RUNNER: Responsible UNfair NEuron Repair for Enhancing Deep Neural Network Fairness

Tianlin Li[*]
tianlin001@e.ntu.edu.sg
Nanyang Technological University
Singapore

Yue Cao[*]
CAOY0033@e.ntu.edu.sg
Nanyang Technological University
Singapore

Jian Zhang[†]
jian_zhang@ntu.edu.sg
Nanyang Technological University
Singapore

Shiqian Zhao
shiqian.zhao@ntu.edu.sg
Nanyang Technological University
Singapore

Yihao Huang
huang.yihao@ntu.edu.sg
Nanyang Technological University
Singapore

Aishan Liu
liuaishan@buaa.edu.cn
Beihang University
China

Qing Guo
tsingqguo@ieee.org
Institute of High Performance Computing (IHPC), Centre for Frontier AI Research (CFAR), A*STAR
Singapore

Yang Liu
yangliu@ntu.edu.sg
Nanyang Technological University
Singapore

ABSTRACT

Deep Neural Networks (DNNs), an emerging software technology, have achieved impressive results in a variety of fields. However, the discriminatory behaviors towards certain groups (a.k.a. unfairness) of DNN models increasingly become a social concern, especially in high-stake applications such as loan approval and criminal risk assessment. Although there has been a number of works to improve model fairness, most of them adopt an adversary to either expand the model architecture or augment training data, which introduces excessive computational overhead. Recent work diagnoses responsible unfair neurons first and fixes them with selective retraining. Unfortunately, existing diagnosis process is time-consuming due to multi-step training sample analysis, and selective retraining may cause a performance bottleneck due to indirectly adjusting unfair neurons on biased samples. In this paper, we propose Responsible UNfair NEuron Repair (RUNNER) that improves existing works in three key aspects: (1) *efficiency*: we design the Importance-based Neuron Diagnosis that identifies responsible unfair neurons in one step with a novel importance criterion of neurons; (2) *effectiveness*: we design the Neuron Stabilizing Retraining by adding a loss term that measures the activation distance of responsible unfair neurons from different subgroups in all sources; (3) *generalization*: we investigate the effectiveness on both structured tabular data and large-scale unstructured image data, which is often ignored in prior studies. Our extensive experiments across 5 datasets show that RUUNER can effectively and efficiently diagnose and repair the DNNs regarding unfairness. On average, our approach significantly reduces computing overhead from 341.7s to 29.65s, and achieves improved fairness up to 79.3%. Besides, RUNNER also keeps state-of-the-art results on the unstructured dataset.

CCS CONCEPTS

• **Software and its engineering → Software testing and debugging**; • **Computing methodologies → Philosophical/theoretical foundations of artificial intelligence**.

KEYWORDS

Deep Learning Repair, Fairness, Model Interpretation

ACM Reference Format:
Tianlin Li, Yue Cao, Jian Zhang, Shiqian Zhao, Yihao Huang, Aishan Liu, Qing Guo, and Yang Liu. 2024. RUNNER: Responsible UNfair NEuron Repair for Enhancing Deep Neural Network Fairness. In *2024 IEEE/ACM 46th International Conference on Software Engineering (ICSE '24), April 14–20, 2024, Lisbon, Portugal.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3597503.3623334

1 INTRODUCTION

Deep neural networks (DNNs) have showcased their great potential in various software applications across many fields such as image classification [27], speech recognition [58], natural language processing [57], and software engineering [66]. However, DNNs are also unveiled to be unreliable and vulnerable in terms of some properties such as robustness, privacy, and fairness, which severely restricts the usability of deep learning [9, 29, 30, 37, 38, 53, 68]. Among these properties, as the growing societal impact, fairness is attracting more and more attention, especially in high-stake applications such as criminal justice, loan approval, and credit scoring.

[*] Both authors contributed equally to this research.
[†] Corresponding Author.

ICSE '24, April 14–20, 2024, Lisbon, Portugal

ACM ISBN 979-8-4007-0217-4/24/04…$15.00
https://doi.org/10.1145/3597503.3623334

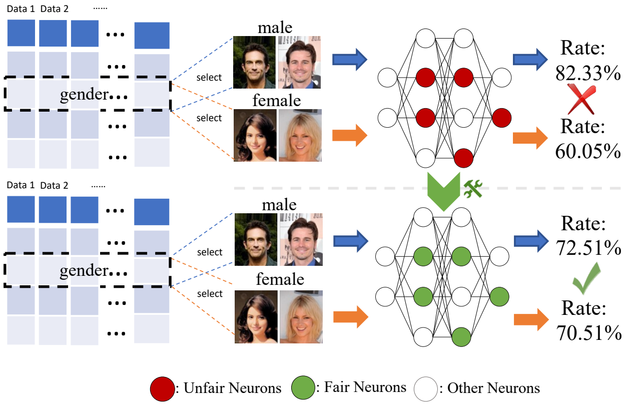


Figure 1: The group fairness problem in the task of crime risk prediction. The crime risk for the male group is far higher than that of females (*i.e.*, 82.33% versus 60.05%), though they have similar attributes except for gender. After repair, a fair model should demonstrate equal prediction qualities.

For example, a recidivism predictor based on the COMPAS dataset is likely to regard African-American offenders with higher risk scores [13]. Generally, there are two main kinds of fairness notations in existing DNN literature, including group fairness [44, 47] and individual fairness [21]. This work focuses on the group fairness of DNNs since it is prevalent in real-world applications and is difficult for automated repair, as illustrated in Fig. 1.

Most of the existing approaches for automated unfair DNNs repair are based on adversary training [1, 16, 65]. These adversary-based repair techniques usually build on the observation that optimizing for both accuracy and fairness can sometimes lead to conflicting objectives during training, and repair unfairness by introducing additional adversary modules to the DNN model or generating adversarial samples to retrain the DNN model. For example, FAD [1] adds a new hidden layer to the architecture in order to enable concurrent adversarial optimization for fairness and accuracy. Alternatively, Ethical Adversaries [16] leverages the adversary model to generate adversarial examples, which are further integrated into the training data to repair the unfairness problem. However, such complex training protocols incur a significantly higher computing overhead and complicated hyperparameter tuning, and mode collapse is also revealed to be an intractable problem for these approaches [12]. Furthermore, adversary-based techniques serve as a black-box algorithm, which means that the decision process is unknown to developers, and thus weakens the practicality. To mitigate this problem, the most recent work FairNeuron [24] conducts a two-stage scheme: 1). diagnose neurons first and 2). selectively retrains the target model, to repair the unfairness. More specifically, this scheme diagnoses the DNN with a neural network slicing technique by identifying *responsible unfair neurons* that select sensitive attributes to make predictions, and then cluster samples that trigger the selection of sensitive attributes. It retrains the unfair neurons on the clustered biased samples, in the purpose of enforcing unfair neurons to consider all features, and thus mitigates the unfairness problem without an adversary model.

Despite remarkable progress, however, prior works still have several major limitations. Firstly, although FairNeuron is more efficient for retraining, the diagnosis process is heavyweight as it requires multiple steps to conduct the profiling, forward, and backward analysis for all training and interested samples, which results in an overall inefficiency. Secondly, since FairNeuron selectively retrains the model on clustered bias samples, the unfair neurons may still pay more attention to sensitive attributes rather than all features, leading to a bottleneck in terms of unfairness repair. Thirdly, existing studies on fixing the group fairness problem only consider structured (i.e., tabular) data or unstructured data (i.e., images) with small scale and size such as MNIST, which could be a gap in real-world applications.

In this paper, we propose Responsible UNfair NEuron Repair (RUNNER) that consists of two main phases: Importance-based Neuron Diagnosis and Neuron Stabilizing Retraining. Specifically, for the efficient diagnosis, the Importance-based Neuron Diagnosis phase includes a novel criterion design based on neuron importance, which allows for the identification of responsible unfair neurons in DNNs without the need for extra multi-step analysis. Also, different from FairNeuron which identifies conflict paths composed of both neurons and synapses, our method only needs to identify the responsible unfair neurons. For effective retraining, instead of indirectly adjusting the unfair neurons, we design the Neuron Stabilizing Retraining by adding a loss term that measures the activation distance of responsible unfair neurons from different subgroups, which *directly* reduces discrimination on these neurons and improves model fairness. On top of the two steps, we design an iterative repair strategy that iteratively updates the model to further enhance the fairness.

We evaluate our approach on five popular datasets, including four tabular datasets (*i.e.*, Adult, COMPAS, Credit, and LSAC) and one large-scale dataset with high-resolution images (*i.e.*, CelebA). The experiment results demonstrate that RUNNER is more efficient and effective, and has a better generalization capability than all the existing methods. On average, RUNNER significantly reduces the computing overhead from 341.7s to 29.65s, while also achieving improved fairness by 79.3% at most. Moreover, our approach also generalizes well in unstructured image domains and achieves state-of-the-art fairness enhancement. In summary, our main contributions are as follows. ❶ We first build Importance-based Neuron Diagnosis to efficiently identify the responsible unfair neurons. ❷ We propose Neuron Stabilizing Retraining to directly repair the responsible unfair neurons to effectively improve the fairness of DNNs. ❸ We propose a novel DNN unfairness repair framework RUNNER that can coordinate the diagnosis and retraining processes by iteratively updating the target model. ❹ We conduct comprehensive experiments on mainstream datasets. The extensive comparison with existing methods confirms that RUNNER is lightweight, effective, and compatible with the unstructured image domain.

2 BACKGROUND

In this section, we briefly introduce the relevant background including Deep Neural Networks (DNNs), popular group fairness measures, and slicing & repair in DNNs.

2.1 Deep Neural Networks

A DNN typically consists of multiple layers of neurons [56], which can be formally defined as follows.

Definition 2.1. A Deep Neural Network (DNN) f consists of L multiple layers $\langle l_0, l_1, \ldots, l_{L-1} \rangle$, where l_0 is the input layer, l_{L-1} is the output layer, and $l_1, \ldots, l_{L-2}$ are hidden layers. The inputs of each layer are the outputs of the previous layer.

In this work, we mainly focus on the classifier $f : \mathcal{X} \rightarrow \mathcal{Y}$, where $\mathcal{X}$ is a set of inputs and $\mathcal{Y}$ is a set of classes. Given an input $x \in \mathcal{X}$, we use $f_l(x)$ to represent the internal features extracted by the layer l (*i.e.*, the output values of neurons at l). $\hat{y}$ represents the DNNs output (*i.e.*, $f(x)$) and y is the label of the given input x. In the real world, the samples could be divided into subgroups according to certain sensitive attributes $a \in \mathcal{A}$ such as gender and race. Without loss of generality, we consider the binary classification and binary attribute setup, *i.e.*, $y \in \{0, 1\}$ and $a \in \{0, 1\}$. For example, $a = 0$ and $a = 1$ could represent males and females, respectively.

2.2 Group Fairness Evaluation Metrics

Fairness is a multifaceted and intricate concept that has varying definitions depending on the particular context or objective being considered. In general, fairness can be divided into two categories: individual fairness [68, 70], which measures whether individuals with similar profiles (*i.e.*, only different in protected attributes) are treated with equity by the learned model, and group fairness [24], which examines whether subpopulations with different sensitive attributes are treated with equity. For instance, considering an online shopping recommendation system, all customers should be treated with equity, which requires individual fairness. On the other hand, for an AI-powered hiring system, it is crucial to treat applicants with different sensitive attributes (such as gender) with equity, making group fairness a relevant concern. At a group level, a fair outcome demands the existence of parity between different protected groups in DNNs, such as those defined by gender or race. In this work, we focus on group fairness and follow the existing works to consider two metrics to evaluate fairness: Demographic Parity [22] and Equalized Odds [26].

Demographic Parity (ΔDP) measures the difference in the probability of favorable outcomes (*i.e.*, higher prediction qualities which are evaluated by accuracy evaluation measures) between unprivileged groups ((*i.e.*, groups of higher prediction qualities)) and privileged groups (*i.e.*, groups of lower prediction qualities). Demographic Parity is achieved when individuals from both categories are predicted to fall into the positive class at the same rate. It is noteworthy that this statistic ignores the ground truth y. Demographic parity disqualifies the ideal predictor when the base rates $p(y|a)$ between the two groups are different [25].

$$\Delta \mathrm{DP} = |P(\hat{y} = 1|a = 0) - P(\hat{y} = 1|a = 1)| \tag{1}$$

Equalized Odds (ΔEO) is measured based on the true positive rate $TPR_{a=A} = P(\hat{y} = 1|a = A, y = 1)$ and the false positive rate $FPR_{a=A} = P(\hat{y} = 1|a = A, y = 0)$ for $A \in \mathcal{A}$. The measure expects favorable outcomes to be independent of the sensitive attribute, given the ground-truth prediction, which can be formulated as $P(\hat{y} = 1|a = 0, y = Y) = P(\hat{y} = 1|a = 1, y = Y)$ for $Y \in \mathcal{Y}$. To evaluate Equalized Odds, ΔEO combines the difference of TPR and FPR across two sensitive groups as where $\Delta TPR = |TPR_{a=0} - TPR_{a=1}|$ and $\Delta FPR = |FPR_{a=0} - FPR_{a=1}|$.

$$\Delta \mathrm{EO} = \Delta TPR + \Delta FPR. \tag{2}$$

Under the above definitions, ΔDP closing to 0 and ΔEO closing to 0 indicate fair classification results.

Note that the demographic parity (ΔDP) and equality of odds (ΔEO) have conflicting natures because of their distinct principles and applicability to different scenarios, making it unreasonable to satisfy both simultaneously. Specifically, a predictor $\hat{Y}$ achieves demographic parity when it is independent of the protected attribute Z. Conversely, a predictor $\hat{Y}$ satisfies equality of odds when it is conditionally independent of the protected attribute Z given the ground truth Y. However, under most scenarios, the independence between $\hat{Y}$ and Z is contradictory with the independence between $\hat{Y}$ and Z conditioned on the ground truth Y. ΔDP and ΔEO have inherent trade-offs because of their contradictory natures. Such fairness metrics have inherent trade-offs due to their conflicting spirits [34]. Following the common setups in previous work [6, 14, 59, 65], we design algorithms to enhance ΔDP and ΔEO metrics separately and conduct the corresponding measuring independently.

2.3 DNNs Slicing and Repair

Program slicing has been widely applied in a range of software engineering tasks, especially in software debugging [61]. Similarly, DNNs could also be regarded as a type of program that is constructed by artificial neurons. As DNNs easily suffer from fairness issues, localizing the DNNs' defects (*i.e.*, responsible neurons for these unfairness issues) could also contribute to the analysis and repair. However, traditional program-slicing techniques cannot be directly applied to DNNs. Therefore, Zhang et al. [69] propose NNSlicer, the first approach for slicing deep neural networks based on data flow analysis. It identifies the neurons and synapses that contribute the most to the slicing criterion by recursively backtracking from the output neurons, which form the slice.

Inspired by it, a recent work FairNeuron [24] proposes to identify responsible neurons for the unfairness through neural network slicing. The neurons are further leveraged to cluster samples to achieve selective retraining for fairness repair. Even though FairNeuron has demonstrated its efficiency and effectiveness for unfair DNN repair, there are still two main limitations. First, for diagnosis, FairNeuron requires a profiling step, a forward analysis step, and a backward analysis step to identify the responsible unfair neurons. In the profiling step, FairNeuron needs to feed all training samples into the model to calculate the average behavior of a neuron. The calculation process involves an inference process for all training samples which is time-consuming. The forward analysis step and the backward analysis step require feeding samples of interest into the model one by one. The two steps could also cause huge time overhead if the interested sample size is large. The drawbacks in these three steps make the diagnosis phase highly inefficient. Second, when retraining, FairNeuron performs sample clustering to locate the biased samples first. However, the sample clustering process might fail to accurately locate all the biased samples. Such an inaccurate selection could lead to the introduction of bias during retraining. Moreover, the retraining process employs a dropout strategy with the same training loss, which is sub-optimal. On one hand, it updates unfair neurons by training on clustered biased samples while keeping the optimization objective (i.e., the loss) unchanged, potentially resulting in unchanged unfair neurons. On the other hand, the dropout ignores neurons during the training phase of a certain set of neurons which is chosen at random. As a result, these ignored unfair neurons will not be updated.

In our work, to overcome the above limitations, we design a novel *efficient* criterion to uncover responsible unfair neurons that reduces the calculation overhead, which is further combined with the quantification of neuron discrimination in the *effective* iterative repairing process, to reduce bias for all parameters in the model and ensure the *generalization* to the unstructured image domain.

3 METHODOLOGY

3.1 Overview

The RUNNER consists of two main components: Importance-based Neuron Diagnosis and Neuron Stabilizing Retraining, as illustrated in Figure 2. In the diagnosis phase, RUNNER uses an importance-based extraction method to identify responsible unfair neurons, which significantly reduces the computational burden. In the retraining phase, we focus on the responsible unfair neurons and mitigate unfair behaviors by introducing new loss terms specifically designed for these neurons. However, retraining the model to repair current unfair neurons may result in the emergence of a new set of unfair neurons since all of the model parameters are updated after the retraining. To address this issue, we propose an iterative repair approach that involves diagnosing and retraining at each iteration in the repair process.

3.2 Importance-based Neuron Diagnosis

The main goal of responsible unfair neuron extraction is to identify neurons that have a significant impact on producing an unfair prediction in the target model. The FairNeuron method accomplishes this by first conducting a profiling process and a forward analysis to identify "unfair" neurons, followed by a backward analysis to further select "responsible" neurons. To reduce the time cost, we need to extract the neurons both responsible and unfair in one single step.

Intuitively, instead of taking multiple steps, we can directly calculate the importance of each neuron regarding ΔDP and ΔEO to distinguish which neurons cause the unfair prediction results. The neurons with high unfairness importance (*i.e.*, importance to ΔDP and ΔEO) values should be regarded as the responsible unfair neurons. However, as discussed in Section 2.2, these measures are based on prediction rates $P(\hat{y}|a=0)$ and $P(\hat{y}|a=A, y=Y)$. While they are designed to evaluate group fairness, they cannot be propagated in a backward way and are not suitable for guiding the identification of responsible unfair neurons in DNNs. To address this challenge, we propose using the relaxed counterparts introduced by [45], denoted as $\text{Gap}_{\text{DP/EO}}$. These relaxed measures can be directly used to calculate neuron unfairness importance via a backward propagation approach. The relaxed counterparts are as follows:

$$\text{Gap}_{\text{DP}} = \left|E_{x\sim P_0}(f(x)) - E_{x\sim P_1}(f(x))\right|, \tag{3}$$

where $P_0 = P(x|a=0)$ and $P_1 = P(x|a=1)$ are the distributions of x condition on a = 0 and a = 1, respectively, and the function E(·) is to calculate the expectation under the distributions.

$$\text{Gap}_{\text{EO}} = \sum_{y\in\{0,1\}} \left|E_{x\sim P_0^y}(f(x)) - E_{x\sim P_1^y}(f(x))\right|, \tag{4}$$

where $P_0^1 = P(x|a=0, y=1)$ denotes the distribution of x condition on the $a=0$ and $y=1$, and we have similar notations for P_0^0, P_1^0, P_1^1 if we set the DNN for a binary classification task and have the label $y \in \{0,1\}$.

To estimate the neuron unfairness importance $\mathbf{UI}_m^l$ of neuron m on layer l of model f regarding $\text{Gap}_{\text{DP/EO}}$ for a group of data, we can calculate the squared difference of the $\text{Gap}_{\text{DP/EO}}$ before and after removing the neuron m:

$$\mathbf{UI}_m^l = (\text{Gap}_{\text{DP/EO}} f(\mathbf{W}) - \text{Gap}_{\text{DP/EO}} f(\mathbf{W}|w_m^l = 0))^2, \tag{5}$$

where $\mathbf{W}$ denotes the parameters of all neurons in the model and w_m^l is the parameters of neuron m on layer l. One straightforward approach to calculate $\mathbf{UI}_m^l$ for each neuron in the DNN model is estimating the changes in $\text{Gap}_{\text{DP/EO}}$ after removing each neuron m (*i.e.*, calculate Equation 5 for all neurons). However, this method is time-consuming for two reasons. To calculate the **UI** for each neuron, we must remove each neuron one by one. For larger models with many parameters, this calculation becomes computationally prohibitive.

To tackle these two key challenges, we propose to use the first-order Taylor expansion to optimize Eq. 5. Using the first-order Taylor expansion, the neuron unfairness importance estimation can be simplified to:

$$\mathbf{UI}_m^l = \left(\frac{\partial \text{Gap}_{\text{DP/EO}}}{\partial w_m^l} w_m^l\right)^2. \tag{6}$$

where $\frac{\partial \text{Gap}_{\text{DP/EO}}}{\partial w_m^l}$ is the gradient of neuron m regarding $\text{Gap}_{\text{DP/EO}}$ when calculated on a subgroup of data. Thus, the calculation of $\mathbf{UI}_m^l$ can be simplified to only require the computing of gradient $\frac{\partial \text{Gap}_{\text{DP/EO}}}{\partial w_m^l}$, which is easy to implement and has a lower time cost.

With the calculated $\mathbf{UI}_m^l$ for each neuron in layer l, the neuron unfairness importance of the layer l composed of M_l neurons could be represented as $\mathbf{UI}^l = \{\mathbf{UI}_0^l, \mathbf{UI}_1^l, ..., \mathbf{UI}_{M_l-1}^l\}$.

We further determine the most responsible unfair neurons according to the calculated neuron importance and denote them as Ω_k^l as shown below:

$$\Omega_k^l = \text{top}k(w_l, \mathbf{UI}^l) \tag{7}$$

where w_l denotes the neurons on layer l and top$k(\cdot)$ represents the top k maximum instances of the input set w_l based on certain metrics. Here we select the metric as neuron unfairness importance $\mathbf{UI}^l$. We can further depict the neuron unfairness importance of the model F as $\mathbf{\Omega_k} = \{\Omega_k^{l_0}, \Omega_k^{l_1}, ..., \Omega_k^{l_{L-1}}\}$. Noted that the number of selected responsible unfair neurons is directly controlled by hyperparameter k.

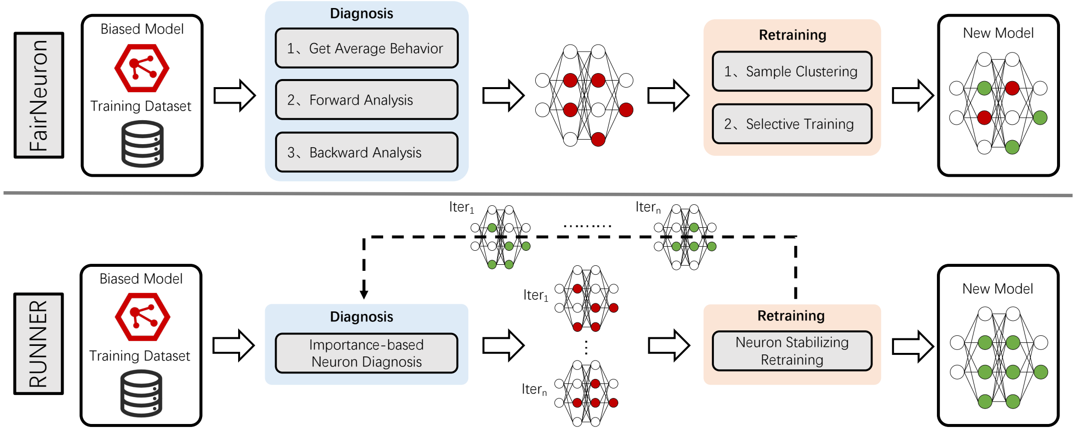


Figure 2: The comparison of the processes of FairNeuron and our method RUNNER.

3.3 Neuron Stabilizing Retraining

After identifying the responsible unfair neurons, we expect to explicitly quantify and reduce the discrimination on each neuron, instead of selecting biased samples and conducting selective training under a naive training loss. To achieve this goal, we propose a Neuron Stabilizing Retraining method that can be easily accomplished by directly adding a loss term. In particular, the loss item could measure the distance between activation values of responsible unfair neurons when the inputs are from different subgroups. We name the activation distance of different subgroups on individual neuron m on layer l as *neuron discrimination* and here we take the L_1 distance to measure *neuron discrimination* δ.

$$\delta(m, l, f, P_0, P_1) = ||E_{x\sim P_0} f_m^l(x) - E_{x\sim P_1} f_m^l(x)||_1, \tag{8}$$

where $P_0 = P(x|a = 0)$ and $P_1 = P(x|a = 1)$ are the distributions of x condition on a = 0 and a = 1, respectively, the function E(·) is to calculate the expectation under the distributions, and $f_m^l(x)$ denotes the activation value of neuron m on layer l when input sample x. We use L_{cls} to denote the cross entropy loss item. Then, the retraining loss item for the ΔDP metric could be revised as:

$$L = L_{cls} + \lambda \sum_{l}^{l\in L} \sum_{m}^{m\in\Omega^l} \delta(m, l, f, P_0, P_1), \tag{9}$$

Correspondingly, the retraining loss item for the ΔEO metric could be revised as:

$$L = L_{cls} + \lambda \sum_{l}^{l\in L} \sum_{m}^{m\in\Omega^l} \sum_{Y}^{Y\in\{0,1\}} \delta(m, l, f, P_0^Y, P_1^Y), \tag{10}$$

3.4 Iterative Repair

The iterative repair approach involves importance-based neuron diagnosis and neuron stabilizing retraining in each repair iteration. By implementing an iterative process, RUNNER can effectively identify the responsible unfair neurons in the updated model. Moreover, this process ensures that the model is continuously updated to mitigate neuron discrimination on each newly identified unfair neuron.

We detail the whole training process under the ΔDP metric and the ΔEO metric in Algorithm 1. In particular, for the ΔEO metric, given a training dataset $\mathcal{D}$, we first sample four groups of samples (*i.e.*, $(\mathbf{X}_{00}, \mathbf{Y}_{00})$, $(\mathbf{X}_{01}, \mathbf{Y}_{01})$, $(\mathbf{X}_{10}, \mathbf{Y}_{10})$ and $(\mathbf{X}_{11}, \mathbf{Y}_{11})$) from the four subgroups which are split according to the protected attribute and ground truth label (*i.e.*, $(\mathbf{X}_{ay}, \mathbf{Y}_{ay}, a \in \{0,1\}, y \in \{0,1\})$) in the dataset, respectively (See lines 2-5). Then, we calculate the cross-entropy loss for these samples (See line 6). After that, we can calculate the gradient of each neuron (i.e., parameter w_m^l) w.r.t. the Gap_{EO} (See line 9). Moreover, we select the responsible unfair neurons ω_k^l (see lines 10-11), and we calculate the neuron discrimination values for ω_k^l (*i.e.* the L_1 distance between each subgroup of data) (see lines 13-14). Finally, we integrate the repair loss item into the loss item to update the model (see lines 16-17).

4 EVALUATION

In this section, we evaluate the performance of RUNNER. We first outline the experimental setup and then, we introduce our evaluation aiming to answer the following research questions:

- **RQ1:** How effective is our method to repair unfairness?
- **RQ2:** How efficient is our method to repair unfairness?
- **RQ3:** How effective is our method applied to the large-scale dataset with high-resolution images?
- **RQ4:** How does the hyperparameter k in Eq. 7 influence the repair performance?

4.1 Experimental Setup

4.1.1 Datasets and Models. In our experiments, we use four tabular benchmarks (**Adult**, **COMPAS**, **Credit** and **LSAC**) and one high-resolution large-scale dataset (**CelebA**) that are all for binary classification tasks: ❶ **Adult** [19]. The dataset was done by Barry Becker from the 1994 Census database with 48842 instances and 14 attributes. The original aim of the dataset Adult is to determine whether a person makes salaries over 50K a year. We consider *gender* as the sensitive attribute, and the Vanilla training will lead the model to predict females to earn less salaries. ❷ **COMPAS** [46]. COMPAS (Correctional Offender Management Profiling for Alternative Sanctions) is a well-known commercial algorithm that judges

Algorithm 1: RUNNER method for fairness repair.

```
Data: Network f with $l$ layers and $M_k$ neurons in each layer, epoch
      index set $\mathcal{E}$, training data $\mathcal{D}$, batch size $bs$, network layers $L$,
      hyper-parameters $\lambda$, learning rate $\eta$.
// The iterative repair process for ΔEO
for $e \in \mathcal{E}$ do
    // Sample data subgroups from $D$
    $[\mathbf{X_{00}}, \mathbf{Y_{00}}] \leftarrow Sample(D, A = 0, Y = 0, bs)$;
    $[\mathbf{X_{01}}, \mathbf{Y_{01}}] \leftarrow Sample(D, A = 0, Y = 1, bs)$;
    $[\mathbf{X_{10}}, \mathbf{Y_{10}}] \leftarrow Sample(D, A = 1, Y = 0, bs)$;
    $[\mathbf{X_{11}}, \mathbf{Y_{11}}] \leftarrow Sample(D, A = 1, Y = 1, bs)$;
    $\mathcal{L}_{cls} \leftarrow \mathcal{L}_{\text{cls}}(\text{F}(\mathbf{X_{00}}), \mathbf{Y_{00}})) + \mathcal{L}_{\text{cls}}(\text{F}(\mathbf{X_{01}}), \mathbf{Y_{01}}) +$
      $\mathcal{L}_{\text{cls}}(\text{F}(\mathbf{X_{10}}), \mathbf{Y_{10}})) + \mathcal{L}_{\text{cls}}(\text{F}(\mathbf{X_{11}}), \mathbf{Y_{11}})$;
    $\text{Gap}_{\text{EO}} = \left|E_{x \sim X_{00}}(f(x)) - E_{x \sim X_{10}}(f(x))\right| +$
      $\left|E_{x \sim X_{01}}(f(x)) - E_{x \sim X_{11}}(f(x))\right|$
    // The Importance-base Neuron Diagnosis.
    for $l \in \mathcal{L}$ do
        $g_m^l = \frac{\partial \text{Gap}_{\text{EO}}}{\partial w_m^l}$
        $\mathbf{UI}_m^l = g_m^l w_m^l$
        $\Omega_k^l = \text{top}k(f, \mathbf{UI}^l)$
        // The Neuron Stabilizing Loss.
        for $m \in \Omega_k^l$ do
            $\delta(m, l, f, X_{00}, X_{10}) =$
              $||E_{x \sim X_{00}} f_m^l(x) - |E_{x \sim X_{10}} f_m^l(x)||_1$,
            $\delta(m, l, f, X_{01}, X_{11}) =$
              $||E_{x \sim X_{01}} f_m^l(x) - |E_{x \sim X_{11}} f_m^l(x)||_1$,
            $\mathcal{L}_{fair} += \delta(m, l, f, X_{00}, X_{10}) + \delta(m, l, f, X_{01}, X_{11})$
    $\mathcal{L} \leftarrow \mathcal{L}_{cls} + \lambda \mathcal{L}_{fair}$;
    $\theta \leftarrow \theta - \eta \nabla_{\Theta} \mathcal{L}$
// The iterative repair process for ΔDP
for $e \in \mathcal{E}$ do
    // Sample data subgroups from $D$
    $[\mathbf{X_0}, \mathbf{Y_0}] \leftarrow Sample(D, A = 0, bs)$;
    $[\mathbf{X_1}, \mathbf{Y_1}] \leftarrow Sample(D, A = 0, bs)$;
    $\mathcal{L}_{cls} \leftarrow \mathcal{L}_{\text{cls}}(\text{F}(\mathbf{X_0}), \mathbf{Y_0})) + \mathcal{L}_{\text{cls}}(\text{F}(\mathbf{X_1}), \mathbf{Y_1})$
    $\text{Gap}_{\text{DP}} = \left|E_{x \sim X_0}(f(x)) - E_{x \sim X_1}(f(x))\right|$
    // The Importance-base Neuron Diagnosis.
    for $l \in \mathcal{L}$ do
        $g_m^l = \frac{\partial \text{Gap}_{\text{DP}}}{\partial w_m^l}$
        $\mathbf{UI}_m^l = g_m^l w_m^l$
        $\Omega_k^l = \text{top}k(f, \mathbf{UI}^l)$
        // The Neuron Stabilizing Loss.
        for $m \in \Omega_k^l$ do
            $\delta(m, l, f, X_0, X_1) = ||E_{x \sim X_0} f_m^l(x) - |E_{x \sim X_1} f_m^l(x)||_1$,
            $\mathcal{L}_{fair} += \delta(m, l, f, X_0, X_1)$
    $\mathcal{L} \leftarrow \mathcal{L}_{cls} + \lambda \mathcal{L}_{fair}$;
    $\theta \leftarrow \theta - \eta \nabla_{\Theta} \mathcal{L}$
```

and parole authorities use to determine whether a criminal defendant is likely to commit another crime (recidivism). Based on a 2-year follow-up research, it has been demonstrated that the algorithm is biased against black inmates and in favor of white defendants (*i.e.*, who committed crimes or violent crimes after 2 years). ❸ **Credit** [20]. This dataset is to give an assessment of credit based on personal and financial records. In our paper, we take the attribute *gender* as the sensitive attribute. ❹**LSAC** [51]. The Law School Admissions dataset from the Law School Admissions Council (LSAC). The target is to predict whether or not a student could pass the bar, based on their Law School Admission Test (LSAT) score and undergraduate GPA. Here we take the attribute *gender* as the sensitive attribute. ❺ **CelebA** [43]. The CelebFaces Attributes dataset consists of 202599 images and is to predict the attributes of faces. We split this dataset into two subgroups according to the attribute *gender*. Here we consider two attributes classification tasks: 1). **Hairstyle classification.** In this task where to predict whether the hair in an image is *wavy*, the standard training would show discrimination towards the male group (*i.e.*, predict males as less possible to have wavy hair); 2). **Attraction Prediction.** For predicting whether the face is *attractive*, the standard training would result in a model prone to predict males as less attractive.

Following the previous work [24], for tabular benchmarks, we use the MLP (multilayer perception) [10] as the classification model, which is commonly adopted in classifying tabular data. For the CelebA dataset, we use AlexNet [35] and ResNet-18 [28], both of which are popular in classifying image data [2]. We show the experimental results of predicting the attribute *wavy hair* using ResNet-18 and predicting the attribute *attractive* using AlexNet.

4.1.2 Metrics. For fairness evaluation, we take two group fairness metrics ΔDP and ΔEO as we introduced in the section 2.2. We use the average precision (AP) for classification accuracy evaluation, which is more robust to imbalanced datasets and gives a more nuanced view of the model's performance compared with the accuracy metric [31]. We also report the accuracy metric. Ideally, a fair model should have minimized ΔDP or ΔEO while maintaining or even improving the AP at the same time.

4.1.3 Mitigation Baselines. Following the common setups in [24], we compare our method with several baselines: ❶ Vanilla. Vanilla means the standard DNN training that is based on the empirical risk minimization (ERM) principle and only with the cross entropy loss. It serves as the basis to measure how different baselines improve the fairness and keep original performance. ❷ Oversample [60]. This method samples from the subgroup with rare examples more often and adopt balanced sampling in each epoch. ❸ Reweighing [32]. In this method, the tuples in the training dataset are assigned weights. This method is identical to the Oversample method when applied to enhance the ΔEO metric. Although effective for the ΔEO metric, this method in [32] is not designed for the ΔDP. The reweighing algorithm for enhancing ΔDP is further proposed in [6]. Thus, in the following, we combine the reweighing method proposed in [6, 32] for comparison. ❹ Adversarial [1, 16, 65]. This method minimizes the adversary's ability to predict sensitive attributes. Since FAD [1] and Ethical Adversaries [16] have been well compared in one recent study [24], we consider another representative work [65] as the Adversarial baseline, which also shows better generality and stability.❺ FairNeuron [24]. FairNeuron follows a diagnosis-retraining repair paradigm and has been demonstrated to achieve state-of-the-art performance as reported. More details are introduced in Section 2.3. ❻ FairSmote [11]. Fair-SMOTE synthetically generates new data points for all the subgroups except the subgroup having the maximum number of data points. As a result, all subgroups become of equal size (same with the maximum one). ❼ ROC [33]. Reject Option-based Classification (ROC), exploits the low confidence region of a single or an ensemble of probabilistic classifiers for discrimination reduction.

4.1.4 Implementation Details. For the tabular datasets, we follow the settings in [14] for data preprocessing. The hidden size of MLP is 200. We use Adam as the learning optimizer and the batch size is set to 1000 for the ΔDP metric and 2000 for the ΔEO metric following the setting in [14]. The learning rate is set as 0.001. For the CelebA dataset, we follow the settings in [14] for data preprocessing. We use Adam as the learning optimizer and the batch size is set as 64 for the ΔDP metric and 128 for the ΔEO metric following the setting in [14]. The learning rate is set as 0.0001.

All these experiments are conducted on the Intel Xeon Silver 4214 Processor with 2 Tesla V100 GPUs with 32GB memory. We have implemented our tool based on Pytorch [48]. The source code could be found on https://github.com/Ace0001/RUNNER.

4.2 RQ1: Effectiveness of our method.

To show how our repair method outperforms other baseline methods in effectiveness, we conduct experiments on 4 widely-used public tabular datasets including Adult, COMPAS, Credit, and LSAC.

From the Table 1 and 2, we can see that, (1) Under most experimental settings, the RUNNER and baseline methods show an improvement in fairness scores (ΔDP and ΔEO), but at the cost of compromising the AP values. For example, on the Adult dataset, the Adversarial method could reduce the ΔDP from 0.170 to 0.066 while the AP also decreases from 0.781 to 0.765. The experimental results show that there might be an inherent trade-off between fairness score and accuracy performance. (2) The Oversample method could effectively improve fairness on the COMPAS, Credit, and LSAC datasets. Especially on the COMPAS dataset, the ΔEO is reduced to less than 1/2 of that of the Vanilla model (0.156 versus 0.348). However, this method also fails to improve fairness on the Adult dataset. The ΔEO score increases from 0.096 to 0.141. The potential reason could be attributed to the nature of the preprocessing method that attempts to balance the data sizes of different subgroups. The intention behind such an approach is to reduce discrimination in each iteration and ensure equal influence on the model. However, it is worth noting that adjusting the data sizes may not yield the anticipated effectiveness in balancing the influence on the model [60]. The Reweighing method performs slightly better than Oversample on the ΔDP metric. (3) The Adversarial method could also consistently improve fairness and the improvement is more salient compared with Oversample. For example, on the Adult dataset, the ΔDP score 0.066 is lower than 0.148 which is achieved by the Oversample method. However, the Adversarial method requires complex hyperparameters adjusting including adversary architecture, the learning rate of the adversary, and training interval settings to achieve a good performance. (4) FairNeuron could not effectively improve ΔDP and ΔEO. On the LSAC dataset, the ΔDP score only changes by 0.002 (from 0.007 to 0.005) while FairNeuron is not effective under other settings. These experimental results show that FairNeuron induces accurate models on the Adult dataset with multiple groups but is sub-optimal in terms of fairness. FairSmote is relatively effective for the ΔEO metric. For example, on the Credit dataset, the ΔEO is enhanced to 0.297. (5) The ROC method could saliently improve fairness, especially for the ΔDP metric while the AP degradation is huge. (6) Our method RUNNER consistently achieves the best fairness performance on all datasets under both the ΔDP and ΔEO metrics. For instance, in the Credit dataset, the ΔDP score achieved by RUNNER is close to half of the best ΔEO score attained by baseline methods (0.056 versus 0.109). Moreover, the average precision (AP) value of RUNNER (0.840) is higher than that of the Vanilla method (0.797). On the COMPAS dataset, the ΔDP improvement compared with the second-best method is 79.3% (0.006 versus 0.029) while the ap degradation is only 2.1%.

Answer to RQ1: After conducting extensive experiments on four public tabular datasets, the results demonstrate that RUNNER outperforms other baseline methods with up to 79.3% improvement in fairness score, with only a minor decline in accuracy performance.

4.3 RQ2: Efficiency of our method.

To demonstrate the superior efficiency of our repair method over the previous approach, we measure the time usage on four widely-used public tabular datasets, namely, Adult, COMPAS, Credit, and LSAC. To eliminate the effect of randomness, we conducted ten trials using random training/test data splitting for all baselines. The Vanilla training takes 28.8s, 10.1s, 8.4s, and 15.7s on the Adult, COMPAS, Credit, and LSAC datasets, respectively. As a post-processing method, the time consumption of ROC is close to that of Vanilla. In comparison, the Oversample training takes 32.1s, 13.4s, 9.3s, and 16.5s on the same datasets, respectively, which are close to the time consumption of the Reweighing method. Table 3 presents the time consumption of three more effective methods (Adversarial, FairNeuron, and RUNNER), including the number of epochs and the time required to complete the repair for each method. The Adversarial method requires an interval between training the target model and the adversary, which lengthens the training process as more epochs are needed for convergence. Similarly, the FairNeuron method requires more epochs to train due to its selective training process. For the FairNeuron method, we present the total time cost in the "time" column, and the diagnosis time and retraining time consumption in the "time$_D$" and "time$_R$" columns, respectively. As our method RUNNER conducts the diagnosis and retraining iteratively, we only show the total time cost. The results indicate that the Vanilla and ROC have the shortest training time, and the Oversample and Reweighing methods have a similar training time. However, these methods are less effective in improving fairness compared to other methods. The Adversarial method takes more time to converge, requiring more than twice the time of the Vanilla method. For example, on the Adult dataset, the Adversarial method takes 83.4 seconds, which is significantly longer than Vanilla. Regarding the FairNeuron method, we find that the diagnosis time greatly exceeds that of other methods. For instance, on the Adult dataset, the total time cost is 661.7 seconds, more than 20 times longer than Vanilla. The diagnosis time alone takes 630.7 seconds, with the responsible unfair neuron analysis process alone taking 454.6 seconds. In contrast, RUNNER takes only 66.5 seconds on the Adult dataset, and its time costs across different datasets are consistently lower

Table 1: Results on the four datasets under ΔDP metric.

	Metric	Vanilla	Oversample	Reweighing	Adversarial	FairNeuron	FairSmote	ROC	RUNNER
Adult	AP	0.781 ± 0.009	0.767 ± 0.011	0.720 ± 0.016	0.765 ± 0.015	0.787 ± 0.002	0.781 ± 0.009	0.646 ± 0.012	0.766 ± 0.011
	ACC	0.850 ± 0.005	0.825 ± 0.005	0.811 ± 0.010	0.823 ± 0.010	0.857 ± 0.001	0.852 ± 0.005	0.845 ± 0.005	0.838 ± 0.005
	ΔDP	0.170 ± 0.021	0.080 ± 0.013	0.102 ± 0.023	0.066 ± 0.011	0.171 ± 0.000	0.146 ± 0.021	0.082 ± 0.010	**0.048 ± 0.012**
COMPAS	AP	0.643 ± 0.004	0.634 ± 0.008	0.637 ± 0.003	0.640 ± 0.010	0.649 ± 0.001	0.632 ± 0.007	0.501 ± 0.012	0.623 ± 0.008
	ACC	0.655 ± 0.003	0.651 ± 0.007	0.643 ± 0.005	0.653 ± 0.010	0.658 ± 0.002	0.654 ± 0.005	0.636 ± 0.007	0.649 ± 0.006
	ΔDP	0.174 ± 0.009	0.140 ± 0.018	0.029 ± 0.017	0.125 ± 0.074	0.190 ± 0.008	0.166 ± 0.018	0.049 ± 0.012	**0.006 ± 0.004**
Credit	AP	0.797 ± 0.023	0.791 ± 0.013	0.811 ± 0.008	0.863 ± 0.003	0.846 ± 0.003	0.811 ± 0.019	0.792 ± 0.015	0.840 ± 0.009
	ACC	0.682 ± 0.036	0.655 ± 0.034	0.608 ± 0.030	0.687 ± 0.015	0.711 ± 0.006	0.660 ± 0.018	0.695 ± 0.019	0.654 ± 0.017
	ΔDP	0.134 ± 0.027	0.109 ± 0.037	0.139 ± 0.035	0.124 ± 0.052	0.156 ± 0.018	0.161 ± 0.049	0.150 ± 0.030	**0.056 ± 0.029**
LSAC	AP	0.924 ± 0.005	0.927 ± 0.004	0.930 ± 0.002	0.918 ± 0.007	0.927 ± 0.004	0.913 ± 0.008	0.879 ± 0.003	0.924 ± 0.003
	ACC	0.839 ± 0.007	0.842 ± 0.003	0.758 ± 0.016	0.836 ± 0.006	0.855 ± 0.008	0.823 ± 0.014	0.847 ± 0.001	0.840 ± 0.002
	ΔDP	0.007 ± 0.004	0.008 ± 0.006	0.008 ± 0.005	0.023 ± 0.211	0.005 ± 0.001	0.007 ± 0.004	0.004 ± 0.003	**0.004 ± 0.003**

Table 2: Results on the four datasets under ΔEO metric.

	Metric	Vanilla	Oversample	Reweighing	Adversarial	FairNeuron	FairSmote	ROC	RUNNER
Adult	AP	0.779 ± 0.013	0.762 ± 0.010	0.762 ± 0.010	0.761 ± 0.014	0.786 ± 0.001	0.785 ± 0.010	0.600 ± 0.016	0.767 ± 0.012
	ACC	0.850 ± 0.004	0.819 ± 0.006	0.819 ± 0.006	0.749 ± 0.159	0.857 ± 0.001	0.852 ± 0.003	0.825 ± 0.007	0.813 ± 0.008
	ΔEO	0.096 ± 0.038	0.141 ± 0.024	0.141 ± 0.024	0.102 ± 0.047	0.138 ± 0.004	0.104 ± 0.038	0.145 ± 0.029	**0.082 ± 0.023**
COMPAS	AP	0.641 ± 0.006	0.645 ± 0.014	0.645 ± 0.014	0.643 ± 0.005	0.649 ± 0.001	0.635 ± 0.004	0.523 ± 0.012	0.637 ± 0.007
	ACC	0.653 ± 0.005	0.653 ± 0.007	0.653 ± 0.007	0.653 ± 0.006	0.658 ± 0.002	0.654 ± 0.006	0.636 ± 0.008	0.651 ± 0.004
	ΔEO	0.348 ± 0.045	0.156 ± 0.017	0.156 ± 0.017	0.057 ± 0.017	0.353 ± 0.020	0.318 ± 0.039	0.246 ± 0.040	**0.046 ± 0.015**
Credit	AP	0.788 ± 0.014	0.784 ± 0.009	0.784 ± 0.009	0.861 ± 0.005	0.843 ± 0.006	0.808 ± 0.023	0.764 ± 0.032	0.838 ± 0.017
	ACC	0.650 ± 0.019	0.639 ± 0.022	0.639 ± 0.022	0.612 ± 0.013	0.721 ± 0.006	0.634 ± 0.021	0.636 ± 0.030	0.660 ± 0.120
	ΔEO	0.343 ± 0.068	0.260 ± 0.099	0.260 ± 0.099	0.197 ± 0.037	0.442 ± 0.001	0.297 ± 0.106	0.250 ± 0.098	**0.179 ± 0.090**
LSAC	AP	0.925 ± 0.004	0.930 ± 0.001	0.930 ± 0.001	0.913 ± 0.006	0.926 ± 0.003	0.916 ± 0.009	0.864 ± 0.006	0.908 ± 0.004
	ACC	0.841 ± 0.003	0.762 ± 0.013	0.762 ± 0.013	0.695 ± 0.024	0.855 ± 0.008	0.823 ± 0.025	0.700 ± 0.024	0.671 ± 0.011
	ΔEO	0.024 ± 0.013	0.023 ± 0.009	0.023 ± 0.009	0.024 ± 0.016	0.037 ± 0.005	0.048 ± 0.011	0.029 ± 0.011	**0.018 ± 0.020**

Table 3: Epochs and time needed to train a model.

	Adversarial		FairNeuron				FairSmote		RUNNER	
	epochs	time	epochs	time_{D}	time_{R}	time	epochs	time	epochs	time
Adult	15	83.4s	10	692.2s	36.7s	728.9s	5	100.9s	5	**66.5s**
COMPAS	15	22.7s	10	201.9s	15.5s	217.4s	5	**15.5s**	**5**	19.4s
Credit	15	21.4s	10	106.4s	10.7s	117.1s	5	10.1s	5	**9.8s**
LSAC	15	35.5s	10	345.1s	25.4s	370.5s	5	26.4s	5	**22.9s**

than those of the Adversarial methods. Although FairSmote is efficiently competitive on smaller datasets like COMPAS, Credit, and LSAC, it is less efficient on Adult since FairSmote needs to generate more data on larger datasets which is time-consuming. RUNNER is the most efficient among the three methods that are effective in improving fairness, i.e., Adversarial, FairSmote, and RUNNER. On average, our approach significantly reduces computing overhead from 341.7s of FairNeuron to 29.65s. These experimental results demonstrate that RUNNER is more efficient.

Answer to RQ2: RUNNER is significantly more efficient compared with FairNeuron and only incurs a slightly longer time cost than Vanilla and Oversample training methods.

4.4 RQ3: Generalization on the image domain.

When working with tabular datasets, the RUNNER approach is not only effective but also efficient. To demonstrate the superiority of our repair method over previous methods on unstructured datasets, we measure the fairness performance and time usage using the large-scale high-resolution public image dataset CelebA. Specifically, we use an AlexNet to classify the "attractive" attribute and a ResNet-18 to classify the "wavy hair" attribute. We compare our approach against four other baselines: Vanilla, Oversample, FairNeuron, and Adversarial which have been shown to be applicable to the image domain.

The experimental results are shown in Tables 4 and 5. We can see that the Oversample method could effectively improve the fairness metrics especially the ΔEO metric. For example, for the "attractive" classification model, Oversample improves ΔEO from 0.496 to 0.056. Moreover, the AP value also increases from 0.821 to 0.864. However, Oversample only achieves a slight change on the ΔDP metric. The AP performance of the Adversarial method is not stable although it can consistently enhance fairness compared with the Vanilla method. For example, the ΔEO scores are improved from 0.496/0.219 to 0.134/0.097, while the AP degradation is salient under the ΔEO metric (from 0.821/0.724 to 0.813/0.748) for the "attractive"/"wavy hair" classification. The reason behind the degradation might be the improper hyperparameter settings, which are extremely difficult to optimize for better performance. Our method RUNNER consistently achieves the best fairness performance for ΔDP and ΔEO. Especially on the "wavy hair" classification prediction task, RUNNER improves the ΔEO score largely (0.078 versus 0.219) and also improves the AP value from 0.724 to 0.776. The ΔEO score of 0.078 largely outperforms the second-best ΔEO score (*i.e.*,

Table 4: Results of "attractive" classification.

Metric	Vanilla	Oversample	Adversarial	FairNeuron	RUNNER
AP	0.881 ± 0.024	0.894 ± 0.015	0.859 ± 0.033	0.880 ± 0.024	0.866 ± 0.008
ACC	0.780 ± 0.037	0.800 ± 0.017	0.756 ± 0.040	0.783 ± 0.035	0.766 ± 0.009
ΔDP	0.450 ± 0.019	0.451 ± 0.018	0.282 ± 0.021	0.448 ± 0.018	**0.215 ± 0.016**
Metric	Vanilla	Oversample	Adversarial	FairNeuron	RUNNER
AP	0.821 ± 0.031	0.864 ± 0.003	0.813 ± 0.022	0.837 ± 0.041	0.867 ± 0.006
ACC	0.718 ± 0.034	0.769 ± 0.010	0.711 ± 0.024	0.734 ± 0.036	0.781 ± 0.005
ΔEO	0.496 ± 0.055	0.056 ± 0.011	0.134 ± 0.035	0.513 ± 0.036	**0.050 ± 0.013**

Table 5: Results of "wavy hair" classification.

Metric	Vanilla	Oversample	Adversarial	FairNeuron	RUNNER
AP	0.829 ± 0.011	0.801 ± 0.039	0.806 ± 0.051	0.817 ± 0.015	0.805 ± 0.050
ACC	0.783 ± 0.019	0.780 ± 0.036	0.771 ± 0.039	0.775 ± 0.026	0.771 ± 0.049
ΔDP	0.250 ± 0.036	0.253 ± 0.064	0.241 ± 0.054	0.245 ± 0.049	**0.239 ± 0.086**
Metric	Vanilla	Oversample	Adversarial	FairNeuron	RUNNER
AP	0.724 ± 0.044	0.774 ± 0.025	0.748 ± 0.041	0.766 ± 0.071	0.776 ± 0.013
ACC	0.701 ± 0.025	0.747 ± 0.061	0.715 ± 0.072	0.739 ± 0.053	0.733 ± 0.088
ΔEO	0.219 ± 0.070	0.089 ± 0.067	0.097 ± 0.054	0.269 ± 0.099	**0.078 ± 0.040**

0.089) which is achieved by the Oversample method. Therefore, we can conclude that our method RUNNER is more effective than other baselines in fairness repair on the high-resolution dataset CelebA.

Answer to RQ3: The RUNNER model exhibits strong generalization capabilities when it comes to repairing unfairness in high-resolution unstructured image datasets.

4.5 RQ4: Effects of Configurable Hyperparameters.

Our RUNNER method requires to search for a single hyperparameter, denoted as k in Equation 7. To assess the effectiveness of different values of k, we conduct a comparison experiment on both the Adult and CelebA datasets, setting k to 5%, 10%, 20%, and 50%. Results are presented in Tables 6 and 7. We find that on the Adult dataset, the best ΔDP score (0.023) is achieved when k is set to 50%, while the best ΔEO score (0.073) is achieved with k set to 5%. In contrast, on the CelebA dataset, the best ΔDP score (0.189) is achieved with k set to 5%, and the best ΔEO score of 0.050 is achieved with k set to 20%. These results suggest that the optimal value of k may not be uniform across all scenarios. Notably, our RUNNER method consistently achieves superior fairness performance on the Adult dataset compared to other baseline methods, under a broad range of hyperparameter settings. In contrast, the FairNeuron method heavily relies on searching two configurable hyperparameters, while Adversarial methods involve adjusting the adversary architecture, learning rate, and learning interval. Therefore, our RUNNER method offers a simpler and more effective solution for addressing unfairness in comparison to existing methods.

Answer to RQ4: Across multiple hyperparameter settings of k, RUNNER consistently and effectively repairs unfairness.

Table 6: Comparison of various hyperparameter k settings on the Adult Dataset.

Adult	5%	10%	20%	50%
AP	0.766 ± 0.013	0.766 ± 0.011	0.762 ± 0.008	0.751 ± 0.013
ACC	0.839 ± 0.006	0.838 ± 0.005	0.836 ± 0.005	0.833 ± 0.005
ΔDP	0.053 ± 0.011	0.048 ± 0.012	0.042 ± 0.012	**0.023 ± 0.014**
	5%	10%	20%	50%
AP	0.762 ± 0.013	0.767 ± 0.012	0.763 ± 0.014	0.762 ± 0.009
ACC	0.810 ± 0.006	0.813 ± 0.008	0.814 ± 0.004	0.807 ± 0.005
ΔEO	**0.073 ± 0.026**	0.082 ± 0.023	0.085 ± 0.026	0.082 ± 0.027

Table 7: Comparison of various hyperparameter k settings on the CelebA Dataset for "attractive" classification.

CelebA	5%	10%	20%	50%
AP	0.858 ± 0.006	0.861 ± 0.007	0.860 ± 0.004	0.866 ± 0.008
ACC	0.751 ± 0.019	0.761 ± 0.019	0.758 ± 0.016	0.766 ± 0.009
ΔDP	**0.189 ± 0.033**	0.197 ± 0.020	0.202 ± 0.029	0.215 ± 0.016
	5%	10%	20%	50%
AP	0.865 ± 0.007	0.861 ± 0.012	0.867 ± 0.006	0.863 ± 0.009
ACC	0.775 ± 0.012	0.773 ± 0.011	0.781 ± 0.005	0.774 ± 0.010
ΔEO	0.068 ± 0.023	0.053 ± 0.013	**0.050 ± 0.013**	0.064 ± 0.025

5 DISCUSSION

As we claim in the previous section, the neurons selected in the diagnosis process should be not only "unfair" but also "responsible" for the final classification. In the following sections, we further introduce our qualitative analysis to empirically verify this point, including the degree of unfairness and responsibility in the selected neurons.

5.1 Degree of Unfairness

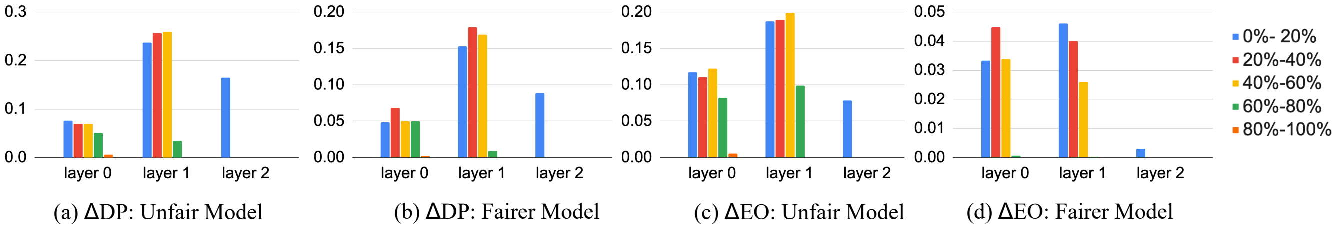
In this section, we aim to investigate the degree of unfairness in neurons selected by Importance-based Neuron Diagnosis. Neurons that exhibit disparate activation patterns towards different subgroups can be considered discriminatory and unfair. To quantify the extent of unfairness in an individual neuron m at layer l, we define its *neuron discrimination* as the difference between activation patterns observed across different subgroups. Specifically, we examine the correlation between the unfairness importance scores **UI** and the *neuron discrimination* scores.

We here take the L_1 distance to calculate the *neuron discrimination* through:

$$\delta(m, l, f, P_0, P_1) = ||E_{x\sim P_0} f(x) - E_{x\sim P_1} f(x)||_1, \qquad (11)$$

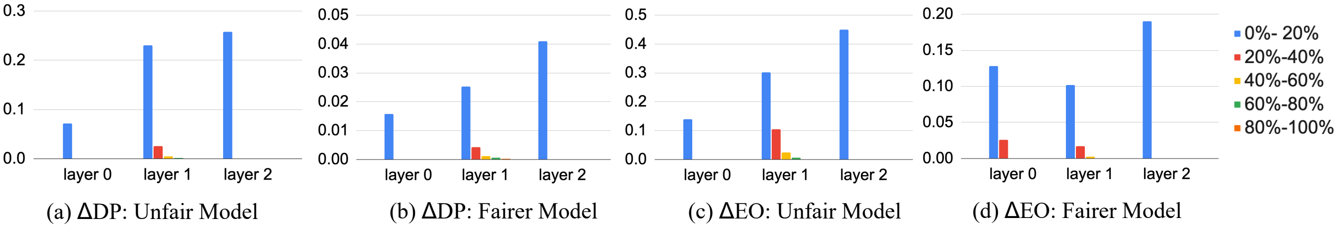
To investigate this correlation, we sorted the neurons at each model layer based on their unfairness importance scores, **UI**, and then divided them into five equal buckets. Specifically, we created buckets for the top 20% of neurons with high unfairness importance scores, the 20%-40% bucket, the 40%-80% bucket, and the last 20% of neurons (i.e., 80%-100%) with low **UI** scores.

We then calculate the average neuron discrimination value for each bucket of neurons and compared them among the buckets. We conduct this analysis on both unfair model and fairer model, and find that an unfair model consistently exhibits higher neuron



(a) ΔDP: Unfair Model (b) ΔDP: Fairer Model (c) ΔEO: Unfair Model (d) ΔEO: Fairer Model

Figure 3: Neuron discrimination on the Adult Dataset on the ΔDP and ΔEO metric.



(a) ΔDP: Unfair Model (b) ΔDP: Fairer Model (c) ΔEO: Unfair Model (d) ΔEO: Fairer Model

Figure 4: Neuron discrimination on the CelebA Dataset on the ΔDP and ΔEO metric.

discrimination values than a fairer model on both datasets and under both fairness metrics as shown in Figures 3 and 4. For example, on the Adult dataset, the average neuron discrimination value of the top 20% **UI** neurons for the fairer model is around 1/4 of that of the unfair model under the ΔEO metric on the first layer. Similarly, on the CelebA dataset, the neuron discrimination score of the top 20% **UI** neurons for the fairer model is 1/3 of that of an unfair model under the ΔEO metric on the second layer. These results indicate that fairer models have smaller neuron discrimination compared to unfair models.

Furthermore, we find that for an unfair model on the Adult dataset, the top 20% neurons exhibit similar neuron discrimination compared to the subsequent neurons. For example, the neuron discrimination of the top 20% neurons under the ΔDP metric was 0.076, which was close to that of the top 20%-40% neurons (i.e., 0.070), but much higher than that of the 60%-80% neurons and the 80%-100% neurons (i.e., 0.050 and 0.001). Different from the Adult dataset, neuron discrimination mainly exists in the top 20% neurons on the CelebA dataset. Figure 4 illustrates that the level of unfairness on the last 80% of neurons is significantly lower. These results suggest that the neurons identified by the Importance-based Neuron Diagnosis method exhibit serious discriminatory behaviors and are highly unfair.

5.2 Degree of Responsibility

We have identified discriminatory behavior in the responsible unfair neurons. We now aim to investigate the importance of these neurons to the final model prediction.

We here employ a dropout strategy to set the activation values of the neurons in each bucket as zero. Specifically, we drop out the activation values of neurons in each bucket (i.e., the top 20%, 20%-40%, 40%-80%, and the last 20% neurons) which are sorted by their unfairness importance scores **UI**, and evaluate the changes on the model's loss (Δloss) and AP (ΔAP) before and after dropout. The results are shown in Figure 5. We observe that as **UI** values decrease, the corresponding Δloss and ΔAP get closer to 0. For example, from the subfigures (c) and (d), we can see that the dropout to the last 80% neurons hardly affects the loss and AP values. This indicates that neurons with higher **UI** values have a greater impact on the final prediction. This analysis further highlights the importance of addressing discriminatory neurons in achieving fairer models.

5.3 Threats to Validity

Limited datasets. Although we evaluate RUNNER using the most common public benchmarks used in fairness repair literature, we only use 5 datasets, and therefore, we cannot conclusively determine the effectiveness and efficiency of the method on other datasets. However, since RUNNER is dataset-independent, it is straightforward to extend our evaluation to include additional datasets if they become available in the future.

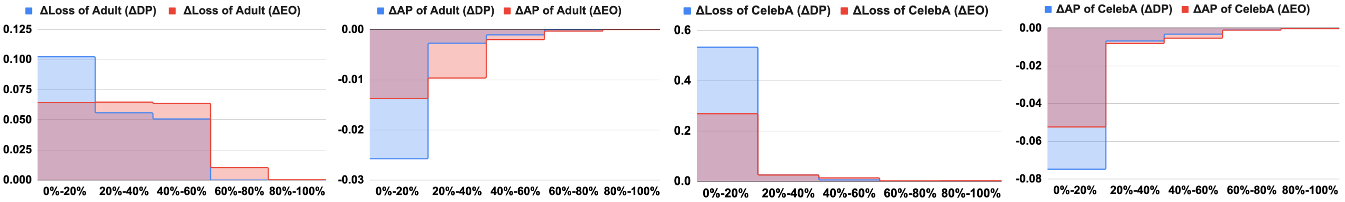
Limited model structures. In our experiments, we limited the evaluation of RUNNER to the MLP model, AlexNet, and ResNet-18. However, it is worth noting that the key idea behind RUNNER is generic and can be easily implemented for more complex neural networks.

Access to model. RUNNER is a white-box algorithm that diagnoses neurons in middle layers based on gradient calculations, which means it requires access to the model. It is widely accepted in the fairness repair literature that having full knowledge of the target model is necessary for an effective repair.

6 RELATED WORK

6.1 Unfairness in Deep Learning Systems

The unfairness of a DNN is an issue of the model's prediction relying too heavily on certain unimportant attributes of the data, resulting in bias. Unfairness is a significant issue for scenarios where equity highly matters such as standardized tests [15] and employment [49]. There are two main notations to evaluate the fairness of deep learning, i.e., individual fairness [7, 21, 23, 36, 42, 62,



(a) Δloss of Adult

(b) ΔAP of Adult

(c) Δloss of CelebA

(d) ΔAP of CelebA

Figure 5: Dropout results on the Adult and CelebA Dataset under the ΔDP and ΔEO metric.

64], group fairness [7, 18, 22, 26, 67]. More specifically, individual fairness holds the view that approximate inputs should yield similar predictions, and thus the definition of input similarity is a key issue worthy of further study. Individual discrimination is defined by whether individuals with similar profiles (*i.e.*, only different in protected attributes) in the dataset are treated equally by the learned model. Group fairness is derived by calculating and comparing the predictions for each group, which is more widely adopted in fairness research [18, 22, 26].

Fairness testing is an essential area of research, and its methods are primarily based on generation techniques. For example, ADF utilizes global and local search methods to systematically explore the input space while leveraging gradient guidance [68]. NeuronFair [70] analyzes neurons' sensitivity to individual discrimination and generates testing cases according to the behaviors of the sensitive neurons. A line of work is dedicated to alleviating this unfairness of DNNs. For example, [60] systematically compares mitigation techniques including oversample [8], adversarial training [3] and domain discriminative training [50], and proposes a simple but effective method. Moreover, [5, 52] propose to disentangle unbiased representations to ensure the fairness of DNNs. On the contrary, [17] directly repairs the classifier head regardless of whether middle representations are still biased. The methods proposed in [14, 39, 40] aim to enhance fairness through in-processing techniques. While these approaches are effective, they necessitate the computation of second-order derivatives, a process that can be computationally intensive and time-consuming. [24] diagnoses the conflict paths and selects biased samples for retraining to repair unfairness, while the diagnosis is time-consuming.

6.2 Neuron Diagnosis in DNNs

To mitigate vulnerabilities in DNNs, a common technique is to first conduct a neuron diagnosis. In addition to the DNN slicing method, the development of interpretable methods enables DNNs to present their behaviors in understandable ways for humans [4, 41, 54], which enlightens superior design of neuron diagnosis to better understand DNNs behaviors. Recently, Xie et al. [63] bases the diagnosis on the neuron relevance calculation and reveals that the responsible neurons of adversarial samples are different from that of normal samples, and it leads to wrong classification decisions. Care [55] also follows a diagnosis- repair paradigm to repair individual discrimination. However, it should be noted that individual fairness and group fairness are different problems and may require dedicated strategies separately. Compared with Care, RUNNER iteratively locates the responsible unfair neurons and repairs identified neurons. RUNNER repairs a subset of neurons in each iteration and could repair all the parameters after all iterations while Care could only repair a part of neurons and the rest remains unchanged. Moreover, RUNNER repairs the neurons by the Neuron Stabilizing Strategy, which provides loss guidance to neuron repair. However, Care relies on the PSO algorithm to search for the best parameters. In other words, RUNNER could be regarded as providing a definite searching orientation (*i.e.*, gradients generated by the loss item) for parameters to be repaired which could reduce the searching space and time cost, avoiding the risk of local optimum. Since Care doesn't claim to be capable of enhancing group fairness, we here do not include Care as a baseline for group fairness repair. Due to the space limitation, we omit to discuss other related works.

7 CONCLUSION AND FUTURE WORK

This paper presents Responsible UNfair NEuron Repair (RUNNER), a novel approach to enhance fairness in DNN models. Different from existing methods that adopt adversarial techniques or multi-step diagnosis processes, RUNNER improves them in three key aspects: efficiency, effectiveness, and generalization. Specifically, RUNNER utilizes Importance-based Neuron Diagnosis to efficiently identify responsible unfair neurons in one step using a novel importance criterion, and Neuron Stabilizing Retraining to effectively fix these neurons by adding a loss term that directly measures the activation distance of responsible unfair neurons from different subgroups. Additionally, we investigate the effectiveness of RUNNER on both structured tabular data and unstructured image data. The experimental results across a variety of datasets demonstrate that RUNNER can significantly improve model fairness in the aforementioned aspects.

In future work, we will explore different distance measures other than L_1 distance to further improve the diagnosis process. Furthermore, we can apply our approach to repair other defects of DNN models to make them more robust in practice.

ACKNOWLEDGMENTS

This research/project is supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG2-PhD-2021-08-022T). It is also supported by A*STAR Centre for Frontier AI Research, the National Research Foundation Singapore and DSO National Laboratories under the AI Singapore Programme (AISG Award No: AISG2-RP-2020-019), National Satellite of Excellence in Trustworthy Software System No. NRF2018NCR-NSOE003-0001, NRF Investigatorship No. NRF-NRFI06-2020-0001, and the National Natural Science Foundation of China 62206009. We gratefully acknowledge the support of NVIDIA AI Tech Center (NVAITC).

REFERENCES

[1] Tameem Adel, Isabel Valera, Zoubin Ghahramani, and Adrian Weller. 2019. One-Network Adversarial Fairness. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence* (Honolulu, Hawaii, USA) *(AAAI'19/IAAI'19/EAAI'19)*. AAAI Press, Article 298, 9 pages. https://doi.org/10.1609/aaai.v33i01.33012412

[2] Md Zahangir Alom, Tarek M Taha, Christopher Yakopcic, Stefan Westberg, Paheding Sidike, Mst Shamima Nasrin, Brian C Van Esesn, Abdul A S Awwal, and Vijayan K Asari. 2018. The history began from alexnet: A comprehensive survey on deep learning approaches. *arXiv preprint arXiv:1803.01164* (2018).

[3] Mohsan Alvi, Andrew Zisserman, and Christoffer Nellåker. 2018. Turning a blind eye: Explicit removal of biases and variation from deep neural network embeddings. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*. 0–0.

[4] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one* 10, 7 (2015), e0130140.

[5] Hyojin Bahng, Sanghyuk Chun, Sangdoo Yun, Jaegul Choo, and Seong Joon Oh. 2020. Learning de-biased representations with biased representations. In *International Conference on Machine Learning*. PMLR, 528–539.

[6] Hubert Baniecki, Wojciech Kretowicz, Piotr Piatyszek, Jakub Wisniewski, and Przemyslaw Biecek. 2021. dalex: Responsible Machine Learning with Interactive Explainability and Fairness in Python. *Journal of Machine Learning Research* 22, 214 (2021), 1–7. http://jmlr.org/papers/v22/20-1473.html

[7] Alex Beutel, Jilin Chen, Tulsee Doshi, Hai Qian, Allison Woodruff, Christine Luu, Pierre Kreitmann, Jonathan Bischof, and Ed H Chi. 2019. Putting fairness principles into practice: Challenges, metrics, and improvements. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*. 453–459.

[8] Steffen Bickel, Michael Brückner, and Tobias Scheffer. 2009. Discriminative learning under covariate shift. *Journal of Machine Learning Research* 10, 9 (2009).

[9] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. 2013. Evasion attacks against machine learning at test time. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part III 13*. Springer, 387–402.

[10] Christopher M. Bishop. 1996. *Neural Networks for Pattern Recognition*. Oxford University Press, Inc., USA.

[11] Joymallya Chakraborty, Suvodeep Majumder, and Tim Menzies. 2021. Bias in machine learning software: Why? how? what to do?. In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 429–440.

[12] Tong Che, Yanran Li, Athul Paul Jacob, Yoshua Bengio, and Wenjie Li. 2016. Mode regularized generative adversarial networks. *arXiv preprint arXiv:1612.02136* (2016).

[13] Alexandra Chouldechova. 2017. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data* 5, 2 (2017), 153–163.

[14] Ching-Yao Chuang and Youssef Mroueh. 2021. Fair Mixup: Fairness via Interpolation. In *International Conference on Learning Representations*. https://openreview.net/forum?id=DNl5s5BXeBn

[15] 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.

[16] 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.

[17] Mengnan Du, Subhabrata Mukherjee, Guanchu Wang, Ruixiang Tang, Ahmed Awadallah, and Xia Hu. 2021. Fairness via representation neutralization. *Advances in Neural Information Processing Systems* 34 (2021), 12091–12103.

[18] Mengnan Du, Fan Yang, Na Zou, and Xia Hu. 2020. Fairness in deep learning: A computational perspective. *IEEE Intelligent Systems* 36, 4 (2020), 25–34.

[19] Dheeru Dua and Casey Graff. 2017. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml

[20] Dheeru Dua and Casey Graff. 2017. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml

[21] 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*. 214–226.

[22] 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*. 259–268.

[23] Pratik Gajane and Mykola Pechenizkiy. 2017. On formalizing fairness in prediction with machine learning. *arXiv preprint arXiv:1710.03184* (2017).

[24] 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 *Proceedings of the 44th International Conference on Software Engineering*. 921–933.

[25] Pratyush Garg, John Villasenor, and Virginia Foggo. 2020. Fairness metrics: A comparative analysis. In *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 3662–3666.

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

[27] 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.

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 770–778. https://doi.org/10.1109/CVPR.2016.90

[29] Ming Hu, Jun Xia, Min Zhang, Xiaohong Chen, Frédéric Mallet, and Mingsong Chen. 2023. Automated Synthesis of Safe Timing Behaviors for Requirements Models using CCSL. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (2023), 1–1. https://doi.org/10.1109/TCAD.2023.3285412

[30] Ming Hu, Zeke Xia, Zhihao Yue, Jun Xia, Yihao Huang, Yang Liu, and Mingsong Chen. 2022. GitFL: Adaptive Asynchronous Federated Learning using Version Control. arXiv:2211.12049 [cs.LG]

[31] Nathalie Japkowicz and Shaju Stephen. 2002. The class imbalance problem: A systematic study. *Intelligent data analysis* 6, 5 (2002), 429–449.

[32] Faisal Kamiran and Toon Calders. 2012. Data preprocessing techniques for classification without discrimination. *Knowledge and information systems* 33, 1 (2012), 1–33.

[33] Faisal Kamiran, Asim Karim, and Xiangliang Zhang. 2012. Decision theory for discrimination-aware classification. In *2012 IEEE 12th international conference on data mining*. IEEE, 924–929.

[34] Jon M. Kleinberg, Sendhil Mullainathan, and Manish Raghavan. 2016. Inherent Trade-Offs in the Fair Determination of Risk Scores. *ArXiv* abs/1609.05807 (2016).

[35] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger (Eds.), Vol. 25. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

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

[37] Anran Li, Lan Zhang, Juntao Tan, Yaxuan Qin, Junhao Wang, and Xiang-Yang Li. 2021. Sample-level data selection for federated learning. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 1–10.

[38] Anran Li, Lan Zhang, Junhao Wang, Juntao Tan, Feng Han, Yaxuan Qin, Nikolaos M Freris, and Xiang-Yang Li. 2021. Efficient federated-learning model debugging. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. IEEE, 372–383.

[39] Tianlin Li, Qing Guo, Aishan Liu, Mengnan Du, Zhiming Li, and Yang Liu. 2023. FAIRER: Fairness as Decision Rationale Alignment. arXiv:2306.15299 [cs.LG]

[40] Tianlin Li, Zhiming Li, Anran Li, Mengnan Du, Aishan Liu, Qing Guo, Guozhu Meng, and Yang Liu. 2023. Fairness via Group Contribution Matching. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, Edith Elkind (Ed.). International Joint Conferences on Artificial Intelligence Organization, 436–445. https://doi.org/10.24963/ijcai.2023/49 Main Track.

[41] Tianlin Li, Aishan Liu, Xianglong Liu, Yitao Xu, Chongzhi Zhang, and Xiaofei Xie. 2021. Understanding adversarial robustness via critical attacking route. *Information Sciences* 547 (2021), 568–578. https://doi.org/10.1016/j.ins.2020.08.043

[42] Tianlin Li, Xiaofei Xie, Jian Wang, Qing Guo, Aishan Liu, Lei Ma, and Yang Liu. 2023. Faire: Repairing Fairness of Neural Networks via Neuron Condition Synthesis. *ACM Trans. Softw. Eng. Methodol.* (aug 2023). https://doi.org/10.1145/3617168 Just Accepted.

[43] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2015. Deep Learning Face Attributes in the Wild. In *Proceedings of International Conference on Computer Vision (ICCV)*.

[44] Gilles Louppe, Michael Kagan, and Kyle Cranmer. 2017. Learning to pivot with adversarial networks. *Advances in neural information processing systems* 30 (2017).

[45] David Madras, Elliot Creager, Toniann Pitassi, and Richard Zemel. 2018. Learning adversarially fair and transferable representations. In *International Conference on Machine Learning*. PMLR, 3384–3393.

[46] Tom Van Mele and many others. 2017-2021. COMPAS: A framework for computational research in architecture and structures. https://doi.org/10.5281/zenodo.2594510 http://compas.dev.

[47] Daniel Moyer, Shuyang Gao, Rob Brekelmans, Aram Galstyan, and Greg Ver Steeg. 2018. Invariant representations without adversarial training. *Advances in Neural Information Processing Systems* 31 (2018).

[48] 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. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 8024–8035. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[49] 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.

[50] Marco Saerens, Patrice Latinne, and Christine Decaestecker. 2002. Adjusting the outputs of a classifier to new a priori probabilities: a simple procedure. *Neural computation* 14, 1 (2002), 21–41.

[51] Richard H Sander. 2004. A systemic analysis of affirmative action in American law schools. *Stan. L. Rev.* 57 (2004), 367.

[52] Mhd Hasan Sarhan, Nassir Navab, Abouzar Eslami, and Shadi Albarqouni. 2020. Fairness by learning orthogonal disentangled representations. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16*. Springer, 746–761.

[53] Reza Shokri and Vitaly Shmatikov. 2015. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. 1310–1321.

[54] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning important features through propagating activation differences. In *International conference on machine learning*. PMLR, 3145–3153.

[55] Bing Sun, Jun Sun, Hong Long Pham, and Jie Shi. 2022. Causality-based Neural Network Repair. arXiv:2204.09274 [cs.SE]

[56] Naftali Tishby and Noga Zaslavsky. 2015. Deep learning and the information bottleneck principle. In *2015 ieee information theory workshop (itw)*. IEEE, 1–5.

[57] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[58] DeLiang Wang and Jitong Chen. 2018. Supervised speech separation based on deep learning: An overview. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 26, 10 (2018), 1702–1726.

[59] Guanchu Wang, Mengnan Du, Ninghao Liu, Na Zou, and Xia Hu. 2022. Mitigating Algorithmic Bias with Limited Annotations. *arXiv preprint arXiv:2207.10018* (2022).

[60] Zeyu Wang, Klint Qinami, Ioannis Christos Karakozis, Kyle Genova, Prem Nair, Kenji Hata, and Olga Russakovsky. 2020. Towards fairness in visual recognition: Effective strategies for bias mitigation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 8919–8928.

[61] Mark Weiser. 1984. Program Slicing. *IEEE Transactions on Software Engineering* SE-10, 4 (1984), 352–357. https://doi.org/10.1109/TSE.1984.5010248

[62] Yisong Xiao, Aishan Liu, Tianlin Li, and Xianglong Liu. 2023. Latent Imitator: Generating Natural Individual Discriminatory Instances for Black-Box Fairness Testing. arXiv:2305.11602 [cs.SE]

[63] Xiaofei Xie, Tianlin Li, Jian Wang, L. Ma, Qing Guo, Felix Juefei-Xu, and Yang Liu. 2022. NPC: Neuron Path Coverage via Characterizing Decision Logic of Deep Neural Networks. *ACM Transactions on Software Engineering and Methodology (TOSEM)* 31 (2022), 1 – 27.

[64] Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. 2013. Learning fair representations. In *International conference on machine learning*. PMLR, 325–333.

[65] 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*. 335–340.

[66] Jian Zhang, Xu Wang, Hongyu Zhang, Hailong Sun, Kaixuan Wang, and Xudong Liu. 2019. A novel neural source code representation based on abstract syntax tree. In *Proceedings of the 41st International Conference on Software Engineering*. IEEE Press, 783–794.

[67] 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*. 1335–1344.

[68] 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*. 949–960.

[69] Ziqi Zhang, Yuanchun Li, Yao Guo, Xiangqun Chen, and Yunxin Liu. 2020. Dynamic Slicing for Deep Neural Networks. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (Virtual Event, USA) *(ESEC/FSE 2020)*. Association for Computing Machinery, New York, NY, USA, 838–850. https://doi.org/10.1145/3368089.3409676

[70] Haibin Zheng, Zhiqing Chen, Tianyu Du, Xuhong Zhang, Yao Cheng, Shouling Ti, Jingyi Wang, Yue Yu, and Jinyin Chen. 2022. NeuronFair: Interpretable White-Box Fairness Testing through Biased Neuron Identification. In *2022 IEEE/ACM 44th International Conference on Software Engineering (ICSE)*. 1519–1531. https://doi.org/10.1145/3510003.3510123