Knowledge-Based Systems 264 (2023) 110239

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

MC-GEN: Multi-level clustering for private synthetic data generation

Mingchen Li [a,1,*], Di Zhuang [b,1,2], J. Morris Chang [a]

[a] University of South Florida, 4202 E Fowler Ave, Tampa, FL 33620, United States of America
[b] Snap Inc., 2850 Ocean Park Blvd, Santa Monica, CA 90405, United States of America

ARTICLE INFO

Article history:
Received 24 July 2022
Received in revised form 5 December 2022
Accepted 24 December 2022
Available online 21 January 2023

Keywords:
Synthetic data generation
Differential privacy
Machine learning

ABSTRACT

With the development of machine learning and data science, data sharing is very common between companies and research institutes to avoid data scarcity. However, sharing original datasets that contain private information can cause privacy leakage. A reliable solution is to utilize private synthetic datasets which preserve statistical information from original datasets. In this paper, we propose MC-GEN, a privacy-preserving synthetic data generation method under differential privacy guarantee for machine learning classification tasks. MC-GEN applies multi-level clustering and differential private generative model to improve the utility of synthetic data. In the experimental evaluation, we evaluated the effects of parameters and the effectiveness of MC-GEN. The results showed that MC-GEN can achieve significant effectiveness under certain privacy guarantees on multiple classification tasks. Moreover, we compare MC-GEN with three existing methods. The results showed that MC-GEN outperforms other methods in terms of utility.

1. Introduction

Machine learning has become an important technology in many fields, such as medical diagnosis, fraud detection, and product analysis. As a data-driven approach, data is considered as the fuel of machine learning algorithm [1], and the performance of the machine learning model often depends on the amount of data. To ensure the performance of machine learning, data sharing happens very often among organizations with similar research interests. For instance, if the research institutes do not have sufficient data for an illness diagnostic system, hospitals can share their data (patient records) to make up for the data gap. However, such data usually carry private information that can cause privacy leakage. Hence, sharing data in a privacy-preserving way for machine learning is of vital importance.

Sharing synthetic datasets generated from original datasets is a common way to protect data privacy. However, the individual information can be easily inferred with some background knowledge. To prevent this issue, differential privacy (DP) [2] has been widely used as a strong and provable privacy guarantee at the individual sample level. Synthetic data releasing under differential privacy emerges as a reliable solution for privacy-preserving data sharing in machine learning to protect individual records in the original datasets. It allows the data owner to publish synthetic datasets to data users without privacy concerns, and data users can make use of synthetic datasets for different purposes, such as machine learning, data mining, etc.

Designing a powerful privacy-preserving synthetic data generation method for machine learning purposes is of great challenge. First, a privacy-preserving method usually introduces perturbations on data samples that hurt the utility of data. Mitigating the perturbation to reach a certain level of utility is not easy. Second, machine learning has multiple tasks, like support vector machine, logistic regression, random forest, and k-nearest neighbor. An effective synthetic data generation method should be applicable to different tasks. Third, some data has a complex distribution. Forming an accurate generator based on the whole data distribution is hard.

In recent years, several works related to private synthetic data release have been proposed in the literature [3–10]. One kind of method is to produce the synthetic database that preserved privacy and utility regarding query sets. These algorithm [3,4] are usually designed to answer different query classes, and data samples in the dataset are not actually released. In [5,6], they proposed algorithms for synthetic data release based on noisy histograms, which are more focused on the categorical feature variables. Algorithm [7,8,10] generate synthetic dataset under statistical model with some preprocessing on the original datasets. However, they only consider generating the synthetic data based on the whole data distribution.

* Corresponding author.
E-mail addresses: mingchenli@usf.edu (M. Li), zhuangdi1990@gmail.com (D. Zhuang), chang5@usf.edu (J.M. Chang).

1 This work was done by Mingchen Li and Di Zhuang when they were at the University of South Florida.

2 Di Zhuang is currently with Snap Inc., 2772 Donald Douglas Loop N, Santa Monica, CA 90405, United States of America.

https://doi.org/10.1016/j.knosys.2022.110239
0950-7051/© 2023 Elsevier B.V. All rights reserved.

Our primary motivation is to allow data owners to share their datasets without privacy concerns by using synthetic datasets instead of original datasets. We proposed Multiple-level Clustering based GENerator for synthetic data (MC-GEN), an approach that uses multi-level clustering (sample level and feature level) and differentially private multivariate Gaussian generative model to release synthetic datasets which satisfy ϵ-differential privacy. Extensive experiments on different original datasets have been conducted to evaluate MC-GEN and its parameters. The results demonstrate that synthetic datasets generated by MC-GEN maintain the utility of classification tasks while preserving privacy. Furthermore, we compared MC-GEN with other existing methods, and our results show that MC-GEN outperforms other existing methods in terms of effectiveness.

The main contributions of MC-GEN are summarized as follows:

- We proposed and released an innovative, effective synthetic data generation method, MC-GEN, which allows the data owners to share synthetic datasets for multiple classification tasks without privacy concerns.[3]
- We demonstrated that applying feature clustering upon sample level clustering engages less differentially private noise to achieve the same level of privacy compared to sample level clustering.
- We conducted extensive experiments to investigate the effects of the parameters and the effectiveness of MC-GEN on three classification datasets. Meanwhile, we compared MC-GEN with three existing methods to demonstrate its utility.

The rest of the paper is organized as follows: Section 2 introduces some related works. Section 3 presents some preliminary of our approach. Section 4 describes our methodology and generative model of MC-GEN. Section 5 presents the experimental evaluation of MC-GEN. Section 6 makes the conclusions.

2. Related work

Moritz et al. [3] proposed an algorithm that combined the multiplicative weighs approach and exponential mechanism for differentially private database release. It finds and improves an approximation dataset to better reflect the original data distribution by using the multiplicative weighs update rule. The weight for each data record answer to desired queries in the approximation dataset will scale up or down depending on its contribution to the query result. The queries are sampled and measured by using the exponential mechanism and Laplace mechanism, which guarantee differential privacy. This work only considers the privacy solution to a set of linear queries.

Some of the research works applied the conditional probabilistic model to synthetic data generation. A new privacy notion, “ Plausible Deniability” [7], has been proposed and achieved by applying a privacy test after generating the synthetic data. The generative model proposed in this paper is a probabilistic model which captures the joint distribution of features based on correlation-based feature selection (CFS) [11]. The original data is transformed into synthetic data by using conditional probabilities in a probabilistic model. This work also proved that by adding some randomizing function to “ Plausible Deniability” can guarantee differential privacy. The synthetic data generated by this approach contains a portion of the original data, which may lead to a potential privacy issue. PrivBayes [8] generated the synthetic data by releasing a private Bayesian network. Starting from a randomly selected feature node, it extends the network iteratively by selecting a new feature node from the parent set using the exponential mechanism. They also applied the Laplace mechanism on the conditional probability to achieve the private Bayesian network. Bayesian network is a good approach for discrete data. However, using the encoding method to represent the dataset containing many numerical data introduced more noise to synthetic datasets.

Some works form a generative model on preprocessed original data. A non-interactive private data release method has been proposed in [9]. They projected the data into a lower dimension and proved it is nearly Gaussian distribution. After projecting the original data, the Gaussian mixture model (GMM) is used to model the original data based on the estimated statistical information. The synthetic data is generated by adding differential privacy noise on GMM parameters. However, there is some information loss while projection the data into a lower dimension. Josep Domingo-Ferrer et al. [10] proposed the release of a differentially private dataset based on microaggregation [12], which reduced the noise required by differential privacy based on k-anonymity [13]. They clustered original data into clusters, and records in each cluster will be substituted by a representative record (mean) computed by each cluster. The synthetic data is generated by applying the Laplace mechanism to representative records. They also mentioned the idea to consider the feature relationship while clustering, but the paper does not come up with a detailed methodology. Using the representative records as seed data to generate synthetic data may lose some variance in the original datasets, which makes the synthetic data not accurate.

DPGAN [14], BGAN [15] and PATE-GAN [16] proposed differential private synthetic data release method based on generative adversarial network (GAN) [17]. Differentially private generative adversarial network (DPGAN) add noise on the gradient and clip the weight during the training process based on Wasserstein GAN to guarantee the privacy. PATE-GAN proposed private aggregation of teacher (PATE) ensembles teacher–student framework to generate synthetic data. It initialized k teacher models and k data subsets, and each teacher is trained to discriminate between the original and fake data using the corresponding subset. The student model is trained on the data label by ensemble results with the noise of teacher models. Then, the data generator is trained to fool the student model to get the synthetic data. However, using deep learning structures would lead to high demand for original data and a time-consuming training process. A recent benchmarking of differential private synthetic data generation [18] shows that the GAN-based methods are not good at preserving the critical statistical information from original data distributions. This paper focuses on non-image data and how to preserve the statistical characteristic. Thus, the GAN-based mechanisms are not considered in the comparison.

3. Preliminary

This section introduces the background knowledge of this paper.

3.1. Differential privacy

Differential privacy proposed by D.work et al. [4] is one of the most popular privacy definitions which guarantee strong privacy protection on individual samples. The formal definition of ϵ- differential privacy for synthetic datasets generation is defined as below:

Definition 1 (*Differential Privacy*)**.** A randomized mechanism M provides ϵ - differential privacy, if for any pair of datasets D_1, D_2

[3] https://github.com/mingchenli/MCGEN-Private-Synthetic-Data-Generator

that differ in a single record (neighboring datasets) and for any possible synthetic data outputs S ⊂ Range(M), it satisfies

$$\frac{Pr(M(D_1) = S)}{Pr(M(D_2) = S)} \leq exp(\epsilon). \tag{1}$$

The privacy parameter (privacy budget) ϵ is used to present the privacy level achieved by the randomized mechanisms M. The privacy budget ranges from 0 to ∞. If ϵ = 0, the probability of outputting synthetic dataset S for D_1 and D_2 are exactly the same. It ensures the perfect privacy guarantee, which means D_1, D_2 cannot be distinguished by observing the synthetic datasets; if $\epsilon = \infty$, the synthetic datasets generated from D_1 and D_2 are always look different (like original datasets), there is no privacy guarantee at all. If differential privacy has been applied to synthetic datasets, it is hard to distinguish whether a specific individual is in the original dataset by observing synthetic datasets. In other words, differential privacy makes the synthetic dataset plausible. For example, there is a patient record dataset and a new patient record (privacy information) has been added recently. After publishing the differential private synthetic dataset, the data users cannot infer the information of the new patient. It is because the original dataset with (D_1) or without (D_2) the new patient are most likely to generate the synthetic dataset(S).

In general, a differential privacy mechanism can be achieved by adding noise to the output (synthetic dataset) with a privacy parameter between 0 and 1. The mechanism used to achieve ϵ - differential privacy is called differentially private sanitizer and it is always associated with the sensitivity, also known as L_1 - Sensitivity, defined as follows:

Definition 2 (L_1 - *Sensitivity*)**.** For a function $f : D^n \rightarrow \mathbb{R}^d$ which maps datasets to real number domain, the sensitivity of the function f for all neighboring datasets pairs D, D' is defined as follows:

$$\Delta D = \max_{D,D'} \|f(D) - f(D')\|. \tag{2}$$

Laplace mechanism achieves ϵ-differential privacy by adding noise drawn from Laplace distribution [19] to the output.

Definition 3 (*Laplace Mechanism*)**.** For a function $f : D^{n*d} \rightarrow \mathbb{R}^{n*d}$ which maps datasets to real number domain, the mechanism M defined in the following equation provides ϵ-differential privacy:

$$M(D) = f(D) + Laplace(\Delta f/\epsilon). \tag{3}$$

3.2. Agglomerative hierarchical clustering

An essential part of our methodology is hierarchical clustering [20]. Hierarchical clustering is an algorithm that clusters input samples into different clusters based on the proximity matrix of samples. The proximity matrix contains the distance between each cluster. Agglomerative hierarchical clustering is a bottom-up approach. It starts by assigning each data sample to its own group and merges the pairs of clusters that have the smallest distance to move up until there is only a single cluster left.

3.3. Microaggregation

Microaggregation [12] is a kind of dataset anonymization algorithm that can achieve k-anonymity. There are several settings for microaggregation, notice, the microaggregation mentioned here is a simple heuristic method called maximum distance to average record (MDAV) proposed by Domingo-Ferrer et al. [12]. MDAV clusters samples into clusters, in which each cluster contains exactly k records, except the last one. Records in the same

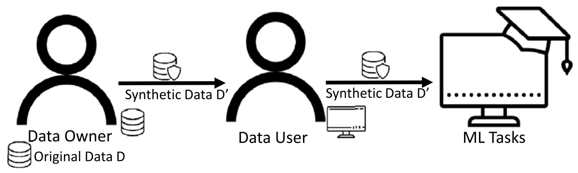


Fig. 1. Synthetic Data Use Case.

cluster are supposed to be as similar as possible in terms of distance. Furthermore, each record in the cluster will be replaced by a representative record of the cluster to complete the data anonymization.

3.4. Multivariate Gaussian generative model

The multivariate Gaussian generative model keeps the multivariate Gaussian distribution [21], which is parameterized by the mean μ and covariance matrix Σ. Formally, the density function of multivariate Gaussian distribution is given by:

$$f(x) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} exp(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)). \tag{4}$$

Data samples drawn from the Multivariate Gaussian generative model are under Multivariate Gaussian distribution.

4. Methodology

In this section, we present the methodology of our approach.

4.1. Problem statement

Given a numerical dataset $D^{n\times d}$ (n samples and d features) which contains sensitive privacy records. The data owner expects to share dataset D to an untrust party in a secure way. In this paper, we aim to use the synthetic dataset D' to substitute the original dataset D in data sharing to prevent privacy leakage. The synthetic dataset D' not only maintains some certain information in original datasets but also protected by a certain level of privacy (see Fig. 1).

4.2. Methodology overview

Fig. 2 illustrates the design of our approach. It includes four processes worked collectively to generate the synthetic dataset which satisfies differential privacy:

- Data preprocessing: Combining independent feature sets and microaggregation [12] (multi-level clustering) to produce data clusters.
- Statistic extraction: Extract the representative statistical information from each data cluster.
- Differential privacy sanitizer: Introduce differential private noise on extracted statistical information.
- Data generator: Generate synthetic data sample by sample from the noisy generative model.

We will explain each component in our design in the following sections.

4.3. Data preprocessing

The conventional way to generate data clusters only groups the data at the sample level. For example, microaggregation

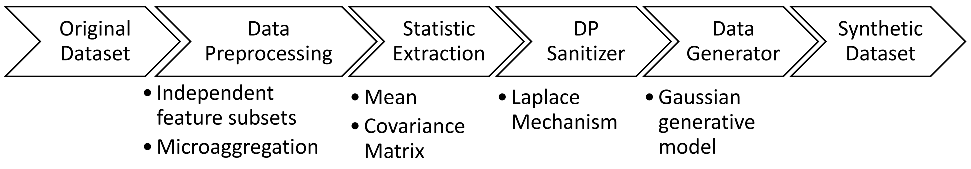


Fig. 2. MC-GEN Algorithm.

(MDAV) [12] cluster the data in full feature dimension and add the differentially private noise on the representative records. Two kinds of errors may be introduced to jeopardize the utility of this process:

- The false feature correlation introduced by sample level clustering. When modeling the output clusters from sample level clustering, some clusters may carry some correlation which not exist while looking into all data samples. This false feature correlation may apply unnecessary constraints when modeling the data clusters, which may lead the synthetic data into a different shape.
- The noise variance introduced by DP mechanism. Intuitively, the less noise introduced from differential privacy results in higher utility. Hence, reducing the noise from DP mechanism also helps us to improve the data utility.

To smooth these two errors, we design multi-level clustering that combines independent feature sets (IFS) with microaggregation in our approach consecutively. Namely, we not only cluster the data at the sample level but also at the feature level. Feature level clustering helps the generative model to capture the correct correlation of features. Compared to sample level clustering, using multi-level clustering also reduced the total noise variance introduced to the synthetic datasets, detailed proof shown in Section 4.4.

4.3.1. Feature level clustering

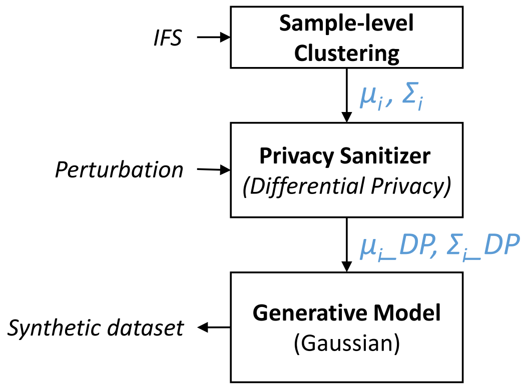
Given a numerical dataset $D^{n \times d}$, we divided data features into m independent feature sets using agglomerative hierarchical clustering. A distance function that converts Pearson correlation to distance has been designed to form the proximity matrix in hierarchical clustering. Features that have a higher correlation should have a lower distance and a lower correlation corresponding to the higher distance. This approach results in that features in the same set are more correlated to each other and less correlated to the features in other feature sets. However, hierarchical clustering needs to specify the number of clusters to be divided. We use Davies–Bouldin [22] index to choose the best split from all possible numbers of clusters. The distance function in the proximity matrix is shown below:

$$d = 2n[1 - Corr(X, Y)], \tag{5}$$

where Corr(X, Y) is the Pearson correlation between two random variables, e.g., feature pair in original datasets.

4.3.2. Sample level clustering

Based on the output of feature level clustering, we applied microaggregation [12] on each independent feature set (IFS). The purpose of microaggregation is to assign the homogeneous samples to the same cluster, which preserves more information from the original data. On the other hand, referring to [12], sensitivity on each sample cluster can be potentially reduced compared to the global sensitivity. This reduction will result in involving less noise in the differential privacy mechanism. In other words, it enhances the data utility under the same level of privacy guarantee.



IFS
Sample-level Clustering
μ_i, Σ_i
Perturbation
Privacy Sanitizer (Differential Privacy)
μ_i_DP, Σ_i_DP
Synthetic dataset
Generative Model (Gaussian)

Fig. 3. Synthetic Data Generator.

4.4. Statistic extraction and privacy sanitizer

4.4.1. Statistic extraction

For a given dataset D, features are divided into m IFSs on the feature level. In each IFS, the data with selected features is clustered into j clusters by microaggregation on the sample level. The combination of feature and sample level clustering is called multi-level clustering. It outputs $m \times j$ clusters in which each cluster has at least k records. The statistical information is extracted from each cluster to generate the synthetic data.

4.4.2. Privacy sanitizer

Assuming each cluster forms a multivariate Gaussian distribution, the mean (μ_i) and covariance matrix (Σ_i) are computed for each cluster c_i. To ensure differential privacy, the generative model is built based on μ_i_DP and Σ_i_DP which are achieved by the privacy sanitizer, as shown in Fig. 3. The privacy sanitizer adds differentially private noise on μ_i and Σ_i. Algorithm 1 illustrates the process of privacy sanitizer. n_{mean} and n_{covarM} denote the noise perturbation on mean and covariance matrix.

Algorithm 1: MC-GEN privacy sanitizer

```
  Input: $\mu_i$, $\Sigma_i$
  Output: $\mu_i\_DP$, $\Sigma_i\_DP$
1 for each cluster: do
2     $\mu_i$, $\Sigma_i \leftarrow c_i$;
3     $n_{mean}$ $\leftarrow$Draw $d$ samples from $Lap(0, \frac{\Delta}{|c_i|\epsilon_m})$;
4     $n_{covarM}$ $\leftarrow$Draw $\frac{d^2+d}{2}$ samples from $Lap(0, \frac{\Delta}{|c_i|\epsilon_m})$;
5     $\mu_i\_DP = \mu_i + n_{mean}$ ;
6     $\Sigma_i\_DP = \Sigma_i + n_{covarM}$;
7     return $\mu_i\_DP$, $\Sigma_i\_DP$
```

The differentially private noise added on μ_i and Σ_i suppose to be as little as possible to preserve the utility of synthetic data. It refers to the generative model captures the statistical information more precisely. Thus, it is very critical to investigate and control the noise variance introduced by privacy sanitizer. A contribution of MC-GEN is applying multi-level clustering reduces the overall noise introduced from differential privacy mechanism compared to sample level clustering. The following proof demonstrates the noise variance of multi-level clustering.

Theorem 1. *The noise variance introduced by multi-level clustering on mean vector* μ *is* $\sum_{m=1}^{m}\sum_{j=1}^{j}\frac{\Delta IFS_m}{|C_j^m|\epsilon}d$.

Proof. If multi-level clustering is applied, dataset D has been vertically partitioned into m IFSs with corresponding data, and data in each IFS has been clustered into j clusters by microaggregation.

Let ΔIFS_m denotes the L_1 - sensitivity of m_{th} IFS, C_j^m denotes the j_{th} cluster in IFS_m, d_m denotes the size of IFS_m, and d denotes the total number of features. The noise variance N_{IFS_m} of m_{th} IFS is the sum of noise variances on each cluster in this IFS. Noise on a single cluster by microaggregation has been shown in [10]. Thus, noise variance N_{IFS_m} can be written as:

$$N_{IFS_m} = \sum_{j=1}^{j}\frac{\Delta IFS_m}{|C_j^m|\epsilon_m}d_m, \tag{6}$$

where the ϵ has been divided proportionally based on the size of each IFS:

$$\begin{cases}\epsilon_m = \dfrac{d_m}{d}\epsilon \\ \epsilon = \epsilon_1 + \epsilon_2 + \cdots + \epsilon_m.\end{cases} \tag{7}$$

The noise on each IFS is independent, and the total noise variance on the mean vector by multi-level clustering N_1 is the sum of noise on all IFSs.

$$N_1 = \sum_{m=1}^{m} N_{IFS_m} = N_{IFS_1} + N_{IFS_2} + \cdots + N_{IFS_m} \tag{8}$$

Based on Eq. (6) and Eq. (7), use addition commutative to rewrite Eq. (8) as following:

$$N_1 = \sum_{m=1}^{m}\sum_{j=1}^{j}\frac{\Delta IFS_m}{|C_j^m|\epsilon}d. \quad \square \quad \square \tag{9}$$

Theorem 2. *The noise variance introduced by multi-level clustering on covariance matrix* Σ *is* $\sum_{j=1}^{j}\sum_{m=1}^{m}\frac{\Delta IFS_m}{|C_j^m|\epsilon}d_m^2$.

Proof. Use the same notation in Theorem 1 to describe the noise variance on the covariance matrix:

$$N_{IFS_m} = \sum_{j=1}^{j}\frac{\Delta IFS_m}{|C_j^m|\epsilon_m}d_m^2. \tag{10}$$

The noise on each IFS is independent, and the total noise variance on the mean vector by multi-level clustering N_2 is the sum of noise on all IFSs.

$$N_2 = \sum_{m=1}^{m} N_{IFS_m} = N_{IFS_1} + N_{IFS_2} + \cdots + N_{IFS_m} \tag{11}$$

Based on Eq. (7) and (10), rewrite Eq. (11):

$$N_2 = \sum_{m=1}^{m}\sum_{j=1}^{j}\frac{\Delta IFS_m}{|C_j^m|\epsilon}d_m^2 \quad . \quad \square \quad \square \tag{12}$$

The following proof demonstrates the noise variance of sample level clustering.

Theorem 3. *The noise variance introduced by sample level clustering on mean vector* μ *is* $\sum_{j=1}^{j}\frac{\Delta D}{|C_j|\epsilon}d$.

Proof. If sample level clustering is applied, dataset D has been clustered into j clusters. Let ΔD denote the L_1 - sensitivity of dataset D, and C_j denote the j_{th} cluster. Refer to [10], the total noise variance on the mean vector by sample level clustering N_3 can be written as follows:

$$N_3 = \frac{\Delta D}{|C_1|\epsilon}d + \frac{\Delta D}{|C_2|\epsilon}d + \cdots + \frac{\Delta D}{|C_j|\epsilon}d = \sum_{j=1}^{j}\frac{\Delta D}{|C_j|\epsilon}d. \quad \square \quad \square \tag{13}$$

Theorem 4. *The noise variance introduced by sample level clustering on covariance matrix* Σ *is* $\sum_{j=1}^{j}\frac{\Delta D}{|C_j|\epsilon}d^2$.

Proof. Using the same notation in Theorem 3, the total noise variance on the covariance matrix by sample level clustering N_4 can be written as follows:

$$N_4 = \frac{\Delta D}{|C_1|\epsilon}d^2 + \frac{\Delta D}{|C_2|\epsilon}d^2 + \cdots + \frac{\Delta D}{|C_j|\epsilon}d^2 = \sum_{j=1}^{j}\frac{\Delta D}{|C_j|\epsilon}d^2. \quad \square \quad \square \tag{14}$$

Proposition 1. *The overall noise introduced by multi-level clustering* N_{MC} *is less than the overall noise introduced by sample clustering* N_{SC}.

Proof. Since the differentially private noise is applied to the mean and covariance matrix respectively. The overall noise introduced on multi-level clustering can be written as:

$$N_{MC} = N_1 + N_2. \tag{15}$$

The overall noise introduced on sample level clustering can be written as:

$$N_{SC} = N_3 + N_4. \tag{16}$$

Since data in each IFS has been clustered by microaggregation with the same cluster size, which means:

$$C_j^m = C_j^{m+1}. \tag{17}$$

and each cluster j is a subset of the original dataset:

$$d^m < d. \tag{18}$$

Eq. (9) can be rewritten as:

$$\begin{aligned} N_1 &= \sum_{m=1}^{m}\sum_{j=1}^{j}\frac{\Delta IFS_m}{|C_j^m|\epsilon}d \\ &= \sum_{j=1}^{j}\frac{\Delta IFS_1 + \Delta IFS_2 + \cdots + \Delta IFS_m}{|C_j|\epsilon}d \\ &= \sum_{j=1}^{j}\frac{\Delta D}{|C_j|\epsilon}d = N_3. \end{aligned} \tag{19}$$

Eq. (12) can be rewritten as:

$$\begin{aligned} N_2 &= \sum_{m=1}^{m}\sum_{j=1}^{j}\frac{\Delta IFS_m}{|C_j^m|\epsilon}d_m^2 \\ &= \sum_{j=1}^{j}\frac{d_1^2\Delta IFS_1 \ldots + d_m^2\Delta IFS_m}{|C_j|\epsilon} \\ &< \sum_{j=1}^{j}\frac{\Delta IFS_1 + \Delta IFS_2 + \cdots + \Delta IFS_m}{|C_j|\epsilon}d^2 \\ &= \sum_{j=1}^{j}\frac{\Delta D}{|C_j|\epsilon}d^2 = N_4. \end{aligned} \tag{20}$$

Based on Eqs. (19) and (20), we can know that $N_{MC} < N_{SC}$. $\square$

Table 1
Baseline of experiments.

Dataset	Task		
	SVM	Logistic regression	Gradient boosting
Scenario 1			
Diabetes [23]	0.80	0.78	0.75
Adult [24]	0.84	0.89	0.91
Phishing [25]	0.96	0.98	0.96
Scenario 2			
Diabetes [23]	0.82	0.80	0.85
Adult [24]	0.85	0.86	0.83
Phishing [25]	0.96	0.94	0.94

Using feature level clustering not only mitigates the false feature correlation error but also helps to reduce the noise variance. Generally, the synthetic datasets generated with less noise will have better utility.

4.5. Synthetic data generator

The original multivariate Gaussian model is parameterized by mean (μ_i) and covariance matrix (Σ_i). The algorithm 1 outputs two parameters μ_i_DP and Σ_i_DP that are protected by differential privacy. Hence the multivariate Gaussian model that is parameterized by μ_i_DP and Σ_i_DP is also protected by differential privacy. Depending on the post-processing invariance of different privacy, all the synthetic data derived from differential private multivariate Gaussian models are also protected by differential privacy. The synthetic data is synthesized sample by sample from the private multivariate Gaussian generative models with the noisy mean (μ_i_DP) and covariance matrix (Σ_i_DP) based on Eq. (4). Features of each synthetic data sample are randomly drawn from private multivariate Gaussian models. These multivariate Gaussian generative models are determined by the multiple-level clustering result of corresponding original data.

5. Experimental evaluation

In this section, we show the experimental design and discuss the results of our approach.

5.1. Experiment setting

To evaluate the performance of the proposed method, we have implemented MC-GEN based on JAVA 8. We generated synthetic datasets and performed experiments under different cluster sizes (k) and different privacy parameters (ϵ). The setting of ϵ varies from 0.1 to 1 and cluster size k is picked proportionally (20%, 40%, 60%, 80%, 100%) based on the corresponding seed dataset. For each synthetic dataset, we evaluate the performance of three classification tasks: support vector machine (SVM), logistic regression, and gradient boosting. The classification models are implemented using python scikit-learn library [26,27]. We also compared our method with other private synthetic data release methods. All the experiments are performed in two scenarios:

- Original data training, synthetic data testing (Scenario 1): The classification algorithm trains on original data and tests on the synthetic data. For each experimental dataset, 20% of the samples are used as seed dataset to generate the synthetic dataset, and 80% is used as original data to train the model.
- Synthetic data training, original data testing (Scenario 2): The classification algorithm trains on synthetic data and tests on the original data. For each experimental dataset, 80% of the samples are used as seed dataset to generate the synthetic dataset, and 20% is used as original data to test the model.

5.2. Experiment dataset

We conducted three datasets in our experiments from public sources (e.g. UCI repository [28], kaggle, libsvm datasets [29]) to examine the performance of different classification algorithms. For dataset that contains categorical variables, we use one-hot encoding to convert it to numerical variables. All the features in the dataset are scaled to [−1,1]:

5.2.1. Diabetes

Diabetes dataset [23] contains the diagnostic measurements of patient records. It has 768 samples and 8 features. The features include some patient information, such as blood pressure, BMI, insulin level, and age. The objective is to identify whether a patient has diabetes.

5.2.2. Adult

Adult dataset [24] is extracted from the census bureau database. It has 48,842 samples and 14 features. The features include some information, such as age, work class, education, sex, and capital gain/loss. The objective is to predict the annual salary (over 50k or not) of each person.

5.2.3. Phishing

Phishing dataset [25] is used to predict phishing websites. It has 11,055 samples and 30 features. The features include information related address, HTML and JavaSrcipt, such as website forwarding, request URL, and domain registration length.

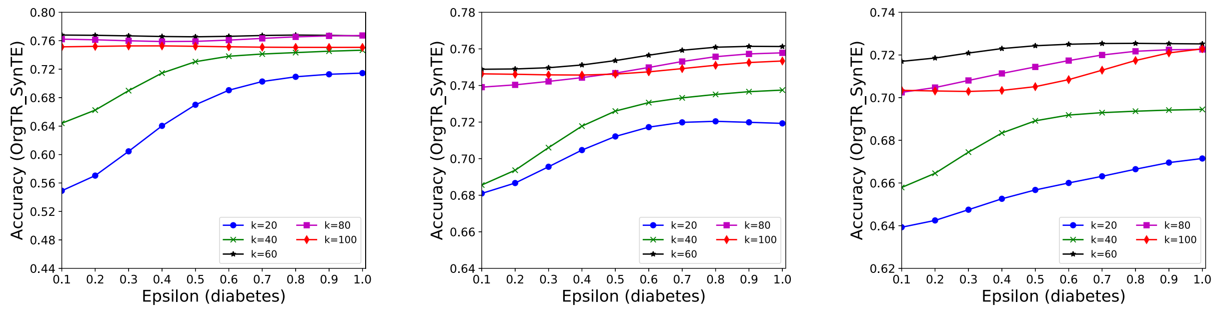
5.3. Experiment setup and metric

In each round of experiments, we assume the data owners randomly select data from the original dataset as seed data. The size of seed data depends on the scenario. The seed data is input to MC-GEN to generate the synthetic dataset under different parameter combinations (ϵ and k). Then, the synthetic dataset is tested on three classification tasks to evaluate the performance. Accuracy is used as the evaluation metric, shown in Eq. (21). For each scenario on each dataset, we ran 100 rounds to get the average performance.

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions} \tag{21}$$

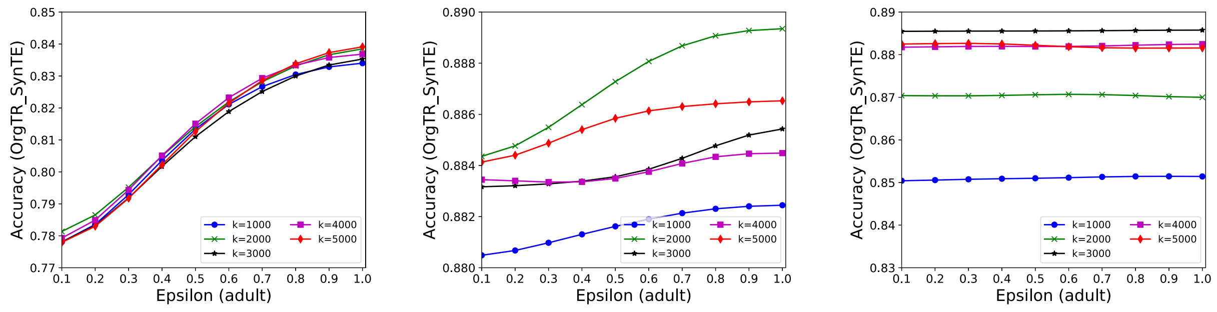
5.4. Effectiveness analysis

In this section, we evaluate the effectiveness of our approach on three classification datasets with three classification algorithms. The best performance of MC-GEN on each dataset is always very close to the baseline (Table 1). Baselines were trained and tested on the corresponding model using original datasets. Each baseline takes the average performance of 100 experiments.



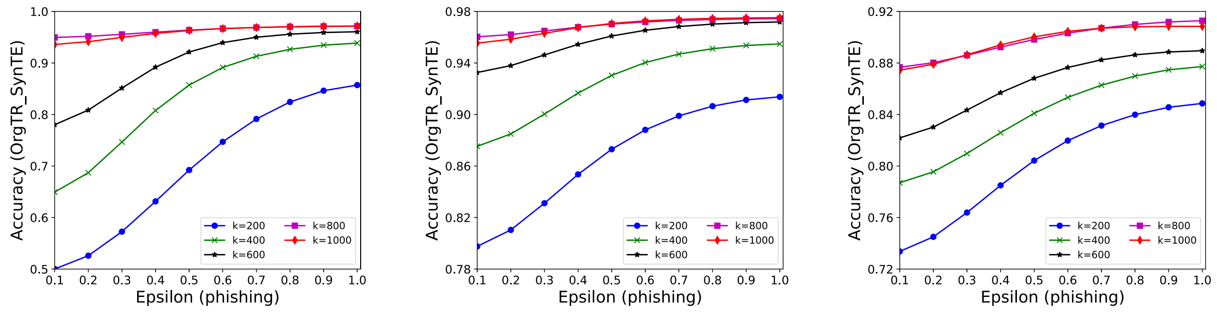
(a) SVM (b) Logistic regression (c) Gradient boosting

Fig. 4. Effect of cluster size (k) on diabetes dataset in scenario 1.



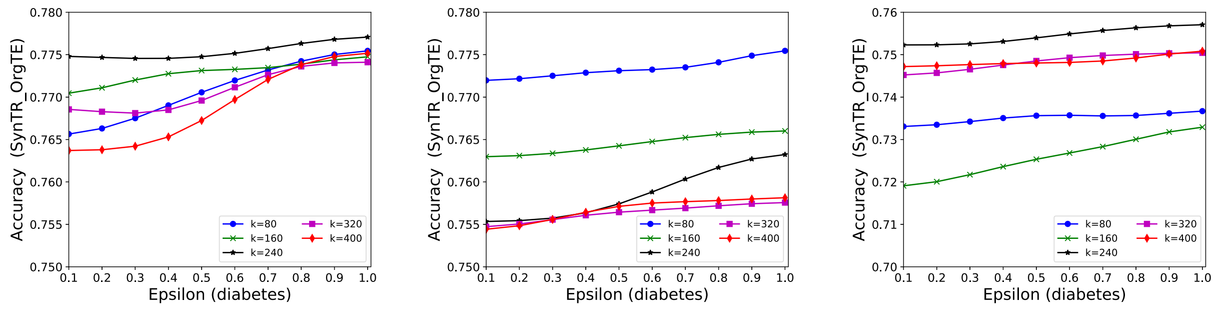
(a) SVM (b) Logistic regression (c) Gradient boosting

Fig. 5. Effect of cluster size (k) on adult dataset in scenario 1.



(a) SVM (b) Logistic regression (c) Gradient boosting

Fig. 6. Effect of cluster size (k) on phishing dataset in scenario 1.



(a) SVM (b) Logistic regression (c) Gradient boosting

Fig. 7. Effect of cluster size (k) on diabetes dataset in scenario 2.

5.4.1. *Effectiveness analysis of different parameters*

The performance of MC-GEN is affected by two parameters: privacy budget (ϵ) and cluster size (k). To evaluate the effectiveness of a single parameter, the other parameter remains the same. As shown in Figs. 4, 5, 6, 7, 8 and 9, we observe that:

(1) Privacy budget (ϵ): This parameter controls the noise variance on the synthetic datasets. In DP mechanism, greater ϵ results in smaller-scaled noise, and vice versa. In our experiments, we investigate ϵ from 0.1 to 1. As shown in the results, the performance in all classification tasks increased

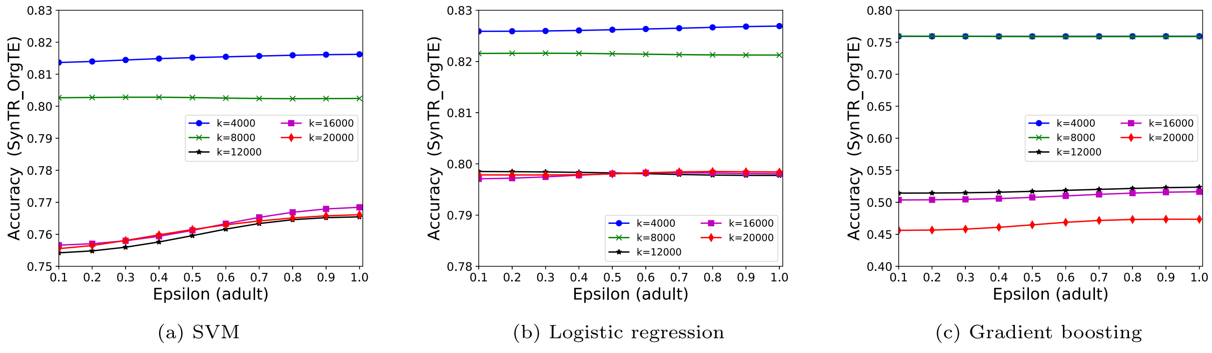


Fig. 8. Effect of cluster size (k) on adult dataset in scenario 2.

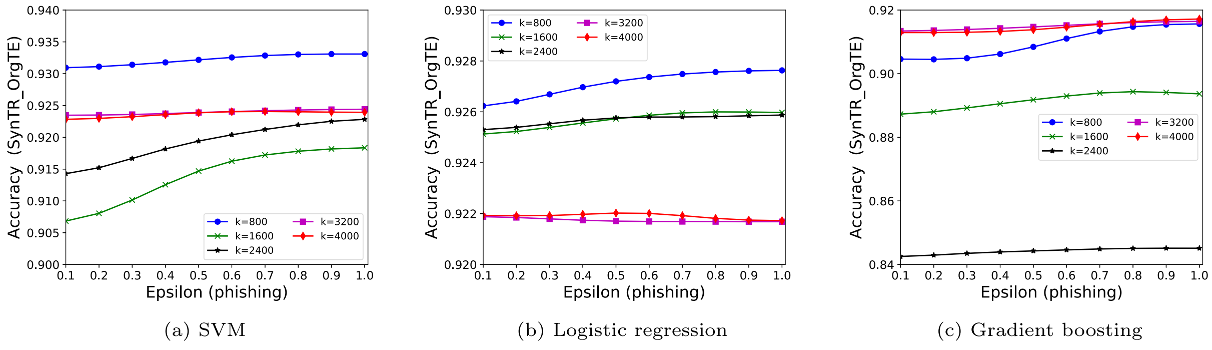


Fig. 9. Effect of cluster size (k) on phishing dataset in scenario 2.

as the privacy budget ϵ increased. While DP sanitizer adds noise on extracted statistical information (Fig. 3), large ϵ (small-scaled noise) make μ_DP and Σ_DP closer to μ and Σ, which results in less noise on synthetic datasets. In other words, the generative model captures more accurate statistical information. Thus, a greater ϵ would result in a better performance.

(2) Cluster size (k): This parameter controls the number of data points in each cluster. Namely, the number of data points used to build each generative model. We investigate it from 20% to 100% based on the total number of seed data. Intuitively, the smaller k would make synthetic data generation more precise since the data is captured in a high-resolution way. However, based on the results shown in Figs. 5, 6, 7, 8 and 9, small k does not always result in the best performance. For instance, Figs. 5(b), 7(b) and 9 (a), (b) show that a moderate k achieve the best performance. Figs. 6 and 9 show that using a large k achieves the best performance. The local optimal k varies on different classification tasks and datasets. As such, it is hard to determine a perfect k that covers all use cases. Finding a local optimal k helps MC-GEN capture the statistical representation of the dataset (clusters) in a good manner. Hence, when the classification task and datasets are assigned, it is worth investigating the local optimal k value for the dataset to ensure performance.

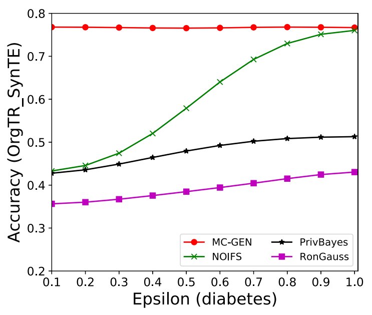
5.4.2. Comparing with existing methods

In this section, we compared MC-GEN with three existing private synthetic data release methods:

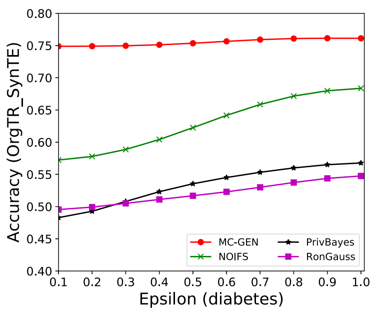
- PrivBayes [8,30]: PrivBayes is designed on the Bayesian network with differential privacy. Starting from a randomly selected feature node, it extends the network iteratively by selecting a new feature node from the parent set using the exponential mechanism. Then it applied the Laplace mechanism on the conditional probability to achieve the private Bayesian network. The synthetic datasets are generated by using the perturbated Bayesian network. PrivBayes [8,30] are evaluated on the code provided by the paper authors.
- RonGauss [9]: RonGauss [9] releases the synthetic data based on Gaussian mixture model. It builds the Gaussian mixture model on the projected data in a lower dimension and generates the synthetic dataset based on the Gaussian mixture model with differential privacy noise. RonGauss [9] are evaluated on the code provided by the paper authors.
- NoIFS: NOIFS follows the same procedure (Fig. 2) as MC-GEN to generate the synthetic data. The difference between NOIFS and MC-GEN is that NOIFS only applies the sample level clustering (MDAV) in the data prepossessing. Comparing it with MC-GEN helps us to evaluate the effectiveness of feature level clustering.

The performance of MC-GEN is evaluated under the local optimal parameter combination. For other methods, we follow the exact settings mentioned in the corresponding paper. Each method is evaluated on an average of 100 experiments. Figs. 10, 11, 12, 13, 14 and 15 show the results of different methods under the same privacy budgets. We can observe that MC-GEN outperforms the other methods in most cases. Besides, there are several findings during the comparison:

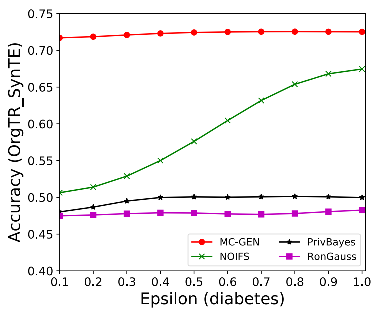
- MC-GEN always outperforms NOIFS due to the reduction in noise variance by feature level clustering (proof in Section 4.4).
- In Fig. 11, PrivBayes has similar performance as MC-GEN, even outperforms on SVM. However, in Fig. 14, MC-GEN outperforms PrivBayes consistently. It may cause by the number of seed data and data types in the datasets. The seed



(a) SVM

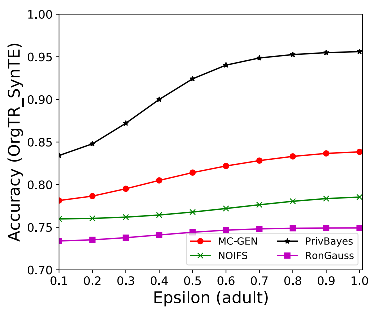


(b) Logistic regression

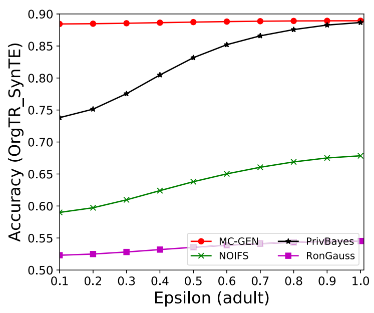


(c) Gradient boosting

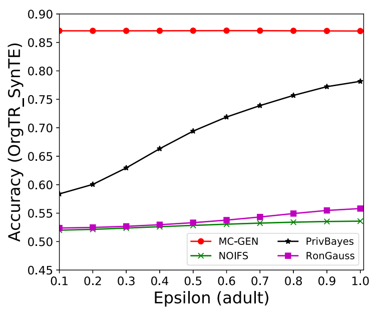
Fig. 10. Comparison with other generation methods on diabetes dataset in scenario 1.



(a) SVM

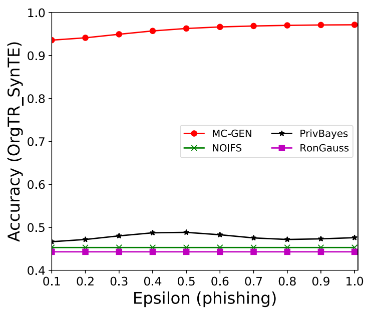


(b) Logistic regression

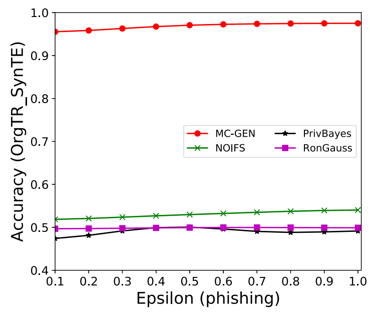


(c) Gradient boosting

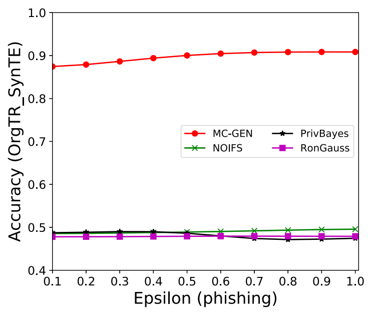
Fig. 11. Comparison with other generation methods on adult dataset in scenario 1.



(a) SVM

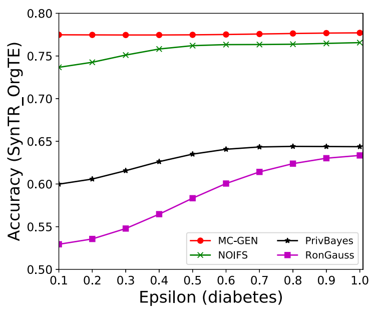


(b) Logistic regression

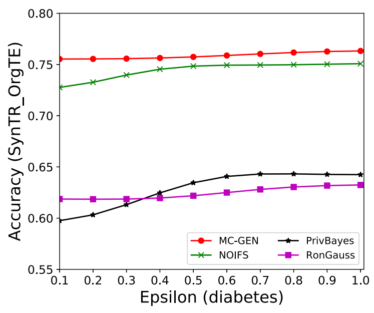


(c) Gradient boosting

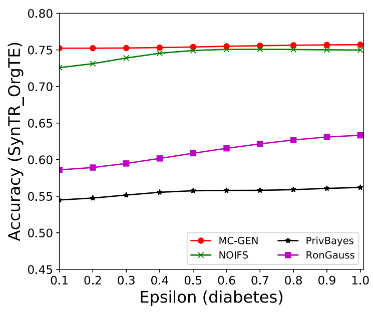
Fig. 12. Comparison with other generation methods on phishing dataset in scenario 1.



(a) SVM

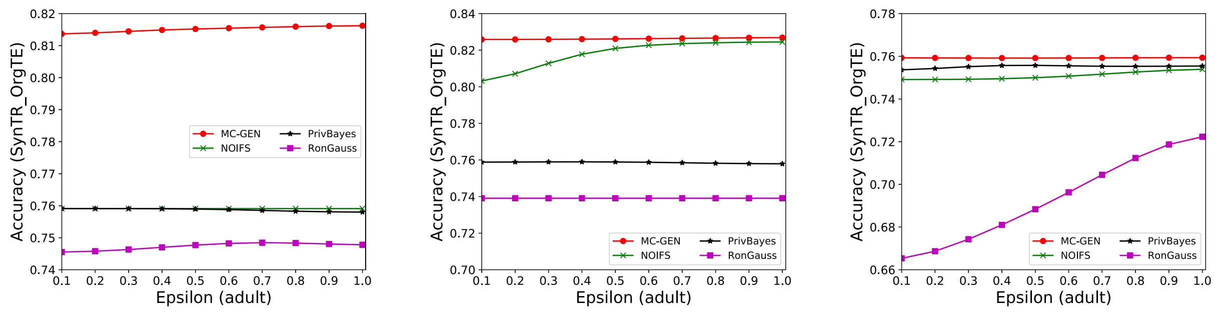


(b) Logistic regression



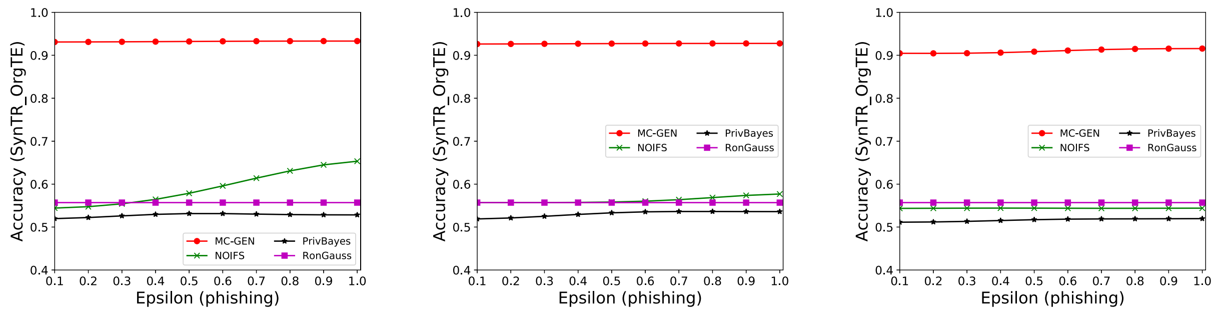
(c) Gradient boosting

Fig. 13. Comparison with other generation methods on diabetes dataset in scenario 2.



(a) SVM (b) Logistic regression (c) Gradient boosting

Fig. 14. Comparison with other generation methods on adult dataset in scenario 2.



(a) SVM (b) Logistic regression (c) Gradient boosting

Fig. 15. Comparison with other generation methods on phishing dataset in scenario 2.

data in scenario 1 is much less than in scenario 2. Limited seed data may affect the performance of MC-GEN. On the other hand, most of the features in adult datasets are categorical features. The Bayesian network can be more effective than the multivariate Gaussian model on categorical data. Therefore, the utility of MC-GEN can be diminished on the small categorical dataset.

5.5. Time complexity of MC-GEN

The time complexity of MC-GEN is discussed in this section. In MC-GEN, there are 4 major components: feature level clustering, sample level clustering, privacy sanitizer, and generative model. Assuming the original dataset has n samples and d features, feature level clustering vertically divided the dataset into m IFSs. Then, each IFS with corresponding data is horizontally clustered into j clusters. At this point, the original dataset is separated into $m \times j$ clusters and each cluster has k samples. In the end, the privacy sanitizer and generative model are applied simultaneously on each cluster. Since privacy sanitizer and generative model take effect at the same time, we consider it as the same part while discussing the time complexity. Table 2 illustrates the detailed time complexity of each component. Feature level clustering mainly relies on agglomerative hierarchical clustering which has time complexity $O(d^2 log_d)$. The time complexity of sample level clustering can be found in [31]. For privacy sanitizer and generative model, it applies the noise on data in each cluster, so the time complexity should be $O(mjk)$. Since $n = j \times k$, the time complexity can be rewritten as $O(mn)$. All the components are sequentially applied, the total time complexity of MC-GEN is $O(d^2 log_d + n^2 + mn)$.

Table 2
Time complexity of MC-GEN.

Component	Time complexity
Feature level Clustering	$O(d^2 log_d)$
Sample level Clustering	$O(n^2)$
Privacy Sanitizer and Generative Model	$O(mn)$

6. Conclusion

We proposed a novel and effective synthetic data generation method, MC-GEN, which targets on generating a private synthetic dataset for data sharing. We demonstrate MC-GEN improves the utility of synthetic datasets by using multi-level clustering. In the experimental evaluation, we show the effectiveness of synthetic datasets generated by MC-GEN and investigate the parameter effect of MC-GEN. With the best parameter settings, synthetic datasets generated by MC-GEN can achieve similar performance as original datasets. Moreover, we compared MC-GEN with three existing methods. The experimental results show that MC-GEN outperforms the other existing methods in terms of utility. In the future, we will apply and enhance the proposed method on different data types and different machine learning tasks. Meanwhile, as the complexity of data increases, a more powerful and precise model is worth exploring.

CRediT authorship contribution statement

Mingchen Li: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing – original draft, Writing – review & editing, Visualization. **Di**

Zhuang: Conceptualization, Methodology, Formal analysis, Writing – review & editing. **J. Morris Chang:** Conceptualization, Writing – review & editing, Supervision, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

[1] E. Talavera, G. Iglesias, Á. González-Prieto, A. Mozo, S. Gómez-Canaval, Data augmentation techniques in time series domain: A survey and taxonomy, 2022, arXiv preprint arXiv:2206.13508.

[2] C. Dwork, Differential privacy: A survey of results, in: International Conference on Theory and Applications of Models of Computation, Springer, 2008, pp. 1–19.

[3] M. Hardt, K. Ligett, F. McSherry, A simple and practical algorithm for differentially private data release, in: Advances in Neural Information Processing Systems, 2012, pp. 2339–2347.

[4] C. Dwork, Differential privacy, Encyclopedia Cryptogr. Secur. (2011) 338–340.

[5] K. Taneja, T. Xie, DiffGen: Automated regression unit-test generation, in: Proceedings of the 2008 23rd IEEE/ACM International Conference on Automated Software Engineering, IEEE Computer Society, 2008, pp. 407–410.

[6] D. Su, J. Cao, N. Li, M. Lyu, PrivPfC: differentially private data publication for classification, VLDB J. 27 (2) (2018) 201–223.

[7] V. Bindschaedler, R. Shokri, C.A. Gunter, Plausible deniability for privacy-preserving data synthesis, Proc. VLDB Endow. 10 (5) (2017) 481–492.

[8] J. Zhang, G. Cormode, C.M. Procopiuc, D. Srivastava, X. Xiao, Privbayes: Private data release via bayesian networks, ACM Trans. Database Syst. 42 (4) (2017) 25.

[9] T. Chanyaswad, C. Liu, P. Mittal, Coupling dimensionality reduction with generative model for non-interactive private data release, 2017, arXiv preprint arXiv:1709.00054.

[10] J. Soria-Comas, J. Domingo-Ferrer, Differentially private data sets based on microaggregation and record perturbation, in: Modeling Decisions for Artificial Intelligence, Springer, 2017, pp. 119–131.

[11] M.A. Hall, Correlation-Based Feature Selection for Machine Learning, University of Waikato Hamilton, 1999.

[12] J. Domingo-Ferrer, V. Torra, Ordinal, continuous and heterogeneous k-anonymity through microaggregation, Data Min. Knowl. Discov. 11 (2) (2005) 195–212.

[13] L. Sweeney, k-anonymity: A model for protecting privacy, Int. J. Uncertain. Fuzziness Knowl.-Based Syst. 10 (05) (2002) 557–570.

[14] L. Xie, K. Lin, S. Wang, F. Wang, J. Zhou, Differentially private generative adversarial network, 2018, http://dx.doi.org/10.48550/ARXIV.1802.06739, URL https://arxiv.org/abs/1802.06739.

[15] S. Imtiaz, M. Arsalan, V. Vlassov, R. Sadre, Synthetic and private smart health care data generation using GANs, in: 2021 International Conference on Computer Communications and Networks, ICCCN, 2021, pp. 1–7, http://dx.doi.org/10.1109/ICCCN52240.2021.9522203.

[16] J. Yoon, J. Jordon, M. van der Schaar, PATE-GAN: Generating synthetic data with differential privacy guarantees, in: International Conference on Learning Representations, 2019, URL https://openreview.net/forum?id=S1zk9iRqF7.

[17] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, Adv. Neural Inf. Process. Syst. 27 (2014).

[18] Y. Tao, R. McKenna, M. Hay, A. Machanavajjhala, G. Miklau, Benchmarking differentially private synthetic data generation algorithms, CoRR, 2021, arXiv:2112.09238.

[19] C. Dwork, F. McSherry, K. Nissim, A. Smith, Calibrating noise to sensitivity in private data analysis, in: Theory of Cryptography Conference, Springer, 2006, pp. 265–284.

[20] S.C. Johnson, Hierarchical clustering schemes, Psychometrika 32 (3) (1967) 241–254.

[21] P. Ahrendt, The multivariate gaussian probability distribution, Tech. Rep., Technical University of Denmark, 2005, p. 203.

[22] D.L. Davies, D.W. Bouldin, A cluster separation measure, IEEE Trans. Pattern Anal. Mach. Intell. (2) (1979) 224–227.

[23] R.A. Rossi, N.K. Ahmed, The network data repository with interactive graph analytics and visualization, in: AAAI, 2015, URL https://networkrepository.com.

[24] Adult, 1996, UCI Machine Learning Repository.

[25] R. Mohammad, L. McCluskey, Phishing websites, 2015, UCI Machine Learning Repository.

[26] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, J. Mach. Learn. Res. 12 (2011) 2825–2830.

[27] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, G. Varoquaux, API design for machine learning software: experiences from the scikit-learn project, in: ECML PKDD Workshop: Languages for Data Mining and Machine Learning, 2013, pp. 108–122.

[28] D. Dua, C. Graff, UCI Machine Learning Repository, University of California, Irvine, School of Information and Computer Sciences, 2017, URL http://archive.ics.uci.edu/ml.

[29] C.-C. Chang, C.-J. Lin, LIBSVM: a library for support vector machines, ACM Trans. Intell. Syst. Technol. 2 (3) (2011) 27.

[30] J. Zhang, G. Cormode, C.M. Procopiuc, D. Srivastava, X. Xiao, PrivBayes: Private data release via Bayesian networks, SIGMOD '14, Association for Computing Machinery, New York, NY, USA, 2014, http://dx.doi.org/10.1145/2588555.2588573.

[31] A. Oganian, J. Domingo-Ferrer, On the complexity of optimal microaggregation for statistical disclosure control, Stat. J. U.N. Econ. Comm. Eur. 18 (4) (2001) 345–353.