GAN-Based Differentially Private Publication of Vertically Partitioned Data

Yufei Zhang

Master's thesis

GAN-Based Differentially Private Publication of Vertically Partitioned Data

Yufei Zhang

November 11, 2022

Chair of Data Processing
Technische Universität München

Yufei Zhang. *GAN-Based Differentially Private Publication of Vertically Partitioned Data.* Master's thesis, Technische Universität München, Munich, Germany, 2022.

Supervised by Prof. Dr.-Ing. Klaus Diepold and Michael Moosmeier; submitted on November 11, 2022 to the Department of Electrical and Computer Engineering of the Technische Universität München.

Chair of Data Processing, Technische Universität München, 80290 München, Germany, http://www.ldv.ei.tum.de/.

Abstract

This thesis pays attention to the scenarios where the user data are distributed among multiple parties (e.g., companies and institutes). Each party possesses a different set of attributes of the same individuals. The parties aim to collaborate with others to publish a private joint dataset for better decision-making or data-driven analysis. Motivated by the challenge of prohibition of illegal collection of private data, this thesis intends to provide a novel deep-learning-based solution for the privacy-preserving publication of vertically partitioned data. More specifically, this study proposes a so-called Vertically Distributed General Adversarial Network (VDGAN) framework, which would be trained jointly among multiple parties and generate synthetic joint data to replace the integrated real one for data mining tasks. What's more, two Differential Privacy (DP) protocols are introduced to the framework to provide strong privacy guarantees. Extensive experiments are conducted to evaluate the proposed framework in terms of data utility and privacy preservability with the help of four public datasets with different attribute domains. The results present an apparent improvement compared with the state-of-the-art tree-based models, while some limitations for practical applications should also be considered.

Contents

Abstract **3**

1. Introduction **7**
1.1. Research Questions 9
1.2. Thesis Contribution 9
1.3. Thesis Outline 10

2. Related Works **11**
2.1. Vertical Data Publication under Privacy 11
2.2. Synthetic Data Generation with DP 12
2.2.1. Non-Neural-Network Approach 12
2.2.2. Neural-Network Approach 12

3. Background **15**
3.1. Differential Privacy 15
3.1.1. Vanilla Mechanisms 15
3.1.2. Composition Theorem 16
3.1.3. Rényi Differential Privacy 17
3.1.4. Deep Learning with Differential Privacy 18
3.2. Data Generative Models 20
3.2.1. GAN 20
3.2.2. WGAN-GP 21
3.3. Baseline Method 22
3.3.1. Data Preparation 22
3.3.2. Generation of Latent Tree Model 23
3.3.3. Data Synthesis 24

4. Methodology **25**
4.1. Problem Statement 25
4.2. Method Overview 25
4.3. Vertically Distributed GAN 26
4.3.1. An Iteration Process 26
4.3.2. Warm-Up before Training 28
4.4. DP Protocols 29
4.4.1. DPSGD-Based Approach 29
4.4.2. PATE-Based-Approach 31

4.4.3. Analysis of Privacy Budgets 35

5. Experiments **37**
5.1. Datasets 37
5.2. Metrics 38
5.2.1. Utility 38
5.2.2. Privacy Preservability 40
5.3. Setup 41
5.4. Evaluation and Discussion 41
5.4.1. Results of Utility 42
5.4.2. Results of Privacy Preservability 48
5.5. Ablations 52
5.5.1. Extension to Multiple Clients 52
5.5.2. Extension to Different Number of Teachers for VDGAN-PATE . . 53

6. Conclusion **57**

List of Figures **66**

List of Algorithms **67**

A. Additional Results **69**

1. Introduction

The past decade has seen the rapid development of digitization technologies. "Data" has become an essential strategical resource like "oil". The big-data-driven analysis plays a vital role for a wide range of industries to make decisions, detect anomalies, and power products. Companies and organizations are able to collect a huge amount of data from their customers. It is common for many users or parties to hold their personal private data, and the local information can be integrated into a huge database. In another view, the valuable dataset is distributed among multiple partitions.

According to the characteristics of the joint database, there are two kinds of partition strategies as shown in Fig. 1.1. In horizontal partition, the local datasets are collected from different individuals while they share the same attributes, which define categories and characteristics of the data. An example is that a car manufacturer collected massive driving trajectory data. The data are recorded by the same types of sensors, while the samples come from different drivers. In contrast, the vertically partitioned data hold identical user IDs while the attributes are distributed among local parties. The study in this thesis aims to focus on the circumstance of vertical partition. It is of great value to aggregate these attributes together and use the joint database for better data mining and decision-making performance. For example, a medical researcher wants to study a potential correlation between travel experiences and certain types of illnesses, so they need to collect data from hospital and airline companies.

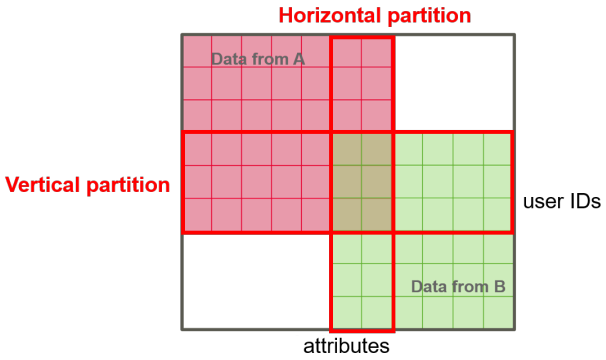


Figure 1.1: Partitioned Data

However, the aggregation of the local databases faces a dilemma. As shown in Fig. 1.2, the private databases cannot be directly shared with a third-party curator

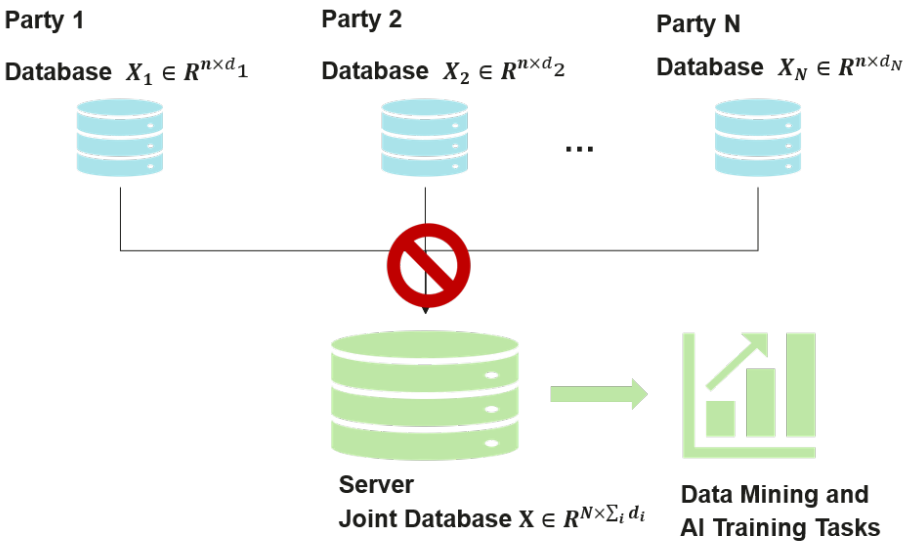


Figure 1.2: Publication of Vertically Distributed Data

due to the potential leakage of sensitive personal information. Moreover, even if each party can publish its privatized database via some protection techniques, it may break the cross-attribute correlations and reduce the utility of the joint database. To solve the above problems, I proposed to use synthetic data for data analysis work instead of real data. Specifically, I will design an effective generative model in vertically distributed settings so that the third-party curator can get thousands of synthetic data that preserve the joint attributes' statistical distribution and can be utilized for data mining tasks. However, there are still inevitable risks of leaking sensitive information from the real data because they have to be involved in the optimization process of the generative model. So the model should be trained in a safe manner with the help of some privacy techniques.

So far, very little attention has been paid to a generative model in vertical settings. Previous studies [28, 40, 41, 49] are almost restricted to the taxonomy- tree model, which aims at approximating the statistical distribution of the real data directly. The performances would decline drastically with the increasing dimensions of attributes. Besides, they suffer from an overload of computation for complicated datasets. So this thesis intends to investigate a Generative Adversarial Network (GAN) in vertical settings, which as a deep-learning-based framework would learn the cross-correlations among the high dimensional attributes automatically, and has achieved outstanding performance for image synthesis [27, 36].

Differential Privacy (DP), as one of the state-of-the-art privacy-enhancing technolo-

gies, has been widely applied in privacy-preserving data publication. It provides strong theoretical guarantees against various kinds of attacks. It seeks to limit the influence of each individual and prevent the model from memorizing the sensitive information of the input data. The term DP will be implemented in each local party so that its private data are protected during the collaborated training with others.

The objective of this approach is set out as follows: Using the proposed framework, the third-party curator can get synthetic data that preserves the distribution and utility of the vertically partitioned real one as well as does not reveal sensitive information.

1.1. Research Questions

This work focuses on the following issues in light of the background and motivations above.

- Framework Realization
  - How to deploy the GAN model into the vertically distributed settings?
  - How to implement DP into the model training process? Can we try multiple protocols of DP?
- Framework Evaluation
  - How to evaluate the synthetic data comprehensively in terms of the similar utility as the real one and the privacy strength against attackers?
  - What are the advantages, limitations, and suggested scenarios of application of our framework compared with the state-of-the-art solutions?

1.2. Thesis Contribution

The contributions of this thesis can be summarized as follows:

- I propose Vertically Distributed GAN (VDGAN), a practical generative framework constructed among multiple local parties and a third-party curator and is able to publish synthetic data with all local attributes.
- I modify the framework to approaches VDGAN-DPSGD and VDGAN-PATE with two DP protocols to provide privacy guarantees. DP will be applied locally to avoid privacy leakages from local model updates.
- I use several public datasets and extensive metrics to evaluate the performance in both utility and privacy aspects. The results show that the proposed frameworks outperform in utility for complicated datasets and strike an excellent utility-privacy balance compared with the published works.

1.3. Thesis Outline

The thesis is composed of five themed chapters. Chapter 2 presents an overview of the previous studies for publishing vertically distributed data under privacy and the state-of-the-art methods of generating synthetic data with DP. Chapter 3 begins by clarifying the core concepts of DP and GAN as the preparation knowledge for this thesis, and also demonstrates the main principle of the work [41], which will be implemented as a baseline for comparison. Chapter 4 consists of the critical methodologies to construct the proposed framework VDGAN-DPSGD and VDGAN-PATE as well as the related algorithms in detail. Chapter 5 is concerned with a series of experiments to extensively evaluate the proposed frameworks' performance and address findings. Lastly, in Chapter 6 a summary of the whole work is given.

2. Related Works

This chapter first reviews the previous works for publishing vertically partitioned data under some privacy strategies in Sec. 2.1, then it gives an overview of the studies to generate synthetic data under Differential Privacy (DP), which includes approaches of vanilla machine learning models in Sec. 2.2.1 and neuron-network-based General Adversarial Networks (GAN) in Sec. 2.2.2.

2.1. Vertical Data Publication under Privacy

To publish vertically partitioned data safely, Jiang *et al.* [18] and Mohammed *et al.* [29] realize privacy guarantee via the k-anonymity model [39], which refers that an individual cannot be distinguished from the group with a size smaller than k. However, their works are limited to two parties, and these models are susceptible to multiple privacy attacks and thus provide weak privacy protection. Wang *et al.* [48] propose a hybrid DP framework to learn generalized linear models from vertically partitioned data. Chen *et al.* [3] use autoencoders [30] and upload latent features under privacy at each party; then they train a global classifier at the server side. Xu *et al.* [51] collect and analyze data jointly from multiple parties under local DP [7]. However, these works only focus on aggregating and analyzing VPD instead of generating synthetic data.

Mohammed *et al.* [28] propose a taxonomy-tree framework called DistDiffGen, the first solution to generate differentially private VPD. Wang *et al.* [49] propose an improved framework called arbDistDP that can release arbitrarily partitioned data. However, these frameworks are limited to two parties and apply only to classification tasks for small value domains because of the top-down specialization algorithm during training. Tajeddine *et al.* [40] train a mixture model on VPD using differentially private variational inference, which can calculate the posterior joint probability distribution of VPD and generate synthetic data from the derived distribution. However, this framework lacks experimental results for the quality of synthetic data and is limited to low-dimensional datasets. Tang *et al.* [41] propose a hierarchical model called differentially private latent tree (DPLT) to analyze the joint distribution among attributes from multiple parties under DP. However, this framework also struggles with worse cross-attribute correlation and higher computational costs for high-dimensional datasets.

2.2. Synthetic Data Generation with DP

2.2.1. Non-Neural-Network Approach

An intuitive idea is to calculate the statistical distribution approximating the given real data under privacy and sample synthetic data based on that. Zhang *et al.* [52] propose the framework PrivBayes that uses a Bayesian Network to model the correlation of high-dimensional attributes. Qardaji *et al.* [35] construct marginal contingency tables. Hardt *et al.* use the multiplicative weights framework to maintain and improve the approximated distribution for a set of counting queries. However, these methods consume too much privacy budget during model construction and make the distribution approximation inaccurate and thus leading to synthetic data with low utility. Moreover, the computation complexity increases fast on high-dimensional data.

2.2.2. Neural-Network Approach

Deep-learning-based model is widely used for generating synthetic data. Instead of calculating the probability distribution of given data, the neural-network-based models [2, 12, 30] are trained to produce synthetic data directly, which have high similarity with the real one. The Autoencoder-based model is hard to be modified into a vertically distributed setting, so we focus on GAN-based frameworks.

Xie *et al.* [50] propose the DP-GAN framework, which firstly provides DP strategies during training the GAN framework. They use the algorithm DPSGD[1], which applies gradient clipping and noise injection while updating the discriminator model. Zhang *et al.* [54] improve the framework using WGAN-GP[14] as well as several optimization strategies, e.g. adaptive clipping and warm starting. Torkzadehmahani *et al.* [44] apply DPSGD on conditional GAN [27] to privately generate synthetic data with specific labels. For distributed training, Augenstein *et al.* propose the DP-FevAvg-GAN framework, which has one global generator on the server side and multiple local discriminators on the clients side. The real data is preserved locally and only the parameters of each discriminator need to be uploaded. Chen *et al.* modify the training process and upload fake data instead of model parameters to the server and realize the differential privacy for the generator. However, most of these frameworks only focus on image data and use Convolutional Neural Networks(CNN) inspired by DCGAN [36] because of the outstanding performance of GAN on high-structured data.

For generating tabular data, Harder *et al.* [15] and Liew *et al.* [21] train the generator by optimizing the random feature representation of Maximum Mean Discrepancy [13] between real and synthetic data and use vanilla Gaussian Mechanism of DP instead of DPSGD. However, these models aim to minimize the distance on a private embedding space instead of the original high-dimensional space. Tantipongpipat *et al.* [42] use an auxiliary Autoencoder [30] to convert discrete attributes into continuous space to fit the training process of GAN. Torfi *et al.* [43] use a similar Autoencoder with 1D-CNN

to capture correlated input attributes. However, to get synthetic data, the output of the generator must be decoded by the decoder, which is hard to be applied in vertical settings. Zhao *et al.* [55, 56] use multiple strategies, e.g., using an auxiliary classifier, modifying generator loss inspired by [34] to fit the tabular input. However, those tricks are also hard to be used in vertically distributed settings.

Apart from DPSGD, there are also some works based on Privacy Aggregation of Teacher Ensembles (PATE) [32, 33] to provide privacy protection. Jordon *et al.* [19] propose the PATE-GAN framework, which consists of multiple teacher discriminators and one student discriminator as well as one generator. The student discriminator is only trained on records that are produced by the generator and labeled by the teacher discriminators. However, it relies on the assumption that the generator would be able to generate the entire real records space to bootstrap the training process. If most of the synthetic records are labeled as fake by the teacher discriminators, the student discriminator would be trained on a biased and fail to learn the true data distribution. Long *et al.* [22] propose G-PATE that applies a private gradient aggregation on the information that flows from the teacher discriminators to the single generator. Furthermore, it implements random projection and discretization before aggregation to reduce the privacy budget consumed by each step. Wang *et al.* improve the framework by compressing gradients. However, it is challenging to analyze the convergence of these two frameworks, and a large number of teachers leads to low training efficiency.

3. Background

This chapter will recall the concepts and definitions of the preliminaries related to the proposed framework in this thesis. Section 3.1 introduces the fundamentals of DP, including features, mechanisms, and applications in deep learning. Section 3.2 demonstrates the principle of the GAN model as well as a state-of-the-art variant. Lastly, Section 3.3 clarifies the methodology of the baseline [41].

3.1. Differential Privacy

Differential Privacy (DP) [9] is a privacy-preserving algorithm introduced by Dwork *et al.* in 2006. It aims at randomizing the answers of querying on dataset so that avoids the information leakage of each record of the dataset even if the adversaries have enough background knowledge.

Definition 1 ((ϵ, δ)-differential privacy [8, 9]) *A randomized mechanism $\mathcal{M}$ guarantees (ϵ, δ)-DP for every set of outputs Ω, and for any neighbouring datasets D and D' differing in one record, if $\mathcal{M}$ satisfies*

$$Pr[\mathcal{M}(D) \in \Omega] \;\leq\; exp(\epsilon) \cdot Pr[\mathcal{M}(D') \in \Omega] \;+\; \delta \tag{3.1}$$

Two datasets D and D' are called neighbouring datasets if they differ only in one record. The parameter ϵ denotes the privacy budget in this mechanism. If ϵ is small, the probabilities of the same output from a pair of neighbouring datasets are similar, Which means the mechanism $\mathcal{M}$ masks the differences between two queries if one record is deleted from or added to the dataset. The lower ϵ, the stronger the privacy level. The parameter δ allows some freedom to violate the strict inequality by a small probability. If $\delta = 0$, the mechanism $\mathcal{M}$ guarantees $\epsilon - DP$.

3.1.1. Vanilla Mechanisms

Generally, differential privacy is achieved by adding some noise to the querying output. A specific amount of noise over a given privacy budget is calculated to perturb the difference between the neighbouring datasets in the worst case.

Definition 2 (Sensitivity [9]) *Given a query function $f : D \rightarrow \mathbb{R}$, for any neighbouring datasets D and D', the sensitivity of f is defined as*

$$S(f) \;=\; \max_{D,D'} \|f(D) - f(D')\|_1 \tag{3.2}$$

Here $\|\cdot\|_1$ represents the L1-norm. The sensitivity denotes the maximal change of magnitude between two queries on neighbouring datasets. It differs in various query functions. The higher sensitivity, the more noise needs to be added to realize the same privacy guarantee.

There are three fundamental mechanisms: the Laplace mechanism [9] and Gaussian mechanism [11] for numeric queries, and the exponential mechanism [11] for non-numeric queries. Here only the Gaussian mechanism related to our work is shown.

Definition 3 (Gaussian Mechanism [11]) *Given a query function $f : D \rightarrow \mathbb{R}$ over a dataset D, a randomized mechanism $\mathcal{M}$ satisfies (ϵ, δ)-differential privacy if*

$$\mathcal{M}(D) \;=\; f(D) \;+\; \mathcal{N}(0, \sigma^2) \tag{3.3}$$

where $\mathcal{N}(0, \sigma^2)$ denotes the noise sampled from a Gaussian distribution with a mean of 0 and a standard deviation of $\sigma = \frac{S_2(f)\sqrt{2\ln(1.25/\delta)}}{\epsilon}$. Here the L2-distance $S_2(f) = \|f(D) - f(D')\|_2$ is used to describe the sensitivity.

3.1.2. Composition Theorem

When a dataset is accessed multiple times via different DP mechanisms, each of them has its own privacy guarantee. The ultimate privacy budget consumed is calculated according to the following composition theorems.

Theorem 1 (Sequential Composition [24]) *Given a set of privacy mechanisms $\mathcal{M} = \mathcal{M}_1, \cdots, \mathcal{M}_m$, $\mathcal{M}$ will provide $(\sum \epsilon_i, \sum \delta_i)$-differential privacy guarantee if $\mathcal{M}$ is sequentially performed on a dataset, and each $\mathcal{M}_i$ satisfies (ϵ_i, δ_i)-differential privacy.*

Theorem 2 (Parallel Composition [24]) *Given a set of privacy mechanisms $\mathcal{M} = \mathcal{M}_1, \cdots, \mathcal{M}_m$, $\mathcal{M}$ will provide $\max \epsilon_i$-differential privacy guarantee if each $\mathcal{M}_i$ satisfies (ϵ_i, δ_i)-differential privacy on a disjointed subset of the entire dataset.*

Sequential composition is used when a series of mechanisms access the same dataset repeatedly. The final privacy budget equals the sum of each budget. In comparison, the parallel composition is applied when the querying targets are disjointed subsets for several mechanisms. The final privacy budget equals the maximum of their budgets. However, the two elemental compositions are valid only when each query is conducted independently. If the adversary can adaptively affect the input dataset of the queries, the mechanisms dependent on each other should follow the Strong Composition theorem [10].

3.1.3. Rényi Differential Privacy

Mironov *et al.* proposed a variant of differential privacy based on the concept of Rényi Divergence [37]. It provides a tighter bound of required noise to satisfy the privacy intensity. Compared with the standard definition, Rényi differential privacy mitigates the parameter explosion after the budget adding-up by composition theorems.

Definition 4 ((α, ϵ)-Rényi differential privacy [25]) *A randomized mechanism $\mathcal{M}$ gives ϵ-Rényi differential privacy of order α for any neighbouring datasets D and D' differing only in one record, if $\mathcal{M}$ satisfies*

$$D_\alpha(\mathcal{M}(D) \parallel \mathcal{M}(D')) = \frac{1}{\alpha - 1} \log \mathbb{E}_{\boldsymbol{x} \sim \mathcal{M}(D)} \left(\frac{Pr[\mathcal{M}(D) = \boldsymbol{x}]}{Pr[\mathcal{M}(D') = \boldsymbol{x}]} \right)^{\alpha - 1} \leq \epsilon \tag{3.4}$$

Here $D_\alpha(P \parallel Q)$ refers to the Rényi divergence [37] for two probability distributions P and Q and the parameter α denotes the order of the divergence. When $\alpha = \infty$, the (α, ϵ)-RDP is equivalent to the strict ϵ-DP with $\delta = 0$. And the Rényi differential privacy with a finite α implies (ϵ, δ)-DP with $\delta > 0$. The relationship between them is as follows:

Theorem 3 (From RDP to DP [25]) *If a privacy mechanism $\mathcal{M}$ guarantees (α, ϵ)-RDP, then $\mathcal{M}$ guarantees (ϵ', δ)-DP for any $\delta \in (0, 1)$ where $\epsilon' = \epsilon + \frac{\log 1/\delta}{\alpha - 1}$*

After the transform from RDP to DP, the consumed privacy budget decreases by $\frac{\log \delta}{\alpha - 1}$, which implies a tighter upper bound from a certain amount of noise.

The Gaussian Mechanism under Rényi differential privacy has a much simpler version.

Theorem 4 (Gaussian Mechanism in RDP [25]) *Given a query function $f : D \to \mathbb{R}$ over a dataset D, a randomized mechanism $\mathcal{M}$ satisfies $\left(\alpha, \frac{s^2 \alpha}{2\sigma^2}\right)$-RDP if*

$$\mathcal{M}(D) = f(D) + \mathcal{N}(0, \sigma^2) \tag{3.5}$$

where s refers to the L2-sensitivity.

Given the sensitivity s and the distribution parameter σ of the injected noise, the consumed privacy budget ϵ can be calculated over different orders α.

Similar to the standard notion, the Rényi differential privacy also satisfies sequential composition when a series of mechanisms access the same dataset. However, the adding-up property works not only for independent mechanisms but also when the latter mechanism depends on the output of the formal one adaptively.

Theorem 5 (Composition in RDP [25]) *Given a sequence of m adaptive mechanisms $\mathcal{M} = \mathcal{M}_1, \cdots, \mathcal{M}_m$, if each each $\mathcal{M}_i$ satisfies (α, ϵ_i)-RDP, $\mathcal{M}$ will provide $(\alpha, \sum \epsilon_i)$-RDP guarantee.*

3.1.4. Deep Learning with Differential Privacy

Deep neuron networks are widely used for solving machine learning tasks. It consists of several layers with many parameters and non-linear active functions. With the training data as input, a loss function describes the difference between the outputs and the target results. By minimizing the loss values via back-propagation and gradient descent iteratively, the optimal parameters can be found, and the networks are well-trained. When training a deep-learning network with differential privacy, the consumed privacy budget ϵ for each iteration should be accumulated together according to the composition theorem, which leads to a dramatic increase in the total budget. So searching for the appropriate methods to apply differential privacy protocols into deep neuron networks has been a popular area in recent years. The proposed frameworks in this thesis are based on the following two protocols.

Differentially Private Stochastic Gradient Descent

The Differentially Private Stochastic Gradient Descent (DPSGD) proposed by Abadi [1] is the most common method at present to train a differential private neuron network. It aims to privatize the model parameters by adding gaussian noise to the gradients during back-propagation. In common cases of complex networks, for each iteration, only a mini-batch of data is involved in performing the stochastic gradient descent for faster convergence. So in each epoch of DPSGD, a batch of training data is randomly selected without replacement for computing the loss values. Moreover, before adding noise, all gradients would be clipped within a threshold so that the influence of any neuron is bounded. It aims to control the global sensitivity and limit the required noise in the worst case. It ensures that for each epoch, the harmful impact of noise on the model training process is within some degree. Finally, the clipped gradients in this batch are added with some noise and then averaged to perform the gradient descent. The distribution of the sampled noise is decided by a given scaling parameter σ and the clipping threshold. The details of DPSGD are described in Alg. 1.

The training dataset is accessed once for each epoch, so the consumed privacy budget is accumulated using some accounting method during the iterative training process. A primary method is to use the Strong Composition Theorem [10] to calculate the upper bound of the budget. Considering the particular noise distribution, the authors of DPSGD also invented a more advanced method called moments accountant [1], which provides a tighter bound and saves more privacy budgets. In the open-source library Tensorflow [23], DPSGD is implemented as the module Tensorflow-Privacy, where RDP is used as an improved privacy accounting method [26].

Algorithm 1 Outline of DPSGD

Require: $\mathcal{L}$: loss function; T: training round; m: batch size; γ: learning rate; σ: noise scale; C: clipping bound;

for $t = 1, \cdots, T$ **do**

Sample a batch of data M_t randomly from training data $\{\boldsymbol{x}_1, ..., \boldsymbol{x}_N\}$ without replacement in size m

for $\boldsymbol{x}_i \in M_t$ **do**

$\boldsymbol{g}_t(\boldsymbol{x}_i) \leftarrow \nabla_{\theta_t} \mathcal{L}(\theta_t, \boldsymbol{x}_i)$

$\bar{\boldsymbol{g}}_t(\boldsymbol{x}_i) \leftarrow \boldsymbol{g}_t(\boldsymbol{x}_i) / \max\left(1, \frac{\|\boldsymbol{g}_t(\boldsymbol{x}_i)\|_2}{C}\right)$

end for

$\tilde{\boldsymbol{g}}_t \leftarrow \frac{1}{L} \sum_i \left(\bar{\boldsymbol{g}}_t(\boldsymbol{x}_i) + \mathcal{N}(0, \sigma^2 C^2 \boldsymbol{I})\right)$

$\theta_{t+1} \leftarrow \theta_t - \gamma \tilde{\boldsymbol{g}}_t$

end for

Private Aggregation of Teacher Ensembles

In contrast to adding noise on gradients, Papernot *et al.* proposed a method called Private Aggregation of Teacher Ensembles (PATE) [32, 33]. Fig. 3.1 shows the overview of the PATE workflow. The framework consists of an ensemble of teacher models and a student model. The sensitive data are split into several disjoint subsets without overlapping, each used to train a teacher model independently. After training, an agreement can be achieved for a given input by voting among all teacher models, which would predict the label independently. To avoid the case that the voting results of two classes differ only by one and the final agreement would be changed by one teacher, the calculated Laplacian or Gaussian noise should be carefully added to the voting counts. Apart from the sensitive dataset and the ensemble of teacher models, which are not accessible by an adversary, a public student model will be trained by knowledge distillation from the ensemble side. There are some incomplete public data lacking ground-truth labels. A limited number of queries on the teacher ensemble are

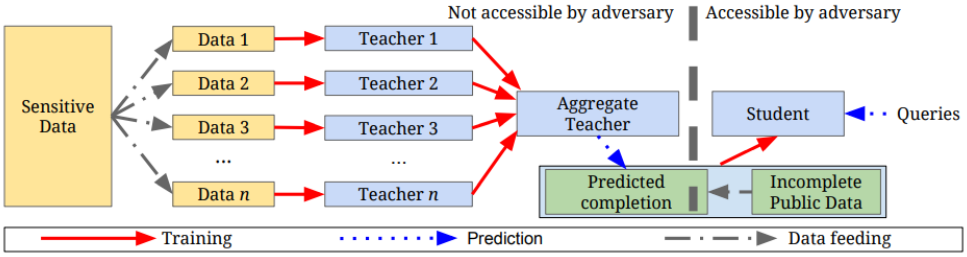


Figure 3.1: Overview of PATE [32]

conducted using the public dataset to get aggregated predictions. The public data and corresponding predictive labels are used to train a student model, which can be queried by adversaries freely.

3.2. Data Generative Models

3.2.1. GAN

Generative Adversarial Nets (GAN) proposed by Goodfellow *et al.* [12] is a deep-learning-based framework for generating synthetic data which has a similar distribution to the given real data. It consists of two neural networks—generator and discriminator as shown in Fig. 3.2. The generator would produce synthetic records from sampled noise, and the discriminator would discriminate synthetic inputs from the real ones. The two models would be trained simultaneously as an adversarial game with the min-imax objective:

$$\min_{G}\max_{D}\ \mathbb{E}_{\boldsymbol{x}\sim\mathbb{P}_r}[log(D(\boldsymbol{x}))]\ +\ \mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_z}[log(1-D(G(\boldsymbol{z})))] \tag{3.6}$$

where the input $\boldsymbol{x}$ is sampled from the real data distribution $\mathbb{P}_r$ and the input $\boldsymbol{z}$ is sampled from a random noise distribution $\mathbb{P}_z$. The generator G would transfer the sampled noise $\boldsymbol{z}$ to synthetic data $\tilde{\boldsymbol{x}} = G(\boldsymbol{z})$. The discriminator D would transfer either real input $\boldsymbol{x}$ or synthetic input $\tilde{\boldsymbol{x}}$ to a single scalar. D is trained to maximize the classification accuracy between the two kinds of inputs. In contrast, G is trained to minimize the probability that its output $G(\boldsymbol{z})$ is assigned with the right label by D. By updating the parameters of the two models alternatively, GAN will get to an optimal equilibrium so that the distribution of synthetic data $\mathbb{P}_g$ will converge to the real one $\mathbb{P}_r$.

An unlimited amount of high-utility data can be sampled via forward propagation from a well-trained GAN framework. Furthermore, only backward propagation is needed during the training process to obtain gradients. No approximate inference or Markov chains are necessary. However, the disadvantages are primarily that there is no explicit representation of $\mathbb{P}_g$, and that D must be synchronized well with G during training. In

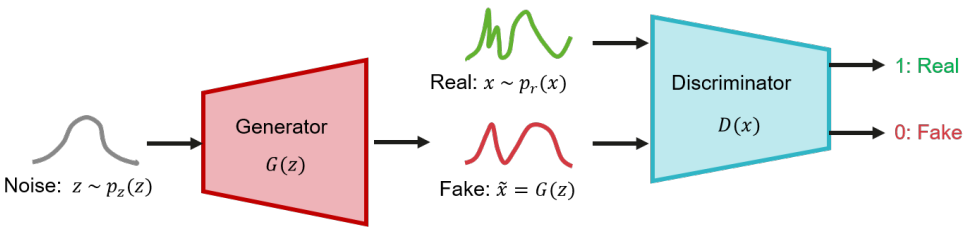


Figure 3.2: Overview of GAN

practice, the discriminator learns very quickly to distinguish between real and fake and becomes saturated, which leads to gradient vanishing and training difficulty of the generator.

3.2.2. WGAN-GP

The work [12] proves that the optimizing process of GAN is equivalent to minimizing the symmetrical Jensen-Shannon(JS) divergence between the real and synthetic distributions $JS(\mathbb{P}_r, \mathbb{P}_g)$. Arjovsky *et al.* [2] argue that the complex training process of GAN is due to the potentially discontinuous JS divergence. They propose a modified framework Wasserstein GAN (WGAN), which uses the Wasserstein-1 (also called Earth-Mover) distance $W(\mathbb{P}_r, \mathbb{P}_g)$ as the optimizing objective instead of the JS divergence. The Wasserstein-1 distance refers to the "cost" of the optimal transport plan from the distribution $\mathbb{P}_r$ into distribution $\mathbb{P}_g$. Given a fixed $\mathbb{P}_r$, it is continuous everywhere and differentiable almost everywhere if the feedforward neural network G is locally Lipschitz and satisfies regularity assumption 1. With the help of the Kantorovich-Rubinstein duality [45], the value function of WGAN is:

$$\min_{G} \max_{D \in \mathcal{D}} \mathbb{E}_{\boldsymbol{x} \sim \mathbb{P}_r}[D(\boldsymbol{x})] - \mathbb{E}_{\tilde{\boldsymbol{x}} \sim \mathbb{P}_g}[D(\tilde{\boldsymbol{x}})] \tag{3.7}$$

where $\mathcal{D}$ is the set of 1-Lipschitz functions. It indicates that under an optimal discriminator, the optimizing process of the generator is to minimize the Wasserstein-1 distance $W(\mathbb{P}_r, \mathbb{P}_g)$. The work realizes the Lipschitz constraint by clipping gradients of the discriminator.

Although the WGAN framework is proven to have better theoretical properties than the vanilla one, Gulrajani *et al.* [14] argue that the implementation of weight clipping leads to some training difficulties. First, the optimizing critic of the discriminator is simplified so that it is harder to capture higher moments of the data distribution. Then the clipping value must be carefully tuned to avoid exploding or vanishing gradients. Thus they propose an alternative solution called Gradient Penalty (GP) to enforce the Lipschitz constraint.

The work [14] proves that a sufficient condition of a discriminator D being 1-Lipschitz is that it has gradients along the straight lines connecting coupled points from $\mathbb{P}_r$ and $\mathbb{P}_g$ with norm 1. Given Eq. 3.7 the new loss function of D is:

$$L_D = \mathbb{E}_{\tilde{\boldsymbol{x}} \sim \mathbb{P}_g}[D(\tilde{\boldsymbol{x}})] - \mathbb{E}_{\boldsymbol{x} \sim \mathbb{P}_r}[D(\boldsymbol{x})] + \lambda \cdot \mathbb{E}_{\hat{\boldsymbol{x}} \sim \mathbb{P}_{\hat{\boldsymbol{x}}}}[(\|\nabla_{\hat{\boldsymbol{x}}} D(\hat{\boldsymbol{x}})\|_2 - 1)^2] \tag{3.8}$$

where the random record $\hat{\boldsymbol{x}} \sim \mathbb{P}_{\hat{\boldsymbol{x}}}$ is sampled uniformly along the straight line between $\tilde{\boldsymbol{x}}$ and $\boldsymbol{x}$. with the help of the penalty coefficient λ, the corresponding gradients are updated towards 1 from two sides. Because the real samples are not used for training the generator, the loss function of G is:

$$L_G = -\mathbb{E}_{\boldsymbol{z} \sim \mathbb{P}_z}[D(G(\boldsymbol{z}))] \tag{3.9}$$

The details of the training process of WGAN-GP are illustrated in Alg. 2

Algorithm 2 Outline of WGAN-GP

Require: n_d: number of iterating steps of D for each iterating step of G; T: training round; m: batch size; γ: learning rate; λ: gradient penalty coefficient

for $t = 1, \cdots, T$ **do**
  for $j = 1, \cdots, n_d$ **do**
    for $i = 1, \cdots, m$ **do**
      Sample real data $\boldsymbol{x} \sim \mathbb{P}_r$, noise $\boldsymbol{z} \sim \mathbb{P}_z$, a random number $\alpha \sim U[0, 1]$
      $\tilde{\boldsymbol{x}} \leftarrow G(\boldsymbol{z})$
      $\hat{\boldsymbol{x}} \leftarrow \alpha \boldsymbol{x} + (1 - \alpha)\tilde{\boldsymbol{x}}$
      $L_D^i = D_{\theta_d}(\tilde{\boldsymbol{x}}) - D_{\theta_d}(\boldsymbol{x}) + \lambda \left(\left\| \nabla_{\hat{\boldsymbol{x}}} D_{\theta_d}(\hat{\boldsymbol{x}}) \right\|_2 - 1 \right)^2$
    end for
    $\mathcal{L}_D \leftarrow \frac{1}{m} \sum_i L_D^i$
    $\theta_d^{j+1} \leftarrow \theta_d^j - \gamma \nabla_{\theta_d} \mathcal{L}_D$
  end for
  Sample a batch of noise $\left\{ \boldsymbol{z}^i \right\}_{i=1}^m \sim \mathbb{P}_z$
  $\mathcal{L}_G \leftarrow \frac{1}{m} \sum_i -D(G_{\theta_g}(\boldsymbol{z}^i))$
  $\theta_g^{t+1} \leftarrow \theta_g^t - \gamma \nabla_{\theta_g} \mathcal{L}_G$
end for

3.3. Baseline Method

There are many approaches in centralized and horizontally distributed settings for differentially private synthetic data generation, but the state-of-the-art researches in vertically distributed settings are limited. One is DistDiffGen [28], which is only suitable for two-parties settings and specific datasets. So we select the other one called DPLT [41] as our baseline. The framework is not published as open-source, so we implement the corresponding algorithms by ourselves.

The DPLT framework represents the joint distribution of given random variables based on a latent tree model. The random noise is injected during model building in the vertical setting to satisfy DP, and the synthetic data are sampled according to the calculated statistical distribution.

3.3.1. Data Preparation

The algorithm of DPLT assumes that all attributes are binary, so each non-binary attribute should be converted into a set of binary attributes. In contrast to the one-hot encoding widely used in neural networks, a specific encoding method [52] is adopted. It encodes each state value of a categorical attribute into a binary representation with multiple bits, and each encoded attribute denotes one bit of the binary number. Such an encoded dataset has lower dimensions than one-hot encoding and provides a high level of flexibility in constructing the tree model. However, the binary output of the

DPLT model should also be decoded to get decimal synthetic data so that the domain size might be enlarged compared with the real one.

3.3.2. Generation of Latent Tree Model

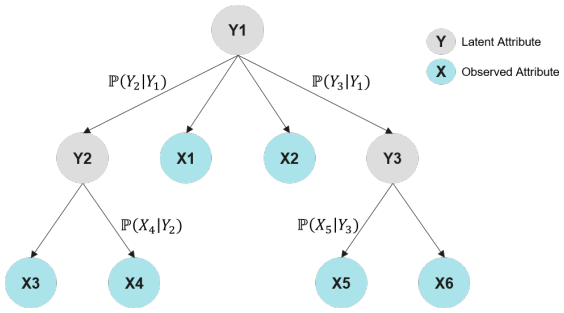


Figure 3.3: Latent Tree Model

The latent tree model [53] shown in Fig. 3.3 is a hierarchical probabilistic graphical model where all internal nodes Y are latent variables and all leaf nodes X are observed variables. Each node is associated with its parent node via the conditional distribution. The approximate joint probability for all variables can be calculated by the set of all conditional distributions. An overview of the DPLT framework is shown in Fig. 3.4. There are four steps to building a latent tree.

1. **Generation of Latent Attributes** Each latent attribute is obtained based on a sub-group of observed attributes with higher correlations. Each client would generate its local latent attributes and send them to the server.

2. **Correlation Quantification** The server and clients calculate the mutual information of latent-attribute pairs together, including local and cross-client pairs.

3. **Construction of Tree Structure** The server generates a maximum spanning tree with the gathered mutual information as edge weights.

4. **Quantification of Tree Parameters** The conditional distribution of each parent-child-node pair in the tree is calculated by the cooperation of the server and clients.

The local datasets participate in step 1, 2 and 4, so to satisfy ϵ-DP, the total privacy budget ϵ is divided into three portions and assigned to each step. In step 1, the Exponential Mechanism is implemented during the grouping of observed attributes; In the other two steps, the Laplace Mechanism is implemented by adding noise to statistical distributions.

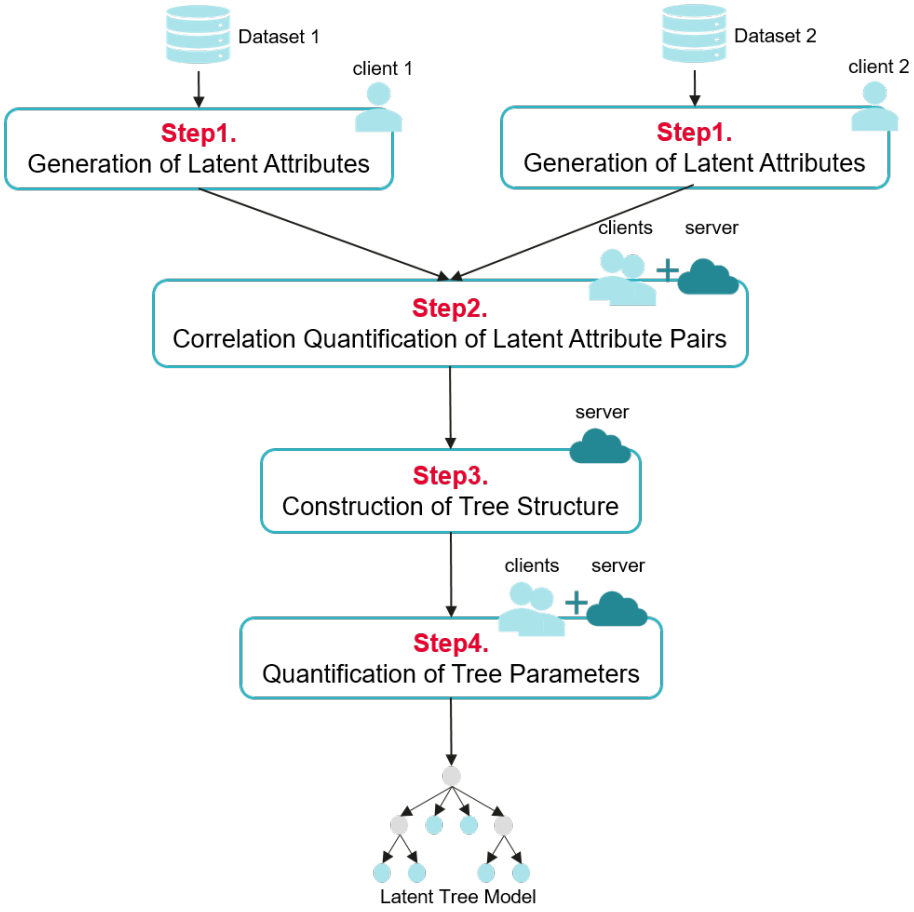


Figure 3.4: Overview of DPLT

3.3.3. Data Synthesis

For categorical data, The domain size of all attributes would increase exponentially with higher dimensions. So in practice, instead of calculating the approximated joint distribution directly, the value of each attribute is sampled one by one from root to leaf according to the known conditional distributions as edges. Finally, the values of all observed attributes would be gathered together as synthetic records.

4. Methodology

This chapter is concerned with the methodology used for the thesis. It first states the problems for dealing with in Sec. 4.1, then illustrates the structures of the proposed framework in Sec. 4.2. Section 4.3 addresses the steps of training VDGAN in detail, and Sec. 4.4 clarifies the novel implementation of DP protocols as well as the accumulation policy of the privacy budget.

4.1. Problem Statement

We assume a scenario where several local parties called clients holding private databases. The local private datasets are aligned with the same individuals while having different attributes, which satisfies the definition of vertically partitioned data. There is an untrusted server that tries to estimate the joint distribution of all attributes. The server is "honest-but-curious", which means it would follow the rules and protocols strictly but intend to gather private information with high sensitivity. So once a model is dispatched among the server and client sides, an attacker is able to infer the membership of training data by querying the published model on the server. Besides, the real data must be kept on the local client safely and only involved in the local part of the optimization. They are not allowed to be passed to the outside directly. After the training process, the synthetic data, which have similar distributions to real ones, would be published and accessed at will.

The proposed situation are formulated as follows: There are N clients $\{C_k\}_{k=1}^{N}$, each holds a local dataset $X_k(ID, \boldsymbol{d}_k)$ over a same set of individuals ID. $\boldsymbol{d}_k$ indicates the set of local attributes and satisfies $\boldsymbol{d}_i \cap \boldsymbol{d}_j = \varnothing$ for any two local datasets X_i and X_j. In another view, there is a high-dimensional private dataset X with attributes $\cup_{k=1}^{N} \boldsymbol{d}_k$ vertically distributed among N clients. The server aims to generate a synthetic dataset $\tilde{X}$ which has the integrated attributes and similar joint distribution with the real one as

$$\mathbb{P}_{\tilde{X}}(\boldsymbol{d}_1, ..., \boldsymbol{d}_N) \approx \mathbb{P}_X(\boldsymbol{d}_1, ..., \boldsymbol{d}_N) \tag{4.1}$$

4.2. Method Overview

Fig. 4.1 demonstrates the overview of our proposed framework. A distributed GAN model would be constructed among the server and client sides and optimized by the joint training among multiple parts. Here we use the algorithm of WGAN-GP [14] as

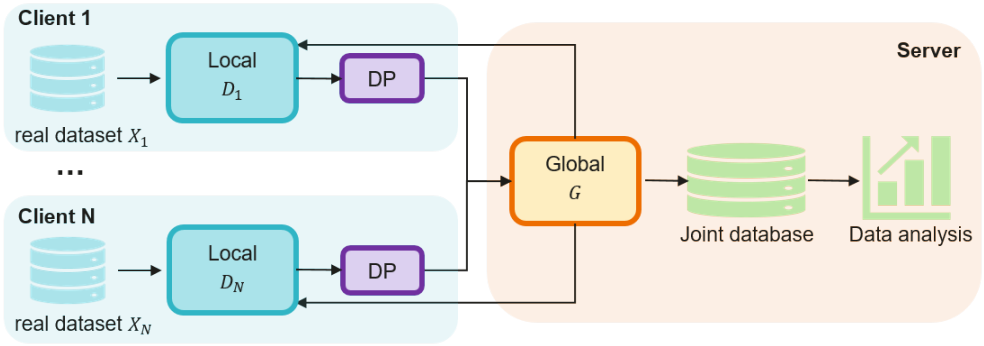


Figure 4.1: Overview of the Whole Framework

the optimization strategy. There is a global generator G on server and N local discriminators $\{D_k\}_{k=1}^{N}$ on each client. All the models consist of Multi-Layers-Perceptrons for the tabular dataset. In Alg. 2 the real data participate in the training process only of the discriminator, so on each client, the corresponding real dataset X_k is always stored on the local side and has never been uploaded. On the server, the training process of generator needs noise sampled from Gaussian distribution and some parameters from clients. The local parameters would be privatized by DP algorithms before uploading. The privacy budgets would be consumed gradually in this process. Here we incorporate two different DP protocols, DPSGD and PATE, into the training scheme. We will illustrate the details and compare their principles in the following sections. Once the global generator is well-trained, we can draw a set of noise samples from a latent space and feed them to the generator. Then we can obtain the synthetic joint data, which can be further used for data analysis and training AI models.

4.3. Vertically Distributed GAN

4.3.1. An Iteration Process

The so-called Vertically Distributed GAN (VDGAN) is proposed as the generative model. If we ignore the DP module temporarily, an iteration consists of the following steps: (1).data synthesis and downloading; (2). optimization of local discriminators; (3). gradients computation and uploading; (4). optimization of the global generator. Step (1) and step (4) are executed on the server side, and the other two are on each client side. For each iteration t, a batch size of real data $X_k^{d_k}$ would be sampled randomly from each private dataset. The parameters θ_g of the global generator would be updated one time while θ_{d_k} of the local discriminators on each client C_k would be updated multiple times. The details are illustrated in Alg. 3.

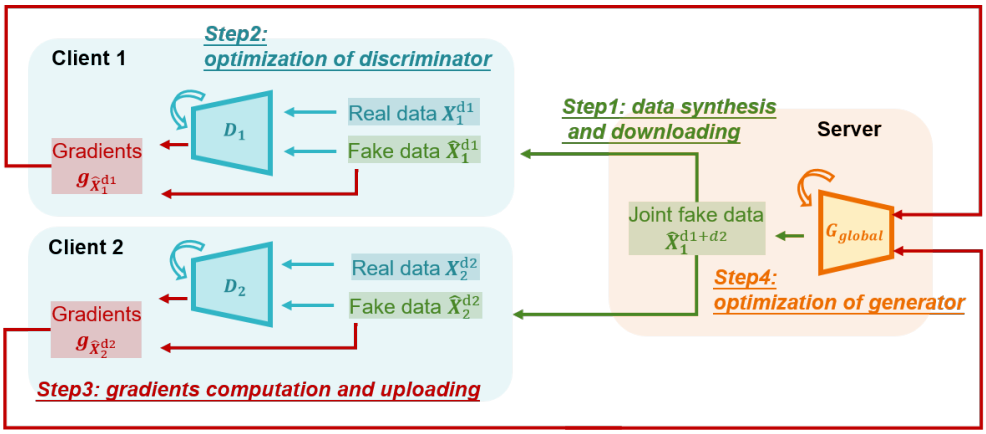


Figure 4.2: Overview of VDGAN

Step 1: Data synthesis and downloading Here we assume that the attributes information from all clients $\{d_1, \cdots, d_N\}$ are public while their corresponding distributions are private. A global generator G on the server side takes noise samples as input and outputs synthetic data. The noise inputs are sampled from a multi-dimensional Gaussian distribution. The generated data $\tilde{X}$ hold attributes $\sum_{k=1}^{N} d_k$, which would be vertically spilt to $\{\tilde{X}_1^{d_1}, \cdots, \tilde{X}_N^{d_N}\}$ as shown in Fig. 4.2 and downloaded to different clients so that on each client the received synthetic data $\tilde{X}_k^{d_k}$ have the identical attributes d_k with the real private data $X_k^{d_k}$.

Step 2: Optimization of local discriminators For the method of VDGAN, each client C_k holds a local discriminator D_k, which takes either real data $X_k^{d_k}$ or synthetic data $\tilde{X}_k^{d_k}$ as input and gives a scalar value for each sample as the discrimination result. To meet the requirements of WGAN-GP [14], the neurons in the output layer of D_k should take no activate function so that the range of the output would not be limited within (0,1). The network size of D_k is proportional to the dimension of attributes d_k. After the forward propagation, the loss $\mathcal{L}_{D_k}$ would be calculated according to Eq. 3.8, and the parameters θ_{d_k} would be updated by gradient descent.

Step 3: Gradients computation and uploading According to Eq. 3.9, the loss of generator derives from the output of discriminator w.r.t. the synthetic input data. So at each client C_k, partial loss of the global generator G can be calculated from the local D_k and $\tilde{X}_k^{d_k}$, which comes from the output of G in step (1). Then the gradients of the partial loss w.r.t. synthetic data $\nabla_{\tilde{X}_k}^{\mathcal{L}_G}$ are calculated and would be uploaded back to the server side. During the execution of all clients the total uploaded gradients have the same size and dimensions as the downloaded synthetic data.

Step 4: Optimization of the global generator On the server side, the Jacobian matrix of each partitioned synthetic data $\tilde{X}_k^{d_k}$ w.r.t. the parameters of the global generator

G is calculated as $\nabla_{\theta_g} \tilde{X}_k$. When the server collects the gradients $\{\nabla_{\tilde{X}_1}^{\mathcal{L}_G}, \cdots, \nabla_{\tilde{X}_N}^{\mathcal{L}_G}\}$ from all clients, according to the chain rule of backpropagation, the final gradients of G w.r.t. its parameters are calculated as:

$$\nabla_{\theta_g} \mathcal{L}_G = \sum_{k=1}^{N} \nabla_{\theta_g} \tilde{X}_k \cdot \nabla_{\tilde{X}_k}^{\mathcal{L}_G} \tag{4.2}$$

This approach avoids collecting the local models $\{D_1, \cdots, D_N\}$ from all clients directly and reconstructing them on the server side. Finally, the parameters of G are updated for one round by the gradient descent.

Algorithm 3 An iteration of training VDGAN

Require: N: number of clients; $\{X_k^{d_k}\}_{k=1}^N$: local real datasets sampled in batch size; s: number of iterating steps of D for each iterating step of G; $\mathcal{N}$: multi-dimensional Gaussian distribution; t: current iterating epoch; γ: learning rate;

Step 1 (server):
Sample noise records $\boldsymbol{z} \sim \mathcal{N}$ in size m
Generate synthetic joint data in size m: $\tilde{X} \leftarrow G(\boldsymbol{z})$
Split $\tilde{X}$ to N datasets $\{\tilde{X}_1^{d_1}, \cdots, \tilde{X}_N^{d_N}\}$

for $k = 1, \cdots, N$ **do**
  Step 2 (client):
  for $i = 1, \cdots, s$ **do**
    Compute loss of discriminator:
    $\mathcal{L}_{D_k} \leftarrow \sum_{j=1}^{m} D_k(\tilde{\boldsymbol{x}}_j) - D_k(\boldsymbol{x}_j) + \textbf{GP}(D_k, \tilde{\boldsymbol{x}}_j, \boldsymbol{x}_j)$ for $\boldsymbol{x}_j, \tilde{\boldsymbol{x}}_j$ in $X_k^{d_k}, \tilde{X}_k^{d_k}$
    Update local discriminator $\theta_{d_k}^{i+1} \leftarrow \theta_{d_k}^{i} - \gamma \nabla_{\theta_{d_k}} \mathcal{L}_{D_k}$
  end for
  Step 3 (client):
  Compute partial loss of generator: $\mathcal{L}_G \leftarrow \sum_{j=1}^{m} -D_k(\tilde{\boldsymbol{x}}_j)$ for $\tilde{\boldsymbol{x}}_j$ in $\tilde{X}_k^{d_k}$
  Compute gradients: $\boldsymbol{g}_{\tilde{X}_k} \leftarrow \{\nabla_{\tilde{\boldsymbol{x}}_1}^{\mathcal{L}_G}, \cdots, \nabla_{\tilde{\boldsymbol{x}}_m}^{\mathcal{L}_G}\}$ for $\tilde{\boldsymbol{x}}_j$ in $\tilde{X}_k^{d_k}$
end for

Step 4 (server):
Aggregate gradients: $\nabla_{\theta_g} \mathcal{L}_G \leftarrow \sum_{k=1}^{N} \nabla_{\theta_g} \tilde{X}_k \cdot \boldsymbol{g}_{\tilde{X}_k}$
Update generator G: $\theta_g^{t+1} \leftarrow \theta_g^{t} - \gamma \nabla_{\theta_g} \mathcal{L}_G$

4.3.2. Warm-Up before Training

Figure 4.3 shows a warm-up process before optimizing VDGAN. It is an optional process only on the client side, aiming to increase the converging speed and improve

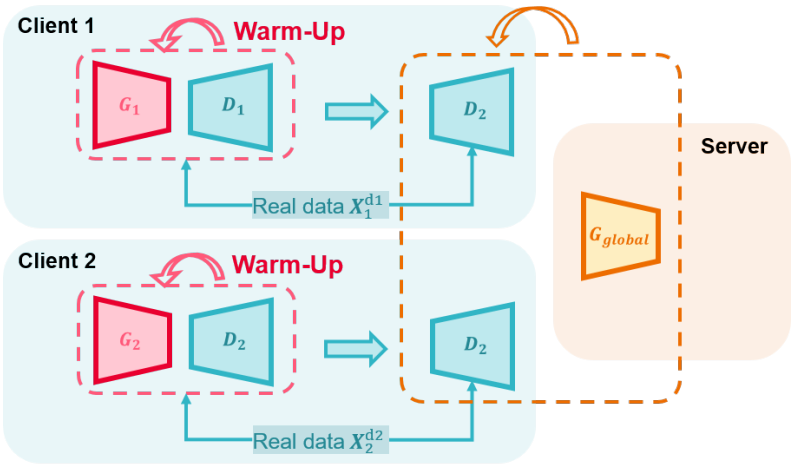


Figure 4.3: Warm-Up for Local Discriminators

the model quality. Because the work [14] proposed that using a stronger discriminator would provide more meaningful gradients information to the generator. So the discriminators are proposed to be initialized in some steps before training VDGAN in distributed mode. On each client, a complete GAN including a generator and a discriminator would be constructed and trained as Alg. 2 in some epochs with the help of its private data. After the warm-up the discriminator model D_k would participate in the co-training with other modules. However, the effects of the local pretraining differ among various datasets, and a robust improvement has not been realized in this thesis.

4.4. DP Protocols

The previous VDGAN framework realizes the data synthesis with high-dimensional joint attributes on the server. However, an adversary still has the ability to recover the personal information of the real data given synthetic ones via some inference attacks. Here two kinds of DP protocols are employed in the training process on the client side to ensure that the gradients have been privatized before leaving. One method is based on the DPSGD [1] approach, and the other one, which is inspired by Wang *et al.* [47], is a variant of PATE [32] framework. Both of them compute the accumulated privacy budgets via RDP [25] over the Gaussian Mechanism.

4.4.1. DPSGD-Based Approach

Fig. 4.4 demonstrates the modification of the VDGAN framework over DPSGD. The Alg. 1 is introduced into the step 2 of Alg. 3 so that the local discriminator is trained

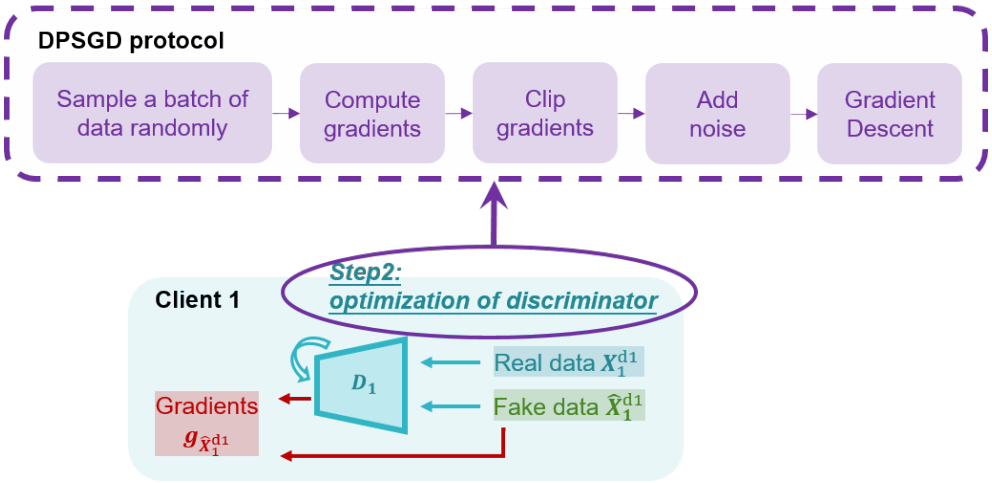


Figure 4.4: Integration of DPSGD in VDGAN

under differential privacy. Because of the post-processing principle of DP, the updated gradients and the global generator would satisfy differential privacy in some budgets. In Alg. 1, the clipping parameter C should be determined carefully to limit the sensitivity s. Recalling that the WGAN-GP algorithm is adopted as the optimization strategy of VDGAN, the gradients' L2-norm in each discriminator have been bounded around 1 due to the gradient penalty. So the optimal parameter $C = 1$ is chosen analytically instead of an intensive grid search. For each pair of input samples $\{\boldsymbol{x}_i, \tilde{\boldsymbol{x}}_i\}$, the gradient

$$\boldsymbol{g}_i = \nabla_{\theta_d} \mathcal{L}_D(\boldsymbol{x}_i, \tilde{\boldsymbol{x}}_i, \theta_d) \tag{4.3}$$

would be clipped to

$$\bar{\boldsymbol{g}}_i = \boldsymbol{g}_i / \max\left(1, \frac{\|\boldsymbol{g}_i\|_2}{C}\right) \tag{4.4}$$

so that its L2-norm is less equal to C. The clipped gradients $\bar{\boldsymbol{g}}_i$ would be added with a bounded Gaussian noise as

$$\bar{\boldsymbol{g}}_i^* = \bar{\boldsymbol{g}}_i + \mathcal{N}(0, \sigma^2 C^2 \boldsymbol{I}) \tag{4.5}$$

All of the clipped gradients with noise in batch size m would be gathered together to update the parameters of the discriminator D:

$$\theta_d^{t+1} = \theta_d^t - \gamma \sum_{i=1}^{m} \bar{\boldsymbol{g}}_i^* \tag{4.6}$$

4.4.2. PATE-Based-Approach

From Fig. 3.1 we see that the PATE framework consists of the private module, including sensitive data and an ensemble of teacher models, as well as the public module, including some public data and a student model. Compared with the VDGAN framework, the server can be regarded as the public module, while the clients are private modules. Here a PATE-based VDGAN framework is proposed to realize differential privacy guarantees. On the public server, the global generator G works as the student model and the synthetic data as the public data of PATE, which would flow to the private modules. While the variant holds some differences on the private clients compared with the original PATE framework.

First is the number of private modules. There are N clients working as private modules in this approach, each holding a sensitive personal dataset. Each real dataset $X_k^{d_k}$ would be split horizontally to E subsets $X_{k,1}^{d_k}, \cdots, X_{k,E}^{d_k}$ without overlapping. As shown in Fig. 4.5, those subsets maintain the same attributes d_k and would be utilized to train E discriminators as the ensemble of teachers. The discriminators $D_{k,1}, \cdots, D_{k,E}$ on one client C_k have identical structures but are trained with different real datasets.

The second difference is the aggregated object via teachers. In the original PATE framework, the predicted results from teacher models would be aggregated into a one-dimensional label, which would be sent to train the student model. However, in the VDGAN framework, the required information from local discriminators to the global generators are high-dimensional gradients. The drastically increasing size of data would consume much more privacy budgets because of a more significant sensitivity. What's more, it leads to a higher communication overhead during the distributed training. Inspired by the work [47], this thesis performs gradient compression during the aggregation process. The details are illustrated in Alg. 4, which consists of two steps and would replace the step 3 in Alg. 3.

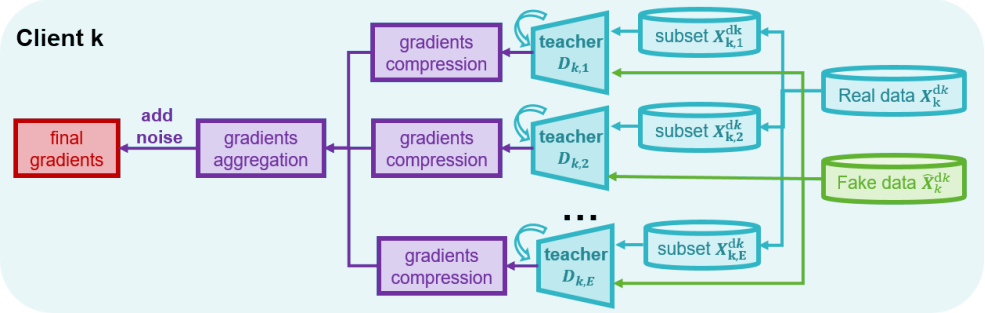


Figure 4.5: Integration of PATE in VDGAN

Step 1: Gradients compression in top-K. There are E teacher discriminators on one client C_k. For any synthetic sample $\tilde{\boldsymbol{x}}_i$, a number of gradients would be calculated

from the teacher ensemble $\{D_{k,j}\}_{j=1}^{E}$ For each gradient

$$\boldsymbol{g}_{i,j} = \nabla_{\tilde{\boldsymbol{x}}_i} \mathcal{L}_G^j \tag{4.7}$$

K entries would be selected as the K-largest absolute values in the gradient vector and the others would be set to 0 as

$$\boldsymbol{h}_{i,j} = \text{arg-topK}(|\boldsymbol{g}_{i,j}|, K) \tag{4.8}$$

$$g_{i,j}^d = 0 \text{ if } d \notin \boldsymbol{h}_{i,j} \tag{4.9}$$

In this way, the dense vector is transformed into a sparsified one so that the consumption speed of the privacy budget would be reduced drastically. Then to bound the sensitivity, each of the K nonzero entries would be clipped within a parameter c so that

$$-c \leq g_{i,j}^d \leq c \tag{4.10}$$

It should be noted that the clipping is performed along each dimension in contrast to clipping the L2-norm of gradients in DPSGD. Then we perform normalization

$$g_{i,j}^d = g_{i,j}^d / \|\boldsymbol{g}_{i,j}\|_\infty \tag{4.11}$$

of the sparsified gradient so that its values are limited within (-1,1). In the last we assign the sign values {1, -1} in the way of

$$\bar{g}_{i,j}^d = \begin{cases} 1 & with\ probability \frac{1+g_{i,j}^d}{2} \\ -1 & with\ probability \frac{1-g_{i,j}^d}{2} \end{cases} \tag{4.12}$$

for each nonzero dimension to replace the real values in the gradient vector. After the sparsification, normalization, and quantization above the complex gradient $\boldsymbol{g}_{i,j}$ would be replaced by a low-rank vector $\bar{\boldsymbol{g}}_{i,j}$ with values only in {-1,0,1}.

Step 2: Gradients aggregation with DP. After the last step, each teacher discriminator $D_{k,j}$ holds a compressed gradient $\bar{\boldsymbol{g}}_{i,j}$ with K nonzero entries of either 1 or -1. In another view, each teacher model can vote for K dimensions, either in the positive direction ($\bar{g}_{i,j}^d = 1$) or the negative direction ($\bar{g}_{i,j}^d = -1$). During the voting process the sum of all gradients should be added with a bounded Gaussian noise which plays the key role of differential privacy as

$$\boldsymbol{g}_i^* = \sum_{j=1}^{E} \bar{\boldsymbol{g}}_{i,j} + \mathcal{N}(0, \sigma^2 \boldsymbol{I}) \tag{4.13}$$

Then we perform quantization on the aggregated noisy gradient $\boldsymbol{g}_i^*$ with sign values {-1,0,1} in the way of

$$\hat{g}_i^d = \begin{cases} 1 & if \quad g_i^{*d} \geq \beta \cdot E \\ -1 & if \quad g_i^{*d} \leq -\beta \cdot E \\ 0 & else \end{cases} \tag{4.14}$$

With the help of a threshold parameter β, an entry $\hat{g}_i^d$ would be assigned as 1 or -1 only if the majority of teachers vote for this dimension in a positive or negative direction respectively. This method improves the robustness of the aggregated result against the outlier votes.

At the end of Alg. 4 for a batch of synthetic input samples, the aggregated gradients in sign values $\{\hat{\boldsymbol{g}}_1, \cdots, \hat{\boldsymbol{g}}_m\}$ would be aligned together as the final gradient matrix $\hat{\boldsymbol{g}}_{\tilde{X}_k}$. Although in this approach, we modify the single local discriminator to a number of teacher discriminators, the final uploading data holds the identical shape to that when training the VDGAN framework without DP.

Algorithm 4 Gradients Compression with DP

Require: $\tilde{X}_k^{d_k}$: a batch of synthetic data; m: batch size; E: number of teachers; $\{D_{k,j}\}_{j=1}^{E}$: an ensemble of discriminators; K: top-K dimensions; c: clipping parameter; β: voting threshold;

for $D_{k,j}, j = 1, \cdots, E$ **do**

compute partial loss of generator: $\mathcal{L}_G^j \leftarrow \mathcal{L}_G(\tilde{X}_k^{d_k}, D_{k,j})$

end for

for $\tilde{\boldsymbol{x}}_i$ in $\tilde{X}_k^{d_k}$ **do**

Step 1: Gradients Compression in Top-K

for $D_{k,j}, j = 1, \cdots, E$ **do**

compute gradient: $\boldsymbol{g}_{i,j} \leftarrow \nabla_{\tilde{\boldsymbol{x}}_i} \mathcal{L}_G^j$

select top-K dimensions: $\boldsymbol{h}_{i,j} \leftarrow$ arg-topK($|\boldsymbol{g}_{i,j}|, K$)

Clip each dimension d in $\boldsymbol{g}_{i,j}$: $g_{i,j}^d \leftarrow \min(\max(g_{i,j}^d, c), c)$

normalize gradient: $\boldsymbol{g}_{i,j} \leftarrow \boldsymbol{g}_{i,j} / \|\boldsymbol{g}_{i,j}\|_\infty$

for each dimension d in $\boldsymbol{g}_{i,j}$ **do**

assign sign values:

if $d \in \boldsymbol{h}_{i,j}$ **then**

$$\bar{g}_{i,j}^d \leftarrow \begin{cases} 1 & with\ probability \frac{1+g_{i,j}^d}{2} \\ -1 & with\ probability \frac{1-g_{i,j}^d}{2} \end{cases}$$

else

$\bar{g}_{i,j}^d \leftarrow 0$

end if

end for

$\bar{\boldsymbol{g}}_{i,j} \leftarrow \{\bar{g}_{i,j}^1 \cdots \bar{g}_{i,j}^d \cdots\}$

end for

Step 2: Gradients aggregation with DP

inject Gaussian noise into aggregated gradient: $\boldsymbol{g}_i^* \leftarrow \sum_{j=1}^{E} \bar{\boldsymbol{g}}_{i,j} + \mathcal{N}(0, \sigma^2 \boldsymbol{I})$

for each dimension d in $\boldsymbol{g}_i^*$ **do**

assign sign values over threshold:

$$\hat{g}_i^d \leftarrow \begin{cases} 1 & if \quad g_i^{*d} \geq \beta \cdot E \\ -1 & if \quad g_i^{*d} \leq -\beta \cdot E \\ 0 & else \end{cases}$$

end for

$\hat{\boldsymbol{g}}_i \leftarrow \{\hat{g}_i^1 \cdots \hat{g}_i^d \cdots\}$

end for

get gradients for all samples: $\hat{\boldsymbol{g}}_{\tilde{X}_k} \leftarrow \{\hat{\boldsymbol{g}}_1, \cdots, \hat{\boldsymbol{g}}_m\}$

4.4.3. Analysis of Privacy Budgets

Both the VDGAN-DPSGD and VDGAN-PATE frameworks utilize the Gaussian Mechanism to provide DP guarantees. The former injects noise to the gradients w.r.t. the model parameters of the local discriminator. However, the latter injects noise to the gradients w.r.t. the synthetic data prepared for the global generator. Compared with the two methods, the former adopts DP when training the local discriminator (step 2 in Alg. 3), while the latter adopts DP when training the global generator (step 3 in Alg. 3). Because of the iterative optimization of the two parts and the post-processing principle of DP, the whole framework satisfies differential privacy.

To compute the consumed privacy budgets via Theorem 4, the L2-sensitivity of the Gaussian Mechanism should be specified for both approaches, where the target samples randomized by the noise hold different features. In VDGAN-DPSGD, the L2-norm of target gradient is clipped with C, so for any two different gradients $\bar{\boldsymbol{g}}_i, \bar{\boldsymbol{g}}_j$, the sensitivity is

$$s_{DPSGD} = \|\bar{\boldsymbol{g}}_i - \bar{\boldsymbol{g}}_j\|_2 \leq 2C \tag{4.15}$$

according to the triangle inequality. In VDGAN-PATE, we should consider a pair of compressed gradients $\bar{\boldsymbol{g}}_i, \bar{\boldsymbol{g}}_j$ from two different teacher models. In the worst case, $\bar{\boldsymbol{g}}_i$ and $\bar{\boldsymbol{g}}_j$ hold the identical indices for the top-K nonzero dimensions, and all of the sign values are opposite. So the sensitivity is

$$s_{PATE} = \|\bar{\boldsymbol{g}}_i - \bar{\boldsymbol{g}}_j\|_2 \leq \sqrt{2^2 \cdot K} = \sqrt{2}K \tag{4.16}$$

After the sensitivity specification, given a loose parameter σ, we can calculate the consumed privacy budget every time we inject a Gaussian noise with scale σ. For the whole training process the consumed ϵ_k at each client C_k is calculated as Alg. 5 according to Theorems 4, 5, 3. It should be noted that We a series of RDP-budgets over a set of α in range $(0, \alpha_{max})$ would be computed and the minimum would be selected as the optimal DP budget ϵ_k for client C_k. The clients $\{C_1, \cdots, C_N\}$ hold parallel private data $\{X_1, \cdots, X_N\}$ without overlapping, so according to Theorem 2, the final privacy budgets ϵ for the distributed framework is the maximal consumed budgets among all clients as

$$\epsilon = \max\{\epsilon_1, \cdots, \epsilon_N\} \tag{4.17}$$

Algorithm 5 Accumulation of Privacy Budget

ComputeDP$(\sigma, s, T, m, \alpha_{max}, \delta)$:
for $\alpha = 1, \cdots, \alpha_{max}$ **do**
$\epsilon^{i}_{\mathsf{RDP},\alpha} = \frac{s^2\alpha}{2\sigma^2}$
$\epsilon_{\mathsf{RDP},\alpha} = \epsilon^{i}_{\mathsf{RDP}} \cdot T \cdot m$
$\epsilon_{\mathsf{DP},\alpha} = \epsilon_{\mathsf{RDP},\alpha} + \frac{\log 1/\delta}{\alpha - 1}$
end for
$\epsilon_{\mathsf{DP}} = max\{\epsilon_{\mathsf{DP},\alpha}\}^{\alpha_{min}}_{\alpha=1}$
Return ϵ_{DP}

5. Experiments

This chapter demonstrates the results of practical experiments and discusses the related findings. It begins by introducing the adopted dataset in Sec. 5.1 and metrics in Sec. 5.2, then addresses the setup of the proposed frameworks and some baselines in Sec. 5.3, finally presents the results in plots and analyze the characteristics and relative merits of different approaches in Sec. 5.4 and Sec. 5.5.

5.1. Datasets

The details of the four chosen public tabular datasets are present as follows.

- **Adult** The Adult dataset [4] contains 48842 records extracted from the 1994 U.S. Census database. Each record consists of 15 categorical or integer attributes describing the personal information, e.g., age, education, and native country. It aims to predict whether a person makes over 50K a year. 14 of them are chosen to discretize the numerical attributes into categorical ones.
- **Vehicle** The Vehicle dataset [6] has 98528 records extracted from a real-life vehicle tracking sensor network. It consists of 101 binary attributes from acoustic and seismic sensors, which are used to classify the vehicle type into two classes.
- **Census** The Census dataset [5] has 2458285 records collected from the 1990 U.S. Census and consists of 68 categorical attributes. It contains personal information like gender, marriage status, and income which are used for classifying the duration of people's active duty service into three classes. 60 attributes are chosen for the setup.
- **Twitter** The Twitter dataset [20] contains 583250 records of buzz events from the social network Twitter. It consists of 78 numerical attributes such as number of created discussions, author increase, and average discussion length. The purpose of this dataset is to predict the number of active discussions. All of the numerical attributes are discretized into categorical ones.

After the pre-processing, all attributes of each dataset are categorical, in which one attribute can be regarded as the class label and applied to classification tasks. To be applied to neural networks, the attributes are encoded in one-hot mode, which describes their domain size. The corresponding original and one-hot dimensions are

Dataset	Sampling Type	Discretization	Dimensions	One-Hot Dimensions
Adult	up-sampling	yes	14	135
Vehicle	up-sampling	no	101	202
Census	down-sampling	no	60	282
Twitter	down-sampling	yes	78	369

Table 5.1.: Pre-processing of Datasets

listed in Tab. 5.1. The Adult dataset is the simplest one with the lowest dimension, then the Vehicle dataset. The other two are more complicated. Although they have different numbers of original records, all the datasets are resampled randomly to the same size. Each consists of 10^3 testing data and 10^5 training data.

5.2. Metrics

To evaluate the quality of a data-generative model efficiently, this thesis investigates the performance of its generated synthetic data via black-box tests. It comprehensively compares the utility of synthetic data in contrast with the real one from diverse aspects. Additionally, the privacy preservability of published synthetic data should also be considered.

5.2.1. Utility

The server aims at generating high-dimensional synthetic data, which preserves the joint-distributional characteristics as the real ones and replaces the role of real data in data mining work. The utility of synthetic data refers to its capability to reproduce properly the behavior observed in the real one. That indicates not only the similarity of distributions between the synthetic and real data but also its performance in data mining tasks in place of the real one.

Statistical Distribution

The Average Variance Distance (AVD) is utilized to quantify the distance between two high-dimensional statistical distributions. For a pair of real and synthetic datasets, the k-AVD is defined as

$$k\text{-AVD} \;=\; \frac{1}{m}\sum_{i=1}^{m}\left[\frac{1}{2}\sum_{\omega_i\in\Omega}\left|\mathbb{P}_r(\omega_i)-\mathbb{P}_s(\omega_i)\right|\right] \tag{5.1}$$

where $\{\omega_i\}$ denotes a set of attribute combinations in k dimensions and m the number of the combinations. $\mathbb{P}_r(\omega_i)$ and $\mathbb{P}_s(\omega_i)$ refer to the k-dimensional marginal distribu-

tions of real and synthetic data for a given attribute combination ω_i. The accumulation inside is along all possible values of one combination to get the corresponding variance distance, while the accumulation outside is along the different combinations to get an average result. The k-AVD value is bounded by [0,1], and a lower value represents that the synthetic data can better simulate the joint distributions of the real one.

Cross-Attributes Correlation

The Pearson Correlation Matrix can measure the linear correlations of all pairs of random variables for each distribution. It can be visualized as a heat map. This thesis evaluates how well the synthetic data can preserve the internal correlations of the given real distribution by comparing the two maps. For a quantitative method, the Distance of Correlation Matrix (DCM) [16] is defined as

$$\text{DCM} = 1 - \frac{tr\{\mathcal{R}_r \mathcal{R}_s\}}{\|\mathcal{R}_r\|_2 \, \|\mathcal{R}_s\|_2} \tag{5.2}$$

where $\mathcal{R}_r$ and $\mathcal{R}_s$ refer to the Pearson Correlation Matrix of real and synthetic data, $tr(\cdot)$ the matrix trace and $||\cdot||_2$ the L2 norm. If the Correlation Matrices are flattened to two vectors, the value of DCM is equivalent to the cosine angle between them. So it ranges from 0 to 1, and a lower value means a higher similarity of the correlations between two distributions.

AI Training Performance

This thesis employs Machine Learning models to evaluate the utility of synthetic data in data analysis work. Here we assume that $\mathcal{D}_{r-train}$ is the real training data that participate in the training process of the generative model, $\mathcal{D}_{r-test}$ the real test data, and $\mathcal{D}_s$ the synthetic data generated from the well-trained model. The Train Real Test Real (TRTR) test is adopted firstly by training a classifier on $\mathcal{D}_{r-train}$ and testing on $\mathcal{D}_{r-test}$, then the Train Synthetic Test Real (TSTR) test by training a new classifier on $\mathcal{D}_s$ and testing on $\mathcal{D}_{r-test}$. Intuitively the TRTR score should be better than the TSTR score. Moreover, if the TSTR score is very close to the TRTR one, the synthetic data would be considered valid as the real one in AI training tasks.
Here two kinds of models as classifiers are exploited:

- The Multi-Layers Perceptrons (MLP), which has one hidden layer with 100 neurons using activation function ReLU and optimizer Adam.
- The AdaBoost Ensemble, which is built based on the Decision Tree Classifier with the maximum boosting steps of 50.

Two kinds of scores are utilized for the classification:

- The mean accuracy of the given test data.

- The Area Under the Receiver Operating Curve (AUC). If the predicted label is not binary, it is calculated by averaging the AUC scores of all possible pairwise combinations of the classes.

Overall, there are four approaches (MLP-Accuracy, MLP-AUC, AdaBoost-Accuracy, ADB-AUC) for measuring the classification precision.

5.2.2. Privacy Preservability

The privacy preservability of given data is assessed by the Membership Inference Attack (MIA) [38], which is launched to identify the participation of a particular target record or a set of records in training a generative model. It considers an honest-but-curious adversary $\mathcal{A}$. The adversary only has access to a black-box generative model and tries to determine whether the target records belong to the training set $\mathcal{D}_{r-train}$ of the given model or not. Inspired by Hilprecht *et al.* [17], the attack is employed as follows.

1. m real records $\{\boldsymbol{x}_i\}_{i=1}^{m}$ from training dataset $\mathcal{D}_{r-train}$ and m real records $\{\boldsymbol{x}_i\}_{i=m+1}^{2m}$ from testing dataset $\mathcal{D}_{r-test}$ are sampled as target records;
2. The adversary $\mathcal{A}$ samples a sufficiently large amount of synthetic records $\{\tilde{\boldsymbol{x}}_j\}_{j=1}^{n}$ from the given generative model G
3. A value function $d(\boldsymbol{x}, \tilde{\boldsymbol{x}})$ is adopted to measure the pairwise distance between real and synthetic records;
4. For each target record $\boldsymbol{x}_i$, the average probability that the distance between itself and each synthetic record $\tilde{\boldsymbol{x}}_j$ is smaller than threshold t is approximated via Monte Carlo integration [31] as

$$f_t(\boldsymbol{x}) \;=\; \frac{1}{n}\sum_{j=1}^{n} \mathbf{1}_{\tilde{\boldsymbol{x}}_j \in U_t(\boldsymbol{x})} \tag{5.3}$$

   where $U_t(\boldsymbol{x}) = \{\tilde{\boldsymbol{x}} \mid d(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \leq t\}$ denotes the t-neighborhood of $\boldsymbol{x}$ and the threshold t is heuristically chosen as $\underset{1\leq i\leq 2m}{median}\left(\underset{1\leq j\leq n}{min}\, d(\boldsymbol{x}_i, \tilde{\boldsymbol{x}}_j)\right)$
5. The adversary $\mathcal{A}$ labels m target records with highest values $f_t(\boldsymbol{x}_i)$ as predicted training set $\{\boldsymbol{x}_i^A\}_{i=1}^{m}$. And the MIA-Accuracy is the proportion of actual training data in this set:

$$\text{MIA-ACC} \;=\; \frac{1}{m}\cdot\left|\left\{i \mid \boldsymbol{x}_i^A \in \{\boldsymbol{x}_i\}_{i=1}^{m}\right\}\right| \tag{5.4}$$

To measure the degree of difference among categorical data significantly, the Hamming distance [46] is adopted as the value function $d(\boldsymbol{x}, \tilde{\boldsymbol{x}})$, which is the number of

positions at which the corresponding values are different for a pair records. For a target record $\boldsymbol{x}_i$, the estimation $f_t(\boldsymbol{x}_i)$ describes the density of synthetic data located at its neighborhood, so this function attains higher values for training data records. Therefore, a higher MIA-Accuracy score means that the adversary would easily identify the training data, and the sensitive information might be disclosed easily by the given generative model.

5.3. Setup

The experiments are conducted over five approaches.

- **Center-NoDP**: the GAN model in centralized settings without DP
- **VD-NoDP**: the GAN model in vertically distributed settings without DP
- **VDGAN-DPSGD**: the vertically distributed GAN with DPSGD protocol
- **VDGAN-PATE**: the vertically distributed GAN with PATE protocol
- **DPLT**: the vertically distributed LTM with DP from the work [41]

There are four GAN-based approaches implemented during evaluation. The first two are deployed without privacy strategies as benchmarks. The other two are the critical approaches of this thesis. The state-of-the-art DPLT framework is implemented as the baseline, which would publish joint synthetic data via a vertically distributed LTM model with vanilla DP mechanisms. The experiments in this section are conducted over a two-client setting for the four approaches in distributed settings. Here the training dataset is partitioned equally into two parts as local datasets. The neuron size of the GAN model for different datasets would be adjusted proportionally to their dimension of attributes. For a better comparison, a fixed random-seed is set when initializing models. And the evaluation results derive from the average values of repeated tests in a fixed set of seeds. The optimal hyper-parameters are set via grid search.

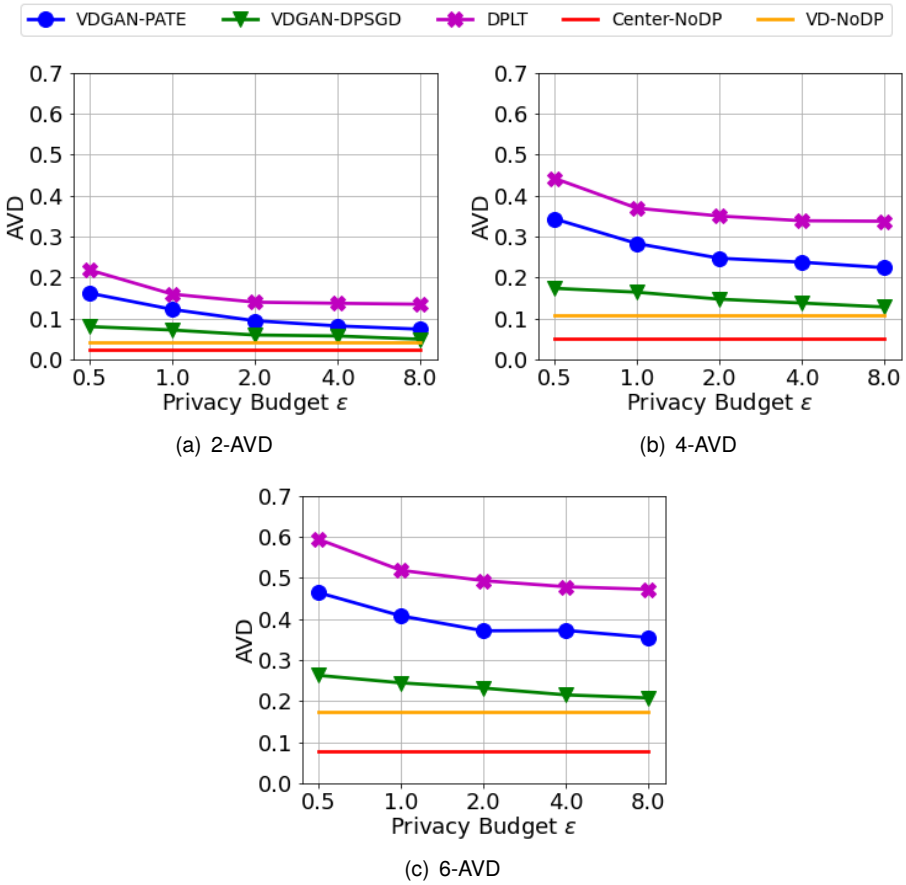
5.4. Evaluation and Discussion

This section presents the experimental results using the mentioned metrics and discusses the findings. After the optimization process, synthetic data are sampled in the same size as the training set from corresponding generative models. Only data instead of models are involved in the evaluation work.

5.4.1. Results of Utility

Statistical Distribution

The k-AVD tests with $k = \{2, 3, 4, 5, 6\}$ w.r.t different privacy budgets $\epsilon = \{0.5, 1.0, 2.0, 4.0, 8.0\}$ are implemented on each approach over all four datasets. The parameter m in Eq. 5.1 is set to 100 so that the k-AVD value denotes the average result of 100 different combinations in k-dimensions.



(a) 2-AVD

(b) 4-AVD

(c) 6-AVD

Figure 5.1: k-AVD of Census Dataset over Various Privacy Budgets

Figure 5.1 shows the results of Census dataset for $k = \{2, 4, 6\}$. The values along the x-axis represent privacy budgets ϵ, and the y-axis the k-AVD values. In each subplot, each line refers to one approach, as shown in the legends at the bottom. Additionally, the horizontal lines without markers denote the approaches without DP protocol. Obviously, the AVD value decreases with a larger ϵ. That verifies that the synthetic data have better similarity in statistical distribution with the real one given more privacy

budgets. What's more, the results from approaches without DP are always better than those with DP. That is because the DP protocols affect the optimization process negatively. Furthermore, the red lines denoting Center-NoDP are better than the yellow line of VD-NoDP, which reflects that the vertical partition harms the model training. Compared with the three subplots together, the k-AVD value increases with a larger k, which means that the simulation of joint distribution would become harder in higher dimensions.

Figure 5.2 compares the results of different datasets on the same 4-AVD. For the three approaches with DP, the baseline DPLT performs better in Adult dataset, while ours perform better in the other three datasets, especially under a larger k. That indicates that the proposed frameworks are more suitable for a high-dimensional dataset. For Twitter and Vehicle datasets, the results of VDGAN-DPSGD/PATE in large ϵ are almost same with or even better than VD-NoDP.

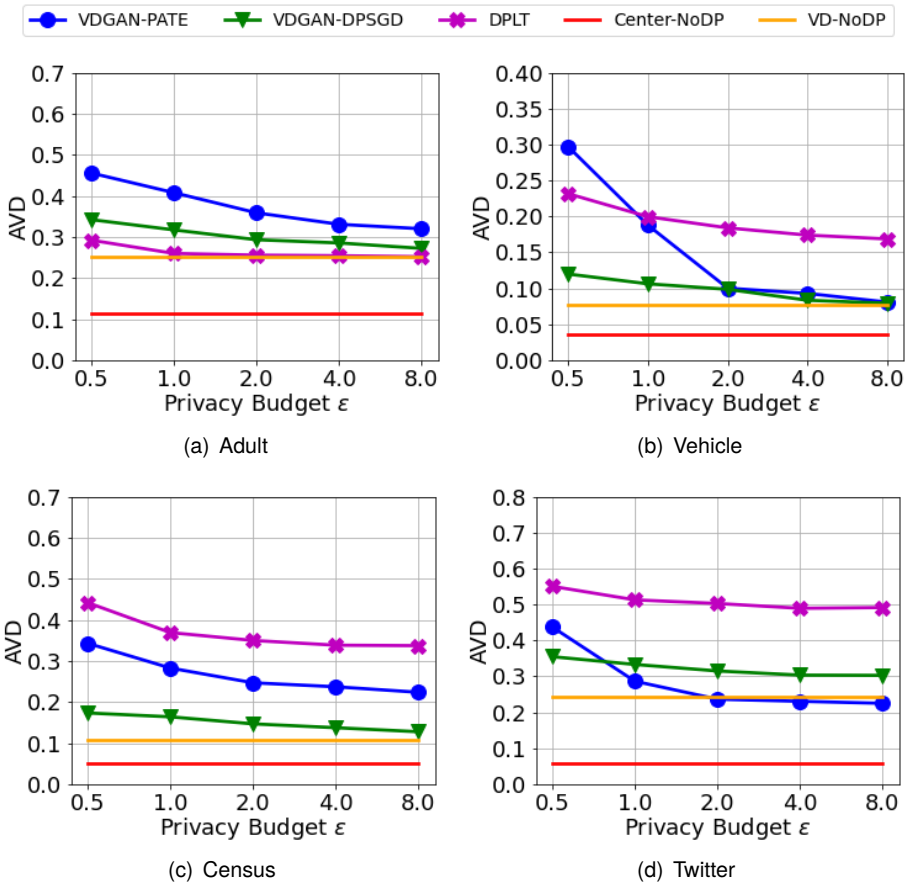
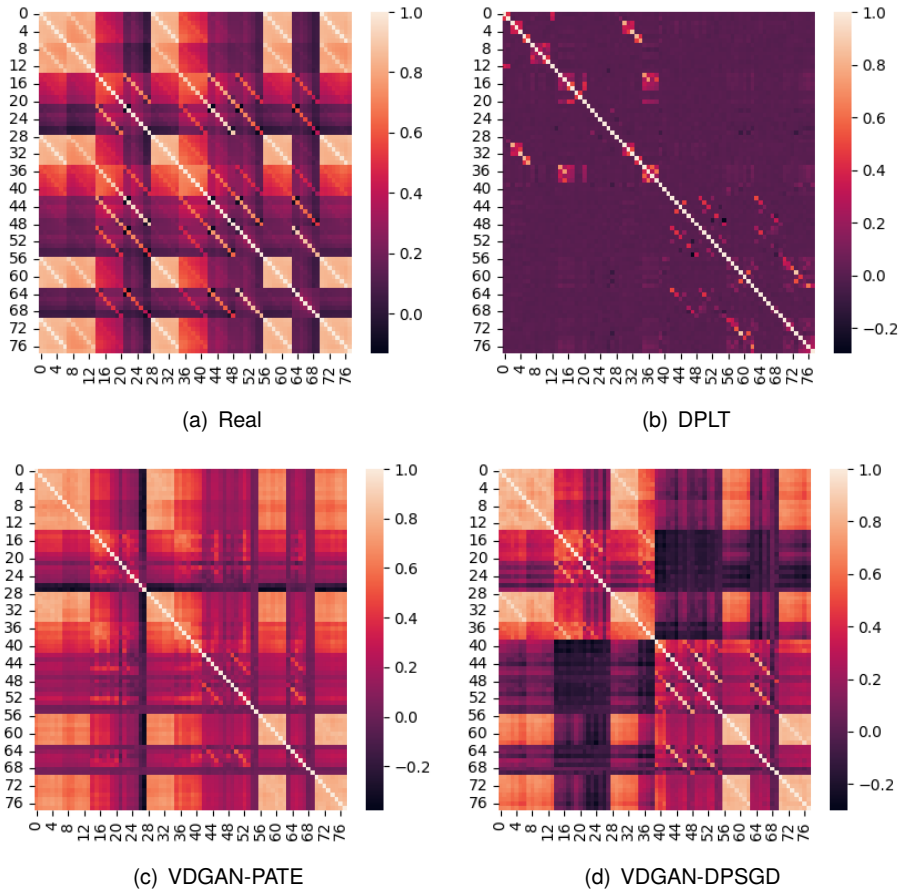


Figure 5.2: 4-AVD of Different Datasets

Cross-Attributes Correlation

Figure 5.3 exhibits the maps of Twitter dataset over real data as well as synthetic ones w.r.t. the same privacy budget $\epsilon = 1.0$. Each map visualizes the Pearson Correlation Matrix of a specific dataset. The subfigure 5.3(a) denotes the results from the real dataset, while the other three ones show the results of synthetic datasets sampled from three different approaches with DP. The maps of the two approaches without DP look almost the same as the real ones, so they are not shown here. It is obvious that the maps of two VDGAN frameworks are much more similar to the real one than DPLT, which confirms that the synthetic data from the proposed approaches can more successfully preserve the real data's correlation features among different attributes than the baseline.



(a) Real (b) DPLT

(c) VDGAN-PATE (d) VDGAN-DPSGD

Figure 5.3: Heatmaps of Pearson Correlation Matrix for Twitter Dataset

Figure 5.4 presents the DCM w.r.t different privacy budgets ϵ over all datasets and all frameworks. Each subfigure denotes the results of one dataset. The x-axis repre-

sents different privacy budgets, and the y-axis the DCM values. Each line represents the results from one approach. In each subfigure, the DCM value declines with the increase of privacy budget ϵ. Besides, the red lines of Center-NoDP indicate the best results with almost 0, then the results from VD-NoDP in yellow lines. However, For Twitter Dataset in Fig. 5.4(d), the results of VDGAN-PATE are better than VD-NoDP. There is a significant improvement over all datasets for the results of VDGAN-DPSGD compared with DPLT. For Adult dataset, the one in lower dimensions, and the dataset Vehicle, the one with binary attributes, the VDGAN-PATE approach performs worse under some privacy budgets. Contradictorily, for more complicated datasets with higher dimensions and larger domain sizes like Census and Twitter, if comparing the y-axis's ranges from different subplots, DPLT suffers from a decreasing performance while VDGAN-based approaches do not have this problem. It is concluded again that the proposed VDGAN framework is more suitable for high-dimensional datasets.

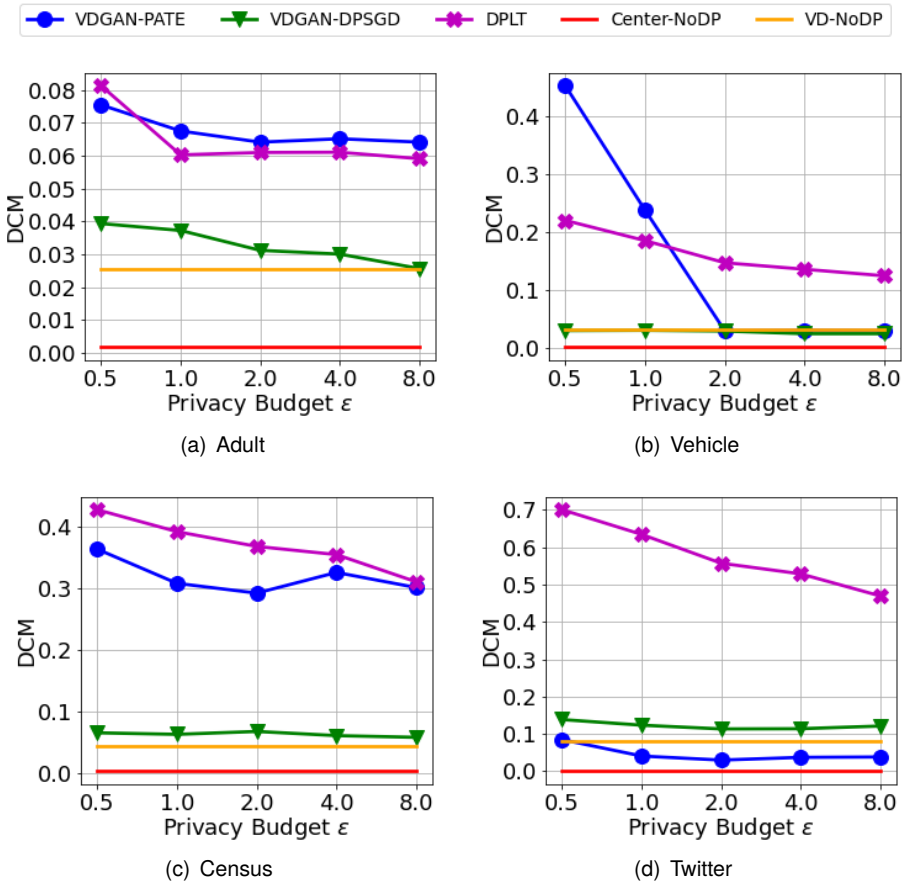
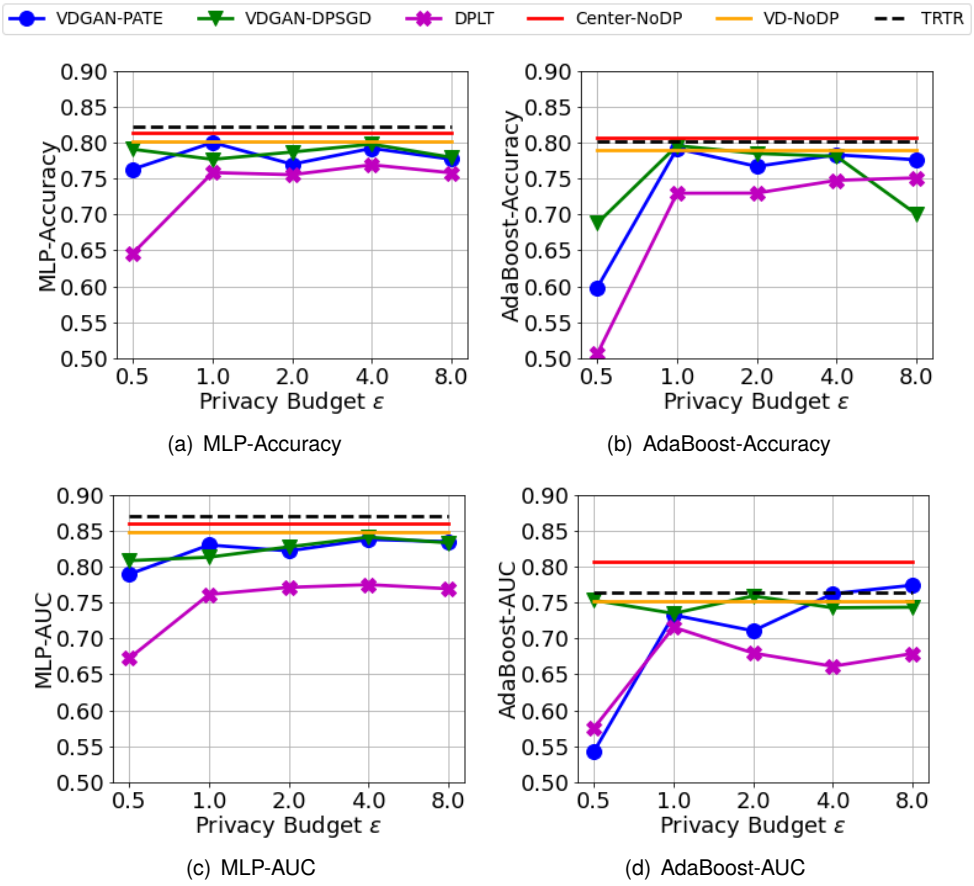


Figure 5.4: DCM under Different Privacy Budgets

AI Training Performance

Figure 5.5 depicts the classification scores w.r.t. different privacy budgets ϵ for Twitter dataset. Each subfigure illustrates the results of specific classification metrics. The x-axis represents different privacy budgets, and the y-axis is the value of corresponding metrics. The black-dash line denotes the results of TRTR tests as a benchmark, while others TSTR tests, where the real test data are the same but the synthetic data are sampled from different approaches. For each subplot, the classification score on the y-axis does not increase strictly with ascending privacy budgets, which is unanticipated from the theoretic assumption. Even though each test has been repeated five times to get an average result, the randomness of Machine Learning models is still non-negligible compared with traditional metrics like k-AVD and DCM. However, In most cases, the classification scores under large privacy budgets are better than those under small ones. Besides, the TSTR results of Center-NoDP are close to those of TRTR and sometimes even better, as shown in Fig. 5.5(d). That confirms the strong ability of data synthesis for the GAN model without DP so that the synthetic data is almost equivalent to the real one.



(a) MLP-Accuracy (b) AdaBoost-Accuracy

(c) MLP-AUC (d) AdaBoost-AUC

Figure 5.5: Classification Scores for Twitter Dataset

Figure 5.6 shows the results of four different datasets according to the AUC score of the MLP classifier. In each subfigure, the TSTR results of VDGAN-DPSGD are generally the best among the three approaches with DP. What's more, for Vehicle dataset, they are even better than those of VD-NoDP. It is possible to hypothesize that the DPSGD protocol can improve the robustness during training a GAN model in vertically distributed settings. However, the results of VDGAN-PATE are complicated. For the most straightforward Adult dataset, the scores in all privacy budgets are worse than the baseline DPLT. In contrast, the Twitter dataset with the highest dimensions performs better. As for the other two datasets, it wins against DPLT when holding a larger privacy budget.

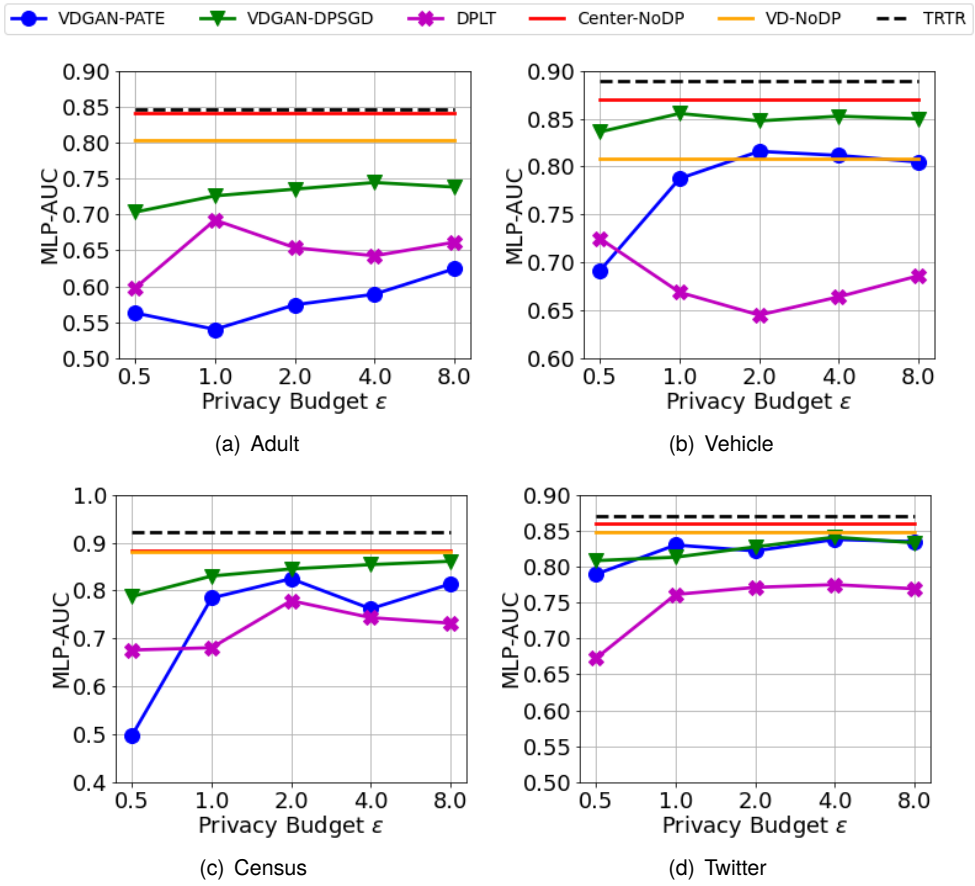
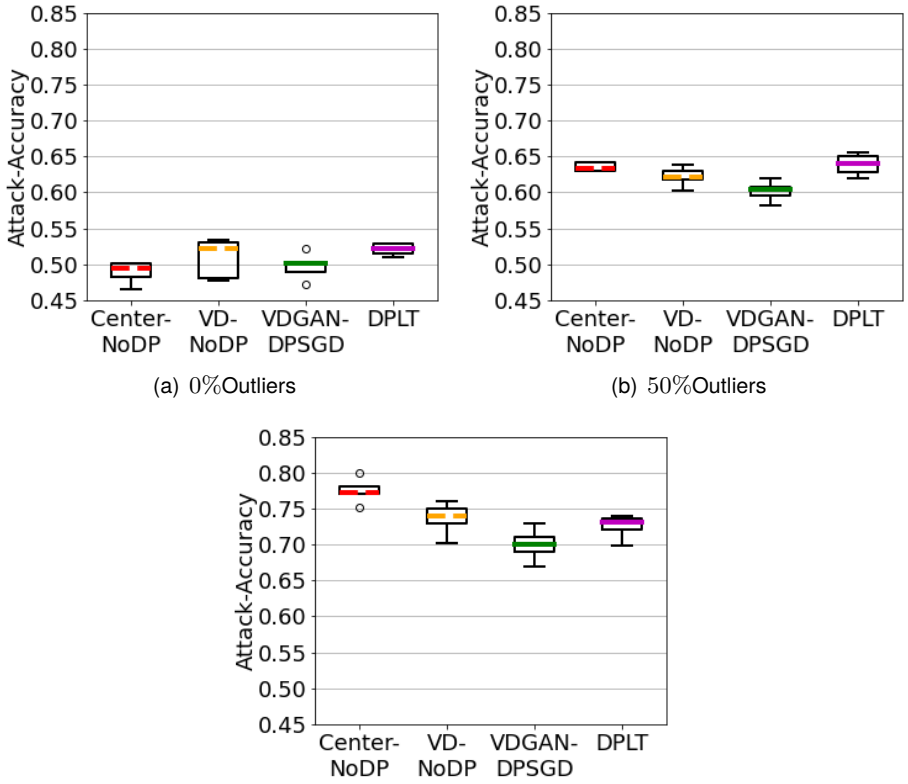


Figure 5.6: AUC Scores of MLP Classifier for Different Datasets

5.4.2. Results of Privacy Preservability

General Comparison

The MIA tests are implemented following the methods in Sec. 5.2.2 with $m = 100$ and $n = 10^5$. To comprehensively consider the protection of different kinds of data, the tests are employed in three modes, where the percentage of outliers in m target training data are 0%, 50% and 100%. Because of the severe randomness, the tests are conducted multiple times.



(a) 0%Outliers (b) 50%Outliers

(c) 100%Outliers

Figure 5.7: MIA-Accuracy with Different Target Data for Adult Dataset

Figure 5.7 demonstrates the boxplots of different approaches. All approaches with DP are implemented with the same privacy budget $\epsilon = 1.0$. Each subfigure represents the MIA-Accuracy in a given mode for Adult dataset. A higher value of MIA-Accuracy denotes a weaker privacy intensity. The four boxplots in vertical arrangement indicate the results from different approaches with or without DP. There is no distinct tendency for Fig. 5.7(a), which depict the results in 0% outliers mode. Whether the DP protocol is adopted, the accuracy scores are mostly around 0.5. That means the adversary cannot identify the records that participated in model training from a set of records with a similar distribution. The value of MIA-Accuracy rises with an ascending proportion of outliers, as shown in Fig. 5.7(b) and Fig. 5.7(c). According to this, it can be inferred that there is severer information leakage for outliers with more distinctive distributions.

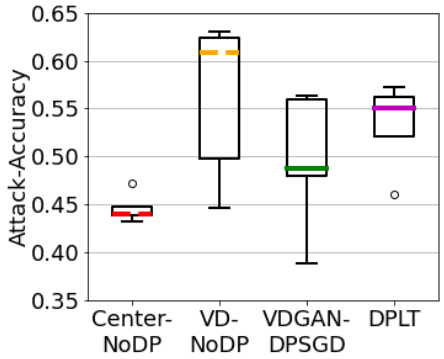


Figure 5.8: MIA-Accuracy with Different Target Data for Census Dataset

Comparing the results of the two approaches without DP shows that the model in vertically distributed settings has better privacy preservability than the centralized one. An exception is the Census Dataset, as shown in Fig. 5.8, where the accuracy of Center-NoDP is much lower than that of all the other approaches. A possible explanation is that the mode collapse phenomenon of the GAN model happens in the centralized setting so that it does not learn the distributions of outliers.

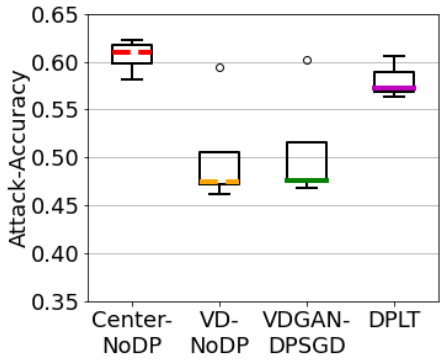


Figure 5.9: MIA-Accuracy with Different Target Data for Twitter Dataset

The MIA-Accuracy of DPLT is sometimes even higher than those of VD-NoDP, for example, the results in Fig. 5.9. That might indicate that the latent tree model of DPLT is more vulnerable under MIA than the GAN-based model.

In general, the proposed VDGAN-DPSGD framework has lower accuracy than other approaches. However, the results still need to be more stable. Sometimes it has apparent advantages, while sometimes just a slight improvement.

Trade-Off between Utility and Privacy

As is known, training a model with DP protocol is to provide privacy protection by sacrificing the utility of its output. A more effective method to investigate the privacy preservability of a given dataset is comparing the trade-off between utility and privacy. Figure 5.10 presents the accuracy of privacy attacks against the utility over all datasets. The MIA-Accuracy in 50%-outliers mode is used to describe the privacy intensity, and the 4-AVD score is adopted to represent data utility in multiple perspectives. For each subfigure, the x-axis denotes values of a specific utility metric, and the y-axis the values of MIA-Accuracy. The dots with the same color derive from one given approach under different privacy budgets. The red star points to the ideal direction where the data have a low attack accuracy while keeping good utility.

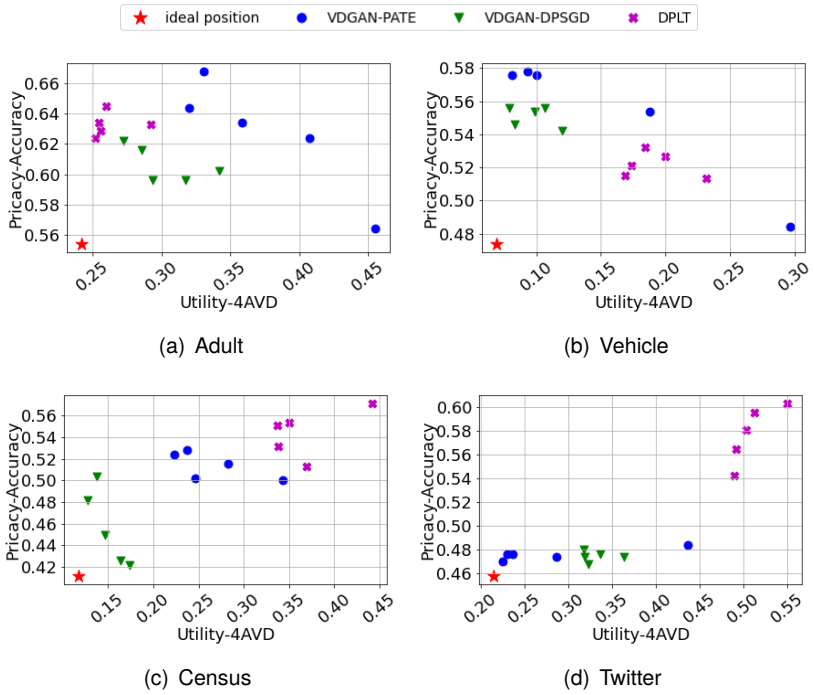


Figure 5.10: MIA-Accuracy against AVD Score for Different Datasets

For the most complex Twitter dataset shown in Fig. 5.10(d), the dots of DPLT and VDGAN approaches are distinctly located far and near the star, which means that the results of VDGAN approaches are better than the baseline in both privacy strength and utility. For the relatively complicated Census dataset, the dots of two VDGAN approaches are still located at the front part of the arrow compared with DPLT. In Fig. 5.10(c), the dots of VDGAN-PATE and DPLT are horizontal while the former location at the left side, which means the results of VDGAN-PATE have better utility

than the baseline under the same privacy strength. The dot distributions for the simpler datasets Adult and Vehicle are approximately perpendicular to the ideal direction. That is consistent with the trade-off between privacy and utility.

5.5. Ablations

In this section, more experiments are conducted by adjusting the settings that greatly influence the results.

5.5.1. Extension to Multiple Clients

Apart from Center-NoDP, the other four approaches in vertical settings are feasibly extended to the multiple-client setting. Figure 5.11 exhibits the utility results using the metric 4-AVD over multiple clients in {2, 3, 4}. The three approaches with DP are implemented with the same privacy budget $\epsilon = 8.0$. Each subfigure shows the results from a specific metric of a given dataset. The x-axis represents the various number of clients, and the y-axis represents the utility metrics value. The bars in different colors denote the results of various methods, and the horizontal line represents the values from Center-NoDP as a baseline. For all datasets, the 4-AVD values increase with more clients, indicating the increasing difficulty of publishing synthetic data over more parties. The results of GAN-based frameworks are much more sensitive to the increase in client numbers than those of DPLT, which seems to be a disadvantage of VDGAN. However, despite the varying results for complicated datasets like Census and Twitter, the proposed VDGAN approaches still have better behaviors than DPLT in most cases. Besides, there are some expectations for Vehicle dataset in Fig. 5.11(b). There is no distinct discrepancy between the lines of VD-NoDP and the other three approaches with DP. That might account for a relatively large privacy budget $\epsilon = 8.0$, which does not harm the utility performance too much.

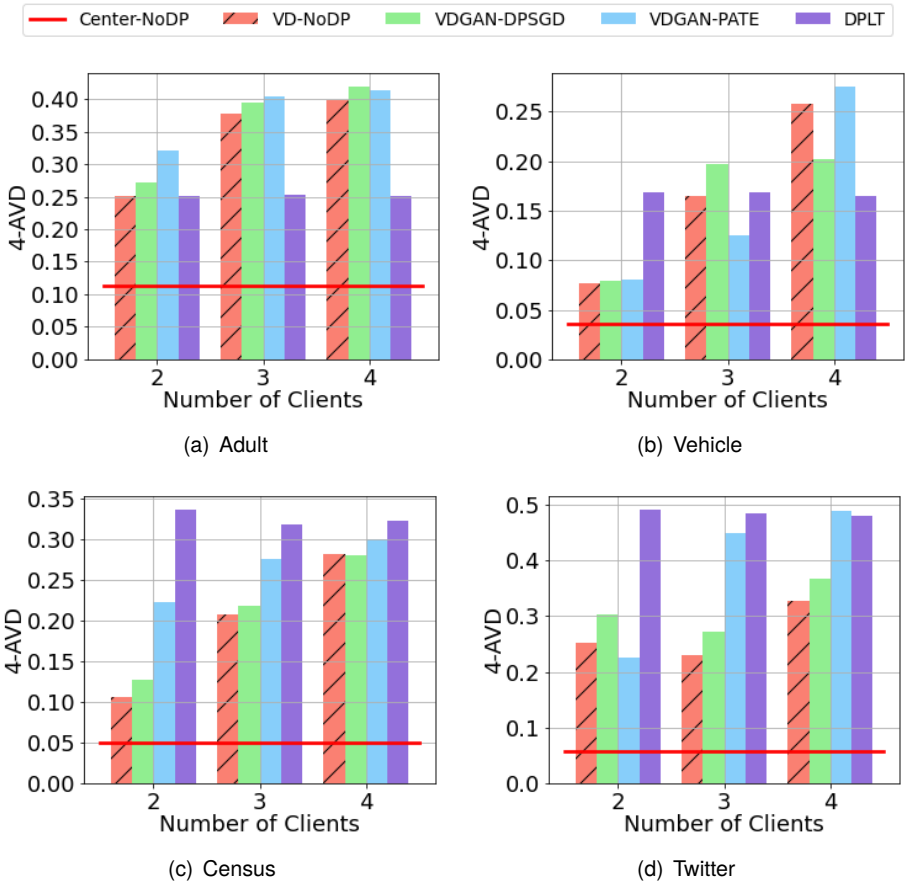
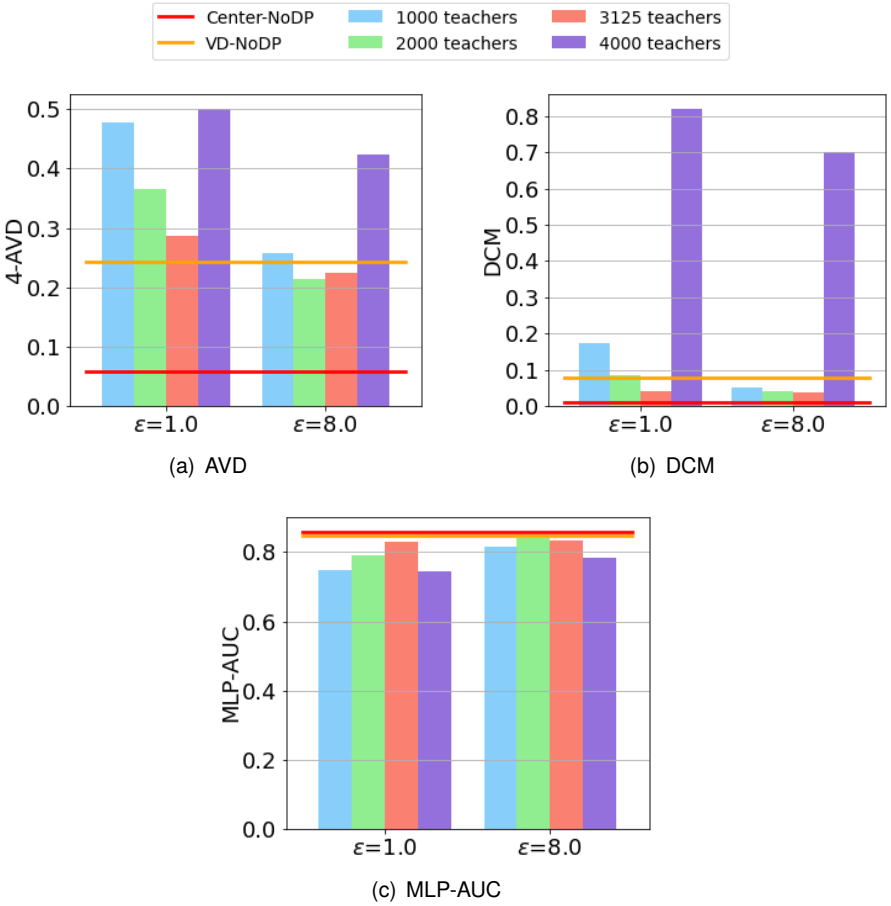


Figure 5.11: 4-AVD with Different Number of Clients

5.5.2. Extension to Different Number of Teachers for VDGAN-PATE

The number of teachers is a critical hyper-parameter in the VDGAN-PATE approach. Figure 5.12 shows the utility results of VDGAN-PATE w.r.t various numbers of teachers for Twitter dataset. Each subplot demonstrates the results of one metric (4-AVD, DCM, MLP-AUC) under different privacy budgets. The y-axis refers to the metric values. Two groups of bars denote the results under a large privacy budget $\epsilon = 8.0$ and a small one $\epsilon = 1.0$ respectively. In each group, the four bars with different colors show the results with different numbers of teachers. The straight lines denote results from the two approaches without DP as benchmarks. In one subplot, the results get worse if the number of teachers is too small (1000) or too large (4000). If the number is too small, the noise with a large amplitude has to be added to fulfill the same privacy budget, which is harmful to the optimization process. If the number is too large, each teacher has too few records as its local dataset, so its discriminator cannot converge very well.

If we compare the two groups of different privacy budgets in one subplot, the ideal number for large ϵ is less than that for small ϵ. For example, in Fig. 5.12(a), the ideal number for $\epsilon = 1.0$ is 3125 while the ideal one for $\epsilon = 8.0$ is 2000. This is because when the target privacy budget gets smaller, more noise should be added during the optimization process, so more teachers are required to resist the influence of noise.



(a) AVD (b) DCM

(c) MLP-AUC

Figure 5.12: Utility Results of VDGAN-PATE with Different Number of Teachers for Twitter Dataset

The optimal number of teachers for different datasets can be found by comparing the results in Fig 5.13. This thesis adopts 3125 as the ideal number of teachers for complicated datasets Census and Twitter, which is larger than the ideal number of 2000 for simpler datasets Adult and Vehicle. This discrepancy might be because more teachers are required to improve the training process for complicated datasets.

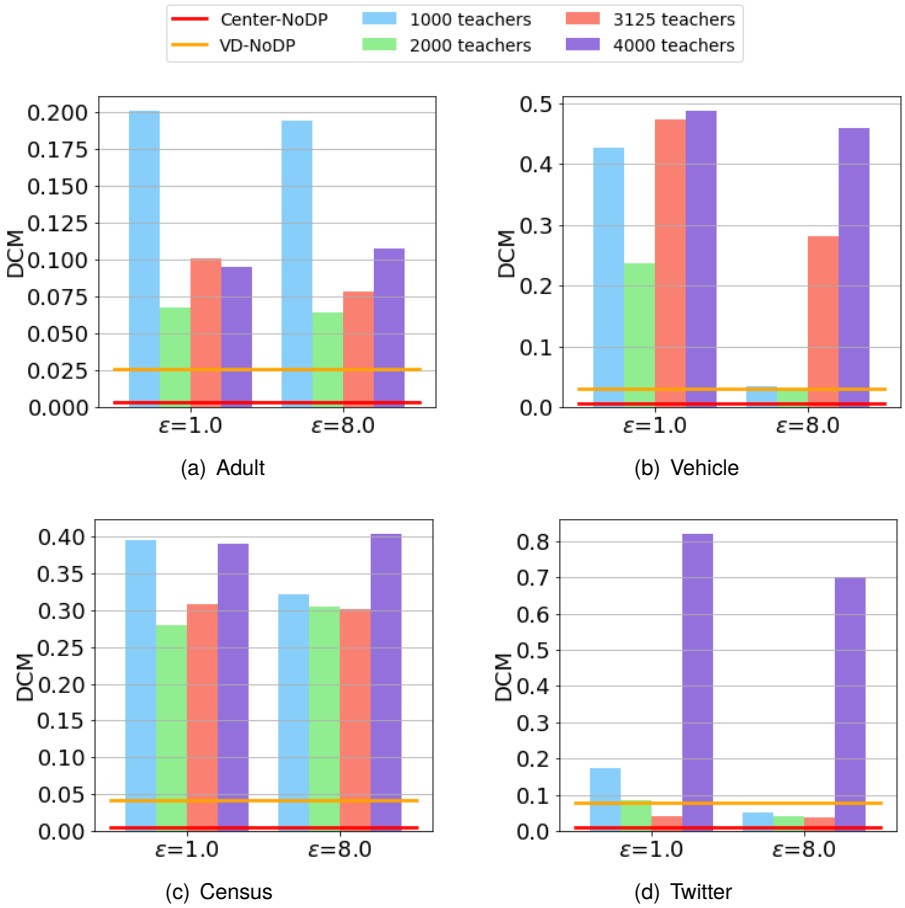


Figure 5.13: DCM Scores of VDGAN-PATE with Different Number of Teachers for All Datasets

6. Conclusion

Databases held by different parties are usually vertically partitioned with different attributes of the same group of users. In light of the challenge that the private partitioned data cannot be collected by an untrusted curator, this thesis proposes a GAN-based framework for vertically partitioned data generation. Besides, DP protocols are integrated into the framework to resist the potential inference attack from adversaries. This work firstly deals with the following research questions for the framework realization:

- *How to deploy a GAN model into the vertically distributed settings?* - I construct a global generator on the server side and a local discriminator on each client side, which are trained alternatively during the optimization. In each iteration, the server would pass the partitioned synthetic data to clients according to their local attributes, and the client would upload the gradients of synthetic data, which carries the useful information from discriminators to improve the model quality of the global generator.

- *How to implement DP into the model training process?* - I introduce two existing DP protocols, DPSGD and PATE, which are designed for deep-learning models specifically, to the client part of our VDGAN framework. The formal protocol is integrated into training local discriminators, which would perturb the gradients with noise during backpropagation. The latter variant consists of an ensemble of teacher discriminators on each client, as well as data compression techniques during gradients aggregation with noise.

Then the proposed frameworks are evaluated in terms of utility and privacy perspectives with multiple metrics. Compared with the baselines, the following findings can be concluded:

- The existing works, e.g. DPLT [41], use tree-based models to publish synthetic data from vertical partitions, which limits the data type to categorical. The proposed neuron-network-based VDGAN model can easily adapt multiple data types by changing the activation functions of the output layer in the generator.

- For the data utility, a number of tests are conducted to measure the similarities between synthetic and real data as well as the different performances in AI training tasks. The results show that the quality of models decreases to some degree after employing DP algorithms from that without DP. While under the same privacy budget, the approaches VDGAN-DPSGD/PATE performs better

than tree-based DPLT generally, especially for the complicated datasets with higher dimensions.

- The MIA tests are employed to evaluate the privacy intensity quantitatively. The proposed models are promising for a balance of privacy-utility trade-off. However, the problem of how to keep the synthetic data in high quality even if the target privacy budgets decrease to a low value still requires further consideration.
- In some exceptions, the approaches VDGAN-DPSGD/PATE have disappointing performances compared with tree-based DPLT for the simpler datasets in relatively low dimensions. Because of this limitation, the proposed frameworks in this thesis are suggested to be utilized for complicated datasets.

Bibliography

[1] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang. “Deep learning with differential privacy”. In: *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 2016, pp. 308–318.

[2] M. Arjovsky, S. Chintala, and L. Bottou. “Wasserstein generative adversarial networks”. In: *International conference on machine learning*. PMLR. 2017, pp. 214–223.

[3] J. Chen, F. Qiang, and N. Ruan. “Adversarial Representation Sharing: A Quantitative and Secure Collaborative Learning Framework”. In: *arXiv preprint arXiv:2203.14299* (2022).

[4] D. Dua and C. Graff. *UCI Machine Learning Repository*. 2017. URL: `https://archive.ics.uci.edu/ml/datasets/Adult`.

[5] D. Dua and C. Graff. *UCI Machine Learning Repository*. 2017. URL: `https://archive.ics.uci.edu/ml/datasets/US+Census+Data+(1990)`.

[6] M. F. Duarte and Y. H. Hu. “Vehicle classification in distributed sensor networks”. In: *Journal of Parallel and Distributed Computing* 64(7) (2004), pp. 826–838.

[7] J. C. Duchi, M. I. Jordan, and M. J. Wainwright. “Local privacy and statistical minimax rates”. In: *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*. IEEE. 2013, pp. 429–438.

[8] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. “Our data, ourselves: Privacy via distributed noise generation”. In: *Annual international conference on the theory and applications of cryptographic techniques*. Springer. 2006, pp. 486–503.

[9] C. Dwork, F. McSherry, K. Nissim, and A. Smith. “Calibrating noise to sensitivity in private data analysis”. In: *Theory of cryptography conference*. Springer. 2006, pp. 265–284.

[10] C. Dwork, G. N. Rothblum, and S. Vadhan. “Boosting and differential privacy”. In: *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*. IEEE. 2010, pp. 51–60.

[11] C. Dwork, A. Roth, et al. “The algorithmic foundations of differential privacy.” In: *Found. Trends Theor. Comput. Sci.* 9(3-4) (2014), pp. 211–407.

[12] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. “Generative adversarial nets”. In: *Advances in neural information processing systems* 27 (2014).

[13] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola. “A kernel two-sample test”. In: *The Journal of Machine Learning Research* 13(1) (2012), pp. 723–773.

[14] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. “Improved training of wasserstein gans”. In: *Advances in neural information processing systems* 30 (2017).

[15] F. Harder, K. Adamczewski, and M. Park. “Dp-merf: Differentially private mean embeddings with randomfeatures for practical privacy-preserving data generation”. In: *International conference on artificial intelligence and statistics*. PMLR. 2021, pp. 1819–1827.

[16] M. Herdin, N. Czink, H. Ozcelik, and E. Bonek. “Correlation matrix distance, a meaningful measure for evaluation of non-stationary MIMO channels”. In: *2005 IEEE 61st Vehicular Technology Conference*. Vol. 1. IEEE. 2005, pp. 136–140.

[17] B. Hilprecht, M. Härterich, and D. Bernau. “Monte Carlo and Reconstruction Membership Inference Attacks against Generative Models”. In: *PoPETs* 2019 (2019), pp. 232–249.

[18] W. Jiang and C. Clifton. “A secure distributed framework for achieving k-anonymity”. In: *The VLDB journal* 15(4) (2006), pp. 316–333.

[19] J. Jordon, J. Yoon, and M. Van Der Schaar. “PATE-GAN: Generating synthetic data with differential privacy guarantees”. In: *International conference on learning representations*. 2018.

[20] F. Kawala, A. Douzal-Chouakria, E. Gaussier, and E. Dimert. “Prédictions d’activité dans les réseaux sociaux en ligne”. In: *4ième conférence sur les modèles et l’analyse des réseaux: Approches mathématiques et informatiques*. 2013, p. 16.

[21] S. P. Liew, T. Takahashi, and M. Ueno. “PEARL: Data Synthesis via Private Embeddings and Adversarial Reconstruction Learning”. In: *arXiv preprint arXiv:2106.04590* (2021).

[22] Y. Long, B. Wang, Z. Yang, B. Kailkhura, A. Zhang, C. Gunter, and B. Li. “G-PATE: Scalable Differentially Private Data Generator via Private Aggregation of Teacher Discriminators”. In: *Advances in Neural Information Processing Systems* 34 (2021).

[23] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Y. Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: `https://www.tensorflow.org/`.

[24] F. D. McSherry. "Privacy integrated queries: an extensible platform for privacy-preserving data analysis". In: *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*. 2009, pp. 19–30.

[25] I. Mironov. "Rényi differential privacy". In: *2017 IEEE 30th computer security foundations symposium (CSF)*. IEEE. 2017, pp. 263–275.

[26] I. Mironov, K. Talwar, and L. Zhang. "R\'enyi differential privacy of the sampled gaussian mechanism". In: *arXiv preprint arXiv:1908.10530* (2019).

[27] M. Mirza and S. Osindero. "Conditional generative adversarial nets". In: *arXiv preprint arXiv:1411.1784* (2014).

[28] N. Mohammed, D. Alhadidi, B. C. Fung, and M. Debbabi. "Secure two-party differentially private data release for vertically partitioned data". In: *IEEE transactions on dependable and secure computing* 11(1) (2013), pp. 59–71.

[29] N. Mohammed, B. Fung, and M. Debbabi. "Anonymity meets game theory: secure data integration with malicious participants". In: *The VLDB Journal* 20(4) (2011), pp. 567–588.

[30] A. Ng et al. "Sparse autoencoder". In: *CS294A Lecture notes* 72(2011) (2011), pp. 1–19.

[31] A. B. Owen. "Monte Carlo theory, methods and examples". In: (2013).

[32] N. Papernot, M. Abadi, U. Erlingsson, I. Goodfellow, and K. Talwar. "Semi-supervised knowledge transfer for deep learning from private training data". In: *arXiv preprint arXiv:1610.05755* (2016).

[33] N. Papernot, S. Song, I. Mironov, A. Raghunathan, K. Talwar, and Ú. Erlingsson. "Scalable private learning with pate". In: *arXiv preprint arXiv:1802.08908* (2018).

[34] N. Park, M. Mohammadi, K. Gorde, S. Jajodia, H. Park, and Y. Kim. "Data synthesis based on generative adversarial networks". In: *arXiv preprint arXiv:1806.03384* (2018).

[35] W. Qardaji, W. Yang, and N. Li. “Priview: practical differentially private release of marginal contingency tables”. In: *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. 2014, pp. 1435–1446.

[36] A. Radford, L. Metz, and S. Chintala. “Unsupervised representation learning with deep convolutional generative adversarial networks”. In: *arXiv preprint arXiv:1511.06434* (2015).

[37] A. Rényi. “On measures of entropy and information”. In: *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*. Vol. 4. University of California Press. 1961, pp. 547–562.

[38] R. Shokri, M. Stronati, C. Song, and V. Shmatikov. “Membership inference attacks against machine learning models”. In: *2017 IEEE symposium on security and privacy (SP)*. IEEE. 2017, pp. 3–18.

[39] L. Sweeney. “k-anonymity: A model for protecting privacy”. In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10(05) (2002), pp. 557–570.

[40] R. Tajeddine, J. Jälkö, S. Kaski, and A. Honkela. “Privacy-preserving Data Sharing on Vertically Partitioned Data”. In: *arXiv preprint arXiv:2010.09293* (2020).

[41] P. Tang, X. Cheng, S. Su, R. Chen, and H. Shao. “Differentially private publication of vertically partitioned data”. In: *IEEE Transactions on Dependable and Secure Computing* 18(2) (2019), pp. 780–795.

[42] U. T. Tantipongpipat, C. Waites, D. Boob, A. A. Siva, and R. Cummings. “Differentially private synthetic mixed-type data generation for unsupervised learning”. In: *2021 12th International Conference on Information, Intelligence, Systems & Applications (IISA)*. IEEE. 2021, pp. 1–9.

[43] A. Torfi, E. A. Fox, and C. K. Reddy. “Differentially private synthetic medical data generation using convolutional gans”. In: *Information Sciences* 586 (2022), pp. 485–500.

[44] R. Torkzadehmahani, P. Kairouz, and B. Paten. “Dp-cgan: Differentially private synthetic data and label generation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2019, pp. 0–0.

[45] C. Villani. “Optimal transport, old and new. Notes for the 2005 Saint-Flour summer school”. In: *Grundlehren der mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]. Springer* (2008).

[46] B. Waggener, W. N. Waggener, and W. M. Waggener. *Pulse code modulation techniques*. Springer Science & Business Media, 1995.

[47] B. Wang, F. Wu, Y. Long, L. Rimanic, C. Zhang, and B. Li. “Datalens: Scalable privacy preserving training via gradient compression and aggregation”. In: *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. 2021, pp. 2146–2168.

[48] C. Wang, J. Liang, M. Huang, B. Bai, K. Bai, and H. Li. “Hybrid differentially private federated learning on vertically partitioned data”. In: *arXiv preprint arXiv:2009.02763* (2020).

[49] R. Wang, B. C. Fung, Y. Zhu, and Q. Peng. “Differentially private data publishing for arbitrarily partitioned data”. In: *Information Sciences* 553 (2021), pp. 247–265.

[50] L. Xie, K. Lin, S. Wang, F. Wang, and J. Zhou. “Differentially private generative adversarial network”. In: *arXiv preprint arXiv:1802.06739* (2018).

[51] M. Xu, B. Ding, T. Wang, and J. Zhou. “Collecting and analyzing data jointly from multiple services under local differential privacy”. In: *Proceedings of the VLDB Endowment* 13(12) (2020), pp. 2760–2772.

[52] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao. “Privbayes: Private data release via bayesian networks”. In: *ACM Transactions on Database Systems (TODS)* 42(4) (2017), pp. 1–41.

[53] N. L. Zhang. “Hierarchical latent class models for cluster analysis”. In: *The Journal of Machine Learning Research* 5 (2004), pp. 697–723.

[54] X. Zhang, S. Ji, and T. Wang. “Differentially private releasing via deep generative model (technical report)”. In: *arXiv preprint arXiv:1801.01594* (2018).

[55] Z. Zhao, A. Kunar, R. Birke, and L. Y. Chen. “CTAB-GAN: Effective Table Data Synthesizing”. In: *Asian Conference on Machine Learning*. PMLR. 2021, pp. 97–112.

[56] Z. Zhao, A. Kunar, R. Birke, and L. Y. Chen. “CTAB-GAN+: Enhancing Tabular Data Synthesis”. In: *arXiv preprint arXiv:2204.00401* (2022).

List of Figures

1.1. Partitioned Data 7
1.2. Publication of Vertically Distributed Data 8

3.1. Overview of PATE [32] 19
3.2. Overview of GAN 20
3.3. Latent Tree Model 23
3.4. Overview of DPLT 24

4.1. Overview of the Whole Framework 26
4.2. Overview of VDGAN 27
4.3. Warm-Up for Local Discriminators 29
4.4. Integration of DPSGD in VDGAN 30
4.5. Integration of PATE in VDGAN 31

5.1. k-AVD of Census Dataset over Various Privacy Budgets 42
5.2. 4-AVD of Different Datasets 43
5.3. Heatmaps of Pearson Correlation Matrix for Twitter Dataset 44
5.4. DCM under Different Privacy Budgets 45
5.5. Classification Scores for Twitter Dataset 47
5.6. AUC Scores of MLP Classifier for Different Datasets 48
5.7. MIA-Accuracy with Different Target Data for Adult Dataset 49
5.8. MIA-Accuracy with Different Target Data for Census Dataset 50
5.9. MIA-Accuracy with Different Target Data for Twitter Dataset 50
5.10. MIA-Accuracy against AVD Score for Different Datasets 51
5.11. 4-AVD with Different Number of Clients 53
5.12. Utility Results of VDGAN-PATE with Different Number of Teachers for Twitter Dataset 54
5.13. DCM Scores of VDGAN-PATE with Different Number of Teachers for All Datasets 55

A.1. k-AVDs under Different Privacy Budgets 70
A.2. Maps of Correlation Matrix over Real Data and Synthetic Data from Different Approaches with 1.0-DP 71
A.3. Classification Scores of TSTR Tests with Different Privacy Budgets . . . 72
A.4. MIA-Accuracy with Different Target Data 73
A.5. MIA Accuracy against Scores of Different Utility Metrics 74

A.6. Utility Results with Different Number of Clients 75
A.7. Utility Results of VDGAN-PATE with Different Number of Teachers . . . 76

List of Algorithms

1. Outline of DPSGD 19
2. Outline of WGAN-GP 22

3. An iteration of training VDGAN 28
4. Gradients Compression with DP 34
5. Accumulation of Privacy Budget 36

A. Additional Results

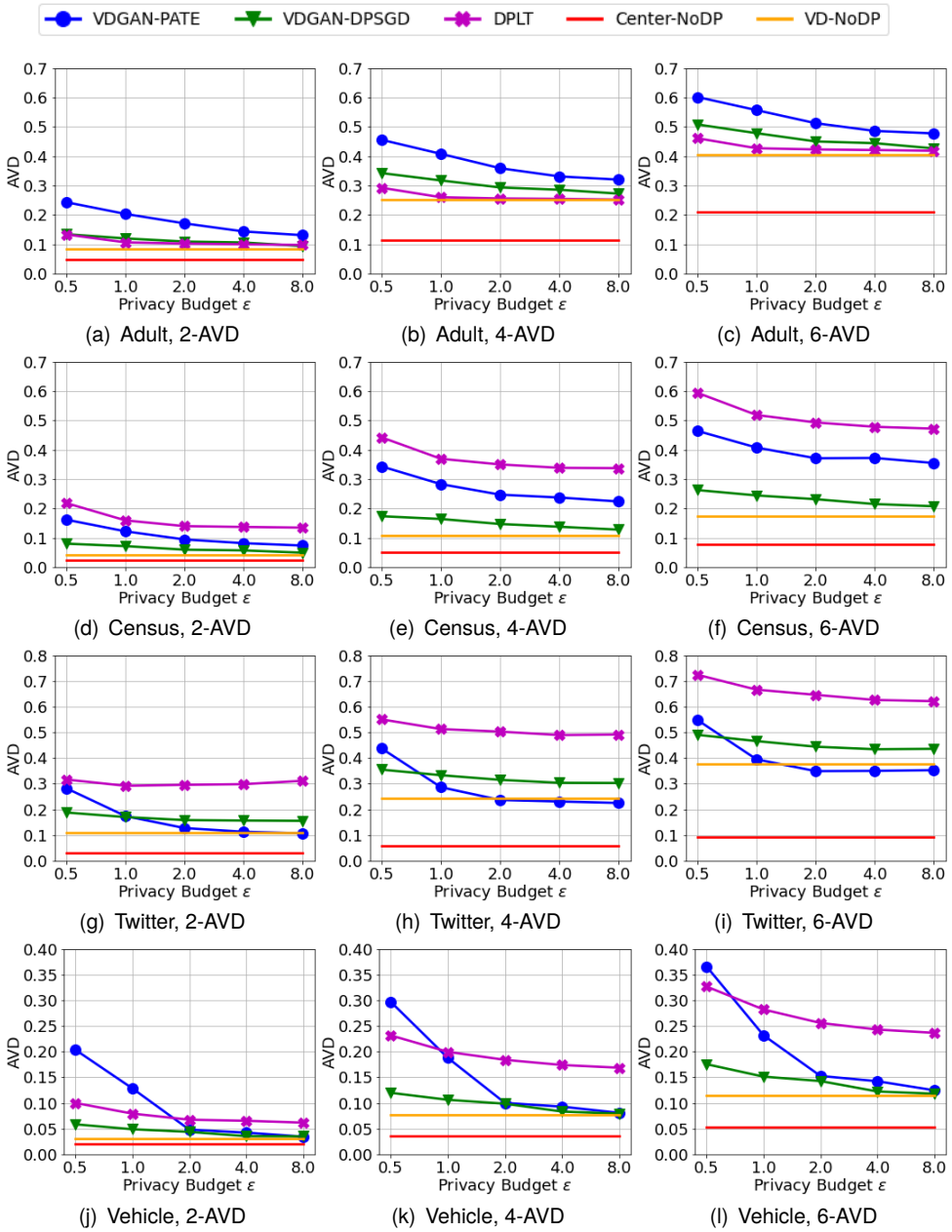
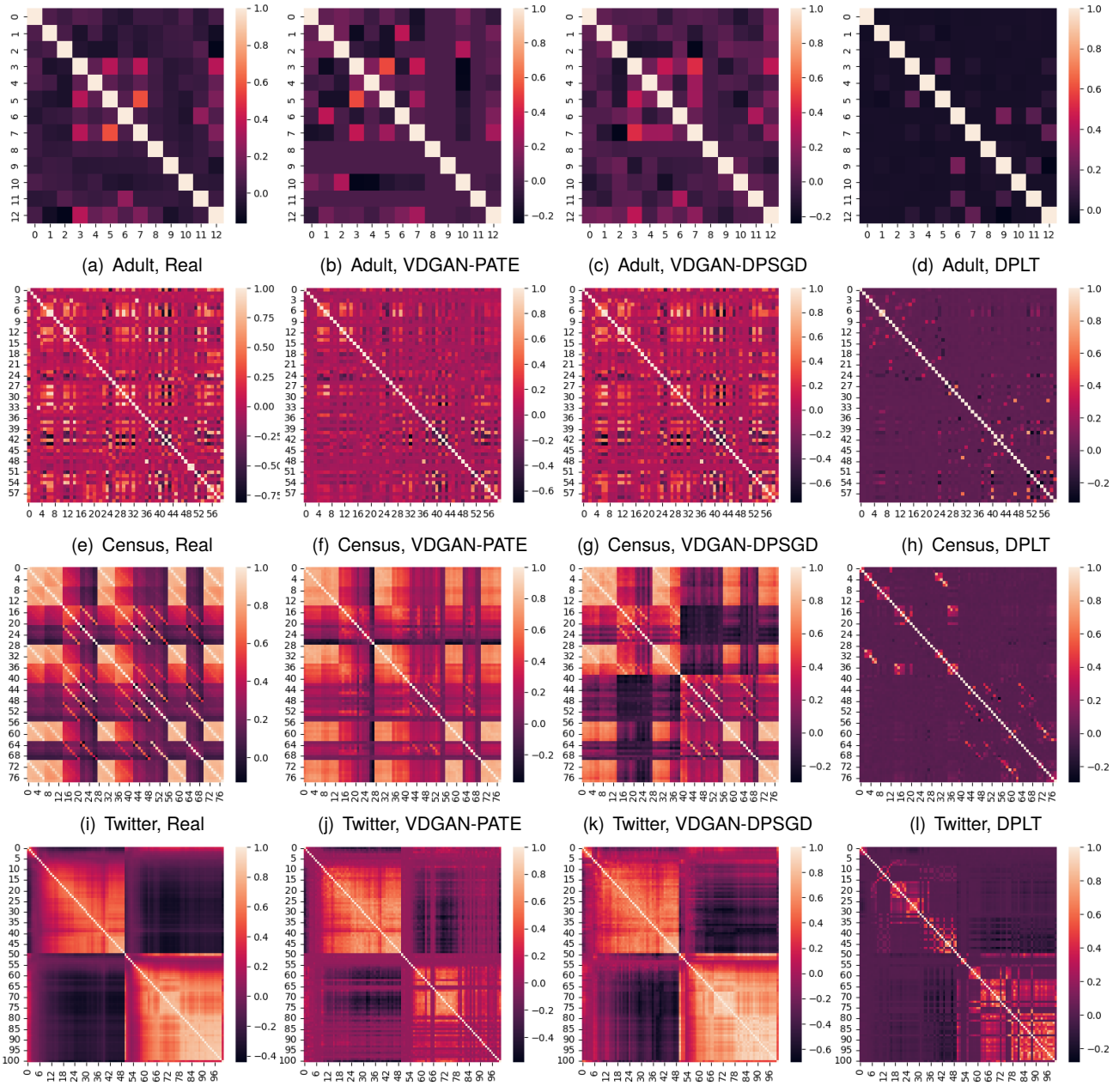


Figure A.1: k-AVDs under Different Privacy Budgets



(a) Adult, Real (b) Adult, VDGAN-PATE (c) Adult, VDGAN-DPSGD (d) Adult, DPLT

(e) Census, Real (f) Census, VDGAN-PATE (g) Census, VDGAN-DPSGD (h) Census, DPLT

(i) Twitter, Real (j) Twitter, VDGAN-PATE (k) Twitter, VDGAN-DPSGD (l) Twitter, DPLT

(m) Vehicle, Real (n) Vehicle, VDGAN-PATE (o) Vehicle, VDGAN-DPSGD (p) Vehicle, DPLT

Figure A.2: Maps of Correlation Matrix over Real Data and Synthetic Data from Different Approaches with 1.0-DP

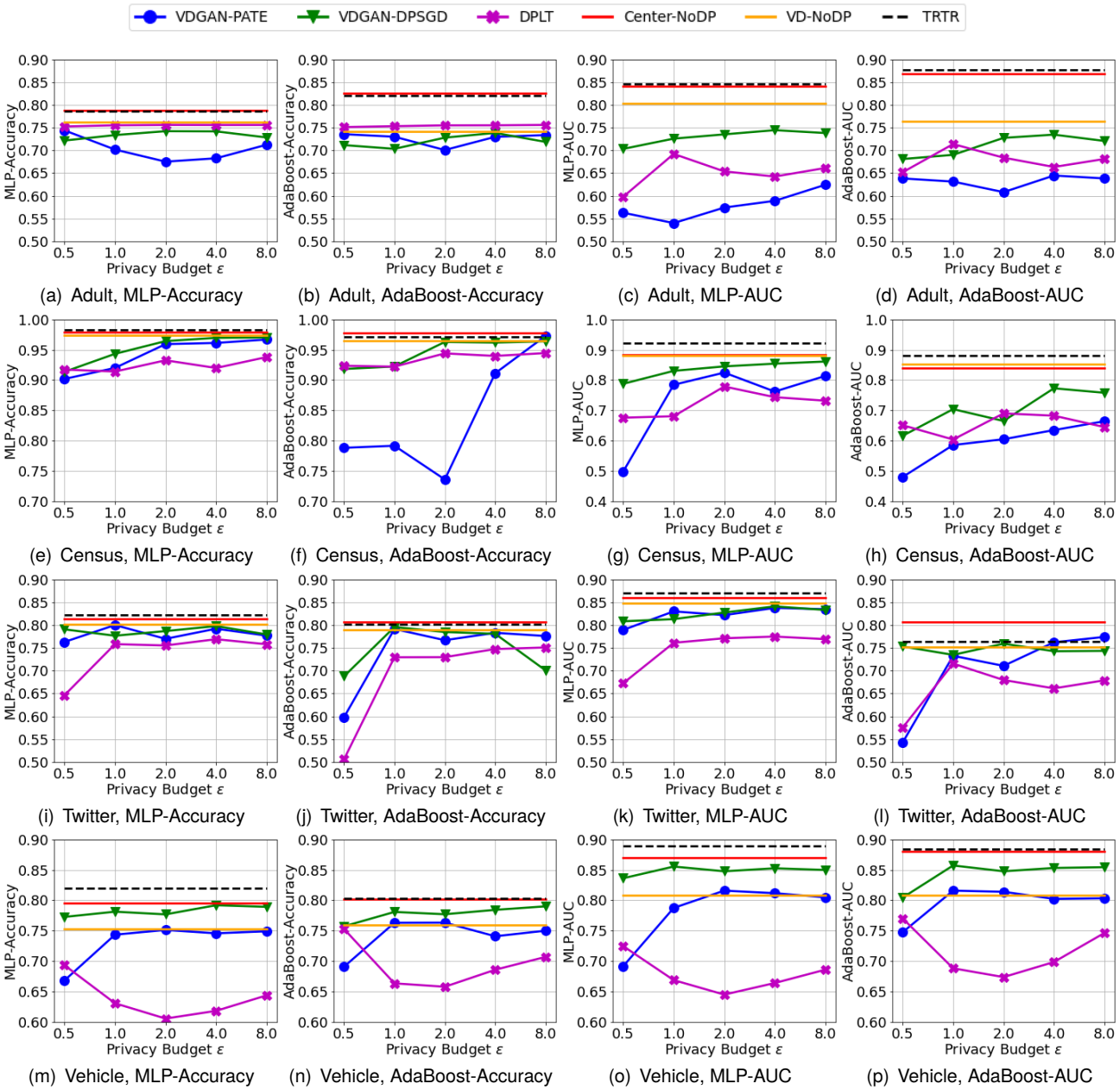
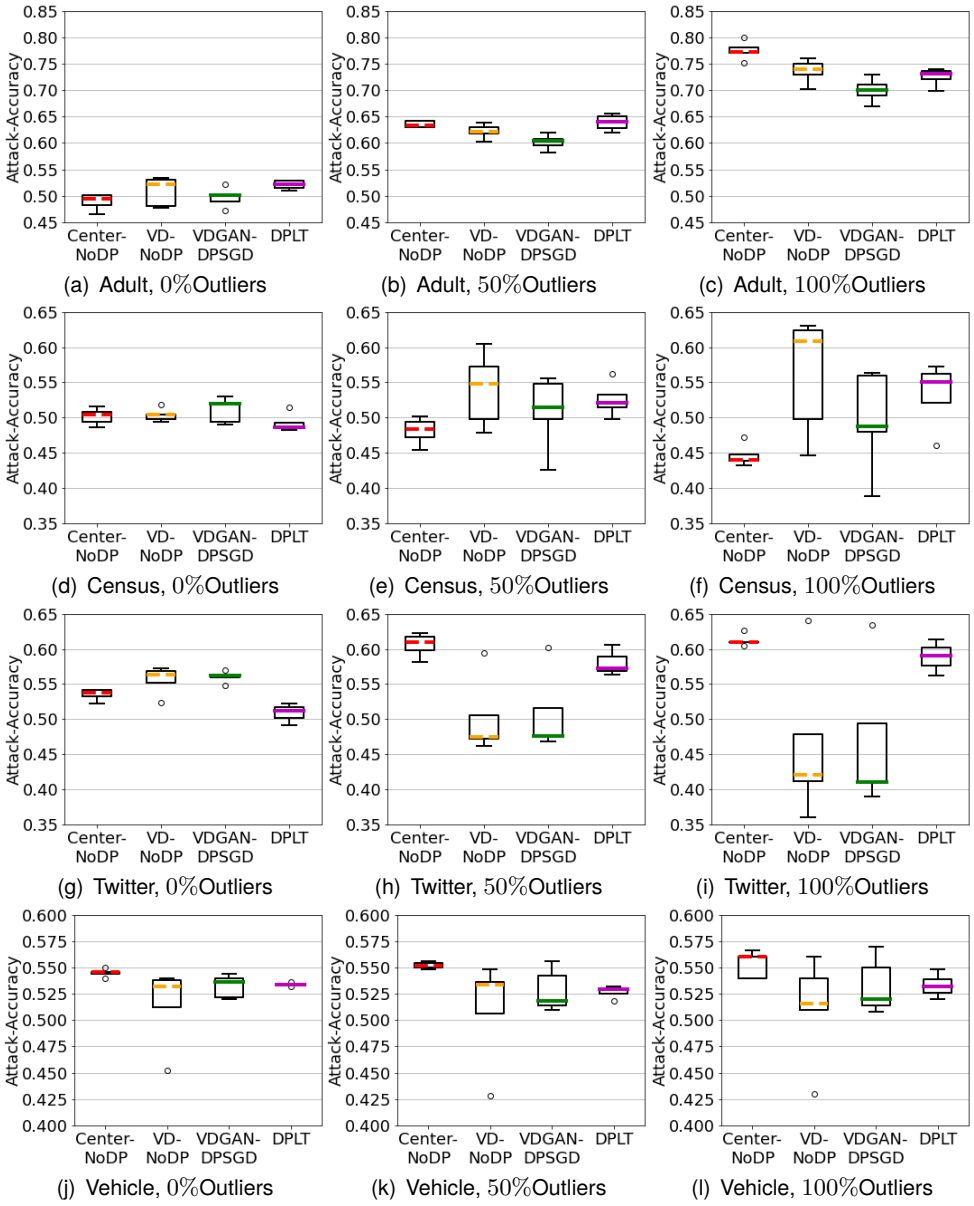


Figure A.3: Classification Scores of TSTR Tests with Different Privacy Budgets



(a) Adult, 0%Outliers (b) Adult, 50%Outliers (c) Adult, 100%Outliers

(d) Census, 0%Outliers (e) Census, 50%Outliers (f) Census, 100%Outliers

(g) Twitter, 0%Outliers (h) Twitter, 50%Outliers (i) Twitter, 100%Outliers

(j) Vehicle, 0%Outliers (k) Vehicle, 50%Outliers (l) Vehicle, 100%Outliers

Figure A.4: MIA-Accuracy with Different Target Data

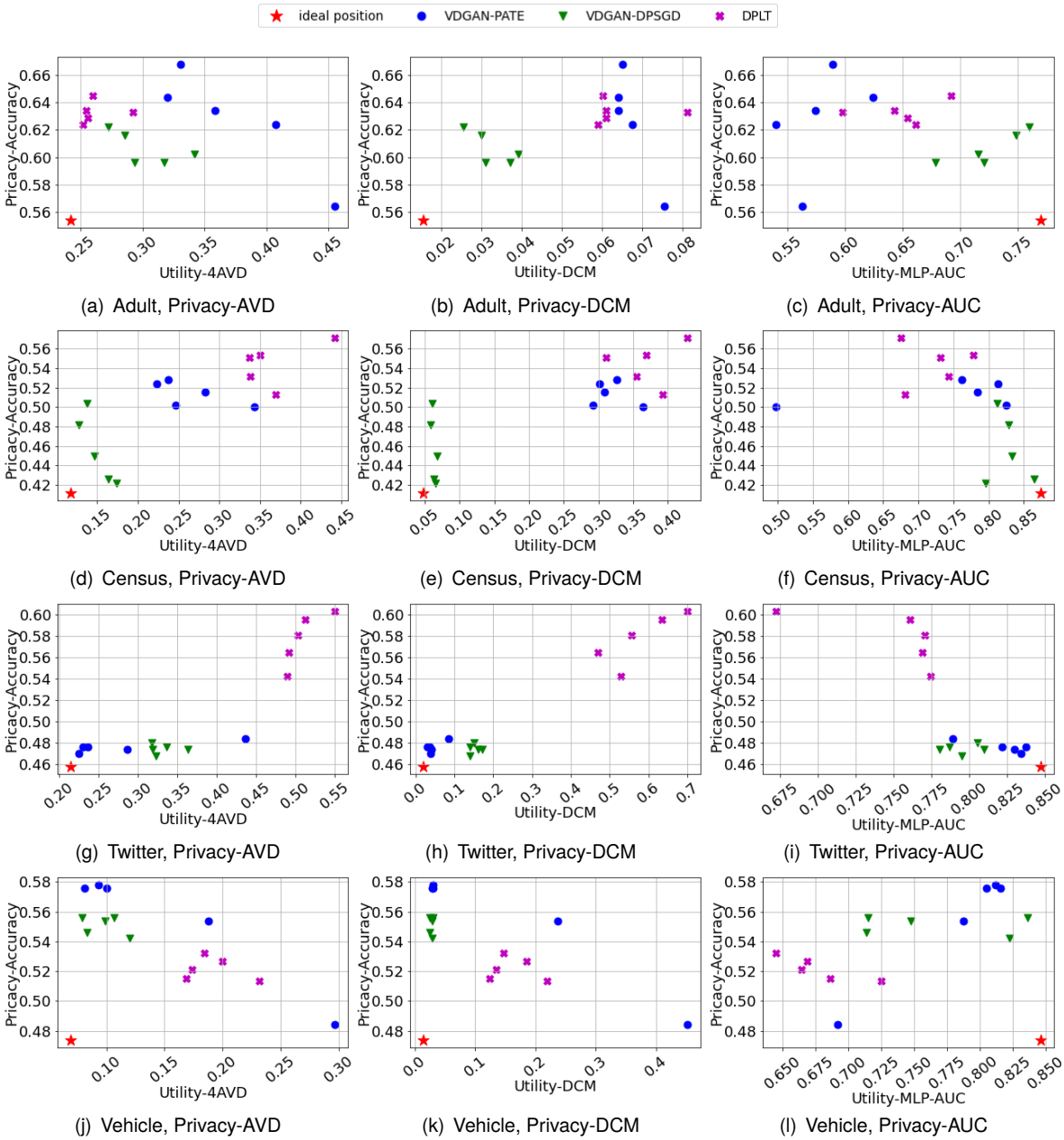


Figure A.5: MIA Accuracy against Scores of Different Utility Metrics

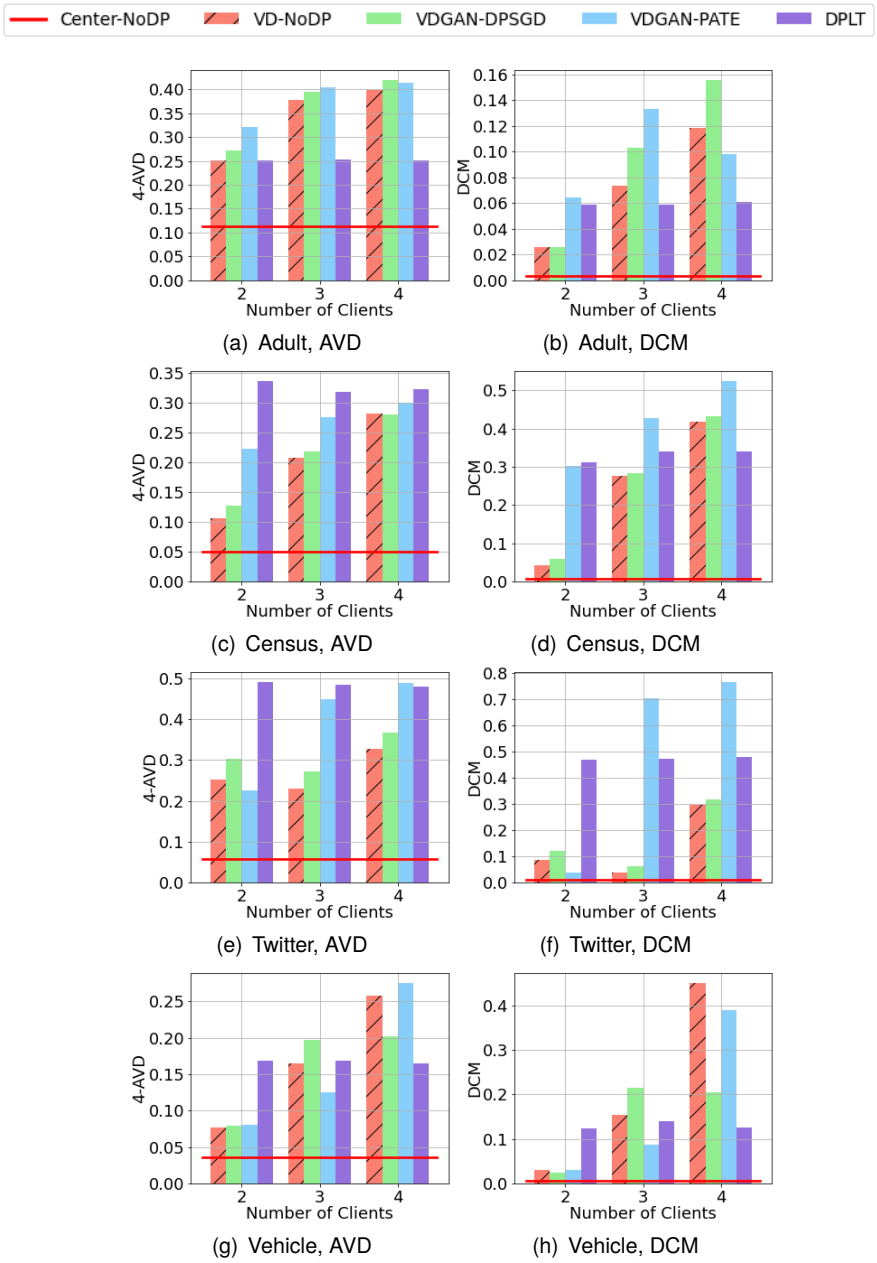


Figure A.6: Utility Results with Different Number of Clients

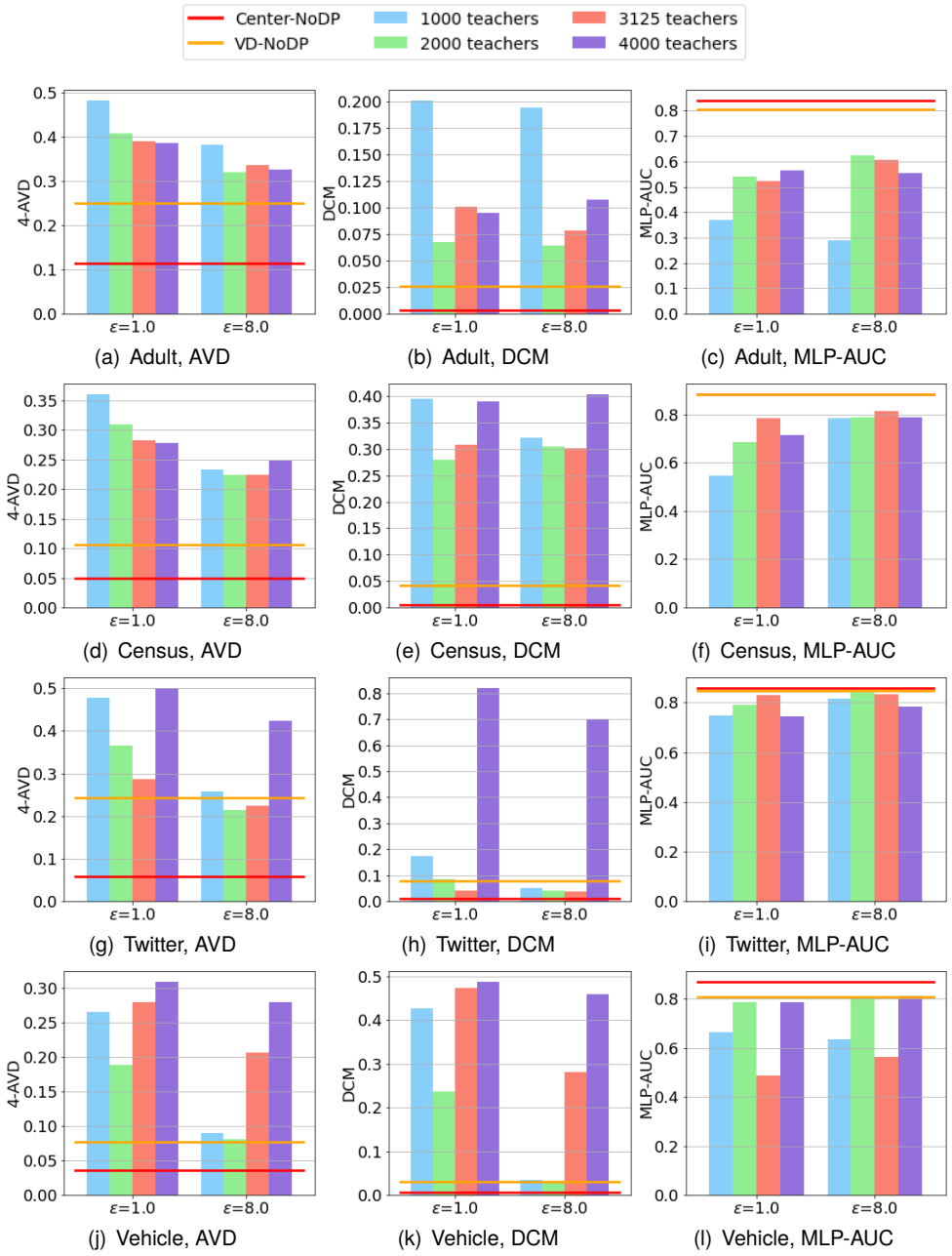


Figure A.7: Utility Results of VDGAN-PATE with Different Number of Teachers