FairNeuron: Improving Deep Neural Network Fairness with Adversary Games on Selective Neurons

Xuanqi Gao
Xi'an Jiaotong University
Xi'an, China
gxq2000@stu.xjtu.edu.cn

Juan Zhai
Rutgers University
United States
juan.zhai@rutgers.edu

Shiqing Ma
Rutgers University
United States
shiqing.ma@rutgers.com

Chao Shen
Xi'an Jiaotong University
Xi'an, China
chaoshen@mail.xjtu.edu.cn

Yufei Chen
Xi'an Jiaotong University
Xi'an, China
yfchen@sei.xjtu.edu.cn

Qian Wang
Wuhan University
Wuhan, China
qianwang@whu.edu.cn

ABSTRACT

With Deep Neural Network (DNN) being integrated into a growing number of critical systems with far-reaching impacts on society, there are increasing concerns on their ethical performance, such as fairness. Unfortunately, model fairness and accuracy in many cases are contradictory goals to optimize during model training. To solve this issue, there has been a number of works trying to improve model fairness by formalizing an adversarial game in the model level. This approach introduces an adversary that evaluates the fairness of a model besides its prediction accuracy on the main task, and performs joint-optimization to achieve a balanced result. In this paper, we noticed that when performing backward propagation based training, such contradictory phenomenon are also observable on individual neuron level. Based on this observation, we propose FairNeuron, a DNN model automatic repairing tool, to mitigate fairness concerns and balance the accuracy-fairness trade-off without introducing another model. It works on detecting neurons with contradictory optimization directions from accuracy and fairness training goals, and achieving a trade-off by selective dropout. Comparing with state-of-the-art methods, our approach is lightweight, scaling to large models and more efficient. Our evaluation on three datasets shows that FairNeuron can effectively improve all models' fairness while maintaining a stable utility.

KEYWORDS

fairness, path analysis, neural networks

ACM Reference Format:
Xuanqi Gao, Juan Zhai, Shiqing Ma, Chao Shen, Yufei Chen, and Qian Wang. 2022. FairNeuron: Improving Deep Neural Network Fairness with Adversary Games on Selective Neurons. In *44th International Conference on Software Engineering (ICSE '22), May 21–29, 2022, Pittsburgh, PA, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3510003.3510087

ICSE '22, May 21–29, 2022, Pittsburgh, PA, USA

ACM ISBN 978-1-4503-9221-1/22/05…$15.00
https://doi.org/10.1145/3510003.3510087

1 INTRODUCTION

Deep neural networks (DNNs) are gradually adopted in a wide range of applications, including image recognition [37], self-driving [19], and natural language processing [38, 39]. One of the most trendy applications is decision-making systems, which requires a high utility DNN with fairness. As examples, artificial intelligent (AI) judge [8] or human resource (HR) [9] try to judge who should get a loan or interview. These systems should provide objective, supposedly consistent decision based on the given data, although there are often societal bias in these data [59]. We wish these systems could counteract unfair decision made by humans, but they still exhibit unfair behavior which affects individuals belonging to specific social subgroups. The COMPAS system is an example. It predicts recidivism of pretrial offenders [23], and continues to make decisions that favor Caucasians compared to African-Americans. Such bias has made very negative societal impacts. Therefore, it is crucial to have systematical methods for automatically fixing fairness problems in a given DNN model.

Intuitively, fairness problems happen when a model tends to make different decision for different instances which only differentiated by some sensitive attributes, such as age, race and gender. Depending on specific tasks, the protected or sensitive attributes can vary. Similarly, there are different fairness notations defined in existing DNN literature, e.g., group fairness [28], individual fairness [26], and max-min fairness [36, 45]. According to existing study [23, 43], these fairness definitions are correlated with each other. In practice, we usually consider a few representative ones, i.e., demographic parity, demographic parity rate and equal opportunity. In this paper, we also consider these.

Existing DNN training frameworks, e.g., TensorFlow and PyTorch [10, 52], have provided no support for fairness problems detection and fixing. Some other works try to fix other model problems [47, 48, 72]. There are existing fairness fixing frameworks, such as FAD [11] and Ethical Adversaries [25] that try to provide such functionality. Based on the observations that optimizing accuracy and fairness can be contradictory goals in training, these frameworks introduce an adversary that monitors the fairness of the current training. When fairness issues are detected, they solve it by various methods, e.g., data augmentation, that is, leveraging the adversary model to generate adversary examples which help fix the unfair problem and using them as part of the new training data. Similar to generative adversary networks, training such

an adversary can be time-consuming and challenging. It has a lot of practical problems such as mode collapse [22], which is hard to solve. Moreover, such methods usually require using a more complex model training protocol, which is heavyweight.

We observe that the essential challenge of fixing model fairness is that optimization on accuracy only can lead to the selection of the usage of sensitive attributes. For example, an AI HR that uses the sensitive attributes gender as an important feature will be biased. Moreover, such feature selection happen in certain neurons or paths, which is different form the ones using all features or distinguishable features. And such paths/neurons take a small portion of the whole network, otherwise, the network will have low accuracy for all samples. Based on our observations, we proposed FairNeuron, a fairness fixing algorithm that detects and repairs potential DNN fairness problems. It works by first identify *conflict paths* with a neural network slicing technique. Conflict paths refer to the paths that contain a lot of neurons that select sensitive attributes to make predictions rather than distinguishable ones. Then, we leverage such paths to cluster samples by measuring if they can trigger the selection of sensitive attributes. Lastly, we retrain the model by selective retraining. That is, for samples that can cause the model to select sensitive attributes as main features to make predictions, we enforce the DNN to reconsider this by muting other neurons that are not in the conflict paths. By doing so, the conflict path neurons have to consider all features, otherwise, it will very low accuracy on other samples. This helps remove the impacts of biased samples, and fix the fairness problem.

FairNeuron has been implemented as a self-contained toolkit. Our experiments on three popular fairness datasets show that FairNeuron improves twice fairness performance and takes one-fifth usage of training time on average than state-of-the-art solution, Ethical Adversaries [25]. Note that FairNeuron only relies on lightweight procedures like path analysis and dropout, which makes it much more effective and scalable than existing methods.

In summary, we make the following main contributions:

- We propose a novel model fairness fixing frameworks. It avoids training an adversary model, and does not require modifying model training protocol or architecture. It also features lightweight analysis and fixing, leading to high efficiency repairing.
- We develop a prototype FairNeuron based on the proposed idea, and evaluate it with 3 popular public datasets. The evaluation results demonstrate that FairNeuron can effectively and efficiently improve fairness performance of models while maintaining a stable utility. On average, the fairness performance DP can be improved by 57.65%, which is 20% higher than that of state-of-the-art adversary training based method, Ethical Adversaries.
- Our implementation, configurations and collected datasets are available at [32].

Roadmap. This paper is organized as follows. Section §2 presents the necessary background on fairness notions and fixing algorithms. In Section §3, we discuss FairNeuron in detail. Section §4 shows our experiment setup and results. We review related works in §5 and conclude this paper in Section §6.

Threat to Validity. FairNeuron is currently evaluated on 3 datasets, which may be limited. Similarly, there are configurable parameters used in FairNeuron, and even though our experiments show that they are good enough to achieve high fixing results, this may not hold when the size of model is significantly larger or smaller. Besides, we assume that most samples activate a limited number of paths, and most paths are activated by samples with certain features. This has been observed by existing works [56, 63]. We also empirically validate this assumption in §4.2. However, it is possible that this assumption may not hold for some models. To mitigate these threats, all the original and repaired training scripts, model architecture and training configuration details, implementation including dependencies, and evaluation data are publicly available at [32] for reproduction.

2 BACKGROUND AND MOTIVATION

2.1 Fairness

Depending on concrete task specifications, fairness can have different notations [21]. These notions can be categorized into two groups: individual fairness [26, 67], which measures if individuals in the dataset is treated equally by the learned model; and group fairness [28, 35], which concerns about whether subpopulation with different sensitive attributes are treated equally. For example, for an online shopping recommendation system, all customers in the dataset should be treated equally, which asks for individual fairness. For an AI powered hiring system, applicants with sensitive attributes (e.g., different genders) should be treated equally, which is a typical case of group fairness.

Before discussing different fairness notations, we first define a set of notations. We denote the sensitive attribute as S and other observable insensitive attributes as A. We assume that the subpopulation with $S = 1$ is the disadvantaged group, and the privileged group is the subpopulation with $S = 0$. Also, we represent the true label as Y, and the predicted output, i.e., positive/negative as $\hat{Y}$ which is a random variable depending on attributes S and A. $\hat{Y} = 1$ and $\hat{Y} = 0$ are the positive and negative outcomes, respectively. Following such notations, we can define commonly used different fairness notations as follows:

Demographic parity (DP). Demographic parity, or statistical parity, is one of the earliest definitions of fairness [26]. It views fairness as different subpopulations (i.e., $S = 0$ and $S = 0$) should have an equal probability of being classified to the positive label. Formally, demographic parity measures the probability differences between different groups:

$$DP = \left| P(\hat{Y} = 1 \mid S = 0) - P(\hat{Y} = 1 \mid S = 1) \right| \tag{1}$$

In an ideal case, we say that a model is when $DP = 0$, which indicates that the prediction output $\hat{Y}$ and sensitive attribute S are statistically independent. If so, the output is not affected by the sensitive attribute, and hence the model is not biased towards certain values of the sensitive attribute showing fairness in prediction. In practice, $DP = 0$ is hard to get and we view a model as fair when $DP \leq \epsilon$ where ϵ is a threshold value that is determined by real world tasks and requirements.

Demographic parity ratio (DPR). Demographic parity ratio, or disparate impact, is similar to demographic parity. The key difference is that it represents the equality or similarly of prediction on different groups as a ratio (instead of a substitution). Formally, it is defined as:

$$DPR = \frac{P(\hat{Y} = 1 \mid S = 1)}{P(\hat{Y} = 1 \mid S = 0)} \tag{2}$$

Like demographic parity, $DPR = 1$ indicates a fair model in the ideal case, and in practice, we say a model is fair when $DPR \geq \tau$ where τ is the fairness threshold. Moreover, it also focuses on the probability of different groups being classified to the positive label. The key difference is that DPR measures the differences in a ratio. This is because its origins are in legal fairness considerations for selection procedures which the Pareto principle, a.k.a., the 80% rule, is commonly used [28]. To make a direct comparison with 80%, DPR calculated the ratio instead of substitution.

Equal opportunity (EO). A limitation of DP and DPR is that they do not consider potential differences in compared subgroups. Equal opportunity (EO) overcomes this by making use of the FPR (false positive rate) and TPR (true positive rate) between subgroups [35]. Formally, EO is defined as:

$$EO = \left| P(\hat{Y} = 1 \mid S = 0, Y = 1) - P(\hat{Y} = 1 \mid S = 1, Y = 1) \right| \tag{3}$$

A model achieves EO fairness when $EO = 0$, namely, the prediction is (conditional) independent of the sensitive attribute S. In practice, we say an model is EO fair when $EO \leq \nu$ and here, ν is the fairness threshold value.

Besides these discussed notions, there are many other fairness definitions, such as fairness through unawareness (FTU) [44], disparate treatment [15], disparate mistreatment [65], counterfactual fairness [44], ex-ante fairness and ex-post fairness [29], etc. Friedler et al. [30] compared different notations and measured their correlations on the Ricci and Adult datasets. Results show that different notations have strong correlations with each other. As a result, most work usually pick a few representative ones. Following existing related work, we choose three most common notations, i.e., DP, DPR [26], and EO [35] in our study.

2.2 Improving DNN Fairness

Many machine learning algorithms including DNNs suffer from the bias problem. Namely, the model can make a decision based on wrong attributes. For example, a biased hiring AI may make admissions based on applicants' gender information. Such issues can be caused by the biased training data or the algorithm itself. DNN has shown to be a biased algorithm, and potentially trained DNN models can make unfair predictions despite its high accuracy. This can lead to severe problems especially when DNNs are becoming more and more popular including applications like AI judge [8], AI based authentication, AI HR [9], etc. For example, COMPAS, a popular system that predicts the risk of recidivism, claimed that "black people re-offend more" [3]; the first beauty contest robot, Beauty.AI [2], "found dark skin unattractive" [7]; and the Microsoft chatbot Tay became a racist and sex-crazed neo-Nazi [4]. Biased AIs in such systems can lead to severe ethical concerns, potentially threatening our daily life and economy. As a response to this issue, existing work has proposed methods to improve DNN fairness by removing such bias.

FAD. Adel et al. [11] proposed a fair adversarial framework FAD, which leverages gradient reversal [31] (which acts as an identity function during forward propagation and multiplies its input by -1 during back propagation) to fix model fairness problems. The authors introduced an adversarial network to encode fairness into the model: a predictor network $\mathcal{F}_P$ and an adversary network $\mathcal{F}_A$. The goal of the predictor is to maximize accuracy in $\hat{Y}$ while the adversary network tries to maximize fairness in protected attribute S. For fairness fixing, we need a new model architecture which can: (i) predict the true label Y, and (ii) not be able to predict the sensitive attribute S:

$$L_{\mathcal{F}_P} = L_{CE} - \alpha L_{\mathcal{F}_A} \tag{4}$$

where $L_{\mathcal{F}_P}, L_{CE}, L_{\mathcal{F}_A}$ denote the predictor loss, predictor logistic loss (a.k.a., CE loss) and the adversary logistic loss, respectively. The hyperparameter α regulates the accuracy-fairness trade off. After that, it uses a post-training process to align TP (true positive) and FP (false positive) across all classes by adjusting class-specific threshold values of logits with a ROC analysis introduced by Hardt et al. [35].

Ethical Adversaries. Delobelle et al. [25] proposed the ethical adversaries framework to solve the fairness problem. The framework has two parties, the external adversary, a.k.a., the feeder, and the reader, which represents the protected attribute S. It is an iterative training procedures, during which each party interacts with each other. The reader is trained with the target label at the same time, and each time, it evaluates if the training has bias or not. If so, it propagates the related gradient back to the network. The feeder can be viewed as a data augmenter which performs evasion attacks to find adversarial examples that can be used in the adversarial training. During this adversarial training, the target label (i.e., main task of the model) and the fairness goal is adjusted by a hyperparameter λ, which is similar to the FAD framework.

Pre-/Post-processing Methods. FAD and Ethical Adversaries are online methods which solves the fairness issue during training. There are other methods that leverages pre-processing or post-processing to solve this problem, e.g., reweighing [41], and reject option classification (ROC [42]). Reweighing assigns different weights to input samples in different group to make the dataset discrimination-free (pre-processing). ROC gives favorable outcomes to unprivileged groups and unfavorable outcomes to privileged groups in a confidence band around the decision boundary with the highest uncertainty (post-processing).

2.3 Motivation and Basic Idea

Limitations of existing work. Existing work has a few limitations. Firstly, they introduce another model as the adversary in the training procedure. Inheriting from existing adversarial networks, training such models is not easy. Problems like mode collapse [22], failing to converge [50], and vanishing gradients are quite common in such a model structure. This will require extra efforts in solving such problems. Secondly, there is no guarantee that training such adversary networks will always converge for now. There is a theoretical guarantee that GAN (generative adversary network)

will converge, despite its practical difficulty. As a minimax game, training such GAN models will converge when it achieves the Nash equilibrium [33]. However, FAD and Ethical Adversaries empirically observe that model accuracy and fairness may conflict with each other in some cases and may not conflict with each other in other cases. On one hand, this shows that there exist models that are both accurate and fair. On the other hand, it also indicates that the designed adversary training is not a zero-sum game, and there is no guarantee to show the existence of Nash equilibrium in this game. As a result, existing work can fail to converge when training the model because the game has no solutions. Empirical results confirm this conclusion. §4.2 reports that FAD may exacerbate fairness problem. As Table 4 shown, FAD results in decreasing of DPR on Census and increasing of EO on COMPAS, which mean the fairness problems has not been mitigated from these perspectives. Elazar and Goldberg made an empirical observation on leakage of protected attributes for neural networks trained on text data, which can also demonstrate this conclusion [27].

Why bias happens in a DNN training? Based on existing literatures and our experiences, we make a few key observations that are important for us to develop our method.

Observation I: Optimizing accuracy and different fairness objectives can be contradictory to each other, but not always. Existing work [11, 25] has shown that accuracy and different fairness goals (e.g., DP, DPR and EO), including different fairness goals themselves, can be contradictory to each other. This is the reason why some models with high accuracy are highly biased: when optimizing during training, directions with higher accuracy gain may be contradictory to directions with higher fairness. The good news is that existing work has empirically demonstrate that it is possible to train a model with high accuracy and fairness at the same time [25].

Observation II: A neuron represents a combination of different features, and model bias indicates that the model focuses on certain features that it should not. As a general understanding of DNNs, each neuron in the network is extracting features from the input. Form the input layer to the final prediction layer, the extracted features are becoming more and more abstract. Each neuron is representing a set of features it receives from the previous layer, and weights can help it determine which set of features are more important compared with others. A model is biased indicates that a model is focusing on the wrong features, e.g, AI judeges should be affected by senstive attributes like genders. For example, a hiring AI is biased on gender when it selects gender feature rather than others as an important factor to decide if a candidate can get an interview. Notice that such importance is represneted by weights in the DNN.

Now, we can use our observations to explain why bias happens in training a deep neural network. When training a DNN, the optimizer tries to pick important features based on gradient information. When updating individual neurons, it may encounter cases where the fairness and accuracy optimization subjects are pointing to different directions. If it only considers accuracy as its training goal, it will select the direction that optimizes the accuracy the most which can lead to low fairness. Furthermore, we know that the gradient information is calculated based on given samples. If we are able to detect such samples, we can potentially fix the problem by enforcing the optimizer to pick the correct set of features.

Our idea. Based on our observations, we argue that it is not necessary to introduce an adversary that detects the potential conflicts of optimizing the accuracy and fairness. Instead, we first monitor the training process to detect neurons whose accuracy and fairness optimization get conflicts with each other. Then, we identify samples that causes such contradictory optimizations. Lastly, we enforce the optimizer to decide a balanced optimal direction that optimizes both accuracy and fairness. By doing so, we remove the need of introducing an adversary. It simplifies the training procedure and is more lightweight compared with existing solutions.

3 DESIGN

3.1 Overview

Workflow. Figure 1 presents the workflow of FairNeuron. It takes a biased model and its training data as inputs, and outputs a fixed model. Firstly, FairNeuron performs *neural network slicing*, which detects neurons and paths that have contradictory optimization directions. Notice that because of the dense connections of DNN, such neurons are typically connected with each, passing the biased features from one layer to the next. As such, we do this in the path granularity. In this step, we leverage a neuron slicing technique which performs a differential analysis to identify the target paths. Next, we leverage such paths to identify the samples that cause such effects, known as the *sample clustering*. After this step, we can separate the samples into two clusters, biased data samples and benign samples. Lastly, we perform *selective retraining* to enforce the model to learn unbiased features. Essentially, for samples in different clusters, we have different training strategies. For samples in the benign data cluster, we do not change anything, while for samples in the biased cluster, we enforce detected neurons to consider a larger set of features and weigh them to learn *all* features that are important for prediction rather than the biased ones.

Algorithm. The overall algorithm of FairNeuron is presented in Algorithm 1, denoted as Procedure `FairNeuron`. As mentioned before, it takes a biased model *BiasedModel* and training dataset *TrainDataset* as inputs, and outputs a fixed model, referred to as *NewModel* in the algorithm. In the main algorithm, FairNeuron analyze the relationship between dataset and model by getting activation paths of each input sample (line 1-5). After acquiring path information, FairNeuron groups the training dataset into two parts (line 6). The first one is consists of benign samples whose activation paths are clustered by samples, and the second one is consists of biased samples and corresponding paths. Then FairNeuron performs different training strategies on them, it deactivates dropout layers when training benign samples and activated them when training biased samples.

3.2 Neural Network Slicing

In Neural Network Slicing, we try to find paths and neurons that contain the optimizer finds contradictory optimization directions for accuracy and fairness. Figure 2 shows the neural network slicing method of FairNeuron. The input of this algorithm is the training dataset and the biased model to fix, a neural network which has already learned the weights based on a training dataset. We will use this example to demonstrate how it works in this section.

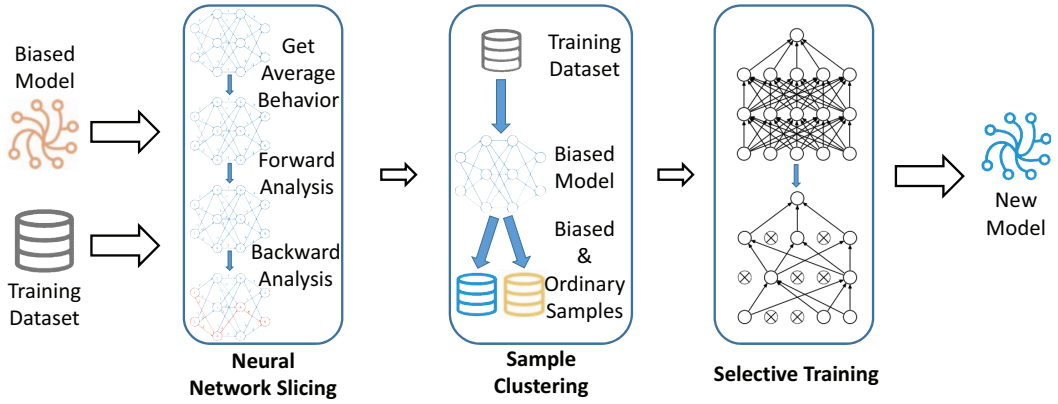


Figure 1: Overview of FairNeuron.

First, FairNeuron gets the *average behavior of a neuron*, by leveraging its activation values. The behavior of a neuron can be represented in many ways, and in FairNeuron, we use the most simple and naive representation, its averaged activation value. Specifically, FairNeuron calculates the average activation values of the neuron for a given dataset, which is usually the training dataset.

Then, FairNeuron performs a *forward analysis* to understand the diversity of a neuron behavior. Similar to the first step, we also use the activation values of a neuron to represent the behavior of the neuron. In this step, we feed individual inputs to the DNN, and record the activation value differences between the average activation and the value for this concrete input. By doing so, we can estimate the contributions of each neuron to the output for a given sample. This helps us to identify neurons that contain biased features.

Afterwards, we obtain paths that contain biased features. Notice that a DNN is a highly connected network, and as a layered structure, behaviors of a single layer will be passed to the next layer. Because of this, biased features will be accumulated in this network, and as a result, neurons in the last few layers will contribute a lot to the biased prediction. On the other hand, these neurons do not denote the root cause of such bias. To completely fix the neuron network, it is important to identify the whole chain of such propagation. So we comprehensively consider neurons and synapses and calculate their contributions, and backtrack these contributions in the network. We show the detail of this phase in the procedure *GetActivationPath* in Algorithm 1. Starting from the output neuron, we iteratively compute the contributions of the previous neurons (line 19-22), which is similar to the backward propagation. Then, we sort them in descending order (line 23), and add the key synapses and corresponding neurons into the path set (line 24-29). To determine if a synapse is a key synapse or not, we need to calculate whether the sum of all the synapses that are connected to the same successor neuron is still less than the threshold. The threshold is determined by the activation value of subsequent neurons and the hyperparameter γ.

Lastly, we identify conflict paths, namely paths that contain features causing the biased prediction. Based on our observations, we know that when making predictions, the model uses the benign feature set to make predictions for benign samples and use the biased feature set to make predictions for biased samples. Considering that a neuron represents a set of features, we know that biased samples are activating neurons/paths that are different from the others. Notice that biased paths/neurons takes a relatively small portion of the whole neural network. Otherwise, the network will make predictions on a lot of biased features, leading to low accuracy. Based on this intuition, we obtain such conflict paths by analyzing the frequency of the activated paths. More specifically, we set the activation frequency of the most frequently activated path as the standard, and compare the activation frequency of each path with it. If the activation frequency of a certain path is less than a certain percentage of the standard (determined by θ), then it can be considered as a conflict path.

Example. Assume the biased model is a simple neural network shown in Figure 2. The weight values have been labeled on the corresponding synapses in Figure 2(a). First we perform path profiling, and the results are set to 0 for simplicity. Then we feed a sample (3,1) into the model, and calculate its relative activation value on each neuron. Take the top neuron of the second layer as an example, its relative activation value is $3 \times 2 + 1 \times (-1) - 0 = 5$, as shown in Figure 2(c). Finally we backtrack the contributions of synapses to get the activation path. Figure 2(d)-(f) shows how we get a path iteratively. Let us denote the k-th neuron in the m-th layer as n_k^m. At first $Q = n_2^4$, assuming $\gamma = 0.8$, we add n_2^3 into Q' and do not add the others because $|6 \times 2| > |0.8 \times 9|$. Then we let $Q = Q' = n_2^3$, and add n_3^2 since $|6 \times 2| > |0.8 \times 6|$. Last we add n_1^1 and n_2^1 into Q' because $|3 \times 1 \leq 0.8 \times 6|$ and $|3 \times 1| + |1 \times 3| > |0.8 \times 6|$. Ultimately we get all the paths iteratively. The conflict paths detection can be regarded as the preceding step of sample clustering, and the example in §3.3 shows its process. □

3.3 Sample Clustering

The sample clustering aims to measure the impact of input samples on fairness. After detecting conflict paths, we can distinguish these neurons exhibiting biased behavior. We handle the corresponding

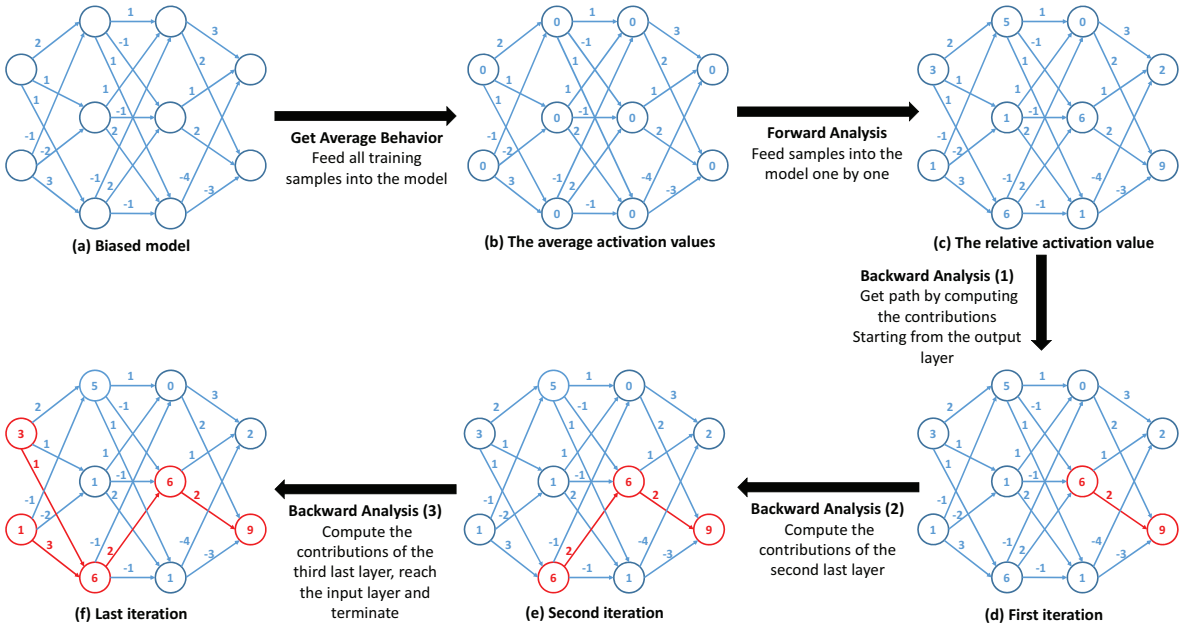


Figure 2: Abnormal path detection example.

samples of these neurons (denoted as *biased sample*) to improve their fairness performance. Since we recorded the relationship between paths and samples (line 4 in Algorithm 1), we can easily find these corresponding samples and get the training dataset divided into two groups.

As shown in Algorithm 1, *PathList* is a list which contains each path and its corresponding activation samples. First, we count the total number of path's corresponding activation samples one by one, and the number is denoted as *activation frequency* (line 34). Second, we sort the activation frequency list we get above, and record its maximum as M (line 35). Third, we check whether these paths' activation frequencies are greater than the threshold $\theta \times M$. We denote the paths which do not meet the above condition as *biased path*, and denote their corresponding samples as *biased sample*. After we detecting the biased paths, these biased samples can be separated from ordinary samples (line 36-40).

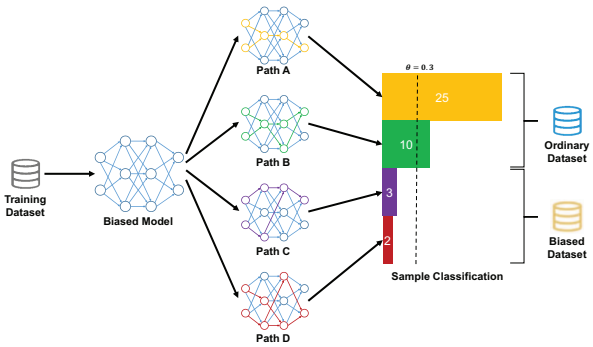


Figure 3: Sample clustering example.

Example. Suppose our training dataset has 40 samples, as shown in Figure 3. We feed the training dataset into the biased model, and obtain 4 different paths A, B, C and D based on the procedure *GetActivationPath* in Algorithm 1. Then we count the number of samples activating these 4 paths, and get 25, 10, 3, and 2 for A, B, C, and D, respectively. We assume that $\theta = 0.3$, so the threshold is $25 \times 0.3 = 7.5$ since the maximum of path activation statistics is 25. Then, path C and D will both be classified as biased paths, which results in 3 samples activating path C and 2 samples activating path D being grouped in biased samples. □

3.4 Selective Training

Finally, we perform ordinary training on the ordinary samples and dropout training on biased samples obtained above. We do not need to change the model structure, only need to change the activation state of the dropout layers. Ordinary training means deactivating the dropout layers for training. With the current training system, we can activate dropout layers when training on these biased samples and vice versa. By performing dropout training on these biased neurons, we enforce them to learn more unbiased features rather than biased ones to mitigate the fairness problems.

Algorithm 1 FairNeuron Algorithm

Input: *BiasedModel*: a biased model to fix
Input: *TrainDataset*: training dataset
Output: *NewModel*: trained model after fixing
procedure FairNeuron
  $PathList \leftarrow []$
  for $sample \in TrainDataset$ **do**
    $P \leftarrow GetActivationPath(BiasedModel, sample, \gamma)$
    $P.sample \leftarrow sample$
    $Append(PathList, P)$
  $O, S \leftarrow GetSamplesDivided(PathList, \theta)$
  $NewModel \leftarrow BiasedModel$
  for $o \in O$ **do**
    $OrdinaryTraining(NewModel, o)$
  for $s \in S$ **do**
    $DropoutTraining(NewModel, s)$
  return $NewModel$

Input: *Model*: model to analyze
Input: *Sample*: samples used in analyze
Input: γ: hyperparameter to determine the activation of neurons
Output: P: path activated
procedure GetActivationPath
  $P \leftarrow \emptyset$
  $Q \leftarrow \emptyset$
  $Q \leftarrow OutputNeuron$
  while $Q \neq \emptyset$ **do**
    $Q' \leftarrow \emptyset$
    for $q \in Q$ **do**
      $N \leftarrow GetPreNeuron(q)$
      for $n \in N$ **do**
        $ContribList[n] \leftarrow ComputeContrib(n)$
      $SortedList \leftarrow Sort(ContribList)$
      $Sum \leftarrow 0$
      for $i \leftarrow 0$ *to* $len(ContribList)$ **do**
        if $Sum \leq \gamma \times q.value$ **then**
          $Sum \leftarrow Sum + SortedList[i].value$
          $Q' \leftarrow Q' \cap SortedList[i].index$
          $P \leftarrow P \cap (SortedList[i].index, q.index)$
    $Q \leftarrow Q'$
  return P

Input: *PathList*: a list of paths used to cluster samples
Input: θ: hyperparameters used to find conflict paths
Output: *OL*, *AL*: benign and biased samples, respectively.
procedure GetSamplesDivided
  $OL \leftarrow []$
  $AL \leftarrow []$
  $PathList.count \leftarrow Count(PathList.samples)$
  $M \leftarrow Max(PathList.count)$
  for $i \leftarrow 0$ *to* $len(PathList)$ **do**
    if $PathList[i].count \leq \theta \times M$ **then**
      $Append(OL, PathList[i].samples)$
    else
      $Append(AL, PathList[i].samples)$
  return $OrdinaryList, AbnormalList$

Table 1: Experimented DNN models.

Dataset	Model	Accuracy
Census	3 Hidden-layer Fully-connected NN	83.9%
Credit	3 Hidden-layer Fully-connected NN	73.4%
COMPAS	3 Hidden-layer Fully-connected NN	62.1%

4 EVALUATION

4.1 Experiment Setup

4.1.1 Hardware and software. We conducted our experiments on a GPU server with 32 cores Intel Xeon 2.10GHz CPU, 256 GB system memory and 1 NVIDIA TITAN V GPU running the Ubuntu 16.04 operating system.

4.1.2 Datasets. We evaluated our method on three popular datasets: the UCI Adult Census, COMPAS, and German Credit.

- **UCI Adult Census.** The UCI Adult Census was extracted from the 1994 Census bureau database, gathering 32,561 instances represented by 9 features such as age, education and occupation. The gender is considered as the sensitive attribute.
- **COMPAS.** The COMPAS system is a popular commercial algorithm used by judges for predicting the risk of recidivism, and the COMPAS dataset is a sample outcome from the COMPAS system. The race of each defendant is the sensitive attribute.
- **German Credit.** This is a small dataset with 600 records and 20 attributes. The original aim of the dataset is to give an assessment of individual's credit based on personal and financial records. The gender is the sensitive attribute.

4.1.3 Models. In our experiment, we built a fully-connected neural network with three hidden layers for each dataset respectively. For the COMPAS and the German Credit dataset, each hidden layer is composed of 32 neurons, while for the UCI Adult Census dataset, each hidden layer is composed of 128 neurons due to its larger encoded input. The details of the models used in the experiments are shown in Table 1.

We use the softmax activation function for Census and German Credit to achieve binary classification, and the linear activation function for COMPAS to get recidivism scores. We randomly separate the dataset into the training, validation, and test sets, by a ratio of 7:1:2, respectively. The neural network is trained by the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.9999$, and initial learning rate $l_r = 0.01$, which is scheduled by a factor of 0.1 when reaching a plateau.

4.1.4 Hyperparameter tuning. To obtain the suitable hyperparameters θ and γ, we run a parallel grid search for hyperparameters to optimize training loss function. We sample θ between the interval $[10^{-4}, 1]$ proportionally, and sample γ between the interval $[0.5, 1]$. Following the standard practice in machine learning, the grid search is performed on a small subset drawn from the training set in a certain proportion (e.g., 10%), and we utilize the *ray tune tool* to perform it automatically [6].

4.1.5 Metrics and baseline methods. We compare our algorithm with other popular in-processing state-of-the-art fixing algorithms, such as FAD [11] and Ethical Adversaries [25]. Besides, we also compared with the representative algorithms of the other two kinds of fixing algorithms, reweighing in pre-processing [41] and Reject Option Classification in post-processing [42].

We aim to answer the following research questions through our experiments:
RQ1: How effective is our algorithm in fixing bias model?
RQ2: How efficient is our algorithm in fixing bias model?
RQ3: How parameters affect model performance?
RQ4: How our algorithm perform in image datasets?

4.2 Effectiveness of FairNeuron

Experiment Design: To evaluate the effectiveness of FairNeuron, we test the following models: the naive baseline model, models fixed by FAD, by Ethical Adversaries, and by FairNeuron. Due to the randomness in these experiments, we ran the training 10 times to ensure the reliability of results and enforced these fixing algorithms to share the same original training dataset. To measure the effectiveness of FairNeuron, we compare the performance between FairNeuron and the other algorithms in terms of both utility and fairness. To demonstrate the effectiveness of the three components of FairNeuron (i.e. neural network slicing, sample clustering and selective training), we conducted a detailed comparison between our algorithm and other popular works.

Results: The details of the comparison results are presented in Table 4. The first column lists the three datasets. The second column shows the different algorithm. The third column lists the utility, and the remaining columns list the fairness criteria. The model utilities are evaluated by binary classification accuracy (Acc), and the fairness performance are measured by demographic parity (DP), demographic parity ratio (DPR), and Equal opportunity (EO). The best results are shown in bold.

Analysis: The experimental results demonstrate the effectiveness of our algorithm. Firstly, FairNeuron can effectively fix the fairness bias of all models trained on different datasets. Secondly, FairNeuron achieves the highest utility among all models with fairness constraints, and even surpasses the naive model on COMPAS and Credit.

Table 4 shows the fairness improvement of naive models on Census, Credit and COMPAS, respectively. FairNeuron improves DPR by 98.47%, 157.23%, and 3895.23%, mitigates fairness problem by 69.69%, 21.12% and 38.95% in terms of EO, and 74.68%, 2.08%, 96.19% in terms of DP. Compared with the other algorithms, FairNeuron achieves the best fairness performance on Census and COMPAS. However, the EO and DP results of FairNeuron on Credit is not satisfactory. After our careful analysis, we found that our neuron network slicing is not fully functional since Credit only has 600 instances. Thus, how to improve the utility of models training on such small datasets will be one of our future works.

Besides, Table 4 demonstrates that FairNeuron has little impact on model utility after a successful fairness fixing, and even has the advantage of increasing accuracy by fixing fairness problems. The average utility of FairNeuron is the highest among all models with fairness constraints, which exceeds ROC by 27.9%, Reweighing by 17.5% , Ethical Adversaries by 3.85% and FAD by 27.22%, and even surpasses the naive model on the German Credit and COMPAS datasets. The detailed average accuracy change is -0.83%, 1.36%, and 28.66%. We found that it is mainly because FairNeuron improves the utility by mitigating the overfitting problem in model training procedures, and the size of Census dataset is relatively large, so its overfitting problem is unobvious.

Table 2: Random clustering vs. our clustering

Method	Acc	DP	DPR	EO
Random	0.749	0.325	1.89	0.159
Ours	0.799	0.013	1.02	0.058

In summary, FairNeuron can effectively fix the bias training procedures, and has little impact on the model utility while improving the fairness performance significantly. Then we prove the effectiveness of each step in FairNeuron separately.

4.2.1 Effectiveness of neuron network slicing. Figure 4 shows a example of the distribution of abnormal paths. Here, the maximum of path activation statistics is 47, and we assume $\theta = 0.03$, so the threshold is 1.41, as the green line shows. We can see that most of non-zero paths are concentrated near 1, but the proportion of their corresponding samples is not high. These paths are the abnormal paths detected by our approach.

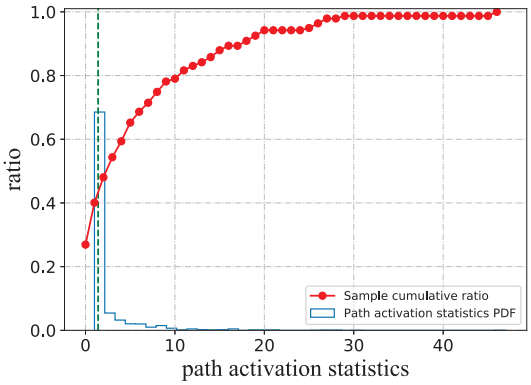


Figure 4: Result of neuron network slicing. The blue step line presents the probability density function of path activation statistics, the red line presents the accumulation of sample ratio, and the green line presents the threshold.

4.2.2 Effectiveness of sample clustering. To demonstrate the effectiveness of sample clustering, we compare the fixing performance between our sample separation and the random clustering methods. We set the number of randomly-obtained abnormal samples to be the same as that of FairNeuron.

Table 2 reports the performance of different clustering methods. With our method, the average accuracy is improved by 6.68%, and the fairness performance has also been greatly improved, which are 96.19%, 97.67% and 38.95% of DP, DPR and EO, respectively.

Table 3: Comparison among dropout, ordinary and selective training.

Training approach	Acc	DP	DPR	EO
Ordinary	0.575	0.733	0.183	0.683
Dropout	0.621	0.341	1.860	0.095
Selective	0.799	0.013	1.021	0.058

Table 4: Results on the three datasets. Best results are in bold.

Dataset	Model	Acc	DP	EO	DPR
Census	Naive model	0.839	0.079	0.102	0.609
	ROC	0.597	0.044	0.051	0.773
	Reweighing	0.719	0.059	0.0141	1.497
	FAD	0.612	0.059	0.061	0.518
	Ethical Adversaries	0.814	0.031	0.179	0.784
	FairNeuron	**0.832**	**0.020**	**0.031**	**0.869**
Credit	Naive model	0.734	0.048	0.142	0.407
	ROC	0.646	0.041	0.073	1.273
	Reweighing	0.632	0.067	0.066	0.828
	FAD	0.710	**0.000**	**0.000**	inf
	Ethical Adversaries	0.715	0.041	0.031	2.442
	FairNeuron	**0.744**	0.047	0.112	**0.834**
COMPAS	Naive model	0.621	0.341	0.095	1.860
	ROC	0.618	0.083	0.069	0.890
	Reweighing	0.671	0.193	0.176	1.406
	FAD	0.567	0.057	0.114	0.926
	Ethical Adversaries	0.759	0.095	0.095	1.203
	FairNeuron	**0.799**	**0.013**	**0.058**	**1.021**

4.2.3 Effectiveness of selective training. To show the effectiveness of selective training, we provide a comparison among pure dropout, pure ordinary and selective training.

Table 3 presents the results of different training approaches. Selective training surpasses the ordinary training by 38.96%, 98.22%, 97.43% and 91.50%, while surpassing the pure dropout training by 22.27%, 96.19%, 97.55% and 38.95% in Acc, DP, DPR and EO, respectively. It confirms that the selective training in FairNeuron can achieve high accuracy and fairness.

4.3 Efficiency of FairNeuron

Experiment Design: To evaluate the efficiency of FairNeuron, we measured the time usage of ordinary training, Ethical Adversaries and FairNeuron training on all three datasets. We performed 10 trials which uses random training/test data splitting, naive model training, hyperparameter tuning and model repairing (for Ethical Adversaries and FairNeuron) and computed the average overhead to avoid randomness. Table 5 presents how much time it takes to complete its fixing for each method. For Ethical Adversaries, it shows the time for per iteration in 50 iterations. For FairNeuron, it shows the time usage per trial. We also recorded the time usage of each step in FairNeuron. Results and analysis are presented below.

Table 5: Time to train a model.

Dataset	Naive	EA (/iteration)	FairNeuron (/trial)
Census	115.74s	1439.96s	254.41s
Credit	3.07s	33.24s	31.49s
COMPAS	11.92s	81.93s	44.31s

Table 6: Time used in each step.

Dataset	Para selection	Slicing	Clustering	Training
Census	115.41s	25.37s	43.70s	74.37s
Credit	30.98s	0.20s	6.73e-4s	0.30s
COMPAS	40.76s	2.09s	0.06s	1.40s

Results: Table 5 shows the comparison of time usage among ordinary training, Ethical Adversaries and FairNeuron training. The first column lists the three datasets and the remaining columns show the different training methods. On average, FairNeuron takes 5.39 times of ordinary training and 55.49% of Ethical Adversaries.

Table 6 reports the time costs of each step. The first column also lists the three datasets and the remaining columns show the time costs of hyperparameters selection, neuron network slicing, sample clustering and selective training respectively.

Analysis: For ordinary training, the runtime overhead all comes from the training procedure, but for FairNeuron, the hyperparameters tuning accounts for a larger ratio of the total time usage, as shown in Table 6. So FairNeuron takes only less than twice of the time usage of ordinary training on large datasets like Census, but on small datasets like the German Credit dataset, it takes relatively a long time. If FairNeuron tries more times, the average time will be reduced because the hyperparameters tuning is only conducted once.

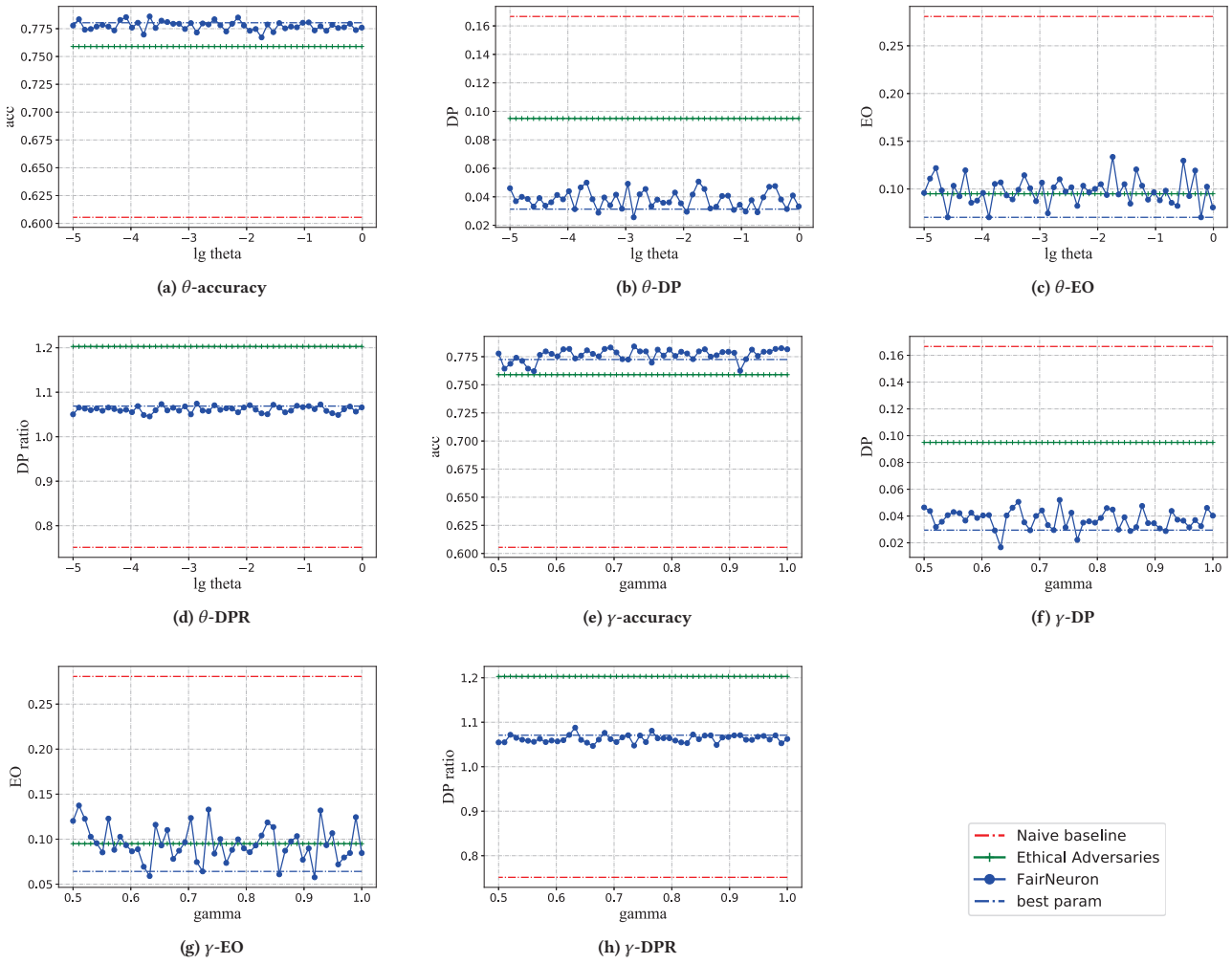
Overall, FairNeuron is more efficient than Ethical Adversaries in fixing models, with an average speedup of 180%.

4.4 Effects of Configurable Hyperparameters

FairNeuron leverages two configurable hyperparameters, θ and γ, to fix fairness problems. The hyperparameters γ represents the threshold of neuron activation, and its value affects the complexity of the path. As its value decreases, more neurons and synapses are included in the path, resulting in a more complex path. And θ represents the threshold of neuron network slicing. The lower is the θ, the fewer paths are classified as abnormal.

We conduct a comparison experiment of these hyperparameters. θ varies between the interval $[10^{-4}, 1]$ and γ varies between the interval $[0.5, 1]$. Note that we use logarithmic coordinates for θ since its value is sampled proportionally.

Figure 5 shows how hyperparameters assignments will affect the performance. Based on our results in comparison with the naive model and Ethical Adversaries, we can conclude that our algorithm does perform better on this task and is not sensitive to hyperparameters assignments except for EO (Figure 5(c) & (g)). By increasing the weight of EO in hyperparameter tuning loss function, we can constrain its fluctuations.



(a) θ-accuracy (b) θ-DP (c) θ-EO

(d) θ-DPR (e) γ-accuracy (f) γ-DP

(g) γ-EO (h) γ-DPR

Figure 5: Effect of hyperparameters θ and γ. θ is sampled proportionally, so we take the logarithm of θ as x axis.

4.5 Performance on Image Datasets

Experiment Design: We also explored the possibility of using our method in fixing models on image datasets, which has not done by baseline methods due to their inefficiency. In our experiment, we leverage a 4-layer fully-connected NN trained on MNIST [5] and ResNet-18 [37] trained on CIFAR-10 [1], compare FairNeuron with the naive model and random dropout. We use Class-wise Variance (CV) and Maximum Class-wise Discrepancy (MCD) as fairness metrics.

Results: Table 7 summarizes our results. The first column lists the datasets. The second column shows the different model, and the remaining columns list the performance. The best results are shown in bold. As we can see from the table, FairNeuron can effectively improve the model fairness by 20% for MCD and 80% for CV. We discuss it further in §6.

Table 7: Results on image datasets. Best results are in bold.

Dataset	Model	Acc	CV	MCD
MNIST	Naive model	0.957	6.66e-5	0.057
	Random dropout	0.949	3.58e-5	0.052
	FairNeuron	**0.961**	**9.54e-6**	**0.046**
CIFAR-10	Naive model	**0.814**	6.04e-4	0.236
	Random dropout	0.798	8.55e-4	0.464
	FairNeuron	0.808	**1.16e-4**	**0.187**

5 RELATED WORK

Neural Network Slicing. Path analysis or dataflow analysis [13] is a fundamental technique in traditional software engineering tasks like testing, debugging and optimization. It offers a window to study program's dynamic behavior. In recent years, with the development of AI security, especially adversarial attack and defense,

conflict path detection has been used for interpretability. Wang et al. [63] proposed a method to interpret neural networks by extracting the critical data routing paths (CDRPs), and they demonstrated its effectiveness on adversarial sample detection problem. Qiu et al. [56] treat a neural network as a dataflow graph, which can be applied the profiling technique to extract its execution path. Zhang et al. [73] apply the dynamic slicing on deep neural networks.

Fairness of ML. With the increasing use of automated decision-making approaches and systems, fairness considerations in ML have gained significant attention. Researchers found many fairness problems with high social impact, such as standardized tests in higher education [24], employment [34, 57, 61], and re-offence judgement [16, 17, 20, 51]. Besides, governments (e.g. the EU [62] and the US [54, 55]), organizations [49], and the media have called for more societal accountability and social understanding of ML.

To address the concern above, numerous fairness notions are proposed. In high level, these fairness notions can be split into three categories: (i) individual fairness, which requires that similar individuals should be treated similarly [26, 46, 67]; (ii) group fairness, which concerns about whether subpopulation with different sensitive characteristics are treated equally [28, 35, 70]; (iii) Max-Min fairness, which try to improve the per-group fairness [36, 45, 69].

Fairness testing is also an important research direction, and its approaches mostly based on generation techniques. THEMIS [14] considers group fairness using causal analysis and uses random test generation to evaluate fairness. AEQUITAS inherits and improves THEMIS, and focuses on the individual discriminatory instances generation [60]. Later, ADF combines global search and local search to systematically search the input space with the guidance of gradient [71]. Symbolic Generation (SG) integrates symbolic execution and local model explanation techniques to craft individual discriminatory instances [12].

The ML model needs to be repaired after the fairness problem is found. These approaches can be generally split into three categories: (i) Pre-processing approaches, which fix the training data to reduce the latent discrimination in dataset. For example, the bias could be mitigated by correcting labels [40, 70], revising attributes [28, 41], generating non-discrimination data [58, 64], and obtaining fair data representations [18]. (ii) In-processing approaches, which revise the training of the bias model to achieve fairness [66, 68]. More specifically, these approaches apply fairness constraints [26, 66], propose an objective function considering the fairness of prediction [68], or design a new training frameworks [11, 64]. (iii) post-processing approaches, which directly change the predictive labels of bias models' output to obtain fairness [35, 53].

6 CONCLUSION

In this paper, we proposed a lightweight algorithm FairNeuron to effectively fixing fairness problems for deep neural network through path analysis. Our algorithm combines a path analysis procedure and a dropout procedure to systematically improve model performance. FairNeuron searches bias instances with the guidance of path analysis and mitigates fairness problems by dropout training. Our evaluation results show that FairNeuron has significantly better performance both in terms of effectively and efficiently in fixing bias models. For CNN model, we can only perform FairNeuron on the last full-connected layer, so its performance is not ideal. We will improve FairNeuron on CNN in the future.

7 ACKNOWLEDGEMENT

We thank the anonymous reviewers for their constructive comments. This research was partially supported by National Key R&D Program (2020YFB1406900), National Natural Science Foundation of China (U21B2018, 62161160337, 61822309, U20B2049, 61773310, U1736205, 61802166) and Shaanxi Province Key Industry Innovation Program (2021ZDLGY01-02). Chao Shen is the corresponding author. The views, opinions and/or findings expressed are only those of the authors.

REFERENCES

[1] [n.d.]. CIFAR-10 and CIFAR-100 datasets. https://www.cs.toronto.edu/~kriz/cifar.html
[2] [n.d.]. The First International Beauty Contest Judged by Artificial Intelligence. http://beauty.ai
[3] [n.d.]. Machine Bias — ProPublica. https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing
[4] [n.d.]. Microsoft's neo-Nazi sexbot was a great lesson for makers of AI assistants. https://www.technologyreview.com/2018/03/27/144290/microsofts-neo-nazi-sexbot-was-a-great-lesson-for-makers-of-ai-assistants/
[5] [n.d.]. MNIST handwritten digit database, Yann LeCun, Corinna Cortes and Chris Burges. http://yann.lecun.com/exdb/mnist/
[6] [n.d.]. Tune: Scalable Hyperparameter Tuning — Ray v1.9.0. https://docs.ray.io/en/latest/tune/index.html
[7] 2016. A beauty contest was judged by AI and the robots didn't like dark skin. http://www.theguardian.com/technology/2016/sep/08/artificial-intelligence-beauty-contest-doesnt-like-black-people Section: Technology.
[8] 2020. NAB turns to AI to decide on small business loans. https://www.afr.com/companies/financial-services/nab-turns-to-artificial-intelligence-to-assess-small-business-loans-20201204-p56kmk Section: financialservices.
[9] 2021. How AI will change the HR industry | HRExecutive.com. http://hrexecutive.com/ai-will-make-traditional-hr-extinct-how-to-prepare-for-whats-next/
[10] Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. TensorFlow: A System for Large-Scale Machine Learning. 265–283. https://www.usenix.org/conference/osdi16/technical-sessions/presentation/abadi
[11] Tameem Adel, Isabel Valera, Zoubin Ghahramani, and Adrian Weller. 2019. One-Network Adversarial Fairness. *Proceedings of the AAAI Conference on Artificial Intelligence* 33 (July 2019), 2412–2420. https://doi.org/10.1609/aaai.v33i01.33012412
[12] Aniya Agarwal, Pranay Lohia, Seema Nagar, Kuntal Dey, and Diptikalyan Saha. 2018. Automated test generation to detect individual discrimination in AI models. *arXiv preprint arXiv:1809.03260* (2018).
[13] Glenn Ammons and James R Larust. [n.d.]. Improving Data-flow Analysis with Path Profiles. ([n. d.]), 13.
[14] Rico Angell, Brittany Johnson, Yuriy Brun, and Alexandra Meliou. 2018. Themis: automatically testing software for discrimination. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering - ESEC/FSE 2018*. ACM Press, Lake Buena Vista, FL, USA, 871–875. https://doi.org/10.1145/3236024.3264590
[15] Solon Barocas and Andrew D. Selbst. 2016. Big data's disparate impact. *Calif. L. Rev.* 104 (2016), 671. Publisher: HeinOnline.
[16] Richard Berk. 2019. Accuracy and fairness for juvenile justice risk assessments. *Journal of Empirical Legal Studies* 16, 1 (2019), 175–194. Publisher: Wiley Online Library.
[17] Richard Berk, Hoda Heidari, Shahin Jabbari, Michael Kearns, and Aaron Roth. 2021. Fairness in criminal justice risk assessments: The state of the art. *Sociological Methods & Research* 50, 1 (2021), 3–44. Publisher: Sage Publications Sage CA: Los Angeles, CA.
[18] Alex Beutel, Jilin Chen, Zhe Zhao, and Ed H. Chi. 2017. Data Decisions and Theoretical Implications when Adversarially Learning Fair Representations. *arXiv:1707.00075 [cs]* (July 2017). http://arxiv.org/abs/1707.00075 arXiv: 1707.00075.
[19] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, and

Jiakai Zhang. 2016. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316* (2016).

[20] Tim Brennan and William L. Oliver. 2013. Emergence of machine learning techniques in criminology: implications of complexity in our data and in research questions. *Criminology & Pub. Pol'y* 12 (2013), 551. Publisher: HeinOnline.

[21] Simon Caton and Christian Haas. 2020. Fairness in Machine Learning: A Survey. *arXiv:2010.04053 [cs, stat]* (Oct. 2020). http://arxiv.org/abs/2010.04053 arXiv: 2010.04053.

[22] Tong Che, Yanran Li, Athul Paul Jacob, Yoshua Bengio, and Wenjie Li. 2017. Mode Regularized Generative Adversarial Networks. *arXiv:1612.02136 [cs]* (March 2017). http://arxiv.org/abs/1612.02136 arXiv: 1612.02136.

[23] Alexandra Chouldechova. 2017. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data* 5, 2 (2017), 153–163. Publisher: Mary Ann Liebert, Inc. 140 Huguenot Street, 3rd Floor New Rochelle, NY 10801 USA.

[24] T. Anne Cleary. 1966. Test bias: Validity of the Scholastic Aptitude Test for Negro and White students in integrated colleges. *ETS Research Bulletin Series* 1966, 2 (1966), i–23. Publisher: Wiley Online Library.

[25] Pieter Delobelle, Paul Temple, Gilles Perrouin, Benoît Frénay, Patrick Heymans, and Bettina Berendt. 2021. Ethical adversaries: Towards mitigating unfairness with adversarial machine learning. *ACM SIGKDD Explorations Newsletter* 23, 1 (2021), 32–41. Publisher: ACM New York, NY, USA.

[26] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. 2012. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference on - ITCS '12*. ACM Press, Cambridge, Massachusetts, 214–226. https://doi.org/10.1145/2090236.2090255

[27] Yanai Elazar and Yoav Goldberg. 2018. Adversarial removal of demographic attributes from text data. *arXiv preprint arXiv:1808.06640* (2018).

[28] Michael Feldman, Sorelle A. Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. 2015. Certifying and Removing Disparate Impact. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '15*. ACM Press, Sydney, NSW, Australia, 259–268. https://doi.org/10.1145/2783258.2783311

[29] Rupert Freeman, Nisarg Shah, and Rohit Vaish. 2020. Best of Both Worlds: Ex-Ante and Ex-Post Fairness in Resource Allocation. *arXiv:2005.14122 [cs]* (May 2020). http://arxiv.org/abs/2005.14122 arXiv: 2005.14122.

[30] Sorelle A. Friedler, Carlos Scheidegger, Suresh Venkatasubramanian, Sonam Choudhary, Evan P. Hamilton, and Derek Roth. 2019. A comparative study of fairness-enhancing interventions in machine learning. In *Proceedings of the Conference on Fairness, Accountability, and Transparency (FAT* '19)*. Association for Computing Machinery, New York, NY, USA, 329–338. https://doi.org/10.1145/3287560.3287589

[31] Yaroslav Ganin and Victor Lempitsky. [n.d.]. Unsupervised Domain Adaptation by Backpropagation. ([n. d.]), 10.

[32] Xuanqi Gao. 2022. FairNeuron. https://github.com/Antimony5292/FairNeuron original-date: 2021-09-01T12:52:43Z.

[33] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, Vol. 27. Curran Associates, Inc. https://papers.nips.cc/paper/2014/hash/5ca3e9b122f61f8f06494c97b1afccf3-Abstract.html

[34] Robert M. Guion. 1966. Employment tests and discriminatory hiring. *Industrial Relations: A Journal of Economy and Society* 5, 2 (1966), 20–37. Publisher: Wiley Online Library.

[35] Moritz Hardt, Eric Price, and Nati Srebro. 2016. Equality of opportunity in supervised learning. *Advances in neural information processing systems* 29 (2016), 3315–3323.

[36] Tatsunori Hashimoto, Megha Srivastava, Hongseok Namkoong, and Percy Liang. 2018. Fairness without demographics in repeated loss minimization. In *International Conference on Machine Learning*. PMLR, 1929–1938.

[37] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[38] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Advances in neural information processing systems*. 2042–2050.

[39] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188* (2014).

[40] Faisal Kamiran and Toon Calders. 2009. Classifying without discriminating. In *2009 2nd international conference on computer, control and communication*. IEEE, 1–6.

[41] Faisal Kamiran and Toon Calders. 2012. Data preprocessing techniques for classification without discrimination. *Knowledge and Information Systems* 33, 1 (Oct. 2012), 1–33. https://doi.org/10.1007/s10115-011-0463-8

[42] Faisal Kamiran, Asim Karim, and Xiangliang Zhang. 2012. Decision Theory for Discrimination-Aware Classification. In *2012 IEEE 12th International Conference on Data Mining*. IEEE, Brussels, Belgium, 924–929. https://doi.org/10.1109/ICDM.2012.45

[43] Jon Kleinberg, Jens Ludwig, Sendhil Mullainathan, and Ashesh Rambachan. 2018. Algorithmic fairness. In *Aea papers and proceedings*, Vol. 108. 22–27.

[44] Matt J Kusner, Joshua Loftus, Chris Russell, and Ricardo Silva. [n.d.]. Counterfactual Fairness. *NIPS 2017* ([n. d.]), 11.

[45] Preethi Lahoti, Alex Beutel, Jilin Chen, Kang Lee, Flavien Prost, Nithum Thain, Xuezhi Wang, and Ed H. Chi. 2020. Fairness without Demographics through Adversarially Reweighted Learning. *arXiv:2006.13114 [cs, stat]* (Nov. 2020). http://arxiv.org/abs/2006.13114 arXiv: 2006.13114.

[46] Preethi Lahoti, Krishna P. Gummadi, and Gerhard Weikum. 2019. Operationalizing individual fairness with pairwise fair representations. *arXiv preprint arXiv:1907.01439* (2019).

[47] Shiqing Ma, Yousra Aafer, Zhaogui Xu, Wen-Chuan Lee, Juan Zhai, Yingqi Liu, and Xiangyu Zhang. 2017. LAMP: data provenance for graph based machine learning algorithms through derivative computation. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*. 786–797.

[48] Shiqing Ma, Yingqi Liu, Wen-Chuan Lee, Xiangyu Zhang, and Ananth Grama. 2018. MODE: automated neural network model debugging via state differential analysis and input selection. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering - ESEC/FSE 2018*. ACM Press, Lake Buena Vista, FL, USA, 175–186. https://doi.org/10.1145/3236024.3236082

[49] Annette Markham and Elizabeth Buchanan. 2012. Ethical decision-making and internet research: Version 2.0. recommendations from the AoIR ethics working committee. *Available online: aoir. org/reports/ethics2. pdf* (2012).

[50] Panayotis Mertikopoulos, Christos Papadimitriou, and Georgios Piliouras. 2018. Cycles in Adversarial Regularized Learning. In *Proceedings of the 2018 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Society for Industrial and Applied Mathematics, 2703–2717. https://doi.org/10.1137/1.9781611975031.172

[51] Cathy O'neil. 2016. *Weapons of math destruction: How big data increases inequality and threatens democracy*. Crown.

[52] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. [n.d.]. PyTorch: An Imperative Style, High-Performance Deep Learning Library. ([n. d.]), 12.

[53] Geoff Pleiss, Manish Raghavan, Felix Wu, Jon Kleinberg, and Kilian Q Weinberger. [n.d.]. On Fairness and Calibration. ([n. d.]), 10.

[54] Executive Office of the President, Cecilia Munoz, Domestic Policy Council Director, Megan (US Chief Technology Officer Smith (Office of Science, Technology Policy)), DJ (Deputy Chief Technology Officer for Data Policy, Chief Data Scientist Patil (Office of Science, and Technology Policy)). 2016. *Big data: A report on algorithmic systems, opportunity, and civil rights*. Executive Office of the President.

[55] United States Executive Office of the President and John Podesta. 2014. *Big data: Seizing opportunities, preserving values*. White House, Executive Office of the President.

[56] Yuxian Qiu, Jingwen Leng, Cong Guo, Quan Chen, Chao Li, Minyi Guo, and Yuhao Zhu. 2019. Adversarial Defense Through Network Profiling Based Path Extraction. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Long Beach, CA, USA, 4772–4781. https://doi.org/10.1109/CVPR.2019.00491

[57] Manish Raghavan, Solon Barocas, Jon Kleinberg, and Karen Levy. 2020. Mitigating bias in algorithmic hiring: Evaluating claims and practices. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*. 469–481.

[58] Prasanna Sattigeri, Samuel C. Hoffman, Vijil Chenthamarakshan, and Kush R. Varshney. 2019. Fairness GAN: Generating datasets with fairness properties using a generative adversarial network. *IBM Journal of Research and Development* 63, 4/5 (2019), 3–1. Publisher: IBM.

[59] Florian Tramer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, Jean-Pierre Hubaux, Mathias Humbert, Ari Juels, and Huang Lin. 2017. FairTest: Discovering Unwarranted Associations in Data-Driven Applications. In *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, Paris, 401–416. https://doi.org/10.1109/EuroSP.2017.29

[60] Sakshi Udeshi, Pryanshu Arora, and Sudipta Chattopadhyay. 2018. Automated directed fairness testing. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering - ASE 2018*. ACM Press, Montpellier, France, 98–108. https://doi.org/10.1145/3238147.3238165

[61] Elmira van den Broek, Anastasia Sergeeva, and Marleen Huysman. 2019. Hiring algorithms: An ethnography of fairness in practice. (2019).

[62] Paul Voigt and Axel Von dem Bussche. 2017. The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing* 10 (2017), 3152676. Publisher: Springer.

[63] Yulong Wang, Hang Su, Bo Zhang, and Xiaolin Hu. 2018. Interpret Neural Networks by Identifying Critical Data Routing Paths. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, Salt Lake City, UT, 8906–8914. https://doi.org/10.1109/CVPR.2018.00928

[64] Depeng Xu, Shuhan Yuan, Lu Zhang, and Xintao Wu. 2018. Fairgan: Fairness-aware generative adversarial networks. In *2018 IEEE International Conference on Big Data (Big Data)*. IEEE, 570–575.
[65] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P. Gummadi. 2017. Fairness Beyond Disparate Treatment & Disparate Impact: Learning Classification without Disparate Mistreatment. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 1171–1180. https://doi.org/10.1145/3038912.3052660
[66] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rogriguez, and Krishna P. Gummadi. 2017. Fairness constraints: Mechanisms for fair classification. In *Artificial Intelligence and Statistics*. PMLR, 962–970.
[67] Richard Zemel. [n.d.]. Learning Fair Representations. ([n. d.]), 9.
[68] Brian Hu Zhang, Blake Lemoine, and Margaret Mitchell. 2018. Mitigating Unwanted Biases with Adversarial Learning. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*. ACM, New Orleans LA USA, 335–340. https://doi.org/10.1145/3278721.3278779
[69] Chongjie Zhang and Julie A. Shah. 2014. Fairness in multi-agent sequential decision-making. In *Advances in Neural Information Processing Systems*. 2636–2644.
[70] Lu Zhang, Yongkai Wu, and Xintao Wu. 2017. Achieving Non-Discrimination in Data Release. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, Halifax NS Canada, 1335–1344. https://doi.org/10.1145/3097983.3098167
[71] Peixin Zhang, Jingyi Wang, Jun Sun, Guoliang Dong, Xinyu Wang, Xingen Wang, Jin Song Dong, and Ting Dai. 2020. White-box fairness testing through adversarial sampling. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ACM, Seoul South Korea, 949–960. https://doi.org/10.1145/3377811.3380331
[72] Xiaoyu Zhang, Juan Zhai, Shiqing Ma, and Chao Shen. 2021. AUTOTRAINER: An Automatic DNN Training Problem Detection and Repair System. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*. IEEE, Madrid, ES, 359–371. https://doi.org/10.1109/ICSE43902.2021.00043
[73] Ziqi Zhang, Yuanchun Li, Yao Guo, Xiangqun Chen, and Yunxin Liu. 2020. Dynamic Slicing for Deep Neural Networks. (2020), 13.