5-2021

Achieving Differential Privacy and Fairness in Machine Learning

Depeng Xu
University of Arkansas, Fayetteville

Achieving Differential Privacy and Fairness in Machine Learning

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy in Engineering with a concentration in Computer Science

by

Depeng Xu
Tsinghua University
Bachelor of Engineering in Environmental Engineering, 2011
University of Arkansas
Master of Science in Statistics and Analytics, 2017

May 2021
University of Arkansas

This dissertation is approved for recommendation to the Graduate Council.

Xintao Wu, Ph.D.
Dissertation Director

Brajendra Panda, Ph.D.
Committee member

Qinghua Li, Ph.D.
Committee member

Mark Arnold, Ph.D.
Committee member

ABSTRACT

Machine learning algorithms are used to make decisions in various applications, such as recruiting, lending and policing. These algorithms rely on large amounts of sensitive individual information to work properly. Hence, there are sociological concerns about machine learning algorithms on matters like privacy and fairness. Currently, many studies only focus on protecting individual privacy or ensuring fairness of algorithms separately without taking consideration of their connection. However, there are new challenges arising in privacy preserving and fairness-aware machine learning. On one hand, there is fairness within the private model, i.e., how to meet both privacy and fairness requirements simultaneously in machine learning algorithms. On the other hand, there is fairness between the private model and the non-private model, i.e., how to ensure the utility loss due to differential privacy is the same towards each group.

The goal of this dissertation is to address challenging issues in privacy preserving and fairness-aware machine learning: achieving differential privacy with satisfactory utility and efficiency in complex and emerging tasks, using generative models to generate fair data and to assist fair classification, achieving both differential privacy and fairness simultaneously within the same model, and achieving equal utility loss w.r.t. each group between the private model and the non-private model.

In this dissertation, we develop the following algorithms to address the above challenges.

1. We develop PrivPC and DPNE algorithms to achieve differential privacy in complex and emerging tasks of causal graph discovery and network embedding, respectively.

2. We develop the fair generative adversarial neural networks framework and three algo-

rithms (FairGAN, FairGAN$^+$ and CFGAN) to achieve fair data generation and classification through generative models based on different association-based and causation-based fairness notions.

3. We develop PFLR and PFLR* algorithms to simultaneously achieve both differential privacy and fairness in logistic regression.

4. We develop a DPSGD-F algorithm to remove the disparate impact of differential privacy on model accuracy w.r.t. each group.

ACKNOWLEDGEMENTS

During my Ph.D. study, I have been fortunate to receive plenty of help and support from many ends. I hereby express my sincere gratitude to all of them.

First and foremost, I am greatly indebted to my advisor, Dr. Xintao Wu. I appreciate all his time, guidance, support, and patience throughout the period of my Ph.D. study. He gave me a chance as a beginner in the field and nurtured me to become a successful researcher. His knowledge and excellency will continue motivating me in my future career.

My gratitude also goes to Dr. Shuhan Yuan and Dr. Lu Zhang with whom I have been truly fortunate to collaborate throughout my Ph.D. journey. Their advice, experience, and ideas are vital in our research projects. I have benefited tremendously from the collaborations and discussions with both of them.

I would like to thank my dissertation committee members, Dr. Qinghua Li, Dr. Brajendra Panda, and Dr. Mark Arnold. Their constructive comments significantly improve the quality of this dissertation.

I also thank my friends and colleagues in the Social Awareness and Intelligent Learning Lab at the University of Arkansas. I appreciate the collaboration with Dr. Yongkai Wu, Wei Du, Qiuping Pan, and Srinidhi Katla. Thanks also go to the other friends and colleagues who give me consistent encouragement: Dr. Panpan Zheng, Wen Huang, Hao Van and Dr. Kevin Labille. I have been truly honored to work with these excellent researchers.

Finally, I would express my deepest gratitude to my family for their unconditional love and support. This dissertation is dedicated to them.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 Introduction

1.1 Motivation

Nowadays, machine learning algorithms are being widely used to automatically make decisions, such as loan application and student admission, based on our individual information. It is important to address individuals' sociological concerns such as privacy and fairness and meet government laws and regulations (e.g., General Data Protection and Regulation on data protection and privacy, and Fair Credit Reporting Act or Equal Credit Opportunity Act on fairness) in training and deploying machine learning algorithms.

Differential privacy is a formal standard for protecting individual privacy in data analysis [1]. It has been established as a standard privacy model to achieve opt-out right of individuals. Generally speaking, differential privacy guarantees the query results or the released model cannot be exploited by attackers to derive whether one particular record is present or absent in the underlining dataset. It ensures that the inclusion or exclusion of a single record from a dataset makes no statistical difference when we perform a data analysis task on the dataset. The mechanisms to achieve differential privacy mainly include the classic approach of adding Laplacian noise [1], the exponential mechanism [2], the objective perturbation approach [3], the functional perturbation approach [4], and the sample and aggregate framework [5]. There have been many studies on the application of differential privacy in some particular analysis tasks, e.g. data collection [6, 7], stochastic gradient descents [8], regression [9], spectral graph analysis [10], and deep learning models[11, 12]. There are still many complex and emerging tasks under-exploited on how to achieve differential privacy with satisfactory utility and efficiency, e.g., causal graph discovery and network embedding.

Fairness-aware learning is increasingly receiving attention in the machine learning field. Discrimination indicates unfair treatment towards individuals based on the group to which they are perceived to belong. In machine learning, discrimination may be unintentional but have powerful effect on vulnerable groups. There are two major tasks in fairness-aware machine learning: (1) detecting and removing discrimination from datasets and (2) building models that make fair predictions. The first task is important for data owners to release data for various purposes, including scientific data analysis and training machine learning models. For the second task, when building machine learning models, if there exist historical biased decisions against the protected group in the training data, models learned from such data may also make discriminative predictions against the protected group [13, 14, 15]. In addition, the learning process can also introduce biases into predicted decisions [16]. Current research to achieve fair classification can be mainly categorized into two groups: in-processing methods which incorporate fairness constraints into the classification models [17, 18], and pre/post-processing methods which modify the training data and/or the potentially unfair predictions made by the classifiers [19, 20, 21, 22, 23]. There is lack of study on how to use generative models to generate fair data and to assist fair classification.

Currently, many studies only focus on protecting individual privacy or ensuring fairness of algorithms separately without taking consideration of their connection. However, there are new challenges arising in privacy preserving and fairness-aware machine learning. On one hand, there is fairness within the private model, i.e., how to meet both privacy and fairness requirements simultaneously in the same machine learning algorithms. Currently, many studies focus on only protecting individual privacy or ensuring fairness of algorithms. However, how to meet both privacy and fairness requirements simultaneously in machine learning algorithms is under exploited. On the other hand, there may exist fairness in terms

of utility loss between the private model and the non-private model, i.e., how to ensure the utility loss due to differential privacy is fair towards each group. When we enforce differential privacy in machine learning, the utility-privacy trade-off is different w.r.t. each group. Gradient clipping and random noise addition disproportionately affect underrepresented and more complex classes and subgroups, which results in inequality in utility loss. The inequality in utility loss by differential privacy needs to be analyzed and mitigated.

1.2 Overview

The goal of this dissertation is to address challenging issues in privacy preserving and fairness-aware machine learning.

First, we focus on causal graph discovery and network embedding, two complex and emerging tasks, and explore how to use different privacy preserving mechanisms to achieve differential privacy in these two tasks. For causal graph discovery, discovering causal relationships by constructing the causal graph provides critical information to researchers and decision makers. Yet releasing causal graphs may risk individual participant's privacy. It is very under-exploit how to enforce privacy preservation in causal graph discovery. For network embedding, learning the low-dimensional representations of the vertices in a network can help users understand the network structure and perform other data mining tasks efficiently. Various network embedding approaches such as DeepWalk [24] and LINE [25] have been developed recently. However, how to protect the individual privacy in network embedding has not been exploited. It is challenging to achieve high utility as the sensitivity of stochastic gradients in random walks and that of edge sampling are very high. We compare a variety of mechanisms and propose efficient novel private algorithms to perform causal graph discovery (Chapter 4) and network embedding (Chapter 5) with privacy guarantee.

Second, we explore how to use generative models to generate fair data and to assist fair classification. How to achieve fairness is important for machine learning. Two tasks that are equally important in fair machine learning are how to obtain fair datasets and how to build fair classifiers. Instead of removing the discrimination from the existing dataset, we focus on generating fair data. Generative adversarial networks [26] can generate high quality synthetic data that are indistinguishable from real data. However, the generated data can inherit the historical bias from the real data. We study different kinds of fairness notions: statistical parity [27] and ϵ-fairness [14] in fair data generation; demographic parity, equality of opportunity and equality of odds [16] in fair classification. Meanwhile, recent researches [28, 29, 30, 31, 32, 33] showed that fairness should be studied from the causal perspective, and proposed a number of fairness criteria based on Pearl's causal modeling framework [34]. We also study fairness notions based on total effect, path-specific effect, and counterfactual effect in causation-based fair data. We investigate the problem of building fairness-aware generative adversarial networks, which can learn a close distribution from a given dataset while also ensuring fairness in data generation and classification based on different association-based (Chapter 6) or causation-based (Chapter 7) fairness notions.

Last but not least, we address the new challenges arising in privacy preserving and fairness-aware machine learning by examining the relationship between their definitions and mechanisms to achieve them and their trade-offs. When we enforce differential privacy onto a regular non-private model, the model trades some utility off for privacy. On one hand, with the impact of differential privacy, the within-model unfairness in the private model may be different from the one in the non-private model. Using logistic regression as an example, we study how to simultaneously achieve both differential privacy and fairness within the same model (Chapter 8). On the other hand, differential privacy may introduce additional

discriminative effect towards the protected group when we compare the private model with the non-private model. The utility loss between the private and non-private models w.r.t. each group, such as reduction in group accuracy, may be uneven. We study the inequality in utility loss due to differential privacy w.r.t. groups and compare the change in prediction accuracy w.r.t. each group between the private model and the non-private model (Chapter 9).

The remainder of this dissertation is organized as follows. In Chapter 2, we discuss related work in differential privacy and fairness-aware machine learning to provide a general review of research achievements in this research field. We introduce the preliminary background for differential privacy and fairness-aware machine learning in Chapter 3. The extra highly related work and preliminary are given at the beginning of each research chapter, as necessary. We present the main body of this dissertation in Chapters 4-9. We conclude this dissertation with a discussion of future work in Chapter 10.

1.3 Summary of Contributions

In Chapter 4, we focus on the PC algorithm [35], a classic constraint-based causal graph discovery algorithm, and propose a differentially private PC algorithm (PrivPC) for categorical data. PrivPC adopts the exponential mechanism [36] and significantly reduces the number of edge elimination decisions. Therefore, it incurs much less privacy budget than the naive approaches that add privacy protection at each conditional independence test. For numerical data, we further develop a differentially private causal discovery algorithm (PrivPC*). The idea is to add noise once onto the covariance matrix from which partial correlations used for conditional independence test can be derived. Experimental results show that PrivPC and PrivPC* achieve good utility and robustness for different settings of causal graphs. To our best knowledge, this is the first work on how to enforce differential

privacy in constraint-based causal graph discovery.

In Chapter 5, we focus on developing a differential privacy preserving method of network embedding based on the equivalent matrix factorization method [37]. We develop a differentially private network embedding method (DPNE). In this method, we leverage the findings that network embedding methods such as DeepWalk and LINE are equivalent to factorization of some matrices derived from the adjacency matrix of the original network and apply objective perturbation on the objective function of matrix factorization. We show that with only adding a small amount of noise onto the objective function, the learned low-dimensional representations satisfy differential privacy. Experimental results show that the embedded representations learned by DPNE achieve good utility with a small privacy budget on both vertex classification and link prediction tasks. To our best knowledge, this is the first work on how to preserve differential privacy in network embedding.

In Chapter 6, we develop a new generative adversarial network (GAN) model, named FairGAN, and its enhanced version FairGAN$^+$. FairGAN contains a generator to generate close-to-real data samples and two discriminators to assist adversarial learning. FairGAN can learn a generator producing fair data and also preserving good data utility. In addition to FairGAN, FairGAN$^+$ also contains a classifier to predict class labels and one more discriminators to assist adversarial learning. FairGAN$^+$ simultaneously achieves fair data generation and fair classification by co-training a generative model and a classifier through joint adversarial games with the discriminators. Experiments using real world census data show that the generator of FairGAN/FairGAN$^+$ can achieve fair data generation with good data utility and free from disparate treatment and disparate impact. The classifier of FairGAN$^+$ can achieve guaranteed classification fairness in notions of demographic parity or equality of odds with good classification utility. The co-training of the generative model and the classifier improves

the performances of each other.

In Chapter 7, we develop a causal fairness-aware generative adversarial network (CFGAN) for generating data that achieve various causal-based fairness criteria. CFGAN adopts two generators, whose structures are purposefully designed to reflect the structures of causal graph and interventional graph. Therefore, the two generators can respectively simulate the underlying causal model that generates the real data, as well as the causal model after the intervention. On the other hand, two discriminators are used for producing a close-to-real distribution, as well as for achieving various fairness criteria based on causal quantities simulated by generators. Experiments on a real-world dataset show that CFGAN can generate high quality fair data.

In Chapter 8, we develop PFLR and PFLR* algorithms to achieve both differential privacy and fairness in logistic regression. In particular, our enhanced method (PFLR*), which adds Laplace noise with non-zero mean as equivalence to fairness constraint, can reduce the amount of added noise and hence better preserve utility. Our idea is based on the connection between ways of achieving differential privacy and fairness. We conduct evaluation on two real-world datasets and results show that our approaches meet both differential privacy and fairness requirements while achieving good utility. To our best knowledge, this is the first work to study how to achieve both differential privacy and fairness in classification models.

In Chapter 9, we provide theoretical analysis on the group level cost of privacy and show the source of disparate impact of differential privacy on each group in the original DPSGD [38]. Then, we propose a modified DPSGD algorithm, called DPSGD-F, to achieve differential privacy with equal utility loss w.r.t. each group. It uses adaptive clipping to adjust the sample contribution of each group, so the privacy level w.r.t. each group is calibrated

based on their cost of privacy. As a result, the final group utility loss is the same for each group in DPSGD-F. In our experimental evaluation, we show how group sample size and group clipping bias affect the impact of differential privacy in DPSGD, and how adaptive clipping for each group helps to mitigate the disparate impact caused by differential privacy in DPSGD-F.

2 Related Work

2.1 Differential Privacy

Differential privacy is widely used on data privacy. The concept of differential privacy was first proposed in Sub-Linear Queries (SuLQ) output perturbation framework of different statistical queries learning models [39]. Many mechanisms have been proposed to enforce differential privacy.

Dwork et al. [1] proved using Laplace mechanism can preserve differential privacy by calibrating the standard deviation of the noise according to the sensitivity of the query function. Laplace mechanism can be used to privately release contingency tables [40], histograms [41], data cubes [42], and spatial decomposition [43].

McSherry and Talwar [36] proposed the exponential mechanism to guarantee differential privacy in non-numeric sensitive queries by sampling according to a mapping function instead of adding noise. The exponential mechanism has been widely explored in several private learning algorithms, such as logistic regression, support vector machine, k-means clustering [44, 45], decision trees [46], and genetic association framework [47]. In Johnson and Shmatikov's work [47], they used the exponential mechanism to privately release the number of significant SNPs (genes) associated with a trait and to privately release the location (GenBank ID) of the significant SNPs. Zhang et al. [48] proposed a differentially private Bayesian network via randomly selecting attribute-parent pairs based on mutual information using exponential mechanism.

For many data mining and machine learning algorithms, we usually optimize some objective functions (e.g., cross entropy) to derive coefficients of released models. Rather than

adding noise to coefficients of the released model, Chaudhuri et al. [3] proposed an objective perturbation approach by perturbing the objective function which is convex and doubly differentiable. Zhang et al. [49] further proposed a functional mechanism to enforce differential privacy on general optimization-based models, such as linear regression and logistic regression.

Nissim et al. [50] introduced a smooth sensitivity and sample and aggregate framework. It calibrates instance-specific noise based on a smooth sensitivity to achieve rigorous differential privacy.

Existing literature in differentially private machine learning targets both convex and non-convex optimization algorithms and can be divided into three main classes, input perturbation, output perturbation, and inner perturbation. Input perturbation approaches [51] add noise to the input data based on local differential privacy model. Output perturbation approaches [52] add noise to the model after the training procedure finishes, i.e., without modifying the training algorithm. Inner perturbation approaches modify the learning algorithm such that the noise is injected during learning. For example, research in [53] modifies the objective of the training procedure, and research in [38] adds noise to the gradient output of each step of the training without modifying the objective.

Research in [38] proposed the idea that limiting users to small contributions keeps noise level at the cost of introducing bias. Research in [54] characterizes the trade-off between bias and variance, and shows that (1) a proper bound can be found depending on properties of the dataset and (2) a concrete cost of privacy cannot be avoided simply by collecting more data. Several works study how to adaptively bound the contributions of users and clip the model parameters to improve learning accuracy and robustness. Research in [55] uses coordinate-wise adaptive clipping of the gradient to achieve the same privacy guarantee with

much less added noise. In federated learning setting, the proposed approach [56] adaptively clips to a value at a specified quantile of the distribution of update norms, where the value at the quantile is itself estimated online, with differential privacy. Other than adaptive clipping, research in [57] adaptively injects noise into features based on the contribution of each to the output so that the utility of deep neural networks under -differential privacy is improved; [58] adaptively allocates per-iteration privacy budget to achieve zCDP on gradient descent.

2.2 Fairness-aware Machine Learning

With the wide adoption of automated decision making systems, fairness-aware learning or anti-discrimination learning becomes an increasingly important task. In fairness-aware learning, discrimination prevention aims to remove discrimination by modifying the biased data and/or the predictive algorithms built on the data. Many approaches have been proposed for constructing discrimination-free classifiers, which can be broadly classified into three categories: the pre-process approaches that modify the training data to remove discriminatory effect before conducting predictive analytics, the in-process approaches that enforce fairness to classifiers by introducing constraints or regularization terms to the objective functions, and the post-process approaches that directly change the predicted labels.

The pre-process approaches that modify the training data are widely studied. The fundamental assumption of the pre-process methods is that, once a classifier is trained on a discrimination-free dataset, the prediction made by the classifier will also be discrimination free [59]. Research in [27] proposed several methods for modifying data including Massaging, which corrects the labels of some individuals in the data, Reweighting, which assigns weights to individuals to balance the data, and Sampling, which changes the sample sizes of different subgroups to remove the discrimination in the data. In [14], authors further studied how to

remove disparate impact by modifying the distribution of the unprotected attributes such that the protected attribute cannot be estimated from the unprotected attributes. Research in [15] proposed a causal graph based approach that removes discrimination based on the block set and ensures that there is no discrimination in any meaningful partition. For the in-process approaches, some tweak or regularizers are applied to the classifier to penalize discriminatory prediction during the training process. In principle, preventing discrimination when training a classifier consists of balancing two contrasting objectives: maximizing the accuracy of the extracted predictive model and minimizing the number of predictions that are discriminatory. Research in [19] proposed a predictive model for maximizing utility subject to the fair constraint that achieves both statistical parity and individual fairness, i.e., similar individuals should be treated similarly. In [16], authors proposed a framework for optimally adjusting any predictive model so as to remove discrimination.

Reweighting or sampling changes the importance of training samples according to an estimated probability that they belong to the protected group so that more importance is placed on sensitive ones [60, 19]. Adaptive sensitive reweighting uses an iterative reweighting process to recognize sources of bias and diminish their impact without affecting features or labels [61]. [62] uses agnostic learning to achieve good accuracy and fairness on all subgroups. However, it requires a large number of iterations, thus incurring a very high privacy loss. Other approaches to balance accuracy across classes include oversampling, adversarial training with a loss function that overweights the underrepresented group, cost-sensitive learning, and resampling.

Recently, several studies have been proposed to remove discrimination through adversarial training. Research in [63] incorporated an adversarial model to learn a discrimination free representation. Based on that, research in [64] studied how the choice of data for the

adversarial training affects the fairness. Studies in [65, 66] further proposed various adversarial objectives to achieve different levels of group fairness including demographic parity, equalized odds and equal opportunity.

Most researches on fairness-aware machine learning study whether the predictive decision made by machine learning model is discriminatory against the protected group [16, 65, 66, 67, 68, 69]. For example, demographic parity requires that a prediction is independent of the protected attribute. Equality of odds [16] requires that a prediction is independent of the protected attribute conditional on the original outcome. These fairness notions focus on achieving non-discrimination within one single model.

As paid increasing attentions recently by researchers, fairness is a causal notion that concerns the causal connection between the sensitive attributes and the challenged decisions or outputs [28, 29, 30, 31, 32, 33]. Based on Pearl's causal modeling framework [34], a number of causal-based fairness notions and criteria have been proposed, including total effect [29], direct discrimination [28], indirect discrimination [28], and counterfactual fairness [31]. Each notion captures fairness in one particular situation from the causal perspective. Total effect treats all causal effects from the sensitive attribute to the decision as unfair. Direct and indirect discrimination, on the other hand, consider the situation where discrimination is transmitted through certain paths in the causal graph. Counterfactual fairness again considers a different situation where we focus on the fairness with respect to a particular individual or a subgroup of individuals instead of the whole population.

2.3 Differential Privacy and Fairness

Recent works study the connection between achieving privacy protection and fairness. Research in [19] proposed a notion of fairness that is a generalization of differential privacy.

Research in [70] developed a pattern sanitization method that achieves k-anonymity and fairness. Most recently, the position paper [71] argued for integrating recent research on fairness and non-discrimination to socio-technical systems that provide privacy protection. Later on, several works studied how to achieve within-model fairness (demographic parity [72, 73], equality of odds [74], equality of opportunity [75]) in addition to enforcing differential privacy in the private model. In addition to the within-model fairness, cross-model fairness also arises in differential privacy preserving machine learning models when we compare the accuracy loss incurred by private model between the majority group and the protected group. Recently, research in [76] shows that the reduction in accuracy incurred by deep private models disproportionately impacts underrepresented subgroups. The unfairness in this cross-model scenario is that the reduction in accuracy due to privacy protection is discriminatory against the protected group. Our work studies how to prevent disparate impact of the private model on model accuracy across different groups.

3 Preliminaries

3.1 Differential Privacy

Differential privacy guarantees the output of a query to be insensitive to any particular record's presence or value in the dataset. Let D be a sensitive dataset that contains n records with p attributes. Differential privacy guarantees the output of a query f be insensitive to one individual record in a dataset. We use D and D' to denote two neighboring datasets which differ in exactly one record ($|D - D'| = 1$).

Definition 1. Differential privacy [1]. A mechanism $\mathcal{M}$ satisfies ε-differential privacy, if for all neighboring datasets D and D' and all subsets Z of $\mathcal{M}$'s range:

$$P(\mathcal{M}(D) \in Z) \leq \exp(\varepsilon) \cdot P(\mathcal{M}(D') \in Z). \tag{3.1}$$

The parameter ϵ denotes the privacy budget, which controls the amount by which the distributions induced by D and D' may differ.

Definition 2. Global sensitivity [1]. Given a function f, the sensitivity $S_f(D)$ is defined as

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

The global sensitivity measures the maximum possible change in $q(D)$ when one record in the dataset changes.

Differential privacy has two useful properties: (a) Composability: let $\mathcal{M}_1$ be an ε_1-differentially private mechanism, and let $\mathcal{M}_2$ be an ε_2-differentially private mechanism. Then their combination is $\varepsilon_1 + \varepsilon_2$-differentially private. (b) Security under post-processing: any

transformation or query over a differentially private result $g(\mathcal{M}(D))$ is still differentially private, as long as it doesn't acquire additional access to input dataset D.

Many mechanisms have been proposed to enforce differential privacy.

3.1.1 Laplace Mechanism

Laplace Mechanism is a popular method to achieve differential privacy. It adds identical independent noise into each output value of $f(D)$. We use $S_f(D)$ to denote sensitivity of $f(D)$. It measures the maximum possible change in $f(D)$ when one tuple in the dataset changes.

Theorem 1. Laplace mechanism [1]. Given a dataset D and a query f, a mechanism $\mathcal{M}(D) = f(D) + \boldsymbol{\eta}$ satisfies ε-differential privacy, where $\boldsymbol{\eta}$ is a random vector drawn from $Lap(S_f(D)/\varepsilon)$.

3.1.2 Gaussian Mechanism

Alternatively, we can use Gaussian noise instead of Laplace noise to achieve differential privacy. The Gaussian mechanism with parameter σ adds Gaussian noise $N(0, \sigma^2)$ to each component of the model output.

Theorem 2. Gaussian mechanism [77]. Let $\epsilon \in [0, 1]$ be arbitrary. For $c^2 > 2\log(1.25/\delta)$, the Gaussian mechanism with parameter $\sigma > c\Delta_f/\epsilon$ satisfies (ϵ, δ)-differential privacy.

The parameter δ is a broken probability. Smaller values of ϵ and δ indicate stronger privacy guarantee.

3.1.3 Exponential Mechanism

The exponential mechanism is a differentially private method which selects one outcome from a set of potential outcomes based on some probability distribution. For a given training dataset D and privacy budget ε, the quality function induces a probability distribution over the output domain (usually class labels), from which the outcomes are exponentially chosen. It favors higher scoring classes, while guaranteeing ε-differential privacy.

Definition 3. L_1-sensitivity [36]. Let $q : D \to R$ be a quality function that scores each output class $r \in R$. The sensitivity of this function is defined as

$$S(q(D, r)) = \max_{D, D', r \in R} ||q(D', r) - q(D, r)||_1. \tag{3.3}$$

Theorem 3. Exponential mechanism [36]. Given a dataset D, the exponential mechanism lets $\mathcal{M}$ randomly select a potential outcome r based on the following probability, then the mechanism $\mathcal{M}^{\epsilon}_{q,S(q)}(D, R)$ is ε-differentially private:

$$P(r \in R \text{ is selected}) \propto \exp\left(\frac{\varepsilon q(D, r)}{2S(q)}\right). \tag{3.4}$$

The exponential mechanism works as follow: assume the goal is to output an output $r \in R$ from its known output domain R. There exists a quality function $q(D, r)$ that measures the quality of an output r, given that the dataset is D. The exponential mechanism $\mathcal{M}^{\epsilon}_{q,S(q)}(D, R)$ outputs $r \in R$ with probability proportional to $\exp\left(\frac{\varepsilon q(D, r)}{2S(q)}\right)$ and ensures ε-differential privacy. The mechanism assigns the highest probability to the best answer, and the probability assigned to any other drops off exponentially in the decline in its quality function.

3.1.4 Objective Perturbation

Chaudhuri et al. [3] proposed an objective perturbation approach by perturbing the objective function $\mathcal{L}$ and then optimizing the perturbed objective function,

$$\mathcal{L}_{priv}(\boldsymbol{\omega}, \mathcal{G}) = \mathcal{L}(\boldsymbol{\omega}, \mathcal{G}) + \boldsymbol{\omega}\boldsymbol{\eta}^T, \tag{3.5}$$

where $\boldsymbol{\eta}$ is a random noise vector and its probability density is given by

$$\Pr(\boldsymbol{\eta}) \propto e^{-\beta||\boldsymbol{\eta}||}, \tag{3.6}$$

and the parameter β is a function of privacy budget ϵ and the scale of $||\bigtriangledown_{\boldsymbol{\omega}}\mathcal{L}(\boldsymbol{\omega}, e_{ij})||$. To implement this, we pick the norm of $\boldsymbol{\eta}$ from the $\Gamma(k, \beta)$ distribution and the direction of $\boldsymbol{\eta}$ uniformly at random. Then we compute the private output $\widehat{\boldsymbol{\omega}}$, where $\widehat{\boldsymbol{\omega}} = \arg\min_{\boldsymbol{\omega}} \mathcal{L}_{priv}(\boldsymbol{\omega}, \mathcal{G})$ satisfies ϵ-differential privacy.

3.1.5 Functional Mechanism

Functional mechanism [4] achieves ε-differential privacy by injecting noise into the objective function of the model and returns privacy preserving parameter $\bar{\mathbf{w}}$ that minimizes the perturbed objective function.

Because the objective function $f_D(\mathbf{w})$ is a complicated function of $\mathbf{w}$, the functional mechanism exploits the polynomial representation of $f_D(\mathbf{w})$. The model parameter $\mathbf{w}$ is a vector that contains d values $w_1, w_2, \cdots, w_d$. Let $\phi(\mathbf{w})$ denote a product of $w_1, w_2, \cdots, w_d$, i.e., $\phi(\mathbf{w}) = w_1^{c_1} \cdot w_2^{c_2} \cdots w_d^{c_d}$ for some $c_1, c_2, \cdots, c_d \in \mathbb{N}$. Let $\boldsymbol{\Phi}_j$ $(j \in \mathbb{N})$ denote the set of all products of $w_1, w_2, \cdots, w_d$ with degree j, i.e., $\boldsymbol{\Phi}_j = \{w_1^{c_1} w_2^{c_2} \cdots w_d^{c_d} | \sum_{l=1}^{d} c_l = j\}$. For

example, $\mathbf{\Phi}_1 = \{w_1, w_2, \cdots, w_d\}$, and $\mathbf{\Phi}_2 = \{w_i \cdot w_j | i, j \in [1, d]\}$.

Based on the Stone-Weierstrass Theorem [78], any continuous and differentiable function can be expressed in the polynomial representation. Hence, the objective function $f_D(\mathbf{w})$ can be expressed as a polynomial of $w_1, w_2, \cdots, w_d$, for some $J \in \mathbb{N}$:

$$f_D(\mathbf{w}) = \sum_{i=1}^{n} \sum_{j=0}^{J} \sum_{\phi \in \mathbf{\Phi}_j} \lambda_{\phi t_i} \phi(\mathbf{w}), \tag{3.7}$$

where $\lambda_{\phi t_i} \in \mathbb{R}$ denotes the coefficient of $\phi(\mathbf{w})$ in the polynomial.

Functional mechanism perturbs the objective function $f_D(\mathbf{w})$ by injecting Laplace noise into its polynomial coefficients $\bar{\lambda}_\phi = \sum_{i=1}^{n} \lambda_{\phi t_i} + Lap(0, \frac{\Delta}{\varepsilon})$, where $\Delta = 2 \max_t \sum_{j=1}^{J} \sum_{\phi \in \mathbf{\Phi}_j} ||\lambda_{\phi t}||_1$. Then the model parameter $\bar{\mathbf{w}}$ is derived to minimize the perturbed function $\bar{f}_D(\mathbf{w})$.

Applying Functional Mechanism on Logistic Regression. A logistic regression on D returns a function which predicts $\hat{y}_i = 1$ with probability:

$$\hat{y}_i = q(\mathbf{x}_i; \mathbf{w}) = \exp(\mathbf{x}_i^T \mathbf{w}) / (1 + \exp(\mathbf{x}_i^T \mathbf{w})). \tag{3.8}$$

The objective function of logistic regression is defined as:

$$f_D(\mathbf{w}) = \sum_{i=1}^{n} \left[\log(1 + \exp(\mathbf{x}_i^T \mathbf{w})) - y_i \mathbf{x}_i^T \mathbf{w} \right]. \tag{3.9}$$

As the polynomial form of $f_D(\mathbf{w})$ in Equation 3.9 contains terms with unbounded degrees, to apply the functional mechanism, Equation 3.9 is rewritten as the approximate polynomial

representation based on Taylor expansion [4]:

$$f_D(\mathbf{w}) = \Big(\sum_{i=1}^{n} \sum_{j=0}^{2} \frac{f_1^{(j)}(0)}{j!} \left(\mathbf{x}_i^T \mathbf{w} \right)^j \Big) - \Big(\sum_{i=1}^{n} y_i \mathbf{x}_i^T \Big) \mathbf{w}, \tag{3.10}$$

where $f_1(\cdot) = \log(1 + \exp(\cdot))$.

When rewriting Equation 3.10 in the form of Equation 3.7, we have

$$\{\lambda_{\phi t_i}\}_{\phi \in \mathbf{\Phi}_1} =: \lambda_{1 t_i} = \big(\tfrac{f_1^{(1)}(0)}{1!} \mathbf{x}_i \big) - \big(y_i \mathbf{x}_i \big), \tag{3.11}$$

$$\{\lambda_{\phi t_i}\}_{\phi \in \mathbf{\Phi}_2} =: \lambda_{2 t_i} = \tfrac{f_1^{(2)}(0)}{2!} \left(\mathbf{x}_i \right)^2. \tag{3.12}$$

The global sensitivity of $f_D(\mathbf{w})$ is:

$$\begin{aligned} \Delta_f &= 2 \max_t \Big(\Big| \big(\frac{f_1^{(1)}(0)}{1!} - y \big) \sum_{l=1}^{d} x_{(l)} \Big| + \Big| \frac{f_1^{(2)}(0)}{2!} \sum_{l,m}^{d} x_{(l)} x_{(m)} \Big| \Big) \\ &\leq 2 (\frac{d}{2} + \frac{d^2}{8}) = \frac{d^2}{4} + d. \end{aligned} \tag{3.13}$$

Thus, to achieve ε-differential privacy, the functional mechanism adds $Lap(0, \frac{\Delta_f}{\varepsilon})$ noise to the polynomial coefficients of the objective function.

3.1.6 Differentially Private Stochastic Gradient Descent

The procedure of deep learning model training is to minimize the output of a loss function through numerous stochastic gradient descent (SGD) steps. [38] proposed a differentially private SGD algorithm (DPSGD). DPSGD uses a clipping bound on l_2 norm of individual updates, aggregates the clipped updates, and then adds Gaussian noise to the aggregate. This ensures that the iterates do not overfit to any individual user's update.

The privacy leakage of DPSGD is measured by (ϵ, δ), i.e., computing a bound for

Algorithm 1 DPSGD (Dataset D, loss function $\mathcal{L}_D(w)$, learning rate r, batch size b, noise scale σ, clipping bound C)

for $t \in [T]$ **do**
  Randomly sample a batch of samples B_t with $|B_t| = b$ from D
  for each sample $x_i \in B_t$ **do**
    $g_i = \nabla L_i(w_t)$
  end for
  for each sample $x_i \in B_t$ **do**
    $\bar{g}_i = g_i \times \min\left(1, \frac{C}{|g_i|}\right)$
  end for
  $\tilde{G}_B = \frac{1}{b}\left(\sum_i \bar{g}_i + N(0, \sigma^2 C^2 \mathbf{I})\right)$
  $\tilde{w}_{t+1} = \tilde{w}_t - r\tilde{G}_B$
end for
Return $\tilde{w}_T$ and accumulated privacy cost (ϵ, δ)

the privacy loss ϵ that holds with certain probability δ. Each iteration t of DPSGD can be considered as a privacy mechanism $\mathcal{M}_t$ that has the same pattern in terms of sensitive data access. [38] further proposed a moment accounting mechanism which calculates the aggregate privacy bound when performing SGD for multiple steps. The moments accountant computes tighter bounds for the privacy loss compared to the standard composition theorems. The moments accountant is tailored to the Gaussian mechanism and employs the log moment of each $\mathcal{M}_t$ to derive the bound of the total privacy loss. The log moment of privacy loss follows linear composability.

Theorem 4. Composability of moments [38]. For a given mechanism $\mathcal{M}$, the λ^{th} moment $\alpha_{\mathcal{M}}(\lambda) \triangleq \max_{aux,D,D'} \alpha_{\mathcal{M}}(\lambda; aux, D, D')$, where the maximum is taken over all possible auxiliary input aux and all neighboring datasets D, D'. Suppose that a mechanism $\mathcal{M}$ consists of a sequence of adaptive mechanisms $\mathcal{M}_1, \ldots, \mathcal{M}_p$, where $\mathcal{M}_i : \prod_{j=1}^{i-1} \mathcal{R}_j \times \mathcal{D} \to \mathcal{R}_i$. Then, for any λ

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

To reduce noise in private training of neural networks, DPSGD [38] truncates the gradient of a neural network to control the sensitivity of the sum of gradients. This is because the sensitivity of gradients and the scale of the noise would otherwise be unbounded. To fix this, a cap C on the maximum size of a user's contribution is adopted (Line 7 in Algorithm 1). This will bias our estimated sum but also reduce the amount of added noise, as the sensitivity of the sum is now C. One question is how to choose the truncation level for the gradient norm. If set too high, the noise level may be so great that any utility in the result is lost. If set too low, a large amount of gradients will be forced to clip. DPSGD simply suggests using the median of observed gradients. [54] investigated this bias-variance trade-off and showed that the limit we should choose is the $(1 - 1/b\epsilon)$-quantile of the gradients themselves. It does not matter how large or small the gradients are above or below the cutoff, only that a fixed number of values are clipped.

3.2 Fairness-aware Machine Learning

In fairness-aware learning, the literature has studied notions of group fairness on data and classification [16, 27].

3.2.1 Fairness in Data

Consider a labeled dataset $\mathcal{D}$, which contains a set of unprotected attributes $\mathbf{X} \in \mathbb{R}^n$, a class label $Y \in \{0, 1\}$ and a protected attribute $S \in \{0, 1\}$. Note that we consider S and Y as binary variables for ease of discussion.

Definition 4. Statistical fairness is a notion of data fairness which measures the potential discrimination caused by the correlation between the class label Y and the protected attribute S. The property of statistical fairness is defined as $P(Y = 1|S = 1) = P(Y = 1|S = 0)$.

Research in [14] proposed the concept of ϵ-**fairness** to examine the potential discrimination caused by the correlation between the unprotected attributes $\mathbf{X}$ and the protected attribute S.

Definition 5. A labeled dataset $\mathcal{D}$ is said to be ϵ-fair if for any classification algorithm $f : \mathbf{X} \to S$, $BER(f(\mathbf{X}), S) > \epsilon$ with empirical probabilities estimated from $\mathcal{D}$, where BER (balanced error rate) is defined as

$$BER(f(\mathbf{X}), S) = \frac{P(f(\mathbf{X})=0|S=1)+P(f(\mathbf{X})=1|S=0)}{2}. \tag{3.14}$$

BER indicates the average class-conditioned error of f on distribution $\mathcal{D}$ over the pair $(\mathbf{X}, S)$.

3.2.2 Fairness in Classification

Consider the classifier $\eta : \mathbf{X} \to Y$ which predicts the class label Y given the unprotected attributes $\mathbf{X}$. Classification fairness requires that the predicted label $\eta(\mathbf{X})$ is unbiased with respect to the protected variable S. The following notions of fairness in classification were defined by [16] and refined by [64].

Definition 6. Demographic parity. Given a labeled dataset $\mathcal{D}$ and a classifier $\eta : \mathbf{X} \to Y$, the property of demographic parity is defined as $P(\eta(\mathbf{X}) = 1|S = 1) = P(\eta(\mathbf{X}) = 1|S = 0)$.

This means that the predicted labels are independent of the protected attribute.

Definition 7. Equality of odds. Given a labeled dataset $\mathcal{D}$ and a classifier η, the property of equality of odds is defined as $P(\eta(\mathbf{X}) = 1|Y = y, S = 1) = P(\eta(\mathbf{X}) = 1|Y = y, S = 0)$, where $y \in \{0, 1\}$.

Hence, for $Y = 1$, equality of odds requires the classifier η has equal true positive rates (TPR) between two subgroups $S = 1$ and $S = 0$; for $Y = 0$, the classifier η has equal false positive rates (FPR) between two subgroups. Equality of odds promotes that individuals who qualify for a desirable outcome should have an equal chance of being correctly classified for this outcome. It allows for higher accuracy with respect to non-discrimination. It enforces both equal true positive rates and false positive rates in all demographics, punishing models that perform well only on the majority.

In many binary classification cases, $Y = 1$ is a more important outcome. With only requiring non-discrimination on the specific outcome group, the equality of odds can be relaxed to the equality of opportunity.

Definition 8. Equality of opportunity. Given a labeled dataset $\mathcal{D}$ and a classifier η, the property of equality of opportunity in a classifier is defined as $P(\eta(\mathbf{X}) = 1|Y = 1, S = 1) = P(\eta(\mathbf{X}) = 1|Y = 1, S = 0)$.

The equality of opportunity only focuses on the true positive rates.

Based on demographic parity, the discrimination of the model can be quantified by risk difference (RD):

$$RD = |\Pr(\hat{Y} = 1|S = 1) - \Pr(\hat{Y} = 1|S = 0)|. \tag{3.15}$$

To achieve classification fairness, the in-processing approaches are to find parameter $\mathbf{w}$ that minimizes the objective function under a fairness constraint:

$$\begin{aligned} \text{minimize} \quad & f_D(\mathbf{w}) \\ \text{subject to} \quad & g_D(\mathbf{w}) \leq \tau,\ g_D(\mathbf{w}) \geq -\tau, \end{aligned} \tag{3.16}$$

where $g_D(\mathbf{w})$ is the constraint term; $\tau \in \mathbf{R}^+$ is the threshold of constraint. For example, in [18], the fairness constraint is defined as the covariance between the users' protected attribute and the signed distance from the users' unprotected attribute vectors to the decision boundary $\{d_{\mathbf{w}}(\mathbf{x}_i)\}_{i=1}^n$,

$$g_D(\mathbf{w}) = \mathbb{E}[(s-\bar{s})d_{\mathbf{w}}(\mathbf{x})] - \mathbb{E}[(s-\bar{s})]d_{\mathbf{w}}(\mathbf{x}) \propto \sum_{i=1}^{n}(s_i-\bar{s})d_{\mathbf{w}}(\mathbf{x}_i), \tag{3.17}$$

where $\bar{s}$ is the mean value of the protected attribute; $\mathbb{E}[(s-\bar{s})] = 0$. For linear classification models, like logistic regression or linear SVMs, the decision boundary is simply the hyperplane defined by $\mathbf{x}^T\mathbf{w} = 0$. Then, Equation 3.17 reduces to $g_D(\mathbf{w}) = \sum_{i=1}^{n}(s_i-\bar{s})\mathbf{x}_i^T\mathbf{w}$. The decision boundary fairness is proven to be a notion of fairness that minimizes the surrogate risk difference [79].

3.2.3 Causation-based Fairness

Recent researches [28, 29, 30, 31, 32, 33] showed that fairness should be studied from the causal perspective, and proposed a number of fairness criteria based on Pearl's causal modeling framework [34].

3.2.3.1 Causal Model and Intervention

Definition 9. A causal model [34] is a triple $\mathcal{M} = \{\mathbf{U}, \mathbf{V}, \mathbf{F}\}$ where

1) $\mathbf{U}$ is a set of hidden random variables that are determined by factors outside the model. A joint probability distribution $P(\mathbf{U})$ is defined over the variables in $\mathbf{U}$.

2) $\mathbf{V}$ is a set of observed random variables that are determined by variables in $\mathbf{U} \cup \mathbf{V}$.

3) $\mathbf{F}$ is a set of deterministic functions; for each $V_i \in \mathbf{V}$, a corresponding function f_{V_i} is a

mapping from $\mathbf{U} \cup (\mathbf{V} \setminus \{V_i\})$ to V_i, i.e., $V_i = f_{V_i}(Pa_{V_i}, \mathbf{U}_{V_i})$, where $Pa_{V_i} \subseteq \mathbf{V} \backslash \{V_i\}$ is called the parents of V_i, and $\mathbf{U}_{V_i} \subseteq \mathbf{U}$.

A causal model is often illustrated by a causal graph $\mathcal{G}$ [34], where each observed variable is represented by a node, and the causal relationships are represented by directed edges $\rightarrow$. In this graphical representation, the definition of parents is consistent with that in the causal model. In addition, each node V_i is associated with a conditional distribution given all its parents, i.e., $P(V_i|Pa_{V_i})$.

Inferring causal effects in the causal model is facilitated by *do*-operator [34], which simulates the physical intervention that forces some variable $X \in \mathbf{V}$ to take certain value x. For a causal model $\mathcal{M}$, intervention $do(X = x)$ is performed by replacing original function $X = f_X(Pa_X, \mathbf{U}_X)$ with $X = x$. After replacing, the distributions of all variables that are the descendants of X may be changed. We call the causal model after the intervention the interventional model, denoted by $\mathcal{M}_x$. Correspondingly, $\mathcal{M}_x$ can be illustrated by the interventional graph $\mathcal{G}_x$ where all incoming edges to X are deleted and node X is replaced with constant x. The interventional distribution for any $\mathbf{Y} \subseteq \mathbf{V} \setminus \{X\}$ is denoted by $P(\mathbf{Y}|do(X = x))$ or $P(\mathbf{Y}_x)$. Symbolically, $P(\mathbf{Y}_x)$ can be expressed as a truncated factorization formula [34] and computed from the observed distribution.

3.2.3.2 Causal Effects

With the help of *do*-operator, we can infer the causal effect of X on Y by comparing the difference in interventional distributions under different interventions. Based on how the intervention is transferred in the causal model (graph), there are mainly three types of causal effects: total effect, path-specific effect, counterfactual effect [34].

The total effect measures the causal effect of X on Y where the intervention is trans-

ferred along all causal paths (i.e., directed paths) from X to Y.

Definition 10. The total effect of the value change of X from x_1 to x_2 on Y is given by $TE(x_2, x_1) = P(Y_{x_2}) - P(Y_{x_1})$.

The path-specific effect measures the causal effect of X on Y where the intervention is transferred only along a subset of causal paths from X to Y, which is also referred to as the π-specific effect denoting the subset of causal paths as π.

Definition 11. Given a path set π, the π-specific effect of the value change of X from x_1 to x_2 on Y (with reference x_1) is given by $SE_\pi(x_2, x_1) = P(Y_{x_2|_\pi}) - P(Y_{x_1|_\pi})$, where $P(Y_{x|_\pi})$ represents the interventional distribution where the intervention is transferred only along π.

In the total effect and path-specific effect, the intervention is performed on the whole population. The counterfactual effect measures the causal effect while the intervention is performed conditioning on only certain individuals or groups specified by a subset of observed variables $\mathbf{O} = \mathbf{o}$.

Definition 12. Given a context $\mathbf{O} = \mathbf{o}$, the counterfactual effect of the value change of X from x_1 to x_2 on Y is given by $CE(x_2, x_1|\mathbf{o}) = P(Y_{x_2}|\mathbf{o}) - P(Y_{x_1}|\mathbf{o})$.

3.3 Generative Adversarial Networks

Generative adversarial networks (GAN) have been shown to able to generate high quality synthetic data that are similar to real data [26, 80]. A GAN model consists of two components: a generator G and a discriminator D. Typically, both G and D are multilayer neural networks. $G(\mathbf{z})$ generates fake samples from a prior distribution $P_\mathbf{z}$ on a noise variable $\mathbf{z}$ and learns a generative distribution P_G to match the real data distribution P_{data}. The

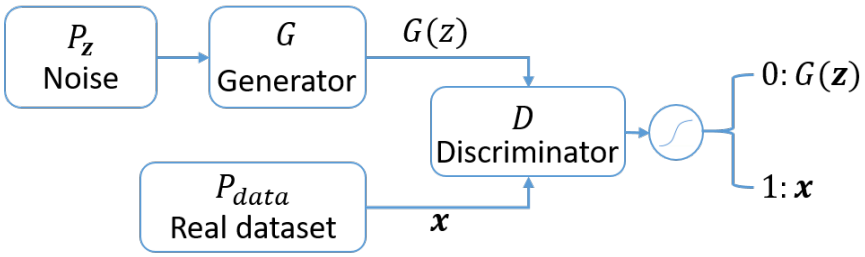


Figure 3.1: Illustration of generative adversarial networks

discriminative component D is a binary classifier that predicts whether an input is real data $\mathbf{x}$ or fake data generated from $G(\mathbf{z})$. Figure 3.1 illustrates the structure of GAN.

The objective function of D is defined as:

$$\max_D \quad \mathbb{E}_{\mathbf{x}\sim P_{\text{data}}}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim P_{\mathbf{z}}}[\log(1 - D(G(\mathbf{z})))], \tag{3.18}$$

where $D(\cdot)$ outputs the probability that $\cdot$ is from the real data rather than the generated fake data. In order to make the generative distribution P_G close to the real data distribution P_{data}, G is trained by fooling the discriminator unable to distinguish the generated data from the real data. Thus, the objective function of G is defined as:

$$\min_G \quad \mathbb{E}_{\mathbf{z}\sim P_{\mathbf{z}}}[\log(1 - D(G(\mathbf{z})))]. \tag{3.19}$$

Minimization of Equation 3.19 ensures that the discriminator is fooled by $G(\mathbf{z})$ and D predicts high probability that $G(\mathbf{z})$ is real data.

Overall, GAN is formalized as a minimax game $\min_G \max_D V(G, D)$ with the value function:

$$V(G, D) = \mathbb{E}_{\mathbf{x}\sim P_{\text{data}}}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim P_{\mathbf{z}}}[\log(1 - D(G(\mathbf{z})))]. \tag{3.20}$$

GAN for discrete data generation. The generator of a regular GAN cannot

generate discrete samples [26]. In order to tackle this limitation, medGAN incorporates an autoencoder in a regular GAN model to generate high-dimensional discrete variables [81]. A basic autoencoder consists of an encoder Enc and a decoder Dec. The objective function of the autoencoder is to make the reconstructed input $\mathbf{x}'$ close to the original input $\mathbf{x}$:

$$\mathcal{L}_{AE} = ||\mathbf{x}' - \mathbf{x}||_2^2, \tag{3.21}$$

where $\mathbf{x}' = Dec(Enc(\mathbf{x}))$.

To generate the dataset which contains discrete attributes, the generator G_{Dec} in medGAN consists of two components, the generator G and the decoder Dec. The generator G is trained to generate the salient representations. The decoder Dec from autoencoder seeks to construct the synthetic data from the salient representations $Dec(G(\mathbf{z}))$. Hence, the generator of medGAN $G_{Dec}(\mathbf{z})$ is defined as: $G_{Dec}(\mathbf{z}) = Dec(G(\mathbf{z}))$, where $\mathbf{z}$ is a noise variable. The discriminator D aims to distinguish whether the input is from real data or $Dec(G(\mathbf{z}))$. The generator G_{Dec} can be viewed as a regular generator G with extra hidden layers that maps continuous salient representations to discrete samples.

Auxiliary classifier GAN. Research in [82] proposed a variant of the GAN structure, called auxiliary classifier generative adversarial networks (ACGAN). Each generated sample has a corresponding class label y in addition to the noise $\mathbf{z}$. A classifier is incorporated into the model to reconstruct the class label from the data samples. The generator $G(y, \mathbf{z})$ can produce class conditional samples that match both the real data distribution and the class conditions.

CausalGAN. Research in [83] shows that GAN can be modified to generate both observational and interventional distributions while preserving the causal structure among

all attributes, referred to as the CausalGAN. Given a causal graph, generator $G(\mathbf{Z})$ attempts to play the role of a causal model that agrees with this causal graph in terms of both the graph structure and conditional distributions. To this end, noises $\mathbf{Z}$ are partitioned into $|\mathbf{V}|$ subsets $\{\mathbf{Z}_{V_1}, \mathbf{Z}_{V_2}, \ldots\}$, each of which $\mathbf{Z}_{V_i}$ plays the role of hidden variables $\mathbf{U}_{V_i}$. Similarly, generator $G(\mathbf{Z})$ is partitioned into $|\mathbf{V}|$ sub-neural networks $\{G_{V_1}, G_{V_2}, \ldots\}$, each of which G_{V_i} plays the role of function f_{V_i} for generating the values of V_i. Then, if node V_j is a parent of V_i in the causal graph, the output of G_{V_j} is designed as an input of G_{V_i} to reflect this connection. Meanwhile, the adversarial game is played to ensure $P_G(G(\mathbf{Z}) = \mathbf{v}) = P(\mathbf{V} = \mathbf{v}), \forall \mathbf{v}$. The authors have proved that $G(\mathbf{Z})$ is consistent with any causal model that agrees with the same causal graph in terms of any identifiable interventional distributions, if: (1) $P(\mathbf{V})$ is strictly positive; (2) the connections of sub-neural networks G_{V_i} are arranged to reflect the causal graph structure; and (3) the generated observational distribution matches the real observational distribution, i.e., $P_G(G(\mathbf{Z}) = \mathbf{v}) = P(\mathbf{V} = \mathbf{v}), \forall \mathbf{v}$. Therefore, CausalGAN can be used to simulate the real causal model that agrees with the causal graph in identifiable situations.

4 Differentially Private Causal Graph Discovery

4.1 Introduction

Understanding the cause-effect relationships helps people make decisions in a rational manner. Traditionally, experimental intervention is conducted to identify causal effect. However, it's often difficult, if not infeasible to set up experiments for high-dimensional datasets. Causal graphs are widely used to discover the causal relationships among multiple random variables [84, 85].

A causal graph is a probabilistic graphical model, which conveniently captures the causal structure among random variables. It is a computationally difficult task to estimate a causal graph from data, as the number of possible causal graphs increases super-exponentially to the number of nodes. Constraint-based methods [35, 34] are shown accurate and efficient in reconstructing causal patterns in many applications. The PC algorithm [35] is one of the commonly-used constraint-based methods for causal graph discovery. Under Causal Markov and faithfulness assumptions, the PC algorithm is demonstrated to obtain accurate causal information with only observational data [35]. The procedure of the PC algorithm is as follows. It first forms a complete, undirected graph and then recursively deletes edges according to conditional independence tests. The PC algorithm is computationally feasible and consistent even for sparse causal graphs. It achieves consistently good and causal graph faithfulness (i.e., that the data can be assumed to be simulated from a probability distribution that factorizes according to a causal graph).

Causal graphs can provide extensive information for individuals and policy makers. However, when causal graph discovery is performed on a large amount of sensitive individual

data, it raises individual participants' concerns about their privacy, e.g. in the fields of medical or financial analysis. This is because releasing aggregated findings may risk leakage of individual private information. For instance, Homer [86] demonstrated that attackers can use aggregated genetic data to infer whether an individual had participated in a genome study. One solution to protect individual privacy against leakage risk is differential privacy.

It has not yet been thoroughly studied how to release causal graph information without risking private information leakage. Currently, there is no differentially private PC algorithm for causal graph discovery. It is essential to infer causal relationships with privacy guarantee. One of the challenges is that the number of conditional independence tests among nodes is tremendously large. The complexity of PC algorithm increases dramatically with the dimension and sparseness of a causal graph. It has a very loose upper bound. One naive method to achieve differential privacy in the PC algorithm is to add protection at each conditional independence test. However, it will require a very large privacy budget. How to reduce the number of tests or the privacy budget is an essential question for a reliable private PC algorithm. In this work, we explore how to guarantee individual privacy in the PC algorithm for causal graph discovery.

We summarize all important notations in Table 4.1.

4.2 Related Work

Causal graph discovery A causal graph is a probabilistic graphical model which is widely used for causality representation, reasoning and inference [34]. The PC algorithm [35] is a widely used algorithm for causal graph discovery. The original PC algorithm [35] is known to be order-dependent, in the sense that the output may depend on the input order of the variables. There are several modified PC algorithms for causal graph discovery.

Table 4.1: Notations

Symbol	Definition
$Adj(\mathcal{G}, V_i)$	the adjacent set of V_i
$Ind(\mathcal{G}, V_i)$	the conditionally independent adjacent set of V_i
$T(ij\|\mathbf{k})$	the conditional independence test for V_i, V_j given $\mathbf{V_k}$
$d(ij\|\mathbf{k})$	a distance score of $T(ij\|\mathbf{k})$
π, π_t	a permutation, the t-th largest element in π
$\|\bullet\|$	the size of $\bullet$
$\hat{\bullet}$	a differentially private $\bullet$
$\mathcal{M}^{\epsilon}_{q,S(q)}(D, R)$	an exponential mechanism with a privacy budget ε, a quality function q and a sensitivity $S(q)$ on an input dataset D and a output domain R

Colombo and Maathuis [87] proposed a simple modification, called "PCstable", which yields order-independent adjacencies in the skeleton. Ramsey et al. [88] proposed a conservative PC algorithm which has a slight variation in computing all potential v-structures of the PC algorithm. Kalisch and Buhlmann [89] showed high-dimensional consistency of the PC algorithm which uses partial correlations derived from independent Gaussian observations that are faithful to a suitably sparse causal graph. It was later proved that consistency of Gaussian model can be carried over to a boarder class of Gaussian copula or nonparanormal models which uses rank-based correlation instead of Pearson correlation [90].

4.3 Preliminaries

In this section, we present the technical preliminaries. We introduce properties of the causal graph and details of the original PC algorithm. For general preliminaries of differential privacy, please refer to Chapter 3.1.

Causal Graph Discovery

4.3.1 Causal Graph

A causal graph is a probabilistic graphical model which is specified by a directed acyclic graph (DAG) $\mathcal{G} = (\mathbf{V}, \mathbf{E})$. $\mathbf{V} = \{V_1, \ldots, V_p\}$ is a set of nodes, each of which corresponds to a variable. $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}$ is a set of edges, which is a subset of ordered pairs of distinct nodes. We use $V_i \to V_j$ to denote a directed edge. $Adj(\mathcal{G}, V_i)$ denotes the adjacency set of a node V_i on graph $\mathcal{G}$, also called neighbors of V_i.

We use P to denote a probability distribution that is faithful with respect to the graph $\mathcal{G}$. Consider the set of random variables $\mathbf{V} \sim P$, faithfulness of P with respect to $\mathcal{G}$ means: if for any $V_i, V_j \in \mathbf{V}(i \neq j)$ and any set $\mathbf{V_k} \subseteq \mathbf{V}$, V_i and V_j are conditionally independent given $\mathbf{V_k}$, nodes V_i and V_j are d-separated by set $\mathbf{V_k}$ [35].

Sometimes two causal graphs can fit the same distribution. These two graphs are called Markov equivalent. Verma and Pearl [91] demonstrated that two causal graphs are equivalent if and only if they have the same skeleton and the same v-structures. The skeleton of a causal graph $\mathcal{G}$ is an undirected graph with undirected edges instead of directed edges. A v-structure is an ordered set of nodes (V_i, V_j, V_k) such that $\mathcal{G}$ contains directed edges $V_i \to V_j$ and $V_k \to V_j$, and V_i and V_k are not adjacent in $\mathcal{G}$. All causal graph that are Markov equivalent belong to the same Markov equivalence class. A Markov equivalence class can be uniquely represented by a completed, partially directed acyclic graph (CPDAG) [92]. The main goal of estimating a causal graph is to identify the CPDAG, in which it contains conditional independence information of node sets.

4.3.2 PC Algorithm

The PC algorithm is a constraint-based method for causal graph discovery [35]. It assumes that the probability distribution P is Markov and faithful with respect to the correct

graph $\mathcal{G}$, which means all conditional independences in the joint distribution are entailed by the Markov condition. PC algorithm first estimates the skeleton of the underlying CPDAG, and then determines the orientation of as many edges as possible.

Algorithm 2 PC algorithm (Vertex set $\mathbf{V}$, Dataset D)

1: Form the complete undirected graph $\mathcal{C}_0$ on vertex set $\mathbf{V}$
2: $ord = 0$; $\mathcal{C} = \mathcal{C}_0$
3: **for** all V_i $|Adj(\mathcal{C}, V_i) \setminus \{V_j\}| < ord$ **do**
4: **for** all $V_i - V_j$ with $|\mathbf{V_k}| = ord$ have been tested for conditional independence **do**
5: Select an edge $V_i - V_j$ that are adjacent in $\mathcal{C}$ s.t. $|Adj(\mathcal{C}, V_i) \setminus \{V_j\}| \geq ord$
6: **for** all $\mathbf{V_k}$ have been tested **do**
7: Choose $\mathbf{V_k} \subseteq Adj(\mathcal{C}, V_i) \setminus \{V_j\}$ with $|\mathbf{V_k}| = ord$
8: **if** V_i, V_j are conditionally independent given $\mathbf{V_k}$ based on $T(ij|\mathbf{k})$ **then**
9: Delete $V_i - V_j$ from $\mathcal{C}$
10: break
11: **end if**
12: **end for**
13: **end for**
14: $ord = ord + 1$
15: **end for**
16: Create a partially directed graph with minimal pattern (Identify unshielded colliders)
17: **for** each pair of non adjacent variables V_i, V_j with a common neighbor V_k **do**
18: **if** V_i, V_j are not conditionally independent given V_k **then**
19: Orient $V_i - V_k - V_j$ as $V_i \rightarrow V_k \leftarrow V_j$
20: **end if**
21: **end for**
22: Convert to complete pattern: use rules by Pearl [34]
23: **for** no more edges can be oriented **do**
24: Rule 1: $V_i \rightarrow V_j - V_k$ goes to $V_i \rightarrow V_j \rightarrow V_k$ (No new collider is introduced)
25: Rule 2: $V_i \rightarrow V_k \rightarrow V_j$ with $V_i - V_j$, then $V_i - V_j$ goes to $V_i \rightarrow V_j$ (Avoid cycle)
26: Rule 3: $V_i - V_j$, $V_i - V_k$, $V_i - V_l$, $V_k \rightarrow V_j$, $V_l \rightarrow V_j$ but V_k and V_l are not connected; then $V_i - V_j$ goes to $V_i \rightarrow V_j$
27: **end for**
28: Return Estimated CPDAG $\mathcal{G}$

a) Estimate the skeleton given data: The first phase of PC algorithm is given in Algorithm 2 (Lines 1−15). Sprites et al [35] proved that: Given a causal graph $\mathcal{G}$ and a distribution P that is faithful to $\mathcal{G}$, this algorithm can construct the true skeleton $\mathcal{C}$ of the causal graph. In the first phase, the skeleton $\mathcal{C}$ is an undirected graph. The edge between

node V_i and V_j is denoted as $V_i - V_j$.

The complexity of the algorithm is bounded by the maximal reached value of *ord*. Note that *ord* is the condition order of conditional independence test, which is the number of vertices in conditioned set. Let l be the maximal degree of order and let p be the number of vertices. Then in the worst case the number of conditional independence tests needed in the algorithm is bounded by

$$2\binom{p}{2}\sum_{ord=0}^{l}\binom{p-1}{ord},$$

which has a maximal value of

$$\frac{p^2(p-1)^{l-1}}{(l-1)!}.$$

The computational requirements increase exponentially with l [35]. PC algorithm has high computational complexity. Each conditional independence test queries input data. Yet, the number of tests is hard to predict and it has a very loose bound. All of these make it challenging to achieve differential privacy in PC algorithm.

b) Orient the edges given the skeleton structure: The second phase of PC algorithm is given in Algorithm 2 (Lines 16−28), which extends skeleton to a CPDAG belonging to the equivalence class of the underlying true causal graph [34].

Note that the first phase accesses input data for conditional independence tests. The second phase doesn't need any information from input data. We only need to preserve privacy in the first phase. We'll discuss it in Section IV and V.

An important part of PC algorithm is to conduct conditional independence tests on dataset D. To estimate if V_i and V_j are conditionally independent given $\mathbf{V_k}$, we use $T(ij|\mathbf{k})$ to denote its corresponding conditional independence test, which outputs a p-value.

For categorical data, all nodes are random categorical variables. We use Cat_i to de-

note the number of categories of variable V_i. A conditional independence test $T(ij|\mathbf{k})$ can commonly be inferred from a corresponding contingency table by two statistics: χ^2 and G^2. The size of a contingency table for the corresponding conditional independence test $T(ij|\mathbf{k})$ is $Cat_i \times Cat_j \times \prod_{h \in \mathbf{k}} Cat_h$. Each cell of the contingency table contains the count number of corresponding value combination of $V_i, V_j, \mathbf{V_k}$ (denoted as $N_{ij|\mathbf{k}}$). Using frequency counts in corresponding contingency table, we calculate χ^2 statistic and G^2 statistic by the following formula:

$$\chi^2 = \sum \frac{(O-E)^2}{E}$$

$$G^2 = 2\sum (O) ln(\frac{O}{E})$$

where O is the observed value of $N_{ij|\mathbf{k}}$ of each cell, E is the expected value of $N_{ij|\mathbf{k}}$ of each cell. The degree of freedom (df) in the test is

$$df = (Cat_i - 1) \times (Cat_j - 1) \times \prod_{h \in \mathbf{k}} Cat_h$$

The size of a contingency table can be very large for large Cat or ord. It increases exponentially with ord and polynomially with Cat. For example, a basic contingency table for all variables has a tremendous size of $\prod_{j=1}^{p} Cat_j$.

In this work we adopt G^2 on corresponding contingency tables for the conditional independence tests in our experiments.

4.4 Differentially Private Causal Graph Discovery Algorithm for Categorical Data

This section describes our approach towards the differentially private causal graph discovery based on exponential mechanism: a differentially private PC (PrivPC) algorithm.

4.4.1 Differentially Private PC (PrivPC) Algorithm

Only the first phase of the PC algorithm engages with input data to estimate the skeleton $\mathcal{C}$ of the causal graph and may breach participants' privacy. More specifically, input data is queried at each conditionally independence test (Line 8 in Algorithm 2) for the contingency table. Thus, we need to protect privacy in the first phase of the PC algorithm. The naive method attempts to guarantee differential privacy for the corresponding contingency table at each conditional independence test between pairs of nodes. However, due to the high complexity of the PC algorithm, this method requires a tremendous amount of the privacy budget ε.

Our motivation for a reliable private PC algorithm is to consume a smaller privacy budget by reducing the number of conditional independence tests (or edge elimination decisions) based on the exponential mechanism. By adopting the exponential mechanism, we can evaluate conditional independence tests on multiple edges simultaneously in one step rather than waste privacy budget on evaluating the test on each edge. We propose a private conditionally independent adjacent set selection algorithm to reduce the number of edge elimination decisions by modifying the original PC algorithm. It makes the decision on edge elimination at each round of the conditionally independent adjacent set selection instead of making the decision at each conditional independence test.

Algorithm 3 PrivPC algorithm (Vertex set $\mathbf{V}$, Dataset D, Privacy budget ε)

1: Form the complete undirected graph $\mathcal{C}_0$ on vertex set $\mathbf{V}$
2: $ord = 0$; $\mathcal{C} = \mathcal{C}_0$
3: **for** $|Adj(\mathcal{C}, V_i)| < ord + 1$ for all V_i, or all privacy budget ε_0 is used up **do**
4: **for** all nodes V_i with $|Adj(\mathcal{C}, V_i)| \geq ord + 1$ have been selected **do**
5: Select a node V_i s.t. $|Adj(\mathcal{C}, V_i)| \geq ord + 1$
6: Privately find the conditionally independent adjacent set $\widehat{Ind}(\mathcal{C}, V_i)$ $findPrivInd\Big(V_i, Adj(\mathcal{C}, V_i), ord, D, \epsilon_1, \epsilon_2\Big)$
7: **for** all $V_j \in \widehat{Ind}(\mathcal{C}, V_i)$ **do**
8: Delete $V_i - V_j$ from $\mathcal{C}$
9: **end for**
10: **end for**
11: $ord = ord + 1$
12: **end for**
13: Run Algorithm 2 from Line 16 to Line 28 to orient the edges from a differentially private skeleton structure

Algorithm 3 shows how to estimate CPDAG with respect to differential privacy using the exponential mechanism. Different from the original PC algorithm which makes decision on the edge deletion for each edge $V_i - V_j$ after every conditional independence test, we propose a *conditionally independent adjacent set selection procedure based on the exponential mechanism* (Line 6). Conditionally independent adjacent set selection can make the exact same edge elimination decision as original PC algorithm, so it's a viable alternative PC algorithm.

In Algorithm 3, we first define an order ord of conditional independence test (Line 5). Then, we select a node V_i and find the private subset $\widehat{Ind}(\mathcal{C}, V_i)$ of the adjacent nodes $Adj(\mathcal{C}, V_i)$ by applying the exponential mechanism. The subset $\widehat{Ind}(\mathcal{C}, V_i)$ is conditionally independent to node V_i given $\mathbf{V_k}$ with $|\mathbf{V_k}| = ord$ (Line 6). Once we get the conditionally independent adjacent set $\widehat{Ind}(\mathcal{C}, V_i)$, we delete $V_i - V_j$ for each node $V_j \in \widehat{Ind}(\mathcal{C}, V_i)$ (Line 7-8). Instead of using part of the privacy budget on each conditional independence test (the naive method), Algorithm 3 splits $(\epsilon_1 + \epsilon_2)$ of the total budget ε onto each round of the

conditionally independent adjacent set selection procedure.

The first phase of the PrivPC algorithm stops if all privacy budget ε is used up. If early terminated, only high-order conditional independence tests are compromised. The early termination introduces acceptable errors, because the high-order conditional independence is rare in many cases.

4.4.2 Differentially Private Conditionally Independent Adjacent Set Selection Procedure: Details

Algorithm 4 Privately find the conditionally independent adjacent set of V_i $findPrivInd(V_i, Adj(\mathcal{C}, V_i), ord,$ Dataset $D, \epsilon_1, \epsilon_2)$

$R_1 = Adj(\mathcal{C}, V_i)$
for each $V_j \in Adj(\mathcal{C}, V_i)$ **do**
    Find $\mathbf{V}_{\mathbf{k}_{ij}} \subseteq Adj(\mathcal{C}, V_i) \setminus \{V_j\}$ with $|\mathbf{V}_{\mathbf{k}_{ij}}| = ord$ which has the maximal $T(ij|\mathbf{k}_{ij})$
    Calculate $d(ij|\mathbf{k}_{ij})$ by Equation 4.1
    $q_1(V_j, D) = d(ij|\mathbf{k}_{ij})$
end for
$R_2 = \{0, 1, \ldots, |Adj(\mathcal{C}, V_i)|\}$
Calculate q_2 for $0, 1, \ldots, |Adj(\mathcal{C}, V_i)|$ by Equations 4.2,4.3,4.4
$\hat{\beta} = \mathcal{M}^{\epsilon_2}_{q_2,1}(D, R_2)$
$\widehat{Ind}(\mathcal{C}, V_i) = \emptyset$
for $t = 1, \ldots, \hat{\beta}$ **do**
    $V_r = \mathcal{M}^{\epsilon_1/\hat{\beta}}_{q_1,1}(D, R_1)$
    $\widehat{Ind}(\mathcal{C}, V_i) = \widehat{Ind}(\mathcal{C}, V_i) \bigcup \{V_r\}$
    $R_1 = R_1 \setminus \{V_r\}$
end for
Return $\widehat{Ind}(\mathcal{C}, V_i)$

To achieve differential privacy in the conditionally independent adjacent set selection, we use the exponential mechanism to privately select nodes from the adjacent set $Adj(\mathcal{C}, V_i)$ into the private conditionally independent adjacent set $\widehat{Ind}(\mathcal{C}, V_i)$ (given in Algorithm 4). At each node selection, we privately select one node from the adjacent set $Adj(\mathcal{C}, V_i)$ until all conditionally independent nodes are selected into $\widehat{Ind}(\mathcal{C}, V_i)$. The output domain of the

node selection is all the nodes in $Adj(\mathcal{C}, V_i)$. We first find a distance score $d(ij|\mathbf{k}_{ij})$ for all V_j, which is used to define the quality functions. Then we use a quality function q_1 to score each node in $Adj(\mathcal{C}, V_i)$ and use the exponential mechanism to select $\widehat{Ind}(\mathcal{C}, V_i)$ according to q_1. However, the iteration number of the node selection β may also risk leakage of participants' privacy because it reveals the number of conditionally independent adjacent nodes. Thus, we also need to privately determine $\hat{\beta}$. The domain of $\hat{\beta}$ is from 0 to $|Adj(\mathcal{C}, V_i)|$. We use the exponential mechanism to determine $\hat{\beta}$ according to the quality function q_2. Thus, all decisions made in the conditionally independent adjacent set selection procedure ensure differential privacy.

Distance score: We need to define a quality function $q : D \to R$ that assigns a value to each potential output. However, the conditional independence test p-value has high sensitivity, making it challenging to produce accurate yet private answers via the exponential mechanism. We first find $\mathbf{V}_{\mathbf{k}_{ij}} \subseteq Adj(\mathcal{C}, V_i) \setminus \{V_j\}$ with $|\mathbf{V}_{\mathbf{k}_{ij}}| = ord$ that has the maximal $T(ij|\mathbf{k}_{ij})$; then we calculate a distance score $d(ij|\mathbf{k}_{ij})$ [47] as follows:

$$d(ij|\mathbf{k}_{ij}) = \begin{cases} \min\{d = |D - D'|\} \Rightarrow T(ij|\mathbf{k}_{ij})\Big|D' \geq \tau \\ \quad \text{if } T(ij|\mathbf{k})\Big|D \leq \tau, \\ -\min\{d = |D - D'|\} \Rightarrow T(ij|\mathbf{k}_{ij})\Big|D' < \tau \\ \quad \text{otherwise,} \end{cases} \tag{4.1}$$

where D' denotes a d-distance neighboring dataset that has d records different from D.

$d(ij|\mathbf{k}_{ij})$ is the minimum number of records in D that must be changed s.t. $T(ij|\mathbf{k}_{ij})$ (which returns a p-value) is above the significance threshold τ. The sign of $d(ij|\mathbf{k}_{ij})$ indicates the significance of independence test for edge $V_i - V_j$. The value of $d(ij|\mathbf{k}_{ij})$ has a good approximation to p-value in terms of the conditional independence information, but it has

a small sensitivity. Hence, we can define the quality functions based on the distance score.

For example, a conditional independence test $T(ij|\mathbf{k}_{ij})$ on dataset D returns a below-threshold p-value (a significant conditional independence); $d(ij|\mathbf{k}_{ij})$ finds the minimum d-distance neighboring dataset D' s.t. $T(ij|\mathbf{k}_{ij})$ on D' returns an above-threshold p-value, and gives it a positive sign to indicate that $T(ij|\mathbf{k}_{ij})$ on D is below the significance threshold.

Privately select a node into the conditionally independent adjacent set: We use $\mathcal{M}_{q_1,1}^{\epsilon_1/\hat{\beta}}(D, R_1)$ (Line 13 of Algorithm 4) to select a node into $\widehat{Ind}(\mathcal{C}, V_i)$. The output domain R_1 is all nodes in $Adj(\mathcal{C}, V_i)$; the quality function q_1 for each node $V_j \in Adj(\mathcal{C}, V_i)$ is the distance score $d(ij|\mathbf{k}_{ij})$. The node V_j with a higher corresponding distance score has a higher probability to be selected by the exponential mechanism.

Corollary 1. The private node selection achieves ϵ_1-differential privacy by applying $\mathcal{M}_{q_1,1}^{\epsilon_1/\hat{\beta}}(D, R_1)$. The L_1-sensitivity $S(q_1(D, r))$ of quality function q_1 is 1.

Proof: as one record can only change the distance score (Equation 4.1) by 1, the sensitivity of q_1 is at most 1.

Then we iterate the private node selection for $\hat{\beta}$ times from the remaining nodes in R_1.

In each round of the conditionally independent adjacent set selection, $\mathcal{M}_{q_1,1}^{\epsilon_1/\hat{\beta}}(D, R_1)$ uses ϵ_1 out of total privacy budget ε_0. Each iteration uses $\epsilon_1/\hat{\beta}$ to select one node into the conditionally independent adjacent set.

Privately select the iteration number of the node selection: Because the iteration number of the node selection need to be privately produced, we use the exponential mechanism $\mathcal{M}_{q_2,1}^{\epsilon}(D, R_2)$ to privately return $\hat{\beta}$ (Line 10 of Algorithm 4). Let π be the permutation that sorts all choices of $V_j \in Adj(\mathcal{C}, V_i)$ by decreasing distance $d(ij|\mathbf{k}_{ij})$, where π_t

denotes the t-th largest element in π. Thus, $d(\pi_t)$ finds the t-th largest $d(ij|\mathbf{k}_{ij})$.

The output domain R_2 of $\hat{\beta}$ is $\{0, 1, \ldots, \alpha\}$ ($\alpha = |Adj(\mathcal{C}, V_i)|$). We assign a score of the quality function q_2 to each output as follows:

For $0 < t < \alpha$,

$$q_2(t, D) = \begin{cases} \min\Big(d(\pi_t), -d(\pi_{t+1})\Big) - 1, \\ \text{ if } \Big(\text{p-value}(\pi_t) < \tau\Big) \wedge \Big(\text{p-value}(\pi_{t+1}) \geq \tau\Big); \\ -d(\pi_{t+1}), \\ \text{if } \Big(\text{p-value}(\pi_t) < \tau\Big) \wedge \Big(\text{p-value}(\pi_{t+1}) < \tau\Big); \\ d(\pi_t) \text{ , otherwise.} \end{cases} \tag{4.2}$$

and

$$q_2(0, D) = \begin{cases} -d(\pi_1) - 1 \text{ , if p-value}(\pi_1) \geq \tau; \\ -d(\pi_1) \text{ , otherwise.} \end{cases} \tag{4.3}$$

$$q_2(\alpha, D) = \begin{cases} d(\pi_\alpha) - 1 \text{ , if p-value}(\pi_\alpha) < \tau; \\ d(\pi_\alpha) \text{ , otherwise.} \end{cases} \tag{4.4}$$

The quality function q_2 is based on the sorted distance scores. The exponential mechanism has a higher probability to select the number in the permutation at which the distance score flips from positive to negative. If all distance scores are negative, the exponential mechanism has a higher probability to select $\hat{\beta} = 0$. If all distance scores are positive, the exponential mechanism has a higher probability to select $\hat{\beta} = |Adj(\mathcal{C}, V_i)|$.

Corollary 2. The private selection of iteration number achieves ϵ_2-differential privacy by applying $\mathcal{M}^{\epsilon}_{q_2,1}(D, R_2)$. The L_1-sensitivity $S(q_2(D, r))$ of quality function q_2 is 1.

Proof: one record can only change the distance score (Equation 4.1) by 1. Because q_2

provides a desired lower bound on the distance score (Equation 4.1), q_2 changes at most 1 when the distance is changed by 1. Thus, the sensitivity of q_2 is at most 1.

In each round of the conditionally independent adjacent set selection, $\mathcal{M}_{q_2,1}^{\epsilon_2}(D, R_2)$ uses ϵ_2 out of total privacy budget ε.

Theorem 5. The PrivPC algorithm achieves ε-differential privacy by recursively applying private conditionally independent adjacent set selection procedure.

Proof: the proof follows by a combination of Corollary 1 and Corollary 2. PrivPC queries D in the private conditionally independent adjacent set selection procedure. One round of the private conditionally independent adjacent set selection achieves $(\epsilon_1 + \epsilon_2)$-differential privacy. PrivPC repeats this procedure until all of total budget ε is used up.

Let l be the maximal degree of order and let p be the number of vertices. The number of rounds of conditionally independent adjacent set selections is bounded by $l \times p$, which is much less than the number of conditional independence tests. Hence, PrivPC requires a much smaller privacy budget than the naive method for the same level of performance.

4.4.3 A Baseline Method (LapMech)

An improved method (LapMech) on the naive method is to add noise once on the input dataset D. A basic contingency table contains adequate information of D for all possible conditional independence tests. We add Laplace noise $Lap(0, 1/\varepsilon)$ on the basic contingency table. Any following conditional independence test only engages with the protected basic contingency table instead of original input dataset D. For each conditional independence test, the corresponding contingency table can be directly summarized from the protected basic contingency table. This algorithm only accesses the input data once. The basic contingency

table has a sensitivity of only 1, since one data point can change the count only by 1. We use the LapMech method as the baseline against our PrivPC algorithm for evaluation.

However, the basic contingency table is computationally infeasible for high-dimensional data or multiple-category variables. It suffers from over-fitting, as a large portion of value combinations may not be observed in the dataset D. The size of basic contingency table is $\prod_{i=1}^{p} Cat_i$, which is larger than n in most cases. Moreover, summarizing the count numbers $N_{ij|\mathbf{k}}$ from the basic contingency table to the corresponding contingency table is inefficient, and it accumulates noises for small tables. Hence, it's not an efficient way to achieve differential privacy in this case.

Applying the exponential mechanism on conditionally independent adjacent set selection has more advantages over the baseline method (LapMech). Because it uses contingency table, it doesn't encounter the problems of oversize, over-fitting or accumulated noise. At each round of conditionally independent adjacent set selection, it's independent on the dimension p or number of categories. We expect our PrivPC algorithm has better performance than the LapMech mechanism, especially in high-dimensional, complex cases.

4.5 Differentially Private Causal Graph Discovery Algorithm for Numerical Data

In this section, we propose a differentially private PC (PrivPC*) algorithm for numerical data.

When all nodes are random variables with the multivariate normal distribution, conditional independence tests are inferred from partial correlations [85]. Kalisch and Buhlmann [89] proposed a modified PC algorithm using partial correlation, which has an uniform consistency for Gaussian samples.

PrivPC* revises the PC algorithm to achieves the differential privacy based on the following procedure. Given a dataset D, we first calculate the covariance matrix Σ. Suppose Σ has a lower triangular structure (as Σ is symmetric), with variance σ_i^2 on diagonal and covariance $Cov(V_i, V_j)$ off diagonal.

We then add Laplace noise to the covariance matrix Σ to ensure differential privacy. The noise is randomly drawn from $Lap(0, S_f(D)/\varepsilon)$. Based on the private covariance matrix $\hat{\Sigma}$, the output of any following conditionally independence test (Algorithm 2 Line 8) between V_i and V_j given $\mathbf{V_k}$ is also differentially private, as it only queries the private the covariance matrix $\hat{\Sigma}$ instead of the input dataset D. Thus, differential privacy is ensure in any transformation or query over the private covariance matrix $\hat{\Sigma}$.

Using the private covariance matrix $\hat{\Sigma}$ in PrivPC*: After we get the private covariance matrix $\hat{\Sigma}$, we convert the private $\hat{\Sigma}$ to the private correlation matrix $\hat{\mathcal{P}}$, where $\hat{\rho}_{ij} = \dfrac{\widehat{Cov}(V_i, V_j)}{\hat{\sigma}_i \hat{\sigma}_j}$ denotes the correlation between V_i and V_j. To calculate the partial correlation $\hat{\rho}_{ij|\mathbf{k}}$ between V_i and V_j given $\mathbf{V_k}$, we use the sub-correlation matrix with entries from corresponding rows and columns and find the respective entries $\hat{\rho}_{ij}^{-1}, \hat{\rho}_{ii}^{-1}, \hat{\rho}_{jj}^{-1}$ from the inverse of the sub-matrix. The partial correlation $\hat{\rho}_{ij|\mathbf{k}}$ is calculated by

$$\hat{\rho}_{ij|\mathbf{k}} = -\frac{\hat{\rho}_{ij}^{-1}}{\sqrt{\hat{\rho}_{ii}^{-1}\hat{\rho}_{jj}^{-1}}}.$$

Then, we apply Fisher's z-transform to get private test statistics $\hat{Z}(ij|\mathbf{k})$:

$$\hat{Z}(ij|\mathbf{k}) = \frac{1}{2}\log\left(\frac{1+\hat{\rho}_{ij|\mathbf{k}}}{1-\hat{\rho}_{ij|\mathbf{k}}}\right)$$

Thus, the decision boundary on the conditional independence in Line 8 of Algorithm

1 becomes:

"If $\sqrt{n-|\mathbf{V_k}|-3}|\hat{Z}(ij|\mathbf{k})| \leq \Phi^{-1}(1-\tau/2)$, then V_i and V_j are conditionally independent on $\mathbf{V_k}$",

where n is the sample size; Φ is the CDF of $N(0,1)$; τ is the significance level.

Theorem 6. The PrivPC* algorithm achieves ε-differential privacy by applying the Laplace mechanism on the covariance matrix Σ. The global sensitivity $S_f(D)$ of the covariance matrix Σ is $\dfrac{p(p+1)}{2(n-1)}$.

Proof: We use $\mathbf{x}_u$ to denote the u-th record in dataset D. We use x_{ui} to denote the value of variable V_i in the u-th record ($i = 1, 2, \ldots, p$). Suppose it follows standard normal distribution $x_{ui} \sim N(0,1)$.

$$
\begin{aligned}
S_f &= ||\Sigma(D) - \Sigma(D')||_1 \\
&= \sum_{i=1}^{p} \left(\frac{\sum_{u=1}^{n} x_{ui}^2}{n-1} - \frac{\sum_{u=1}^{n-1} x_{ui}^2}{n-2} \right) + \sum_{i=1}^{p}\sum_{j=1}^{p} \left(\frac{\sum_{u=1}^{n} x_{ui}x_{uj}}{n-1} - \frac{\sum_{u=1}^{n-1} x_{ui}x_{uj}}{n-2} \right) \\
&\leq \sum_{i=1}^{p} \left(\frac{\sum_{u=1}^{n} x_{ui}^2}{n-1} - \frac{\sum_{u=1}^{n-1} x_{ui}^2}{n-1} \right) + \sum_{i=1}^{p}\sum_{j=1}^{p} \left(\frac{\sum_{u=1}^{n} x_{ui}x_{uj}}{n-1} - \frac{\sum_{u=1}^{n-1} x_{ui}x_{uj}}{n-1} \right) \\
&= \sum_{i=1}^{p} \left(\frac{x_{ni}^2}{n-1} \right) + \sum_{i=1}^{p}\sum_{j=1}^{p} \left(\frac{x_{ni}x_{nj}}{n-1} \right) \\
&= \frac{2\sum_{i=1}^{p} x_{ni}^2 + 2\sum_{i=1}^{p}\sum_{j=1}^{p} x_{ni}x_{nj}}{2(n-1)} = \frac{\sum_{i=1}^{p} x_{ni}^2 + \left(\sum_{i=1}^{p} x_{ni}\right)^2}{2(n-1)} \\
&\leq \frac{p + \left(p \times \sqrt{\frac{p}{p}}\right)^2}{2(n-1)} = \frac{p(p+1)}{2(n-1)}
\end{aligned}
\tag{4.5}
$$

where $i \neq j$, $\sum_{i=1}^{p} x_{ni}^2 = p$ because $\sigma^2 = \dfrac{\sum_{i=1}^{p} x_{ni}^2}{p} = 1$.

The global sensitivity S_f of $\Sigma(D)$ is $\dfrac{p(p+1)}{2(n-1)}$. When n is large, the added Laplace noise would be relatively small. $\square$

Compared to a basic contingency table, the correlation matrix has a much smaller size $(p \times p)$. Hence, partial correlation test via the private covariance matrix $\hat{\Sigma}$ doesn't introduce extra noise to test statistics, whereas the basic contingency table wastes privacy budget on under-represented entries in its cells. Hence, it's very plausible to add noise on the covariance matrix for numerical data.

4.6 Experiments

In our experiments, we implement our private PC algorithms (PrivPC and PrivPC*) based on the R-package *pcalg* [93]. The procedure of evaluating private PC algorithms against non-private PC algorithm is shown as follows. First, we randomly generate 5 graphs $\mathcal{G}$ with p nodes according to the maximal sparseness $s = \dfrac{\text{number of edges in } \mathcal{G}}{p(p-1)/2}$. Then, we generate a dataset D with n records according to the causal graph $\mathcal{G}$. After that, the non-private and private PC algorithms are applied on the dataset D to estimate a CPDAG. For all conditional independence tests used in our experiments, we restrict significance level to $\tau = 0.01$. Finally, we compare the true causal graph $\mathcal{G}$ and the estimated CPDAG to evaluate the performance of the private PC algorithms. We evaluate the estimated CPDAG for the undirected skeleton. Edge directions are not considered because they are decided in the second phase where the PC algorithm doesn't leak the privacy of the dataset D. The evaluation metrics used in our

experiments are:

$$\text{True Positive Rate (TPR)} = \frac{\text{correctly found edges}}{\text{true edges}}$$

$$\text{False Positive Rate (FPR)} = \frac{\text{misplaced edges}}{\text{true gaps (no edge)}}$$

$$\text{True Discovery Rate (TDR)} = \frac{\text{correctly found edges}}{\text{found edges}}$$

For TPR and TDR, the higher value indicates better performance. For FPR, the lower value indicates better performance.

4.6.1 Categorical Data

To generate the categorical dataset D, we use $TETRAD$ [94] to build causal graphs according to dimension p and sparseness s. Each node is assigned as a categorical variable with the number of category ranging from 2 to 5; then conditional probability tables corresponding to the structure of $\mathcal{G}$ are constructed; finally, n samples are independently drawn using the conditional probability tables. We evaluate the performance of PrivPC against LapMech and the non-private PC algorithm under different parameter settings. Table 4.2 shows the parameter values, with the default values in bold. Table 4.3 to Table 4.6 show the experimental results when one parameter varies and the others are fixed at default values. We highlight the better values between LapMech and PrivPC.

Table 4.3 shows the performance of non-private PC, LapMech, and PrivPC algorithms for different privacy budgets ε on categorical data. With $\varepsilon = 10$, both LapMech and PrivPC have high TPR and TDR. However, the utility of LapMech is much less preserved than that of PrivPC when a stronger privacy is enforced (smaller ε). For example, when $\varepsilon = 1$,

Table 4.2: Experimental parameters and values for categorical data

Parameter	Values
Privacy Budget ε	1, 3, **5**, 10
Sample Size n	1k, 3k, **5k**, 10k
Number of Nodes p	5, **10**, 15
Sparseness s	**0.2**, 0.4, 0.6

Table 4.3: The experimental results for different privacy budgets ε on categorical data $(n = 10000, p = 10, s = 0.2)$

ε		Non-private PC	LapMech	PrivPC
10	TPR	1.000±0.000	0.862±0.060	**0.896±0.136**
	FPR	0.006±0.012	0.011±0.016	**0.006±0.012**
	TDR	0.980±0.045	0.960±0.055	**0.980±0.044**
5	TPR	1.000±0.000	0.742±0.117	**0.873±0.125**
	FPR	0.006±0.012	0.012±0.017	**0.006±0.013**
	TDR	0.980±0.045	0.960±0.055	**0.980±0.045**
3	TPR	1.000±0.000	0.748±0.108	**0.841±0.121**
	FPR	0.006±0.012	0.018±0.017	**0.017±0.025**
	TDR	0.980±0.045	**0.940±0.055**	0.940±0.089
1	TPR	1.000±0.000	0.613±0.157	**0.817±0.206**
	FPR	0.006±0.012	0.105±0.024	**0.073±0.035**
	TDR	0.980±0.045	0.640±0.089	**0.740±0.134**

LapMech loses 25% on TPR and 30% on TPR. Whereas the PrivPC only loses 10% and 20% respectively. Hence, PrivPC is robust to different privacy budgets ε.

As mentioned in Section 4.4, PrivPC makes a much smaller number of edge elimination decisions than the number of conditional independence tests in the original PC algorithm. The number of edge elimination decisions and the number of conditional independence tests are bounded by the dimension p and the maximal degree of order l. We evaluate the performance of PrivPC over causal graphs with different dimensions p. Table 4.4 shows the results for different p. For $p = 5$, both LapMech and PrivPC have great util-

Table 4.4: The experimental results for different numbers of nodes p on categorical data ($\varepsilon = 5, n = 5000, s = 0.2$)

p		Non-private PC	LapMech	PrivPC
5	TPR	1.000±0.000	**1.000±0.000**	**1.000±0.000**
	FPR	0.025±0.056	**0.025±0.056**	**0.025±0.056**
	TDR	0.933±0.149	**0.933±0.149**	**0.933±0.149**
10	TPR	1.000±0.000	0.738±0.158	**0.884±0.121**
	FPR	0.011±0.015	0.035±0.036	**0.023±0.013**
	TDR	0.960±0.055	0.880±0.130	**0.920±0.045**
15	TPR	0.990±0.022	0.455±0.174	**0.815±0.128**
	FPR	0.019±0.016	0.181±0.080	**0.053±0.021**
	TDR	0.926±0.061	0.227±0.093	**0.798±0.073**

Table 4.5: The experimental results for different sample sizes n on categorical data ($\varepsilon = 5, p = 10, s = 0.2$)

n		Non-private PC	LapMech	PrivPC
10k	TPR	1.000±0.000	0.742±0.117	**0.873±0.125**
	FPR	0.006±0.012	0.012±0.017	**0.006±0.013**
	TDR	0.980±0.045	0.960±0.055	**0.980±0.045**
5k	TPR	1.000±0.000	0.738±0.158	**0.884±0.121**
	FPR	0.011±0.015	0.035±0.036	**0.023±0.013**
	TDR	0.960±0.055	0.880±0.130	**0.920±0.045**
3k	TPR	0.982±0.041	0.812±0.129	**0.894±0.161**
	FPR	0.011±0.015	0.050±0.035	**0.045±0.038**
	TDR	0.960±0.055	0.820±0.130	**0.840±0.134**
1k	TPR	1.000±0.000	**0.836±0.157**	0.812±0.077
	FPR	0.049±0.021	0.140±0.033	**0.101±0.032**
	TDR	0.820±0.084	0.440±0.152	**0.620±0.130**

ity. However, PrivPC outperforms LapMech significantly with increasing the dimension p of $\mathcal{G}$. For example, LapMech has only 45% on TPR and 22% on TDR when $p = 15$. On the contrary, PrivPC has about 80% on both FPR and TDR. Furthermore, LapMech will be infeasible for high-dimensional datasets (e.g., when $p \geq 20$, the size of contingency table

Table 4.6: The experimental results for different values of sparseness s on categorical data ($\varepsilon = 5, n = 5000, p = 10$)

s		Non-private PC	LapMech	PrivPC
0.2	TPR	1.000±0.000	0.738±0.158	**0.884±0.121**
	FPR	0.011±0.015	0.035±0.036	**0.023±0.013**
	TDR	0.960±0.055	0.880±0.130	**0.920±0.045**
0.4	TPR	1.000±0.000	**0.886±0.080**	0.774±0.062
	FPR	0.069±0.055	0.129±0.062	**0.084±0.041**
	TDR	0.866±0.100	0.741±0.121	**0.846±0.092**
0.6	TPR	1.000±0.000	**0.877±0.113**	0.809±0.073
	FPR	0.464±0.066	0.503±0.048	**0.280±0.081**
	TDR	0.328±0.165	0.266±0.160	**0.764±0.118**

is larger than $\prod_{i=1}^{20} Cat_i$). Therefore, PrivPC achieves much better utility when estimating a high-dimensional causal graph.

In LapMech, the sensitivity of a contingency table is 1. In PrivPC, the sensitivities of both distance score-based quality functions are also 1. As the sensitivities of LapMech and PrivPC are all based on count numbers, we evaluate the performance of each algorithm for different sample sizes n. Table 4.5 shows the experimental results for different sample sizes n. We can observe that the sample size n of input data D has strong impact on the performance of LapMech and PrivPC algorithm. When the sample size $n = 10k$, both LapMech and PrivPC have enough data to reach good utility. When the sample size n drops to $1k$, all the results of LapMech and PrivPC drop significantly. However, comparing with the LapMech which has only 44% on TDR, PrivPC achieves 62% on TDR. Thus, PrivPC can achieve a better true discovery rate when the sample size n is small.

The maximal degree of order l of a causal graph $\mathcal{G}$ is related to the dimension p and sparseness s. The maximal degree is roughly bounded by $p \times s$. We further evaluate the

PrivPC algorithm on causal graphs with various sparseness s. Table 4.6 shows the performance with different sparseness s on categorical datasets. For a dense causal graph ($s \geq 0.6$), LapMech has a high FPR and low TDR as expected. Conversely, PrivPC has a much lower FPR and higher TDR than LapMech. It even out-performs the non-private PC algorithm on TDR. Thus, a dense causal graph discovery via PrivPC can achieve a better result in terms of FPR and TDR.

We also compare the average processor time of LapMech and PrivPC. When $n = 5000, p = 10, s = 0.2$, LapMech has an average runtime of $111s$, whereas PrivPC only needs $9s$. As sparseness s or dimension p increases, the average runtime of LapMech becomes up to 1000 times worse than PrivPC. Hence, PrivPC requires a significantly less processor runtime than LapMech.

Overall, PrivPC achieves much better utility than LapMech. Meanwhile, PrivPC is also more robust to a smaller privacy budget ε, smaller sample size n, higher dimensional or more dense graphs. The advantage of the conditionally independent adjacent set selection approach allows PrivPC to achieve a high true discovery rate when estimating a more complex causal graph $\mathcal{G}$. Additionally, PrivPC runs much faster than LapMech.

4.6.2 Numerical Data

To evaluate the performance of PrivPC*, we generate causal graphs $\mathcal{G}$ with different p nodes and maximal sparseness s. Then, the numerical datasets D with n records are generated according to the given causal graphs $\mathcal{G}$ (with weights) with Gaussian distribution. We implement our PrivPC* algorithm using Laplace mechanism. We evaluate the performance of our PrivPC* algorithm against the non-private PC algorithm under different parameter settings (Table 4.7).

Table 4.7: Experimental parameters and values for numerical data

Parameter	Values
Privacy Budget ε	0.01, 0.1, **1**, 10
Sample Size n	100, 1000, **10000**
Number of Nodes p	10, 20, 50, 100
Sparseness s	0.1, **0.4**, 0.7

Table 4.8: The experimental results for different privacy budgets ε on numerical data ($n = 10000, p = 10, s = 0.4$)

ε		Non-private PC	PrivPC*
10	TPR	0.975±0.056	**0.975±0.056**
	FPR	0.094±0.096	**0.094±0.096**
	TDR	0.851±0.129	**0.851±0.129**
1	TPR	0.975±0.056	**0.975±0.056**
	FPR	0.094±0.096	**0.087±0.102**
	TDR	0.851±0.129	**0.865±0.143**
0.1	TPR	0.975±0.056	0.789±0.056
	FPR	0.094±0.096	0.187±0.093
	TDR	0.851±0.129	0.688±0.111
0.01	TPR	0.975±0.056	0.516±0.383
	FPR	0.094±0.096	0.373±0.062
	TDR	0.851±0.129	0.113±0.116

Table 4.8 to Table 4.11 show the experimental results when one parameter varies and the others are fixed at default values. We highlight the values on which PrivPC* has *no significant difference* from the non-private PC.

Table 4.8 shows the performance for different privacy budgets ε on the same numerical datasets. With $\varepsilon = 1$ or higher, the PrivPC* algorithm achieves good performance on all three metrics (no significant differences between non-private and private PC algorithms). With the smaller privacy budget $\varepsilon = 0.1$ or 0.01, the utility of PrivPC* decreases as the trade-off for a stronger privacy. Hence, PrivPC* achieves a good utility with privacy budget

Table 4.9: The experimental results for different sample sizes n on numerical data ($\varepsilon = 1, p = 10, s = 0.4$)

n		Non-private PC	PrivPC*
10000	TPR	0.975±0.056	**0.975±0.056**
	FPR	0.094±0.096	**0.087±0.102**
	TDR	0.851±0.129	**0.865±0.143**
1000	TPR	1.000±0.000	**0.932±0.075**
	FPR	0.140±0.096	**0.196±0.082**
	TDR	0.746±0.153	**0.632±0.126**
100	TPR	1.000±0.000	0.433±0.435
	FPR	0.255±0.058	0.372±0.064
	TDR	0.463±0.060	0.081±0.091

Table 4.10: The experimental results for different numbers of nodes p on numerical data ($\varepsilon = 1, n = 10000, s = 0.4$)

p		Non-private PC	PrivPC*
10	TPR	0.975±0.056	**0.975±0.056**
	FPR	0.094±0.096	**0.087±0.102**
	TDR	0.851±0.129	**0.865±0.143**
20	TPR	0.886±0.035	**0.853±0.048**
	FPR	0.299±0.024	**0.297±0.025**
	TDR	0.388±0.028	**0.401±0.050**
50	TPR	0.819±0.043	**0.785±0.082**
	FPR	0.374±0.009	**0.378±0.012**
	TDR	0.112±0.008	**0.094±0.019**
100	TPR	0.836±0.025	0.712±0.060
	FPR	0.393±0.008	**0.397±0.008**
	TDR	0.039±0.008	**0.023±0.004**

$\varepsilon \geq 1$. Therefore, PrivPC* has reasonable utility for stronger privacy restriction.

Because the sensitivity of covariance matrix Σ_D in PrivPC* $\frac{p(p+1)}{2(n-1)}$ is a function about n and p, we evaluate our PrivPC* algorithm against non-private algorithm over different sizes n of D and dimensions p of $\mathcal{G}$. Table 4.9 shows the experimental results for different

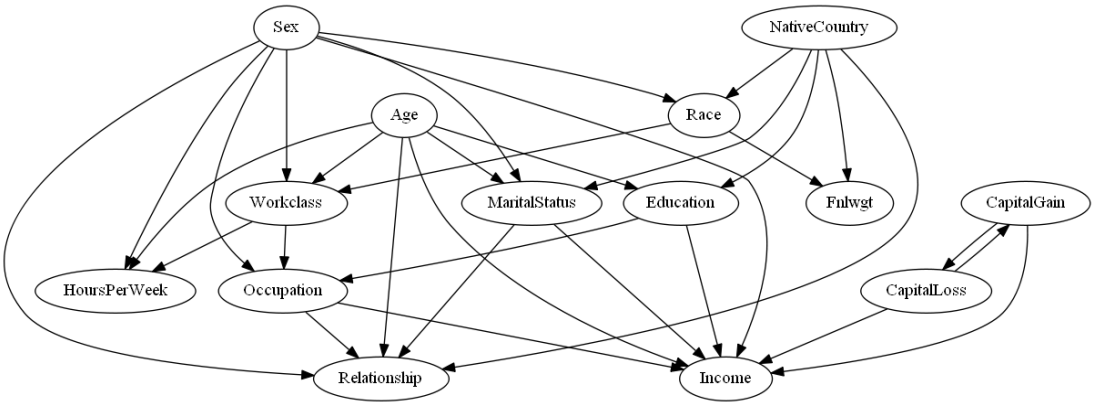
Table 4.11: The experimental results for different values of sparseness s on numerical data ($\varepsilon = 1, n = 10000, p = 20$)

s		Non-private PC	PrivPC*
0.1	TPR	0.918±0.080	0.701±0.193
	FPR	0.004±0.003	**0.004±0.003**
	TDR	0.971±0.027	**0.971±0.027**
0.4	TPR	0.886±0.035	**0.853±0.048**
	FPR	0.299±0.024	**0.297±0.025**
	TDR	0.388±0.028	**0.401±0.050**
0.7	TPR	0.803±0.102	**0.789±0.078**
	FPR	0.684±0.039	**0.688±0.028**
	TDR	0.190±0.030	**0.179±0.029**

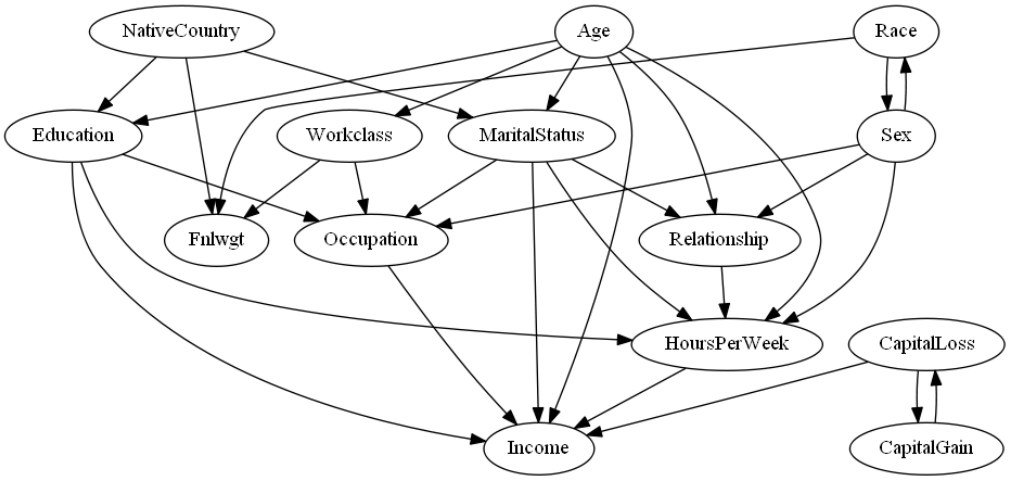
sample sizes n. For datasets with sample size $n = 1000$ or larger, PrivPC* can achieve as good utility as the non-private PC algorithm. When n is very small (e.g., $n = 100$), the effect of noise is magnified, so the performance of PrivPC* significantly decreases.

Table 4.10 shows the performance for different p-dimensional causal graphs. In the low dimensional setting, the PrivPC* algorithm can achieve as good utility as non-private algorithm. In the high dimensional setting ($p = 50$ or more), the TDR of both non-private and private PC algorithms are very low. When $p = 100$, the TPR of PrivPC* is significantly lower than the non-private PC algorithm. As we expected from the sensitivity, the utility of our algorithm will be compromised under a large p or small n. Yet, PrivPC* still achieves good performance in a wide range of cases ($n \geq 1000, p \leq 50$).

The sparseness s is another essential parameter of a causal graph other than the dimension p. We evaluate PrivPC* on causal graphs with same dimension but different sparseness. Table 4.11 shows the performance for different sparseness s. For a dense causal graph ($s \geq 0.4$), the PrivPC* algorithm has as good utility as non-private algorithm. However, for a very sparse causal graph $s = 0.1$, the PrivPC* algorithm has a lower TPR than



(a) Estimated CPDAG by Non-private PC



(b) Estimated CPDAG by PrivPC

Figure 4.1: Case analysis on the UCI Adult dataset ($\varepsilon = 10, n = 48842, p = 14$)

the non-private algorithm.

Overall, PrivPC* using Laplace mechanism on Gaussian data achieves a consistently good utility and robustness for most experiment settings.

4.6.3 Case Analysis on the UCI Adult Dataset

We further conduct experiments on the UCI Adult datset [95] which is made up of census data. The Adult dataset consists of 48842 records with 14 attributes such as age, eduation, sex, occupation, income, etc.. The computational complexity of the PC algorithm is an exponential function of the number of attributes and their domain sizes. For computational

feasibility we discretize each attribute's domain values into two to four categories to reduce the domain sizes. There's no "true graph" of Adult dataset. We apply the non-private PC algorithm to get an estimated CPDAG (shown in Figure 4.1a) and use it as the approximate true graph $\mathcal{G}$; then we apply the PrivPC algorithm to get a differentially private estimated CPDAG $\hat{\mathcal{G}}$ (shown in Figure 4.1b) and compare it to $\mathcal{G}$.

We assume the estimated CPDAG by the non-private PC algorithm is identical to the true graph $\mathcal{G}$. This approximate true graph $\mathcal{G}$ has 31 edges. The estimated CPDAG $\hat{\mathcal{G}}$ by the PrivPC algorithm has 28 edges. Most causal relationships discovered by the non-private algorithm are also captured by PrivPC. PrivPC achieves 78.6% on TPR, 14.3% on FPR, and 71.0% on TDR. Hence, the PrivPC algorithm is also effective on the real dataset.

4.7 Summary

In this work, we studied how to preserve differential privacy in constraint-based causal graph discovery. The naive differentially private PC algorithm requires a large privacy budget since it adds protection at each conditional independence test. To solve this problem, we developed a differentially private PC algorithm (PrivPC) based on the exponential mechanism. Because PrivPC reduces the number of edge elimination decisions, it reduces the required privacy budget significantly. The proposed PrivPC algorithm is robust over a wide range of parameter settings. Moreover, the conditionally independent adjacent set selection strategy achieves high true discovery rates for high-dimensional and dense causal graphs. Meanwhile, we also developed a Laplace mechanism based differentially private PC algorithm (PrivPC*) for numerical data. The experimental results also showed that our algorithm has good performance. The early version of this work is published at IEEE PAC 2017 [96].

5 Differentially Private Network Embedding

5.1 Introduction

Network embedding learns the lower dimensional representations of the vertices in a high-dimensional social network [97]. The latent representations encode the social relations in a continuous vector space, which can be used to conduct a variety of applications such as vertex classification and link prediction. The first network embedding model, DeepWalk [24], uses the sequences of vertices generated by random walks to learn the vertex representations. It adopts SkipGram [98], which was previously used to learn word representations in natural language processing. Several models based on the neural language model have been proposed, such as node2vec [99], Discriminative Deep Random Walk (DDRW) [100], Large-scale Information Network Embedding (LINE) [25] and Signed Network Embedding (SNE) [101].

However, releasing the representations of vertices in a social network gives malicious attackers a potential way to infer the sensitive information of individuals. For example, the widely-used network embedding methods like DeepWalk [24] and LINE [25] train the vertex representations based on the linkage information between vertices. Hence, the released vertex representations may potentially breach link privacy of social network users. Currently, it is under-exploited how to preserve differential privacy in network embedding. In this work, we aim to ensure no "privacy loss" in case of the inclusion or exclusion of an edge between two vertices from a network, a.k.a. link privacy.

In DeepWalk, the inputs of the two embedding models are generated by random walks and edge sampling. The objective function derived from random walks has an uncertain and

complex mapping from edges. Edge sampling directly discloses the presence or absence of an edge between a vertex pair, which leads to a high sensitivity for privacy protection. Thus, it is difficult to directly incorporate well-studied differential privacy mechanisms [8, 44] onto DeepWalk. Meanwhile, the matrix factorization based method, which is proven to be equivalent to DeepWalk, learns representations through factorizing a matrix with the pointwise mutual information of the vertices pairs in a network. It is convenient to achieve differential privacy in network embedding via small perturbation on matrix factorization.

In this work, we focus on developing a differential privacy preserving method of network embedding based on the equivalent matrix factorization method. We propose a differentially private network embedding method (DPNE). In this method, we leverage the findings that network embedding methods such as DeepWalk and LINE are equivalent to factorization of some matrices derived from the adjacency matrix of the original network and apply objective perturbation on the objective function of matrix factorization. We show that with only adding a small amount of noise onto the objective function, the learned low-dimensional representations satisfy differential privacy. Experimental results show that the embedded representations learned by DPNE achieve good utility with a small privacy budget on both vertex classification and link prediction tasks.

5.2 Related Work

Network Embedding The traditional network embedding methods [102] [103] [104] don't scale well for real world large information networks. Graph Factorization [105] uses a streaming algorithm for graph partitioning to improve factorization based embedding method. Word2vec [98] is an efficient tool using SkipGram to learn the representations of words in natural language processing tasks. There are several models based on the neural lan-

guage model: DeepWalk [24], node2vec [106], Discriminative Deep Random Walk (DDRW) [100], Deep Graph Kernels [107] and Large-scale Information Network Embedding (LINE) [25].

5.3 Preliminaries

For general preliminaries of differential privacy, please refer to Chapter 3.1.

Network Embedding. A social network is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a set of vertices and $\mathcal{E}$ is the set of edges. We use v_i to denote a vertex, and we use e_{ij} to denote an edge between a pair of vertices v_i and v_j. The goal of the network embedding is to learn the low-dimensional representations $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}| \times k}$ for all vertices in $\mathcal{V}$, where $k \ll |\mathcal{V}|$. The i-th row of $\mathbf{X}$ (denoted as $\mathbf{x}_i$) is the k-dimensional latent representation of vertex v_i.

DeepWalk. DeepWalk adopted SkipGram [98], which was previously used to learn word representations, to learn vertex representations according to the network structure. DeepWalk first generates short random walks for each vertex. Then the model uses the sequences of vertices S generated by random walks as the input of SkipGram function to learn the vertex representations. In particular, for each target vertex $v_i \in \mathcal{V}$ and a context vertex v_j within t window size of v_i in a walk sequence ($v_j \in C_i = \{v_{i-t}, \ldots, v_{i+t}\} \setminus \{v_i\}$), DeepWalk optimizes the co-occurrence probability between v_i and its context vertices within S:

$$\mathcal{L}(\mathcal{S}) = \frac{1}{|S|} \sum_{v_i \in S} \sum_{v_j \in C_i} \log \Pr(v_j | v_i), \tag{5.1}$$

where the probability function $\Pr(v_j|v_i)$ is defined by the softmax function:

$$\Pr(v_j | v_i) = \frac{\exp(\mathbf{x}_i \mathbf{y}_j^T)}{\sum_{v_{j'} \in \mathcal{V}_c} \exp(\mathbf{x}_i \mathbf{y}_{j'}^T)}, \tag{5.2}$$

where $\mathbf{x}_i$ and $\mathbf{y}_j$ are the k-dimensional representations of the target vertex v_i and the context vertex v_j, respectively; and $\mathcal{V}_c$ denotes the set of context vertices.

DeepWalk as Matrix Factorization. DeepWalk using SkipGram with Negative Sampling (SGNS) has been proven that it is equivalent to factorize a matrix $\mathbf{M}$ derived from $\mathcal{G}$ $\underset{|\mathcal{V}|\times|\mathcal{V}_c|}{\mathbf{M}} = \underset{|\mathcal{V}|\times k}{\mathbf{W}} \underset{k\times|\mathcal{V}_c|}{\mathbf{H}^T}$ [37]. The factorized matrices $\mathbf{W}, \mathbf{H}$ are equivalent to the vertex/context representations, as $\mathbf{w}_i = \mathbf{x}_i$ and $\mathbf{h}_j = \mathbf{y}_j$. Each value m_{ij} in $\mathbf{M}$ represents logarithm of the average probability that vertex v_i randomly walks to vertex v_j within fixed t steps. Formally, m_{ij} is defined as:

$$m_{ij} = \log \frac{[I_i(\mathbf{P} + \mathbf{P}^2 + \cdots + \mathbf{P}^t)]_j}{t}, \tag{5.3}$$

where $\mathbf{P}$ is the transition matrix of $\mathcal{G}$ with $p_{ij} = \frac{1}{d_i}$ if $e_{ij} \in \mathcal{E}$; d_i is the degree of v_i in $\mathcal{G}$; and I_i denotes an indicator vector, in which the i-th entry is 1 and the others are all 0.

Hence, we formalize the DeepWalk as matrix factorization $\mathbf{M} = \mathbf{W}\mathbf{H}^T$. Let Ω be the set of vertex pairs referenced by each entry m_{ij} of $\mathbf{M}$. The model aims to find matrices $\mathbf{W}$ and $\mathbf{H}$ to minimize the objective function as follows:

$$\mathcal{L}(\mathbf{w}_i, \mathbf{h}_j, \mathcal{G}) = \sum_{(i,j)\in\Omega} ||m_{ij} - \mathbf{w}_i\mathbf{h}_j^T||_2^2 + \lambda \sum_{i\in\mathcal{V}} ||\mathbf{w}_i||_2^2 + \mu \sum_{j\in\mathcal{V}_c} ||\mathbf{h}_j||_2^2, \tag{5.4}$$

where λ and μ are the weights of regularization terms.

We adopt the stochastic gradient descent (SGD) approach to minimize Equation 5.4. SGD iteratively learns $\mathbf{W}$ and $\mathbf{H}$. The partial derivatives of $\mathcal{L}(\mathbf{w}_i, \mathbf{h}_j, \mathcal{G})$ with respect to $\mathbf{w}_i$ and $\mathbf{h}_j$ are as follows:

$$\bigtriangledown_{\mathbf{w}_i} \mathcal{L}(\mathbf{w}_i, \mathbf{h}_j, \mathcal{G}) = -2 \sum_{j\in\mathcal{V}_c} \mathbf{h}_j(m_{ij} - \mathbf{w}_i\mathbf{h}_j^T), \tag{5.5}$$

$$\bigtriangledown_{\mathbf{h}_j} \mathcal{L}(\mathbf{w}_i, \mathbf{h}_j, \mathcal{G}) = -2 \sum_{i \in \mathcal{V}} \mathbf{w}_i (m_{ij} - \mathbf{w}_i \mathbf{h}_j^T). \tag{5.6}$$

5.4 Differentially Private Network Embedding

In this section, we describe the differentially private network embedding method (DPNE) based on the matrix factorization approach to achieve differential privacy in network embedding.

5.4.1 Differentially Private Network Embedding (DPNE)

DPNE adopts the objective perturbation mechanism on matrix factorization to protect the individual's link privacy in the social network. Note that $\mathbf{M}$ represents logarithm of the average probability that one vertex randomly walks to another vertex within fixed steps. When an edge e_{ij} in $\mathcal{G}$ is added or removed, the entire entries in $\mathbf{M}$ are changed accordingly. Hence, although there are some works [108] to protect the privacy in terms of a value in a matrix, it is not straightforward to adapt the existing models to our scenario. We need to derive the scale of the objective function in terms of changing one edge in $\mathcal{G}$.

In DPNE, we define the perturbed objective function of matrix factorization in Equation 5.4 as follows:

$$\mathcal{L}_{priv}(\mathbf{w}_i, \mathcal{G}) = \sum_{(i,j) \in \Omega} ||m_{ij} - \mathbf{w}_i \mathbf{h}_j^T||_2^2 + \lambda \sum_{i \in \mathcal{V}} ||\mathbf{w}_i||_2^2 + \mu \sum_{j \in \mathcal{V}_c} ||\mathbf{h}_j||_2^2 + \sum_{i \in \mathcal{V}} \mathbf{w}_i \boldsymbol{\eta}_i^T, \tag{5.7}$$

where $\mathbf{N} = [\boldsymbol{\eta}_i]_{|\mathcal{V}| \times k}$ is a noise matrix with each row $\boldsymbol{\eta}_i$ of $\mathbf{N}$ as a k-dimensional noise vector. In practice, the context representation matrix $\mathbf{H}$ is kept confidential. We first minimize Equation 5.4 to update $\mathbf{H}$, then fix $\mathbf{H}$ and learn $\mathbf{W}$ by minimizing Equation 5.7. Hence, $\mathbf{W}$ is the only matrix variable in Equation 5.7.

Theorem 7. Let $\mathbf{M}$ be a matrix where each of its entry m_{ij} is defined by Equation 5.3. Let $\boldsymbol{\eta}_i$ in Equation 5.7 be a k-dimensional noise vector that is independently and randomly picked for each vertex v_i from the density function $\Pr(\boldsymbol{\eta}_i) \propto \exp(-\frac{\epsilon||\boldsymbol{\eta}_i||}{2\Delta})$, where $\Delta = \max ||\mathbf{M}' - \mathbf{M}||$. The derived $\widehat{\mathbf{W}} = \arg\min_{\mathbf{W}} \mathcal{L}_{priv}(\mathbf{w}_i, \mathcal{G})$ by minimizing Equation 5.7 satisfies ϵ-differential privacy.

Proof. Let $\mathcal{G}$ and $\mathcal{G}'$ be two neighboring graphs differing by one edge, where $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, $\mathcal{G}' = (\mathcal{V}, \mathcal{E}')$, $\mathcal{E}' = \mathcal{E} \bigcup \{e_{rs}\}$. Let $\mathbf{M}$ and $\mathbf{M}'$ be two matrices derived from $\mathcal{G}$ and $\mathcal{G}'$ following Equation 5.3. Let $\mathbf{N}$ and $\mathbf{N}'$ be the noise matrices in Equation 5.7 when training with $\mathcal{G}$ and $\mathcal{G}'$. Meanwhile, $\mathcal{L}_{priv}(\mathbf{w}_i, \mathcal{G})$ is differentiable anywhere.

Let $\bar{\mathbf{W}} = \arg\min_{\mathbf{W}} \mathcal{L}_{priv}(\mathbf{w}_i, \mathcal{G}) = \arg\min_{\mathbf{W}} \mathcal{L}_{priv}(\mathbf{w}_i, \mathcal{G}')$, we have that $\forall v_i \in \mathcal{V}$, $\bigtriangledown_{\mathbf{w}_i} \mathcal{L}_{priv}(\bar{\mathbf{w}}_i, \mathcal{G}) = \bigtriangledown_{\mathbf{w}_i} \mathcal{L}_{priv}(\bar{\mathbf{w}}_i, \mathcal{G}') = 0$. Thereby,

$$\boldsymbol{\eta}_i - 2\sum_{j \in \mathcal{V}_c} \mathbf{h}_j (m_{ij} - \bar{\mathbf{w}}_i \mathbf{h}_j^T) = \boldsymbol{\eta}'_i - 2\sum_{j \in \mathcal{V}_c} \mathbf{h}_j (m'_{ij} - \bar{\mathbf{w}}_i \mathbf{h}_j^T); \tag{5.8}$$

We can derive from Equation 5.8 that:

$$\boldsymbol{\eta}_i - \boldsymbol{\eta}'_i = 2\sum_{j \in \mathcal{V}_c} \mathbf{h}_j (m'_{ij} - \bar{\mathbf{w}}_i \mathbf{h}_j^T) - 2\sum_{j \in \mathcal{V}_c} \mathbf{h}_j (m_{ij} - \bar{\mathbf{w}}_i \mathbf{h}_j^T) = 2\sum_{j \in \mathcal{V}_c} \mathbf{h}_j (m_{ij} - m'_{ij});$$

$$\sum_{i \in \mathcal{V}} (\boldsymbol{\eta}_i - \boldsymbol{\eta}'_i) = 2\sum_{i \in \mathcal{V}} \sum_{j \in \mathcal{V}_c} \mathbf{h}_j (m_{ij} - m'_{ij});$$

We normalize $||\mathbf{h}_j|| \leq 1$. Since $||\mathbf{M}' - \mathbf{M}|| \leq \Delta$, we have $||\mathbf{N} - \mathbf{N}'|| \leq 2\Delta$ regardless

of $\mathbf{H}$. Then for $\mathcal{G}$ and $\mathcal{G}'$,

$$\begin{aligned}\frac{\Pr[W=\bar{W}|\mathcal{G}]}{\Pr[W=\bar{W}|\mathcal{G}']} &= \frac{\prod_{i\in\mathcal{V}}\Pr(\boldsymbol{\eta}_i)}{\prod_{i\in\mathcal{V}}\Pr(\boldsymbol{\eta}_i')} = \exp(-\frac{\epsilon(\sum_{i\in\mathcal{V}}||\boldsymbol{\eta}_i|| - \sum_{i\in\mathcal{V}}||\boldsymbol{\eta}_i'||)}{2\Delta}) \\ &\leq \exp(-\frac{\epsilon(||\mathbf{N}-\mathbf{N}'||)}{2\Delta}) \leq \exp(\epsilon).\end{aligned} \tag{5.9}$$

□

The above proof also holds when we use $\left|\sum_{v_i\in\mathcal{V}}\mathbf{w}_i\boldsymbol{\eta}_i^T\right|$ as the noise term in the perturbed objective function. In our implementation, we use the absolute noise term to get a better performance on the optimization. Next, we show the upper bound of $\max||\mathbf{M}'-\mathbf{M}||$, which will be used for adding noise to the objective function.

Lemma 1. The L_2-sensitivity of $\mathbf{M}$ is $\max||\mathbf{M}'-\mathbf{M}|| \leq \sqrt{2}$.

Proof. The worst case for any t is when $\mathcal{E}=\emptyset$ in $\mathcal{G}$ and $\mathcal{E}'=\{e_{rs}\}$ in $\mathcal{G}'$. To find the bound of $\max||\mathbf{M}'-\mathbf{M}||$ under the worst case, we have, for $i=1,2,\cdots,t$, $\mathbf{P}'^i-\mathbf{P}^i$ is $\begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}$ and $\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}$ repeatedly. Hence, $||\mathbf{P}'^t-\mathbf{P}^t|| = ||\mathbf{P}'-\mathbf{P}||$. When window size $t=1$ under the worst case, $||\mathbf{M}'-\mathbf{M}|| \leq \sqrt{2}$. When window size $t\geq 2$ (still under the worst case),

$$\begin{aligned}\max_{(t\geq 2)}||\mathbf{M}'-\mathbf{M}|| &= \left|\left|\frac{\mathbf{P}'+\ldots+\mathbf{P}'^t}{t} - \frac{\mathbf{P}+\ldots+\mathbf{P}^t}{t}\right|\right| \\ &= \left|\left|\frac{(\mathbf{P}'-\mathbf{P})+\ldots+(\mathbf{P}'^t-\mathbf{P}^t)}{t}\right|\right| \\ &\leq \sum_{i=1}^{t}\left|\left|\frac{(\mathbf{P}'^i-\mathbf{P}^i)}{t}\right|\right| = \left|\left|\frac{t(\mathbf{P}'-\mathbf{P})}{t}\right|\right| = \max_{(t=1)}||\mathbf{M}'-\mathbf{M}||.\end{aligned} \tag{5.10}$$

□

To understand the worst case, we think about the physical meaning of $\mathbf{M}' - \mathbf{M}$. $\mathbf{M}$ represents all the t-step random walks generated from $\mathcal{G}$ and $\mathbf{M}'$ represents all the t-step random walks generated from $\mathcal{G}'$. $\mathbf{M}' - \mathbf{M}$ captures those walks that go through e_{rs}. Take the case where the original network $\mathcal{G}$ has no edges and we add edge e_{rs}. The set of random walks from $\mathcal{G}$ is $\emptyset$. The set of random walks from $\mathcal{G}'$ are all e_{rs}. The proportional change in $\mathbf{M}$ is maximum. Hence, it is the worst case where $\mathcal{E} = \emptyset$ in $\mathcal{G}$ and $\mathcal{E}' = \{e_{rs}\}$ in $\mathcal{G}'$. This is true for any $|\mathcal{V}|$ and any t. In such worst case, the bound of $\max ||\mathbf{M}' - \mathbf{M}||$ is $\sqrt{2}$.

5.4.2 DPNE vs. Other DP-preserving Embedding Approaches

A naive way to achieve differential privacy in network embedding is to get a differentially private matrix $\mathbf{M}$ (dpM) and then to apply matrix factorization, where $\mathbf{M}$ is calculated with the transition matrix $\mathbf{P}$ and its powers $\mathbf{P}^t$. To get a differentially private matrix $\mathbf{M}$, we can use the Laplace mechanism to add a noise matrix $\mathbf{N}$ on $\mathbf{M}$, where each entry n_{ij} of $\mathbf{N}$ is drawn i.i.d. from $Lap(S_f(\mathcal{G})/\epsilon)$, let $S_f(\mathcal{G})$ be the sensitivity of $\mathbf{M}$, $S_f(\mathcal{G}) = \max ||\mathbf{M}' - \mathbf{M}|| \leq \sqrt{2}$. The worst case for $||\mathbf{M}' - \mathbf{M}||$ is when adding an edge to two isolated vertices in $\mathcal{G}$. We will use dpM as a baseline in our empirical evaluation.

Another way to achieve differential privacy in network embedding is to enforce differential privacy in the process of network embedding models. Let D be a vertex-context set generated from the random walk sequences, where each member of D is a vertex-context pair (v_i, v_j). For a "walk step" from v_i to v_j in random walks, it is uncertain how many times in total the same "walk step" appears in D. For example, the number of times that an edge e_{ij} gets walked through by all the random walks sequences has the upper bound $\sum_{i=0}^{\min\{b, \lceil \log_a |\mathcal{V}| \rceil\}} a^i \times c \times (b - i)$, where a is the maximal degree of $\mathcal{G}$, b is the walk length, c is the number of walks starting at each vertex. Stochastic Gradient Descent (SGD) is used

in SkipGram to learn the embedding vectors based on the objective function. The input for SGD is the vertex-context set D. Although there are existing works on how to apply objective perturbation [8] or exponential mechanism [44] on SGD to make private updates, the privacy of D and the privacy of $\mathcal{G}$ are not the same. For example, if we want to apply the functional mechanism [4] on DeepWalk with hierarchical softmax [24], in terms of the privacy of D, the hierarchical softmax function iterates one time over each vertex-context pair in D; the sensitivity of the hierarchical softmax function on D is about $\lceil \log_2 |\mathcal{V}| \rceil \, (k/2 + k^2/8)$. But in terms of the privacy of $\mathcal{G}$, for each edge in $\mathcal{G}$, the hierarchical softmax function iterates an unset number of times; thus, the sensitivity of the hierarchical softmax function on $\mathcal{G}$ is very large after multiplying the iteration times.

Also, negative sampling is used for SkipGram function in DeepWalk or LINE. The processes of positive sampling and negative sampling already indicate link privacy. It is viable to make the sampling process private by applying the exponential mechanism [2] or the Laplace mechanism [1]. However, the sensitivity of edge sampling domain is also large. Hence, it is challenging to intuitively apply the existing differentially private approaches in DeepWalk to achieve privacy protection on $\mathcal{G}$.

The equivalent matrix factorization approach avoids random walks and negative sampling. The effect of window size in DeepWalk is expressed as the powers of the transition matrix. It considers the expected times that the vertex pairs appear rather than the randomly generated times. It is more convenient to apply differential privacy mechanisms on the matrix factorization based network embedding method. The only remaining challenge is to bridge the "gap" between the privacy of D and the privacy of $\mathcal{G}$. Theorem 7 addresses this problem.

5.5 Experiments

In this section, we evaluate the performance of DPNE on two tasks: vertex classification and link prediction. For vertex classification, we predict the label of each vertex in the network by using vertex embeddings as inputs to build a classifier. For link prediction, we aim to use vertex embeddings to predict whether there is an edge between two vertices.

Baselines. We compare our differentially private network embedding method (DPNE) with the naive method (dpM) and the non-private network embedding as matrix factorization method (non-priv).

Datasets. We adopt the following three datasets to evaluate the proposed model. 1) Wiki contains 2,405 documents from 19 classes and 17,981 links between them. 2) Cora is a research paper set which contains 2,708 machine learning papers from 7 classes and 5,429 links between them. 3) Citeseer is another research paper set which contains 3,312 publications from 6 classes and 4,732 links between them.

Parameter Settings. In our experiments, we set the window size $t = 2$ and $\mathbf{M} = (\mathbf{P} + \mathbf{P}^2)/2$. For all three datasets, we choose a series of values for the embedding size $k = \{10, 20, 50, 100, 200, 500, 1000\}$ for vertex representations, and a series of values for the privacy budget $\epsilon = \{0.01, 0.1, 1, 10, 100, 1000\}$. We set the regularization coefficients $\lambda = \mu = 0.001$ and the learning rate $\gamma = 0.015$. The train ratios of SVM and logistic regression in the two tasks are both 10%. For each parameter setting, we report the average result over 10 different runs.

Table 5.1: Comparing the accuracy for vertex classification with different privacy budget ϵ [non-priv: (a) Wiki 0.555 (b) Cora 0.700 (c) Citeseer 0.505]

Dataset	(a) Wiki		(b) Cora		(c) Citeseer	
ϵ	dpM	DPNE	dpM	DPNE	dpM	DPNE
0.01	0.094	0.093	0.187	0.181	0.179	0.178
0.1	0.091	0.096	0.189	0.213	0.182	0.186
1	0.088	0.521	0.190	0.662	0.177	0.452
10	0.090	0.527	0.196	0.669	0.180	0.459
100	0.355	0.537	0.500	0.677	0.311	0.466
1000	0.545	0.552	0.679	0.700	0.497	0.490

5.5.1 Vertex Classification Task

For the vertex classification task, we first get the vertex representation $\mathbf{W}$ based on the matrix factorization method. Then, we train a multi-class support vector machine (SVM) on a training dataset L based on a subset of vertex representations $\mathbf{W}_L$ and further predict the labels of a testing dataset U by the SVM classifier based on $\mathbf{W} \backslash \mathbf{W}_L$.

5.5.1.1 Different Privacy Budgets

We evaluate the performance of two private algorithms on all three datasets with embedding size $k = 100$ and different privacy budgets ϵ. Table 5.1 shows the comparison results of each method for vertex classification on three datasets. We can observe that both DPNE and dpM have the similar trend on three datasets in terms of accuracy while we increase the privacy budget. The classification results of DPNE are close to the non-private method when $\epsilon \geq 1$. However, the performance of dpM method has a big improvement when $\epsilon \geq 100$. It indicates that, compared with dpM, DPNE can achieve the same performance with a much smaller privacy budget. Meanwhile, when the $\epsilon \leq 0.1$, both private methods have poor performance. It indicates that when the privacy budget is low, the matrix factorization method cannot be converged due to the large noisy injected to the objective function or

Table 5.2: Comparing the accuracy for vertex classification with different embedding size k and privacy budget ϵ (dataset=Wiki)

$k\epsilon$	non-priv	DPNE					
		0.01	0.1	1	10	100	1000
10	0.324	0.080	0.108	0.255	0.265	0.272	0.350
20	0.466	0.072	0.095	0.420	0.428	0.429	0.457
50	0.537	0.080	0.087	**0.526**	0.520	0.515	0.539
100	**0.555**	0.093	0.096	0.521	**0.527**	**0.537**	**0.552**
200	0.530	0.096	0.101	0.470	0.511	0.515	0.539
500	0.474	0.105	0.110	0.285	0.437	0.424	0.482
1000	0.429	**0.109**	**0.112**	0.169	0.270	0.295	0.405

matrix itself.

5.5.1.2 Different Embedding Sizes

Based on Equation 3.6, η_i increases with embedding size k. There is potentially a compromised performance of DPNE at a larger k. We evaluate the performance of the DPNE algorithm on the Wiki dataset with different embedding size k and privacy budget ϵ. For non-priv, as shown in the second column of Table 5.2, the highest accuracy 55.5% is achieved when $k = 100$. When increasing or decreasing embedding size k, the accuracy decreases. For DPNE, with relatively large privacy budget, $\epsilon = 10, 100, 1000$, high accuracy is also achieved when $k = 100$. When $\epsilon = 1$, high accuracy is achieved when $k = 50$. However, with very small privacy budget, $\epsilon = 0.01, 0.1$, DPNE has significantly lower accuracy comparing to non-priv no matter how we choose k.

5.5.2 Link Prediction Task

For the link prediction task, we first use vertex representations to compose edge representations. Given a pair of vertices v_i, v_j connected by an edge, we use an Hadamard operator to combine the vertex vectors $\mathbf{w}_i$ and $\mathbf{w}_j$ to compose the edge vector $\widehat{\mathbf{e}}_{ij} = \mathbf{w}_i * \mathbf{w}_j$

Table 5.3: Comparing the accuracy for link prediction under different privacy budget ϵ [non-priv: (a) Wiki 0.734 (b) Cora 0.697 (c) Citeseer 0.699]

Dataset	(a) Wiki		(b) Cora		(c) Citeseer	
ϵ	dpM	DPNE	dpM	DPNE	dpM	DPNE
0.01	0.502	0.520	0.501	0.532	0.498	0.535
0.1	0.502	0.524	0.501	0.541	0.501	0.523
1	0.503	0.743	0.500	0.698	0.503	0.690
10	0.527	0.713	0.552	0.673	0.566	0.666
100	0.708	0.720	0.729	0.666	0.751	0.662
1000	0.725	0.725	0.706	0.703	0.695	0.696

[106]. Then, we use the constructed edge vectors as inputs to train a logistic regression classifier and adopt the classifier to predict the presence or absence of an edge.

5.5.2.1 Different Privacy Budgets

We evaluate the performance of two private algorithms on three datasets for link prediction with embedding size $k = 100$ and different privacy budgets ϵ. Table 5.3 shows the link prediction accuracy of DPNE and dpM. We observe that when $\epsilon \leq 0.1$, both private algorithms have only about 50% accuracy on three datasets. The accuracy of DPNE has a big improvement while ϵ increases to 1. However, dpM achieves the comparable results when the $\epsilon \geq 100$. It indicates DPNE can achieve better performance with a small privacy budget.

5.5.2.2 Different Embedding Sizes

We also evaluate the performance of the DPNE algorithm on the Wiki dataset with different embedding size k and privacy budget ϵ. Table 5.4 shows our result. For non-priv, it achieves the highest accuracy 74.7% when $k = 200$. For DPNE, it often achieves the highest accuracy with $k = 200$ for large privacy budget values and with $k = 1000$ for small privacy budget values. More interestingly, DPNE outperforms non-priv at large k for $\epsilon = 10$ or 100.

Table 5.4: Comparing the accuracy for link prediction with different embedding size k and privacy budget ϵ (dataset=Wiki)

$k\epsilon$	non-priv	DPNE					
		0.01	0.1	1	10	100	1000
10	0.607	0.499	0.552	0.614	0.612	0.609	0.605
20	0.648	0.500	0.538	0.661	0.650	0.654	0.649
50	0.717	0.510	0.525	0.718	0.694	0.695	0.718
100	0.734	0.520	0.524	0.743	0.713	0.720	0.725
200	**0.747**	0.532	0.541	**0.763**	0.734	**0.741**	**0.746**
500	0.670	0.550	0.566	0.733	0.778	0.721	0.695
1000	0.650	**0.568**	**0.595**	0.698	**0.784**	0.726	0.679

5.6 Summary

In this work, we developed a differentially private network embedding method (DPNE) based on DeepWalk as matrix factorization. We applied the objective perturbation approach on the objective function of matrix factorization. Our evaluation shows that on both vertex classification and link prediction tasks our DPNE achieves satisfactory performance. The early version of this work is published at PAKDD 2018 [109].

6 Achieving Fair Data Generation and Classification through Generative Adversarial Networks

6.1 Introduction

Fairness-aware learning is increasingly important in data mining. Discrimination prevention aims to prevent discrimination in the training data before it is used to conduct the predictive analysis. In this paper, we first focus on fair data generation that ensures the generated data is discrimination free. Inspired by generative adversarial networks (GAN), we present fairness-aware generative adversarial networks, called FairGAN, which are able to learn a generator producing fair data and also preserving good data utility. Another task that is equally important in fair machine learning is how to build fair classifiers. We also propose FairGAN$^+$ to address both tasks simultaneously. In addition to releasing fair data, FairGAN$^+$ can also produce a classifier that makes guaranteed fair predictions. Inspired by ACGAN, we incorporate a classifier into the FairGAN$^+$ structure. The classifier learns to predict labels based on the unprotected attribute. Experiments on a real dataset show the effectiveness of FairGAN and FairGAN$^+$.

6.2 FairGAN

For general preliminaries of fairness-aware machine learning and generative adversarial networks, please refer to Chapter 3.2 and Chapter 3.3, respectively.

FairGAN aims to generate a fair dataset which achieves the statistical parity w.r.t the protected attribute.

FairGAN consists of one generator G and two discriminators D_1 and D_2. We adopt

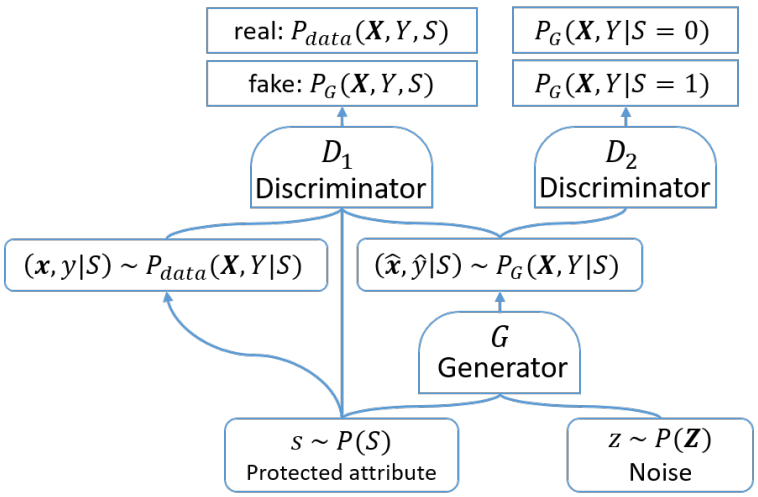


Figure 6.1: The structure of FairGAN

the revised generator from medGAN [81] to generate both discrete and continuous data. Figure 6.1 shows the structure of FairGAN. In FairGAN, every generated sample has a corresponding value of the protected attribute $s \sim P(S)$. The generator G generates a fake pair $(\hat{\mathbf{x}}, \hat{y}|S) \sim P_G(\mathbf{X}, Y|S)$. The discriminator D_1 is trained to distinguish between the real data from $P_{\text{data}}(\mathbf{X}, Y, S)$ and the generated fake data from $P_G(\mathbf{X}, Y, S)$.

Meanwhile, in order to make the generated dataset achieve fairness, a constraint is applied to the generated samples, which aims to keep $P_G(\mathbf{X}, Y|S = 1) = P_G(\mathbf{X}, Y|S = 0)$. Therefore, another discriminator D_2 is incorporated into the FairGAN model and trained to distinguish the two categories of generated samples, $P_G(\mathbf{X}, Y|S = 1)$ and $P_G(\mathbf{X}, Y|S = 0)$.

The value function of the minimax game is described as:

$$\min_G \max_{D_1, D_2} V(G, D_1, D_2) = V_1(G, D_1) + \lambda V_2(G, D_2), \tag{6.1}$$

where

$$\begin{aligned} V_1(G, D_1) =& \mathbb{E}_{s\sim P(S),(\mathbf{x},y)\sim P_{\text{data}}(\mathbf{X},Y|S)}[\log D_1(\mathbf{x}, y, s)] \\ &+ \mathbb{E}_{s\sim P(S),(\hat{\mathbf{x}},\hat{y})\sim P_G(\mathbf{X},Y|S)}[\log(1 - D_1(\hat{\mathbf{x}}, \hat{y}, s))], \end{aligned} \tag{6.2}$$

$$V_2(G, D_2) = \mathbb{E}_{(\hat{\mathbf{x}},\hat{y})\sim P_G(\mathbf{X},Y|S=1)}[\log D_2(\hat{\mathbf{x}}, \hat{y})] + \mathbb{E}_{(\hat{\mathbf{x}},\hat{y})\sim P_G(\mathbf{X},Y|S=0)}[\log(1 - D_2(\hat{\mathbf{x}}, \hat{y}))], \tag{6.3}$$

λ is a hyperparameter that specifies a trade off between utility and fairness of data generation.

The first value function V_1 is similar to a conditional GAN model [110], where the generator G seeks to learn the joint distribution $P_G(\mathbf{X}, Y|S)$ over real data $P_{\text{data}}(\mathbf{X}, Y|S)$. The second value function V_2 aims to make the generated samples not encode any information supporting to predict the value of protected attribute S. Therefore, D_2 is trained to correctly predict S given a generated sample while the generator G aims to fool the discriminator D_2. Once the generated sample $\{\hat{\mathbf{x}}, \hat{y}\}$ cannot be used to predict the protected attribute, the correlation between $\{\hat{\mathbf{x}}, \hat{y}\}$ and S is removed, i.e., $\{\hat{\mathbf{x}}, \hat{y}\} \perp S$. FairGAN can ensure that the generated samples do not have the disparate impact.

6.3 FairGAN$^+$

FairGAN$^+$ aims to simultaneously achieve fair data generation and fair classification through the GAN architecture. For fair data generation, we consider statistical fairness and/or ϵ-fairness. For fair classification, we consider demographic parity, equality of odds and/or equality of opportunity.

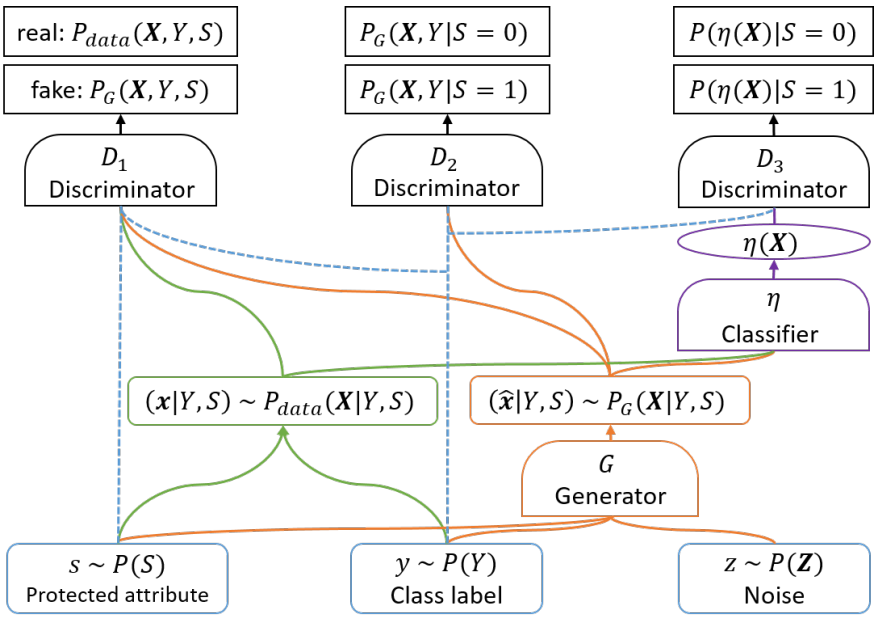


Figure 6.2: The structure of FairGAN$^+$

6.3.1 Model Framework

FairGAN$^+$ consists of one generator G, one classifier η and three discriminators D_1, D_2 and D_3. Figure 6.2 shows the structure of FairGAN$^+$. In FairGAN$^+$, each generated sample $G(\mathbf{z})$ has a corresponding value pair of the protected attribute $s \sim P_{data}(S)$ and the class label $y \sim P_{data}(Y)$. The generator $G(\mathbf{z})$ generates fake $\hat{\mathbf{x}}$ following the conditional distribution $P_G(\mathbf{X}|Y, S)$ given a noise variable $\mathbf{z}$. We adopt the revised generator from medGAN [81] so the generator $G(\mathbf{z})$ can generate both discrete and continuous data. The classifier $\eta : \mathbf{X} \to Y$ outputs the prediction $\eta(\mathbf{X})$ from $\mathbf{X}$. The prediction $\eta(\mathbf{X})$ is trained to both accurately predict the label Y and be free from discrimination. The discriminator D_1 is trained to distinguish between the real data from $P_{data}(\mathbf{X}, Y, S)$ and the generated fake data from $P_G(\mathbf{X}, Y, S)$. The generator plays adversarial game with D_1 to generate close-to-real fake data. The discriminator D_2 is trained to distinguish values of the protected attribute of each sample, $P_G(\mathbf{X}, Y|S = 1)$ and $P_G(\mathbf{X}, Y|S = 0)$. D_2 works like a fairness constraint to data

generation. The generator plays another adversarial game with D_2 so the generated data satisfy fairness notions in data. The discriminator D_3 is trained to distinguish values of the protected attribute from the predictions made by η, $P(\eta(\mathbf{X}) = 1|S = 1)$ and $P(\eta(\mathbf{X}) = 1|S = 0)$. D_3 works like a fairness constraint to classification. The classifier plays adversarial game with D_3 so its predictions satisfy fairness notions in classification.

The objective function of FairGAN$^+$ is $J = V + L$, where V is the value function of the overall minimax games between generator, classifier and discriminators; L is the objective function of the classifier.

The value function V of the overall minimax game is described as:

$$\min_{G,\eta} \max_{D_1,D_2,D_3} V(G, \eta, D_1, D_2, D_3) = V_1(G, D_1) + \lambda V_2(G, D_2) + \mu V_3(\eta, D_3),$$

where

$$\begin{aligned} V_1(G, D_1) =& \mathbb{E}_{s\sim P(S),y\sim P(Y),\mathbf{x}\sim P_{data}(\mathbf{X}|Y,S)}[\log D_1(\mathbf{x}, y, s)] \\ &+ \mathbb{E}_{s\sim P(S),y\sim P(Y),\hat{\mathbf{x}}\sim P_G(\mathbf{X}|Y,S)}[\log(1 - D_1(\hat{\mathbf{x}}, y, s))], \end{aligned} \tag{6.4}$$

$$V_2(G, D_2) = \mathbb{E}_{y\sim P(Y),\hat{\mathbf{x}}\sim P_G(\mathbf{X}|Y,S=1)}[\log D_2(\hat{\mathbf{x}}, y)] + \mathbb{E}_{y\sim P(Y),\hat{\mathbf{x}}\sim P_G(\mathbf{X}|Y,S=0)}[\log(1 - D_2(\hat{\mathbf{x}}, y))], \tag{6.5}$$

$$V_3(\eta, D_3) = \mathbb{E}_{\mathbf{x}\sim P(\mathbf{X}|Y,S=1)}[\log D_3(\eta(\mathbf{x}))] + \mathbb{E}_{\mathbf{x}\sim P(\mathbf{X}|Y,S=0)}[\log(1 - D_3(\eta(\mathbf{x}))], \tag{6.6}$$

λ is a hyperparameter that specifies a trade-off between data utility and fairness of data generation, μ is a hyperparameter that specifies a trade-off between classification accuracy and classification fairness of η.

The first two value functions V_1, V_2 are similar to the ones in FairGAN. The third value function V_3 aims to make the prediction $\eta(\mathbf{X})$ of samples not encode any information supporting to predict the value of the protected attribute S. Therefore, D_3 is trained to correctly predict S given a sample while the classifier η aims to fool the discriminator D_3. Once the prediction of η cannot be used to predict the protected attribute S, the correlation between $\eta(\mathbf{X})$ and S is removed, i.e., $\eta(\mathbf{X}) \perp S$. Then FairGAN$^+$ can release η, which achieves the desired fairness notion.

The objective function L of the classifier is described as:

$$\max_{G,\eta} L(G,\eta) = \mathbb{E}_{y\sim P(Y),\mathbf{x}\sim P_{data}(\mathbf{X}|Y,S)}[y \log \eta(\mathbf{x})] + \mathbb{E}_{y\sim P(Y),\hat{\mathbf{x}}\sim P_G(\mathbf{X}|Y,S)}[y \log \eta(\hat{\mathbf{x}})].$$

The classifier η maximizes the log-likelihood of the correct class labels as it makes more accurate predictions during training. The generator G also maximizes log-likelihood of the correct class labels as it generates samples that match each class accordingly. We take advantage of the feedback loop of D_1, G, η. Improving the utility of G can improve the utility of η on making correct predictions. Improving η can improve G on generating more realistic samples conditioned on each class. We also take advantage of the feedback loop of D_2, G, D_3, η. Improving the fairness of G can improve η on making fair predictions. Improving the fairness of η can improve G on generating fair data. Hence, by simultaneously learning a generative model and a classifier, FairGAN$^+$ can perform better than a standalone generative model or a standalone classifier.

6.3.2 Application to Different Classification-based Fairness

Models that modify data or generate fair data apply fairness constraints to enforce independence between the class label Y and the protected attribute S. However, such pre-processing methods cannot guarantee the independence between the predicted class label $\eta(\mathbf{X})$ and the protected attribute S. Constraints for some classification-based fairness notions require to constrain not only the association between the predicted class label $\eta(\mathbf{X})$ and the protected attribute S, but also the conditioned association when conditioning on the real class label Y. In our model, we connect the real class label Y to the constraints on the predicted class label $\eta(\mathbf{X})$ through the discriminator D_3. The inputs of D_3 are both the predicted class label $\eta(\mathbf{X})$ from η and the real class label Y from the prior distribution. This is not necessary for demographic parity but essential for equality of odds and equality of opportunity.

Demographic parity is defined as $P(\eta(\mathbf{X}) = 1|S = 1) = P(\eta(\mathbf{X}) = 1|S = 0)$. It simply requires that the predictions of η is independent of the protected attribute S. In the above general framework, we mostly discuss the model in terms of demographic parity. The value function V_3 is exactly as Equation 6.6. Once the prediction of η cannot be used to predict the protected attribute S, the correlation between $\eta(\mathbf{X})$ and S is removed, i.e., $\eta(\mathbf{X}) \perp S$. Then FairGAN$^+$ can release η, which achieves demographic parity.

Equality of odds is defined as $P(\eta(\mathbf{X}) = 1|Y = y, S = 1) = P(\eta(\mathbf{X}) = 1|Y = y, S = 0)$, where $y \in \{0, 1\}$. As a classification-based fairness notion, equality of odds considers that individuals who qualify for a desirable outcome should have an equal chance of being correctly classified for this outcome. It evaluates correlation between the prediction of classifier $\eta(\mathbf{X})$ and the protected attribute S based on the value of real label Y. To achieve equality of

odds for a classifier in the FairGAN$^+$ framework, we need to provide the corresponding real label y of the prediction $\eta(\mathbf{X})$ to the discriminator D_3. As a constraint of equality of odds to classification. D_3 aims to keep $P(\eta(\mathbf{X}) = 1|Y = y, S = 1) = P(\eta(\mathbf{X}) = 1|Y = y, S = 0)$. D_3 is trained to distinguish the two categories of the predictions made by η given the real label $Y = y$, $P(\eta(\mathbf{X}) = 1|Y = y, S = 1)$ and $P(\eta(\mathbf{X}) = 1|Y = y, S = 0)$. In this case, the third value function V_3 of the minimax game becomes

$$V_3(\eta, D_3) = \mathbb{E}_{y\sim P(Y),\mathbf{x}\sim P(\mathbf{X}|Y=y,S=1)}[\log D_3(\eta(\mathbf{x})|Y = y)]$$
$$+ \mathbb{E}_{y\sim P(Y),\mathbf{x}\sim P(\mathbf{X}|Y=y,S=0)}[\log(1 - D_3(\eta(\mathbf{x})|Y = y))].$$

V_3 aims to make the prediction $\eta(\mathbf{X})$ of samples not encode any information supporting to predict the value of the protected attribute S given the real label $Y = y$. Therefore, D_3 is trained to correctly predict S given a sample and the corresponding real label Y while the classifier η aims to fool the discriminator D_3. Once the prediction of η cannot be used to predict the protected attribute S given the real label $Y = y$, the conditional correlation between $\eta(\mathbf{X})$ and S is removed, i.e., $\eta(\mathbf{X}) \perp S|Y = y$. Then FairGAN$^+$ can release η, which achieves equality of odds.

Equality of opportunity is a special case of equality of odds. For FairGAN$^+$ based on equality of opportunity, the idea is straightforward. It only needs to consider the case $Y = 1$, so the discriminator D_3 only aims to constrain a subgroup instead of the whole population. In this case, the third value function V_3 of the minimax game becomes

$$V_3(\eta, D_3) = \mathbb{E}_{y\sim P(Y),\mathbf{x}\sim P(\mathbf{X}|Y=1,S=1)}[\log D_3(\eta(\mathbf{x})|Y = 1)]$$
$$+ \mathbb{E}_{y\sim P(Y),\mathbf{x}\sim P(\mathbf{X}|Y=1,S=0)}[\log(1 - D_3(\eta(\mathbf{x})|Y = 1))].$$

Once the prediction of η cannot be used to predict the protected attribute S given the real label $Y = 1$, the conditional correlation between $\eta(\mathbf{X})$ and S is removed, i.e., $\eta(\mathbf{X}) \perp S|Y = 1$. Then FairGAN$^+$ can release η, which achieves equality of opportunity.

6.4 Experiments

We evaluate the performance of FairGAN, FairGAN$^+$ on fair data generation and fair classification.

6.4.1 Experimental Setup

6.4.1.1 Baselines

We compare with the ACGAN model [82] on effectiveness, and with adversarial debiasing [66] on fair classification.

ACGAN aims to generate the synthetic samples that have the same distribution as the real data given the values of labels, i.e., $P_G(\mathbf{X}|Y, S) = P_{data}(\mathbf{X}|Y, S)$. ACGAN also contains a classifier, but does not enforce fairness in either data generation or classification. Note that, in order to match our scenario, we independently generate both the label Y and the protected attribute S in the ACGAN. Thus, ACGAN achieves statistical fairness in data generation similar to random shuffle the protected attribute. The classifiers built in both FairGAN$^+$ and ACGAN are logistic regression model.

Adversarial debiasing (AD) also applies adversarial learning to achieve fairness in classification. AD cannot generate fair data as it is not a generative model.

6.4.1.2 Datasets

We evaluate FairGAN, FairGAN$^+$ and baselines on the UCI Adult dataset[95]. It contains 48,842 samples. The class label Y is "Income", and the protected attribute S is "Gender". There are 12 unprotected attributes $\mathbf{X}$. We convert each attribute to a one-hot vector and combine all of them to a feature vector with 57 dimensions.

Besides adopting the original Adult dataset (**D1-Real**), we also generate four types of synthetic data, **D2-ACGAN** that is generated by ACGAN, **D3-FairGAN** that is generated by FairGAN with $\lambda = 1$, **D4-FairGAN$^+$(DP)** that is generated by FairGAN$^+$ based on demographic parity with $\lambda = \mu = 1$, and **D5-FairGAN$^+$(EO)** that is generated by FairGAN$^+$ based on equality of odds with $\lambda = \mu = 1$. We set the sample sizes of the synthetic datasets the same as the real dataset.

6.4.1.3 Classifiers

After training, we get five logistic regression classifiers: **C1-Real** that is a regular logistic regression model trained on the real Adult data, **C2-ACGAN** that is trained by ACGAN, **C3-FairGAN** that is trained on data generated by FairGAN with the assumption that fair data generation automatically achieves fair classification for any classification-based fairness, **C4-AD** that is a debiased classifier trained on the real Adult data through adversarial learning, **C5-FairGAN$^+$(DP)** that is trained by FairGAN$^+$ based on demographic parity, and **C6-FairGAN$^+$(EO)** that is trained by FairGAN$^+$ based on equality of odds. The evaluation of the classifiers is on the real dataset.

For each model, we train five times and report the means of evaluation results.

Table 6.1: Data fairness and utility of real and synthetic datasets

	Metric	D1-Real	D2-ACGAN	D3-FairGAN	D4-FairGAN$^+$(DP)	D5-FairGAN$^+$(EO)
Fairness	$RD(\mathcal{D})$	0.1989	0.0120	0.0411	0.0106	0.0116
	BER	0.1538	0.1964	0.3862	0.3867	0.3207
Utility	$dist(\mathbf{X}, Y)$	NA	0.0245	0.0233	0.0232	0.0239
	$dist(\mathbf{X}, Y, S)$		0.0204	0.0208	0.0212	0.0210

6.4.2 Fair Data Generation

We evaluate whether FairGAN and FairGAN$^+$ can generate fair data that satisfy statistical fairness and ϵ-fairness while learning the distribution of real data precisely.

6.4.2.1 Data Fairness

We adopt the risk difference $RD(\mathcal{D}) = P(Y = 1|S = 1) - P(Y = 1|S = 0)$ to compare the performance of different GAN models on statistical fairness. Table 6.1 shows the risk differences in the real and synthetic datasets. The risk difference in the Adult dataset (D1-Real) is 0.1989, which indicates discrimination against female. D2-ACGAN has low risk difference due to independent priors. D3-FairGAN also has low risk difference as result of the adversarial game between the generator and the discriminator D_2. D4-FairGAN$^+$(DP) and D5-FairGAN$^+$(EO) both have low risk difference as well.

We further evaluate the ϵ-fairness (disparate impact) by calculating the balanced error rates (BERs) shown in Equation 3.14. Note that we adopt a linear SVM to predict S and then calculate BER. Table 6.1 shows the BERs in the real and synthetic datasets. The BER in D1-Real is 0.1538, which means a linear SVM can predict S given $\mathbf{x}$ with high accuracy. Hence, there is disparate impact in the real dataset. The BER in D2-ACGAN is 0.1964$\pm$0.0033, which shows that ACGAN captures the disparate impact in the real dataset due to its unawareness of ϵ-fairness. On the contrary, the BERs in D3-FairGAN, D4-FairGAN$^+$(DP) and D5-FairGAN$^+$(EO) are 0.3862$\pm$0.0036, 0.3867$\pm$0.0049 and 0.3207$\pm$0.0121, respectively,

which indicates using the generated $\hat{\mathbf{x}}$ to predict the real S has much higher error rate. So the disparate impacts in FairGAN and FairGAN$^+$ are small.

6.4.2.2 Data Utility

We evaluate the closeness between each synthetic dataset and the real dataset by calculating the Euclidean distance of joint probabilities w/o the protected attribute S, i.e., $P(\mathbf{X}, Y)$ and $P(\mathbf{X}, Y, S)$. The Euclidean distance is calculated between the estimated probability mass functions on the sample space, where $dist(\mathbf{X}, Y) = ||P_{data}(\mathbf{X}, Y) - P_G(\mathbf{X}, Y)||_2$ and $dist(\mathbf{X}, Y, S) = ||P_{data}(\mathbf{X}, Y, S) - P_G(\mathbf{X}, Y, S)||_2$. A smaller distance indicates better closeness. In Table 6.1, all synthetic datasets have small distances to the real dataset for joint and conditional probabilities. The distances of FairGAN$^+$ is even smaller than FairGAN. We can observe that FairGAN$^+$ still achieves good data utility after satisfying fairness constraints. As the co-training of the generative model and the classifier improves G on data generation, FairGAN$^+$ has a more efficient trade-off between utility and fairness of data generation.

6.4.3 Fair Classification

We evaluate the classifiers that are trained in ACGAN and FairGAN$^+$ on the real dataset.

6.4.3.1 Classification Fairness

We first compare different classifiers including C5-FairGAN$^+$(DP) on demographic parity using risk difference $RD(\eta) = P(\eta(\mathbf{X}) = 1|S = 1) - P(\eta(\mathbf{X}) = 1|S = 0)$. Table 6.2 shows the risk difference of different classifiers. C1-Real has a risk difference of 0.1834, which

Table 6.2: Classification fairness and accuracy of different classifiers

		C1-Real	C2-ACGAN	C3-FairGAN	C4-AD	C5-FairGAN$^+$(DP)	C6-FairGAN$^+$(EO)
DP	$RD(\eta)$	0.1834	0.1119	0.0901	0.0760	0.0141	NA
EO	$DTPR$	0.1017	0.0854	0.1473	0.0388	NA	0.0312
	$DFPR$	0.0746	0.0395	0.0361	0.0184		0.0245
Accuracy		0.8448	0.8359	0.8256	0.7902	0.8178	0.8218

indicates a regular logistic regression model learns the risk difference in the real Adult data and makes unfair predictions. C2-ACGAN has a high risk difference 0.1119±0.0182, which is discriminative due to ACGAN's unawareness of demographic parity. C3-FairGAN has a lower risk difference 0.0901±0.0220 but still discriminative (above the threshold 0.05). This indicates that FairGAN's assumption on fair classification from fair data generation does not always hold. The risk difference is lower as the result of fair data but it is still not small enough to guarantee fairness. However, C4-AD also has a lower risk difference 0.0760±0.0058 but still discriminative (above the threshold 0.05). C5-FairGAN$^+$(DP) has a low risk difference 0.0141±0.0065. As C5-FairGAN$^+$(DP) learns to make predictions uncorrelated to S, FairGAN$^+$ based on demographic parity can achieve demographic parity.

Then we compare different classifiers including C6-FairGAN$^+$(EO) on equality of odds. Equality of odds is a more complex fairness notion based on classification. We use difference in true positive rates $DTPR = P(\eta(\mathbf{X}) = 1|Y = 1, S = 1) - P(\eta(\mathbf{X}) = 1|Y = 1, S = 0)$ and difference in false positive rates $DFPR = P(\eta(\mathbf{X}) = 1|Y = 0, S = 1) - P(\eta(\mathbf{X}) = 1|Y = 0, S = 0)$ to measure equality of odds. Table 6.2 shows the $DTPR$ and $DFPR$ of different classifiers. C1-Real has 0.1017 difference in TPR and 0.0746 difference in FPR, which indicates a regular logistic regression model without any fairness constraint makes predictions with inequality of odds. C2-ACGAN has 0.0854±0.0316 difference in TPR and 0.0395±0.0098 difference in FPR, which is smaller than C1-Real but still slightly discriminative. C3-FairGAN has 0.1473±0.0608 difference in TPR and 0.0361±0.0145 difference in

FPR, which is largely discriminative for subgroup with $Y = 1$. This also indicates that FairGAN's assumption on fair classification from fair data generation is wrong. Especially for advanced classification-based fairness notions, simply training on fair data has very limited effect on such fairness notions. Hence, it is more practical to achieve such classification-based fairness by directly applying constraints on the classifier. C4-AD has has 0.0388±0.0234 difference in TPR and 0.0184±0.0121 difference in FPR, which indicates AD is effective on achieving equality of odds. C6-FairGAN$^+$(EO) has 0.0312±0.0316 difference in TPR and 0.0245±0.0124 difference in FPR, which is similar to C4-AD. As C6-FairGAN$^+$(EO) learns to make predictions uncorrelated to S based on the real label Y, FairGAN$^+$ based on equality of odds can achieve equality of odds in the classifier.

6.4.3.2 Classification Accuracy

We evaluate the accuracy of different classifiers on the real Adult dataset. Table 6.2 shows the classification accuracy of different classifiers. The accuracy of C1-Real is 0.8448. The accuracy of C2-ACGAN is 0.8359±0.0017. The accuracy of C3-FairGAN is 0.8256±0.0021. However, these classifiers are unfair to the protected group. The accuracy of C4-AD is 0.7902±0.0043. The accuracies of C5-FairGAN$^+$(DP) and C6-FairGAN$^+$(EO) are 0.8178±0.0035 and 0.8218±0.0062, respectively. They are slightly lower than other classifiers, which indicates that AD and FairGAN$^+$ models have a trade-off between classification accuracy and fairness. In FairGAN$^+$, small utility loss is caused by playing the adversarial game with the third discriminator D_3 to achieve respective notions of classification fairness. We can observe that C5-FairGAN$^+$(DP) and C6-FairGAN$^+$(EO) have higher accuracy than C4-AD after satisfying the desired classification-based fairness. As the co-training of the generative model and the classifier improves η on classification, FairGAN$^+$ has a more efficient

trade-off between the utility and fairness of classification.

6.5 Summary

We first developed FairGAN to generate fair data, which is free from disparate treatment and disparate impact, while retaining high data utility. FairGAN consists of one generator and two discriminators. We then developed FairGAN$^+$ to (1) generate fair data, which is free from disparate treatment and disparate impact w.r.t. the real protected attribute, while retaining high data utility, and (2) release a fair classifier that satisfies classification-based fairness, such as demographic parity, equality of odds and equality of opportunity. FairGAN$^+$ consists of one generator, one classifier and three discriminators. The experimental results showed the effectiveness of FairGAN on fair data generation, the effectiveness of FairGAN$^+$ on both fair data generation and fair classification, and the better trade-off of FairGAN$^+$ between the utility and fairness when co-training a generative model and a classifier. The early version of this work is published at IEEE BigData 2018 [111] and IEEE BigData 2019 [112].

7 Achieving Causal Fairness through Generative Adversarial Networks

7.1 Introduction

In this work, we propose a causal fairness-aware generative adversarial network (CFGAN) for generating data that achieve various causal-based fairness criteria. Motivated by CausalGAN [83], we preserve the causal structure in the generator by arranging the neural network structure of the generator following a given causal graph. As a result, the generator can be considered as to simulate the underlying causal model of generating the observational data. (*It is worth noting that causal effects may not be estimated from observational data in certain situations, referred to as unidentifiable situations. The generator can be treated as simulating the true causal model only in identifiable situations.*) Then, in order to handle different fairness criteria, we adopt two generators for explicitly modeling the real world and the world after we perform some hypothetical interventions. The two generators differ in some aspects to reflect the effect of interventions, but are also synchronized in terms of sharing parameters to reflect the connections between the two worlds. Then we adopt two discriminators for achieving both the high data utility and causal-based fairness. Experiments using the real world dataset show that CFGAN can generate high quality fair data based on different criteria.

7.2 CFGAN

For general preliminaries of causal modeling and generative adversarial networks, please refer to Chapter 3.2.3 and Chapter 3.3, respectively.

To discuss the design of CFGAN, we first formulate our problem (Section 7.2.1), and

then discuss the overall framework (Section 7.2.2). The CFGAN based on different fairness criteria will be discussed in Sections 7.2.3, 7.2.4 and 7.2.5. For all types of causal effects, we simply assume they are identifiable.

7.2.1 Problem Statement

In this work, we follow the conventional notations in fairness-aware learning. We consider $\mathbf{V} = \{\mathbf{X}, Y, S\}$, where S denotes the sensitive variable, Y denotes the decision variable, and $\mathbf{X}$ denotes the set of all other variables (profile attributes). Given a causal graph $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ and a dataset with m samples $(\mathbf{x}, y, s) \sim P_{data} = P(\mathbf{V})$, the goal of CFGAN is to (1) generate new data $(\hat{\mathbf{x}}, \hat{y}, \hat{s})$ which preserves the distribution of all attributes in the real data and (2) ensure that in the generated data $\hat{S}$ has no discriminatory effect on $\hat{Y}$ in terms of various causal-based criteria. Note that we use the hatted variables to denote the fake data generated by the generator. For ease of discussion, we consider both S and Y as binary variables, where s^+ denotes $S = 1$ and s^- denotes $S = 0$. It's straightforward to extend this to the multi-categorical or numerical cases. In this work we mostly discuss the causal effect of a single variable S on another single variable Y. However, the model is capable to handle causal effects between multiple variables as well.

We consider causal fairness criteria based on total effect [29], direct discrimination [28], indirect discrimination [28], and counterfactual fairness [31], defined below.

Definition 13. There is no total effect in the data if $TE(s^+, s^-) = 0$.

Definition 14. There is no direct discrimination in the data if $SE_{\pi_d}(s^+, s^-) = 0$, where π_d is the path set that only contains the direct edge from S to Y, i.e., $S \to Y$.

Definition 15. Given a subset of attributes $\mathbf{R} \subseteq \mathbf{X}$ that cannot be objectively justified in

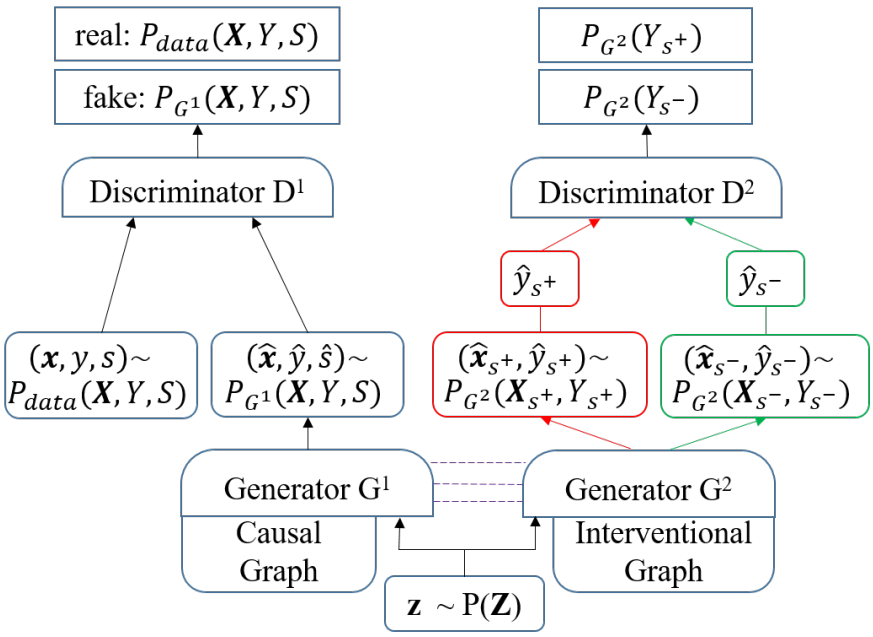


Figure 7.1: The structure of CFGAN

decision making, there is no indirect discrimination in the data if $SE_{\pi_i}(s^+, s^-) = 0$, where π_i is the set of causal paths from S to Y that pass through $\mathbf{R}$.

Definition 16. Given a subset of attributes $\mathbf{O} \subseteq \mathbf{X}$, counterfactual fairness is achieved in the data if $CE(s^+, s^-|\mathbf{o}) = 0$ under any context $\mathbf{O} = \mathbf{o}$.

7.2.2 Model Framework

We propose the CFGAN model which consists of two generators (G^1, G^2) and two discriminators (D^1, D^2). Figure 7.1 shows the framework of CFGAN.

As shown in Sections 3.2.3.2 and 7.2.1, in general, causal-based fairness criteria compare the intervention distributions of Y under two different interventions $do(S = s^+)$ and $do(S = s^-)$. To implement these criteria, CFGAN adopts two generators. One generator G^1 plays the role of original causal model $\mathcal{M}$ similar to CausalGAN, while the other generator G^2 explicitly plays the roles of different interventional models $\mathcal{M}_s$ based on the type

of causal effects. Generator G^1 aims to generate observational data whose distribution is close to the real observational distribution, and generator G^2 aims to generate interventional data that satisfy the criterion defined in Section 7.2.1. The two generators share the input noises and parameters to reflect the connections between the two causal models, and differ in connections of sub-neural networks to reflect the intervention. Then, CFGAN adopts two discriminators, where one discriminator D^1 tries to distinguish the generated data from the real data, and the other discriminator D^2 tries to distinguish the two intervention distributions under $do(S = s^+)$ and $do(S = s^-)$. Finally, generators and discriminators play the adversarial game to produce high quality fair data.

Next, we give the details in designing the generators and discriminators for different fairness criteria.

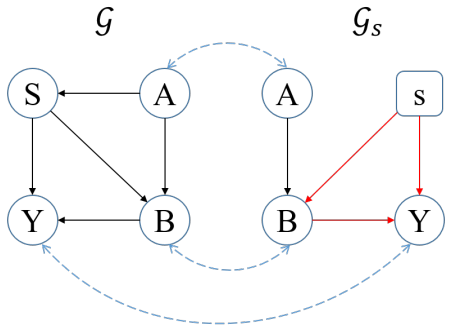
7.2.3 CFGAN based on Total Effect

We first show the CFGAN with no total effect (Definition 13).

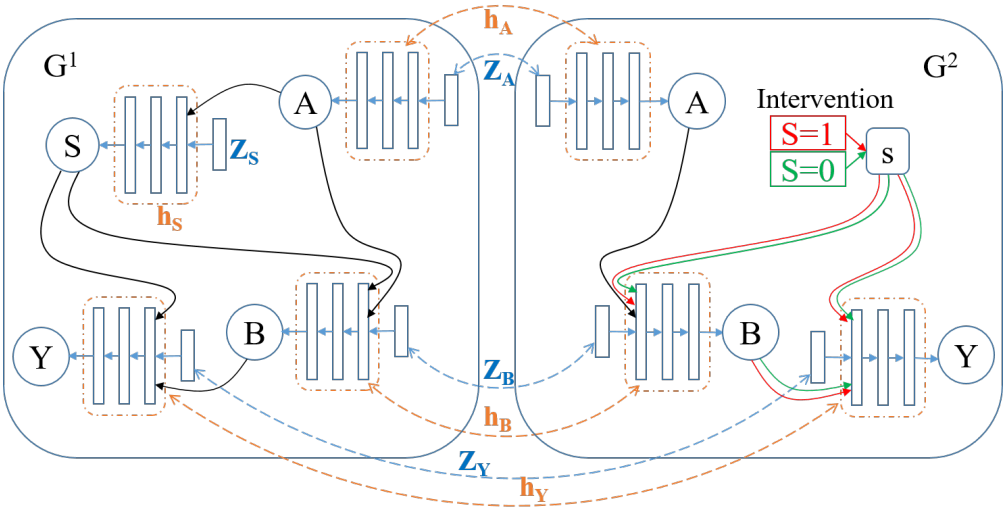
Generators. Generator G^1 is designed to agree with the causal graph $\mathcal{G} = (\mathbf{V}, \mathbf{E})$. It consists of $|\mathbf{V}|$ sub-neural networks, where each of them corresponds to a node in $\mathbf{V}$. All sub-neural networks are connected following the connections in $\mathcal{G}$. To be specific, each sub-neural network $G^1_{V_i}$ takes as input an independent noise vector $\mathbf{Z}_{V_i}$ as well as the output of any other sub-neural network $G^1_{V_j}$ if V_j is a parent of V_i in $\mathcal{G}$. Then, it outputs sample values of V_i, i.e., $\hat{v}_i$.

The other generator G^2 is designed to agree with the interventional graph $\mathcal{G}_s = (\mathbf{V}, \mathbf{E} \setminus \{V_j \rightarrow S\}_{V_j \in \mathbf{Pa}_S})$, where all incoming edges to S are deleted under intervention $do(S = s)$. The structure of G^2 is similar to that of G^1, except for that sub-neural network G^2_S is set as $G^2_S \equiv 1$ if $s = s^+$, and $G^2_S \equiv 0$ if $s = s^-$. The two generators G^1 and G^2

are synchronized by sharing the same set of parameters for each pair of corresponding sub-neural networks, i.e., $G^1_{V_i}$ and $G^2_{V_i}$ for each V_i except for S, as well as the same noise vectors $\mathbf{Z} = \mathbf{z}$. As a result, G^1 can generate samples from the observational distribution, and G^2 can generate samples from two interventional distributions, i.e., $(\hat{\mathbf{x}}, \hat{y}, \hat{s}) \sim P_{G^1}(\mathbf{X}, Y, S)$, $(\hat{\mathbf{x}}_{s^+}, \hat{y}_{s^+}) \sim P_{G^2}(\mathbf{X}_{s^+}, Y_{s^+})$, if $s = s^+$, $(\hat{\mathbf{x}}_{s^-}, \hat{y}_{s^-}) \sim P_{G^2}(\mathbf{X}_{s^-}, Y_{s^-})$, if $s = s^-$.



(a) Causal graph $\mathcal{G}$ and interventional graph $\mathcal{G}_s$



(b) Generators G^1 and G^2

Figure 7.2: An example of the generators G^1 and G^2 for CFGAN based on total effect. S is set to 1 or 0 to sample from the interventional distributions $P_{G^2}(A_{s^+}, B_{s^+}, Y_{s^+})$ (red) and $P_{G^2}(A_{s^-}, B_{s^-}, Y_{s^-})$ (green) respectively.

Consider an example in Figure 7.2 which involves 4 variables $\{A, S, B, Y\}$. Figure 7.2a shows the causal graph $\mathcal{G}$ and the interventional graph $\mathcal{G}_s$ under $do(S = s)$, where the

double headed arrows indicate the pair of nodes that share the same hidden variables and the function. Figure 7.2b shows the structures of the generators where G^1 agrees with $\mathcal{G}$ and G^2 agrees with $\mathcal{G}_s$. The double headed arrows indicate the sharing of noises and parameters of sub-neural networks. As shown, the edge from A to S is deleted in $\mathcal{G}_s$, which is also reflected in G^2. In addition, for each pair of nodes in the graphs, e.g., B in $\mathcal{G}$ and B in $\mathcal{G}_s$, the corresponding sub-neural networks are also synchronized, e.g., G^1_B and G^2_B.

Discriminators. Discriminator D^1 is designed to distinguish between the real observational data $(\mathbf{x}, y, s) \sim P_{data}(\mathbf{X}, Y, S)$ and the generated fake observational data $(\hat{\mathbf{x}}, \hat{y}, \hat{s}) \sim P_{G^1}(\mathbf{X}, Y, S)$. The other discriminator D^2 is designed to distinguish between the two interventional distributions $\hat{y}_{s^+} \sim P_{G^2}(Y_{s^+})$ and $\hat{y}_{s^-} \sim P_{G^2}(Y_{s^-})$.

Putting the generators and discriminators together, generator G^1 plays the adversarial game with the discriminator D^1, and generator G^2 plays the adversarial game with the discriminator D^2. The overall minimax game is described as:

$$\min_{G^1,G^2} \max_{D^1,D^2} J(G^1,G^2,D^1,D^2) = J_1(G^1,D^1) + \lambda J_2(G^2,D^2),$$

where

$$J_1(G^1, D^1) = \mathbb{E}_{(\mathbf{x},y,s)\sim P_{data}(\mathbf{X},Y,S)}[\log D^1(\mathbf{x}, y, s)] + \mathbb{E}_{(\hat{\mathbf{x}},\hat{y},\hat{s})\sim P_{G^1}(\mathbf{X},Y,S)}[1 - \log D^1(\hat{\mathbf{x}}, \hat{y}, \hat{s})],$$

$$J_2(G^2, D^2) = \mathbb{E}_{\hat{y}_{s^+}\sim P_{G^2}(Y_{s^+})}[\log D^2(\hat{y}_{s^+})] + \mathbb{E}_{\hat{y}_{s^-}\sim P_{G^2}(Y_{s^-})}[1 - \log D^2(\hat{y}_{s^-})],$$

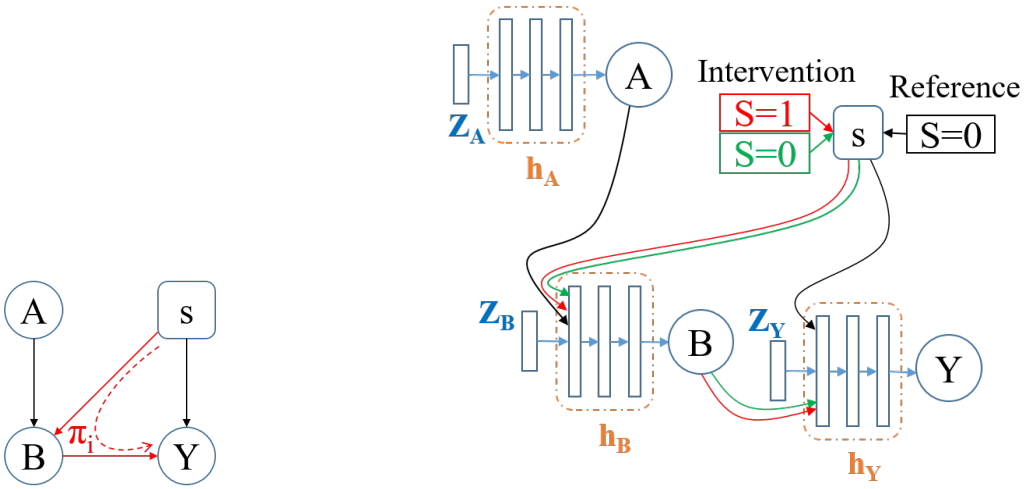
and λ is a hyperparameter which controls a trade-off between utility and fairness of data generation. The first value function J_1 aims to achieve $P_{G^1}(\mathbf{X}, Y, S) = P_{data}(\mathbf{X}, Y, S)$, i.e., to make the generated observational data indistinguishable from the real data. The second value function J_2 aims to achieve $P_{G^2}(Y_{s^+}) = P_{G^2}(Y_{s^-})$. Since Definition 13 requires

$TE(s^+, s^-) = 0$, or equivalently $P(Y_{s^+}) = P(Y_{s^-})$, J_2 actually makes the generated interventional data satisfy the fairness criterion. As G^1 and G^2 share the same sets of parameters, the observational data generated by G^1 can be considered as being generated by a causal model which is close to the real causal model and also satisfies the fairness criterion. Finally, the generated fair data can be released to public.

7.2.4 CFGAN based on Direct and Indirect Discrimination

Both direct and indirect discrimination are based on path-specific effects. In this section, we focus on the indirect discrimination criterion, and direct discrimination criterion can be achieved similarly. Given a path set π_i that contains the paths pass through unjustified attributes, Definition 15 requires that $SE_{\pi_i}(s^+, s^-) = 0$, or equivalently $P(Y_{s^+|_{\pi_i}}) = P(Y_{s^-|_{\pi_i}})$ with reference s^-.

The design of generator G^1 is similar to that in Section 7.2.3, but G^2 is different in that it needs to simulate the situation where the intervention is transferred along π_i only. To this end, we first similarly design the structure of G^2 to agree with the interventional graph $\mathcal{G}_s = (\mathbf{V}, \mathbf{E} \setminus \{V_j \to S\}_{V_j \in \mathbf{Pa}_S})$. Then, we consider two types of value settings for sub-neural network G^2_S: the reference setting and the interventional setting. For the reference setting, G^2_S is always set as $G^2_S \equiv 0$. For the interventional setting, G^2_S is set as $G^2_S \equiv 1$ if $s = s^+$ and $G^2_S \equiv 0$ if $s = s^-$. On the other hand, each of other sub-neural networks may output two types of sample values according to the value setting of G^2_S, referred to as the reference value and interventional value respectively. For a sub-neural network, if its corresponding node is not on any path in π_i, it always takes reference values as input and outputs reference values. However, for any other sub-neural network $G^2_{V_j}$ that is on at least one path in π_i, it may take both types of values as input and output both. Specifically, for any sub-neural network



(a) $\mathcal{G}_s$ and π_i (b) Generator G^2

Figure 7.3: An example of the generator G^2 for CFGAN based on indirect discrimination. S is set to 1 or 0 and the transmission is set only along $\pi = \{S \to B \to Y\}$ to sample from the interventional distributions $P_{G^2}(A_{s^+|_\pi}, B_{s^+|_\pi}, Y_{s^+|_\pi})$ (red) and $P_{G^2}(A_{s^-|_\pi}, B_{s^-|_\pi}, Y_{s^-|_\pi})$ (green) respectively. S is set to be 0 for the reference setting.

$G^2_{V_i}$ where V_i is a child of V_j, if edge $V_j \to V_i$ does not belong to any path in π_i, then $G^2_{V_j}$ will feed the reference output values to $G^2_{V_i}$. Otherwise, the interventional output values will be fed. As a result, the interventional distribution generated by G^2 simulates the situation of the path-specific effect, which we denote as $P_{G^2}(\mathbf{X}_{s|_\pi}, Y_{s|_\pi})$.

Consider an example (Figure 7.3) with the same causal graph in Figure 7.2. The interventional graph $\mathcal{G}_s$ and $\pi_i = \{S \to B \to Y\}$ is shown in Figure 7.3b, and generator G^2 is shown in Figure 7.3b. Since B is on the path in π_i, G^2_B takes interventional values of S as input and outputs interventional values to G^2_Y. On the other hand, G^2_Y takes interventional values from G^2_B and reference values from $G^2_S \equiv 0$ as input.

To achieve no indirect discrimination, discriminator D^2 is designed to distinguish between two interventional distributions $\hat{y}_{s^+|_{\pi_i}} \sim P_{G^2}(Y_{s^+|_{\pi_i}})$ and $\hat{y}_{s^-|_{\pi_i}} \sim P_{G^2}(Y_{s^-|_{\pi_i}})$. By playing the adversarial game with G^2, the corresponding value function J_2 aims to achieve $P_{G^2}(Y_{s^+|_{\pi_i}}) = P_{G^2}(Y_{s^-|_{\pi_i}})$. Similarly, since G^1 and G^2 share the parameters, the observational

data generated by G^1 can also be considered as satisfying the no indirect discrimination criterion.

7.2.5 CFGAN for Counterfactual Fairness

In counterfactual fairness, the intervention is performed conditioning on a subset of variables $\mathbf{O} = \mathbf{o}$. Thus, different from previous fairness criteria that concern the interventional model only, counterfactual fairness concerns the connection between the original causal model and the interventional model. We reflect this connection in CFGAN by building a direct dependency between the samples generated by G^1 and the samples generated by G^2. Specifically, the structures of G^1 and G^2 are similar to those in Section 7.2.3. However, for each noise vector $\mathbf{z}$, we first generate the observational sample by using G^1, and observe whether in the sample we have $\mathbf{O} = \mathbf{o}$. Only for those noise vectors with $\mathbf{O} = \mathbf{o}$ in the generated samples, we use them for generating interventional samples by using G^2. Thus, the interventional distribution generated by G^2 is conditioned on $\mathbf{O} = \mathbf{o}$, denoted by $P_{G^2}(\mathbf{X}_s, Y_s|\mathbf{o})$. Finally, discriminator D^2 is designed to distinguish between $\hat{y}_{s^+}|\mathbf{o} \sim P_{G^2}(Y_{s^+}|\mathbf{o})$ and $\hat{y}_{s^-}|\mathbf{o} \sim P_{G^2}(Y_{s^-}|\mathbf{o})$, producing the value function that aims to achieve $P_{G^2}(Y_{s^+}|\mathbf{o}) = P_{G^2}(Y_{s^-}|\mathbf{o})$.

7.3 Experiments

7.3.1 Experiment Setup

The dataset we use for evaluation is the UCI Adult income dataset[95]. It contains 65,123 samples with 11 variables. Following the setting in [28], we binarize each attribute to reduce the complexity for causal graph discovery. We treat *sex* as the sensitive variable S, *income* as the decision variable Y. The estimated causal graph is shown in Figure 7.4.

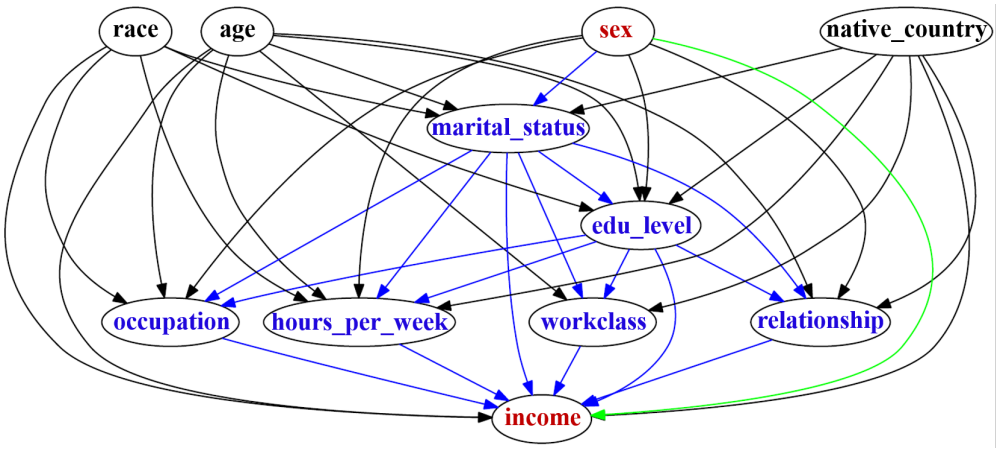


Figure 7.4: The causal graph for Adult dataset: the blue paths represent the indirect path set π_i.

We evaluate the performance of CFGAN in generating fair data for different types of causal fairness. The fairness threshold is 0.05, i.e., the effect should be in $[-0.05, 0.05]$ to be fair. We compare CFGAN with other data generating approaches for different fairness respectively as other approaches may only be able to achieve one or two types of fairness.

Specifically, we consider two baselines: (1) the original dataset; and (2) CausalGAN [83], which preserves the causal structure of the original data but is unaware of the fairness constraint. For total effect, we compare with FairGAN [111], which removes all information correlated to S in other attributes. For indirect discrimination (we skip the results for direct discrimination as the original dataset contains no direct discrimination), we further compare with PSE-DR [28], which is a direct/indirect discrimination removing algorithm by modifying the causal graph and generating new fair data based on the modified causal graph. For counterfactual fairness, we instead compare with A1 and A3 [31]. A1 generates fair decisions using a classifier that is built on non-descendants of S. A3 is similar to A1 but presupposes an additive noise model for estimating noise terms, which are then used for building the classifier. For both A1 and A3, we use SVM as the classifier for generating fair decisions.

	Total effect	Indirect discrimination	χ^2	Classifier accuracy			
				SVM	DT	LR	RF
Real data	0.1936	0.1754	0	0.8178	0.8177	0.8170	0.8178
CausalGAN	0.1721	0.1508	14482	0.8143	0.8136	0.8160	0.8137
FairGAN	0.0021	0.0133	41931	0.8088	0.8081	0.8136	0.8082
PSE-DR	NA	0.0243	12468	0.8073	0.8073	0.8128	0.8075
CFGAN (TE)	0.0102	NA	14566	0.8134	0.8126	0.8120	0.8127
CFGAN (SE)	NA	0.0030	19724	0.8037	0.8030	0.8103	0.8024

Table 7.1: The total effect and indirect discrimination of real and generated datasets

For data utility, we compute the χ^2 distance, where a smaller χ^2 indicates better utility. We also use the generated data to train classifiers and measure the accuracy. We evaluate 4 classifiers: support vector machine (SVM), decision tree (DT), logistic regression (LR) and random forest (RF).

7.3.2 Total Effect

We calculate the total effect for the original dataset and different generated datasets. The results are shown in Table 7.1. As can be seen, the original data has a total effect of 0.1936, and CausalGAN preserves similar total effect. FairGAN produces no total effect, but with the worst utility in terms of χ^2. This may be because FairGAN removes too much information due to its causal blindness. The generated data by CFGAN based on total effect (CFGAN (TE), $\lambda = 1$) produces no total effect, and also preserves good data utility.

7.3.3 Indirect Discrimination

For indirect discrimination, we consider all the paths passing through *marital_status* as π_i. The results are also shown in Table 7.1. Similar to total effect, CausalGAN preserves indirect discrimination close to the original data, and FairGAN removes indirect discrimination but causes the largest utility loss. On the other hand, PSE-DR and our method

	Counterfactual effect				χ^2	Classifier accuracy			
	$\mathbf{o}_1$	$\mathbf{o}_2$	$\mathbf{o}_3$	$\mathbf{o}_4$		SVM	DT	LR	RF
Real data	0.2023	0.1293	0.1266	0.1785	0	0.8178	0.8177	0.8170	0.8178
CausalGAN	0.1824	0.1155	0.1466	0.0959	14482	0.8143	0.8136	0.8160	0.8137
A1	0.0000	0.0000	0.0000	0.0000	17757	0.7615	0.7615	0.7615	0.7615
A3	0.2159	0.1127	0.1056	0.1860	12313	0.8159	0.8159	0.8159	0.8159
CFGAN (CE)	0.0209	0.0034	-0.0030	-0.0482	13904	0.8130	0.8123	0.8130	0.8115

Table 7.2: The counterfactual effect of real and generated datasets

(CFGAN (SE), $\lambda = 1$) can remove indirect discrimination and also have good data utility. We see that the two methods achieve comparable performance based on different techniques.

7.3.4 Counterfactual Fairness

For counterfactual fairness, we consider the observation of two attributes, i.e., $\mathbf{O} = \{race, native\_country\}$, which has 4 value combinations. Table 7.2 shows the results for all 4 subgroups. As can be seen, the original data and CausalGAN contain biases in terms of counterfactual fairness in all subgroups. A1 is counterfactual fair as expected since it is proved to be so in [31]. However, the data utility is bad especially in terms of classifier accuracy, since it only uses non-descendants of *sex* in labeling decisions. A3 cannot achieve counterfactual fairness, probably because its linear assumption does not fit the original data well. Finally, our method (CFGAN (CE), $\lambda = 1$) achieves both counterfactual fairness and good data utility.

7.3.5 Parameter Sensitivity

We evaluate the trade-off between utility and fairness when changing λ in the overall minimax game. A larger λ indicates a stronger enforcement on the fairness and compromise on utility. Figure 7.5 shows the results for total effect, where we get a fairly good trade-off between utility and fairness at $\lambda = 1$. We observe similar results for other fairness types.

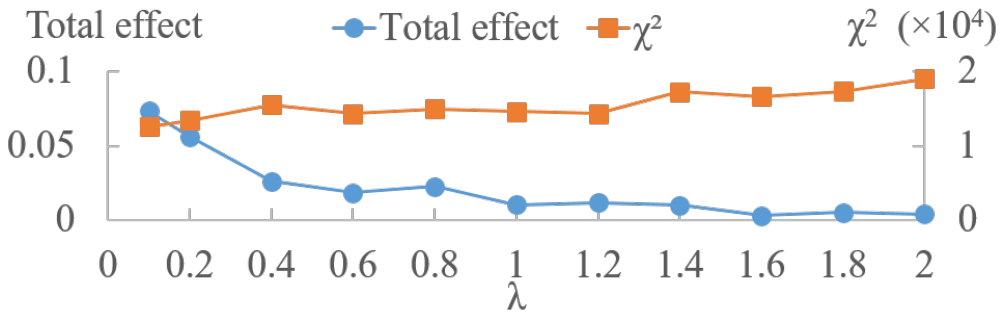


Figure 7.5: Total effect and χ^2 under different λ

7.4 Summary

We developed the causal fairness-aware generative adversarial networks (CFGAN) for generating high quality fair data. We considered various causation-based fairness criteria, including total effect, direct discrimination, indirect discrimination, and counterfactual fairness. CFGAN consists of two generators and two discriminators. The two generators aim to simulate the original causal model and the interventional model. This is achieved by arranging the neural network structure of the generators following the original causal graph and the interventional graph. Then, two discriminators are adopted for achieving both the high data utility and causal fairness. Experiments using the Adult dataset showed that CFGAN can achieve all types of fairness with relatively small utility loss. The early version of this work is published at IJCAI 2019 [113].

8 Achieving Differential Privacy and Fairness in Logistic Regression

8.1 Introduction

In this work, we focus on how to achieve both differential privacy and fairness in logistic regression – a widely-used classification model. It's challenging to achieve both requirements efficiently. The goal of differential privacy in a classification model is to make sure the classifier output is indistinguishable whether an individual record exists in the dataset or not. Its focus is on the individual level. The goal of fairness-aware learning is to make sure that predictions of the protected group are identical to those of the unprotected group, e.g., admission rate of female (as protected group) should be same to male (as unprotected group). Its focus is on the group level. There is no formal study on how to achieve both differential privacy and fairness in classification models.

We develop two methods to achieve differential privacy and fairness in logistic regression. Our simple method incorporates the decision boundary fairness constraint into the objective function of the logistic regression as a penalty term and then applies the functional mechanism to the whole constrained objective function to achieve differential privacy. The decision boundary fairness constraint of logistic regression is defined as the covariance between the users' protected attribute and the signed distance from the users' unprotected attribute vectors to the decision boundary, and can be further formulated as the signed distance between the centroids of the protected and unprotected groups. To achieve differential privacy, the functional mechanism brings randomness to the polynomial coefficients of the constrained objective function by introducing Laplace noise with zero mean. Because the penalty term contributes to the global sensitivity of objective function, this simple approach

may inject too much noise to the objective function, which reduces the utility of the built logistic regression model. We further develop an enhanced model that injects Laplace noise with shifted mean to the objective function of logistic regression. Our idea is based on the connection between ways of achieving differential privacy and fairness. We notice that both the fairness constraint and functional mechanism perturb the polynomial coefficients of the original objective function. Hence, we can combine them as a single term. In fact, the decision boundary fairness constraint of logistic regression can be treated as a shift of the polynomial coefficients by the signed distance between the centroids of the protected and unprotected groups. As a result, we add noise from a Laplace distribution with non-zero mean that is derived from the fairness constraint. In this way, the fairness constraint is not a penalty term, so we can use privacy budget more efficiently and add less noise.

8.2 Preliminaries

Let $D = \{\mathbf{X}, S, Y\}$ be a dataset with n tuples $t_1, t_2, \cdots, t_n$, where $\mathbf{X} = (X_1, X_2, \cdots, X_d)$ indicates d unprotected attributes; S denotes the protected attribute; Y is the decision. For each tuple $t_i = \{\mathbf{x}_i, s_i, y_i\}$, without loss of generality, we assume $x_{i(l)} \in [0, 1]$ for $l = (1, 2, \cdots, d)$, $s_i \in \{0, 1\}$, and $y_i \in \{0, 1\}$. Our objective is to build a classification model $\hat{y} = q(\mathbf{x}; \mathbf{w})$ with parameter $\mathbf{w}$ from D that achieves reasonable utility and meets both fairness and differential privacy requirements. To fit $\mathbf{w}$ to make accurate predictions, we have an objective function $f_D(\mathbf{w}) = \sum_{i=1}^{n} f(t_i; \mathbf{w})$ that takes t_i and $\mathbf{w}$ as input. The optimal model parameter $\bar{\mathbf{w}}$ is defined as: $\bar{\mathbf{w}} = \arg\min_{\mathbf{w}} \sum_{i=1}^{n} f(t_i; \mathbf{w})$.

For general preliminaries of differential privacy and fairness-aware machine learning, please refer to Chapter 3.1 and Chapter 3.2, respectively.

8.3 Differentially Private and Fair Logistic Regression

In this section, we first present a simple approach (PFLR) to achieve differentially private and fair logistic regression. Then we show it leads to an enhanced approach (PFLR*) that achieves the same level of differential privacy and fairness with more flexibility and less noise.

8.3.1 PFLR: A Simple Approach

One straightforward approach to achieve both differential privacy and fairness is to apply the functional mechanism to the objective function with fairness constraint $\tilde{f}_D(\mathbf{w})$. We consider the fairness constraint as a penalty term to the objective function. Then, the objective function ends up as:

$$\tilde{f}_D(\mathbf{w}) = f_D(\mathbf{w}) + \alpha|g_D(\mathbf{w}) - \tau|, \tag{8.1}$$

where α is a hyper-parameter to balance the trade-off between utility and fairness. We set $\alpha = 1, \tau = 0$ for ease of discussion. The theoretical analysis still holds if α and τ are set to other values.

For logistic regression, $f_D(\mathbf{w})$ is the objective function shown in Equation 3.9, and $g_D(\mathbf{w})$ indicates the decision boundary fairness constraint shown in Equation 3.17. We then rewrite $\tilde{f}_D(\mathbf{w})$ in the polynomial form based on Taylor expansion.

$$\tilde{f}_D(\mathbf{w}) = \left(\sum_{i=1}^{n} \sum_{j=0}^{2} \frac{f_1^{(j)}(0)}{j!} \left(\mathbf{x}_i^T \mathbf{w}\right)^j \right) - \left(\sum_{i=1}^{n} y_i \mathbf{x}_i^T \right) \mathbf{w} + \left| \sum_{i=1}^{n} (s_i - \bar{s}) \mathbf{x}_i^T \mathbf{w} \right|. \tag{8.2}$$

By transforming $\tilde{f}_D(\mathbf{w})$ in the form of Equation 3.7, we have $\tilde{\lambda}_{1t_i} = \lambda_{1t_i} + |(s_i - \bar{s})\mathbf{x}_i|$ and

$\tilde{\lambda}_{2t_i} = \lambda_{2t_i}$, where λ_{1t_i} and λ_{2t_i} are defined in Equations 3.11 and 3.12, respectively.

The global sensitivity of $\tilde{f}_D(\mathbf{w})$ is:

$$\begin{aligned}\Delta_{\tilde{f}} &= 2\max_t \Big(\Big| (\frac{f_1^{(1)}(0)}{1!} - y + |s - \bar{s}|) \sum_{l=1}^{d} x_{(l)} \Big| + \Big| \frac{f_1^{(2)}(0)}{2!} \sum_{l,m}^{d} x_{(l)} x_{(m)} \Big| \Big) \\ &\leq 2(\frac{3d}{2} + \frac{d^2}{8}) = \frac{d^2}{4} + 3d.\end{aligned} \tag{8.3}$$

The derived $\bar{\mathbf{w}}$ satisfies ε-differential privacy by applying Algorithm 5. Since the objective function contains the fairness constraint as a penalty term, the classification model also achieves fairness.

Algorithm 5 PFLR (Dataset D, objective function $f_D(\mathbf{w})$, fairness constraint $g_D(\mathbf{w})$, privacy budget ε)

1: Set $\tilde{f}_D(\mathbf{w})$ by Equation 8.1
2: Compute λ_{1t_i} and λ_{2t_i} by Equations 3.11 and 3.12
3: Set $\tilde{\lambda}_{1t_i} = \lambda_{1t_i} + |(s_i - \bar{s})\mathbf{x}_i|$ and $\tilde{\lambda}_{2t_i} = \lambda_{2t_i}$
4: Set $\Delta_{\tilde{f}}$ by Equation 8.3
5: Set $\bar{\lambda}_1 = \big(\sum_{i=1}^{n} \tilde{\lambda}_{1t_i}\big) + Lap(0, \frac{\Delta_{\tilde{f}}}{\varepsilon})$
6: Set $\bar{\lambda}_2 = \big(\sum_{i=1}^{n} \tilde{\lambda}_{2t_i}\big) + Lap(0, \frac{\Delta_{\tilde{f}}}{\varepsilon})$
7: Let $\bar{f}_D(\mathbf{w}) = \bar{\lambda}_1^T \mathbf{\Phi}_1 + \bar{\lambda}_2^T \mathbf{\Phi}_2$
8: Compute $\bar{\mathbf{w}} = \arg\min_{\mathbf{w}} \bar{f}_D(\mathbf{w})$
9: Return $\bar{\mathbf{w}}$

Theorem 8. Algorithm 5 satisfies ε-differential privacy.

Proof. Assume D and D' are two neighbouring datasets. Without loss of generality, D and D' differ in row t_r and t'_r. Δ is calculated by Equation 8.3. $\bar{f}(\mathbf{w})$ is the output of Line 7. We have

$$\frac{\Pr\{\bar{f}(\mathbf{w})|D\}}{\Pr\{\bar{f}(\mathbf{w})|D'\}} = \frac{\prod_{j=1}^{2}\prod_{\phi\in\mathbf{\Phi}_j}\exp\left(\frac{\varepsilon\left|\left|\sum_{t_i\in D}\tilde{\lambda}_{\phi t_i}-\bar{\lambda}_{\phi}\right|\right|_1}{\Delta_{\tilde{f}}}\right)}{\prod_{j=1}^{2}\prod_{\phi\in\mathbf{\Phi}_j}\exp\left(\frac{\varepsilon\left|\left|\sum_{t'_i\in D'}\tilde{\lambda}_{\phi t'_i}-\bar{\lambda}_{\phi}\right|\right|_1}{\Delta_{\tilde{f}}}\right)}$$

$$\leq \prod_{j=1}^{2}\prod_{\phi\in\mathbf{\Phi}_j}\exp\left(\frac{\varepsilon}{\Delta_{\tilde{f}}}\cdot\left|\left|\sum_{t_i\in D}\tilde{\lambda}_{\phi t_i}-\sum_{t'_i\in D'}\tilde{\lambda}_{\phi t'_i}\right|\right|_1\right) = \prod_{j=1}^{2}\prod_{\phi\in\mathbf{\Phi}_j}\exp\left(\frac{\varepsilon}{\Delta_{\tilde{f}}}\cdot\left|\left|\tilde{\lambda}_{\phi t_r}-\tilde{\lambda}_{\phi t'_r}\right|\right|_1\right) \tag{8.4}$$

$$= \exp\left(\frac{\varepsilon}{\Delta_{\tilde{f}}}\cdot\sum_{j=1}^{2}\sum_{\phi\in\mathbf{\Phi}_j}\left|\left|\tilde{\lambda}_{\phi t_r}-\tilde{\lambda}_{\phi t'_r}\right|\right|_1\right) \leq \exp\left(\frac{\varepsilon}{\Delta_{\tilde{f}}}\cdot 2\max_t\sum_{j=1}^{2}\sum_{\phi\in\mathbf{\Phi}_j}||\tilde{\lambda}_{\phi t}||_1\right) = \exp(\varepsilon).$$

□

8.3.2 PFLR*: An Enhanced Approach

We further enhance the simple approach by incorporating the fairness constraint into the Laplace noise.

In Equation 8.2, the fairness constraint $g_D(\mathbf{w}) = \sum_{i=1}^{n}(s_i - \bar{s})\mathbf{x}_i^T\mathbf{w}$ can be considered as shifting the first degree polynomial coefficients of $\mathbf{\Phi}_1$ in the objective function by $\sum_{i=1}^{n}(s_i - \bar{s})\mathbf{x}_i$. Since $\sum_{i=1}^{n}(s_i - \bar{s})\mathbf{x}_i$ is the signed distance between the centroids of the protected and unprotected groups, the derived $\bar{\mathbf{w}}$ based on the shifted coefficients ensures that the centroids of the protected and unprotected groups have the same distance to the decision boundary. Thus, the decision boundary fairness is achieved.

Meanwhile, the functional mechanism adds Laplace noise to inject randomness to the polynomial coefficients of the objective function. Because $\Pr\{\bar{f}(\mathbf{w})|D\}$ depends on the probability of the noise distribution, the designed Laplace noise provides the property of differential privacy.

Following this observation, instead of applying the fairness constraint as a penalty term to the objective function, we design a new functional mechanism that incorporates

fairness constraint into the Laplace noise. In particular, we shift the polynomial coefficients when adding Laplace noise. The shift is achieved by setting the mean of Laplace distribution. As $g_D(\mathbf{w})$ only affects $\mathbf{\Phi}_1$, we change the mean of Laplace distribution $\mu = \{\mu_{(l)}\}_{l=1}^{d}$ from 0 to $\sum_{i=1}^{n}(s_i - \bar{s})\mathbf{x}_i$ for the coefficients related with $\mathbf{\Phi}_1$, so it has the equivalent effect to the fairness constraint. Formally, we have $\mu_{(l)} = \sum_{i=1}^{n}(s_i - \bar{s})x_{(l)t_i}$. Because the fairness constraint is not a penalty term of the objective function, PFLR* has the same objective function as the regular logistic regression $f_D(\mathbf{w})$ (defined in Equation 3.10). The global sensitivity of PFLR* is $\Delta_f = d^2/4 + d$ as shown in Equation 3.13.

Note that, given a dataset, $\mu = \sum_{i=1}^{n}(s_i - \bar{s})\mathbf{x}_i$ is fixed. As we also access data when calculating μ, a small part of privacy budget ε_g is used to calculate μ in a differentially private manner by Laplace mechanism (Algorithm 6 Line 2). The sensitivity of μ is

$$\Delta_g = 2 \max_t \left| \sum_{l=1}^{d} (s_{t_r} - \bar{s}) x_{t_r(l)} \right| \leq 2d. \tag{8.5}$$

We formalize our new functional mechanism with fairness constraint as Algorithm 6. We split the total privacy budget ε into two parts ε_f and ε_g. We first calculate the differentially private μ with the privacy budget ε_g (Lines 1-2). Then, we introduce Laplace noise $Lap(\mu, \frac{\Delta_f}{\varepsilon_f})$ to the polynomial coefficients of the objective function with the privacy budget ε_f (Lines 3-7). Note that we only add the shifted Laplace noise to coefficients with $\mathbf{\Phi}_1$. Finally, we derive the optimized $\bar{\mathbf{w}}$ according to $\bar{f}_D(\mathbf{w})$ (Line 8). Next we show PFLR* achieves ε-differential privacy.

Theorem 9. Algorithm 6 satisfies ε-differential privacy.

Proof. Assume D and D' are two neighbouring datasets. Without loss of generality, D and D' differ in row t_r and t'_r. Δ_f is calculated by Equation 3.13. $\bar{f}(\mathbf{w})$ is the output of Line 7.

Adding Laplace noise with non-zero mean to coefficients still satisfies ε_f-differential privacy.

$$\frac{\Pr\{\bar{f}(\mathbf{w})|D\}}{\Pr\{\bar{f}(\mathbf{w})|D'\}} = \frac{\prod_{\phi\in\mathbf{\Phi}_1}\exp\left(\frac{\varepsilon_f\left|\left|\sum_{t_i\in D}\lambda_{\phi t_i}-\bar{\lambda}_\phi-\mu\right|\right|_1}{\Delta_f}\right)\prod_{\phi\in\mathbf{\Phi}_2}\exp\left(\frac{\varepsilon_f\left|\left|\sum_{t_i\in D}\lambda_{\phi t_i}-\bar{\lambda}_\phi\right|\right|_1}{\Delta_f}\right)}{\prod_{\phi\in\mathbf{\Phi}_1}\exp\left(\frac{\varepsilon_f\left|\left|\sum_{t'_i\in D'}\lambda_{\phi t'_i}-\bar{\lambda}_\phi-\mu\right|\right|_1}{\Delta_f}\right)\prod_{\phi\in\mathbf{\Phi}_2}\exp\left(\frac{\varepsilon_f\left|\left|\sum_{t'_i\in D'}\lambda_{\phi t'_i}-\bar{\lambda}_\phi\right|\right|_1}{\Delta_f}\right)}$$

$$\leq \prod_{\phi\in\mathbf{\Phi}_1}\exp\left(\frac{\varepsilon_f}{\Delta_f}\cdot\left|\left|\sum_{t_i\in D}\lambda_{\phi t_i}-\sum_{t'_i\in D'}\lambda_{\phi t'_i}\right|\right|_1\right)\cdot\prod_{\phi\in\mathbf{\Phi}_2}\exp\left(\frac{\varepsilon_f}{\Delta_f}\cdot\left|\left|\sum_{t_i\in D}\lambda_{\phi t_i}-\sum_{t'_i\in D'}\lambda_{\phi t'_i}\right|\right|_1\right) \quad (8.6)$$

$$= \prod_{j=1}^{2}\prod_{\phi\in\mathbf{\Phi}_j}\exp\left(\frac{\varepsilon_f}{\Delta_f}\cdot\left|\left|\lambda_{\phi t_r}-\lambda_{\phi t'_r}\right|\right|_1\right) = \exp\left(\frac{\varepsilon_f}{\Delta_f}\cdot\sum_{j=1}^{2}\sum_{\phi\in\mathbf{\Phi}_j}\left|\left|\lambda_{\phi t_r}-\lambda_{\phi t'_r}\right|\right|_1\right)$$

$$\leq \exp\left(\frac{\varepsilon_f}{\Delta_f}\cdot 2\max_t\sum_{j=1}^{J}\sum_{\phi\in\mathbf{\Phi}_j}||\lambda_{\phi t}||_1\right) = \exp(\varepsilon_f)$$

Using Laplace mechanism, Line 2 satisfies ε_g-differential privacy on calculating μ. Since $\varepsilon_f + \varepsilon_g = \varepsilon$, Algorithm 6 satisfies ε-differential privacy. □

Algorithm 6 PFLR* (Database D, objective function $f_D(\mathbf{w})$, fairness constraint $g_D(\mathbf{w})$, privacy budget $\varepsilon_f, \varepsilon_g$)

1: Set Δ_g by Equation 8.5
2: Calculate $\mu = \{\mu_{(l)}\}_{l=1}^{d}$ by $\mu_{(l)} = \sum_{i=1}^{n}(s_i - \bar{s})x_{(l)t_i} + Lap(0, \frac{\Delta_g}{\varepsilon_g})$
3: Compute λ_{1t_i} and λ_{2t_i} by Equation 3.11 and 3.12
4: Set Δ_f by Equation 3.13
5: Set $\bar{\lambda}_1 = \left(\sum_{i=1}^{n}\lambda_{1t_i}\right) + Lap(\mu, \frac{\Delta_f}{\varepsilon_f})$
6: Set $\bar{\lambda}_2 = \left(\sum_{i=1}^{n}\lambda_{2t_i}\right) + Lap(0, \frac{\Delta_f}{\varepsilon_f})$
7: Let $\bar{f}_D(\mathbf{w}) = \bar{\lambda}_1^T\mathbf{\Phi}_1 + \bar{\lambda}_2^T\mathbf{\Phi}_2$
8: Compute $\bar{\mathbf{w}} = \arg\min_{\mathbf{w}} \bar{f}_D(\mathbf{w})$
9: Return $\bar{\mathbf{w}}$

Comparison between PFLR and PFLR*. In PFLR, the fairness constraint term $g_D(\mathbf{w})$ contributes to the sensitivity of the polynomial coefficients of the objective function. In PFLR*, the fairness constraint is achieved by adding Laplace noise with non-zero mean value, so the sensitivity of the polynomial coefficient is not related to $g_D(\mathbf{w})$. PFLR* uses

separate budgets on objective function and fairness constraint, so it's more flexible to find good trade-offs among privacy, fairness and utility. The fairness constraint has a much smaller sensitivity than the objective function ($\Delta_g \ll \Delta_f$). Hence, we can allocate a relatively small privacy budget on calculating the fairness constraint with Laplace mechanism, the utility is still satisfactory. Then, more privacy budget can be used to the objective function, resulting in a smaller scale of noise. More concretely, we compare the amount of noise that is introduced to the two proposed approaches. The variance of $\lambda_{\phi t_i} \in \{\bar{\lambda}_1, \bar{\lambda}_2\}$ in PFLR is $2(\Delta_{\tilde{f}}/\varepsilon)^2$. Thus, the variance of total noise added in PFLR is $2(d^2 + d)(\Delta_{\tilde{f}}/\varepsilon)^2$. On the other hand, the variance of $\lambda_{\phi t_i} \in \bar{\lambda}_2$ in PFLR* is $Var(\bar{\lambda}_2) = 2(\Delta_f/\varepsilon_f)^2$. Because PFLR* injects Laplace noise to both $\bar{\lambda}_1$ and μ, based on the law of total variance, the variance of $\lambda_{\phi t_i} \in \bar{\lambda}_1$ in PFLR* is $Var(\bar{\lambda}_1) = \mathbb{E}[Var(\bar{\lambda}_1|\mu)] + Var(\mathbb{E}[\bar{\lambda}_1|\mu]) = 2(\Delta_f/\varepsilon_f)^2 + 0$. Thus, the variance of total noise added in PFLR* is $2(d^2 + d)(\Delta_f/\varepsilon_f)^2$. If we set $\varepsilon_f \geq (\Delta_f/\Delta_{\tilde{f}})\varepsilon$, PFLR* injects less noise.

8.4 Experiments

8.4.1 Experiment Setup

Dataset. We evaluate our methods on Adult [95] and Dutch [114]. For both datasets, we consider "Sex" as protected attribute and "Income" as decision. For unprotected attributes, we convert categorical attributes to one-hot vectors and normalize numerical attributes to $x \in [0, 1]$. The Adult dataset has 45222 records and 40 features. The Dutch dataset has 60420 records and 35 features.

Baselines. We compare the proposed differentially private and fair logistic regression models (**PFLR** and **PFLR***) with the following baselines: 1) a regular logistic regression

Table 8.1: Accuracy and risk difference (mean ± std.) of each method ($\varepsilon = 1$)

Method	Adult		Dutch	
	Accuracy	Risk Difference	Accuracy	Risk Difference
LR	0.8380 ± 0.0023	0.1577 ± 0.0064	0.8164 ± 0.0048	0.1747 ± 0.0033
PrivLR	0.7238 ± 0.0612	0.0502 ± 0.0581	0.6412 ± 0.0458	0.0739 ± 0.0574
FairLR	0.7739 ± 0.0521	0.0095 ± 0.0071	0.7673 ± 0.0064	0.0299 ± 0.0067
PFLR	0.7400 ± 0.0182	0.0213 ± 0.0258	0.6278 ± 0.0408	0.0206 ± 0.0204
PFLR*	0.7552 ± 0.0092	0.0053 ± 0.0070	0.6482 ± 0.0188	0.0430 ± 0.0265

model (**LR**); 2) a differentially private only LR using functional mechanism (**PrivLR**); 3) a fair only LR using Equation 3.17 as the fairness constraint (**FairLR**).

Metrics. We evaluate the performance of the proposed approaches and baselines on utility and fairness. We use *accuracy* as the utility metric and *risk difference (RD)* as the fairness metric. We run all models 10 times for each setting and report the mean and standard deviation of each metric.

8.4.2 Experimental Results

We first compare the performance of all five methods when $\varepsilon = 1$ ($\varepsilon_f = \varepsilon/2$ in PFLR*). As shown in Table 8.1, the regular logistic regression (LR) achieves the accuracy of 0.8380 on Adult and 0.8164 on Dutch, but it doesn't protect privacy nor achieves fairness ($RD = 0.1577$ and 0.1747, respectively). PrivLR has privacy protection but the accuracy decreases 11.42% on Adult and 17.52% on Dutch compared with LR as the result of the trade-off between privacy and utility. The risk difference of PrivLR is lower than LR, yet still larger than 0.05. The decrease of risk difference is mostly due to its low accuracy instead of fairness. FairLR achieves fairness ($RD = 0.0095$ on Adult and $RD = 0.0299$ on Dutch) as expected but it has no privacy guarantee. For PFLR and PFLR*, they both meet the privacy and fairness requirements. PFLR* has significantly higher accuracy than PFLR on both datasets (based on t-tests with p-values < 0.05). It indicates that PFLR* adds less

Table 8.2: Accuracy and risk difference with different privacy budgets ε

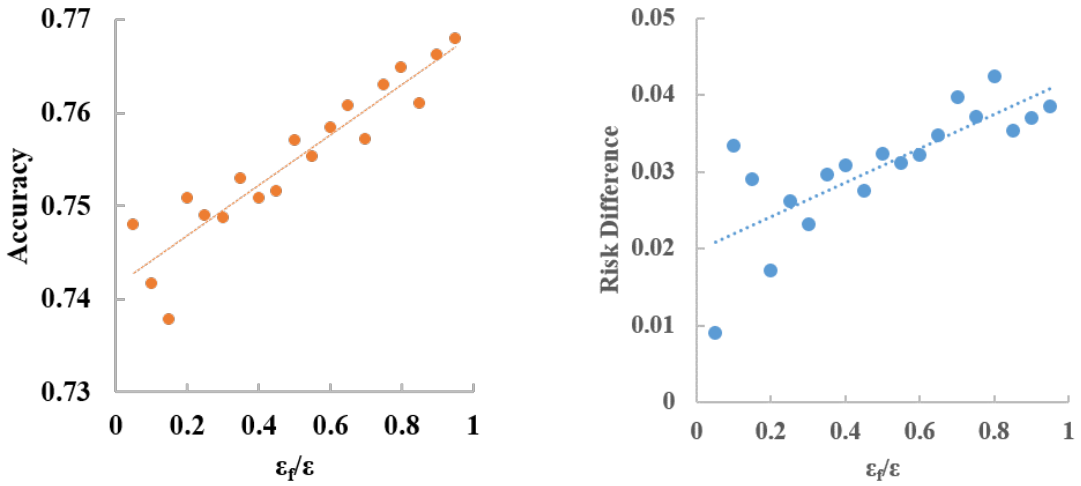
	ε	PrivLR		PFLR		PFLR*	
		Accuracy	Risk Difference	Accuracy	Risk Difference	Accuracy	Risk Difference
Adult	0.1	0.6263 ± 0.1480	0.0883 ± 0.0805	0.6172 ± 0.1187	0.0351 ± 0.0493	0.7491 ± 0.0040	0.0028 ± 0.0039
	1	0.7238 ± 0.0612	0.0502 ± 0.0581	0.7400 ± 0.0182	0.0213 ± 0.0258	0.7552 ± 0.0092	0.0053 ± 0.0070
	10	0.7270 ± 0.0877	0.1459 ± 0.0798	0.7631 ± 0.0155	0.0338 ± 0.0255	0.7632 ± 0.0093	0.0204 ± 0.0140
	100	0.8295 ± 0.0032	0.1624 ± 0.0116	0.7835 ± 0.0318	0.0332 ± 0.0243	0.7913 ± 0.0200	0.0234 ± 0.0189
Dutch	0.1	0.5241 ± 0.0396	0.0317 ± 0.0187	0.5069± 0.0459	0.0441 ± 0.0245	0.6158 ± 0.0239	0.0516 ± 0.0204
	1	0.6412 ± 0.0458	0.0739 ± 0.0574	0.6278 ± 0.0408	0.0206 ± 0.0204	0.6482 ± 0.0188	0.0460 ± 0.0265
	10	0.7239 ± 0.0902	0.1346 ± 0.0563	0.7282 ± 0.0493	0.0211 ± 0.0152	0.7080 ± 0.0329	0.0220 ± 0.0208
	100	0.8154 ± 0.0042	0.1687 ± 0.0054	0.7681 ± 0.0054	0.0301 ± 0.0085	0.7618 ± 0.0144	0.0250 ± 0.0128

noise to meet the same level of privacy guarantee.

Different Privacy Budgets. Table 8.2 shows how different settings of privacy budget ε affect our two methods and PrivLR. For PrivLR, its accuracy decreases dramatically when a stronger privacy requirement (smaller ε) is enforced. The risk difference of PrivLR decreases with the decrease of ε (the increase of noise). When ε is large, the accuracy is good and the risk difference is high. When ε is small, the accuracy is bad and the risk difference is low but with high variance.

For PFLR and PFLR*, when $\varepsilon = 0.1, 1$, PFLR* has significantly higher accuracy than PFLR on both Adult and Dutch (based on t-tests with p-values < 0.05). Especially, when $\varepsilon = 0.1$, PFLR's accuracy is only 0.6172 on Adult and 0.5069 on Dutch while PFLR*'s accuracy is 0.7491 on Adult and 0.6158 on Dutch. The accuracy of PFLR* is more consistent and relatively more resilient under different settings of privacy budget ε. When ε is relaxed to 100, PFLR and PFLR* have similar accuracy to FairLR (shown in Table 8.2). Overall, PFLR* outperforms PFLR especially when privacy budget is small.

Different Privacy Budget Splits $\varepsilon_f/\varepsilon$ for PFLR*. PFLR* splits the privacy budget ($\varepsilon = \varepsilon_f + \varepsilon_g$) into two parts: computing the fairness constraint (ε_g) and building the classification model (ε_f). Therefore, there is a trade-off between fairness and utility by controlling $\varepsilon_f/\varepsilon$. We further evaluate the performance of PFLR* in terms of accuracy and risk



(a) Accuracy (b) Risk difference

Figure 8.1: PFLR* with different privacy budget splits $\varepsilon_f/\varepsilon$ (Adult dataset, $\varepsilon = 10$)

difference with various privacy budget splits by ranging the values of $\varepsilon_f/\varepsilon$ from 0.05 to 0.95 with an interval as 0.05. In Fig. 8.1a, we observe that with the increase of ε_f, the accuracy increases accordingly. This is because when ε_f keeps increasing, the privacy budget for the objective function becomes large, which reduces noise added to the classification model. For the risk difference, as shown in Fig. 8.1b, when ε_f increases, the risk difference increases. This is because PFLR* injects more noise to compute the fairness constraint. However, the risk difference is consistently smaller than 0.05 while increasing ε_f. Hence, as the result of a small sensitivity Δ_g, the utility of fairness constraint is well preserved even with a small ε_g.

8.5 Summary

In this work, we developed two differentially private and fair logistic regression models, PFLR and PFLR*. PFLR is to apply the functional mechanism to the objective function with fairness constraint as a penalty term. Our enhanced model, PFLR*, takes advantage of the connection between ways of achieving differential privacy and fairness and adds the

Laplace noise with non-zero mean. The experimental results on two datasets demonstrate the effectiveness of two approaches and show the superiority of PFLR*. In this work, we consider logistic regression as the classification model and the covariance between decision boundary and the protected attribute as the fairness constraint. The early version of this work is published at WWW Workshop: FATES 2019 [73].

9 Removing Disparate Impact of Differentially Private Stochastic Gradient Descent on Model Accuracy

9.1 Introduction

In this work, we study the inequality in utility loss due to differential privacy, which compares the changes in prediction accuracy w.r.t. each group between the private model and the non-private model. Differential privacy guarantees that the query results or the released model cannot be exploited by attackers to derive whether one particular record is present or absent in the underlying dataset [1]. When we enforce differential privacy onto a regular non-private model, the model trades some utility off for privacy. On one hand, with the impact of differential privacy, the within-model unfairness in the private model may be different from the one in the non-private model [74, 73, 75]. On the other hand, differential privacy may introduce additional discriminative effect towards the protected group when we compare the private model with the non-private model. The utility loss between the private and non-private models w.r.t. each group, such as reduction in group accuracy, may be uneven. The intention of differential privacy should not be to introduce more accuracy loss on the protected group regardless of the level of within-model unfairness in the non-private model.

There are several empirical studies on the relationship between the utility loss due to differential privacy and groups with different represented sample sizes. Research in [76] shows that the accuracy of private models tends to decrease more on classes that already have lower accuracy in the original, non-private model. In their case, the direction of inequality in utility loss due to differential privacy is the same as the existing within-model

discrimination against the underrepresented group in the non-private model, i.e. "the poor become poorer". Research in [115] shows the similar observation that the contribution of rare training examples is hidden by random noise in differentially private stochastic gradient descent, and that random noise slows down the convergence of the learning algorithm. Research in [116] shows different observations when they analyze if the performance on emotion recognition is affected in an imbalanced way for the models trained to enhance privacy. They find that while the performance is affected differently for the subgroups, the effect is not consistent across multiple setups and datasets. In their case, there is no consistent direction of inequality in utility loss by differential privacy against the underrepresented group. Hence, the impact of differential privacy on group accuracy is more complicated than the observation in [76] (details in Section 9.3.1). It needs to be cautionary to conclude that differential privacy introduces more utility loss on the underrepresented group. The bottom line is that the objective of differential privacy is to protect individual's privacy instead of introducing unfairness in the form of inequality in utility loss w.r.t. groups. Though the privacy metric increases when a model is adversarially trained to enhance privacy, we need to ensure that the performance of the model on that dataset does not harm one subgroup more than the other.

In this work, we first analyze the inequality in utility loss by differential privacy. We use "cost of privacy" to refer to the utility loss between the private and non-private models as the result of the utility-privacy trade-off. We study the cost of privacy w.r.t. each group in comparison with the whole population and explain how the group sample size is related to the privacy impact on group accuracy along with other factors (Section 9.3.2). The difference in group sample sizes leads to the difference in average group gradient norms, which results in different group clipping biases under the uniform clipping bound. It costs less utility

trade-off to achieve the same level of differential privacy for the group with larger group sample size and/or smaller group clipping bias. In other words, the group with smaller group sample size and/or larger group clipping bias incurs more utility loss when the algorithm achieves the same level of differential privacy w.r.t. each group. Furthermore, we propose a modified DPSGD algorithm, called DPSGD-F, to remove the potential inequality in utility loss among groups (Section 9.4.2). DPSGD-F adjusts the contribution of samples in a group depending on the group clipping bias. For the group with smaller cost of privacy, their contribution is decreased and the achieved privacy w.r.t. their group is stronger; and vise versa. As a result, the final utility loss is the same for each group, i.e. differential privacy has no disparate impact on group accuracy in DPSGD-F. Our evaluation shows the effectiveness of our removal algorithm on achieving equal costs of privacy with satisfactory utility (Section 9.5).

9.2 Preliminaries

Let D be a dataset with n tuples $x_1, x_2, \cdots, x_n$, where each tuple x_i includes the information of a user i on d unprotected attributes $A_1, A_2, \cdots, A_d$, the protected attribute S, and the decision Y. Let D^k denote a subset of D with the set of tuples with $S = k$. Given an set of examples D, the non-private model outputs a classifier $\eta(a; w)$ with parameter w which minimizes the loss function $\mathcal{L}_D(w) = \frac{1}{n}\sum_{i=1}^{n} L_i(w)$. The optimal model parameter w^* is defined as: $w^* = \arg\min_w \frac{1}{n}\sum_{i=1}^{n} L_i(w)$. A differentially private algorithm outputs a classifier $\tilde{\eta}(a; \tilde{w})$ by selecting $\tilde{w}$ in a manner that satisfies differential privacy while keeping it close to the actual optimal w^*.

For general preliminaries of differential privacy and fairness-aware machine learning, please refer to Chapter 3.1 and Chapter 3.2, respectively.

9.3 Disparate Impact on Model Accuracy

In this section, we first discuss how differentially private learning, specifically DPSGD, causes inequality in utility loss through our preliminary observations. Then we study the cost of privacy with respect to each group in comparison with the whole population and explain how group sample size is related to the privacy impact on group accuracy along with other factors.

9.3.1 Preliminary Observations

To explain why DPSGD has disparate impact on model accuracy w.r.t. each group, [76] constructs an unbalanced MNIST dataset to study the effects of gradient clipping, noise addition, the size of the underrepresented group, batch size, length of training, and other hyperparameters. Training on the data of the underrepresented subgroups produces larger gradients, thus clipping reduces their learning rate and the influence of their data on the model. They also show random noise addition has the biggest impact on the underrepresented inputs. However, [116] reports inconsistent observations on whether differential privacy has negative discrimination towards the underrepresented group in terms of reduction in accuracy. To complement their observations, we use the unbalanced MNIST dataset used in [76] to reproduce their result, and we also use two benchmark census datasets (Adult and Dutch) in fair machine learning to study the inequality of utility loss due to differential privacy. We include the setup details in Section 9.5.1. Table 9.1 shows the model accuracy w.r.t. the total population, the majority group and the minority group for SGD and DPSGD on the MNIST, Adult and Dutch datasets.

On the unbalanced MNIST dataset, the minority group (class 8) has significantly

Table 9.1: Model accuracy w.r.t. the total population, the majority group and the minority group for SGD and DPSGD on the unbalanced MNIST ($\epsilon = 6.55, \delta = 10^{-6}$), the original Adult ($\epsilon = 3.1, \delta = 10^{-6}$) and the original Dutch ($\epsilon = 2.66, \delta = 10^{-6}$) datasets

Dataset	MNIST			Adult			Dutch		
Group	Total	Class 2	Class 8	Total	M	F	Total	M	F
Sample size	54649	5958	500	45222	30527	14695	60420	30273	30147
SGD	0.9855	0.9903	0.9292	0.8099	0.7610	0.9117	0.7879	0.8013	0.7744
DPSGD	0.8774	0.9196	0.2485	0.7507	0.6870	0.8836	0.6878	0.6479	0.7278
DPSGD vs. SGD	-0.1081	-0.0707	-0.6807	-0.0592	-0.0740	-0.0281	-0.1001	-0.1534	-0.0466

larger utility loss than the other groups in private model. DPSGD only results in -0.0707 decrease in accuracy on the well-represented classes but accuracy on the underrepresented class drops -0.6807, exhibiting a disparate impact on the underrepresented class. Figure 9.1 shows that the small sample size reduces both the convergence rate and the optimal utility of class 8 in DPSGD in comparison with the non-private SGD. The model is far from converging, yet clipping and noise addition do not let it move closer to the minimum of the loss function. Furthermore, the addition of noise, whose magnitude is similar to the update vector, prevents the clipped gradients of the underrepresented class from sufficiently updating the relevant parts of the model. Training with more epochs does not reduce this gap while exhausting the privacy budget. Differential privacy also slows down the convergence and degrades the utility for each group. Hence, DPSGD introduces negative discrimination against the minority group (which already has lower accuracy in the non-private SGD model) on the unbalanced MNIST dataset. This matches the observation in [76].

However, on the Adult and Dutch datasets, we have different observations from MNIST. The Adult dataset is an unbalanced dataset, where the female group is under-represented. Even though the male group is the majority group, it has lower accuracy in the SGD and more utility loss in DPSGD than the female group. The Dutch dataset is a balanced dataset, where the group sample sizes are similar for male and female. However,

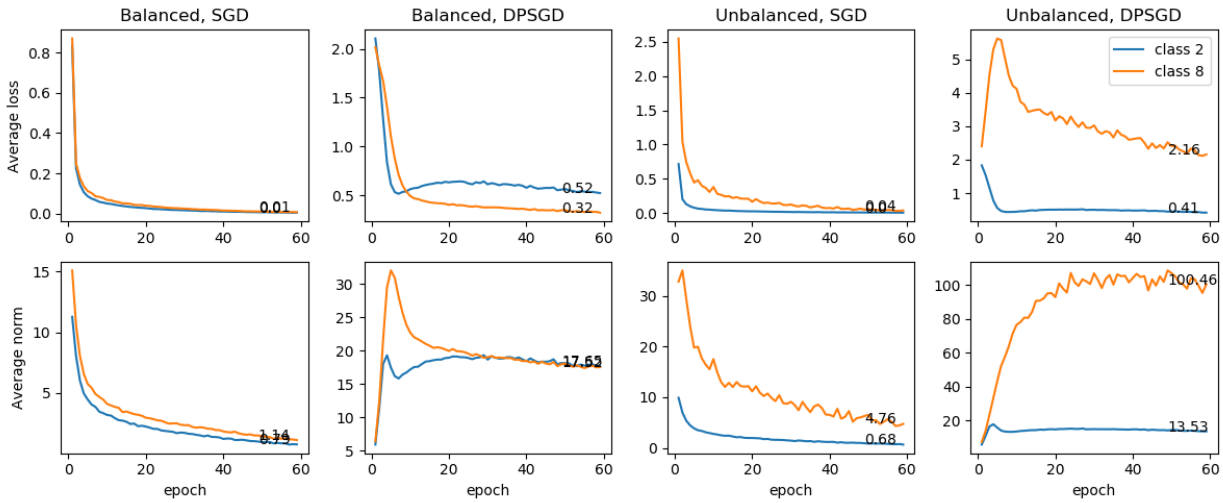


Figure 9.1: The average loss and the average gradient norm w.r.t. class 2 and 8 over epochs for SGD and DPSGD on the MNIST dataset (Balanced: $\epsilon = 6.23, \delta = 10^{-6}$, Unbalanced: $\epsilon = 6.55, \delta = 10^{-6}$)

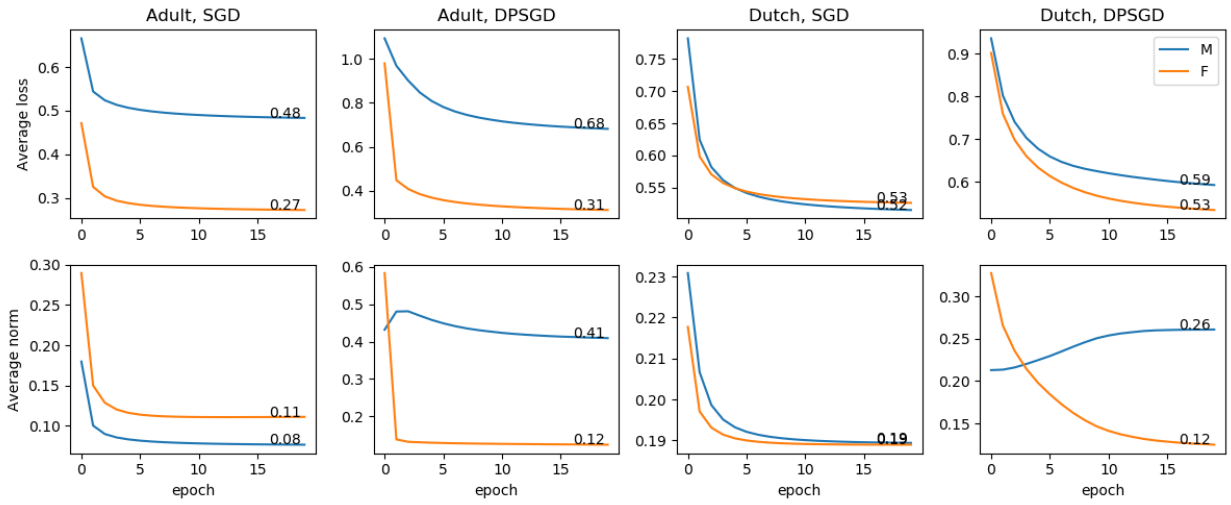


Figure 9.2: The average loss and the average gradient norm w.r.t. male and female groups over epochs for SGD and DPSGD on the original Adult and the original Dutch datasets (Adult: $\epsilon = 3.1, \delta = 10^{-6}$, Dutch: $\epsilon = 2.66, \delta = 10^{-6}$)

DPSGD introduces more negative discrimination against the male group and its direction (male group loses more accuracy due to DP) is even opposite to the direction of within-model discrimination (female group has less accuracy in SGD). Figure 9.2 shows that the average gradient norm is much higher for the male group in DPSGD on both datasets. It is not simply against the group with smaller sample size or lower accuracy in the SGD. Hence, differential privacy does not always introduce more accuracy loss to the minority group on the Adult and Dutch datasets. This matches the observation in [116].

From the preliminary observations, we learn that the disproportionate effect from differential privacy is not guaranteed towards the underrepresented group or the group with "poor" accuracy. Why does differential privacy cause inequality in utility loss w.r.t. each group? It may depend on more than just the represented sample size of each group: the classification model, the mechanism to achieve differential privacy, and the relative complexity of data distribution of each group subject to the model. One common observation across all settings is that the group with more utility loss has larger gradients and worse convergence. The underrepresented class 8 has average gradient norm of over 100 and bad utility in DPSGD. The male group has much larger average gradient norm than the female group in DPSGD on both Adult and Dutch datasets. It is important to address the larger gradients and worse convergence directly in order to mitigate inequality in utility loss.

9.3.2 Analysis on Cost of Privacy w.r.t. Each Group

In this section, we conduct analysis on the cost of privacy from the viewpoint of a single batch, where the utility loss is measured by the expected error of the estimated private gradient w.r.t. each group. For ease of discussion, our analysis follows [54] that investigates the bias-variance trade-off due to clipping in DPSGD with Laplace noise. Suppose that B_t

is a collection of b samples, $x_1, \cdots, x_b$. Each x_i corresponds to a sample and generates the gradient g_i. We would like to estimate the average gradient G_B from B_t in a differentially private way while minimizing the objective function.

We denote the original gradient before clipping $G_B = \frac{1}{b}\sum_{i=1}^{b} g_i$, the gradient after clipping but before adding noise $\bar{G}_B = \frac{1}{b}\sum_{i=1}^{b} \bar{g}_i$, and the gradient after clipping and adding noise $\tilde{G}_B = \frac{1}{b}(\sum_{i=1}^{b} \bar{g}_i + Lap(\frac{C}{\epsilon}))$. The expected error of the estimate $\tilde{G}_B$ consists of a variance term (due to the noise) and a bias term (due to the contribution limit):

$$\mathbb{E}|\tilde{G}_B - G_B| \leq \mathbb{E}|\tilde{G}_B - \bar{G}_B| + |\bar{G}_B - G_B| \leq \frac{1}{b}\frac{C}{\epsilon} + \frac{1}{b}\sum_{i=1}^{b} \max(0, |g_i| - C).$$

In the above derivation, we base the fact that the mean absolute deviation of a Laplace variable is equal to its scale parameter. We can find the optimal C by noting that the bound is convex with sub-derivative $\frac{1}{\epsilon} - |\{i : g_i > C\}|$, thus the minimum is achieved when C is equal to the $\lceil 1/\epsilon \rceil$th largest value in gradients.

The expected error is tight as we have

$$\mathbb{E}|\tilde{G}_B - G_B| \geq \frac{1}{2}\left[\frac{1}{b}\frac{C}{\epsilon} + \frac{1}{b}\sum_{i=1}^{b} max(0, |g_i| - C)\right].$$

In other words, the limit we should choose is just the $(1 - 1/b\epsilon)$-quantile of the gradients themselves.

For the same batch of samples, we derive the cost of privacy w.r.t. each group. Suppose the batch of samples B_t are from K groups and group k has sample size b^k. We have $G_B^k = \frac{1}{b^k}\sum_{i=1}^{b^k} g_i^k$ and $G_B = \frac{1}{b}\sum_{k=1}^{K} b^k G_B^k$. DPSGD bounds the sensitivity of gradient by clipping each sample's gradient with a clipping bound C. $\bar{G}_B^k = \frac{1}{b^k}\sum_{i=1}^{b^k} \bar{g}_i^k = \frac{1}{b^k}\sum_{i=1}^{b^k} g_i^k \times \min(1, \frac{C}{|g_i^k|})$.

Then, DPSGD adds Laplace noise on the sum of clipped gradients. $\tilde{G}_B^k = \frac{1}{b^k}(b^k \bar{G}_B^k + Lap(\frac{C}{\epsilon}))$. The expected error of the estimate $\tilde{G}_B^k$ also consists of a variance term (due to the noise) and a bias term (due to the contribution limit):

$$\begin{aligned}\mathbb{E}|\tilde{G}_B^k - G_B^k| &\leq \mathbb{E}|\tilde{G}_B^k - \bar{G}_B^k| + |\bar{G}_B^k - G_B^k| \\ &\leq \frac{1}{b^k}\frac{C}{\epsilon} + \frac{1}{b^k}\sum_i^{b^k} \max(0, |g_i^k| - C) = \frac{1}{b^k}\frac{C}{\epsilon} + \frac{1}{b^k}\sum_i^{m^k}(|g_i^k| - C),\end{aligned} \tag{9.1}$$

where $m^k = |\{i : |g_i^k| > C\}|$ is the number of examples that get clipped in group k. Similarly, we can get the tight bound w.r.t. each group k is $\mathbb{E}|\tilde{G}_B^k - G_B^k| \geq \frac{1}{2}\left[\frac{1}{b^k}\frac{C}{\epsilon} + \frac{1}{b^k}\sum_i^{b^k} \max(0, |g_i^k| - C)\right]$.

From Equation 9.1, we know the utility loss of group k, measured by the expected error of the estimated private gradient, is bounded by two terms, the bias $\frac{1}{b^k}\sum_i^{b^k} \max(0, |g_i^k| - C)$ due to contribution limit (depending on the size of gradients and the size of clipping bound) and the variance of the noise $\frac{1}{b^k}\frac{C}{\epsilon}$ (depending on the scale of the noise). Next, we discuss their separate impacts in DPSGD.

Given the clipping bound C, the bias due to clipping w.r.t. the group with large gradients is larger than the one w.r.t. the group with small gradients. Before clipping, the group with large gradients has large contribution in the total gradient G_B in SGD, but it is not the case in DPSGD. The direction of the total gradient after clipping $\bar{G}_B$ is closer to the direction of the gradient of the group with small bias (small gradients) in comparison with the direction of the total gradient before clipping G_B. Due to clipping, the contribution and convergence of the group with large gradients are reduced.

The added noise increases the variance of the model gradient, as it tries to hide the influence of a single record on the model. It slows down the convergence rate of the model. Because the noise scales $\frac{C}{\epsilon}$ and the sensitivity of clipped gradients C are the same for all

groups, the noisy gradients of all groups achieve the same level of differential privacy ϵ. The direction of the noise is random, i.e., it does not favor a particular group in expectation.

Overall in DPSGD, the group with large gradients has larger cost of privacy, i.e., they have more utility loss to achieve ϵ level of differential privacy under the same clipping bound C.

We can also consider the optimal choice of C which is $(1 - \frac{1}{b\epsilon})$-quantile for the whole batch. For each group, the optimal choice of C^k is $(1-\frac{1}{b^k\epsilon})$-quantile for group k. The distance between C and C^k is not the same for all groups, and C is closer to the choice of C^k for the group with small bias (small gradients).

Now we look back on the preliminary observations in Section 9.3.1. On MNIST, the group sample size affects the convergence rate for each group. The group with large sample size (the majority group, class 2) has larger contribution in the total gradient than the group with small sample size (the minority group, class 8), and therefore it leads to a relatively faster and better convergence. As the result, the gradients of the minority group are larger than the gradients of the majority group later on. In their case, the small sample size is the main cause of large gradient norm and large utility loss in class 8. On Adult and Dutch, the average bias due to clipping for each group is different because the distributions of gradients are quite different. The average gradient norm of the male group is larger than the average gradient norm of the female group, even though the male group is not underrepresented. As the result, the male group's contribution is limited due to clipping and it has larger utility loss in DPSGD. In there case, the group sample size is not the only reason to cause difference in the average gradient norm, and the other factors (e.g., the relative complexity of data distribution of each group subject to the model) out-weighs sample size, so the well-represented male group has larger utility loss.

This gives us an insight on the relation between differential privacy and the inequality in utility loss w.r.t. each group. The direct cause of the inequality is the large cost of privacy due to large average gradient norm (which can be caused by small group sample size along with other factors). In DPSGD, the clipping bound is selected uniformly for each group without consideration of the difference in clipping biases. As a result, the noise addition to achieve (ϵ, δ)-differential privacy on the learning model results in different utility-privacy trade-off for each group, where the underrepresented or the more complex group incurs a larger utility loss. After all, DPSGD is designed to protect individual's privacy with nice properties without consideration of its different impact towards each group. In order to avoid disparate utility loss among groups, we need to modify DPSGD such that each group needs to achieve different level of privacy to counter their difference in costs of privacy.

9.4 Removing Disparate Impact

Our objective is to build a learning algorithm that outputs a classifier $\tilde{\eta}(a; \tilde{w})$ with parameter $\tilde{w}$ that achieves differential privacy, equality of utility loss w.r.t each group, and good accuracy. Based on our preliminary observation and analysis on cost of privacy, we propose a heuristic removal algorithm to achieve equal utility loss w.r.t. each group, called DPSGD-F.

9.4.1 Equal Costs of Differential Privacy

In the within-model fairness, equal odds results in the equality of accuracy for different groups. Note that equal accuracy does not result in equal odds. As a trade-off for privacy, differential privacy results in accuracy loss on the model. However, different groups may incur different levels of accuracy loss. We use reduction in accuracy w.r.t. group k to measure utility

loss between the private model $\tilde{\eta}$ and the non-privacte model η, denoted by Δ^k. We define a new fairness notion called *equal costs of differential privacy*, which requires that the utility loss due to differential privacy is the same for all groups.

Definition 17. Equal costs of differential privacy Given a labeled dataset D, a classifier η and a differentially private classifier $\tilde{\eta}$, a differentially private mechanism satisfies equal equal costs of privacy if

$$\Delta^i(\tilde{\eta} - \eta) = \Delta^j(\tilde{\eta} - \eta),$$

where i, j are any two values of the protected attribute S.

9.4.2 Removal Algorithm

We propose a heuristic approach for differentially private SGD that removes disparate impact across different groups. The intuition of our heuristic approach is to balance the level of privacy w.r.t. each group based on their utility-privacy trade-off. Algorithm 7 shows the framework of our approach. Instead of uniformly clipping the gradients for all groups, we propose to do adaptive sensitive clipping where each group k gets its own clipping bound C^k. For the group with larger clipping bias (due to large gradients), we choose a larger clipping bound to balance their higher cost of privacy. The large gradients may be due to group sample size or other factors.

Based on our observation and analysis in the previous section, to balance the difference in costs of privacy for each group, we need to adjust the clipping bound C^k such that the contribution of each group is proportional to the size of their average gradient (Line 14 in Algorithm 7). Ideally, we would like to adjust the clipping bound based on the private estimate of the average gradient norm. However, the original gradient before clipping has

Algorithm 7 DPSGD-F (Dataset D, loss function $\mathcal{L}_D(w)$, learning rate r, batch size b, noise scales σ_1, σ_2, base clipping bound C_0)

1: **for** $t \in [T]$ **do**
2: Randomly sample a batch of samples B_t with $|B_t| = b$ from D
3: **for** each sample $x_i \in B_t$ **do**
4: $g_i = \nabla L_i(w_t)$
5: **end for**
6: **for** each group $k \in [K]$ **do**
7: $m^k = \left|\left\{i : |g_i^k| > C_0\right\}\right|$
8: $o^k = \left|\left\{i : |g_i^k| \leq C_0\right\}\right|$
9: **end for**
10: $\left\{\tilde{m}^k, \tilde{o}^k\right\}_{k\in[K]} = \left\{m^k, o^k\right\}_{k\in[K]} + N(0, \sigma_1^2\mathbf{I})$
11: $\tilde{m} = \sum_{k\in[K]} \tilde{m}^k$
12: **for** each group $k \in [K]$ **do**
13: $\tilde{b}^k = \tilde{m}^k + \tilde{o}^k$
14: $C^k = C_0 \times \left(1 + \frac{\tilde{m}^k/\tilde{b}^k}{\tilde{m}/b}\right)$
15: **end for**
16: **for** each sample $x_i \in B_t$ **do**
17: $\bar{g}_i = g_i \times \min\left(1, \frac{C^k}{|g_i|}\right)$
18: **end for**
19: $C = \max_k C^k$
20: $\tilde{G}_B = \frac{1}{b}\left(\sum_i \bar{g}_i + N(0, \sigma_2^2 C^2 \mathbf{I})\right)$
21: $\tilde{w}_{t+1} = \tilde{w}_t - r\tilde{G}_B$
22: **end for**
23: Return $\tilde{w}_T$ and accumulated privacy cost (ϵ, δ)

unbounded sensitivity. It would not be practical to get its private estimate. We need to construct a good approximate estimate of the relative size of the average gradient w.r.t. each group and it needs to have a small sensitivity for private estimation.

In our algorithm, we choose adaptive clipping bound C^k based on the m^k, where $m^k = |\{i : |g_i^k| > C_0\}|$. To avoid the influence of group sample size, we use the fraction of $\frac{m^k}{b^k}$ that represents the fraction of samples in the group with gradients larger than C_0. The relative ratio of $\frac{m^k}{b^k}$ and $\frac{m}{b}$ can approximately represent the relative size of the average gradient (Line 14). To choose the clipping bound C^k for group k in a differentially private way, we get the private $\tilde{m}^k, \tilde{b}^k$ and $\tilde{m}$ from the collection $\{m^k, o^k\}_{k\in[K]}$ (Line 6-13). The

collection $\{m^k, o^k\}_{k\in[K]}$ has sensitivity of 1, which is much smaller than the sensitivity of the actual gradients when we estimate the relative size of the average gradient.

After the adaptive clipping, the sensitivity of the clipped gradient of group k is $C^k = C_0 \times (1 + \frac{\tilde{m}^k/\tilde{b}^k}{\tilde{m}/b})$. The sensitivity of the clipped gradient of the total population would be $\max_k C^k$ as the worst case in the total population needs to be considered.

Note that in Algorithm 7 we have two steps of adding noise in each iteration t. We first use a relatively large noise scale σ_1 (small privacy budget) to get a private collection $\{\tilde{m}^k, \tilde{o}^k\}_{k\in[K]}$ (Line 10). Then we use a relatively small noise scale σ_2 to perturb the gradients (Line 20). The composition theorem (Theorem 4) is applied when we compute the accumulated privacy cost (ϵ, δ) from moments accountant (Line 23). Because $\sigma_1 > \sigma_2$, only a small fraction of privacy budget is spent on getting C^k.

For the total population, Algorithm 7 still satisfies (ϵ, δ)-differential privacy as it accounts for the worst clipping bound $\max_k C^k$. On the group level, each group achieves different levels of privacy depending on their utility-privacy trade-off.

With our modified DPSGD algorithm, we continue our discussion in Secion 9.3.2. In the case of [76], the difference in gradient norms is primarily decided by group sample size. Consider a majority group s^+ and a minority group s^-. In Algorithm 1, each group achieves the same level of privacy, but the underrepresented group s^- has higher privacy cost (utility loss). In Algorithm 7, we choose a higher clipping bound C^- for the underrepresented group. Because the noise scale is $\frac{C}{\epsilon} = \frac{C^-}{\epsilon}$ and the sensitivity of clipped gradients for the underrepresented group is C^-, the noisy gradient w.r.t. the underrepresented group achieves ϵ-differential privacy. The well-represented group s^+ has a smaller cost of privacy, so we choose a lower clipping bound C^+. Because the noise scale is $\frac{C}{\epsilon} = \frac{C^-}{\epsilon}$ and the sensitivity of clipped gradients for the underrepresented group is C^+, the noisy gradient w.r.t. the

underrepresented group then achieves $(\frac{C^+}{C^-}\epsilon)$-differential privacy. Two groups have different clipping bounds C^+, C^- and the same noise addition based on $C = \max(C^+, C^-)$ (same ϵ but different relative scales w.r.t. their group sensitivities). Hence, when we enforce the same level of utility loss for groups with different sample sizes, the well-represented group achieves stronger privacy (smaller than ϵ) than the underrepresented group. In the case of Adult/Dutch, the male group has larger gradients regardless of the sample size. The group with smaller gradients based on model and data distribution has smaller cost of privacy. Algorithm 7 can adjust the clipping bound for each group. As a result, the group with smaller gradients achieves stronger level of privacy. Eventually, they can have similar clipping bias to the ones in Algorithm 1.

9.4.3 Baseline

There is no previous work on how to achieve equal utility loss in DPSGD. For experimental evaluation, we also present a naïve baseline algorithm based on reweighting (shown as Algorithm 8) in this section, since reweighting is a common way to mitigating biases in machine learning. The naïve algorithm considers group sample size as the main cause of disproportional impact in DPSGD and adjusts sample contribution of each group to mitigate the impact of sample size.

For the group with larger group sample size, we reweight the sample contribution with $\theta^k \propto \frac{1}{\tilde{b}^k}$ instead of using uniform weight of 1 for all groups, where $\tilde{b}^k$ is privately estimated (Line 6 in Algorithm 8). Note that G_B in Algorithm 1 is estimated based on uniform weight of each sample regardless of their group membership. The sensitivity for group k is $C^k = C_0 \times \theta^k$. The sensitivity for the total population would be $C^0 \times \max_k \theta^k$. The result also matches the idea that we limit the sample contribution of the group with

Algorithm 8 Naïve (Dataset D, loss function $\mathcal{L}_D(w)$, learning rate r, batch size b, noise scales σ_1, σ_2, base clipping bound C_0)

for $t \in [T]$ **do**
  Randomly sample a batch of samples B_t with $|B_t| = b$ from D
  for each sample $x_i \in B_t$ **do**
    $g_i = \nabla L_i(w_t)$
  end for
  $\left\{\tilde{b}^k\right\}_{k\in[K]} = \left\{b^k\right\}_{k\in[K]} + N(0, \sigma_1^2\mathbf{I})$
  for each group $k \in [K]$ **do**
    $\theta^k = 1 \times \frac{b/K}{\tilde{b}^k}$
  end for
  for each sample $x_i \in B_t$ **do**
    $\bar{g}_i = \theta^k \times g_i \times \min\left(1, \frac{C_0}{|g_i|}\right)$
  end for
  $C = C_0 \times \max_k \theta^k$
  $\tilde{G}_B = \frac{1}{b}\left(\sum_i \bar{g}_i + N(0, \sigma_2^2 C^2 \mathbf{I})\right)$
  $\tilde{w}_{t+1} = \tilde{w}_t - r\tilde{G}_B$
end for
Return $\tilde{w}_T$ and accumulated privacy cost (ϵ, δ)

smaller cost of privacy to achieve stronger privacy level w.r.t. the group. However, Naïve only considers the group sample size. As we know from previous observation and analysis, the factors that affect the gradient norm and bias due to clipping are more complex than just the group sample size. We will compare with this Naïve approach as a baseline in our experiments.

9.5 Experiments

9.5.1 Experiment Setup

9.5.1.1 Datasets

We use MNIST dataset and replicate the setting in [76]. The original MNIST dataset is a balanced dataset with 60,000 training samples and each class has about 6,000 samples.

Class 8 has the most false negatives, hence we choose it as the artificially underrepresented group (reducing the number of training samples from 5,851 to 500) in the unbalanced MNIST dataset. We compare the underrepresented class 8 with the well-represented class 2 that shares fewest false negatives with the class 8 and therefore can be considered independent. The testing dataset has 10,000 testing samples with about 1,000 for each class.

We also use two census datasets, Adult [95] and Dutch [114]. For both datasets, we consider "Sex" as the protected attribute and "Income" as decision. For unprotected attributes, we convert categorical attributes to one-hot vectors and normalize numerical attributes to $[0, 1]$ range. After preprocessing, we have 40 unprotected attributes for Adult and 35 unprotected attributes for Dutch. The original Adult dataset has 45,222 samples (30,527 males and 14,695 females). We sample a balanced Adult dataset with 14,000 males and 14,000 females. The original Dutch dataset is close to balanced with 30,273 males and 30,147 females. We sample an unbalanced Dutch dataset with 30,000 males and 10,000 females. In all settings, we split the census datasets into 80% training data and 20% testing data.

9.5.1.2 Model

For the MNIST dataset, we use a neural network with 2 convolutional layers and 2 linear layers with 431K parameters in total. We use learning rate $r = 0.01$, batch size $b = 256$, and the number of training epochs is 60.

For the census datasets, we use a logistic regression model with regularization parameter 0.01. We use learning rate $r = 1/\sqrt{T}$, batch size $b = 256$, and the number of training epochs is 20.

9.5.1.3 Baseline

We compare our proposed method DPSGD-F (Algorithm 7) with the original DPSGD (Algorithm 1) and the Naïve approach (Algorithm 8). For each setting, the learning parameters are the same. We set C_0, σ_2 in DPSGD-F and Naïve equal to C, σ in DPSGD, respectively. We set $\sigma_1 = 10\sigma_2$. For the MNIST dataset, we set noise scale $\sigma = 0.8$, clipping bound $C = 1$, and $\delta = 10^{-6}$. For the census datasets, we set noise scale $\sigma = 1$, clipping bound $C = 0.5$, and $\delta = 10^{-6}$. The accumulated privacy budget ϵ for each setting is computed using the privacy moments accounting method [38]. Because we set $\sigma_1 = 10\sigma_2$, most of ϵ is spent on gradients from σ_2. Only about 0.01 budget is from σ_1. To compare DPSGD-F and Naïve with DPSGD under the same privacy budget, the algorithm runs a few less iterations than DPSGD in the last epoch, where the total number of iterations $T = \text{epochs} \times n/b$ in SGD and DPSGD. For DPSGD-F and Naïve, T is 22 and 19 less on the balanced and unbalanced MNIST datasets, respectively; 5 and 11 less on the balanced and unbalanced Adult datasets, respectively; 17 and 9 less on the balanced and unbalanced Dutch datasets, respectively. These differences are very small in proportion to T. All DP models are compared with the non-private SGD when we measure the utility loss due to differential privacy.

9.5.1.4 Metric

We use the test data to measure the model utility and fairness. Based on Section 17, we use reduction in model accuracy for each group between the private SGD and the non-private SGD (Δ^k) as the metric to measure the cost of differential privacy w.r.t. each group. The difference between the costs on groups ($|\Delta^i - \Delta^j|$ for any i, j) measures the level of inequality in utility loss due to differential privacy. If the costs for all groups are independent of the protected attribute ($|\Delta^i - \Delta^j| \leq \tau$, $\tau = 0.05$ used in the paper), we

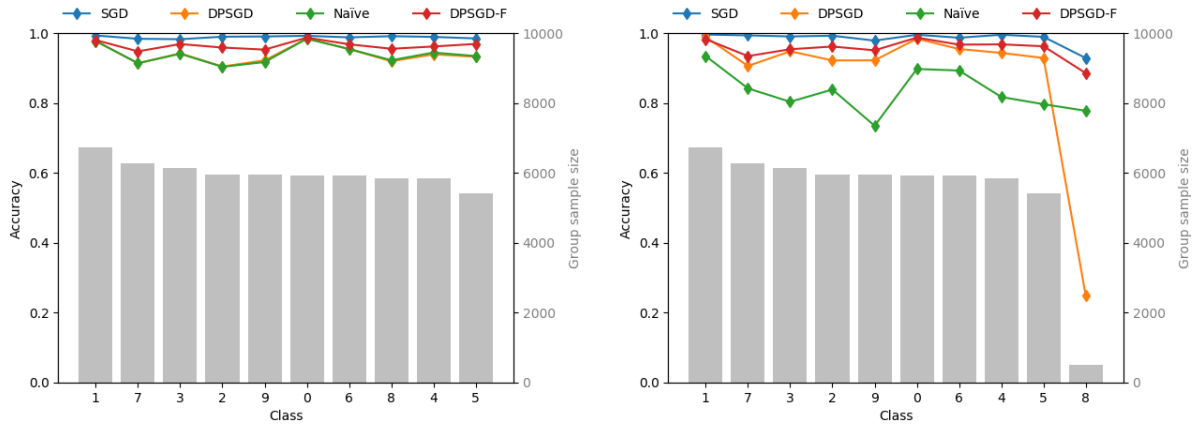
Table 9.2: Model accuracy w.r.t. class 2 and 8 on the MNIST dataset (Balanced: $\epsilon = 6.23, \delta = 10^{-6}$, Unbalanced: $\epsilon = 6.55, \delta = 10^{-6}$)

Dataset	Balanced			Unbalanced		
Group	Total	Class 2	Class 8	Total	Class 2	Class 8
Sample size	60000	5958	5851	54649	5958	500
SGD	0.9892	0.9932	0.9917	0.9855	0.9903	0.9292
DPSGD vs. SGD	-0.0494	-0.0853	-0.0719	-0.1081	-0.0707	-0.6807
Naïve vs. SGD	-0.0491	-0.0891	-0.0687	-0.1500	-0.1512	-0.1510
DPSGD-F vs. SGD	-0.0236	-0.0339	-0.0359	-0.0293	-0.0281	-0.0432

consider the private SGD has equal reduction in model accuracy w.r.t. each group, i.e. the private SGD achieves equal cost of differential privacy. We also report the average loss and average gradient norm to show the convergence w.r.t. each group during training.

9.5.2 MNIST Dataset

Table 9.2 shows the model accuracy w.r.t. class 2 and 8 on the balanced and unbalanced MNIST datasets. On the balanced dataset, each private or non-private model achieves similar accuracy across all groups. When we artificially reduce the sample size of class 8, class 8 becomes the minority group in the unbalanced dataset. The non-private SGD model converges to 0.9292 accuracy on class 8 vs. 0.9903 accuracy on class 2. The DPSGD model causes -0.6807 accuracy loss on class 8 vs. -0.0707 on class 2, which exhibits a significant disparate impact on the underrepresented class. The Naïve approach achieves -0.1510 accuracy loss on class 8 vs. -0.1512 on class 2, which achieves equal costs of privacy. Our DPSGD-F algorithm has -0.0432 accuracy loss on class 8 vs. -0.0281 on class 2, which also achieves equal costs of privacy. The total model accuracy also drops less for DPSGD-F (-0.0293) than the original DPSGD (-0.1081). Figure 9.3 shows the model accuracy w.r.t. all classes on the MNIST dataset. The difference between DPSGD and DPSGD-F is small and consistent across all classes.



(a) Balanced: $\epsilon = 6.23, \delta = 10^{-6}$ (b) Unbalanced: $\epsilon = 6.55, \delta = 10^{-6}$

Figure 9.3: Model accuracy w.r.t. each class for SGD, DPSGD, Naïve and DPSGD-F on the MNIST dataset

Table 9.3: The average loss and the average gradient norm w.r.t. groups at the last training epoch on the unbalanced MNIST ($\epsilon = 6.55, \delta = 10^{-6}$), the unbalanced Adult ($\epsilon = 3.1, \delta = 10^{-6}$) and the unbalanced Dutch ($\epsilon = 3.29, \delta = 10^{-6}$) datasets

	Average loss						Average gradient norm					
Dataset	MNIST		Adult		Dutch		MNIST		Adult		Dutch	
Group	Class 2	Class 8	M	F	M	F	Class 2	Class 8	M	F	M	F
SGD	0.04	0.04	0.48	0.27	0.49	0.58	0.68	4.76	0.08	0.11	0.12	0.30
DPSGD	0.41	2.16	0.68	0.31	0.52	0.83	13.53	100.46	0.41	0.12	0.11	0.52
Naïve	3.08	1.89	0.71	0.32	0.58	0.55	0.83	0.76	0.43	0.13	0.23	0.17
DPSGD-F	0.20	0.42	0.50	0.27	0.48	0.61	1.45	2.53	0.12	0.08	0.09	0.35

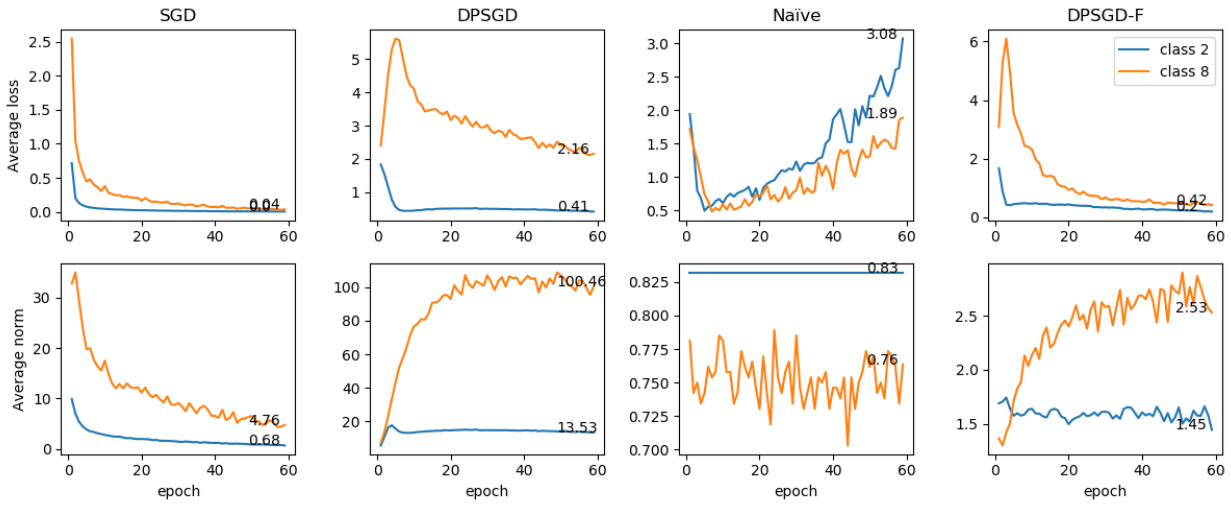


Figure 9.4: The average loss and the average gradient norm w.r.t. class 2 and 8 over epochs for SGD, DPSGD, Naïve and DPSGD-F on the unbalanced MNIST dataset ($\epsilon = 6.55, \delta = 10^{-6}$)

Table 9.3 shows the average loss and average gradient norm w.r.t. class 2 and 8 for SGD and different DP models at the last training epoch. In DPSGD, the average gradient norm for class 8 is over 100 and the average loss for class 8 is 2.16. Whereas, in DPSGD-F, the average gradient norm for class 8 is only 2.53 and the average loss for class 8 is only 0.42. The convergence loss and the gradient norm for class 8 are much closer to the ones for class 2 in DPSGD-F. Figure 9.4 shows the convergence trend during training. The trend in DPSGD-F is the closest to the trend in SGD among all DP models. It shows our adjusted clipping bound helps to achieve the same group utility loss regardless of the group sample size.

Figure 9.5 shows how our adaptive clipping bound changes over epochs in DPSGD-F. Because class 8 has larger clipping bias due to its underrepresented group sample size, DPSGD-F gives class 8 a higher clipping bound to increase its sample contribution in the total gradient. The maximal C^k is close to 3.

To show that the fair performance of DPSGD-F is not caused by increasing clipping

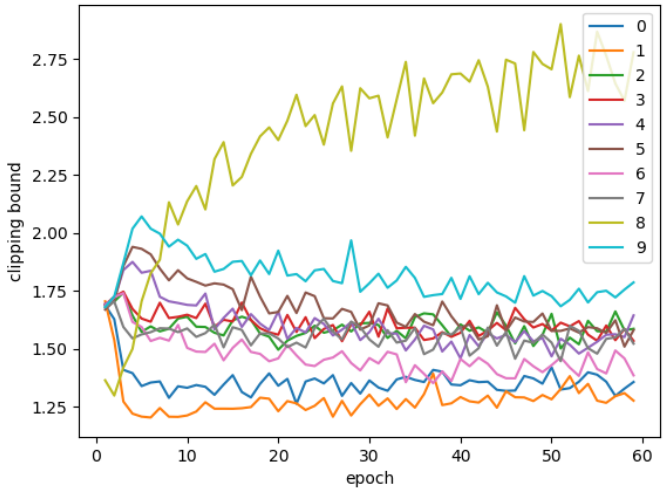


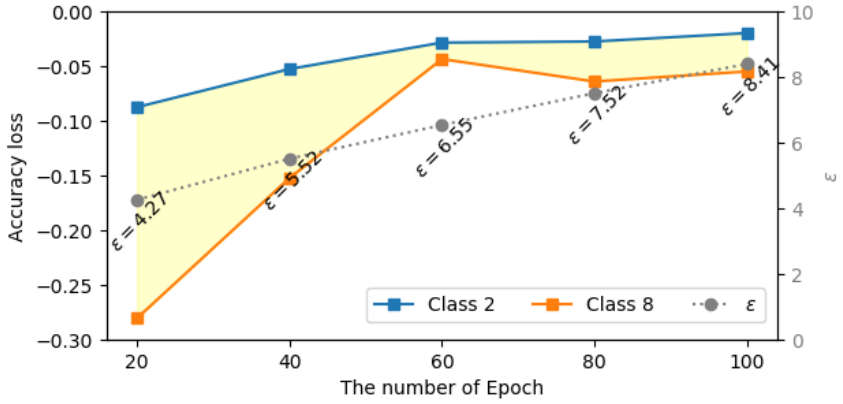
Figure 9.5: The clipping bound C^k w.r.t. each class over epochs for DPSGD-F on the unbalanced MNIST dataset ($\epsilon = 6.55, \delta = 10^{-6}$)

Table 9.4: Model accuracy w.r.t. class 2 and 8 for different uniform clipping bound ($C = 1, 2, 3, 4, 5$) in DPSGD vs. adaptive clipping bound ($C_0 = 1$) in DPSGD-F on the unbalanced MNIST dataset ($\epsilon = 6.55, \delta = 10^{-6}$)

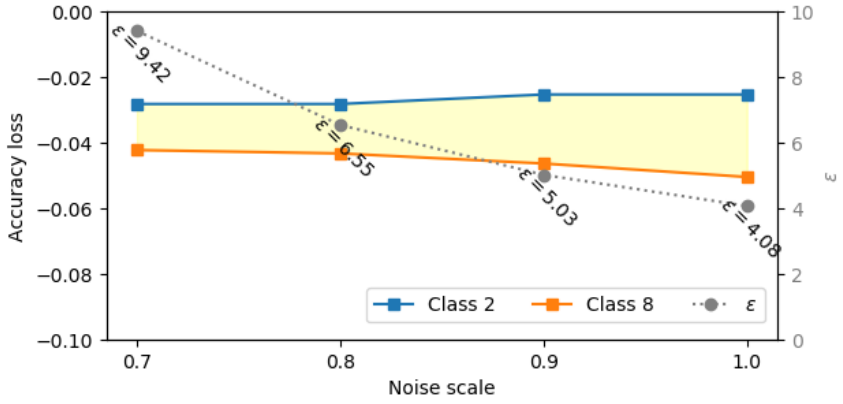
Group	Total	Class 2	Class 8
Sample size	54649	5958	500
SGD	0.9855	0.9903	0.9292
DPSGD ($C = 1$) vs. SGD	-0.1081	-0.0707	-0.6807
DPSGD ($C = 2$) vs. SGD	-0.0587	-0.0426	-0.3286
DPSGD ($C = 3$) vs. SGD	-0.0390	-0.0232	-0.2013
DPSGD ($C = 4$) vs. SGD	-0.0286	-0.0194	-0.1376
DPSGD ($C = 5$) vs. SGD	-0.0240	-0.0145	-0.1099
DPSGD-F ($C_0 = 1$) vs. SGD	-0.0293	-0.0281	-0.0432

bound uniformly, we run the original DPSGD with increasing clipping bound from $C = 1$ to $C = 5$. Table 9.4 shows the level of inequality in utility loss for different clipping bound in DPSGD vs. the adaptive clipping bound in DPSGD-F. Even though increasing clipping bound C in DPSGD can improve the accuracy on class 8, there is still significant difference between the accuracy loss on class 8 (-0.1099 when $C = 5$) and the accuracy loss on class 2 (-0.0145 when $C = 5$). This is because the utility-privacy trade-offs are different for the minority group and the majority group under the same clipping bound. So the inequality

in utility loss cannot be removed by simply increasing the clipping bound in DPSGD. On the contrary, DPSGD-F achieves equal costs of privacy by adjusting the clipping bound for each group according to the utility-privacy trade-off. The group with smaller cost of privacy achieves a stronger level of privacy as a result of adaptive clipping.



(a) Accuracy loss vs. the number of epochs



(b) Accuracy loss vs. noise scale

Figure 9.6: The accuracy loss on class 2 and 8 (DPSGD-F vs. SGD) with different ϵ on the MNIST dataset

We also evaluate the effectiveness of DPSGD-F over different privacy costs. There are two factors, the number of epochs and the noise scale, affecting the accumulated privacy cost ϵ. Figure 9.6a shows the group accuracy loss over different accumulated ϵ by altering the number of epochs while setting other parameters the same as default. With the number

of epochs increasing, the privacy cost ϵ increases, and the difference between the accuracy losses of class 2 and 8 decreases. From 60 epochs on, the difference is below the threshold τ, i.e. DPSGD-F achieves equal costs of privacy. Figure 9.6b shows the group accuracy loss over different accumulated ϵ by altering the noise scale while setting others parameters the same as default. With the noise scale increasing, the privacy cost ϵ decreases, and the difference between the accuracy losses of class 2 and 8 slightly increases, yet the difference is consistently below the threshold τ, i.e. DPSGD-F achieves equal costs of privacy. It suggests that it is better to enforce stronger privacy by increasing the noise scale than prematurely terminate training.

9.5.3 Adult and Dutch Datasets

Table 9.5 shows the model accuracy w.r.t. male and female on the balanced and unbalanced Adult and Dutch datasets. The clipping biases for both census datasets are not primarily decided by group sample size. We observe disparate impact on DPSGD in comparison to SGD against the male group, even though the male group is not underrepresented. The Naïve approach does not work at all to achieve equal costs of privacy in this case, as the importance of group sample size is not as much as in the MNIST dataset. There are still other factors that affect the gradient norm and the clipping bias w.r.t. each group. DPSGD-F can achieve similar accuracy loss for male and female in all four settings. It shows the effectiveness of our approach.

Table 9.3 shows the average loss and average gradient norm w.r.t. male and female for SGD and different DP models at the last training epoch. On the unbalanced Adult dataset, the average gradient norm in DPSGD for male is 5 times of the one in SGD and the average loss in DPSGD for male is 50% more than the one in SGD. Whereas, in DPSGD-F, the

Table 9.5: Model accuracy w.r.t. the total population and each group on the Adult and Dutch datasets (Balanced Adult (sampled): $\epsilon = 3.99, \delta = 10^{-6}$, Unbalanced Adult (original): $\epsilon = 3.1, \delta = 10^{-6}$, Balanced Dutch (original): $\epsilon = 2.66, \delta = 10^{-6}$, Unbalanced Dutch (sampled): $\epsilon = 3.29, \delta = 10^{-6}$)

Dataset	Balanced Adult			Unbalanced Adult			Balanced Dutch			Unbalanced Dutch		
Group	Total	M	F	Total	M	F	Total	M	F	Total	M	F
Sample size	28000	14000	14000	45222	30527	14695	60420	30273	30147	40000	30000	10000
SGD	0.824	0.748	0.899	0.809	0.761	0.911	0.787	0.801	0.774	0.802	0.834	0.706
DPSGD vs. SGD	-0.036	-0.054	-0.019	-0.059	-0.074	-0.028	-0.100	-0.153	-0.046	-0.124	-0.086	-0.240
Naïve vs. SGD	-0.036	-0.054	-0.019	-0.059	-0.074	-0.028	-0.100	-0.155	-0.046	-0.101	-0.142	0.025
DPSGD-F vs. SGD	-0.009	-0.014	-0.005	-0.025	-0.029	-0.016	-0.013	-0.016	-0.009	-0.032	-0.028	-0.044

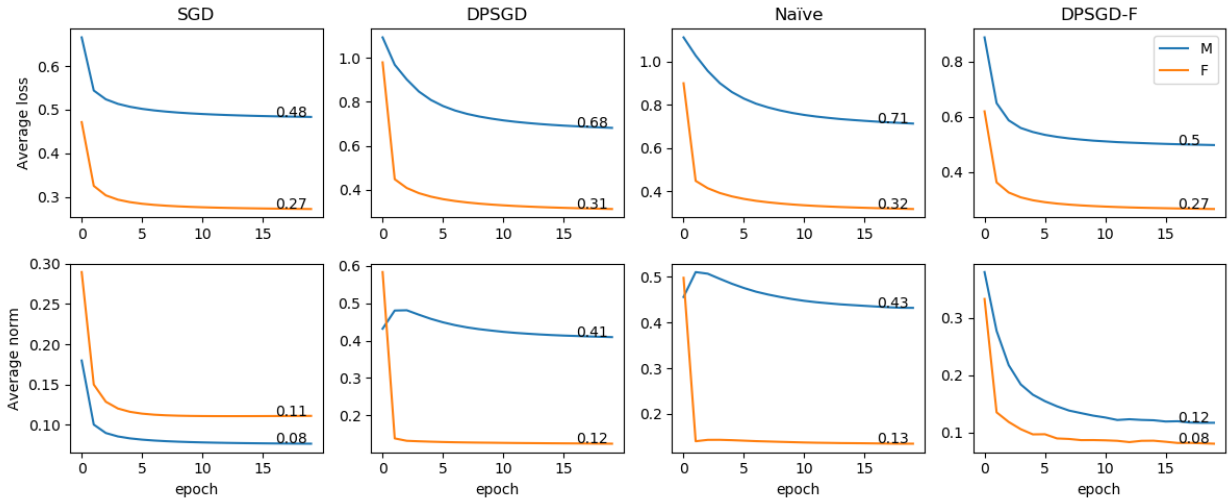


Figure 9.7: The average loss and the average gradient norm w.r.t. each group over epochs for SGD, DPSGD, Naïve and DPSGD-F on the unbalanced Adult dataset ($\epsilon = 3.1, \delta = 10^{-6}$)

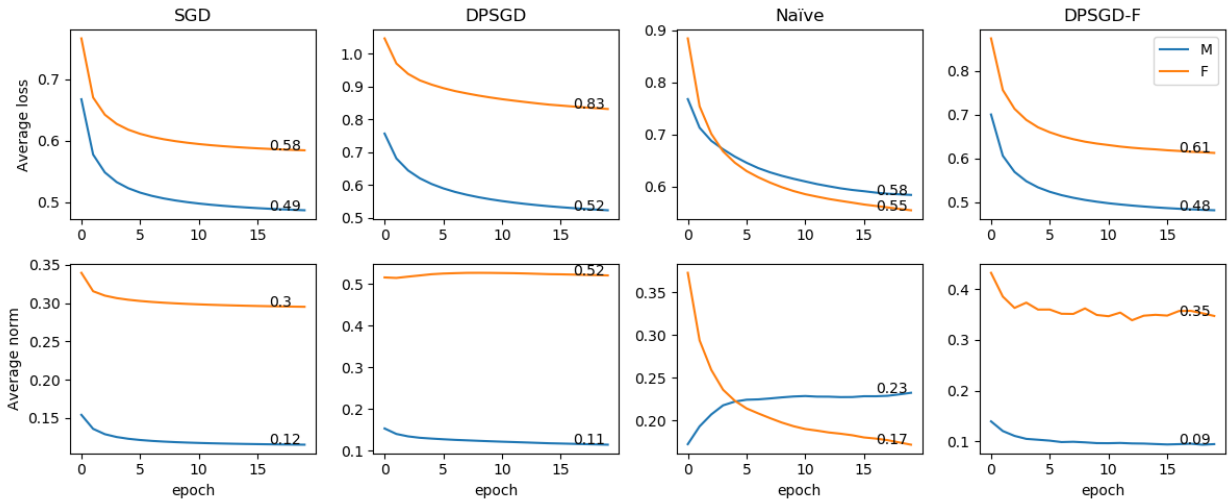


Figure 9.8: The average loss and the average gradient norm w.r.t. each group over epochs for SGD, DPSGD, Naïve and DPSGD-F on the unbalanced Dutch dataset ($\epsilon = 3.29, \delta = 10^{-6}$)

average gradient norm and the average loss for the male group are much closer to the ones in SGD. Similar to the Adult dataset, on the unbalanced Dutch dataset, the average gradient norm and the average loss in DPSGD-F for the male group are much closer to the ones in SGD. Figure 9.7 and 9.8 show the convergence trends on the unbalanced Adult and Dutch datasets during training. The trends in DPSGD-F is the closest to the trends in SGD among all DP models. It shows that our adjusted clipping bound helps to achieve the same group utility loss.

9.6 Summary

Gradient clipping and random noise addition, which are the core techniques in differentially private SGD, disproportionately affect underrepresented and complex classes and subgroups. As a consequence, DPSGD has disparate impact: the accuracy of a model trained using DPSGD tends to decrease more on these classes and subgroups vs. the original, non-private model. If the original model is unfair in the sense that its accuracy is not the same across all subgroups, DPSGD exacerbates this unfairness. In this work, we developed DPSGD-F to achieve differential privacy, equal costs of differential privacy, and good utility. DPSGD-F adjusts the contribution of samples in a group depending on the group clipping bias such that differential privacy has no disparate impact on group utility. Our experimental evaluation shows how group sample size and group clipping bias affect the impact of differential privacy in DPSGD, and how adaptive clipping for each group helps to mitigate the disparate impact caused by differential privacy in DPSGD-F. Gradient clipping in the non-private context may improve the model robustness against outliers. However, examples in the minority group are not outliers. They should not be ignored by the (private) learning model. The early version of this work is posted at arXiv [117].

10 Conclusion and Future Work

In this chapter, we summarize our works and, based on the observed results and performance analysis, propose several potential research directions associated with privacy preserving and fairness-aware machine learning.

10.1 Conclusion

Around privacy preserving and fairness-aware machine learning, in this dissertation, the works are delivered by addressing the following problems:

1. How to achieve differential privacy in complex and emerging tasks, such as causal graph discovery and network embedding;
2. How to achieve fair data generation and classification through generative models based on different fairness notions;
3. How to simultaneously achieve both differential privacy and fairness in logistic regression;
4. How to ensure the utility loss due to differential privacy is fair towards each group. Aiming to solve the above challenges, we have done the works as follows.

In Chapter 4, we studied how to preserve differential privacy in constraint-based causal graph discovery. The naive differentially private PC algorithm requires a large privacy budget since it adds protection at each conditional independence test. To solve this problem, we developed a differentially private PC algorithm (PrivPC) based on the exponential mechanism. Because PrivPC reduces the number of edge elimination decisions, it

reduces the required privacy budget significantly. The proposed PrivPC algorithm is robust over a wide range of parameter settings. Moreover, the conditionally independent adjacent set selection strategy achieves high true discovery rates for high-dimensional and dense causal graphs. Meanwhile, we also developed a Laplace mechanism based differentially private PC algorithm (PrivPC*) for numerical data. The experimental results also showed that our algorithms have good performance.

In Chapter 5, we developed a differentially private network embedding method (DPNE) based on DeepWalk as matrix factorization. We applied the objective perturbation approach on the objective function of matrix factorization. Our evaluation shows that on both vertex classification and link prediction tasks our DPNE achieves satisfactory performance.

In Chapter 6, we first developed FairGAN to generate fair data, which is free from disparate treatment and disparate impact, while retaining high data utility. FairGAN consists of one generator and two discriminators. We then developed FairGAN$^+$ to (1) generate fair data, which is free from disparate treatment and disparate impact w.r.t. the real protected attribute, while retaining high data utility, and (2) release a fair classifier that satisfies classification-based fairness, such as demographic parity, equality of odds and equality of opportunity. FairGAN$^+$ consists of one generator, one classifier and three discriminators. The experimental results showed the effectiveness of FairGAN on fair data generation, the effectiveness of FairGAN$^+$ on both fair data generation and fair classification, and the better trade-off of FairGAN$^+$ between the utility and fairness when co-training a generative model and a classifier.

In Chapter 7, we developed the causal fairness-aware generative adversarial networks (CFGAN) for generating high quality fair data. We considered various causal-based fairness criteria, including total effect, direct discrimination, indirect discrimination, and counterfac-

tual fairness. CFGAN consists of two generators and two discriminators. The two generators aim to simulate the original causal model and the interventional model. This is achieved by arranging the neural network structure of the generators following the original causal graph and the interventional graph. Then, two discriminators are adopted for achieving both the high data utility and causal fairness. Experiments using the Adult dataset showed that CF-GAN can achieve all types of fairness with relatively small utility loss.

In Chapter 8, we developed two differentially private and fair logistic regression models, PFLR and PFLR*. PFLR is to apply the functional mechanism to the objective function with fairness constraint as a penalty term. Our enhanced model, PFLR*, takes advantage of the connection between ways of achieving differential privacy and fairness and adds the Laplace noise with non-zero mean. The experimental results on two datasets demonstrate the effectiveness of two approaches and show the superiority of PFLR*. In this work, we considered logistic regression as the classification model and the covariance between decision boundary and the protected attribute as the fairness constraint.

In Chapter 9, we developed DPSGD-F to remove the potential disparate impact of differential privacy on the protected group. DPSGD-F adjusts the contribution of samples in a group depending on the group clipping bias such that differential privacy has no disparate impact on group utility. Our experimental evaluation shows how group sample size and group clipping bias affect the impact of differential privacy in DPSGD, and how adaptive clipping for each group helps to mitigate the disparate impact caused by differential privacy in DPSGD-F. Gradient clipping in the non-private context may improve the model robustness against outliers. However, examples in the minority group are not outliers. They should not be ignored by the (private) learning model.

10.2 Future Work

In this section, based on the works we have done, we propose some potential works on privacy preserving and fairness-aware machine learning.

In Chapter 4, we studied how to preserve differential privacy in the widely used classic PC algorithm. The PC algorithm still has some limitations including (a) vulnerability to errors in statistical independence test results, (b) no ranking or estimation of the confidence in the causal predictions, and (c) erroneous conclusions due to hidden variables. In future work, we will extend our study to other alternative causal modeling such as local causal discovery [118, 119, 120] and the Fast Causal Inference [34].

In Chapter 5, we developed DPNE based on DeepWalk as matrix factorization. DPNE can be easily employed in other network embedding methods if there exists an equivalent matrix factorization of a certain matrix. For example, LINE is proven to be factorizing a similar matrix to $\mathbf{M}$ [121]. We would derive differential privacy preserving LINE similarly. One potential limitation of the matrix factorization based methods is that they are not scalable to large networks. Graph factorization [105] uses a streaming algorithm for graph partitioning to improve factorization based embedding methods. In future work, we will extend our DPNE to deal with large networks.

In Chapter 6, we developed FairGAN and FairGAN$^+$ to generate fair data and release a fair classifier that satisfies classification-based fairness, such as demographic parity, equality of odds and equality of opportunity. In Chapter 7, we developed CFGAN to generate fair data that satisfies causation-based fairness in data, including total effect, direct discrimination, indirect discrimination, and counterfactual fairness. In future, we plan to achieve causation-based fairness in classification, so it can release a well-trained classifier

which makes predictions with no causal bias against the protected group.

In Chapter 8, we developed PFLR and PFLR* to simultaneously achieve differential privacy and decision boundary fairness in logistic regression. In future work, we plan to extend our methods to other classification models and other fairness constraints. Another research direction is to study allocation strategies of privacy budget, e.g., adding different amount of noise to coefficients containing different attributes.

In Chapter 9, we developed DPSGD-F to remove the potential disparate impact of differential privacy on the protected group. In future work, we can further improve our adaptive clipping method from group-wise adaptive clipping to element-wise (from user and/or parameter perspectives) adaptive clipping, so the model can be fair even to the unseen minority class.

When we consider both privacy and fairness issues in machine learning, there are many new research challenges unexploited. In this dissertation, we studied the problems of the within-model fairness in the differentially private models and the cross-model fairness in terms of group accuracy when applying differential mechanisms to a non-private model. In future work, we will continue along this general direction. First, there are malicious adversaries who may use membership attacks to target the privacy of machine learning models. It is unclear if unprivileged groups are more vulnerable to such attack. It is also unclear if state-of-the-art defense methods have the same performance across all groups. Second, the robustness of the private models and fair models is more complicated than the robustness of the vanilla models. There are new challenges on how to improve the robustness of a private and/or fair model. Third, federated learning is required when a large amount of data is distributed among different parties. It is not trivial to extend the privacy preserving and/or fairness-aware mechanisms to the distributed setting from the centralized setting.

Bibliography

[1] C. Dwork, F. McSherry, K. Nissim, and A. D. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of Cryptography, Third*, 2006, pp. 265–284.

[2] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *FOCS '07*, 2007, pp. 94–103.

[3] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially private empirical risk minimization," *J. Mach. Learn. Res.*, vol. 12, pp. 1069–1109, 2011.

[4] J. Zhang, Z. Zhang, X. Xiao, Y. Yang, and M. Winslett, "Functional mechanism: regression analysis under differential privacy," *PVLDB*, vol. 5, no. 11, pp. 1364–1375, 2012.

[5] K. Nissim, S. Raskhodnikova, and A. Smith, "Smooth sensitivity and sampling in private data analysis," in *Proceedings of the thirty-ninth annual ACM Symposium on Theory of Computing*. ACM, 2007, pp. 75–84.

[6] U. Erlingsson, V. Pihur, and A. Korolova, "Rappor: Randomized aggregatable privacy-preserving ordinal response," in *CCS'14*, 2014, pp. 1054–1067.

[7] Y. Wang, X. Wu, and D. Hu, "Using randomized response for differential privacy preserving data collection." in *Proceedings of the 9th International Workshop on Privacy and Anonymity in the Information Society*, 2016.

[8] S. Song, K. Chaudhuri, and A. D. Sarwate, "Stochastic gradient descent with differentially private updates," in *GlobalSIP*, 2013, pp. 245–248.

[9] K. Chaudhuri and C. Monteleoni, "Privacy-preserving logistic regression," in *NIPS'08*, 2008, pp. 289–296.

[10] Y. Wang, X. Wu, and L. Wu, "Differential privacy preserving spectral graph analysis," in *Pacific-Asia conference on knowledge discovery and data mining*. Springer, 2013, pp. 329–340.

[11] 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.

[12] N. Phan, Y. Wang, X. Wu, and D. Dou, "Differential privacy preservation for deep auto-encoders: an application of human behavior prediction." in *AAAI*, 2016, pp. 1309–1316.

[13] L. Zhang, Y. Wu, and X. Wu, "A causal framework for discovering and removing direct and indirect discrimination," ser. IJCAI'17, 2017, pp. 3929–3935.

[14] M. Feldman, S. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian, "Certifying and removing disparate impact," in *KDD*, 2015.

[15] L. Zhang, Y. Wu, and X. Wu, "Achieving non-discrimination in data release," in *KDD*, 2017, pp. 1335–1344.

[16] M. Hardt, E. Price, and N. Srebro, "Equality of opportunity in supervised learning," in *NeurIPS*, 2016.

[17] T. Kamishima, S. Akaho, and J. Sakuma, "Fairness-aware learning through regularization approach," in *ICDMW*, 2011.

[18] M. B. Zafar, I. Valera, M. Gomez-Rodriguez, and K. P. Gummadi, "Fairness constraints: Mechanisms for fair classification," in *AISTATS*, 2017.

[19] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. S. Zemel, "Fairness through awareness," in *Innovations in Theoretical Computer Science*, 2012, pp. 214–226.

[20] M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian, "Certifying and removing disparate impact," in *KDD*, 2015.

[21] L. Zhang, Y. Wu, and X. Wu, "A causal framework for discovering and removing direct and indirect discrimination," in *IJCAI*, 2017.

[22] M. Hardt, E. Price, and N. Srebro, "Equality of opportunity in supervised learning," in *NIPS*, 2016.

[23] L. Zhang, Y. Wu, and X. Wu, "Achieving non-discrimination in prediction," in *IJCAI*, 2018.

[24] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: online learning of social representations," in *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, 2014, pp. 701–710.

[25] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: large-scale information network embedding," in *Proceedings of the 24th International Conference on World Wide Web, WWW 2015, Florence, Italy, May 18-22, 2015*, 2015, pp. 1067–1077.

[26] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," in *NIPS*, 2014.

[27] F. Kamiran and T. Calders, "Data preprocessing techniques for classification without discrimination," *Knowl. Inf. Syst.*, vol. 33, no. 1, pp. 1–33, 2011.

[28] L. Zhang, Y. Wu, and X. Wu, "A causal framework for discovering and removing direct and indirect discrimination," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia*, 2017.

[29] J. Zhang and E. Bareinboim, "Fairness in Decision-Making – The Causal Explanation Formula," *The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, 2018.

[30] R. Nabi and I. Shpitser, "Fair Inference On Outcomes," *The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, 2018.

[31] M. J. Kusner, J. Loftus, C. Russell, and R. Silva, "Counterfactual fairness," in *Advances in Neural Information Processing Systems*, 2017.

[32] S. Chiappa, "Path-Specific Counterfactual Fairness," *The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)*, 2019.

[33] Y. Wu, L. Zhang, and X. Wu, "Counterfactual fairness: Unidentification, bound and algorithm," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, 2019.

[34] J. Pearl, *Causality: Models, Reasoning and Inference*, 2nd ed. New York, NY, USA: Cambridge University Press, 2009.

[35] P. Spirtes, C. N. Glymour, and R. Scheines, *Causation, prediction, and search*. MIT press, 2000.

[36] F. McSherry and K. Talwar, "Mechanism Design via Differential Privacy," *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pp. 94–103, 2007.

[37] C. Yang, Z. Liu, D. Zhao, M. Sun, and E. Y. Chang, "Network representation learning with rich text information," in *Proceedings of the IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, 2015, pp. 2111–2117.

[38] M. Abadi, A. Chu, I. J. 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, Vienna, Austria, October 24-28, 2016*, 2016, pp. 308–318.

[39] A. Blum, C. Dwork, F. McSherry, and K. Nissim, "Practical privacy: the SuLQ framework," in *Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*. ACM, 2005, pp. 128–138.

[40] B. Barak, K. Chaudhuri, C. Dwork, S. Kale, F. McSherry, and K. Talwar, "Privacy, accuracy, and consistency too: a holistic solution to contingency table release," *Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pp. 273–282, 2007.

[41] M. Hay, V. Rastogi, G. Miklau, and D. Suciu, "Boosting the Accuracy of Differentially-Private Histograms Through Consistency," *Proceedings of the VLDB Endowment*, vol. 3, no. 1-2, p. 15, 2009.

[42] B. Ding, M. Winslett, J. Han, and Z. Li, "Differentially private data cubes: optimizing noise sources and consistency," in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD 2011)*. ACM SIGMOD, January 2011, pp. 217–228.

[43] G. Cormode, C. Procopiuc, D. Srivastava, E. Shen, and T. Yu, *Differentially private spatial decompositions*, 2012, pp. 20–31.

[44] J. Zhang, X. Xiao, Y. Yang, Z. Zhang, and M. Winslett, "Privgene: Differentially private model fitting using genetic algorithms," in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '13. New York, NY, USA: ACM, 2013, pp. 665–676.

[45] C. Dwork, "A firm foundation for private data analysis," *Communications of the ACM*, vol. 54, no. 1, p. 86, 2011.

[46] A. Friedman and A. Schuster, "Data mining with differential privacy," *Knowledge discovery and data mining*, pp. 493–502, 2010. [Online]. Available: http://portal.acm.org/citation.cfm?id=1835804.1835868

[47] A. Johnson, "Privacy-Preserving Data Exploration in Genome-Wide Association Studies Categories and Subject Descriptors," pp. 1079–1087, 2013.

[48] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao, "Privbayes: Private data release via bayesian networks," in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. ACM, 2014, pp. 1423–1434.

[49] J. Zhang, Z. Zhang, X. Xiao, Y. Yang, and M. Winslett, "Functional mechanism: regression analysis under differential privacy," *Proceedings of the 38th International Conference on Very Large Data Bases*, vol. 5, no. 11, pp. 1364–1375, 2012.

[50] K. Nissim, S. Raskhodnikova, and A. Smith, "Smooth sensitivity and sampling in private data analysis," *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing - STOC '07*, no. x, p. 75, 2007.

[51] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Local privacy and statistical minimax rates," in *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, 2013, pp. 429–438.

[52] R. Bassily, A. G. Thakurta, and O. D. Thakkar, "Model-agnostic private learning," in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*, 2018, pp. 7102–7112.

[53] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially private empirical risk minimization," *J. Mach. Learn. Res.*, vol. 12, pp. 1069–1109, 2011.

[54] K. Amin, A. Kulesza, A. M. Medina, and S. Vassilvitskii, "Bounding user contributions: A bias-variance trade-off in differential privacy," in *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, 2019, pp. 263–271.

[55] V. Pichapati, A. T. Suresh, F. X. Yu, S. J. Reddi, and S. Kumar, "Adaclip: Adaptive clipping for private SGD," *CoRR*, vol. abs/1908.07643, 2019.

[56] O. Thakkar, G. Andrew, and H. B. McMahan, "Differentially private learning with adaptive clipping," *CoRR*, vol. abs/1905.03871, 2019.

[57] N. Phan, X. Wu, H. Hu, and D. Dou, "Adaptive laplace mechanism: Differential privacy preservation in deep learning," in *2017 IEEE International Conference on Data Mining, ICDM 2017, New Orleans, LA, USA, November 18-21, 2017*, 2017, pp. 385–394.

[58] J. Lee and D. Kifer, "Concentrated differentially private gradient descent with adaptive per-iteration privacy budget," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, 2018, pp. 1656–1665.

[59] F. Kamiran and T. Calders, "Classifying without discriminating," in *Control and Communication 2009 2nd International Conference on Computer*, 2009, pp. 1–6.

[60] T. Calders, F. Kamiran, and M. Pechenizkiy, "Building classifiers with independency constraints," in *2009 IEEE International Conference on Data Mining Workshops*, 2009, pp. 13–18.

[61] E. Krasanakis, E. S. Xioufis, S. Papadopoulos, and Y. Kompatsiaris, "Adaptive sensitive reweighting to mitigate bias in fairness-aware classification," in *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, 2018, pp. 853–862.

[62] M. J. Kearns, S. Neel, A. Roth, and Z. S. Wu, "Preventing fairness gerrymandering: Auditing and learning for subgroup fairness," in *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, 2018, pp. 2569–2577.

[63] H. Edwards and A. J. Storkey, "Censoring representations with an adversary," in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2016.

[64] A. Beutel, J. Chen, Z. Zhao, and E. H. Chi, "Data decisions and theoretical implications when adversarially learning fair representations," in *FAT/ML*, 2017.

[65] D. Madras, E. Creager, T. Pitassi, and R. S. Zemel, "Learning adversarially fair and transferable representations," in *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, ser. Proceedings of Machine Learning Research, J. G. Dy and A. Krause, Eds., vol. 80. PMLR, 2018, pp. 3381–3390.

[66] B. H. Zhang, B. Lemoine, and M. Mitchell, "Mitigating unwanted biases with adversarial learning," in *AAAI Conference on AI, Ethics and Society*, 2018.

[67] T. Kamishima, S. Akaho, and J. Sakuma, "Fairness-aware learning through regularization approach," in *2011 IEEE 11th International Conference on Data Mining Workshops*, 2011, pp. 643–650.

[68] M. B. Zafar, I. Valera, M. G. Rodriguez, and K. P. Gummadi, "Fairness constraints: Mechanisms for fair classification," in *AISTATS*, 2017.

[69] F. Kamiran, T. Calders, and M. Pechenizkiy, "Discrimination aware decision tree learning," in *2010 IEEE International Conference on Data Mining*, 2010, pp. 869–874.

[70] S. Hajian, J. Domingo-Ferrer, A. Monreale, D. Pedreschi, and F. Giannotti, "Discrimination- and privacy-aware patterns," *Data Min. Knowl. Discov.*, vol. 29, no. 6, pp. 1733–1782, 2015.

[71] M. D. Ekstrand, R. Joshaghani, and H. Mehrpouyan, "Privacy for all: Ensuring fair and equitable privacy protections," in *Conference on Fairness, Accountability and Transparency*, 2018, pp. 35–47.

[72] J. Ding, X. Zhang, X. Li, J. Wang, R. Yu, and M. Pan, "Differentially private and fair classification via calibrated functional mechanism," in *AAAI*, 2020.

[73] D. Xu, S. Yuan, and X. Wu, "Achieving differential privacy and fairness in logistic regression," in *Companion of The 2019 World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, 2019, pp. 594–599.

[74] M. Jagielski, M. J. Kearns, J. Mao, A. Oprea, A. Roth, S. Sharifi-Malvajerdi, and J. Ullman, "Differentially private fair learning," in *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, 2019, pp. 3000–3008.

[75] R. Cummings, V. Gupta, D. Kimpara, and J. Morgenstern, "On the compatibility of privacy and fairness," in *Adjunct Publication of the 27th Conference on User Modeling, Adaptation and Personalization, UMAP 2019, Larnaca, Cyprus, June 09-12, 2019*, 2019, pp. 309–315.

[76] E. Bagdasaryan, O. Poursaeed, and V. Shmatikov, "Differential privacy has disparate impact on model accuracy," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, Eds., 2019, pp. 15 453–15 462.

[77] C. Dwork and A. Roth, "The Algorithmic Foundations of Differential Privacy," *Foundations and Trends in Theoretical Computer Science*, vol. 9, no. 2013, pp. 211–407, 2014.

[78] W. Rudin, *Principles of mathematical analysis*, ser. International series in pure and applied mathematics. McGraw-Hill, 1953.

[79] Y. Wu, L. Zhang, and X. Wu, "On convexity and bounds of fairness-aware classification," in *WWW*. ACM, 2019, pp. 3356–3362.

[80] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv:1511.06434 [cs]*, 2015.

[81] E. Choi, S. Biswal, B. Malin, J. Duke, W. F. Stewart, and J. Sun, "Generating multi-label discrete patient records using generative adversarial networks," in *MLHC*, 2017.

[82] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier gans," in *ICML*, 2017.

[83] M. Kocaoglu, C. Snyder, A. G. Dimakis, and S. Vishwanath, "Causalgan: Learning causal implicit generative models with adversarial training," in *6th International Conference on Learning Representations*, 2018.

[84] D. Edwards, *Introduction to graphical modelling*. Springer Science & Business Media, 2012.

[85] S. L. Lauritzen, *Graphical models*. Clarendon Press, 1996, vol. 17.

[86] N. Homer, S. Szelinger, M. Redman, D. Duggan, W. Tembe, J. Muehling, J. V. Pearson, D. A. Stephan, S. F. Nelson, and D. W. Craig, "Resolving individuals contributing trace amounts of DNA to highly complex mixtures using high-density SNP genotyping microarrays," *PLoS Genetics*, vol. 4, no. 8, 2008.

[87] D. Colombo and M. H. Maathuis, "Order-independent constraint-based causal structure learning," *Order-Independent Causal Structure Learning*, vol. 15, no. 2010, pp. 1–40, 2010.

[88] J. Ramsey, "A PC-Style Markov Blanket for High Dimensional Datasets," Tech. Rep., 2006.

[89] M. Kalisch and P. Bühlmann, "Estimating high-dimensional directed acyclic graphs with the pc-algorithm," *Journal of Machine Learning Research*, vol. 8, no. Mar, pp. 613–636, 2007.

[90] N. Harris and M. Drton, "Pc algorithm for nonparanormal graphical models." *Journal of Machine Learning Research*, vol. 14, no. 1, pp. 3365–3383, 2013.

[91] T. Verma and J. Pearl, "Equivalence and synthesis of causal models," in *Proceedings of Sixth Conference on Uncertainty in Artificial Intelligence*, 1991, pp. 220–227.

[92] S. A. Andersson, D. Madigan, M. D. Perlman *et al.*, "A characterization of markov equivalence classes for acyclic digraphs," *The Annals of Statistics*, vol. 25, no. 2, pp. 505–541, 1997.

[93] A. Hauser and P. Bühlmann, "Characterization and greedy learning of interventional Markov equivalence classes of directed acyclic graphs," *Journal of Machine Learning Research*, vol. 13, pp. 2409–2464, 2012. [Online]. Available: http://jmlr.org/papers/v13/hauser12a.html

[94] R. Scheines, P. Spirtes, C. Glymour, C. Meek, and T. Richardson, "The tetrad project: Constraint based aids to causal model specification," *Multivariate Behavioral Research*, vol. 33, no. 1, pp. 65–117, 1998.

[95] M. Lichman, "UCI Machine Learning Repository," http://archive.ics.uci.edu/ml, 2013.

[96] D. Xu, S. Yuan, and X. Wu, "Differential privacy preserving causal graph discovery," in *Privacy-Aware Computing (PAC), 2017 IEEE Symposium on*, 2017, pp. 60–71.

[97] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *CoRR*, vol. abs/1705.02801, 2017.

[98] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, 2013, pp. 3111–3119.

[99] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, 2016, pp. 855–864.

[100] J. Li, J. Zhu, and B. Zhang, "Discriminative deep random walk for network classification," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*, 2016.

[101] S. Yuan, X. Wu, and Y. Xiang, "Sne: signed network embedding," in *Pacific-Asia conference on knowledge discovery and data mining*. Springer, 2017, pp. 183–195.

[102] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *SCIENCE*, vol. 290, pp. 2323–2326, 2000.

[103] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, 2000.

[104] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, 2001, pp. 585–591.

[105] A. Ahmed, N. Shervashidze, S. M. Narayanamurthy, V. Josifovski, and A. J. Smola, "Distributed large-scale natural graph factorization," in *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*, 2013, pp. 37–48.

[106] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of KDD*. ACM, 2016, pp. 855–864.

[107] P. Yanardag and S. V. N. Vishwanathan, "Deep graph kernels," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, 2015, pp. 1365–1374.

[108] J. Hua, C. Xia, and S. Zhong, "Differentially private matrix factorization," in *Proceedings of IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, 2015, pp. 1763–1770.

[109] D. Xu, S. Yuan, X. Wu, and H. Phan, "DPNE: differentially private network embedding," in *PAKDD*, 2018.

[110] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv:1411.1784 [cs, stat]*, 2014.

[111] D. Xu, S. Yuan, L. Zhang, and X. Wu, "Fairgan: Fairness-aware generative adversarial networks," in *BigData*. IEEE, 2018, pp. 570–575.

[112] ——, "Fairgan$^+$: Achieving fair data generation and classification through generative adversarial nets," in *2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, December 9-12, 2019*. IEEE, 2019, pp. 1401–1406.

[113] D. Xu, Y. Wu, S. Yuan, L. Zhang, and X. Wu, "Achieving causal fairness through generative adversarial networks," in *IJCAI*, 2019.

[114] I. Zliobaite, F. Kamiran, and T. Calders, "Handling conditional discrimination," in *11th IEEE International Conference on Data Mining, ICDM 2011, Vancouver, BC, Canada, December 11-14, 2011*, 2011, pp. 992–1001.

[115] M. Du, R. Jia, and D. Song, "Robust anomaly detection and backdoor attack detection via differential privacy," *CoRR*, vol. abs/1911.07116, 2019.

[116] M. Jaiswal and E. M. Provost, "Privacy enhanced multimodal neural representations for emotion recognition," *CoRR*, vol. abs/1910.13212, 2019.

[117] D. Xu, W. Du, and X. Wu, "Removing disparate impact of differentially private stochastic gradient descent on model accuracy," *CoRR*, vol. abs/2003.03699, 2020.

[118] C. F. Aliferis, A. Statnikov, I. Tsamardinos, S. Mani, and X. D. Koutsoukos, "Local causal and markov blanket induction for causal discovery and feature selection for classification part i: Algorithms and empirical evaluation," *The Journal of Machine Learning Research*, vol. 11, pp. 171–234, 2010.

[119] ——, "Local causal and markov blanket induction for causal discovery and feature selection for classification part ii: Analysis and extensions," *The Journal of Machine Learning Research*, vol. 11, pp. 235–284, 2010.

[120] A. Statnikov, J. Lemeir, and C. F. Aliferis, "Algorithms for discovery of multiple markov boundaries," *The Journal of Machine Learning Research*, vol. 14, no. 1, pp. 499–566, 2013.

[121] J. Qiu, Y. Dong, H. Ma, J. Li, K. Wang, and J. Tang, "Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, 2018, pp. 459–467.