(D_2) = O] + \delta$, where two datasets are neighboring if they differ in only one record.

The Laplace and Gaussian mechanisms achieve certain form of (ϵ, δ)-DP. In particular, the former aims to release the output of a numeric function F by adding i.i.d. noise η into each output value of F. The noise η is sampled from a Laplace distribution $\text{Lap}(\lambda)$ with $\Pr[\eta = x] = \frac{1}{2\lambda} e^{-|x|/\lambda}$. The latter provides (ϵ, δ)-DP if the Gaussian noise $\mathcal{N}(0, \Delta_F^2 \cdot \sigma^2)$ is applied, where $\mathcal{N}(0, \Delta_F^2 \cdot \sigma^2)$ is the Gaussian distribution with mean 0 and standard deviation $\Delta_F \cdot \sigma$, $\delta \geq \exp(-\sigma^2\epsilon^2/2)/1.25$, and $\epsilon < 1$.

CCS '17, October 30-November 3, 2017, Dallas, TX, USA

ACM ISBN 978-1-4503-4946-8/17/10.
https://doi.org/10.1145/3133956.3138823

Given a differentially private algorithm, the composition of the algorithm and post-processing steps is still differentially private. The most useful is the sequential composition theorem, In essence, given a (ϵ_1, δ_1)-DP algorithm A_1 and (ϵ_2, δ_2)-DP algorithm A_2, the $A_2(A_1(D), D)$ satisfies $(\epsilon_1 + \epsilon_2, \delta_1 + \delta_2)$-DP. A special case is that given a (ϵ_1, δ_1)-DP algorithm A_1 and the post-processing A_2, the $A_2(A_1(D))$ is still (ϵ_1, δ_1)-DP. Namely, the post-processing without accessing D does not consume privacy budget ϵ.

1.2 Recent Work on Private Data Release

Many efforts have been devoted to developing techniques for private data release. A useful technique in publishing tabular data is to first learn the inherent data distribution from original dataset D. Then, the data owner generates and releases synthetic dataset D' via random sampling from the data distribution. Since D' and D share the same data distribution, the analysis results are supposed to be similar. PrivBayes [6] constructs a Bayesian network (BN) to learn the data distribution. After adding noises to BN, data owner performs random sampling from the noisy data distribution. A subsequent work, JTree [3], follows the similar strategy.

On the other hand, different techniques are used in releasing graph statistics. Consider the case of publishing node degree distribution. A common approach is to construct a projective graph $\hat{G}$ from the original graph G. After that, a differentially private histogram for node degree distribution is then released [4, 5].

Unfortunately, we face two research challenges. **(C1)** First, for high-dimensional dataset with the large domain size of each attribute and with complex correlation among attributes, modeling the data distribution consumes considerable time and even computationally infeasible. Very often, the design choice is to keep the pairwise correlation among attributes in the learned data distribution. However, by doing so, the data user may gain inaccurate result when issuing multidimensional query to synthetic data. **(C2)** Second, while currently different DP techniques need to be used on different types of data, the lack of a unified framework for generating differentially private synthetic data leads to the increased complexity in managing the risk of information disclosure. In this paper, we propose to use generative adversarial network as a foundation of such a unified framework.

1.3 DNN and GAN

Deep neural networks (DNN) composed of multiple interconnected layers, each of which contains a number of neurons, have proved the remarkable capability on various machine learning tasks. In essence, DNN, as a parameterized function, extracts the hidden structure behind the data. The DNN with varying parameters can be trained to fit any given finite set of input/output examples. To train a DNN, we define a loss function $\mathcal{L}$ that represents the mismatch between the data and DNN output. A common setting of the loss is $\mathcal{L}(\theta) = \frac{1}{n} \sum_i \mathcal{L}(\theta, x_i)$ with the training examples $\{x_1, \ldots, x_n\}$.

Training aims to find θ such that the $\mathcal{L}(\theta)$ is minimized. However, due to the complex structure of DNN, $\mathcal{L}$ is usually non-convex and difficult to minimize. In practice, one usually resorts to stochastic gradient decent (SGD) algorithm to minimize $\mathcal{L}$.

Generative models, as opposed to discrimitive models (e.g., DNN), allow one to generate samples from the data distribution. Generative adversarial network (GAN) is composed of two networks, generator and discriminator. The former seeks to create synthetic data satisfying the data distribution, while the latter determines whether the data from generator is a synthetic or true data sample. Recently, GAN gains a huge success in synthesizing meaningful images.

2 PROPOSED METHOD

In this section, we present a series of differentially private synthetic data generation algorithms for tabular data and graphs[1].

2.1 Basic Idea

Here, we still follow the similar strategy in [3, 6], learning the input distribution for private data release. However, since training an optimal BN is NP-Hard, one explicitly removes correlation among attributes for efficiency [6]. We make an observation that a well-trained machine learning (ML) model also aims to approximate the input distribution without removal of explicit structures. Thus, we deal with the challenge **(C1)** by learning the input distribution with ML model. Note that we particularly consider deep learning (DL) family, because DL can easily gain speedup with the GPUs whereas whether BN and JT algorithms can be parallelized remain unclear. On the other hand, in fact, if the input data are well-represented, DL is able to learn the input distribution, irrespective of the form of input data. Hence, one can develop a differential private DL model to approximate the input distribution, from which one generates and publishes the synthetic data, resolving the challenge **(C2)**.

2.2 DP-DNN for Tabular Data

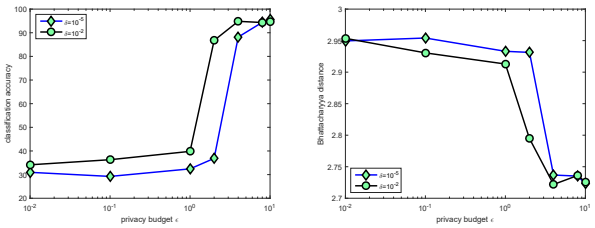
We first present a differential private data release scheme, DP-DNN, by taking advantage of DNN, as a strawman solution. The idea behind our design is the observation that a well-trained DNN inherently models the original dataset $D \in \mathbb{R}^{n\times m}$. Thus, a DNN model trained from D can be used by the sampling procedure for data synthesis.

However, we need to deal with two obstacles when implementing the above idea. First, recent research results show that DNN model itself may expose the privacy of training data. Thus, we instead train a DNN in a differentially private manner. In particular, by using the recent moment accountant approach in manipulating the privacy parameters ϵ and δ in differential privacy [1], we construct a DNN with fully connected layers via differentially private SGD. In essence, due to the use of differentially private SGD, DP-DNN reveals no individual information in D. Second, DNN is supposed to take as input the labelled dataset but D might be unlabelled (i.e., no clear distinction between quasi-identifiers and sensitive attributes). Here, we propose to randomly pick an attribute as a label for records in D, turning an unlabelled dataset to labelled one. In essence, this design choice does not change the structure of D; only the way for DP-DNN to perceive the data is changed.

After DP-DNN construction, data owner generates a uniformly random sample $a \in \mathbb{R}^{1\times(m-1)}$. Data owner sets $D' = D' \cup \{[a \text{ DP-DNN}(a)]\in \mathbb{R}^{1\times m}\}$, where D' denotes the synthetic dataset and DP-DNN(a) denotes the prediction made by DP-DNN on a. The above procedures are repeated until the number of records of D' reaches n. The above sampling procedures can be thought of as a post-processing and do not consume privacy budget ϵ. The above sampling procedures can also be optimized. First, with the knowledge of the value ranges of attributes, one can guarantee that the random samples will not be significantly deviated from the population, if samples are randomly drawn from a limited value range. Second, with the knowledge of proportions of each label, when the inclusion of $[a$ DP-DNN(a)$]$ results in a significant deviation from the proportion of DP-DNN(a) in D, $[a$ DP-DNN(a)$]$ is dropped. Figures 1 and 2b show the experiment results of DP-DNN.

The dataset for all the experiment results in Secs. 2.2~2.3 are Wine Data Set[2], and each result in Figures 1~5 is the average of ten independent experiments. *Classification accuracy* quantifies the similarity between how classification work on D and D', *Bhattacharyya distance* measures the dissimilarity between D and D', and *correlation matrix* in Figure 2 visualizes whether D' preserves the correlation structure in D.

Though Figure 1 shows promising results, the correlation matrix, as shown in Figure 2, shows the weakness of DP-DNN. More precisely, due to the independent sampling, by comparing Figure 2a and Figure 2b, one can easily find that the synthetic dataset does not preserve the correlation among attributes in D.



(a) classification accuracy. **(b) Bhattacharyya distance.**

Figure 1: DP-DNN results with varying ϵ's and δ's.

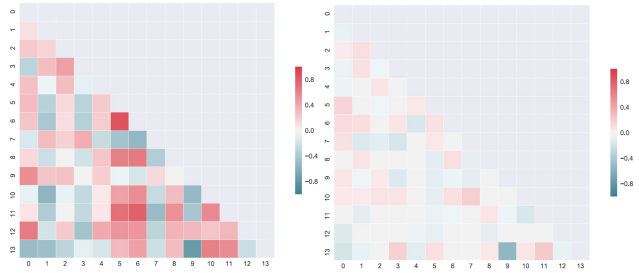
2.3 DP-GAN for Tabular Data

With the observation that GAN can approximate the input distribution, we construct a differentially private GAN, DP-GAN, for private data synthesis. More specifically, we train a DP-GAN, perform random sampling from DP-GAN, and release the dataset composed of random samples. The only obstacle is how to make GAN differentially private. We make an observation that though GAN consists of two parts, generator and discriminator, only the latter has the access to D, when learning the input distribution. Therefore, since the discriminator of GAN in our consideration is a DNN, we construct DP-GAN by simply using DP-DNN in Sec. 2.2 in place of the ordinary DNN.

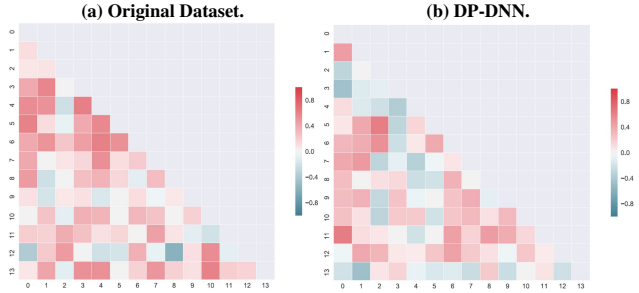
We note that at the time of writing, we also found that Beaulieu-Jones et al. [2] also propose similar idea on differentially private data synthesis. However, they craft a DP-GAN based on AC-GAN, a variant of GAN, while our design of DP-GAN is based on the DC-GAN (ordinary GAN) with DNN as discriminator. Moreover, our further discussion on the novel use of DP-GAN (see Sec. 2.4 and Sec. 2.5 is also not included in [2].

[1]The source code of all the proposed algorithms can be downloaded from https://goo.gl/94qyQz.

[2]Wine Data Set: https://archive.ics.uci.edu/ml/datasets/wine



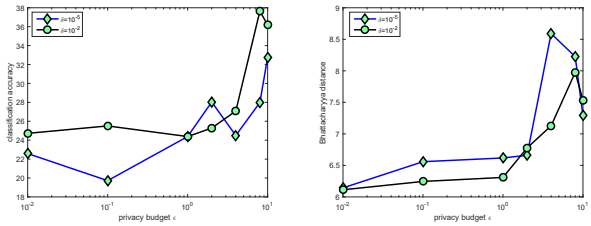
(a) Original Dataset. (b) DP-DNN.



(c) DP-GAN. (d) DP-GAN-DNN.

Figure 2: Correlation matrices from different solutions.

The experiment results are shown in Figures 2c and 3. One can easily see that, compared to DP-DNN, though more correlations among attributes in D are preserved in DP-GAN, both classification accuracy and Bhattacharyya distance are even worse, rendering the impracticality of such a design of DP-GAN.



(a) classification accuracy. (b) Bhattacharyya distance.

Figure 3: DP-GAN results with varying ϵ's and δ's.

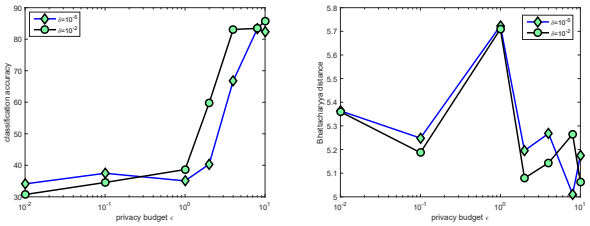
2.4 DP-GAN-DNN for Tabular Data

The failure of DP-DNN in preserving the correlation structure in D stems from the independently random sampling. In essence, if random samples to be sent to DP-DNN naturally satisfy the input distribution, D' generated from DP-DNN could better preserve the correlation structure in D. Thus, a straightforward idea is to combine the use of DP-DNN and DP-GAN such that random samples to be sent to DP-DNN are first generated by DP-GAN. All the remaining procedure are exactly the same as those in Sec. 2.2.

The experiment results are shown in Figures 2d and 4, where the classification accuracy reaches the acceptable level and Bhattacharyya distance is also reduced, compared to DP-GAN. One can also see from Figures 2a and 2d that more correlations among attributes in D are now preserved.

2.5 DP-GAN for Graph

We also consider publishing node degree distribution of a given grpah in the sense of node-DP [4, 5], instead of the tabular data. The similar approach is conducted; data owner learns the input distribution, and publishes the synthetic node degree distribution. Nonetheless, the obstacle is that, in contrast to the tabular data case

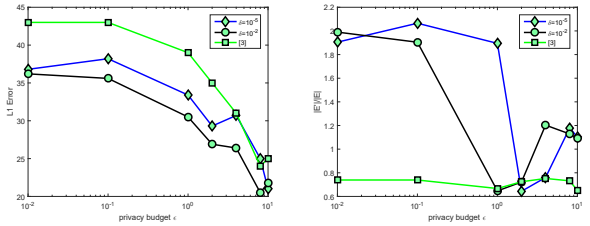


(a) classification accuracy. (b) Bhattacharyya distance.

Figure 4: DP-GAN-DNN results with varying ϵ's and δ's.

where one see each record as a sample from an inherent data distribution, now we have a single instance of graph and therefore cannot train a DP-GAN. To solve this issue, we first generate graph isomorphisms of the original graph G. Then, we vectorize the adjacency matrix of each graph isomorphism as a row vector (record). All the vectorized adjacency matrices can be stacked together and regarded as a tabular data, each record from which is sampled from an inherent edge distribution. All the remaining procedures are the same as ones in Sec. 2.3.

The dataset for all the experiment results in Sec. 5 is Zachary's Karate Club[3], and each result in Figure 5 is the average of ten independent experiments. *L1 Error* measures the dissimilarity between the released and true node degree distribution. $|E'|/|E|$ is a ratio of the number $|E'|$ of edges in the synthetic graph G' and the number $|E|$ of edges in G. The experiment results in Figure 5 show that DP-GAN achieves less information loss than the state-of-the-art solution [4].



(a) L1 error. (b) $|E'|/|E|$.

Figure 5: DP-GAN for Graph.

3 CONCLUSION

The GAN-based framework of differentially private data release shows a great potential in preserving the data utility and unifying the DP approaches for different types of data.

REFERENCES

[1] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang. Deep learning with differential privacy. *ACM CCS*, 2016.
[2] B. K. Beaulieu-Jones, Z. Wu, C. Williams, and C. S. Greene. Privacy-preserving generative deep neural networks support clinical data sharing. bioRxiv, July 5, 2017.
[3] R. Chen, Q. Xiao, Y. Zheng, and J. Xu. Differentially private high-dimensional data publication via sampling-based inference. *ACM KDD*, 2015.
[4] W.-Y. Day, N. Li, and M. Lyu. Publishing graph degree distribution with node differential privacy. *ACM SIGMOD*, 2016.
[5] S. Raskhodnikova and A. Smith. Lipschitz extensions for node-private graph statistics and the generalized exponential mechanism. *IEEE FOCS*, 2016.
[6] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao. Privbayes: private data release via bayesian networks. *ACM SIGMOD*, 2014.

[3] https://networkdata.ics.uci.edu/data.php?id=105