NeuronFair: Interpretable White-Box Fairness Testing through Biased Neuron Identification

Haibin Zheng
Zhejiang University of Technology
haibinzheng320@gmail.com

Zhiqing Chen
Zhejiang University
zqc@zju.edu.cn

Tianyu Du
Zhejiang University
zjradty@zju.edu.cn

Xuhong Zhang
Zhejiang University
zhangxuhong@zju.edu.cn

Yao Cheng
Huawei International
chengyao101@huawei.com

Shouling Ji
Zhejiang University
sji@zju.edu.cn

Jingyi Wang
Zhejiang University
wangjyee@zju.edu.cn

Yue Yu
National University of Defense Technology
yuyue@nudt.edu.cn

Jinyin Chen*
Zhejiang University of Technology
chenjinyin@zjut.edu.cn

ABSTRACT

Deep neural networks (DNNs) have demonstrated their outperformance in various domains. However, it raises a social concern whether DNNs can produce reliable and fair decisions especially when they are applied to sensitive domains involving valuable resource allocation, such as education, loan, and employment. It is crucial to conduct fairness testing before DNNs are reliably deployed to such sensitive domains, i.e., generating as many instances as possible to uncover fairness violations. However, the existing testing methods are still limited from three aspects: interpretability, performance, and generalizability. To overcome the challenges, we propose *NeuronFair*, a new DNN fairness testing framework that differs from previous work in several key aspects: (1) *interpretable* - it quantitatively interprets DNNs' fairness violations for the biased decision; (2) *effective* - it uses the interpretation results to guide the generation of more diverse instances in less time; (3) *generic* - it can handle both structured and unstructured data. Extensive evaluations across 7 datasets and the corresponding DNNs demonstrate NeuronFair's superior performance. For instance, on structured datasets, it generates much more instances ($\sim\times$5.84) and saves more time (with an average speedup of 534.56%) compared with the state-of-the-art methods. Besides, the instances of NeuronFair can also be leveraged to improve the fairness of the biased DNNs, which helps build more fair and trustworthy deep learning systems. The code of NeuronFair is open-sourced at *https://github.com/haibinzheng/NeuronFair*.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**; • **Software and its engineering** → *Software reliability*.

*Corresponding author.

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

ACM ISBN 978-1-4503-9221-1/22/05…$15.00
https://doi.org/10.1145/3510003.3510123

KEYWORDS

Interpretability, fairness testing, discriminatory instance, deep learning, biased neuron

ACM Reference Format:
Haibin Zheng, Zhiqing Chen, Tianyu Du, Xuhong Zhang, Yao Cheng, Shouling Ji, Jingyi Wang, Yue Yu, and Jinyin Chen. 2022. NeuronFair: Interpretable White-Box Fairness Testing through Biased Neuron Identification. 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/10.1145/3510003.3510123

1 INTRODUCTION

Deep neural networks (DNNs) [39] have been increasingly adopted in many fields, including computer vision [5], natural language processing [21, 22, 40, 41], software engineering [13, 18, 34, 42, 51], etc. However, one of the crucial factors hindering DNNs from further serving applications with social impact is the unintended individual discrimination [47, 49, 59]. Individual discrimination exists when a given instance different from another only in sensitive attributes (e.g., gender, race, etc.) but receives a different prediction outcome from a given DNN [3]. Taking gender discrimination in salary prediction as an example, for two identical instances except for the gender attribute, male's annual income predicted by the DNN is often higher than female's [37]. Thus, it is of great importance for stakeholders to uncover fairness violations and then to reduce DNNs' discrimination so as to responsibly deploy fair and trustworthy deep learning systems in many sensitive scenarios [12, 26, 32, 45, 50].

Much effort has been put into uncovering fairness violations [7, 8, 11, 16, 24, 27, 31, 46, 56]. The most common method is fairness testing [2, 3, 9, 23, 25, 55, 60, 61], which solves this problem by generating as many instances as possible. Initially, fairness testing is designed to uncover and reduce the discrimination in traditional machine learning (ML) with low-dimensional linear models. However, such methods are suffering from several problems. First, most of them (e.g., FairAware [23], BlackFT [3], and FlipTest [9]) cannot handle DNNs with high-dimensional nonlinear structures. Then, though some of them (e.g., Themis [25], SymbGen [2], and Aequitas [55]) can be applied to test DNNs, they are still challenged by the high time cost and numerous duplicate instances. Recently, several methods have been specifically developed for DNNs, such as ADF [61] and EIDIG [60], etc. These methods make progress in

effectiveness and efficiency through gradient guidance, but they still suffer from the following problems.

First, these methods can hardly be generalized to unstructured data. As we know, DNNs are originally designed to process unstructured data (e.g., image, text, speech, etc.), but most existing fairness testing methods [2, 25, 55] cannot be applied to these data. It is mainly because these methods cannot determine which features are related to sensitive attributes, and cannot implement appropriate modifications to these features, e.g., how to determine pixels related to gender attribute in face images, and how to modify these pixel values to change gender [57]. However, even a seemingly simple task such as face detection [36] is subject to extreme amounts of fairness violations. It is especially concerning since these facial systems are often not deployed in isolation but rather as part of the surveillance or criminal detection pipeline [4]. Therefore, these testing methods still cannot serve DNNs widely until we solve the problem of data generalization.

Second, the generation effectiveness of these methods is challenged by gradient vanishing. They leverage the gradient-guided strategy to improve generation efficiency, but the gradient may vanish and cause instance generation to fail. Additionally, when the gradient is small, the generated instances are highly similar. However, the purpose of fairness testing is to generate not only the numerous instances, but also the diverse instances.

Third, almost all existing methods hardly provide interpretability. They only focus on generating numerous instances, but cannot interpret how the biased decisions occurred. DNNs' decision results are determined by neuron activation, then we try to study these neurons that cause biased decisions. We find that the instances generated by existing testing methods will miss the coverage of these neurons that cause biased decisions (refer to the experiment result in Fig. 6). More seriously, we cannot even know which neurons related to biased decisions have been missed for testing when there is a lack of interpretability. Therefore, we need an interpretable testing method so as to interpret DNNs' biased decisions and evaluate instances' utility for uncovering fairness violations. Based on these, the interpretation results can guide us to design effective testing to uncover more discrimination. In summary, the current fairness testing challenges lie in the lack of data generalization, generation effectiveness, and discrimination interpretation.

To overcome the above challenges, our design goals are as follows: 1) we intend to uncover and quantitatively interpret DNNs' discrimination; 2) then, we plan to apply this interpretation results to guide fairness testing; 3) furthermore, we want to generalize our testing method to unstructured data. Due to the decision results of DNNs are determined by the nonlinear combination of each neuron's activation state, thus we imagine whether the biased decisions are caused by some neurons. Then, we try to observe the neuron activation state in DNNs' hidden layers through feeding instance pair, which is two identical instances except for the sensitive attribute. Surprisingly, we find that the activation state follows such a pattern, i.e., neurons with drastically varying activation values are overlapping for different instance pairs. We observe that DNNs' discrimination is reduced when these overlapped neurons are zeroed out. Therefore, we speculate that these neurons cause the DNNs' discrimination. Then, we intend to quantitatively interpret DNNs' discrimination by computing the neuron activation difference (ActDiff) values.

According to the interpretation results, we further design a testing method, NeuronFair, to optimize gradient guidance. First, we determine the main neurons that cause discrimination, called biased neurons. Then, we search for discriminatory instances with the optimization object of increasing the ActDiff values of biased neurons. Because the optimization from the biased neuron shortens the derivation path, it reduces the probability of the gradient vanishing and time cost. Moreover, we can produce more diverse instances through the dynamic combination of biased neurons. All in all, we leverage the interpretation results to optimize gradient guidance, which is beneficial to the generation effectiveness.

We leverage adversarial attacks [14, 28, 38] to determine which features are related to sensitive attributes, and make appropriate modifications to these features. The adversarial attack is originally to test the DNNs' security, e.g., slight modifications to some image pixels will cause the predicted label to flip [10, 15, 19, 53]. Taking the gender attribute of face image as an example, we consider training a classifier with 'male' and 'female' as labels, then adding the perturbation to the face image until its predicted gender label flips. Based on this generalization framework, we can modify the sensitive attributes of any data, thereby generalizing NeuronFair to any data type.

In summary, we first implement to quantitatively interpret the discrimination using neuron-based analysis; then, we leverage the interpretation results to optimize the instance generation; finally, we design a generalization framework for sensitive attribute modification. The main contributions are as follows.

- Through the neuron activation analysis, we quantitatively interpret DNNs' discrimination, which provides a new perspective for measuring discrimination and guides DNNs' fairness testing.
- Based on the interpretation results, we design a novel method for DNNs' discriminatory instance generation, NeuronFair, which significantly outperforms previous works in terms of effectiveness.
- Inspired by adversarial attacks, we design a generalization framework to modify sensitive attributes of unstructured data, which generalizes NeuronFair to unstructured data.
- We publish our NeuronFair as a self-contained open-source toolkit online.

2 BACKGROUND

To better understand the problem we are tackling and the methodology we propose in later sections, we first introduce DNN, data form, individual discrimination, and our problem definition.

DNN. A DNN can be represented as $f(x;\Theta) : \mathcal{X} \rightarrow \mathcal{Y}$, including an input layer, several hidden layers, and an output layer [29, 58]. Two popular architectures of DNNs are fully connected network (FCN) and convolutional neural network (CNN). For a FCN, we denote the activation output of each neuron in the hidden layer as: $f_l^k(x;\Theta)$, where Θ is the weights, $l \in \{1, 2, ..., N_l\}$, N_l is the number of neural layers, $k \in \{1, 2, ..., N_l^k\}$, N_l^k is the number of neurons in the l-th layer. For a CNN, we flatten the output of the convolutional layer for the calculation of neuron activation. The loss function of

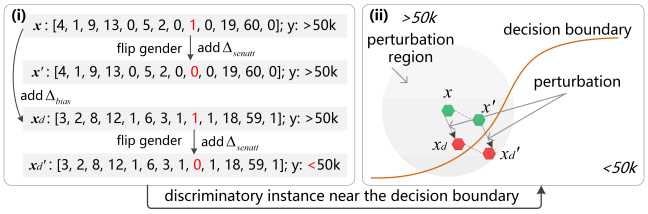


Figure 1: Illustration of discriminatory instance generation on Adult dataset [37]. (i) The discriminatory instance generation process. The normal instance pair is $< x, x' >$ and the discriminatory instance pair is $< x_d, x'_d >$. x'=x+Δ_{senatt}, x_d=x+Δ_{bias}, x'_d=x+Δ_{senatt}+Δ_{bias}, where Δ_{bias} is the bias perturbation, Δ_{senatt} is the perturbation added to the gender attribute to flip gender. (ii) Discrimination exists when the instance's predicted label changes as the gender attribute is flipped, i.e., the instance crosses the decision boundary.

DNNs is defined as follows:

$$J(x, y; \Theta) = -\frac{1}{N}\left[\sum_{i=0}^{N-1}\sum_{j=0}^{M-1}\left(y_{i,j} \times \log(\hat{y}_{i,j})\right)\right] \quad (1)$$

where N is the number of instances, M is the number of classes, y_i is the ground-truth of x_i, $\hat{y}_i = f(x_i; \Theta)$ is the predicted probability, $\log(\cdot)$ is a logarithmic function.

Data Form. Denote $X = \{x_i\}$, $Y = \{y_i\}$ as a normal dataset, and its instance pairs by $<X, X'>$= $\{<x_i, x'_i>\}$, $i \in \{0, 1, ..., N-1\}$. For an instance, we denote its attributes by $A = \{a_i\}$, $i \in \{0, 1, ..., N_a-1\}$, where $A_s \subset A$ is a set of sensitive attributes, and $A_{ns} = \{a_i^{ns} | a_i^{ns} \in A$, and $a_i^{ns} \notin A_s\}$ is a set of non-sensitive attributes. Note that sensitive attributes (e.g., gender, race, age, etc.) are usually given in advance according to specific sensitive scenes.

Individual Discrimination. As stated in previous work [3, 23], individual discrimination exists when two valid inputs which differ only in the sensitive attributes but receive a different prediction result from a given DNN, as shown in Fig. 1. Such two valid inputs are called individual discriminatory instances (IDIs).

Definition 1: IDI determination. We denote $< X_d, X'_d >= \{< x_{d,i}, x'_{d,i} >\}$ as a set of IDI pairs, which satisfies:

$$\begin{aligned} & f(x_{d,i}; \Theta) \neq f(x'_{d,i}; \Theta) \\ \text{s.t. } & x_{d,i}[\mathrm{A_s}] \neq x'_{d,i}[\mathrm{A_s}],\ x_{d,i}[\mathrm{A_{ns}}] = x'_{d,i}[\mathrm{A_{ns}}] \end{aligned} \quad (2)$$

where $i \in \{0, 1, ..., N_d - 1\}$, $x_{d,i}[\mathrm{A_s}]$ represents the value of $x_{d,i}$ with respect to attribute A_s. Note that our instances are generative (e.g., maybe the age of a generated instance is 150 years old on Adult dataset), thus we need to clip their attribute values that do not exist in the input domain $\mathbb{I}$.

3 NEURONFAIR

An overview of NeuronFair is presented in Fig. 2. NeuronFair has two parts, i.e., discrimination interpretation and IDI generation based on interpretation results. During the discrimination interpretation, we first interpret why discrimination exists through neuron-based analysis. Then, we design a discrimination metric based on the interpretation result, i.e., AUC value, as shown in Fig. 2 (i). AUC is the area under AS curve, where the AS curve records the percentage of neurons above the ActDiff threshold. Finally, we leverage the AS curve to adaptively identify biased neurons, which serves for

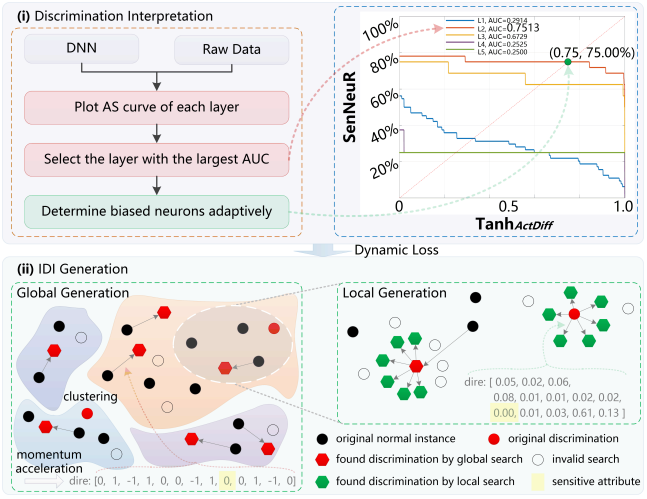


Figure 2: An overview of NeuronFair.

IDI generation. During the IDI generation, we employ the biased neurons to perform global and local generations. The global phase guarantees the diversity of the generated instances, and the local phase guarantees the quantity, as shown in Fig. 2 (ii). On the one hand, the global generation uses the normal instance as a seed and stops if an IDI is generated or it times out. On the other hand, the generated IDIs are adopted as seeds of local generation, leading to generate as many IDIs as possible near the seeds. Besides, we implement dynamic combinations of biased neurons to increase diversity, and use the momentum strategy to accelerate IDI generation. In the following, we first quantitatively interpret DNNs' discrimination, then present details of IDI generation based on interpretation results, and finally generalize NeuronFair to unstructured data.

3.1 Quantitative Discrimination Interpretation

First, we draw AS curve and compute AUC value to measure DNNs' discrimination. Then, based on the measurement results, we identify the key neurons that cause unfair decisions as biased neurons.

3.1.1 Discrimination Measurement. The ActDiff is calculated as follows:

$$z_l^k = \frac{1}{N}\sum_{i=0}^{N-1}\left[\mathrm{abs}(f_l^k(x_i; \Theta) - f_l^k(x'_i; \Theta))\right] \quad (3)$$

where z_l^k is the ActDiff of the k-th neuron in the l-th layer, $l \in \{1, 2, ..., N_l\}$, N_l is the layer number, N is the number of normal instance pairs $<X, X'>$= $\{<x_i, x'_i>\}$, $i \in \{0, 1, 2, ..., N-1\}$, $\mathrm{abs}(\cdot)$ returns an absolute value, $f_l^k(x; \Theta)$ returns the activation output of the k-th neuron in the l-th layer, Θ represents the model weights.

Based on Eq. (3), we plot AS curve and compute AUC value. We first compute each neuron's ActDiff and normalize it by hyperbolic tangent function Tanh$(\cdot)$, as shown in Fig.3 (i). "L1" means the 1-st hidden layer of a DNN with 64 neurons. Then, we set several ActDiff thresholds at equal intervals, count the neuron percentages above the ActDiff thresholds, and record them as sensitive neuron rate (SenNeuR). Finally, we plot AS curve according to the SenNeuR under different ActDiff thresholds, and then compute the area under AS curve as AUC value, as shown in Fig.3 (ii), where the x-axis is the ActDiff value normalized by Tanh function, the y-axis is SenNeuR.

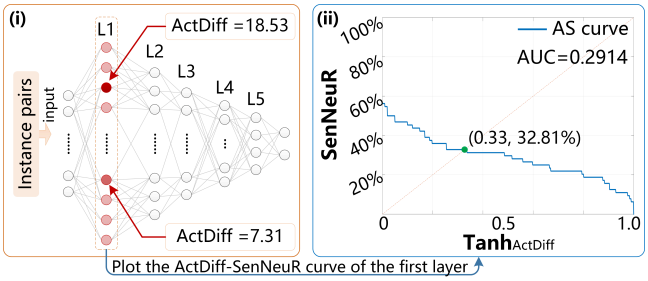


Figure 3: Illustration of the neuron-based discrimination interpretation. Dataset: Adult [37]; dimension: 13; FCN-based DNN structure: [13, 64, 32, 16, 8, 4, 1].

Algorithm 1: AS curve drawing and AUC calculation.

Input: The activation output $f_l^k(x;\Theta)$, ActDiff threshold interval $step\_interval = 0.005$, instance pairs $<X, X'>$.
Output: AS curve and AUC value of each layer.

1 Calculate the average ActDiff of each neuron: $z_l^k = \frac{1}{N}\sum_{i=0}^{N-1}[\text{abs}(f_l^k(x_i;\Theta) - f_l^k(x_i';\Theta))]$
2 **For** $l = 1 : N_l$
3 $z_l = \text{Tanh}(z_l)$
4 $max\_z = \max(z_l)$
5 $x_{tmp} = 0 : step\_interval : max\_z$
6 $y_{tmp} = [\,]$
7 **For** $count = 1 : \text{length}(x_{tmp})$
8 $y_{tmp} = [y_{tmp}, \text{length}(\text{find}(z_l > x_{tmp}[count]))]$
9 **End For**
10 $y_{tmp} = y_{tmp}/\text{length}(z_l) \times 100\%$
11 $area = \sum_{count=1}(y_{tmp}[count] \times step\_interval)$
12 Plot the AS curve based on (x_{tmp}, y_{tmp}).
Save $area$ as the AUC of the l-th layer.
13 **End For**

Repeat such an operation for each layer, we can intuitively observe the discrimination in each layer and find the most biased layer with the largest AUC value. As shown in Fig. 2 (i), the 2-nd layer 'L2' is selected as the most biased layer with AUC=0.7513.

More specific operations on AS curve drawing and AUC calculation are shown in **Algorithm 1**. First, we compute the average ActDiff values of each neuron z_l^k at line 1. In the loop from lines 7 to 9, for each neural layer, we get SenNeuR for plotting the AS curve. Then, we compute AUC value by integration at line 11.

3.1.2 Biased Neuron Identification. The most biased layer is selected for adaptive biased neuron identification. A neuron with a large z_l^k value demonstrates that it responds violently to the modification of sensitive attributes, thus it carries more discrimination. We define biased neuron as follows.

Definition 2: Biased neuron. For a given discrimination threshold T_d of the most biased layer, the biased neurons satisfy the condition $z^k > T_d$. z^k is the average ActDiff normalized by Tanh$(\cdot)$ of the k-th neuron in the most biased layer, $k \in \{1, 2, ..., N^k\}$, $T_d \in (0, 1)$.

Based on the *Definition 2*, we know that once T_d is determined, biased neurons can be found. Here we give a strategy for adaptively determining T_d. We draw a line $y = x$ that intersects the AS curve. The x-axis's value of this intersection is T_d. As shown in Fig. 3 (ii), the intersection is the point (0.33, 32.81%) and T_d=0.33, After determining T_d, we record these biased neurons and save their position p, where p is a vector with N^k elements. The strategy works well in practice as the discrimination is often concentrated on a few certain neurons, rendering a normal distribution of the ActDiff's frequency map.

Algorithm 2: Global generation guided by biased neurons.

Input: Normal instance $X = \{x_i\}$, initial set $\Omega_g = \varnothing$, $X_c = \text{KMeans}(X, N_c)$, $c \in \{1, 2, ..., N_c\}$, the number of seeds for global generation num_g, the maximum number of iterations for each seed max_iter_g, the perturbation size of each iteration $step\_size_g$, the decay rate of momentum μ_g, the step size for random disturbance r_step_g.
Output: A set of IDI pairs found globally Ω_g.

1 **For** $i = 0 : \text{INT}(num_g/N_c) - 1$
2 **For** $c = 1 : N_c$
3 Select seed x from X_c, $g_0 = g_0' = 0$.
4 **For** $t = 0 : max\_iter_g$
5 **If** ($\text{mod}(step\_size_g, r\_step_g) == 0$) **Then**
6 $r = Rand_{(0,1)}(p_r)$
7 **End If**
8 Create $<x, x'>$ s.t. $x[\text{A}_\text{s}] \neq x'[\text{A}_\text{s}]$, $x[\text{A}_\text{ns}] = x'[\text{A}_\text{ns}]$.
9 **If** $(f(x;\Theta) \neq f(x';\Theta))$ **Then**
10 $\Omega_g = \Omega_g \cup <x, x'>$
11 **break**
12 **End If**
13 $g_{t+1} = \mu_g \times g_t + \nabla_x J_{dl}(x;\Theta)$
14 $g'_{t+1} = \mu_g \times g'_t + \nabla_{x'} J_{dl}(x';\Theta)$
15 $dire = \text{sign}(g_{t+1} + g'_{t+1})$
16 $dire[\text{A}_\text{s}] = 0$
17 $x = x + dire \times step\_size_g$
18 $x = \text{Clip}(x, \mathbb{I})$
19 **End For**
20 **End For**
21 **End For**

3.2 Interpretation-based IDI Generation

NeuronFair generates IDIs in two phases, i.e., a global generation phase and a local generation phase. The global phase aims to acquire diverse IDIs. The IDIs' diversity in the global phase is crucial since these instances serve as seeds for the local phase. Instead, to guarantee the IDIs' quantity, the local phase aims to search for as many IDIs as possible near the seeds.

3.2.1 Global Generation. To increase the IDIs' diversity, we design a dynamic loss as follows:

$$J_{dl}(x;\Theta) = -\frac{1}{N}\sum_{i=0}^{N-1}\sum_{k=1}^{N^k}[(p_k|r_k) \times f^k(x_i';\Theta) \times \log(f^k(x_i;\Theta))] \quad (4)$$

where x_i' comes from x_i after flipping its sensitive attribute, N^k is the number of neurons in the most biased layer, f^k is the activation output of the k-th neuron. p is the position of biased neurons, r is the position of randomly selected neurons to increase the dynamics of $J_{dl}(x;\Theta)$. $r = Rand_{(0,1)}(p_r)$, where $Rand_{(0,1)}(p_r)$ returns a random vector with only '0' or '1'. r has the same size as p and satisfies $\sum r = \text{INT}(N^k \times p_r)$, where INT$(\cdot)$ returns an integer. Here, we set $p_r = 5\%$. '|' means 'or', $p_k|r_k = 0$ if and only $p_k = 0$ and $r_k = 0$. The optimization object of IDI generation is: **arg max** $J_{dl}(x;\Theta)$.

Algorithm 2 shows the details of global generation with momentum acceleration. We first adopt k-means clustering function KMeans(X, N_c) to process X into N_c clusters, and then get seeds from clusters in a round-robin fashion at line 3. We update random

Algorithm 3: Local generation guided by biased neurons.

Input: IDI pairs $\Omega_g = \{<x_{d,i}, x'_{d,i}>\}$, $i = \{0, 1, ..., N_g-1\}$, initial set $\Omega_l = \varnothing$, the maximum number of iterations for each seed max_iter_l, the perturbation size of each iteration $step\_size_l$, the decay rate of momentum μ_l, step size for random disturbance r_step_l.

Output: A set of IDI pairs found locally Ω_l.

1 **For** $i = 0 : N_g - 1$
2 Select seed $<x, x'>$ from Ω_g, $g_0 = g'_0 = 0$.
3 **For** $t = 0 : max\_iter_l$
4 **If** (mod($step\_size_l$, r_step_l) == 0) **Then**
5 $r = Rand_{(0,1)}(p_r)$
6 **End If**
7 $g_{t+1} = \mu_l \times g_t + \nabla_x J_{dl}(x; \Theta)$
8 $g'_{t+1} = \mu_l \times g'_t + \nabla_{x'} J_{dl}(x'; \Theta)$
9 $dire = \text{sign}(g_{t+1} + g'_{t+1})$
10 $p_{dire} = \text{SoftMax}(|g_{t+1} + g'_{t+1}|^{-1})$
11 **For** $a^{ns} \in A_{ns}$
12 Generate a random number $p_{tmp} \in (0, 1]$.
13 **If** ($p_{tmp} < p_{dire}[a^{ns}]$) **Then**
14 $x[a^{ns}] = x[a^{ns}] + dire[a^{ns}] \times step\_size_l$
15 **End If**
16 **End For**
17 $x = \text{Clip}(x, \mathbb{I})$
18 Create $<x, x'>$ s.t. $x[\text{A}_\text{s}] \neq x'[\text{A}_\text{s}]$, $x[\text{A}_\text{ns}] = x'[\text{A}_\text{ns}]$.
19 **If** ($f(x; \Theta) \neq f(x'; \Theta)$) **Then**
20 $\Omega_l = \Omega_l \cup <x, x'>$
21 **End If**
22 **End For**
23 **End For**

vector r at equal intervals from lines 5 to 7, not only to increase the dynamics but also to avoid excessively disturbing the generation task. According to *Definition 1*, we determine the IDIs from lines 8 to 12. We employ the momentum acceleration operation at lines 13 and 14, which can effectively use historical gradient and reduce invalid searches. Note that we keep the value of the sensitive attribute in x at line 16. Finally, we clip the value of x to satisfy the input domain $\mathbb{I}$.

3.2.2 Local Generation. Since the local generation aims to find as many IDIs as possible near the seeds, we increase the iteration number of each seed max_iter_l, and reduce the bias perturbation added in each iteration, as shown in **Algorithm 3**. Compared to the global phase, the major difference is the loop from lines 11 to 16, where we add perturbation to the non-sensitive attributes of large gradients with a small probability. We automatically get the probability of adding perturbation to each attribute in x at line 10.

3.3 Generalization Framework on Unstructured Data

We intend to solve the challenge of A_s modification to generalize NeuronFair to unstructured data. Here we take image data as an example. Attributes of an image are determined by pixels with normalized values between 0 and 1, i.e., the input domain of images is $\mathbb{I} \in [0, 1]$. Motivated by the adversarial attack, we design a generalization framework to implement the image's A_s modification, which modifies A_s through adding a small perturbation to most pixels, as shown in Fig. 4.

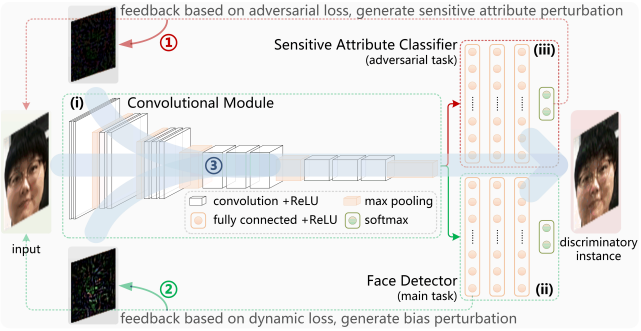


Figure 4: An overview of generalization framework on image data type.

We consider a fairness testing scenario for face detection, which determines whether the input image contains a face. The face detector consists of a CNN module (i.e., Fig. 4 (i)) and a FCN module (i.e., Fig. 4 (ii)). As shown in Fig. 4, for a given face image x and a detector $f(x, \Theta)$, there are three steps: ① build a sensitive attribute classifier; ② produce Δ_{senatt} based on Eq. (5), Δ_{senatt} is the perturbation added to image to flip sensitive attribute; ③ generate Δ_{bias} based on NeuronFair, where Δ_{bias} is the bias perturbation added to an image to flip the detection result.

First, we need a sensitive attribute classifier $f_{sa}(x; \Theta_{sa})$ that can distinguish the face image's A_s (e.g., gender). We build the A_s classifier by adding a new FCN module (i.e., Fig. 4 (iii)) to the face detector's CNN module (i.e., Fig. 4 (i)). Then, we froze the weights of the CNN module, and train the weights of the newly added FCN module.

Next, we modify the face image's A_s based on the adversarial attack. A classic adversarial attack FGSM [28] is adopted to flip the predicted result of the sensitive attribute by generating Δ_{senatt} as follows:

$$\begin{aligned} \Delta_{senatt} &= \epsilon \times \text{sign}(\nabla_x J(x, y_{sa}; \Theta_{sa})) \\ \text{satisfying that } & f_{sa}(x; \Theta_{sa}) \neq f_{sa}(x + \Delta_{senatt}; \Theta_{sa}) \end{aligned} \tag{5}$$

where ϵ is a hyper-parameter to determine perturbation size, sign$(\cdot)$ is a signum function return "-1", "0", or "1", x is an input image, y_{sa} is the ground-truth of x about sensitive attributes, Θ_{sa} is the weights of the A_s classifier.

Finally, we leverage NeuronFair to generate Δ_{bias}, and then determine whether the instance pair $<x+\Delta_{bias}, x+\Delta_{bias}+\Delta_{senatt}>$ satisfy *Definition 1*. We determine the discrimination at each layer of the detector at first. For the CNN module, the activation output of the convolutional layer is flattened. Then, in the process of image IDI generation, only the global generation is employed, which is due to the different data forms between image and structured data. Taking the input image in Fig. 4 as an example, its attributes can be regarded as $A \in \mathbb{R}^{64\times64\times3}$. Based on a seed image IDI generated in the global phase, numerous image IDIs will evolve in the local phase. However, these image IDIs are similar, with only a few pixel differences, which have little effect on fairness improvement of the face detector. Besides, we cancel the signum function sign$(\cdot)$ at line 15 of **Algorithm 2** for image data.

For other unstructured data (e.g., text), we could first process it into a vector structure similar to an image through standard embedding techniques (e.g., word embedding), then apply the framework

in Fig. 4 in a similar way. To ensure the generated inputs are realistic, we follow previous works (e.g., ADF [61] and EIDIG [60]) to limit the range of perturbed features for structured data (e.g., maximum age limit) and the maximum perturbation size for unstructured data (e.g., human-imperceptible perturbation). Distance evaluation results show that the distances between the generated inputs and the seed inputs are relatively small, which means the generated inputs are not overtly surprising in comparison to the training data.

4 EXPERIMENTAL SETTING

4.1 Datasets

We evaluate NeuronFair on 7 datasets of which five are structured datasets and two are image datasets. Each dataset is divided into three parts, i.e., 70%, 10%, 20% as training, validation, and testing, respectively.

The 5 open-source structured datasets include Adult, German credit (GerCre), bank marketing (BanMar), COMPAS, and medical expenditure panel survey (MEPS). The details of these datasets are shown in Table 1. All datasets can be downloaded from GitHub [1] and preprocessed by AI Fairness 360 toolkit (AIF360) [6].

The 2 image datasets (i.e., ClbA-IN and LFW-IN) are constructed by ourselves for face detection. ClbA-IN dataset consists of 60,000 face images from CelebA [44] and 60,000 non-face images from ImageNet [17]. LFW-IN dataset consists of 10,000 face images from LFW [33] and 10,000 non-face images from ImageNet [17]. The pixel value of each image is normalized to [0,1].

4.2 Classifiers

We implement 5 FCN-based classifiers [6, 61] for structured datasets and 2 CNN-based face detectors [30, 50, 52] for image datasets since FCN and CNN are the most widely used basic structures in real-world classification tasks.

The 5 FCN-based classifiers can be divided into two types. The one is composed of 5 hidden layers for processing low-dimensional data (i.e., Adult, GerCre, BanMar), denoted as LFCN. The another is composed of 8 hidden layers for processing high-dimensional data (i.e., COMPAS and MEPS), denoted as HFCN. The activation functions in hidden layers and the output layer are ReLU and Softmax, respectively. The hidden layer structures of LFCN and HFCN are [64, 32, 16, 8, 4] and [256, 256, 64, 64, 32, 32, 16, 8], respectively.

The 2 CNN-based face detectors serve for face detection, which are variants from two pre-trained models (i.e., VGG16 [52] and ResNet50 [30]) of *keras.applications*. We use the CNN module of VGG16 and ResNet50 as Fig. 4 (i), and design the FCN module of Fig. 4 (ii) and (iii) as [512, 256, 128, 64, 16].

4.3 Baselines

We implement and compare 4 state-of-the-art (SOTA) methods with NeuronFair to evaluate their performance, including Aequitas [55], SymbGen [2], ADF [61], and EIDIG [60]. Note that Themis [25] has been shown to be significantly less effective for DNN and thus is omitted [2, 61]. We obtained the implementation of these baselines from GitHub [2] [3]. All baselines are configured according to the best performance setting reported in the respective papers.

Table 1: Details of the datasets.

Datasets	Scenarios	Sensitive Attributes	# records	Dimensions
Adult	census income	gender, race, age	48,842	13
GerCre	credit	gender, age	1,000	20
BanMar	credit	age	41,188	16
COMPAS	law	race	5,278	400
MEPS	medical care	gender	15,675	137
ClbA-IN	face detection	gender, race	120,000	64×64×3
LFW-IN	face detection	gender, race	20,000	64×64×3

4.4 Evaluation Metrics

Five aspects of NeuronFair are evaluated, including generation *effectiveness*, *efficiency*, *interpretability*, the *utility* of AUC metric, and *generalization* of NeuronFair.

4.4.1 Generation Effectiveness Evaluation. We evaluate the effectiveness of NeuronFair on structured data from two aspects: generation quantity and quality.

(1) **Quantity**. To evaluate the generation quantity, we first count the total number of IDIs, then count the global IDIs' number and local IDIs' number respectively, recorded as '#IDIs'. Note that the duplicate instances are filtered.

(2) **Quality**. We use generation success rate (GSR), generation diversity (GD), and IDIs' contributions to fairness improvement (i.e., DM-RS [55, 60, 61]) to evaluate IDIs' quality.

$$\text{GSR} = \frac{\text{\# IDIs}}{\text{\# non}-\text{duplicate instances}} \times 100\% \tag{6}$$

where non-duplicate instances represent the input space.

$$\text{GD}_{\text{NF}}(\rho_{\text{cons}}, \text{baseline}) = \frac{\text{CR}_{\text{NF}-\text{bl}}}{\text{CR}_{\text{bl}-\text{NF}}} \tag{7}$$

where $\text{CR}_{\text{NF}-\text{bl}} = \frac{\text{\# IDIs of baselines fall in } \Pi_{\text{NF}}}{\text{\# IDIs of baseline}}$ represents the coverage rate of the NeuronFair's IDIs to baseline's IDIs, Π_{NF} is the area with NeuronFair's IDIs as the center and cosine distance ρ_{cons} as the radius; similar to $\text{CR}_{\text{bl}-\text{NF}} = \frac{\text{\# IDIs of NeuronFair fall in } \Pi_{\text{bl}}}{\text{\# IDIs of NeuronFair}}$. The NeuronFair's IDIs are more diverse when $\text{GD}_{\text{NF}} > 1$.

The generated IDIs serve to improve DNN's fairness by using these IDIs to retrain it. DM-RS is the percentage of IDIs in randomly sampled instances. High DM-RS value represents that the DNN is biased, i.e., the IDI's contribution to fairness improvement is low.

$$\text{DM}-\text{RS} = \frac{\text{\# IDIs}}{\text{\# instances randomly sampled}} \times 100\% \tag{8}$$

4.4.2 Efficiency Evaluation. We evaluate the efficiency of NeuronFair by generation speed [61], i.e., the time cost of generating 1,000 IDIs (#sec/1,000 IDIs).

4.4.3 Interpretability Evaluation based on Biased Neurons. To interpret the utility of NeuronFair, we refer to paper [48] to design the coverage of biased neurons, which is defined as follows: for a given instance, compute the activation output of the most biased layer; normalize the activation values; select neurons with activation values greater than 0.5 as the activated neurons; compare the coverage of the activated neurons to the biased neurons.

[1] https://github.com/Trusted-AI/AIF360/tree/master/aif360/data

[2] https://github.com/pxzhang94/ADF

[3] https://github.com/LingfengZhang98/EIDIG

Table 2: Parameter setting of experiments.

No.	Parameters	Values (Glo. / Loc.)	Descriptions
1	N_c	4 / ✗	the number of clusters for global generation
2	num_g	1,000 / ✗	the number of seeds for global generation
3	max_iter	40 / 1,000	the maximum number of iterations for each seed
4	$step\_size$	1.0 / 1.0	the perturbation size of each iteration
5	μ	0.1 / 0.05	the decay rate of momentum, $\mu \in (0.01, 0.20)$
6	r_step	10 / 50	the step size for random disturbance, $r\_step \in (5, 100)$

Table 3: The accuracy of classifiers and face detectors.

Datasets	Adult	GerCre	BanMar	COMPAS	MEPS	ClbA-IN	LFW-IN
Classifiers	LFC-A	LFC-G	LFC-B	HFC-C	HFC-M	VGG16	ResNet50
accuracy	88.36%	100.00%	96.71%	92.20%	98.13%	99.83%/ 92.80%/ 94.30%	99.56%/ 94.40%/ 94.20%

4.4.4 Utility Evaluation of AUC Metric. In our work, based on the interpretation results, we design AUC value to measure the discrimination. We evaluate the utility of AUC metrics from three aspects: consistency, significance, and complexity between AUC and DM-RS.

(1) **Consistency**. To evaluate the consistency, we adopt Spearman's correlation coefficient, as follows:

$$\rho_s = 1 - \frac{6\sum_{i=1}^{n} d_i^2}{n(n^2-1)} \tag{9}$$

where $d_i = a_i - b_i$, $i \in \{1, 2, ..., n\}$, a_i and b_i are the rank of AUC and DM-RS values, respectively. High ρ_s means more consistent.

(2) **Significance**. To evaluate whether AUC can measure discrimination more significantly than DM-RS, we use the standard deviation, as follows:

$$\sigma = \sqrt{\frac{\sum_{i=1} n(c_i - \mu)^2}{n-1}} \tag{10}$$

where c_i is AUC or DM-RS of different testing methods, $i \in \{1, 2, ..., n\}$, μ is the mean value of c_i. Large σ means more significant.

4.4.5 Generalization Evaluation on Image Data. We evaluate the generalization of NeuronFair on image data from two aspects: generation quantity, and quality.

(1) **Quantity**. To evaluate the generation quantity on image data, we only count the global image IDIs' number, recorded as '#IDIs'.

(2) **Quality**. We adopt GSR and IDIs' contributions to face detector's fairness improvement based on AUC value to evaluate IDIs' quality, then compute its detection rate (DR) after retraining.

4.5 Implementation Details

To fairly study the performance of the baselines and NeuronFair, our experiments have the following settings: (1) the hyperparameters of each method are set according to Table 2, where 'Glo.' and 'Loc.' represent the global and local phases, respectively; (2) the experimental results are repeated 5 times and then averaged; (3) for the FCN-based classifier, we set the learning rate to 0.001, and choose Adam as the optimizer; for the CNN-based face detector, we set the learning rate to 0.01, and choose SGD as the optimizer; the training results are shown in Table 3, where "99.83%/92.80%/94.30%" represents the accuracy of face detector, gender classifier, and race classifier, respectively.

We conduct all the experiments on a server with one Intel i7-7700K CPU running at 4.20GHz, 64 GB DDR4 memory, 4 TB HDD and one TITAN Xp 12 GB GPU card.

Table 4: Comparison with Aequitas, ADF, and EIDIG based on the total number of generated IDIs.

Datasets	**Sen. Att.**	**Aequitas**		**ADF**		**EIDIG**		**NeuronFair**	
		#IDIs	**GSR**	**#IDIs**	**GSR**	**#IDIs**	**GSR**	**#IDIs**	**GSR**
Adult	gender	1,995	8.35%	33,365	16.42%	57,386	27.24%	**122,370**	**28.19%**
	race	13,132	8.65%	57,716	23.32%	88,650	32.81%	**172,995**	**34.19%**
	age	24,495	10.48%	188,057	46.94%	251,156	48.69%	**358,201**	**49.39%**
GerCre	gender	4,347	15.24%	57,386	15.43%	64,075	17.23%	**68,218**	**36.57%**
	age	44,800	38.63%	236,551	58.74%	239,107	59.38%	**255,971**	**63.35%**
BanMar	age	10,138	27.21%	167,361	30.75%	197,341	36.26%	**302,821**	**47.76%**
COMPAS	race	658	**18.87%**	12,335	2.22%	**13,451**	2.32%	11,232	1.62%
MEPS	gender	6,132	13.51%	77,794	16.37%	101,132	21.28%	**130,898**	**27.91%**

5 EXPERIMENTAL RESULTS

We evaluate NeuronFair through answering the following five research questions (RQ): (1) how *effective* is NeuronFair; (2) how *efficient* is NeuronFair; (3) how to *interpret* the utility of NeuronFair; (4) how *useful* is the AUC metric; (5) how *generic* is NeuronFair?

5.1 Research Questions 1

How effective is NeuronFair in generating IDIs?

When reporting the results, we focus on the following aspects: generation *quantity* and *quality*.

Generation Quantity. The evaluation results are shown in Tables 4, 5, and 6, including three scenarios: the *total* number of IDIs, the IDIs number in *global* phase, and the IDIs number in *local* phase.

Implementation details for quantity evaluation: (1) SymbGen works differently from other baselines, thus we follow the comparison strategy of Zhang et al. [61], i.e., evaluating the generation quantity of NeuronFair and SymbGen within the same time (i.e., 500 sec) limit, as shown in Table 5; (2) for a fair global phase comparison, we generate 1,000 non-duplicate instances without constrained by num_g, then count IDIs number and record it in Table 6, where the seeds used are consistent for different methods; (3) for a fair local phase comparison, we mix IDIs generated globally by different methods, and randomly sample 100 as the seeds in local phase; then generate 1,000 non-duplicate instances for each seed without constrained by max_iter_l, count the IDIs number on average for each seed and record it in Table 6.

- NeuronFair generates more IDIs than baselines, especially for densely coded structured data. For instance, in Table 4, on Adult dataset with different attributes, the IDIs number of NeuronFair is 217,855 on average, which is 16.5 times and 1.6 times that of Aequitas and EIDIG, respectively. In addition, in Table 5, NeuronFair generates much more IDIs than SymbGen on all datasets. The outstanding performance of NeuronFair is mainly because the optimization object of NeuronFair takes into account the whole DNNs' discrimination information through the biased neurons while Aequitas and EIDIG only depend on the output layer. However, the IDIs number on COMPAS dataset with race attribute is 11,232, which is slightly lower than that of EIDIG. Since the COMPAS is encoded as one-hot in AIF360 [6], we speculate the reason is that too sparse data coding reduces the derivation efficiency from biased neurons.
- As shown in Table 6, NeuronFair generates much more IDIs than all baselines in the global phase, which is beneficial to increase the diversity of NeuronFair's IDIs in the subsequent local phase. For instance, on all datasets, the IDIs number of NeuronFair is 866

Table 5: Comparison with SymbGen based on the number of IDIs generated in 500 seconds.

Datasets	Sen. Att.	SymbGen		NeuronFair	
		#IDIs	GSR	#IDIs	GSR
Adult	gender	195	13.89%	**4,048**	**25.24%**
	race	452	11.01%	**4,532**	**39.54%**
	age	531	12.17%	**5,760**	**50.74%**
GerCre	gender	821	18.92%	**3,610**	**27.55%**
	age	1,034	37.19%	**3,796**	**51.40%**
BanMar	age	672	30.79%	**3,095**	**56.79%**
COMPAS	race	42	1.33%	**124**	**2.08%**
MEPS	gender	404	14.22%	**3,252**	**26.35%**

Table 6: '#IDIs' measurement in the global and local phases.

Datasets	Sen. Att.	Global Phase					Local Phase				
		Aequitas	Symb Gen	ADF	EIDIG	Neuron Fair	Aequitas	Symb Gen	ADF	EIDIG	Neuron Fair
Adult	gender	35	51	261	404	**864**	57	63	128	142	**143**
	race	98	143	332	459	**959**	134	158	174	**193**	189
	age	115	331	538	695	**974**	213	267	350	361	**367**
GerCre	gender	69	128	541	577	**599**	63	86	106	111	**113**
	age	175	247	598	599	**600**	256	301	396	400	**426**
BanMar	age	74	244	678	697	**999**	137	198	247	283	**303**
COMPAS	race	94	187	745	749	**930**	7	6	17	**18**	12
MEPS	gender	73	210	650	692	**1,000**	84	92	120	146	**149**

on average, which is 9.45 times and 1.42 times that of Aequitas and EIDIG, respectively. This is mainly because the optimization object of NeuronFair takes into account the dynamics through the dynamic combination of biased neurons. Thus, NeuronFair searches a larger space to generate more global IDIs. Besides, we conduct a preliminary T-test about "#IDIs" on the Adult and GerCre datasets. The p-values of all models are small enough to reject the null hypothesis (i.e., less than 0.05), which demonstrates the significance of NeuronFair.

- In the local phase, NeuronFair is much more efficient than baselines in general. For instance, in Table 6, on average, NeuronFair returns 78.97%, 45.35%, 10.81%, and 2.90% more IDIs than Aequitas, SymbGen, ADF, and EIDIG, respectively. Recall that Aequitas, ADF, EIDIG, and NeuronFair all guide local phase through a probability distribution, which is the likelihood of IDIs by modifying several certain attributes (i.e., the loop from lines 11 to 16 of **Algorithm 3**). The probability determination of NeuronFair takes into account the momentum and SoftMax activation (i.e., at line 10 of **Algorithm 3**) while the baselines do not. Hence, NeuronFair generates more local IDIs.

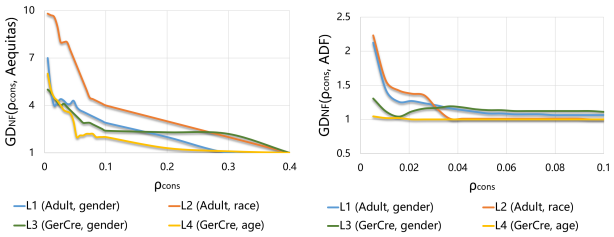
Generation Quality. The evaluation results are shown in Tables 4, 5, 7, and Fig. 5, including the generation success rate (*GSR*), generation diversity (*GD*), and fairness improvement (*DM-RS*).

Implementation details for quality evaluation: (1) for a fair diversity comparison, we seed each method with the same set of 10 global IDIs and apply them to generate 100 local IDIs for each seed without considering max_iter_l, as shown in Fig. 5; (2) we randomly select 10% IDIs of each method to retrain DNNs, then compute their fairness improvement results; to avoid contingency, we repeat 5 times and record the average DM-RS value in Table 7.

- As shown in Tables 4 and 5, the GSR values of NeuronFair are higher than that of baselines on almost all datasets, i.e., NeuronFair can search for a larger valid input space, where the input space is calculated by '#IDIs/GSR'. For instance, in Table 4, on all datasets, Aequitas has a GSR of 17.62% on average, whereas NeuronFair achieves a GSR of 36.12%, which is ~×2.1 more than that of Aequitas. The outstanding performance of NeuronFair is

Table 7: Fairness improvement measured by DM-RS, where 'Before' and 'After' represent the original and the retrained DNNs, respectively.

Datasets	Sen. Att.	Before	After				
			Aequitas	Symb Gen	ADF	EIDIG	Neuron Fair
Adult	gender	2.88%	0.45%	0.44%	0.26%	0.21%	**0.19%**
	race	8.91%	0.61%	0.81%	0.75%	0.69%	**0.57%**
	age	14.56%	4.40%	4.38%	4.18%	3.74%	**3.30%**
GerCre	gender	5.16%	0.76%	0.67%	0.55%	0.56%	**0.49%**
	age	30.90%	3.66%	3.46%	3.31%	3.21%	**2.32%**
BanMar	age	1.38%	0.68%	0.52%	0.76%	0.55%	**0.39%**
COMPAS	race	2.03%	1.48%	1.20%	0.75%	0.76%	**0.52%**
MEPS	gender	5.10%	1.30%	2.15%	1.28%	1.27%	**1.26%**



(a) Comparison with Aequitas (b) Comparison with ADF

Figure 5: Generation diversity of NeuronFair compared to Aequitas (left) and ADF (right).

mainly because it not only considers the whole DNN's discrimination through biased neurons, but also takes into account the dynamics of the optimization object through the combination of biased neurons. Thus, NeuronFair searches a larger valid input space than Aequitas.

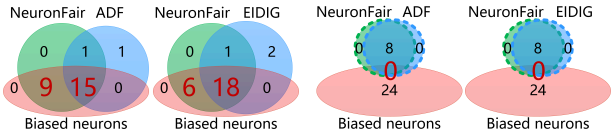
Meanwhile, the GSR value of NeuronFair on different sensitive attributes is more robust than ADF and EIDIG. For instance, in Table 4, on the GerCre dataset with gender and age attributes, the GSR values of NeuronFair are 36.57% and 63.35%, whereas that of EIDIG are 17.23% and 59.38%. We speculate the reason is that the discrimination about the gender attribute in the output layer is not obvious, but NeuronFair can find potential fairness violations through the internal discrimination information of biased neurons. Therefore, we can realize stable testing for different sensitive attributes.

In addition, the valid input space of NeuronFair is larger than baselines in general, i.e., a larger input space supports more diverse IDI generation. For instance, in Table 5, the average input space of NeuronFair is 3.30 times that of SymbGen. It is mainly because the momentum acceleration strategy employs historical gradient as auxiliary guidance, which reduces the number of invalid searches. Hence, NeuronFair generates more IDIs in a large input space.

- In all cases, NeuronFair can generate more diverse IDIs, which is beneficial to discover more potential discrimination and then improve fairness through retraining. For instance, in Fig. 5, compare to Aequitas and ADF, the GD_{NF} values are all greater than '1' under different radius values ρ_{cons}, and as the radius increases, the value of GD_{NF} gradually converges to '1'. It demonstrates that the IDIs generated by NeuronFair can always cover that of baselines. We speculate the reason is that the dynamic loss

Table 8: Time (sec) taken to generate 1,000 IDIs.

Datasets	Sen. Att.	Aequitas	Symb Gen	ADF	EIDIG	Neuron Fair
Adult	gender	345.68	1,568.20	298.46	156.38	**121.56**
	race	1,219.35	5,168.24	268.34	146.14	**114.25**
	age	484.00	2,431.09	213.76	118.85	**105.64**
GerCre	gender	436.00	2,014.68	488.19	344.10	**296.46**
	age	531.00	2,834.12	209.44	116.14	**103.91**
BanMar	age	557.00	3,015.21	472.59	246.64	**116.52**
COMPAS	race	524.13	2,315.94	253.69	199.93	**187.50**
MEPS	gender	498.16	2,537.58	217.65	182.34	**152.36**



(a) Activated neurons by IDIs (b) Activated neurons by non-IDIs

Figure 6: The overlap of biased neurons and neurons activated by different instances.

function expands the valid input space by combining different biased neurons as the optimization object.

Besides, a close investigation shows that there is a similar trend in the generation diversity for the same sensitive attribute. For instance, when ρ_{cons} <0.1 in Fig. 5 (a) or ρ_{cons} <0.02 in Fig. 5 (b), the line 'L2' with race is always the highest, the line 'L4' with age is always the lowest, while lines 'L1' and 'L3' with gender are in the middle. Since both datasets Adult and GerCre are related to money (i.e., salary and loans), we speculate that there is similar discrimination for gender in classifiers LFC-A and LFC-G for similar tasks, thus GD_{NF} shows similar trends in gender attribute.

- In all cases, NeuronFair can obtain smaller DM-RS values, i.e., the IDIs generated by NeuronFair contribute more to the DNNs' fairness improvement. For instance, in Table 7, measured by DM-RS, NeuronFair realizes fairness improvement of 87.24% on average, versus baselines, i.e., 81.18% for Aequitas, 80.78% for SymbGen, 83.30% for ADF, and 84.49% for EIDIG. It is because the IDIs of NeuronFair are more diverse than those of baselines, so it can discover more potential fairness violations and implement higher fairness improvement through retraining.

Answer to RQ1: NeuronFair outperforms the SOTA methods (i.e., Aequitas, SymbGen, ADF, and EIDIG) in two aspects: (1) *quantity* - it generates ∼×5.84 IDIs on average compared to baselines; (2) *quality* - it searches ∼×3.03 input space with more than ∼×1.65 GSR on average compared to baselines, it generates IDIs that are ∼×6.24 and ∼×1.38 more diverse than Aequitas and ADF on average with ρ_{cons} <0.02, it is beneficial to DNNs' fairness improvement of 87.24% on average.

5.2 Research Questions 2

How efficient is NeuronFair in generating IDIs?

When answering this question, we refer to the generation *speed*. The evaluation results are shown in Table 8, where the time cost of SymbGen includes generating the explainer and constraint solving. Here we have the following observation.

- NeuronFair generates IDIs more efficiently, which meets the rapidity requirements of software engineering testing. For instance, in Table 8, on average, NeuronFair takes only 26.07%, 5.47%, 49.47%, and 79.32% of the time required by Aequitas, SymbGen, ADF, and EIDIG, respectively. The outstanding performance of NeuronFair is mainly because it uses a momentum acceleration strategy and shortens the derivation path to reduce computational complexity. Hence, it takes less time than baselines.

Answer to RQ2: NeuronFair is more efficient in generation *speed* - it produces IDIs with an average speedup of 534.56%.

5.3 Research Questions 3

How to interpret NeuronFair's utility by biased neurons?

When interpreting the utility, we refer to the biased neuron *coverage*. The evaluation results are shown in Fig. 6.

Implementation details for interpretation: (1) we conduct experiments on the Adult dataset with gender attribute for the LFC-A classifier; (2) we compare the interpretation results of NeuronFair with ADF and EIDIG; (3) for a fair interpretation, we randomly select 10% IDIs and 10% non-IDIs (i.e., the generated failure instances) for each method, and then compute the coverage of biased neurons, as shown in Fig. 6.

- Biased neurons can be adopted to interpret the utility of IDIs and NeuronFair. First, IDIs trigger discrimination by activating biased neurons. For instance, the neurons activated by IDIs can cover most of the biased neurons in Fig. 6 (a), while the coverage of the biased neurons by non-IDIs of different methods is 0 in Fig. 6 (b). We can further interpret the utility of testing methods is related to the coverage of biased neurons, i.e., NeuronFair is more effective than ADF and EIDIG because they miss some discrimination contained in biased neurons while NeuronFair does not. For instance, in Fig. 6 (a), the NeuronFair's IDIs activate all 24 biased neurons in the 2-nd layer of LFC-A classifier, while the neurons activated by other IDIs cannot cover all (15 for ADF and 18 for EIDIG).

Answer to RQ3: The main reason for NeuronFair's utility is that its IDIs can activate more biased neurons. NeuronFair's IDIs activate 100% biased neurons, while 62.5% for ADF and 75% for EIDIG.

5.4 Research Questions 4

How useful is the AUC metric for measuring DNNs' fairness?

When answering this question, we refer to the following aspects: the *consistency*, *significance*, and *complexity* between AUC and DM-RS. The evaluation results on Adult and GerCre datasets with multiple sensitive attributes are shown in Table 9. From the results, we have the following observations.

- In all cases, AUC can correctly distinguish DNNs' fairness violations, i.e., AUC can serve the discrimination measurement of DNN. For instance, in Table 9, all of the ρ_s values are 1.00, indicating that the discrimination ranking results of different DNNs

Table 9: The consistency and significance between DM-RS and AUC (ρ_s, σ). Note that here we only use normal instance pairs to compute each layer's AUC, and select the maximum AUC as the classifier's discrimination.

Datasets	Sen. Att.	Metrics	Before	After					ρ_s	σ
				Aequitas	SymbGen	ADF	EIDIG	NeuronFair		
Adult	gender	DM-RS	2.88%	0.45%	0.44%	0.26%	0.21%	**0.19%**	1.00	0.0106
		AUC	0.7513	0.1492	0.1482	0.1331	0.1098	**0.0897**		**0.2563**
	race	DM-RS	8.91%	0.61%	0.81%	0.75%	0.69%	**0.57%**	1.00	0.0336
		AUC	0.8045	0.1466	0.1795	0.1652	0.1599	**0.1070**		**0.2677**
	age	DM-RS	14.56%	4.40%	4.38%	4.18%	3.74%	**3.30%**	1.00	0.0433
		AUC	0.8591	0.1565	0.1520	0.1482	0.1347	**0.1082**		**0.2941**
GerCre	gender	DM-RS	5.16%	0.76%	0.67%	0.55%	0.56%	**0.49%**	1.00	0.0186
		AUC	0.6308	0.1733	0.1568	0.1422	0.1524	**0.0960**		**0.2004**
	age	DM-RS	30.90%	3.66%	3.46%	3.31%	3.21%	**2.32%**	1.00	0.1132
		AUC	0.8608	0.2046	0.1691	0.1568	0.1432	**0.1204**		**0.2879**

Table 10: '#IDIs' and 'GSR' measurements in the global phase on image datasets.

Datasets	Sen. Att.	ADF		EIDIG		NeuronFair	
		#IDIs	GSR	#IDIs	GSR	#IDIs	GSR
ClbA-IN	gender	1,087	11.58%	2,895	12.50%	**10,578**	**69.90%**
	race	11,908	33.54%	25,180	59.87%	**51,529**	**90.15%**
LFW-IN	gender	1,204	33.20%	1,105	40.10%	**3,950**	**61.40%**
	race	2,269	31.70%	5,304	62.40%	**5,457**	**64.17%**

based on AUC are completely consistent with those based on DM-RS. Since DNN's decision results are determined by the neurons' activation, we speculate that the biased decisions are also caused by the neurons' activation, i.e., neurons contain discrimination information. Therefore, we can leverage the discrimination information in neurons to determine DNNs' fairness.

- In all cases, AUC can distinguish the different DNN's discrimination more significantly than DM-RS, which is beneficial for a more accurate evaluation of IDIs of different testing methods. For instance, in Table 9, all σ values of AUC are higher than those of DM-RS, and the average σ value of AUC is 8.46 times that of DM-RS. The outstanding performance of AUC is mainly because we use the neurons' ActDiff to measure the discrimination, which extracts more bias-related information from the whole DNN; while DM-RS only uses the bias-related information from the output layer.
- The computational complexity of AUC is much lower than that of DM-RS, which is beneficial to quickly distinguish DNNs' discrimination or the IDIs' effect. The time frequency of AUC is $T(N_l) = (7+count)\times N_l+1$ based on **Algorithm 1**. Thus the time complexity of AUC is $O(N_l)$, while that of DM-RS is $O(n \log n)$, where n is the instance number, N_l is the layer number, $n >> N_l$. It is mainly because AUC only conducts matrix operations, while DM-RS requires iterative operations until convergence.

Answer to RQ4: The AUC is useful for discrimination measurement. Compared to the results in Table 9, AUC is (1) 100% *consistent* with DM-RS, (2) ×8.46 more *significant* than DM-RS, (3) low computational *complexity* with $O(N_l)$.

5.5 Research Questions 5

How generic is NeuronFair for the task of image IDI generation?

When reporting the results, we focus on two aspects: generation *quantity* and *quality*.

Table 11: Fairness improvement of face detectors.

Datasets	Sen. Att.	Before		After					
				ADF		EIDIG		NeuronFair	
		AUC	DR	AUC	DR	AUC	DR	AUC	DR
ClbA-IN	gender	0.3587	99.83%	0.3328	97.20%	0.3091	95.40%	**0.1650**	**98.40%**
	race	0.4438		0.4045	96.50%	0.3720	95.50%	**0.2501**	**98.90%**
LFW-IN	gender	0.3910	99.56%	0.3524	95.30%	0.3678	92.30%	**0.1091**	**98.90%**
	race	0.4251		0.3984	98.10%	0.3933	96.40%	**0.2240**	**99.10%**

Implementation details for NeuronFair generalized on image data: (1) we only perform comparisons with ADF and EIDIG at global phase, because the effect of ADF and EIDIG on DNNs is much better than that of Aequitas and SymbGen; (2) we remove the KMeans($\cdot$) operation, set $step\_size_g$=0.15 for image, and all face images are used as input; (3) we retrain the face detector with all image IDIs of each method and measure its fairness improvement by AUC, (4) we measure the bias perturbation Δ_{bias} and sensitive attribute perturbation Δ_{senatt} by L_2-norm.

Generation Quantity. The evaluation results are shown in Table 10 measured by the IDIs number in *global* phase. From the results, we have the following observation.

- In all cases, NeuronFair can obtain more IDIs than ADF and EIDIG, especially for the discrimination against race attribute. For instance, in Table 10, on average, NeuronFair generates 4.34 times and 2.07 times IDIs of ADF and EIDIG, respectively. The outstanding performance of NeuronFair is because it adopts dynamic loss to expand the valid input space while ADF and EIDIG do not consider the dynamics of search. Meanwhile, the number of IDIs generated by NeuronFair for race is 3.91 times that for gender. We speculate the reason is that the pixel information related to race is mainly skin color (i.e., light & dark, or black & white), while the pixel information related to gender is more diverse (such as hair, makeup, face shape, etc.). Therefore, the image IDIs generation for race is easier through manipulating skin color.

Generation Quality. The evaluation results are shown in Tables 10 and 11, including three scenarios: the generation success rate (*GSR*), the fairness improvement (*AUC*), and the detection rate (*DR*).

- Among image data, image IDIs of NeuronFair are of higher quality than those of ADF and EIDIG, which can be applied to retrain face detectors and contribute to their fairness improvement in face detection scenarios. For instance, in Table 10, on average, the GSR value of NeuronFair is 2.60 times and 1.63 times that of ADF and EIDIG, respectively. It is because NeuronFair reduces the probability of gradient vanishing, which in turn improves the probability of non-duplicate IDIs generation guided by the gradient. Hence, all GSR values of NeuronFair are higher than those of baselines. Meanwhile, the valid input space of NeuronFair is 1.67 times and 1.27 times that of ADF and EIDIG, respectively. Since the probability of falling into a local optimum is reduced by dynamically combining biased neurons, we can perform valid searches in limited instance space.
- NeuronFair contributes more to the fairness improvement of the face detector, i.e., its generalization on image data is better than that of ADF and EIDIG. For instance, in Table 11, on average, the discrimination of detectors retrained with IDIs of NeuronFair dropped by 53.77%, while the AUC values of ADF and EIDIG

only dropped by 8.06% and 10.90%, respectively. We speculate that the valid input space of NeuronFair is larger, so its IDIs can find potential discrimination that other methods' IDIs cannot. Then improve the detector's fairness through retraining.

- NeuronFair hardly affects the detector's DR values while improving its fairness. For instance, in Table 11, on average, the DR value of detectors retrained with NeuronFair's IDIs only dropped by 0.87%, while that of ADF and EIDIG dropped by 2.92% and 4.80%, respectively. We compare the L_2 norm of Δ_{bias} and Δ_{senatt} generated by different methods, and find that Δ_{bias} of NeuronFair is much lower than that of ADF and EIDIG. Therefore, NeuronFair can not only improve the detector's fairness but also maintain its detection performance.

Answer to RQ5: The generalization performance of NeuronFair on the image dataset is better than the SOTA methods (i.e., ADF and EIDIG) in two aspects: (1) *quantity* - it generates ∼×4.34 and ∼×2.07 image IDIs on average compared to ADF and EIDIG, respectively; (2) *quality* - it searches ∼×1.47 input space with more than ∼×2.11 GSR on average, it is beneficial to detectors' fairness improvement of 53.77% on average but hardly affects their detection performance. Thus, NeuronFair shows better generalization performance than ADF and EIDIG.

6 THREATS TO VALIDITY

Correlation between attributes. The attributes of unstructured data are not as clear as structured data, so we provide a generalization framework that can modify sensitive attributes. However, there is a correlation between attributes, i.e., after the perturbation for one sensitive attribute is added, another attribute may also be changed. Since the transferability of perturbation is not robust. the slight attribute change will not affect our IDI generation.

Sensitive attributes. We consider only one sensitive attribute at a time for our experiments. However, considering multiple protected attributes will not hamper the effectiveness or generalization offered by our novel testing technique, but will certainly lead to an increase in execution time. This increase is attributed towards the fact that the algorithm in such a case, needs to consider all the possible combinations of their unique values.

Access to DNNs. NeuronFair is white-box testing that generates IDIs based on the biased neurons, which means it requires accessing to DNNs. It is widely accepted that DNN testing could have full knowledge of the target model in software engineering.

7 RELATED WORKS

Fairness Testing. Based on the software engineering point of view, several works on testing the fairness of traditional ML models are proposed [1, 2, 25, 54, 55, 59]. To uncover their fairness violations, Galhotra et al. [25] firstly proposed Themis, a fairness testing method for software, which measures the discrimination in software through counting the frequency of IDIs in the input space. However, its efficiency for IDIs generation is unsatisfactory. To improve the generation speed of Themis, Udeshi et al. [55] proposed a faster generation algorithm, Aequitas, which uncovers fairness violations by probabilistic search over the input space. Aequitas adopts a two-phase operation in which the IDIs generated globally are used as seeds for the local generation. However, Aequitas uses a global sampling distribution for all the inputs, which leads to the limitation that it can only search in narrow input space and easily falls into the local optimum. Thus, Aequitas's IDIs lack diversity. To further improve the instance diversity, Agarwal et al. [2] designed a new testing method, SymbGen, which combines the symbolic execution along with the local interpretation for the generation of effective instances. SymbGen constructs the local explainer of the complex model at first and then searches for IDIs based on the fitted decision boundary. Therefore, its instance effectiveness almost depends on the performance of the explainer.

The above-mentioned methods mainly deal with traditional ML models, which cannot directly be applied to deal with DNNs. Recently, several methods have been proposed specifically for DNNs. For instance, Zhang et al. [61] first proposed a fairness testing method specifically for DNNs, ADF, which guides the search direction through gradients. The authors proved that its effectiveness and efficiency of IDIs generation for DNNs are greatly improved based on the guidance of gradients. Based on the ADF [61], Zhang et al. [60] designed a framework EIDIG for discovering individual fairness violations, which adopts prior information to accelerate the convergence of iterations. However, there is still a problem of gradient vanishing, which may lead to a local optimum.

Neuron-based DNN Interpretation. Kim et al. [35] first introduced concept activation vectors, which provide an interpretation of a DNN's internal state (i.e., the activation output in the hidden layer). They viewed the high-dimensional internal state of a DNN as an aid, and interpreted which concept is important to the classification result. Inspired by the concept activation vectors, Du et al. [20] suggested that interpretability can serve as a useful ingredient to diagnose the reasons that lead to algorithmic discrimination. The above methods study the activation output of one hidden layer, while Liu et al. [43] studied the activation state of a single neuron. They observed that the neuron activation is related to the DNNs' robustness, and used the abnormal activation of a single neuron to detect backdoor attacks. These methods leverage the internal state to interpret DNNs' classification performance and robustness, which inspires us to use it to interpret DNNs' biased decision.

8 CONCLUSIONS

We propose an interpretable white-box fairness testing method, NeuronFair, to efficiently generate IDIs for DNNs based on biased neurons. Our method provides discrimination interpretation and IDI generation for different data forms. In the discrimination interpretation, AS curve and AUC measurement are designed to qualitatively and quantitatively interpret the severity of discrimination in each layer of DNNs, respectively. In the IDI generation, a global phase and a local phase collaborate to systematically search the input space for IDIs with the guidance of momentum acceleration and dynamic loss. Further, NeuronFair can process not only structured data but also unstructured data, e.g., image, text, etc. We compare NeuronFair with four SOTA methods in 5 structured datasets and 2 face image datasets against 7 DNNs, the results show that NeuronFair has significantly better performance in terms of interpretability, generation effectiveness, and data generalization.

ACKNOWLEDGMENT

This research was supported by NSFC (Nos. 62072406, 62102359, 61772466, 62102360), Open Research Projects of Zhejiang Lab (No. 2022RC0AB01), the Zhejiang Provincial Natural Science Foundation for Distinguished Young Scholars (No. LR19F020003), the Key R&D Projects in Zhejiang Province (Nos. 2021C01117, 2022C01018), the "Ten Thousand Talents Program" in Zhejiang Province (No. 2020R52011).

REFERENCES

[1] Julius Adebayo and Lalana Kagal. 2016. Iterative Orthogonal Feature Projection for Diagnosing Bias in Black-Box Models. *CoRR* abs/1611.04967 (2016), 1–5. arXiv:1611.04967 http://arxiv.org/abs/1611.04967

[2] Aniya Aggarwal, Pranay Lohia, Seema Nagar, Kuntal Dey, and Diptikalyan Saha. 2018. Automated Test Generation to Detect Individual Discrimination in AI Models. *CoRR* abs/1809.03260 (2018), 1–8. arXiv:1809.03260 http://arxiv.org/abs/1809.03260

[3] Aniya Aggarwal, Pranay Lohia, Seema Nagar, Kuntal Dey, and Diptikalyan Saha. 2019. Black box fairness testing of machine learning models. In *Proceedings of the ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/SIGSOFT FSE 2019, Tallinn, Estonia, August 26-30, 2019*. ACM, New York, NY, 625–635. https://doi.org/10.1145/3338906.3338937

[4] Alexander Amini, Ava P. Soleimany, Wilko Schwarting, Sangeeta N. Bhatia, and Daniela Rus. 2019. Uncovering and Mitigating Algorithmic Bias through Learned Latent Structure. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society, AIES 2019, Honolulu, HI, USA, January 27-28, 2019*. ACM, New York, NY, 289–295. https://doi.org/10.1145/3306618.3314243

[5] Maryam Badar, Muhammad Haris, and Anam Fatima. 2020. Application of deep learning for retinal image analysis: A review. *Computer Science Review* 35 (2020), 1–18. http://dx.doi.org/10.1016/j.cosrev.2019.100203

[6] Rachel K. E. Bellamy, Kuntal Dey, Michael Hind, Samuel C. Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Lohia, Jacquelyn Martino, Sameep Mehta, Aleksandra Mojsilovic, Seema Nagar, Karthikeyan Natesan Ramamurthy, John Richards, Diptikalyan Saha, Prasanna Sattigeri, Moninder Singh, Kush R. Varshney, and Yunfeng Zhang. 2018. AI Fairness 360: An Extensible Toolkit for Detecting, Understanding, and Mitigating Unwanted Algorithmic Bias. https://arxiv.org/abs/1810.01943

[7] Sumon Biswas and Hridesh Rajan. 2020. Do the machine learning models on a crowd sourced platform exhibit bias? an empirical study on model fairness. In *ESEC/FSE '20: 28th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Virtual Event, USA, November 8-13, 2020*, Prem Devanbu, Myra B. Cohen, and Thomas Zimmermann (Eds.). ACM, New York, NY, 642–653. https://doi.org/10.1145/3368089.3409704

[8] Sumon Biswas and Hridesh Rajan. 2021. Fair preprocessing: towards understanding compositional fairness of data transformers in machine learning pipeline. In *ESEC/FSE '21: 29th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Athens, Greece, August 23-28, 2021*, Diomidis Spinellis, Georgios Gousios, Marsha Chechik, and Massimiliano Di Penta (Eds.). ACM, New York, NY, 981–993. https://doi.org/10.1145/3468264.3468536

[9] Emily Black, Samuel Yeom, and Matt Fredrikson. 2020. FlipTest: fairness testing via optimal transport. In *FAT* '20: Conference on Fairness, Accountability, and Transparency, Barcelona, Spain, January 27-30, 2020*. ACM, New York, NY, 111–121. https://doi.org/10.1145/3351095.3372845

[10] Wieland Brendel, Jonas Rauber, and Matthias Bethge. 2018. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30-May 3, 2018*. OpenReview.net, [C/OL, 2018-2-16], 1–12. https://openreview.net/forum?id=SyZI0GWCZ

[11] Yuriy Brun and Alexandra Meliou. 2018. Software fairness. In *Proceedings of the 2018 ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/SIGSOFT FSE 2018, Lake Buena Vista, FL, USA, November 04-09, 2018*, Gary T. Leavens, Alessandro Garcia, and Corina S. Pasareanu (Eds.). ACM, New York, NY, 754–759. https://doi.org/10.1145/3236024.3264838

[12] Joy Buolamwini and Timnit Gebru. 2018. Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification. In *Conference on Fairness, Accountability and Transparency, FAT 2018, 23-24 February 2018, New York, NY, USA (Proceedings of Machine Learning Research, Vol. 81)*. PMLR, Stockholm, Sweden, 77–91. http://proceedings.mlr.press/v81/buolamwini18a.html

[13] Jinyin Chen, Keke Hu, Yue Yu, Zhuangzhi Chen, Qi Xuan, Yi Liu, and Vladimir Filkov. 2020. Software visualization and deep transfer learning for effective software defect prediction. In *ICSE '20: 42nd International Conference on Software Engineering, Seoul, South Korea, 27 June - 19 July, 2020*, Gregg Rothermel and Doo-Hwan Bae (Eds.). ACM, New York, NY, 578–589. https://doi.org/10.1145/3377811.3380389

[14] Jinyin Chen, Haibin Zheng, Hui Xiong, Shijing Shen, and Mengmeng Su. 2020. MAG-GAN: Massive Attack Generator via GAN. *Information Sciences* 536 (2020), 67–90. http://dx.doi.org/10.1016/j.ins.2020.04.019

[15] PinYu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and ChoJui Hsieh. 2017. ZOO: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, AISec@CCS 2017, Dallas, TX, USA, November 3, 2017*. ACM, New York, NY, 15–26. https://doi.org/10.1145/3128572.3140448

[16] Benjamin S. Clegg, Siobhán North, Phil McMinn, and Gordon Fraser. 2019. Simulating student mistakes to evaluate the fairness of automated grading. In *Proceedings of the 41st International Conference on Software Engineering: Software Engineering Education and Training, ICSE (SEET) 2019, Montreal, QC, Canada, May 25-31, 2019*, Sarah Beecham and Daniela E. Damian (Eds.). IEEE / ACM, Piscataway, NJ, 121–125. https://doi.org/10.1109/ICSE-SEET.2019.00021

[17] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, Vol. 1-4. IEEE Computer Society, Computer Society Los Alamitos, CA, 248–255. https://doi.org/10.1109/CVPR.2009.5206848

[18] Prem Devanbu, Matthew Dwyer, Sebastian Elbaum, Michael Lowry, Kevin Moran, Denys Poshyvanyk, Baishakhi Ray, Rishabh Singh, and Xiangyu Zhang. 2020. Deep Learning & Software Engineering: State of Research and Future Directions. arXiv:2009.08525 [cs.SE]

[19] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. 2018. Boosting Adversarial Attacks with Momentum. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. IEEE Computer Society, Computer Society Los Alamitos, CA, 9185–9193. https://doi.org/10.1109/CVPR.2018.00957

[20] Mengnan Du, Fan Yang, Na Zou, and Xia Hu. 2019. Fairness in Deep Learning: A Computational Perspective. *CoRR* abs/1908.08843 (2019), 1–9. arXiv:1908.08843 http://arxiv.org/abs/1908.08843

[21] Tianyu Du, Shouling Ji, Jinfeng Li, Qinchen Gu, Ting Wang, and Raheem Beyah. 2020. SirenAttack: Generating Adversarial Audio for End-to-End Acoustic Systems. In *ASIA CCS '20: The 15th ACM Asia Conference on Computer and Communications Security, Taipei, Taiwan, October 5-9, 2020*. ACM, 357–369. https://doi.org/10.1145/3320269.3384733

[22] Tianyu Du, Shouling Ji, Lujia Shen, Yao Zhang, Jinfeng Li, Jie Shi, Chengfang Fang, Jianwei Yin, Raheem Beyah, and Ting Wang. 2021. Cert-RNN: Towards Certifying the Robustness of Recurrent Neural Networks. In *CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, Republic of Korea, November 15 - 19, 2021*. ACM, 516–534. https://doi.org/10.1145/3460120.3484538

[23] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard S. Zemel. 2012. Fairness through awareness. In *Innovations in Theoretical Computer Science 2012, Cambridge, MA, USA, January 8-10, 2012*. ACM, New York, NY, 214–226. https://doi.org/10.1145/2090236.2090255

[24] Ali Farahani, Liliana Pasquale, Amel Bennaceur, Thomas Welsh, and Bashar Nuseibeh. 2021. On Adaptive Fairness in Software Systems. In *16th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, SEAMS@ICSE 2021, Madrid, Spain, May 18-24, 2021*. IEEE, Piscataway, NJ, 97–103. https://doi.org/10.1109/SEAMS51251.2021.00022

[25] Sainyam Galhotra, Yuriy Brun, and Alexandra Meliou. 2017. Fairness testing: testing software for discrimination. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering, ESEC/FSE 2017, Paderborn, Germany, September 4-8, 2017*. ACM, New York, NY, 498–510. https://doi.org/10.1145/3106237.3106277

[26] Bodhvi Gaur, Gurpreet Singh Saluja, Hamsa Bharathi Sivakumar, and Sanjay Singh. 2021. Semi-supervised deep learning based named entity recognition model to parse education section of resumes. *Neural Computing and Applications* 33 (2021), 5705–5718. https://doi.org/10.1007/s00521-020-05351-2

[27] Daniel M. Germán, Gregorio Robles, Germán Poo-Caamaño, Xin Yang, Hajimu Iida, and Katsuro Inoue. 2018. "Was my contribution fairly reviewed?": a framework to study the perception of fairness in modern code reviews. In *Proceedings of the 40th International Conference on Software Engineering, ICSE 2018, Gothenburg, Sweden, May 27 - June 03, 2018*, Michel Chaudron, Ivica Crnkovic, Marsha Chechik, and Mark Harman (Eds.). ACM, New York, NY, 523–534. https://doi.org/10.1145/3180155.3180217

[28] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015*. Arxiv, [C/OL, 2015-5-20], 1–10. http://arxiv.org/abs/1412.6572

[29] Keji Han, Yun Li, and Bin Xia. 2021. A cascade model-aware generative adversarial example detection method. *Tsinghua Science and Technology* 26, 6 (2021), 800–812. https://ieeexplore.ieee.org/document/9449325

[30] 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 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, Computer Society Los Alamitos, CA, 770–778. https://doi.org/10.1109/CVPR.2016.90

[31] Tobias Holstein and Gordana Dodig-Crnkovic. 2018. Avoiding the intrinsic unfairness of the trolley problem. In *Proceedings of the International Workshop on Software Fairness, FairWare@ICSE 2018, Gothenburg, Sweden, May 29, 2018*, Yuriy Brun, Brittany Johnson, and Alexandra Meliou (Eds.). ACM, New York, NY, 32–37. https://doi.org/10.1145/3194770.3194772

[32] Chao Huang, Junbo Zhang, Yu Zheng, and Nitesh V. Chawla. 2018. DeepCrime: Attentive Hierarchical Recurrent Networks for Crime Prediction. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*. ACM, New York, NY, 1423–1432. https://doi.org/10.1145/3269206.3271793

[33] Gary B. Huang, Marwan Mattar, Tamara Berg, and Eric Learned-Miller. 2008. Labeled Faces in the Wild: A Database forStudying Face Recognition in Unconstrained Environments. In *Workshop on Faces in 'Real-Life' Images: Detection, Alignment, and Recognition*. Erik Learned-Miller and Andras Ferencz and Frédéric Jurie, HAL-inria, Marseille, France, 1–15. https://hal.inria.fr/inria-00321923

[34] Yujin Huang, Han Hu, and Chunyang Chen. 2021. Robustness of on-Device Models: Adversarial Attack to Deep Learning Models on Android Apps. In *43rd IEEE/ACM International Conference on Software Engineering: Software Engineering in Practice, ICSE (SEIP) 2021, Madrid, Spain, May 25-28, 2021*. IEEE, Piscataway, NJ, 101–110. https://doi.org/10.1109/ICSE-SEIP52600.2021.00019

[35] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie J. Cai, James Wexler, Fernanda B. Viégas, and Rory Sayres. 2018. Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV). In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018 (Proceedings of Machine Learning Research, Vol. 80)*. PMLR, Stockholm, Sweden, 2673–2682. http://proceedings.mlr.press/v80/kim18d.html

[36] Brendan Klare, Mark James Burge, Joshua C. Klontz, Richard W. Vorder Bruegge, and Anil K. Jain. 2012. Face Recognition Performance: Role of Demographic Information. *IEEE Trans. Inf. Forensics Secur.* 7, 6 (2012), 1789–1801. https://doi.org/10.1109/TIFS.2012.2214212

[37] Ron Kohavi. 1996. Scaling Up the Accuracy of Naive-Bayes Classifiers: A Decision-Tree Hybrid. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA*. AAAI Press, Menlo Park, CA, 202–207. http://www.aaai.org/Library/KDD/1996/kdd96-033.php

[38] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. 2017. Adversarial examples in the physical world. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net, [C/OL, 2017-2-11], 1–14. https://openreview.net/forum?id=HJGU3Rodl

[39] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *NATURE* 521, 7553 (2015), 436–444. https://doi.org/10.1038/nature14539

[40] Jinfeng Li, Tianyu Du, Shouling Ji, Rong Zhang, Quan Lu, Min Yang, and Ting Wang. 2020. TextShield: Robust Text Classification Based on Multimodal Embedding and Neural Machine Translation. In *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020*. USENIX Association, 1381–1398. https://www.usenix.org/conference/usenixsecurity20/presentation/li-jinfeng

[41] Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2019. TextBugger: Generating Adversarial Text Against Real-world Applications. In *26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019*. The Internet Society. https://www.ndss-symposium.org/ndss-paper/textbugger-generating-adversarial-text-against-real-world-applications/

[42] Yuanchun Li, Jiayi Hua, Haoyu Wang, Chunyang Chen, and Yunxin Liu. 2021. DeepPayload: Black-box Backdoor Attack on Deep Learning Models through Neural Payload Injection. In *43rd IEEE/ACM International Conference on Software Engineering, ICSE 2021, Madrid, Spain, 22-30 May 2021*. IEEE, Piscataway, NJ, 263–274. https://doi.org/10.1109/ICSE43902.2021.00035

[43] Yingqi Liu, Wen-Chuan Lee, Guanhong Tao, Shiqing Ma, Yousra Aafer, and Xiangyu Zhang. 2019. ABS: Scanning Neural Networks for Back-doors by Artificial Brain Stimulation. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019*. ACM, New York, NY, 1265–1282. https://doi.org/10.1145/3319535.3363216

[44] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2015. Deep Learning Face Attributes in the Wild. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*. IEEE Computer Society, Computer Society Los Alamitos, CA, 3730–3738. https://doi.org/10.1109/ICCV.2015.425

[45] Feng Mai, Shaonan Tian, Chihoon Lee, and Ling Ma. 2019. Deep learning models for bankruptcy prediction using textual disclosures. *Eur. J. Oper. Res.* 274, 2 (2019), 743–758. https://doi.org/10.1016/j.ejor.2018.10.024

[46] Hayden Melton. 2018. On fairness in continuous electronic markets. In *Proceedings of the International Workshop on Software Fairness, FairWare@ICSE 2018, Gothenburg, Sweden, May 29, 2018*, Yuriy Brun, Brittany Johnson, and Alexandra Meliou (Eds.). ACM, New York, NY, 29–31. https://doi.org/10.1145/3194770.3194771

[47] Linghan Meng, Yanhui Li, Lin Chen, Zhi Wang, Di Wu, Yuming Zhou, and Baowen Xu. 2021. Measuring Discrimination to Boost Comparative Testing for Multiple Deep Learning Models. In *43rd IEEE/ACM International Conference on Software Engineering, ICSE 2021, Madrid, Spain, 22-30 May 2021*. IEEE, Piscataway, NJ, 385–396. https://doi.org/10.1109/ICSE43902.2021.00045

[48] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. 2019. DeepXplore: automated whitebox testing of deep learning systems. *Commun. ACM* 62, 11 (2019), 137–145. https://doi.org/10.1145/3361566

[49] Qusai Ramadan, Amir Shayan Ahmadian, Daniel Strüber, Jan Jürjens, and Steffen Staab. 2018. Model-based discrimination analysis: a position paper. In *Proceedings of the International Workshop on Software Fairness, FairWare@ICSE 2018, Gothenburg, Sweden, May 29, 2018*, Yuriy Brun, Brittany Johnson, and Alexandra Meliou (Eds.). ACM, New York, NY, 22–28. https://doi.org/10.1145/3194770.3194775

[50] Vikram V. Ramaswamy, Sunnie S. Y. Kim, and Olga Russakovsky. 2021. Fair Attribute Classification Through Latent Space De-Biasing. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*. Computer Vision Foundation / IEEE, 9301–9310. https://openaccess.thecvf.com/CVPR2021?day=2021-06-23

[51] Ali Sedaghatbaf, Mahshid Helali Moghadam, and Mehrdad Saadatmand. 2021. Automated Performance Testing Based on Active Deep Learning. In *2nd IEEE/ACM International Conference on Automation of Software Test, AST@ICSE 2021, Madrid, Spain, May 20-21, 2021*. IEEE, Piscataway, NJ, 11–19. https://doi.org/10.1109/AST52587.2021.00010

[52] Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015*. Arxiv, [C/OL, 2015-4-10], 1–14. http://arxiv.org/abs/1409.1556

[53] Pedro Tabacof and Eduardo Valle. 2016. Exploring the space of adversarial images. In *2016 International Joint Conference on Neural Networks, IJCNN 2016, Vancouver, BC, Canada, July 24-29, 2016*, Vol. 2016-October. IEEE, Piscataway, NJ, 426–433. https://doi.org/10.1109/IJCNN.2016.7727230

[54] Florian Tramèr, Vaggelis Atlidakis, Roxana Geambasu, Daniel J. 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 2017, Paris, France, April 26-28, 2017*. IEEE, Piscataway, NJ, 401–416. https://doi.org/10.1109/EuroSP.2017.29

[55] 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, Montpellier, France, September 3-7, 2018*. ACM, New York, NY, 98–108. https://doi.org/10.1145/3238147.3238165

[56] Sahil Verma and Julia Rubin. 2018. Fairness definitions explained. In *Proceedings of the International Workshop on Software Fairness, FairWare@ICSE 2018, Gothenburg, Sweden, May 29, 2018*. ACM, New York, NY, 1–7. https://doi.org/10.1145/3194770.3194776

[57] 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 *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*. IEEE, Piscataway, NJ, 8916–8925. https://doi.org/10.1109/CVPR42600.2020.00894

[58] Ruyue Xin, Jiang Zhang, and Yitong Shao. 2020. Complex network classification with convolutional neural network. *Tsinghua Science and Technology* 25, 4 (2020), 447–457.

[59] Jie M. Zhang and Mark Harman. 2021. "Ignorance and Prejudice" in Software Fairness. In *43rd IEEE/ACM International Conference on Software Engineering, ICSE 2021, Madrid, Spain, 22-30 May 2021*. IEEE, Piscataway, NJ, 1436–1447. https://doi.org/10.1109/ICSE43902.2021.00129

[60] Lingfeng Zhang, Yueling Zhang, and Min Zhang. 2021. Efficient white-box fairness testing through gradient search. In *ISSTA '21: 30th ACM SIGSOFT International Symposium on Software Testing and Analysis, Virtual Event, Denmark, July 11-17, 2021*, Cristian Cadar and Xiangyu Zhang (Eds.). ACM, New York, NY, 103–114. https://doi.org/10.1145/3460319.3464820

[61] 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 *ICSE '20: 42nd International Conference on Software Engineering, Seoul, South Korea, 27 June - 19 July, 2020*. ACM, New York, NY, 949–960. https://doi.org/10.1145/3377811.3380331