Towards Fair and Robust Classification

Anonymous

Abstract

Recent studies have shown that machine learning (ML) models could be deliberately evaded and misled. These studies result in profound implications of model robustness, especially when employing ML to critical applications such as autonomous driving, surveillance systems, and disease diagnosis. Additionally, recent studies have revealed potential societal biases in ML models, where the models make unintentional "unfair" decisions that favor particular individuals or groups of individuals while discriminating against the others. It raises the great needs for trustworthy ML systems that are robust to resist adversary attacks, and produce fair decisions.

Despite the active research on robustness and fairness of ML recently, these efforts focus on either fairness or robustness, but not both. To bridge this gap, in this paper, we design Fair and Robust Classification (FRoC) models. Meeting both fairness and robustness constraints is not trivial due to the conflicts between them as well as the trade-off between fairness, robustness, and model accuracy. To address these challenges, we design two FRoC methods, namely FRoC-Pre that modifies the input data before model training, and FRoC-In that modifies the model with an adversarial objective function to address both fairness and robustness during training. FRoC-In is suitable to the settings where the users (e.g., ML service providers) only have the access to the model but not the original data, while FRoC-Pre works for the settings where the users (e.g., data owners) have the access to both data and a surrogate model that may have similar architecture as the target model. Our extensive experiments on real-world datasets demonstrate that both FRoC-In and FRoC-Pre can achieve both fairness and robustness with insignificant accuracy loss of the target model.

1 Introduction

Machine learning (ML) techniques are increasingly providing decision making and operational support across multiple domains and applications. One important concern for the adoption of ML techniques into operational decision processes is the trustworthiness of these techniques.

Recent years have seen a proliferation of research in technologies for trustworthy ML. Two important issues of trustworthy ML are *fairness* and *robustness*. On one hand, ML models have been criticized for systemic biases that result in unintentional "unfair" decisions that favor particular individuals or groups of individuals while discriminating against others [2, 37]. On the other hand, ML models are vulnerable to those carefully crafted adversarial examples [43, 45] and thus can be easily misled and manipulated. The discovery that ML models are neither fair nor robust hinders significantly their practical deployment in the security-critical applications such as healthcare, finance, and transportation. Therefore, ensuring both fairness and robustness of ML models is paramount to the widespread adoption of ML in our society.

There has been a considerable body of studies of fairness and robustness of machine learning. However, most of these studies focus on either fairness or robustness, but not both. In this paper, we consider both fairness and robustness, and aim to design ML models that are not only fair but also robust against adversarial attacks. We consider classification models as our target model. In terms of fairness, we consider statistical parity [16], a widely-used fairness notion, as our fairness definition. In terms of robustness, we consider the robustness against two popular evasion attacks, namely *Fast Gradient Sign Method* (FGSM) [21] and *Projected Gradient Descent* (PGD) attack [34], by which the adversary aims to perturb test inputs to ML classifiers in order to cause misclassification.

A straightforward solution to meet both fairness and robustness requirements is to realize the two requirements on the ML models independently in a sequential fashion, i.e., ensuring the classifiers satisfy one of the requirements first before handling the other. Although the solution is seemly sound, it is indeed incorrect due to the conflicts between fairness and robustness. A recent study [9] has shown that the susceptibility of fair models to the data poisoning attacks indeed increases compared with the original models without fairness enhancement. Our study (will be presented later in this paper) also shows that imposing one constraint can counteract the effect of the other. For example, applying defense on fair ML

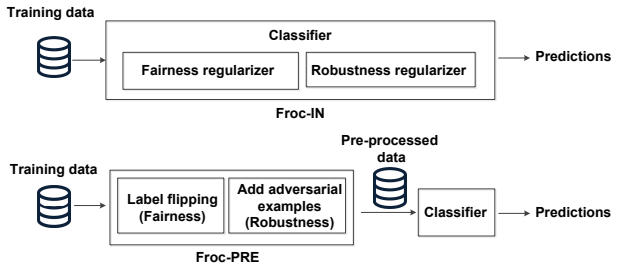


Figure 1: An illustration of FRoC-In and FRoc-Pre methods

classifiers against the evasion attacks can bring bias and thus make the fair model unfair again.

In this paper, we present the design of Fair and Robust classification (FROC) models that satisfy both fairness and robustness constraints simultaneously. There exists tension between fairness and model accuracy, between robustness and accuracy, as well as between fairness and robustness. Therefore, one of the main challenges that we will address is the fundamental trade-off between fairness, robustness, and model accuracy. To address this challenge, we design two different FROC methods that are deployed at different phases of the ML pipeline: (1) A *pre-processing* method (FROC-PRE) that is deployed before training - it modifies the training data so that the models trained on that data will be fair and robust; and (2) An *in-processing* method (FROC-IN) that is deployed during training - it modifies the objective function of the target model to address both fairness and robustness constraints. Figure 1 illustrates both methods at high level. FROC-IN is suitable to the settings where the users (e.g., ML service providers) only have the access to the model but not the original training data, while FROC-PRE is suitable for the settings where the users (e.g., data owners) have the access to not only the training data but also a surrogate model which may have similar architecture as the target model.

We make the following contributions. First, we formalize the definition of *bias score* to measure model fairness in terms of statistical parity, and *robustness score* to measure model robustness against two types of evasion attacks. We also formalize our problem of designing a fair and robust classifier model as an optimization problem that maximizes model accuracy while satisfying δ_F-*fairness* and δ_R-*robustness* (i.e., the bias score and robustness score should meet the constraints of user-specified thresholds δ_F and δ_R respectively).

Second, for the FROC-IN method, we design the fairness regularizer of statistical parity, and the robustness regularizer for both evasion attacks (FGSM and PGD). We add both regularizers to the target model so that it is trained with an adversarial objective function. We address the trade-off between fairness, robustness, and accuracy by controlling the weights of both regularizers.

Third, for the FROC-PRE method, we design an iterative approach that applies two operations to modify the training data in several rounds, where a small portion of training data is modified in each round. These two operations are: (1) flipping the binary labels of a set of original training samples (for fairness); and (2) augmenting the training data with a set of adversarial examples (for robustness). Since the users may not have the access to the target model, FROC-PRE considers a surrogate model that has either identical or similar architecture as the target model. To address the trade-off between fairness, robustness, and accuracy, it identifies the training samples (for label flipping) and adversarial examples (for data augmentation) that have the highest influence on the performance of the surrogate model. In particular, FROC-PRE estimates two influence scores: (1) the *fairness-accuracy influence score* that quantifies the influence of flipping a label on model fairness and accuracy; and (2) the *accuracy influence score* that quantifies the influence of adding an adversarial example to the training data on model accuracy. We design simple and efficient *influence functions* that estimate both influence scores without model re-training.

Last but not least, we extensively evaluate the performance of both FROC-IN and FROC-PRE methods on multiple real-world datasets. We made the following observations from our experimental results.

- Imposing both robustness and fairness constraints independently can fail to meet one of the constraints, as the enforcement of one constraint can counteract the effect of the other. Thus both constraints should be dealt with simultaneously.
- Both FROC-IN and FROC-PRE well address the trade-off between fairness, robustness, and model accuracy. In particular, both methods deliver at most 10.26% accuracy loss among the three datasets while the model satisfies δ_F-fairness and δ_R-robustness for various δ_F and δ_R settings.
- FROC-IN is more suitable for the settings of strong robustness requirement, while FROC-PRE is more suitable for the settings that require strong fairness.

2 Preliminaries

2.1 Algorithmic Fairness

A multitude of formal, mathematical definitions of algorithmic fairness has been proposed in the last few years. These definitions can be categorized into two categories: (1) *Group fairness* that is concerned with particular demographic groups (such as racial or gender groups) and requires that some statistic of interest be approximately equalized across groups [7, 16, 23]; and (2) *Individual fairness* [15] that prevents discrimination against individuals and requires similar individuals are treated similarly. In this paper, we mainly focus on group fairness.

In general, the group fairness model is defined as following: given a dataset of domain $A \times X \times Y$, where A denotes the *protected attributes* (e.g., gender), X denotes the non-protected attributes, and Y is an outcome feature, the classifier model $\mathbf{M}$ learned from these samples should not have discriminatory

effects towards the *protected groups* (e.g., female) defined by the values associated with the protected attribute compared with the *un-protected groups* (e.g., male). For simplicity, we only consider one protected/un-protected group in this paper. In the following discussions, we use $a = 0$ and $a = 1$ to indicate the protected and un-protected groups respectively.

2.2 Adversarial Robustness

There has been active research on adversarial attacks on ML in recent years. The main attacks can be categorized into two types [4]: (1) The *evasion attacks* by which the adversary tries to evade the system by adjusting malicious samples during testing phase; (2) The *poisoning attacks* by which an adversary tries to poison the training data by injecting carefully designed samples during training phase. In this paper, we only focus on the evasion attacks because we focus on the performance of models at inference time. Specifically, we consider two popular evasion attacks: *Fast Gradient Sign Method* (FGSM) [21] and *Projected Gradient Descent* (PGD) attack [34], which are explained below.

Fast gradient sign method (FGSM) attack. FGSM [21] uses linear perturbation on the features of the testing samples. In particular, let θ be the parameter of the given model $\mathbf{M}$, and $\mathbf{L}()$ be the loss function. The perturbation is performed by adding a noise vector η on the sample $\mathbf{x}$, where the noise is calculated as $\eta = \varepsilon \cdot \text{sign}\left(\nabla_{\mathbf{x}} \mathbf{L}(\theta, \mathbf{x}, y)\right)$. The parameter ε controls the intensity of the attack. The adversarial testing examples are generated as:

$$\widetilde{\mathbf{x}} = \mathbf{x} + \eta = \mathbf{x} + \varepsilon \cdot \text{sign}\left(\nabla_{\mathbf{x}} \mathbf{L}(\theta, \mathbf{x}, y)\right) \tag{1}$$

Projected gradient descent (PGD) attack. While FGSM is considered as a simple one-step scheme, PGD attack [34] is the multi-step variant of FGMS. Formally, the adversarial examples at the $(t+1)$-th iteration is generated as following:

$$\widetilde{\mathbf{x}}_{t+1} = \Pi_{\mathbf{x}+\mathcal{S}}\left(\widetilde{\mathbf{x}}_t + \xi \cdot \text{sign}\left(\nabla_{\mathbf{x}} \mathbf{L}\left(\theta, \mathbf{x}_t, y\right)\right)\right) \tag{2}$$

where Π is the projection operator, $\mathcal{S}$ is a set of perturbation candidates, and ξ is the parameter that controls the intensity of the attack in every iteration.

Adversarial training as defense. There have been significant efforts on designing defense techniques against the evasion attacks. *Adversarial training* (AT) is one of the most promising ways to obtain the adversarial robustness of learning models. The key idea of AT is to generate adversarial examples and augment these perturbed data while training the target model. The augmentation can be done either by feeding the model with both the original data and the crafted data [22, 30] or by learning with a modified objective function [21]. In this paper, we consider the latter approach as the defense against adversarial examples. Specifically, when training a model $\mathbf{M}$ with the utility loss function $\mathbf{L}$ on a training dataset $\{a_i, \mathbf{x}_i, y_i\}_{i=1}^n$, the new loss $\widetilde{\mathbf{L}}$ is defined as:

$$\widetilde{\mathbf{L}}(\mathbf{M}, \{\mathbf{x}\}, \{y\}) = \mathbf{L}(\mathbf{M}, \{\mathbf{x}\}, \{y\}) + \lambda \cdot \mathbf{L}(\mathbf{M}, \{\widetilde{\mathbf{x}}\}, \{y\}) \tag{3}$$

where $\{\widetilde{\mathbf{x}}\}_{i=1}^n$ are the adversarial examples generated by either FGSM or PGD attack, and λ controls the trade-off between robustness and model accuracy. Higher (lower) λ indicates higher (lower) robustness but worse (better) utility.

3 Problem Formulation

Consider a training dataset (A, X, Y) with m samples, where A, X, and Y are the protected attributes, non-protected attributes, and labels respectively. We consider a binary classifier $\mathbf{M}$ that predicts the labels of given data samples .

Disparate treatment. To prevent the impacts of the protected attribute on prediction, we apply *disparate treatment* [16], a simple bias mitigation method that is widely used by the fairness community, to our model training process. Disparate treatment simply excludes the protected attributes from model training. However, disparate treatment is insufficient to make fair models, due to the correlations between protected and un-protected attributes [11, 16]. Additional fairness-enhancing techniques need to be deployed.

Model accuracy. Typically, the binary classification can be performed from the prediction of a posterior distribution (called posteriors) over the class labels. We use $\mathbf{M_c}$ to denote a probability classifier that outputs the posteriors in $[0, 1]$. Then $\mathbf{M}$ can be considered as a binary classifier that binarizes the output of $\mathbf{M_c}$. Formally, the model $\mathbf{M}$ makes the binary prediction as: $\mathbf{M}(\mathbf{x}) = \mathbb{1}(\mathbf{M_c}(\mathbf{x}) \geq 0.5), \forall \mathbf{x} \in X$, where $\mathbb{1}(\cdot)$ is the indicator function, which returns 1 if the condition holds, otherwise 0. Intuitively, $\mathbf{M}$ predicts $\hat{y} = 1$ if the posterior is no less than the threshold 0.5, otherwise $\hat{y} = 0$. Note that the protected attribute A is not included in training of $\mathbf{M}$ due to the disparate treatment.

There are various evaluation metrics to measure the accuracy of classification models. In this paper, we consider accuracy of the model $\mathbf{M}$ as the fraction of predictions that are correct. Formally:

$$\texttt{Acc}(X, Y; \mathbf{M}) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}(\mathbf{M}(\mathbf{x}_i) = y_i). \tag{4}$$

We use *binary cross entropy* (BCE) as the loss function. BCE is commonly used to measure the performance of a classification model whose output is a probability. Formally, the BCE loss function $\mathbf{L}_{\text{U}}$ is defined as following:

$$\mathbf{L}_{\text{U}}(X, Y) = \frac{1}{n} \sum_{i=1}^{n} \left[y_i \log \mathbf{M_c}(\mathbf{x}_i) + (1 - y_i) \log(1 - \mathbf{M_c}(\mathbf{x}_i)) \right] \tag{5}$$

Note that $\mathbf{L}_{\text{U}}(X, Y)$ does not consider the protected attribute due to the disparate treatment.

Fairness. As fairness is a complex and multi-faceted concept which depends on many factors (e.g., context and domains), many statistical definitions of fairness have been introduced in the literature [35]. And yet none is universally applicable. Therefore, we only consider a fairness definition,

namely **statistical parity**, that is widely used in the fairness community. Statistical parity [6, 16] (also known as demographic parity) requires that the probability of being classified with positive labels should be the same across the protected and un-protected groups. Formally,

$$\Pr(\hat{y}=1|a=1)=\Pr(\hat{y}=1|a=0) \tag{6}$$

Following the definition of statistical parity, we define fairness as the difference in the positive rates of the protected and un-protected groups. Formally, given a classification model $\mathbf{M}$, a set of testing samples $\{(a_i^{\text{test}},\mathbf{x}_i^{\text{test}},y_i^{\text{test}})\}_{i=1}^n$ and their predictions $\hat{Y}^{\text{test}}=\{\hat{y}_i^{\text{test}}\}_{i=1}^n$ made by $\mathbf{M}$, the fairness of $\mathbf{M}$ is measured as the *bias score* $\mathcal{S}_B$ of $\hat{Y}$:

$$\mathcal{S}_B=\left|\frac{\sum_{i=1}^n\left[\mathbb{1}(a_i^{\text{test}}=0\wedge\hat{y}_i^{\text{test}}=1)\right]}{\sum_{i=1}^n\mathbb{1}(a_i^{\text{test}}=0)}-\frac{\sum_{i=1}^n\left[\mathbb{1}(a_i^{\text{test}}=1\wedge\hat{y}_i^{\text{test}}=1)\right]}{\sum_{i=1}^n\mathbb{1}(a_i^{\text{test}}=1)}\right| \tag{7}$$

Obviously, $\mathcal{S}_B\in[0,1]$. The closer $\mathcal{S}_B$ is to 0, the more fair the model $\mathbf{M}$ is. We say *a model* $\mathbf{M}$ *is* δ_F*-fair* if its bias score $\mathcal{S}_B\leq\delta_F$, where δ_F is a user-specified threshold.

Robustness. Given a target model $\mathbf{M}$, a set of testing samples $\{(a_i^{\text{test}},\mathbf{x}_i^{\text{test}},y_i^{\text{test}})\}_{i=1}^n$ and their predictions $\hat{Y}^{\text{test}}=\{\hat{y}_i^{\text{test}}\}_{i=1}^n$ made by $\mathbf{M}$, we follow the metric of *adversarial accuracy* [25, 34, 38] to measure the *robustness score* $\mathbf{M}$. Informally, adversarial accuracy measures the accuracy of $\mathbf{M}$ on the adversarial examples $\{\widetilde{\mathbf{x}}_i^{\text{test}}\}$. Accordingly, we define the *robustness score* $\mathcal{S}_R$ as following:

$$\mathcal{S}_R=\texttt{Acc}\left(\{\widetilde{\mathbf{x}}_i^{\text{test}}\},\{y_i^{\text{test}}\};\mathbf{M}\right) \tag{8}$$

Intuitively, $\mathcal{S}_R$ measures the percentage of adversarial examples that are correctly predicted by $\mathbf{M}$ (i.e., they fail the evasion attack). Apparently, larger $\mathcal{S}_R$ indicates better robustness. We say *a model* $\mathbf{M}$ *is* δ_R*-robust* if its robustness score $\mathcal{S}_R\geq\delta_R$, where δ_R is a user-specified threshold.

Problem definition. The primary research question that we study in this paper is how to make the model $\mathbf{M}$ δ_B-fair (in terms of statistical parity) and δ_R-robust against both evasion attacks without much sacrifice on model accuracy. Formally, the problem is defined by:

$$\begin{array}{ll}\max\limits_{\mathbf{M}} & \texttt{Acc}(X^{\text{test}},Y^{\text{test}};\mathbf{M})\\ \text{s.t.} & \mathcal{S}_B(A^{\text{test}},X^{\text{test}};\mathbf{M})\leq\delta_F\\ & \mathcal{S}_R(X^{\text{test}},Y^{\text{test}};\mathbf{M})\geq\delta_R\end{array} \tag{9}$$

where δ_F and δ_R are the user-specified thresholds for bias score and robustness score respectively.

4 FROC-IN: Our In-processing Method

Quite a few prior works use adversarial regularization to realize either fairness or robustness constraint. For example, [9, 40] equip the objective function of the classification model with a fairness regularizer. Similarly, [21, 33, 39] adds an adversarial regularizer to the objective function of the model to enhance its robustness. Following these works, we design FROC-IN that enforces both fairness and robustness constraints as regularizers to the objective function of the target model. The new loss function of the model is defined as:

$$\mathbf{L}(A,X,Y)=\mathbf{L}_{\text{U}}(X,Y)+\lambda_F\cdot\mathbf{F}(A,X)+\lambda_R\cdot\mathbf{R}(X,Y) \tag{10}$$

where $\mathbf{L}_{\text{U}}$ is the accuracy loss of the target model (Equation 5), $\mathbf{F}$ and $\mathbf{R}$ are the regularizer terms of fairness and robustness respectively, and λ_F and λ_R are the parameters that control the trade-off between fairness, robustness, and accuracy. Larger λ_F (λ_R, resp.) indicates stronger fairness (robustness, resp.) but worse accuracy. Next, we explain how we design the two regularizers $\mathbf{F}$ and $\mathbf{R}$.

Fairness regularizer. Intuitively, the bias score (Equation 7) can be used to impose a penalty on the loss function as the fairness regularizer. However, it cannot be directly used as the fairness regularizer as the indicator function $\mathbb{1}(\cdot)$ is not continuous and thus its gradient cannot be properly calculated during training. To address this problem, we apply the following approximation on the bias score. First, we apply the following transformation

$$\mathbb{1}(a=z\wedge\hat{y}=1)\Rightarrow\mathbb{1}(a=z)\cdot\mathbb{1}(\hat{y}=1)(z\in\{0,1\}).$$

As we only consider binary sensitive attributes, we can further apply the following transformation:

$$\mathbb{1}(a=z)=\begin{cases}1-a, & z=0\\ a, & z=1\end{cases} \tag{11}$$

Next, we apply approximate $\mathbb{1}(\hat{y}=1)$ to make it outputs continuous values:

$$\mathbb{1}(\hat{y}=1)\approx\mathbf{M}_{\mathbf{c}}(\mathbf{x}). \tag{12}$$

where $\mathbf{M}_{\mathbf{c}}$ outputs the posterior of x (defined in Section 3). In other words, the binary label output by $\mathbf{M}$ is approximated as the continuous probability output by $\mathbf{M}_{\mathbf{c}}$.

Based on Equations (11) and (12), each component in Equation (7) is transformed to the followings:

$$\begin{cases}\mathbb{1}(a_i=1)=a_i,\\ \mathbb{1}(a_i=0)=1-a_i,\\ \mathbb{1}(a_i=1\wedge\hat{y}_i=1)\approx a_i\mathbf{M}_{\mathbf{c}}(\mathbf{x}_i),\\ \mathbb{1}(a_i=0\wedge\hat{y}_i=1)\approx(1-a_i)\mathbf{M}_{\mathbf{c}}(\mathbf{x}_i)\end{cases}$$

Thus the fairness regularizer term $\mathbf{F}$ can be written as:

$$\mathbf{F}(A,X)=\left|\frac{\sum_{i=1}^n(1-a_i)\mathbf{M}_{\mathbf{c}}(\mathbf{x}_i)}{\sum_{i=1}^n(1-a_i)}-\frac{\sum_{i=1}^n a_i\mathbf{M}_{\mathbf{c}}(\mathbf{x}_i)}{\sum_{i=1}^n a_i}\right| \tag{13}$$

We further simplify Equation (13). Let $C_0=\frac{1}{\sum_{i=1}^n(1-a_i)}$ and $C_1=\frac{1}{\sum_{i=1}^n a_i}$. Apparently, both of them are constants for a

given dataset. Thus Equation (13) can be rewritten as:

$$\begin{aligned}\mathbf{F}(A,X) &= \left|\sum_{i=1}^{n} C_0(1-a_i)\mathbf{M_c}(\mathbf{x}_i) - \sum_{i=1}^{n} C_1 a_i \mathbf{M_c}(\mathbf{x}_i)\right| \\ &= \left|\sum_{i=1}^{n} (C_0 - a_i C_0 - a_i C_1)\mathbf{M_c}(\mathbf{x})\right| \\ &= \left|\sum_{i=1}^{n} c_i \mathbf{M_c}(\mathbf{x})\right|\end{aligned} \tag{14}$$

where $c_i = C_0 - a_i C_0 - a_i C_1$. With a set of given training samples, c_i is a constant and thus can be calculated by one traversal of the training samples.

Robustness regularizer. We follow the same idea of adversarial training [21] to define our robustness regularizer. Intuitively, during adversarial training, a set of adversarial examples are generated dynamically during training. These adversarial examples are derived from the parameters of the model in the previous epoch, and participate the current epoch as a penalty term. We adapt this to our setting and define the robustness regularizer as following:

$$\mathbf{R}(X,Y) = \mathbf{L}_{\mathrm{U}}(\widetilde{X},Y) \tag{15}$$

where $\mathbf{L}_{\mathrm{U}}$ is the accuracy function (Equation (5)), and $\widetilde{X} = \{\widetilde{\mathbf{x}}\}$ are the adversarial examples.

Model training with both regularizers. The in-processing model with both regularizers (Equation (10)) can be trained by the stochastic gradient descent (SGD) method. In particular, the gradient for each iteration is calculated as:

$$\nabla_{\theta}\mathbf{L}(A,X,Y) = \nabla_{\theta}\mathbf{L}_{\mathrm{U}}(X,Y) + \lambda_F \cdot \nabla_{\theta}\mathbf{F}(A,X) + \lambda_R \cdot \nabla_{\theta}\mathbf{R}(X,Y)$$

First, as the accuracy loss $\mathbf{L}_{\mathrm{U}}(X,Y)$ is defined as a *binary cross entropy* function, its gradient is calculated as:

$$\nabla_{\theta}\mathbf{L}_{\mathrm{U}}(X,Y) = \tfrac{1}{n}\textstyle\sum_{i=1}^{n}[y_i \nabla_{\theta}\log\mathbf{M_c}(\mathbf{x}_i) + (1-y_i)\log\nabla_{\theta}(1-\mathbf{M_c}(\mathbf{x}_i))]$$

Next, the gradient of the fairness term in Equation (14) is computed as:

$$\nabla_{\theta}\mathbf{F}(A,X) = \mathtt{sign}\left(\left|\sum_{i=1}^{n} c_i \mathbf{M_c}(\mathbf{x})\right|\right) \cdot \frac{1}{n}\sum_{i=1}^{n} c_i \nabla_{\theta}\mathbf{M_c}(\mathbf{x})$$

Finally, the gradient of the robustness term in Equation (15) is computed as:

$$\nabla_{\theta}\mathbf{R}(X,Y) = \tfrac{1}{n}\textstyle\sum_{i=1}^{n}[y_i \nabla_{\theta}\log\mathbf{M_c}(\widetilde{\mathbf{x}}_i) + (1-y_i)\log\nabla_{\theta}(1-\mathbf{M_c}(\widetilde{\mathbf{x}}_i))]$$

where $\widetilde{\mathbf{x}}_i$ is the adversarial example generated by either FGSM or PGD attack.

5 FRoC-PRE: Our Pre-Processing Method

In this paper, we assume the testing data follows the same distribution as the training data. Thus reducing the bias in the training data should lead to more fair outcomes on the testing data. Following this assumption, we design FROC-PRE that modifies the training data so that the model trained on the modified data is fair and robust. We consider two types of data modification: (1) flip the binary labels of a set of original training samples $D_F \subseteq D_{train}$; and (2) augment D_{train} with a set of adversarial examples D_R. Label flipping aims to remove the bias from the training data, while augmenting with adversarial examples is to enhance the robustness of the model. Intuitively, we aim to find D_F and D_R that make the model $\mathbf{M}$ trained on the pre-processed dataset (A',X',Y') satisfy both requirements of δ_F-fairness and δ_R-robustness.

Although both δ_F-fairness and δ_R-robustness are required on the testing data (Equation (9)) which may not be available during the pre-processing phase, a model that is δ_F-fair and δ_R-robust on the training data should be δ_F-fair and δ_R-robust on the testing data too, due to the assumption that both training and testing data have the same distribution. Thus we formalize the pre-processing problem as an optimization problem defined as following:

$$\begin{aligned}\max_{A',X',Y'} \quad & \mathtt{Acc}(X,Y;\mathbf{M}) \\ \text{s.t.} \quad & \mathbf{M} = \arg\max_{\mathbf{M}'} \mathtt{Acc}(X',Y';\mathbf{M}') \\ & \mathcal{S}_B(A,X;\mathbf{M}) \le \delta_F \\ & \mathcal{S}_R(X,Y;\mathbf{M}) \ge \delta_R\end{aligned} \tag{16}$$

where the function ACC, $\mathcal{S}_B$ and $\mathcal{S}_R$ are the accuracy, bias score and robustness score of the model $\mathbf{M}$ respectively, and δ_F and δ_R are the user-specified thresholds for bias score and robustness score. Our experimental results show that, FROC-PRE ensures that the model is δ_F-fair and δ_R-robust on the testing data. More details can be found in Section 6.

Choosing both D_F and D_R in one shot may be too rigid and lead to significant accuracy loss. Therefore, we take a greedy, sequential approach to pick samples of D_F and D_R in multiple trials. In each trial, we pick and modify one small portion of training data, and observe the change of model fairness and robustness by the modification. If the model achieves both requirements δ_F-fairness and δ_R-robustness, we terminate the modification. Otherwise, we continue with the next trial.

Following this idea, we design an iterative method that picks the data samples for modification. Specifically, consider the original training data D^0_{train}. At the i-th iteration, FROC-PRE applies the following two steps on the current dataset D^i_{train} and generates the dataset D^{i+1}_{train} for the next iteration:

- **Step 1.** If δ_F-fairness is not satisfied, select ℓ_1 samples D^i_F from D^0_{train}. Flip the labels of each sample in D^i_F.
- **Step 2.** If δ_R-robustness is not satisfied, select ℓ_2 samples D^i_R from D^0_{train}, generate ℓ_2 adversarial examples D^i_R accordingly. Augment D^i_{train} with D^i_R (i.e., $|D^{i+1}_{train}| = |D^i_{train}| + \ell_2$);

The algorithm repeats the two steps until either both δ_F-fairness and δ_R-robustness are met or the number of iterations

has reached a pre-defined budget. Apparently, $D_R = \cup_{D_R^i}$ and $D_F = \cup_{D_F^i}$. Both ℓ_1 and ℓ_2 values control the impacts of fairness and robustness on model. To ensure equal impact of fairness and robustness, we use $\ell_1 = \ell_2$ in the algorithm, and use $\ell = \ell_1(\ell_2)$ in the following discussions.

Given m samples in the training data, there are $\binom{m}{\ell}$ choices to pick ℓ samples. A naive solution to the optimization problem in Equation (16) is to enumerate all choices and pick the one that returns the best accuracy. Apparently this is unacceptable due to its high complexity. Therefore, we design a greedy algorithm to solve the optimization problem. Intuitively, first, for each training sample, we estimate the "influence" of flipping its label on both model fairness and accuracy, and pick ℓ samples of the largest influence by label flipping. Next, we estimate the "influence" of each adversarial example on model accuracy, and pick ℓ ones that have the minimal influence for data augmentation. We note that the model that FROC-PRE estimates the influence on is not necessarily the same as the target model $\mathbf{M}$. We denote this model as the *surrogate model* $\mathbf{M}_S$, which is also a binary classification model that may have identical or similar architecture as $\mathbf{M}$. Since both $\mathbf{M}$ and $\mathbf{M}_S$ are classification models, they use the same loss function $\mathbf{L}_\mathrm{U}$ (Equation (5)). In the following discussion, we first present how to estimate these two types of influence on a particular classification model. Then we discuss the FROC-PRE in details.

5.1 Estimating Influence of Label Flipping on Model Fairness and Accuracy

To estimate the influence of flipping the label of a particular training sample on model fairness and accuracy, we first estimate the influence of label flipping on model fairness alone. Then we estimate the "combined" influence of a label flipping on model fairness and accuracy together. Next, we present the details of these two steps.

Estimating fairness influence of label flipping. Changing the label of a training sample can impact the model's bias score. Intuitively, the influence of a particular label flipping on model fairness can be measured by asking the counterfactual: *how would model fairness change if the model sees a different label of this sample during training?* A simple solution of estimating a label's influence on model fairness is to flip the label, re-train the model from scratch, and measure the bias score of the re-trained model. Apparently, this process is not acceptable due to its high computation cost. Unfortunately, the influence function in the literature [28] only estimates the impact of a training sample on model accuracy. It cannot estimate the influence of label flipping on model fairness.

In this paper, we design a new influence function that estimates the influence of flipping a label y on model fairness. The estimation is not straightforward, as the bias score (Equation (7)) is computed from binary labels. To facilitate our estimation, we consider the approximation of the bias score in Equation (14). However, Equation (14) does not involve Y, which makes difficult to estimate the influence on model fairness. Therefore, we estimate the influence on fairness in two steps: First, we estimate the change on the model parameters θ by flipping a label. Then we estimate the influence on bias score of the model by parameter changes. We use $\bar{y}$ to denote flipping the label y.

Step 1: Estimate the change on model parameters. The change in model parameters due to flipping a label in the training set can be formalized as $\theta_{\bar{y}} - \theta$. To estimate this change efficiently, we adapt the basic idea of the influence function [14] to our setting. The idea is to compute the parameter change if considering flipping label y as y being upweighted by some small τ, so that $\theta_\tau = argmin_\theta \frac{1}{n}\sum_{i=1}^{n} \mathbf{L}_\mathrm{U}(z_i,\theta) + \tau \mathbf{L}_\mathrm{U}(z,\theta)$. The influence of upweighting y on the parameters θ can be computed as the following [14]: $I_{up}(\mathbf{z})|_{\tau=0} = -H_\theta^{-1}\nabla_\theta \mathbf{L}_\mathrm{U}(\mathbf{x},y)$, where $H_\theta = \frac{1}{n}\sum_{i=1}^{n}\nabla_\theta^2 \mathbf{L}_\mathrm{U}(X,Y)$ is the Hessian and is positive by assumption. Now, consider flipping the label of a sample $\mathbf{z} = (\mathbf{x},y)$ to $\bar{\mathbf{z}} = (\mathbf{x}, 1-y)$. The change of parameters θ is computed as:

$$I_\theta^f(\mathbf{z}) = I_{up}(\bar{\mathbf{z}}) - I_{up}(\mathbf{z}) = -H_\theta^{-1}\left[\nabla_\theta \mathbf{L}_\mathrm{U}(\mathbf{x},1-y) - \nabla_\theta \mathbf{L}_\mathrm{U}(\mathbf{x},y)\right]$$

Step 2: Estimate the change on bias score. We estimate the influence of parameter changes (by Step 1) on the approximate bias score (Equation (14)) as following:

$$I_f = -\nabla_\theta \mathbf{F}(A,X)$$

Then, the *fairness influence score* of flipping the label y of a particular training sample on model fairness is estimated as:

$$I_F(y) = I_f^\top I_\theta^f(\mathbf{z}), \tag{17}$$

where a negative (resp., positive) $I_F(y)$ value indicates that flipping y will lower (resp., increase) the bias. The absolute value $|I_F(y)|$ indicates the intensity of influence.

Estimating fairness-utility influence of label flipping. As the optimization problem (Equation (16)) aims to minimize the accuracy loss, we estimate the influence of flipping a label y on model accuracy loss as:

$$I_U^f(y) = I_u^\top I_\theta^f(\mathbf{z})$$

where $I_u = -\nabla_\theta \mathbf{L}_\mathrm{U}(X,Y)$. Intuitively, a positive (resp., negative) $I_U(y)$ indicates a demotion (resp., promotion) of model accuracy by flipping y.

Next, we combine I_F and I_U into one influence score, namely the *fairness-accuracy influence score* $I_{FU}(y)$ that quantifies the influence of flipping a particular label y on both model fairness and accuracy:

$$I_{FU}(\mathbf{z}) = I_F(\mathbf{z}) \cdot \exp\left(-\left|I_U^f(\mathbf{z})\right|\right), \tag{18}$$

A negative (resp., positive) I_{FU} indicates that flipping y will lower (resp., increase) the bias. Furthermore, For a negative

$I_{FU}(y)$ value, the larger $|I_{FU}(y)|$ is, the more improvement to fairness and the less impact on model accuracy when the label y is flipped. Thus we pick the labels of the negative $I_{FU}(y)$ that with the largest absolute value for flip.

5.2 Estimating Adversarial Example's Influence on Model Accuracy

Generating adversarial examples for tabular data. While most of the existing works consider how to generate adversarial examples of image data, how to generate adversarial examples for structured (tabular) data is largely unexplored. We design the following approach to generate the adversarial examples for tabular data. First we transform all categorical data to numerical by one-hot encoding. Then we add noise on the numerical values. In particular, given a data sample $\{a, \mathbf{x}, y\}$, we generate its adversarial example as $\{a, \widetilde{\mathbf{x}}, y\}$, where $\widetilde{\mathbf{x}}$ is generated by either FGSM or PGD attack. In other words, the adversary example only perturbs the non-protected attributes X, but keeps the protected attributes A and labels Y unchanged. We do not perturb the protected attribute A because it is not included in training of the target model due to the disparate treatment.

Estimating influence of adversarial examples on model accuracy. Intuitively, adding adversarial examples into the training data can either improve or downgrade model accuracy. A naive method to measure the impact of an adversarial sample on model accuracy is to add it into the training data and retrain the model. Apparently this method is computationally expensive. Therefore, given a training sample $(\mathbf{x}, y)$ and its adversarial example $\widetilde{\mathbf{x}}$, we estimate the *accuracy influence score* of $\widetilde{\mathbf{x}}$ as the following:

$$I_U(\widetilde{\mathbf{x}}) = \|\nabla_\theta \mathbf{L}_U(\widetilde{\mathbf{x}}, y)\|_2 \tag{19}$$

where $\mathbf{L}_U$ is the accuracy function (Equation (5)). Intuitively, the adversarial examples that have lower accuracy should have smaller influence on model accuracy if they are added into the training dataset.

5.3 Algorithm Details

The pseudo-code is shown in Algorithm 1. We highlight some details that were not covered in the previous discussions. First, when the algorithm picks training samples to flip, it does not pick any sample that has been flipped in previous iterations. Second, when the algorithm flips the labels of particular training samples, it also flips the labels of the adversarial examples of these training samples if there is any. Otherwise the generated adversarial example will have the opposite labels and lose its ability to enhance model robustness.

After the iterations, the algorithm picks ℓ_F labels in total of the largest fairness-accuracy influence score (Equation (18)) to flip, and ℓ_R adversarial examples of the smallest accuracy influence score (Equation (19)) to be added into the training data. The value of ℓ_F is not necessarily the same as ℓ_R.

Algorithm 1: FROC-PRE algorithm

Input : Training data (A, X, Y) with m samples; Fairness threshold δ_F; Robustness threshold δ_R; # of iterations L; # of samples to modify per iteration l; Surrogate model $\mathbf{M}_S$

Output : Pre-processed training dataset (A', X', Y')

$(A^{(0)}, X^{(0)}, Y^{(0)}) = (A, X, Y)$;
Initialize the set of flipped labels $\mathcal{F} = \emptyset$;
Initialize the set of adversarial examples $\mathcal{R} = \emptyset$;
for $i = 1$ *to* L **do**
  Train model $\mathbf{M}_S^{(i)}$ on dataset $(A^{(i-1)}, X^{(i-1)}, Y^{(i-1)})$;
  Calculate $\mathcal{S}_B$ (Eqn. (7)) and $\mathcal{S}_R$ (Eqn. (8)) of $\mathbf{M}_S^{(i)}$;
  if $\mathcal{S}_R \geq \delta_R$ **and** $\mathcal{S}_B \leq \delta_F$ **then** `// Meet both constraints`
    break;
  $(A^{(i)}, X^{(i)}, Y^{(i)}) = (A^{(i-1)}, X^{(i-1)}, Y^{(i-1)})$;
  if $\mathcal{S}_B > \delta_F$ **and** $|\mathcal{F}| < m$ **then** `// Fairness`
    $Z_F = \left\{ y_j \middle| y_j \in Y^{(0)} \wedge j \notin \mathcal{F} \right\}$;
    for $y_j \in Z_F$ **do**
      Calculate $I(y_j)$ (Eqn. (18));
    Pick top-l labels of the smallest negative $I(y_j)$ (i.e., largest fairness-accuracy influence score) as P;
    $\mathcal{F} = \mathcal{F} \cup \{i | y_i \in P\}$;
    Flip $\left\{ y_j \in Y^{(i)} \middle| y_j \in P \wedge \left(\mathbf{x}_j \in X^{(i)} \vee \widetilde{\mathbf{x}}_j \in X^{(i)} \right) \right\}$;
  if $\mathcal{S}_R < \delta_R$ **and** $|\mathcal{R}| < n$ **then** `// Robustness`
    $\widetilde{Z}_R = \left\{ \texttt{Adv}(\mathbf{x}_j, y_j; \mathbf{M}^{(i)}) \middle| \mathbf{x}_j \in X^{(0)} \wedge j \notin \mathcal{R} \right\}$;
    for $\widetilde{\mathbf{x}}_j \in Z_R$ **do**
      Calculate $\mathcal{L}(\widetilde{\mathbf{x}}_j)$ (Eqn. (19));
    Pick top-l adversarial examples of the smallest accuracy influence score as P;
    $\mathcal{R} = \mathcal{R} \cup \{i | \widetilde{\mathbf{x}}_i \in P\}$;
    Augment $\left\{ (a_j, \widetilde{\mathbf{x}}_j, y_i) \middle| \widetilde{\mathbf{x}}_j \in P \right\}$ with $(A^{(i)}, X^{(i)}, Y^{(i)})$;
return $(A^{(i)}, X^{(i)}, Y^{(i)})$;

Our empirical results show that both flipped labels and adversarial examples take a small portion of the training data (at most 17.1% flipped labels and at most 13.3% as adversarial examples). More details can be found in Section 6.

Since FROC-PRE executes a fixed number of iterations, the output data may not allow the model to satisfy both δ_F-fairness and δ_R-robustness constraints. However, our experimental results show that this only happens when either the fairness or the robustness requirement is too strong. More details are included in Section 6.

6 Experiments

6.1 Experimental Setup

Computational environment. All the experiments are performed on a server with Ubuntu 18.04.4, two × Intel(R) Xeon(R) Silver 4214 CPU @ 2.20GHz, 384GB memory, and four NVIDIA Quadro RTX 6000 graphic cards. Codes are

Dataset	# of samples	# of features	Protected Attribute		Label
			Race	Gender	
Adult	45,222	96	non-white$^+$ white$^-$	F$^+$	Binary
Hospital	52,778	122		M$^-$	
COMPAS	20,000	37		M$^+$, F$^-$	

Table 1: Summary of the real-world datasets ($+$ and $-$ indicate the protected and un-protected groups, M and F stands for male and female).

implemented and executed in Python 3.7.9 with PyTorch 1.7.1 and Scikit-learn 0.23.2.

Datasets. We use three real-world datasets: *Adult* dataset [29], *COMPAS* dataset [24] and *Hospital* dataset [36]. We consider these three datasets because they are widely used in the fairness literature [16, 32, 47]. The statistics as well as the fairness setup of these datasets are shown in Table 1. More details of these datasets can be found in Appendix A. All categorical values in the datasets are transformed to numerical ones by one-hot encoding.

Target model. We consider neural network (NN) as the target model. The NN model consists of one hidden layer with 128 neurons and `ReLU` as the activation function. The output layer is activated by `Sigmoid` function. All models are trained on the training dataset that consists of 70% data samples randomly selected from each dataset, and tested on the remaining 30% data samples. Models are trained with 500 epochs, a batch size of 256 and the learning rate of 0.01.

Parameter setting of FROC-PRE algorithm. We use $L = 1{,}000$, $\delta_F \in [0.01, 0.2]$, and $\delta_R \in [0, 1]$ for the FROC-PRE algorithm.

Evaluation metrics. We use the bias score (Equation (7)), robustness scores (Equation (8)), and the accuracy (Equation (4)) to evaluate fairness, robustness, and model accuracy respectively.

Alternative sequential methods. We consider the alternative solution that enforces fairness and robustness independently in a sequence. We consider two different sequential methods: (1) **Robustness-then-fairness** method: we generate a number of adversarial examples and augment the training data with these examples. Then we apply fairness-enhancing methods to train the model on the training data with adversarial examples; (2) **Fairness-then-robustness** method: we apply fairness-enhancing methods on the training data to remove bias. Then we perform adversarial training with a robustness regularizer added to the model to train on the data with bias removed. We choose two fairness-enhancing algorithms from IBM's AIF360 fairness toolbox[1] that provide statistical parity: (1) *Reweighing* (RW) [27] method is a pre-processing method that associates a *weight* value with each training sample to indicate the independence between the protected attribute and the label; and (2) *Disparate Impact Remover* (DIR) [16] method is a pre-processing method that modifies the non-protected attributes so that their distribution across different groups is similar.

[1] IBM AIF360 fairness toolbox: https://aif360.mybluemix.net

6.2 Weakness of Sequential Method

We evaluated the performance of both sequential methods. It turns out that the robustness-then-fairness method cannot provide effective robustness effect, even without fairness. Thus we only show the results for the fairness-then-robustness method in the following discussion. We show the results of the fairness-then-robustness method on Adult and COMPAS datasets in Figure 2. The results of Hospital dataset are similar and can be found in Appendix C. We vary the value of the parameter $\lambda = \{0, 0.01, 0.05, 0.1, 0.5, 1.0\}$ for adversarial training (Equation (3)) to control the degree of robustness.

The most important observation is that, in most of the settings, the bias score increases (i.e., worse fairness) when the robustness parameter λ grows (i.e., more robustness). We take a deep analysis of this phenomenon and found that adding adversarial examples introduce bias, which counteracts the fairness-enhancing effect and thus make the model unfair again. Therefore, enforcing fairness and robustness independently cannot ensure both fairness and robustness.

6.3 Performance of FRoC-In Method

Trade-off between fairness, robustness, and accuracy. We measure model accuracy, fairness, and robustness of FROC-IN on the three datasets. The results are shown in Figure 3, with robustness scores and bias scores at x- and y- axis respectively. To help illustration, we use colors to visualize model accuracy. Lighter (deeper) color indicates better (worse) accuracy.

Unsurprisingly, the trade-off always exists between fairness, robustness and accuracy. In most of the settings, the accuracy downgrades when the robustness scores grows (i.e., stronger robustness). Similarly, the accuracy decreases when the bias scores decreases (i.e., more fairness). However, FROC-IN well addresses the trade-off between fairness, robustness and accuracy. For example, for Adult dataset, the model accuracy decreases at most 4.57% in all the settings, even when the robustness score as high as 0.816 and the bias score is as low as 0.009. For COMPAS dataset, the model accuracy decreases at most 4.41% in all the settings.

Interaction between fairness and robustness regularizers. In this part of experiments, we try to answer the research question: *How fairness and robustness interact with each other during model training?* We measure the angle between $\nabla_\theta \mathbf{F}$ and $\nabla_\theta \mathbf{R}$, namely the gradients of fairness and robustness regularizers, during model training. Intuitively, an angle that is greater than 90° indicates that fairness and robustness compete with each other, otherwise they are aligned with each other.

We tried various settings of $\lambda_\mathbf{F}$ and $\lambda_\mathbf{R}$ parameters, and vary the fairness setting on the protected attribute. We run 500 epochs of FROC-PRE, and monitor the angel between $\nabla_\theta \mathbf{F}$ and $\nabla_\theta \mathbf{R}$ during these epochs. Figure 4 shows the results

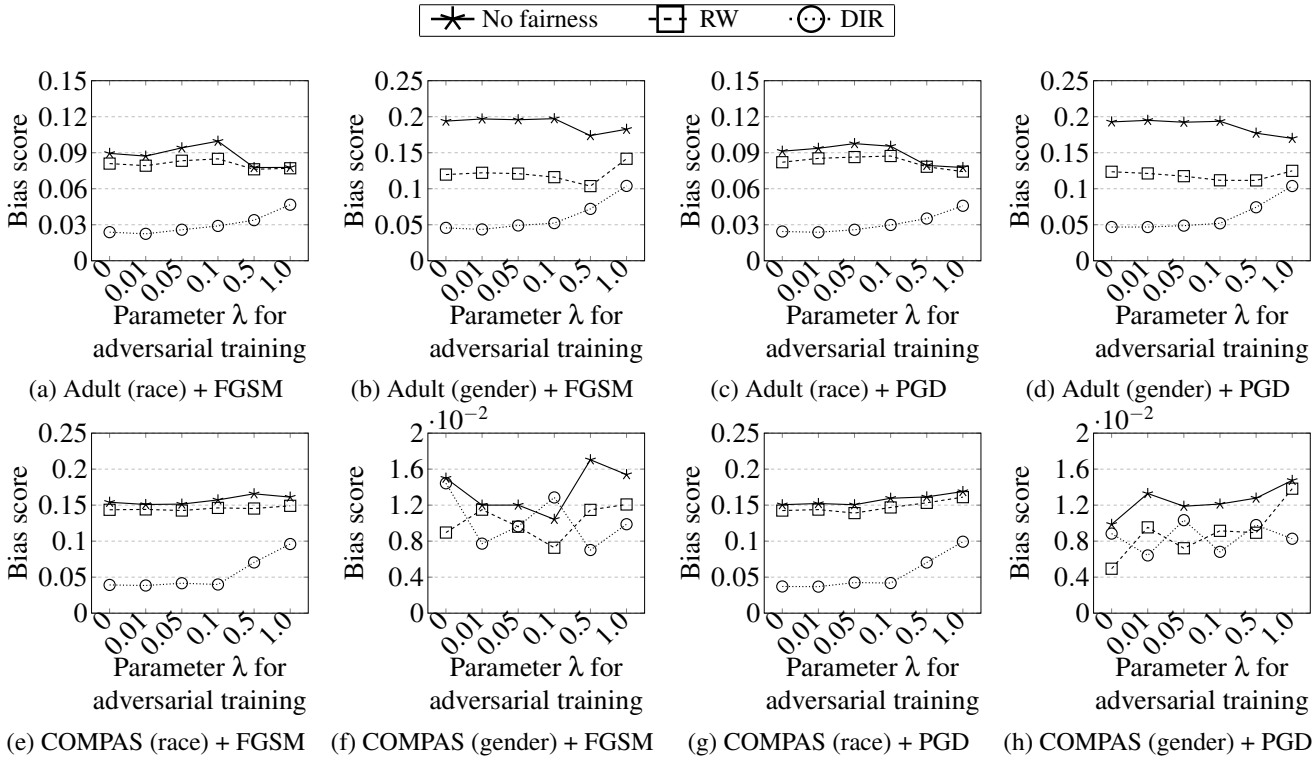


Figure 2: Performance of sequential method. The training dataset is pre-processed by either Reweighing (RW) [27] or Disparate Impact Remover method (DIR) [16]

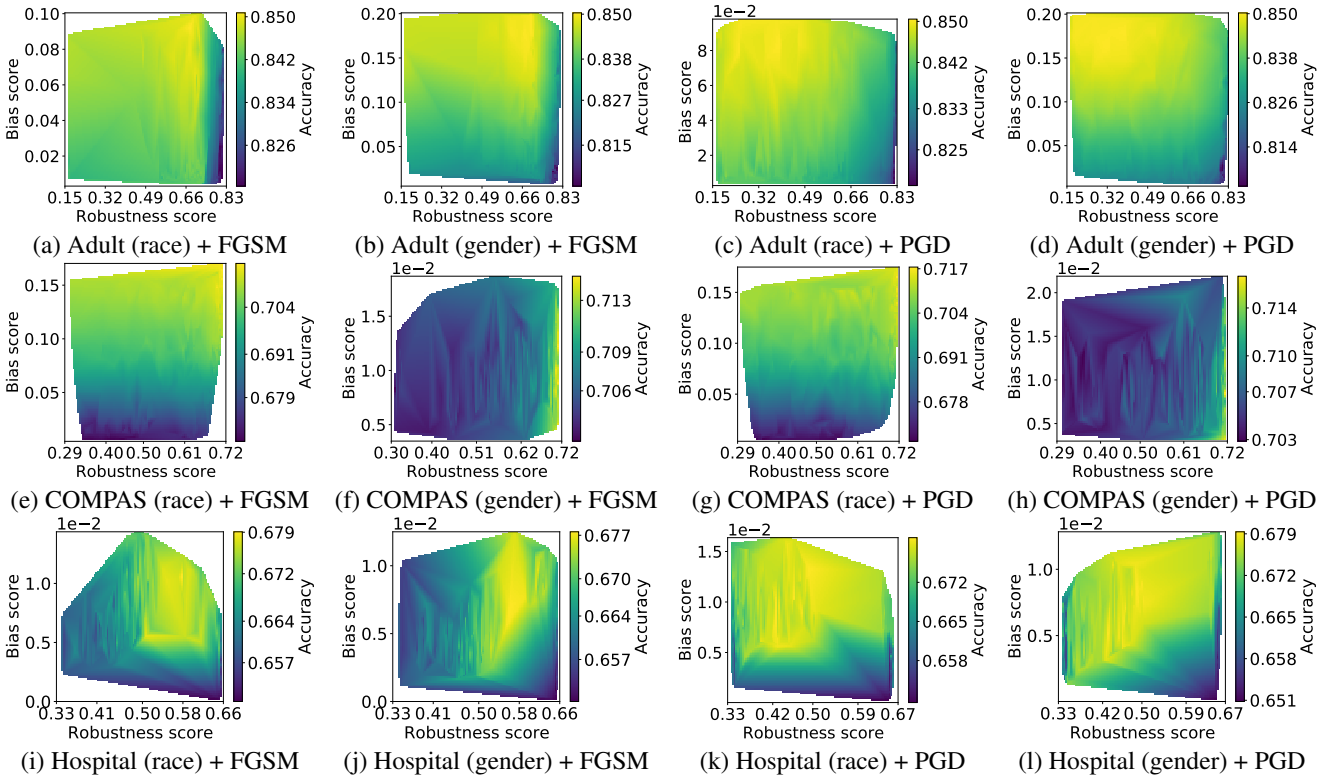
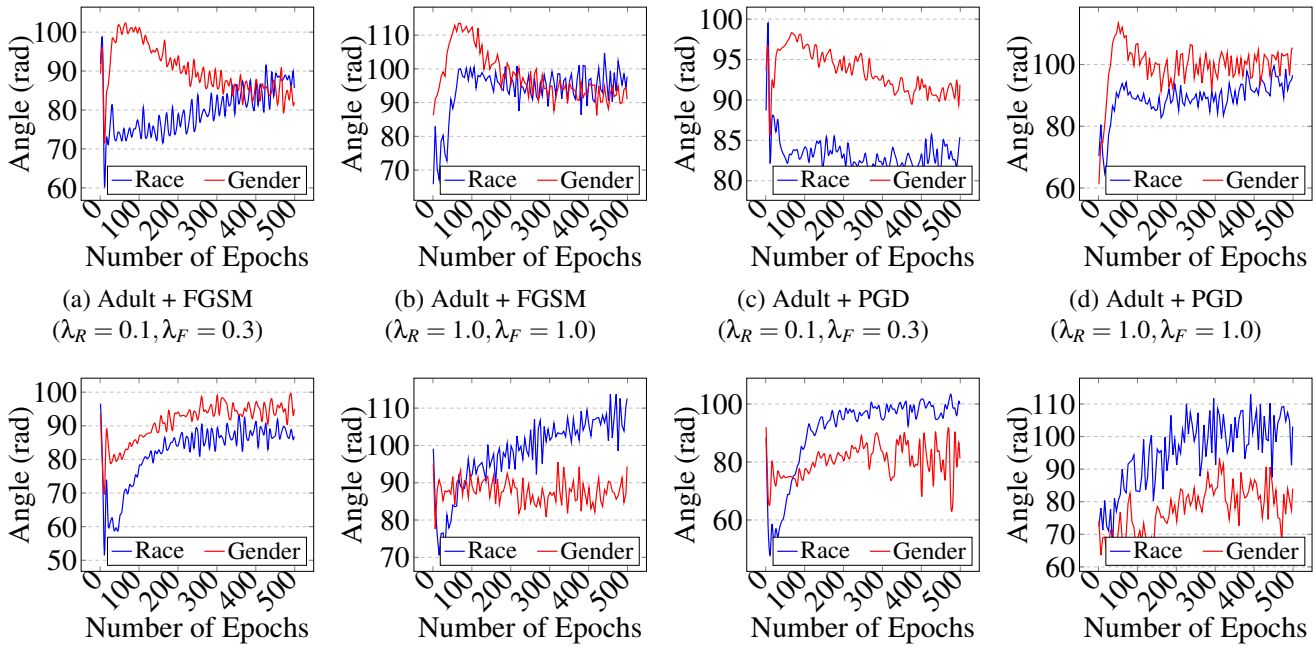


Figure 3: Model fairness, robustness, and accuracy of FROC-IN method. Accuracy is visualized in colors; light (deep) color indicates higher (lower) accuracy



(a) Adult + FGSM ($\lambda_R = 0.1, \lambda_F = 0.3$)
(b) Adult + FGSM ($\lambda_R = 1.0, \lambda_F = 1.0$)
(c) Adult + PGD ($\lambda_R = 0.1, \lambda_F = 0.3$)
(d) Adult + PGD ($\lambda_R = 1.0, \lambda_F = 1.0$)
(e) COMPAS + FGSM ($\lambda_R = 0.1, \lambda_F = 0.3$)
(f) COMPAS + FGSM ($\lambda_R = 1.0, \lambda_F = 1.0$)
(g) COMPAS + PGD ($\lambda_R = 0.1, \lambda_F = 0.3$)
(h) COMPAS + PGD ($\lambda_R = 1.0, \lambda_F = 1.0$)

Figure 4: Interaction between fairness and robustness regularizers during training of FROC-IN method, measured as the angle between the gradients of fairness and robustness regularizers

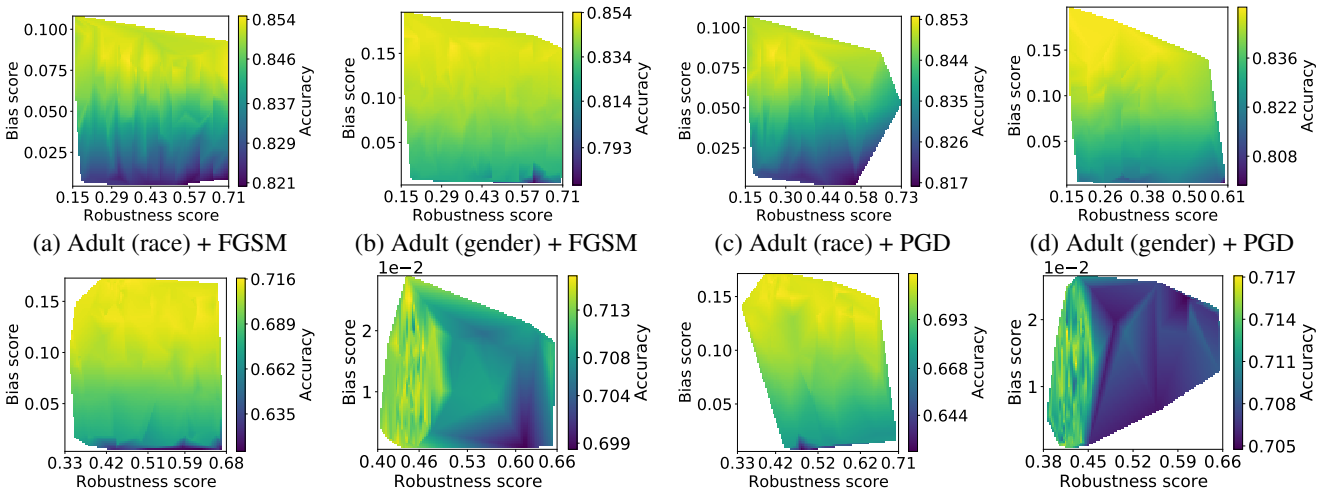
for Adult and COMPAS datasets. The results for Hospital dataset can be found in Appendix C. First, the angle between $\nabla_\theta \mathbf{F}$ and $\nabla_\theta \mathbf{R}$ is in the range of $[70°, 110°]$ after the initial 20 epochs, for all the settings. The angle grows fast at first, and quickly becomes larger than 90° in the first 20 epochs. This indicates that the fairness and robustness regularizers are competing with each other at beginning of model training. Then the angle eventually becomes relatively stable when the model approaches the convergence. In around 45.8% of all settings, the stabilized angle (for either Race or Gender protected attribute) is no larger than 90°. This indicates that the fairness and robustness regularizers become aligned with each other when the model is stabilized. The alignment facilities the model to achieve both fairness and robustness requirements with small model accuracy.

6.4 Performance of FRoC-Pre Method

Trade-off between fairness, robustness, and accuracy. Figure 5 shows the model accuracy, fairness, and robustness of FROC-PRE for both Adult and COMPAS datasets. The results for Hospital dataset are similar and can be found in Appendix C. First, FROC-PRE well addresses the trade-off between fairness, robustness, and model accuracy. In particular, the accuracy decreases when either the fairness constraint or the robustness constraint gets stronger. Nevertheless, the accuracy loss remains insignificant. For example, the accuracy decreases at most 7.46 % for Adult dataset and 10.26 % for COMPAS dataset.

We also measured the number of labels to be flipped as well as the number of adversarial examples to be added by FROC-PRE for the three datasets. The results show that only a small percentage of labels are flipped as well as the adversarial examples are inserted. In particular, for Adult dataset, the algorithm flips 1,995 (6.3%) labels and adds 4,940 (15.6%) adversarial examples. For COMPAS dataset, the algorithm flips 2,395 (17.1%) labels and adds 1,858 (13.3%) adversarial examples. For Hospital dataset, the algorithm flips 810 (2.2%) labels and adds 4,406 (11.9%) adversarial examples.

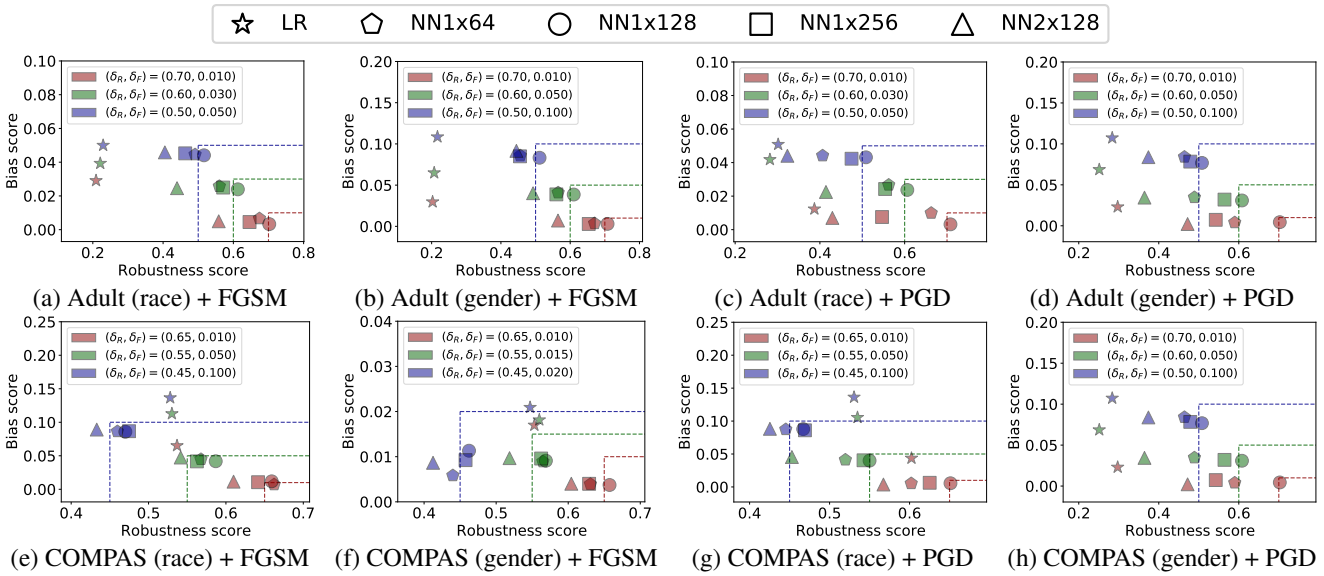
Varying target model. One advantage of FROC-PRE is that it considers a surrogate model, so it can be used for different target classification models. We consider the following five classification models as the target model: (1) logistic regression (LR), (2) neural networks with one hidden layer and 64 neurons (NN1x64), (3) neural networks with one hidden layer and 128 neurons (NN1x128), which is the same as the surrogate model in FROC-PRE, (4) neural networks with one hidden layer and 256 neurons (NN1x256); and (5) neural networks with two hidden layer and 128 neurons on each layer (NN2x128). All the NN models use the same activation function. We train these five models on the training data that is pre-processed by FROC-PRE, and use the original testing data to evaluate the performance of the trained model. We consider different protected attributes and measure fairness and robustness of each classification model, and show the results in Figure 6. We also consider three different settings of the fairness and robustness thresholds for *strong*,



(a) Adult (race) + FGSM (b) Adult (gender) + FGSM (c) Adult (race) + PGD (d) Adult (gender) + PGD

(e) COMPAS (race) + FGSM (f) COMPAS (gender) + FGSM (g) COMPAS (race) + PGD (h) COMPAS (gender) + PGD

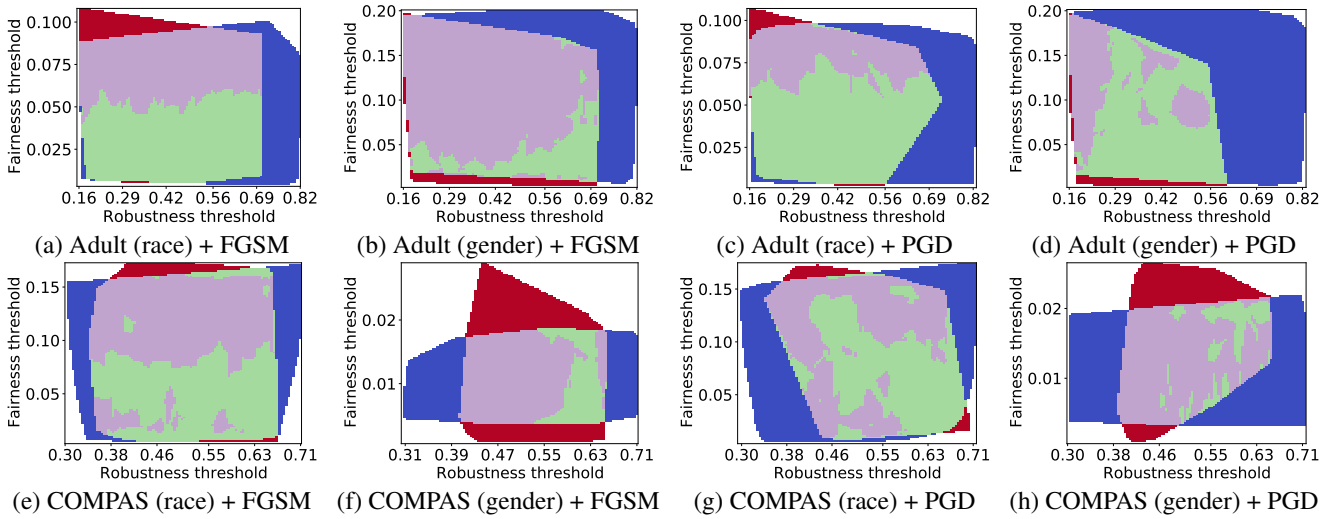
Figure 5: Model fairness, robustness, and accuracy of FROC-PRE method. Accuracy is visualized in colors; light (deep) color indicates higher (lower) accuracy



(a) Adult (race) + FGSM (b) Adult (gender) + FGSM (c) Adult (race) + PGD (d) Adult (gender) + PGD

(e) COMPAS (race) + FGSM (f) COMPAS (gender) + FGSM (g) COMPAS (race) + PGD (h) COMPAS (gender) + PGD

Figure 6: Performance of FROC-PRE method on various classification models. The colored rectangles indicate the δ_F-fairness and δ_R-robustness requirements for different δ_F and δ_R thresholds. The (bias score, fairness score) points that lie either on the edges of the rectangles or inside of the rectangle indicate that the model satisfies δ_F-fairness and δ_R-robustness, otherwise it fails either δ_F-fairness or δ_R-robustness or both. NN1x128 model is the surrogate model.



(a) Adult (race) + FGSM (b) Adult (gender) + FGSM (c) Adult (race) + PGD (d) Adult (gender) + PGD

(e) COMPAS (race) + FGSM (f) COMPAS (gender) + FGSM (g) COMPAS (race) + PGD (h) COMPAS (gender) + PGD

Figure 7: Comparison between FROC-IN and FROC-PRE. Red area: the cases that only FROC-PRE can satisfy both fairness and robustness thresholds; Blue area: the cases that only FROC-IN can satisfy both fairness and robustness thresholds; Purple area: FROC-PRE outperforms FROC-IN in terms of model accuracy; Green area: FROC-IN outperforms FROC-PRE in terms of model accuracy

medium, and *weak* fairness-robustness settings. The details of δ_F and δ_R of these settings can be found in Figure 6. We use the colored rectangles to indicate the δ_F-fairness and δ_R-robustness requirements. Intuitively, the (bias score, fairness score) points that lie either on the edges of the rectangles or inside of the rectangle indicate that the model satisfies δ_F-fairness and δ_R-robustness, otherwise it fails either δ_F-fairness or δ_R-robustness or both. The result for Hospital dataset can be found in Appendix C.

We have two main observations. First, in all settings, the bias score and robustness score of the NN1x128 model (the surrogate model) always satisfies the given δ_F and δ_R thresholds. In other words, the model that satisfies δ_F-fairness and δ_R-robustness on the training data also satisfies δ_F-fairness and δ_R-robustness on the testing data (as required). Second, all NN models have very similar fairness performance. Among these NN models, the NN1x128 model always has the best performance and satisfies both fairness and robustness requirements. This is unsurprising as it is the surrogate model used by FROC-PRE. Nevertheless, the robustness of NN1x64 and NN1x256 models are close to the NN1x128 model as they have similar structure as the NN1X128 model, while the NN2x128 model has the worst robustness among all four NN models. On the other hand, LR model has the worst robustness and fairness performance compared with all the four NN models. This indicates that FROC-PRE is effective for influence estimation especially for those target models of similar architecture (e.g., same number of layers of NN) as the surrogate model. Different number of neurons at these layers does not impact much on the fairness and robustness performance of the target model.

6.5 FRoC-In versus FRoc-Pre

We compare model accuracy, fairness, and robustness of FROC-IN and FROC-PRE. We show the results for Adult and COMPAS datasets in Figure 7. The results for Hospital dataset can be found in Appendix C. For ease of demonstration, we use different colors to illustrate the performance of two algorithms. In particular, the red area shows the cases that FROC-PRE can meet both fairness and robustness thresholds but FROC-IN cannot. The blue area shows the cases that only FROC-IN can meet both fairness and robustness thresholds but FROC-PRE cannot. The purple area shows the cases that FROC-PRE outperforms FROC-IN in terms of model accuracy when they have comparable fairness and robustness performance. And the green area shows the cases that FROC-IN outperforms FROC-PRE in terms of model accuracy under the same fairness and robustness performance.

From the results, we observe that, first, the "winner" of FROC-PRE and FROC-IN varies even on the same dataset but with different protected attributes. This is not surprising as the performance of both methods depends on the data distributions of the whole dataset as well as the distributions of both protected and un-protected groups. Second, when the robustness threshold is very large (i.e., strong robustness requirement), FROC-PRE is more likely to fail to meet both robustness and fairness requirements than FROC-IN. On the other hand, when the fairness threshold is very small (i.e., strong fairness requirement), FROC-IN may fail to meet both robustness and fairness requirements while FROC-PRE can satisfy. In other words, FROC-PRE is more suitable for the strong fairness setting, while FROC-IN is more suitable for the strong robustness setting.

7 Discussions

Incremental learning for FROC-PRE method. One weakness of FROC-PRE is that it has to re-train the model at each iteration for the estimation of influence scores. One possible optimization is to let the model $\mathbf{M}^{(i)}$ at the i-th iteration first inherit the parameters of the model $\mathbf{M}^{(i-1)}$ from the last iteration, and use *incremental learning* techniques [10] that allows remodelling the network in an incremental way without retraining. We can adapt the step-wise updating algorithm in [10] to remodel the network by only computing the pseudoinverse of the flipped label. Since we only flip a small portion labels in each iteration, the incremental learning approach should be cost-efficient.

Adapt FROC methods to other fairness definitions. In this paper, we only consider statistical parity. If we change to other fairness definitions (e.g., equal opportunity [23] and equalized odds [23]), both FROC-IN and FROC-PRE methods can be easily adapted to the new fairness definitions by re-designing the fairness regularizer $\mathbf{F}$ (Equation (13)) for the new fairness definitions. The fairness influence scores of FROC-PRE can be efficiently estimated by taking the gradient of the new fairness regularizer $\mathbf{F}$ (Equation (17)).

Robustness against the data poisoning attacks. In this paper, we only consider the robustness against the evasion attacks. If the attack models change to the data poisoning attacks [5, 46], we can apply the following method to realize both fairness and robustness. First, we identify the poisoned data points and filter them out by the existing methods [12, 31, 42]. Then we apply the existing bias mitigation methods (e.g., [6, 16, 26]) on the cleaned data to meet the fairness requirement. Our claim is that applying bias mitigation after cleaning of poisoned samples will not counteract model robustness, as it will not insert any poisoned samples into the training data. This is different from the sequential method (Section 6.2) that provides robustness against the evasion attack, which is destined to fail as it has to insert adversarial examples which may bring new bias into the model.

8 Related Work

Algorithmic fairness. Algorithmic fairness in ML has caught increasing attention from the ML community [13]. Several competing notions of algorithmic fairness have been recently proposed. These definitions can be categorized into two categories: (1) *Group fairness* that is concerned with the protected groups and requires that some statistic of interest be approximately equalized across groups [7, 16, 23]; and (2) *Individual fairness* [15] that prevents discrimination against individuals and requires similar individuals are treated similarly. In this paper, we focus on group fairness.

Techniques to design bias mitigation algorithms typically identify a fairness notion of interest first and modify a particular point of the ML pipeline to satisfy it. Methodologically, they fall broadly into three categories: (1) *pre-processing*: the bias in the training data is mitigated [6, 16, 26]; (2) *in-processing*: the ML model is modified by adding fairness as additional constraint [7, 19, 47]; and (3) *post-processing*: the results of a previously trained classifier are modified to achieve the desired results on different groups [23]. In this paper, we consider both pre-processing and in-processing methods for bias mitigation.

Robust machine learning. A considerably large amounts of research on adversarial ML and defense strategies have been performed recently. We refer the audience to some excellent surveys [1, 8, 44] of recent developments in robust ML. In this paper, we focus on two types of evasion attacks, namely proposed *Fast Gradient Sign Method* (FGSM) [21] and *Projected Gradient Descent* (PGD) attack [34]. We consider adversarial training as the defense mechanism.

Interaction between fairness and robustness. There is very few study of the interaction between robustness and fairness. Chang *et al.* [9] show that a conflict exists between fairness and robustness - ensuring fairness of ML models can increase the susceptibility of these models to the data poisoning attacks. Their work focuses on the data poisoning attacks during the training phase, while we consider the evasion attack during the inference phase. Furthermore, their goal is to show the impact of fairness on model robustness, while we aim to realize both fairness and robustness simultaneously. Sharma *et al.* [40] investigate the fairness and robustness issues of neural networks for structured data. They measure fairness in terms of the equalized ability to achieve recourse for different groups. Their key idea is two-fold: (1) adjust the average distance to the decision boundary between groups so that the network is more fair with respect to the ability to obtain resources, and (2) increase the average distance of data points to the boundary to promote adversarial robustness. Their fairness definitions and robustness requirements are fundamentally different from ours.

9 Conclusion and Future Work

In this paper, we study the problem of equipping the classification models with both fairness and adversarial robustness against evasion attacks. We design two algorithms, namely FROC-IN and FROC-PRE. FROC-IN is an in-processing method that adds fairness and adversarial robustness as two regularizers to the objective function of the model, while FROC-PRE is a pre-processing method that modifies the training data to remove data bias and add adversarial examples. Our experimental results show that both FROC-IN and FROC-PRE well address the trade-off among fairness, robustness, and model accuracy.

For future work, we will take privacy, another important issue of trustworthy ML, into consideration. We will consider various types of privacy inference attacks (e.g., membership inference attack [18, 41], attribute inference attack [3, 20], and model inversion attack [17]) as well as their impacts on fairness and robustness. We will also address the trade-off between fairness, robustness, privacy, and model accuracy.

References

[1] Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *Ieee Access*, 6:14410–14430, 2018.

[2] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. Machine bias: there's software used across the country to predict future criminals. and it's biased against blacks. propublica 2016, 2016.

[3] Giuseppe Ateniese, Luigi V Mancini, Angelo Spognardi, Antonio Villani, Domenico Vitali, and Giovanni Felici. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *International Journal of Security and Networks*, 10(3):137–150, 2015.

[4] Battista Biggio, Giorgio Fumera, and Fabio Roli. Security evaluation of pattern classifiers under attack. *IEEE transactions on knowledge and data engineering*, 26(4):984–996, 2013.

[5] Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*, 2012.

[6] Toon Calders, Faisal Kamiran, and Mykola Pechenizkiy. Building classifiers with independency constraints. In *2009 IEEE International Conference on Data Mining Workshops*, pages 13–18. IEEE, 2009.

[7] Toon Calders and Sicco Verwer. Three naive bayes approaches for discrimination-free classification. *Data Mining and Knowledge Discovery*, 21(2):277–292, 2010.

[8] Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. Adversarial attacks and defences: A survey. *arXiv preprint arXiv:1810.00069*, 2018.

[9] Hongyan Chang, Ta Duy Nguyen, Sasi Kumar Murakonda, Ehsan Kazemi, and Reza Shokri. On adversarial bias and the robustness of fair machine learning. *arXiv preprint arXiv:2006.08669*, 2020.

[10] CL Philip Chen and John Z Wan. A rapid learning and dynamic stepwise updating algorithm for flat neural networks and the application to time-series prediction. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 29(1):62–72, 1999.

[11] Jiahao Chen, Nathan Kallus, Xiaojie Mao, Geoffry Svacha, and Madeleine Udell. Fairness under unawareness: Assessing disparity when protected class is unobserved. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 339–348, 2019.

[12] Jian Chen, Xuxin Zhang, Rui Zhang, Chen Wang, and Ling Liu. De-pois: An attack-agnostic defense against data poisoning attacks. *IEEE Transactions on Information Forensics and Security*, 2021.

[13] Alexandra Chouldechova and Aaron Roth. The frontiers of fairness in machine learning. *arXiv preprint arXiv:1810.08810*, 2018.

[14] R Dennis Cook and Sanford Weisberg. *Residuals and influence in regression*. New York: Chapman and Hall, 1982.

[15] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226. ACM, 2012.

[16] Michael Feldman, Sorelle A Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and removing disparate impact. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 259–268. ACM, 2015.

[17] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1322–1333, 2015.

[18] Matthew Fredrikson, Eric Lantz, Somesh Jha, Simon Lin, David Page, and Thomas Ristenpart. Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing. In *23rd Security Symposium ({USENIX})*, pages 17–32, 2014.

[19] Gabriel Goh, Andrew Cotter, Maya Gupta, and Michael P Friedlander. Satisfying real-world goals with dataset constraints. In *Advances in Neural Information Processing Systems*, pages 2415–2423, 2016.

[20] Neil Zhenqiang Gong and Bin Liu. You are who you know and how you behave: Attribute inference attacks via users' social friends and behaviors. In *25th {USENIX} Security Symposium ({USENIX})*, pages 979–995, 2016.

[21] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[22] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens Van Der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017.

[23] Moritz Hardt, Eric Price, Nati Srebro, et al. Equality of opportunity in supervised learning. In *Advances in neural information processing systems*, pages 3315–3323, 2016.

[24] Lauren Kirchner Jeff Larson, Surya Mattu and Julia Angwin. Data and analysis for 'machine bias'. https://github.com/propublica/compas-analysis/.

[25] Robin Jia and Percy Liang. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, 2017.

[26] Faisal Kamiran and Toon Calders. Classifying without discriminating. In *2009 2nd International Conference on Computer, Control and Communication*, pages 1–6. IEEE, 2009.

[27] Faisal Kamiran and Toon Calders. Data preprocessing techniques for classification without discrimination. *Knowledge and Information Systems*, 33(1):1–33, 2012.

[28] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, pages 1885–1894. PMLR, 2017.

[29] Ronny Kohavi and Barry Becker. Uci machine learning repository: Adult data set. https://archive.ics.uci.edu/ml/datasets/Adult.

[30] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

[31] Ricky Laishram and Vir Virander Phoha. Curie: A method for protecting svm classifier from poisoning attack. *arXiv preprint arXiv:1606.01584*, 2016.

[32] Yanying Li, Haipei Sun, and Wendy Hui Wang. Towards fair truth discovery from biased crowdsourced answers. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 599–607, 2020.

[33] Chunchuan Lyu, Kaizhu Huang, and Hai-Ning Liang. A unified gradient regularization family for adversarial examples. In *2015 IEEE international conference on data mining*, pages 301–309. IEEE, 2015.

[34] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[35] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *arXiv preprint arXiv:1908.09635*, 2019.

[36] Texas Department of State Health Services. Hospital discharge data use agreement. https://www.dshs.texas.gov/THCIC/Hospitals/Download.shtm.

[37] Cathy O'neil. *Weapons of math destruction: How big data increases inequality and threatens democracy*. Broadway Books, 2016.

[38] Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *arXiv preprint arXiv:1904.12843*, 2019.

[39] Uri Shaham, Yutaro Yamada, and Sahand Negahban. Understanding adversarial training: Increasing local stability of neural nets through robust optimization. *arXiv preprint arXiv:1511.05432*, 2015.

[40] Shubham Sharma, Alan H Gee, David Paydarfar, and Joydeep Ghosh. Fair-n: Fair and robust neural networks for structured data. *arXiv preprint arXiv:2010.06113*, 2020.

[41] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE, 2017.

[42] Jacob Steinhardt, Pang Wei Koh, and Percy Liang. Certified defenses for data poisoning attacks. *arXiv preprint arXiv:1706.03691*, 2017.

[43] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[44] Elham Tabassi, Kevin Burns, Michael Hadjimichael, Andres Molina-Markham, and Julian Sexton. A taxonomy and terminology of adversarial machine learning. *NIST IR*, 2019.

[45] Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Generating adversarial examples with adversarial networks. *arXiv preprint arXiv:1801.02610*, 2018.

[46] Huang Xiao, Battista Biggio, Blaine Nelson, Han Xiao, Claudia Eckert, and Fabio Roli. Support vector machines under adversarial label contamination. *Neurocomputing*, 160:53–62, 2015.

[47] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P Gummadi. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In

Proceedings of the 26th International Conference on World Wide Web, pages 1171–1180, 2017.

A Details of Real-World Datasets

We use the following datasets in the experiments: (1) **Adult dataset** [29] includes 45,222 instances and 14 attributes (such as age, gender, education, marital status, occupation, working hours, and native country) that describe the information about individuals from the 1994 U.S. census. The prediction task is to determine whether a person makes over $50K annually. (2) **COMPAS dataset** [24] contains criminal history, jail and prison time, demographics and COMPAS (which stands for Correctional Offender Management Profiling for Alternative Sanctions) risk scores for defendants from Broward County, Florida. The prediction task is to infer a criminal defendant's likelihood of becoming a recidivist (i.e., a criminal who re-offend) within two years. (3) **Hospital dataset** [36] is released by the Texas Department of State Health Services. It contains records of inpatient stays in some health facilities. The features include types of external causes of injury, diagnosis, the procedures the patient underwent, and demographic information such as gender, age, and race. The classification task is to predict the patient's main procedure. We categorize the main procedures into two groups (corresponding to the prediction labels): cardiology and pulmonology.

B Mutual Impacts between Fairness and Robustness

In this section, we investigate the research question: *When fairness or robustness is enforced alone, can it promote/hinder the other as a side-effect?* through empirical study. All results are reported as the average of twenty repeats.

Fairness and robustness on original clean and biased data. First, we measure testing accuracy, fairness, and robustness of the target model on the three datasets, and show the results in Table 2. It can be observed that, first, the target model has noticeable bias ($\mathcal{S}_B \in [0.09, 0.19]$) in its prediction results. Second, the target model is not robust against both FGSM and PGD attacks ($\mathcal{S}_R \in [0.146, 0.343]$).

Impact of fairness on robustness. To evaluate the impact of fairness on robustness, we measure the difference in the robustness of a model before and after applying the two fairness-enhancing methods (RW and DIR), which is calculated as the change C_R of the robustness score:

$$C_R = \mathcal{S}_R^F - \mathcal{S}_R^O,$$

where $\mathcal{S}_R^O$ and $\mathcal{S}_R^F$ indicate the robustness score of the original model and the fair model. Positive (negative, resp.) C_R indicates that fairness promotes (hinders, resp.) robustness.

Data	PA	Acc.	$\mathcal{S}_B$	$\mathcal{S}_R$	
				FGSM	PGD
Adult	Race	0.8472	0.0904	0.1579	0.1477
	Gender	0.8473	0.1933	0.1577	0.1460
COMPAS	Race	0.7096	0.1523	0.3015	0.2867
	Gender	0.7043	0.0194	0.3035	0.2834
Hospital	Race	0.6593	0.0153	0.3407	0.3328
	Gender	0.6572	0.0175	0.3430	0.3299

Table 2: Testing accuracy (Acc.), fairness ($\mathcal{S}_B$), and robustness ($\mathcal{S}_R$) of three datasets with the average of 20 repeats (PA = protected attribute)

The results are shown in Table 3. The main observation is that the value C_R is negligible for most of the settings. This indicates that enforcing fairness on models has little impacts on model robustness.

Impact of robustness on fairness. To evaluate the impact of robustness on fairness, we compare the bias score of models trained with and without using the adversarial training. When using adversarial training in Equation (3), we also try different λ values to adjust the weight of the adversarial regularizer. We measure the change of the bias score C_B as the following:

$$C_B = \mathcal{S}_B^R - \mathcal{S}_B^O,$$

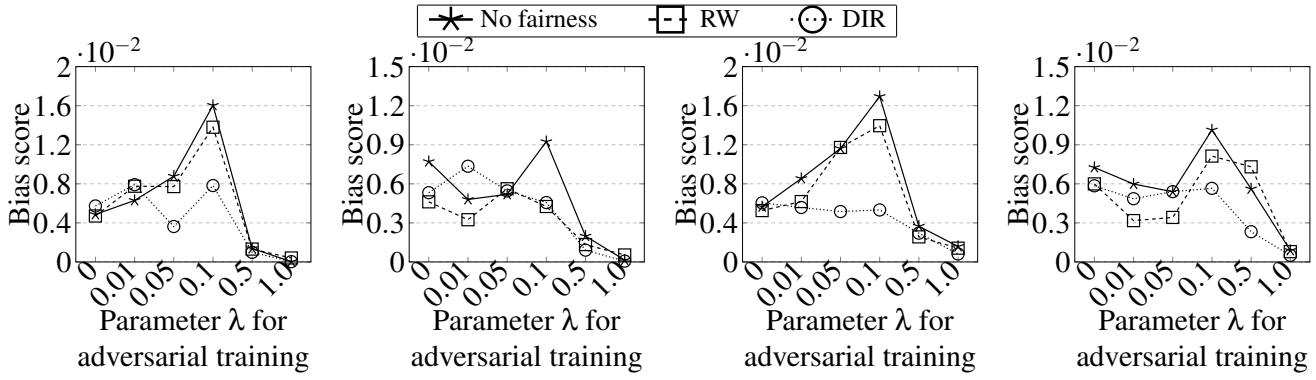
where $\mathcal{S}_B^O$ and $\mathcal{S}_B^R$ indicate the bias score of the original model on the clean data and the one trained on data with adversarial examples. Since lower $\mathcal{S}_B$ indicates less bias, a negative (positive, resp.) C_B value indicates that improvement of robustness improves (harms, resp.) fairness.

The results are shown in Table 4. We observe that the C_B value for most of the settings is small (no more than 0.0228). Such small C_B value indicates that adversarial examples do not impact the fairness of the model.

C Experiment Results (Hospital Dataset)

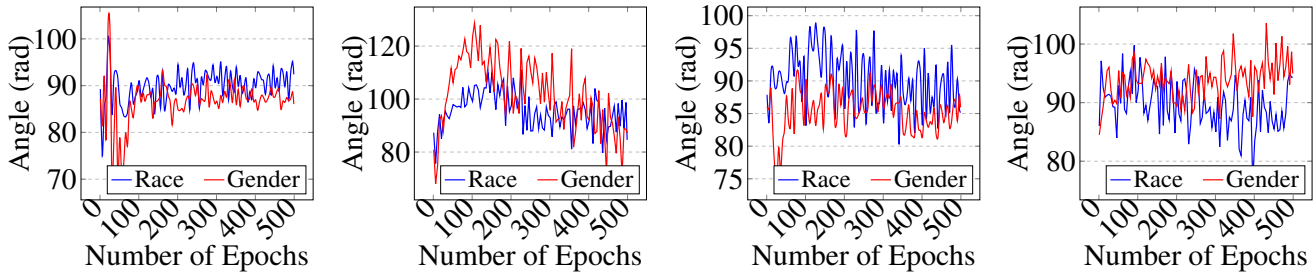
Performance of sequential method Performance of sequential method on Hospital dataset is shown in Figure 8. The pattern on Hospital dataset is different from other datasets: the bias score for no fairness setting decreases when λ becomes larger. This pattern is because the bias score on Hospital dataset is mainly caused by overfitting, and the increment of λ enlarges the weight of the robustness regularizer, which helps eliminate the overfitting effect. Therefore the bias score decreases even without fairness mitigations.

Interaction between fairness and robustness regularizers during training. Figure 9 shows the angle between fairness and robustness regularizers during training. Similar to Adult and COMPAS dataset, the angle between $\nabla_\theta \mathbf{F}$ and $\nabla_\theta \mathbf{R}$ is in the range of $[70°, 110°]$ after the initial 20 epochs for all the settings of Hospital dataset, and then keeps stable when it is close to $90°$.



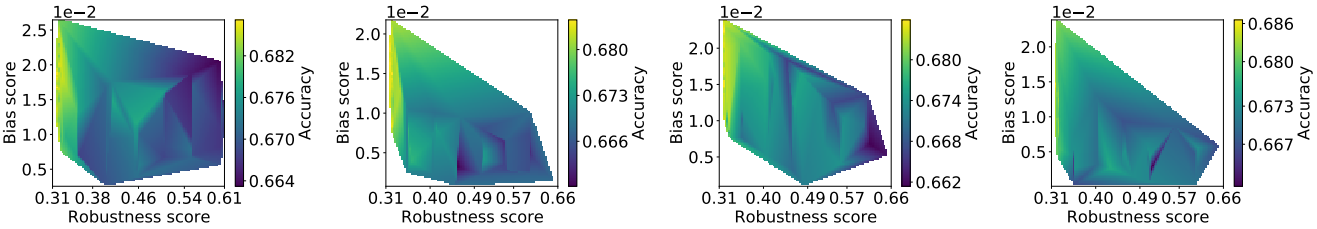
(a) Hospital (race) + FGSM (b) Hospital (gender) + FGSM (c) Hospital (race) + PGD (d) Hospital (gender) + PGD

Figure 8: Performance of sequential method (Hospital dataset). The training dataset is pre-processed by either Reweighing (RW) [27] or Disparate Impact Remover method (DIR) [16]



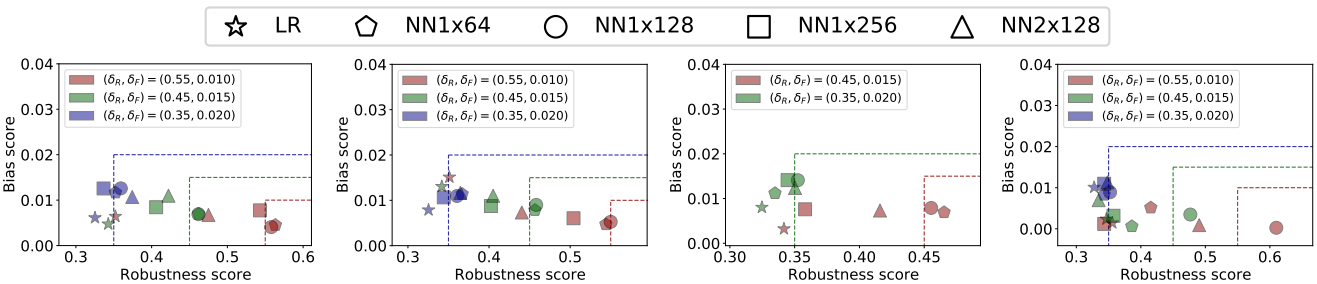
(a) Hospital (race) + FGSM (b) Hospital (gender) + FGSM (c) Hospital (race) + PGD (d) Hospital (gender) + PGD

Figure 9: Interaction between fairness and robustness regularizers during training of FROC-IN, measured as the angle between the gradients of the two regulariziers (Hospital dataset)



(a) Hospital (race) + FGSM (b) Hospital (gender) + FGSM (c) Hospital (race) + PGD (d) Hospital (gender) + PGD

Figure 10: Model fairness, robustness, and accuracy of FROC-PRE (Hospital dataset). Accuracy is visualized in colors; light (deep) color indicates higher (lower) accuracy



(a) Hospital (race) + FGSM (b) Hospital (gender) + FGSM (c) Hospital (race) + PGD (d) Hospital (gender) + PGD

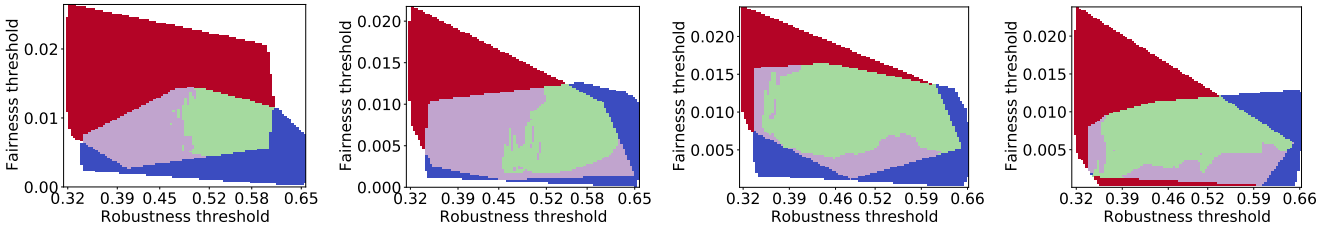
Figure 11: Performance of FROC-PRE on various downstream classification models (Hospital dataset).

Dataset	Sensitive Attribute	Attack	*Reweighing*	*Disparate Impact Remover* (with repair level p)				
				$p = 0.2$	$p = 0.4$	$p = 0.6$	$p = 0.8$	$p = 1.0$
Adult	Race	FGSM	-0.0002	-0.0005	-0.0002	-0.0003	-0.0008	0.0003
		PGD	0.0001	0.0003	-0.0003	-0.0004	0.0002	-0.0002
	Gender	FGSM	-0.0045	-0.0011	-0.0004	0.0004	-0.0011	-0.0005
		PGD	-0.0038	0.0000	-0.0011	0.0002	-0.0010	-0.0007
COMPAS	Race	FGSM	-0.0001	-0.0019	0.0013	0.0025	0.0009	0.0004
		PGD	0.0007	-0.0002	0.0003	0.0000	-0.0006	-0.0008
	Gender	FGSM	0.0014	-0.0014	-0.0008	-0.0010	-0.0000	-0.0005
		PGD	0.0007	0.0006	0.0013	0.0006	0.0009	0.0001
Hospital	Race	FGSM	0.0005	0.0012	0.0009	0.0008	0.0007	-0.0008
		PGD	0.0004	-0.0006	0.0005	0.0008	0.0008	0.0011
	Gender	FGSM	0.0003	0.0016	0.0008	-0.0014	0.0001	-0.0028
		PGD	-0.0002	0.0008	-0.0016	-0.0004	0.0013	0.0001

Table 3: Impact of fairness on robustness. Negative (positive, resp.) values indicate fairness enhances (hurts. resp.) robustness.

Dataset	Sensitive Attribute	FGSM (with loss weight λ)					PGD (with loss weight λ)				
		$\lambda = 0.01$	$\lambda = 0.05$	$\lambda = 0.1$	$\lambda = 0.5$	$\lambda = 1.0$	$\lambda = 0.01$	$\lambda = 0.05$	$\lambda = 0.1$	$\lambda = 0.5$	$\lambda = 1.0$
Adult	Race	-0.0023	0.0046	0.0101	-0.0118	-0.0117	0.0024	0.0063	0.0041	-0.0118	-0.0136
	Gender	0.0030	0.0019	0.0034	-0.0201	-0.0114	0.0022	-0.0004	0.0009	-0.0156	-0.0228
COMPAS	Race	-0.0031	-0.0026	0.0032	0.0117	0.0071	0.0017	-0.0001	0.0090	0.0108	0.0183
	Gender	-0.0030	-0.0030	-0.0046	0.0020	0.0004	0.0034	0.0020	0.0023	0.0029	0.0049
Hospital	Race	0.0014	0.0039	0.0112	-0.0035	-0.0048	0.0029	0.0060	0.0113	-0.0020	-0.0041
	Gender	-0.0029	-0.0025	0.0015	-0.0057	-0.0076	-0.0013	-0.0019	0.0029	-0.0016	-0.0064

Table 4: Impact of robustness on fairness. Results are the average of 20 repeats. Negative (positive) values indicate the fairness mitigation enhances (hinders) robustness.



(a) Hospital (race) + FGSM (b) Hospital (gender) + FGSM (c) Hospital (race) + PGD (d) Hospital (gender) + PGD

Figure 12: Comparison between FROC-IN and FROC-PRE on Hospital dataset. Red area: the cases that only FROC-PRE can satisfy both fairness and robustness thresholds; Blue area: the cases that only FROC-IN can satisfy both fairness and robustness thresholds; Purple area: FROC-PRE outperforms FROC-IN in terms of model accuracy; Green area: FROC-IN outperforms FROC-PRE in terms of model accuracy

Trade-off among fairness, robustness, and model accuracy. Figure 10 shows the results of our FROC-PRE for Hospital dataset. Similar to the results of FROC-IN (Figure 3), FROC-PRE well addresses the trade-off among fairness, robustness, and model accuracy. The accuracy decreases at most 1.31 % for Hospital in all settings.

Varying downstream models. The result of Hospital dataset is shown in Figure 11 The main observations are similar to the other two datasets. All NN models have very similar fairness performance: The NN1x128 model used by FROC-PRE always has the best robustness; The robustness of NN1x64 and NN1x256 models are close to the NN1x128 model as they have similar structure as the NN1X128 model; The NN2x128 model has the worst robustness among all four NN models. Furthermore, LR model has the worst robustness and fairness performance compared with all the four NN models. Note that for Hospital (race) + PGD setting (Figure 11 (c)), a bias score that is lower than 0.15 is not reachable, so we only choose two parameter settings in this figure.

FROC-PRE versus FROC-IN From the results in Figure 12, we observe similar patterns as the other two datasets: First, the "winner" of FROC-PRE and FROC-IN varies even on

the same dataset but with different protected attributes. Second, when the robustness threshold is very large (i.e., strong robustness requirement), FROC-IN is more likely to fail to meet both robustness and fairness requirements than FROC-PRE. On the other hand, when the fairness threshold is very small (i.e., strong fairness requirement), FROC-PRE may fail to meet both robustness and fairness requirements more frequently than FROC-IN.