1-2024

Towards explainable neural network fairness

Mengdi ZHANG
Singapore Management University, mdzhang.2019@phdcs.smu.edu.sg

Towards Explainable Neural Network Fairness

Mengdi Zhang

SINGAPORE MANAGEMENT UNIVERSITY

2023

Towards Explainable Neural Network Fairness

Mengdi Zhang

Submitted to School of Computing and Information Systems
in partial fulfillment of the requirements for
the Degree of Doctor of Philosophy in Computer Science

SINGAPORE MANAGEMENT UNIVERSITY
2023

I hereby declare that this dissertation is my original work and it has been written by me in its entirety.
I have duly acknowledged all the sources of information which have been used in this dissertation.

This dissertation has also not been submitted for any degree in any university previously.

Zhang Mengdi

Mengdi Zhang
January 5, 2024

Toward Explainable Neural Network Fairness

Mengdi Zhang

Abstract

Neural networks are widely applied in solving many real-world problems. At the same time, they are shown to be vulnerable to attacks, difficult to debug, non-transparent and subject to fairness issues. Discrimination has been observed in various machine learning models, including Large Language Models (LLMs), which calls for systematic fairness evaluation (i.e., testing, verification or even certification) before their deployment in ethic-relevant domains. If a model is found to be discriminating, we must apply systematic measure to improve its fairness. In the literature, multiple categories of fairness improving methods have been discussed, including pre-processing, in-processing and post-processing. In this dissertation, we aim to develop methods which identity fairness issues in neural networks and mitigate discrimination in a systematic explainable way.

To achieving this goal, we start with developing a method of explaining how a neural network makes decisions based on simple rules. One factor contributing to fairness concerns is the inherent black-box nature of neural networks. This makes it challenging to discern the rationale behind specific decisions, potentially resulting in biased outcomes. So, in the first work, we focus on explaining neural networks using rules which are not only accurate but also provide insights into the underlying decision-making process. We provide two measurements for neural network decision explainability, and develop automated evaluation algorithms.

In the second research work, we apply the rule-based idea to identify fairness issues that can be explained. We notice that group discrimination is mostly

hidden and less studied. Therefore, we propose TESTSGD, an interpretable testing approach which systematically identifies and measures hidden group discrimination of a neural network characterized by an interpretable rule set which indicates conditions over combinations of the sensitive features. Specifically, given a neural network, TESTSGD first automatically generates a rule set and then provides an estimated group fairness score to measure the degree of the identifies subtle group discrimination with theoretical error bounds.

In the third research work, we design an approach which explore the causes of fairness issue and mitigate them systematically. Specifically, we first apply and empirical study which shows that existing fairness improvement methods are not always effective (e.g., they may improve fairness by paying the price of huge accuracy drop) or even not helpful (e.g., they may even worsen both fairness and accuracy). Then, we propose an approach which adaptively chooses the fairness improving method based on causality analysis. That is, we choose the method based on how the neurons and attributes responsible for unfairness are distributed among the input attributes and the hidden neurons.

Lastly, we present a method which would allow us to extend our fairness mitigation approach to Large Language Models. As existing bias mitigation research typically do not apply in the era of LLMs, we propose a non-intrusive bias mitigation approach which does not require accessing or modifying the internals of LLMs. Specifically, we propose a parameter-efficient debias adapter that not only improves fairness systematically but also provides a theoretical statistical guarantee on the achieved fairness whenever feasible during debiasing.

Contents

1 Introduction **1**

1.1 Motivation 1

1.2 Problem Definition 5

1.3 Research Contributions 6

1.4 Dissertation Structure 8

2 Background **11**

2.1 Neural Networks 11

2.2 Interpretability Definitions and Testing 13

2.3 Fairness Definitions 14

2.4 Fairness Testing and Verification 16

2.5 Fairness Improvement and Bias Mitigation 18

3 Interpretability Testing **20**

3.1 Introduction 20

3.2 Problem Definition 22

3.2.1 What is human-understandable? 23

3.2.2 Learning Decision Trees 24

3.2.3 Defining Measurements 27

3.3 Testing Decision Explainability 29

3.3.1 K-Explainability Testing against Training Set 29

3.3.2 K-Explainability Testing against All Inputs 31

3.3.3 K, ϕ-Explainability Testing 34

3.4 Implementation and Evaluation 37
3.5 Related Work 48
3.6 Summary 50

4 Subtle Group Discrimination Identification 51

4.1 Introduction 51
4.2 Background 55
4.2.1 Input Type 55
4.2.2 Group Fairness Definitions 56
4.2.3 Problem Definition 57
4.3 Methodology 58
4.3.1 Generating Frequent Rule Sets 58
4.3.2 Identifying Group Fairness 62
4.3.3 Input Sampling 67
4.4 Implementation and Evaluation 68
4.5 Related Work 78
4.6 Summary 80

5 Adaptive Fairness improvement 81

5.1 Introduction 81
5.2 Background 83
5.2.1 Fairness Definitions 83
5.2.2 Fairness Improving Methods 86
5.3 An Empirical Study 88
5.3.1 Experimental Setup 88
5.3.2 Evaluation Results 90
5.4 An Adaptive Approach 96
5.4.1 Causality Analysis 96
5.4.2 Adaptive Fairness Improvement 99
5.5 Implementation and Evaluation 103

5.6 Related Work 108
5.7 Summary 110

6 Non-Intrusive Bias Mitigation 111

6.1 Introduction 111
6.2 Background 114
6.2.1 Textual Data 114
6.2.2 Bias Definition 115
6.2.3 Problem Definition 118
6.3 Methodology 118
6.3.1 Overview 118
6.3.2 Mitigating Bias with Statistical Confidence 120
6.3.3 Mitigating Group bias 124
6.3.4 Mitigating Individual bias 125
6.4 Experiment Setup 128
6.5 Implementation and Evaluation 131
6.6 Related Work 138
6.7 Summary 141

7 Conclusion and Future Work 142

7.1 Conclusions 142
7.2 Future Work 144

List of Figures

3.1 Human-understandable models 24
3.2 A sample decision tree on the Bank dataset 30

4.1 An Overview of TESTSGD. 52

5.1 Group Fairness Improvement of Models with respect to Different Protected Attributes 90
5.2 Accuracy Changes of Models with respect to Different Protected Attributes After Processing 91
5.3 Comparison between group fairness improvement and individual discrimination reduction 94
5.4 Causality analysis result of *Adult-gender* 102
5.5 Our Approach vs SOTA on Group Fairness 107
5.6 Our Approach vs SOTA on Individual Discrimination 109

6.1 Overview of the Proposed Debias Strategy 119
6.2 Overview of Debias Adapter 120
6.3 Group Bias 'δ_g' Changes and Accuracy Changes 132
6.4 Individual Bias 'δ_i' and Accuracy Changes 134
6.5 'δ_i' Changes with Increasing λ 136
6.6 'Improved δ_i - Acc Drop' Changes with Increasing λ 136
6.7 'Improved δ_g - Acc Drop' Changes 138
6.8 'Improved δ_i - Acc Drop' Changes 139

List of Tables

3.1 Dataset and Models of Experiments 39
3.2 Parameters of the experiments 39
3.3 K-explainability testing results 40
3.4 Training time with different #instances 41
3.5 K,ϕ-explainability testing result 42
3.6 K,ϕ-explainability testing result based on Shapley values 42
3.7 Results after robust training 46
3.8 Results after training with 'counterexamples' 47

4.1 Identity Sensitive Terms 59
4.2 Parameters of the Experiments 70
4.3 Dataset and Models of Experiments 70
4.4 Rule Sets and Fairness Scores for Neural Networks 71
4.5 Time Taken to Identify the subtle discrimination 73
4.6 Effect of Different θ 74
4.7 Discrimination Mitigation for Neural Networks 75
4.8 Comparisons Between TESTSGD , Themis and FairFictPlay. '-
' means timeout. 77

5.1 Dataset Privileged Groups Definition 89
5.2 Neural Networks in Experiments 90
5.3 Best Method for Group Fairness Improvement 92
5.4 Distribution of high causal effects with Group Fairness 104
5.5 Distribution of high causal effects with Individual Discrimination 105
5.6 Time overhead for causality analysis 108

6.1 Metrics of Baseline BERT Models w.r.t. Overall Performance
and Individual Identity Categories 129
6.2 Metrics after Debiasing with Regards to Group Bias 130
6.3 Debiasing with Regards to Individual Bias 133
6.4 Group Bias Degree Changes After Debiasing by Related Works
and Ours 138
6.5 Individual Bias Degree Changes After Debiasing by Related
Works and Ours 139

Acknowledgement

First of all, I would like to express my deepest appreciation to my supervisor, Prof. Jun Sun, for his unwavering support, invaluable guidance, and endless patience throughout my PhD journey. His expertise, insightful feedback and dedication to my academic growth have been instrumental in shaping my research works. I would also like to express my deepest gratitude to Prof. Jingyi Wang from Zhejiang University, for his generous sharing of knowledge, constructive feedback and invaluable advice. He is also the guiding mentor when I first entered as a PhD student, providing me with a great deal of patient assistance and guidance especially in the early stages. I'm also thankful to Prof. David Lo and Prof. Chris Poskitt from my dissertation committee, for their generous suggestions during PhD Qualifying Examination and Dissertation Proposal.

I extend my heartfelt thanks to the members of our research group. They have provided a nurturing and intellectually stimulating environment. I am grateful for their willingness to share their knowledge and experience, for always being available for offer insightful advice and cheerful friendships. Especially I would like to thanks the seniors from our research group, who have taught me how to try my best to make work life balance during my tough PhD life.

I would also like to thank the Singapore Management University who offered me a great opportunity to start my studies. I would like to say a special thank to administrative staff from PGR office, including BOO Chui Ngoh, Seow Pei Huan, Caroline TAN, ONG Chew Hong, for their efforts.

Finally, I would like to express my heartful thanks to my family, who always by my side throughout the darks and the brights. I'm especially grateful to my boyfrind, Jipeng Fei, for his love, unconditional support and encouragement in the past four years.

Chapter 1

Introduction

1.1 Motivation

Neural networks have found their way into a variety of systems, including many which potentially have significant societal impact, such as personal credit rating [42], criminal sentencing [130, 10], face recognition [141] and resume shortlisting [129]. While these neural networks often have high accuracy, multiple issues have been identified as well. For instance, it has been shown that neural networks are subject to adversarial attacks, i.e., a correctly classified sample would be easily misclassified once some perturbation is applied [55]. For another instance, a neural network model may be embedded with a backdoor to predict differently in the presence of certain trigger [181] or may be discriminative against certain groups [107]. Actually, more and more attention has been paid to the fairness issues of these machine learning models [153, 9, 51, 176, 177, 130, 131, 23, 47] as discrimination has been discovered in many applications [52, 130, 150, 165]. That is, the predictions made by these neural networks may be biased with regard to certain protected attributes such as race, gender and religion. For instance, it has been shown [14] that a neural network trained to predict people's income level based on an individual's personal information (which can be used in applications such as bank loan approval) is much more likely to predict male individuals with high-income level. Further analysis shows that for many individuals, changing only the gender or race causes the output of the predictions to flip [47]. For another instance, it has been shown [10] that a machine learning model used to predict the recidivism risk for suspected criminals is more likely to mislabel black defendants as having a high recidivism risk.

To minimize such ethical risks and achieve trust in neural networks, it is desirable to develop approaches and software toolkits to systematically identify and mitigate discrimination. In this thesis, we start with developing a method for explaining AI decisions, which lays the foundation for subsequently research on detecting, explaining and mitigating discrimination. In the context of fairness, interpretability becomes an essential tool for scrutinizing the potential biases within a neural network. By gaining insights into the features and patterns that the model relies on, we can identify any hidden biases that may have been learned from the data. Additionally, a more interpretable model not only facilitates fairness but also enhances the overall transparancy of the system, ultimately benefiting broader domains.

Unlike traditional software, AI models, especially complex neural networks, work like a black box. Traditional software operates through predefined rules and instructions provided by human programmers, which can be easily inspected and understood. In contrast, neural networks learn patterns and relations from vast amounts of data. This inherent opacity due to the intricate web of interconnected nodes and weights makes it challenging for humans to understand how a specific conclusion is reached or why a wrong prediction arises. This lack of transparency raises important concerns regarding accountability, as well as ethical and legal considerations, particularly in critical applications like healthcare, finance, and autonomous systems. For example, when a neural network is tasked with determining the approval or rejection of a bank loan application, it becomes problematic if we are unable to explain the specific reasons behind the rejection of one application while approving a similar one. In such instances, stakeholders are keen to minimize the occurrence of inexplicable decisions. Particularly in the context of fairness, when a model is found to make its decisions solely based on sensitive features or protected patterns, it can lead to serious ethical dilemmas. This concern emphasizes the need for interpretability techniques to bridge the gap between the internal workings of these models and human comprehension, ensuring that their outputs are reliable and trustworthy. To address these concerns, some significant regulatory frameworks such as the General Data Protection Regulation (GDPR), Algorithmic Accountability Act (US AAA), and the Artificial Intelligence Act (EU AIA) have been proposed to address concerns related to data privacy, algorithmic accountability and the ethical use of artificial intelligence.

In recent years, researchers have proposed multiple approaches to provide hints on how neural networks work internally, such as local/global interpretability analysis. Studies on local interpretability focus on investigating neural network prediction on one or a specific set of samples [166, 146, 35, 179]. For instance, they aim to provide hints on how a neural network makes certain decisions through highlighting certain pixels of the input picture of certain words of an input text. Studies on the global interpretability of neural networks aim to come up with models that are simple enough to be human-understandable and yet expressive enough to predict the predictions of the neural network (at least in most cases). Such candidates include regression models [133], decision trees [93], decision rules [87].

When delving into the domain of fairness testing, multiple efforts have also been made in the testing community to first search for (and then guide the mitigating of) discrimination of machine learning models spanning from traditional ones to neural networks [176, 177, 150, 165, 47]. Typically, discrimination can be classified into two categories, e.g., individual discrimination, i.e., identifying or generating individual discriminatory instances of a machine learning model [176, 177, 150, 165, 47] and group discrimination, i.e., characterizing a model's discrimination against a certain group [47, 165, 148, 81]. For specific cases of discrimination, group discrimination has been less analyzed so far, as it imposes new challenges. First, it is highly non-trivial to effectively enumerate all combinations of sensitive features (especially when the sensitive features have multiple or even continuous values). Second, group discrimination can be hidden, i.e., there might be 'subtle' group discrimination. Additionally, existing fairness testing approaches seldom offer interpretable indications of discriminations.

Accordingly, many methods have been proposed to improve the fairness of neural networks [71, 44, 27, 167]. Typically, they can be categorized into three groups according to when they are applied, i.e., pre-processing, in-processing and post-processing. For instance, the pre-processing method Reweighing assigns different weights to training samples in order to reduce the effect of data biases [71]. In-processing methods modify the original model to reduce the discrimination [29, 168, 73, 5, 6, 76]. Finally, post-processing methods modify the prediction results instead of the inputs or the internals of the model [61, 123, 72]. In our work, we recognize that existing fairness improvement methods are not always effective in mitigating discriminations and, in some cases, may even yield adverse effects.

Then, with the emergence of transformer-based models, like GPT and BERT, the evolution of Large Language Models (LLM) has taken impressive strides. Meanwhile, some potential problems such as textual biases have been concerned [15, 138, 2] as well. Various existing studies have demonstrated that biases are present in different parts of LLMs, including word-level representation [102, 96, 26, 134], sentence encoders [105] and downstream tasks [138, 2]. Various works on mitigating bias in language models have been proposed. Common approaches include modifying the training data through data augmentation [84, 180], manipulating word embeddings or sentence representation [49, 120, 106, 158], causality-based bias repair [144] and other in-processing methods [63, 91]. However, as training LLMs is increasingly beyond the capacity of ordinary users or small business owners, many of these methods become impractical. Most in-processing methods relying on model retraining or fine-tuning become too costly. Even pre-processing methods such as Counterfactual Data Augmentation [96, 68] still need an additional phase of pre-training. Moreover, due to the latent nature of biases embedded and propagated through complex connections in language models, they often appear in various ways. It is thus challenging to produce a LLM-based production that avoids biases (e.g., according to the regulatory requirements) when we only have API access to the LLMs.

1.2 Problem Definition

The above-mentioned research has sparked serious concerns regarding fairness issues in machine learning models, especially neural networks. Furthermore, existing approaches regarding to fairness testing and bias mitigation have certain limitations, especially when dealing with more complex and opaque models. This motivates us to propose more comprehensive theories and develop more comprehensive approaches to test fairness and mitigate discriminations. The principal problem is defined as follows.

Problem statement: Given a machine learning model, we aim to develop explainable tools to systematically detect the discriminations within a model's decision making process and mitigate such biases.

We mainly address this problem in three aspects: *Interpretability Testing, Fairness Testing and Bias Mitigation.*

- Interpretability Testing: aims to evaluate and assess the degree to which a model's decisions can be understood and explained by humans. It aims to provide insights into how a model make arrives at its predictions, making the inner workings of the model more transparent and interpretable. This is crucial for gaining trust, ensuring accountability and most importantly in this dissertation, providing insights on unfair decisions.

- Fairness Testing: aims to evaluate the model's behavior to ensure the model treats different groups or different individuals fairly and does not privilege any particular group or individual based on sensitive attributes, (also known as protected attributes), such as gender, race, religion, or age. Furthermore, we aim to generate testing results with certain statistical confidence, e.g., the change of reporting non-existing discrimination is low. It is also essential for building AI systems that align with ethical regulations.
- Bias Mitigation: aims to reduce or eliminate discriminatory outcomes that may arise from the model's decisions. It aims to promote equity in the deployment of machine learning models, particularly in contexts where decisions have significant societal or ethical implications. The goal of bias mitigation is to enhance the model's fairness, while minimizing the potential effects on model's overall performance.

1.3 Research Contributions

To effectively address the aforementioned problem, this dissertation presents several contributions. First, we introduce multiple tools encompassing interpretability testing, discrimination detection, automatic fairness enhancement for tabular data, and bias mitigation for textual data. Second, we develop open source self-contained implementations of these approaches. Finally, we evaluate the effectiveness of each tool through a series of comprehensive experiments, providing detailed and insightful results. The major contributions are summarized as follows.

We first propose multiple measurements on how easy it is to explain a user-provided neural network's decisions. Intuitively, we define the measurements based on how often the prediction results can be explained consistently using a human-understandable model and how simple the model is. These measurements allow us to compare different neural networks in terms of how explainable their decisions are. Afterward, we develop multiple algorithms that allow us to automatically evaluate and compare neural networks based on the measurements. We remark that our approach can be regarded as a generalization of the recent work on evaluating fairness [7, 176] (where fairness is defined in terms of individual discrimination). This enables us not only to gain valuable insights into potential fairness concerns but also to identify instances where model decisions are highly related to sensitive features, prompting a closer examination of fairness considerations.

We then propose tool TESTSGD, an effective method to systematically test a given machine learning model against such hidden Subtle Group Discrimination. It consists of three main phases: 1) candidate rule set generation, 2) group fairness identification, and 3) discrimination mitigation. In the first phase, TESTSGD will automatically generate a candidate set of rules concerning multiple sensitive features. In the second phase, the rule set effectively partitions the samples into two groups. The key intuition behind is to develop effective criteria to automatically mine interpretable rules that are practical and relevant in real-world applications. Then we measure if the model suffers from group discrimination (against the groups partitioned by the rule set) by measuring the group fairness score.

We then focus on fairness improvement and propose an approach to choose the 'right' fairness improving method based on causality analysis. Formally, we use causality analysis to evaluate the causes of unfairness and then use the probability of high causal effects and Coefficient of Variation to characterize the distribution of the causal effects. Depending on the result of the causality analysis, we then choose the fairness improving method accordingly. Our approach is designed based on the results of an empirical study which evaluates 9 fairness improving methods (i.e., 2 pre-processing methods, 4 in-processing methods and 3 post-processing methods) on 4 different benchmark datasets with respect to different fairness metrics. Our approach is systematically evaluated with the same models. The results show that our selected processing approach is the optimal choice to improve group fairness in all cases and the optimal choice to reduce individual discrimination in most cases.

We propose a novel bias mitigation approach with regards to textual data, which treats the language model as a black box. That is, we do not access or modify the internals of a given model. Instead, we apply a debias adapter based on the sequence representation output from language models, which aims to mitigate bias in a parameter-efficient manner. In addition, our approach provides statistical confidence in the achieved fairness during debias training, which is essential for many applications that need to satisfy regulatory requirements on fairness. Compared to bias mitigation through fine-tuning the whole model, our approach is much more efficient. To evaluate our approach, we consider biases based on a person's identity and support both group bias mitigation and individual bias mitigation. The results show that our approach is effective in mitigating both the two biases across all models with negligible accuracy drops.

1.4 Dissertation Structure

This dissertation consists of four works. The first two works are on testing neural networks regarding to interpretability and fairness [174, 175]. These two works are joint works with Jun Sun, Jingyi Wang and Bing Sun. The remaining two works focus on fairness improvement and bias mitigation [172, 173], which are joint works with Jun Sun. Each work is presented in one chapter from Chapter 3 to Chapter 6 in a self-contained manner, i.e., across different chapters, there may be some redundant content, particularly in the background and related work discussions. The detailed outline of this thesis is as follows.

Chapter 2 discusses the background for comprehending our works. It presents the fundamental knowledge of machine learning models, especially neural networks, basic definitions of fairness, interpretability and existing works of fairness testing, improvement as well as their limitations.

Chapter 3 presents an approach to evaluate and measure neural networks interpretability. This chapter is based on the paper:

* Zhang, M., Sun, J., & Wang, J. (2022). Which neural network makes more explainable decisions? An approach towards measuring explainability. *Automated Software Engineering, 29(2), 39.*

Chapter 4 presents a toolkit named TESTSGD which automatically systematically tests a given machine learning model against such hidden subtle group discrimination. This chapter is based on the paper:

* Zhang, M., Sun, J., Wang, J., & Sun, B. (2023). TestSGD: Interpretable Testing of Neural Networks Against Subtle Group Discrimination. *ACM Transactions on Software Engineering and Methodology, 32(6): 1-24.*

Chapter 5 presents an approach to choose the 'right' fairness improvement methods adaptively based on causality analysis. This chapter is based on the paper:

* Zhang, M., & Sun, J. (2022). Adaptive fairness improvement based on causality analysis. *In Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (pp. 6-17).*

Chapter 6 presents an efficient approach to mitigate bias for Large Language Models with statistical confidence. This chapter is based on the paper:

* Zhang, M., & Sun, J. Non-intrusive Mitigation of Biases in Language Models with Statistical Confidence. *(Under Review)*

Chapter 7 summarizes the dissertation and outlines some potential future works.

Chapter 2

Background

In this chapter, we provide the background knowledge necessary to understand the works in this dissertation. We briefly introduce the models we consider, including neural networks and Large Language Models (LLMs), interpretability, fairness as well as some notable existing research in this area and their limitation.

2.1 Neural Networks

Neural networks, also known as artificial neural networks are one kind of machine learning models and are at the heart of deep learning algorithms. it is inspired by the structure of the human brain, and aim to mimic the behavior of biological neurons. They consist of layers of interconnected nodes, each emulating a neuron, forming a complex network capable of learning intricate patterns within data. Through training, neural networks adapt their internal parameters to minimize the difference between the predicted outcomes and the ground-truth labels. This makes them capable of learning complicated relationships for solving tasks such as image recognition, autonomous driving, natural language processing and generating winning gaming strategy.

As the demand for natural language processing grows, LLMs are also gradually emerging, especially after the transformer-based models are developed.

The transformer architecture is the fundamental building block of all LLMs. Its core idea is the “attention machanism”, which lets the model focus on different part of the input sequence, enabling it to capture long-range dependencies effectively [152]. LLMs are considered a specialized form of neural network for text processing tasks. They have revolutionized the field of Natural Language Processing by enabling machines to understand and generate human-like language patterns. This makes them invaluable for applications like chatbots, content generation, and various forms of automated text processing. For example, OpenAI’s GPT models have demonstrated remarkable capability in tasks such as text completion, translation, and even creative writing. Although these LLMs vary widely in size, they all have high demands for space and computational resources. For example, the original BERT models encompass 110 million parameters, while larger versions like BERT-large further escalate this figure to 340 million. The GPT models, developed by OpenAI, takes even more space. GPT-2 has 1.5 billion parameters and GPT-3 has a staggering 175 billion parameters.

Meanwhile, researchers ahve identified multiple safety/security issues regarding neural networks. For example, Neural networks have been found to be vulnerable to small, carefully crafted perturbations in input data, known as adversarial attacks [83]. For another example, as neural network is trained on a large volume of data, the quality of training data can effect the model predictions and can pose privacy risks as the training data may contain some personality information. Moreover, understanding how neural networks arrive at their decisions can be challenging, especially for complex models like deep neural networks.

2.2 Interpretability Definitions and Testing

In this section, we introduce the background related to interpretability and its evaluation as well as a brief summary of existing research on evaluating interpretability, providing a fundation of key concepts. The term interpretability refers to a the ability to understand and explain the decisions made by a model. Various definitions of interpretability have been proposed in the literature [111, 108, 80]. These definitions can be broadly classified into two cateogries, i.e., local interpretability and global interpretability. Local interpretability focus on investigating neural network prediction on one or a specific set of samples. It explores and measures certain metrics such as feature importance scores. Lundberg *et al.* proposed an approach named SHAP (i.e., Shapley Addictive exPlanations) which assigns each feature an importance value and explains a particular prediction based on the weighted sum of features [98]. In terms of image classification, some methods try to explain predictions by exploring the contribution to the prediction score at pixel level [98] or the object level [179]. However, while these local interpretations provide explanations for specific inputs, they are less effective in offering a comprehensive understanding of the neural network's behavior as a whole.

Studies on global interpretability aim to come up with models that are simple enough to be human-understandable and yet expressive enough to mimic the behaviors of neural networks (at least in most cases). The simple model can be regression models, rule-based models and decision trees. For example, Lakkaraju *et al.* proposed interpretable decision sets working as a high-accurate prediction model and interpretable in the same time [87]. While these approaches share the same idea of using simple models to mimic neural networks, they do not provide ways of testing and measuring the degree of decision explainability.

2.3 Fairness Definitions

For fairness evaluation, existing works provided various definitions and measurements with respect to tabular data [41, 25, 44, 61, 84, 47, 76], textual data [26, 105] and image data [23, 127, 48, 139].

With respect to tabular data, existing definitions can be broadly classified into two categories, i.e., individual fairness and group fairness. Note that, in this dissertation, we only consider classification tasks on tabular data and we leave the studies on other forms of data (such as texts and images) for future work. For individual fairness, the definition of Individual Discriminatory Instance (IDI) is widely adopted. A sample is considered as an IDI if and only if its label changes once its protected attribute(s) changes. This concept is widely used in fairness testing [176, 150], such as works on searching or generating idi which can be used to retrain the model so as to improve the individual fairness. The measurement based on individual fairness typically involves calculating the percentage of idi within a set, such as a test set. This metric is formally referred to as the Causal Discrimination Score [47]. Another common definition is counterfactual fairness, which is also an individual-level definition [84]. Counterfactual fairness defines that a model achieves fairness if its predictions remain consistent when applied to the actual group and a distinct demographic group generated by introducing variations only to sensitive attributes. When evaluating model's individual fairness, it requires abundant validation cases. In reality, it is difficult to find cases that happen to share the same features but differ in the sensitive attributes. Thus, the notion of counterfactual fairness is commonly used to generate test cases for fairness evaluation [59, 99, 12] In terms of group fairness, which is also known as statistical fairness, it focuses on sensitive groups such as ethic minority. This definition of group fairness commonly divides the whole input space into two groups, i.e., privileged group and unprivileged group. Typically, the privileged group refers to populations with favored sensitive attributes, such as 'gender=male' or 'race=white'. The unprivileged group refers to the rest. There are multiple metrics for quantifying the level of group fairness. One basic definition is Statistical Parity Difference [25]. It is based on positive classification rate and measure the difference in the probability of positive outcomes between privileged group and unprivileged group. Another kind of group fairness definitions are based on true positive rate. For example, the definition Equality of Opportunity requires that the true positive rate is the same for all groups defined by sensitive attributes [70]. It ensures the model is equally accurate at predicting positive label for all demographic groups, regardless of the sensitive attributes. Another example is Equalized Odds, which extend the previous concept and requires the probability of correct prediction is the same [43]. In other words, the false positive rates and false negative rates should be equal across different groups.

With respect to textual data, existing definitions can be broadly divided into two popular categories, i.e., intrinsic bias and extrinsic bias. Intrinsic bias refers to bias inherent in the representations, e.g., word embeddings [96, 63] and sentence representation [105, 113]. For instance, the Word Embedding Association Test (WEAT) measures the association between two sets of target concepts and two sets of attributes [26]. The Sentence Encoder Association Test (SEAT) extends WEAT to sentence contexts [105]. Extrinsic bias refers to bias in downstream tasks, e.g., provide some concrete examples of such tasks. So the group fairness measurements for tabular data can also been used in a similar way to measure the extrinsic bias for textual data, e.g., disparity in false positive rate and equality of opportunity [142, 91, 68]. Due to the complexity of LLMs, measuring intrinsic bias is not necessary to reflect bias in downstream tasks [53].

Moreover, the notion of fairness in machine learning is a multifaceted concept that encompasses various definitions and assumptions. Fairness through Unawareness and Fairness through Awareness represent two fundamental concept in this context. Fairness through Awareness discusses conditions when model acknowledges the presence of sensitive attributes and actively incorporates them into the training and prediction process. Fairness through Unawareness involves intentionally omitting sensitive attributes, aiming to prevent explicit discrimination. Dwork et al. defined an algorithm be fair as long as any protected attributes are not used in the decision-making process [41]. However, a notable limitation of this definition is that non-sensitive attributes may still contain discriminatory information that is correlated with sensitive attributes.

2.4 Fairness Testing and Verification

Unlike traditional software systems that are based on logic, the lack of interpretability of neural networks makes system testing difficult. The fairness testing refers to any methodology designed to detect discriminations or bias.

In the context of fairness through Awareness and Unawareness, different techniques are involved. Fairness through awareness acknowledges the presence of sensitive attributes and actively incorporates them into the model prediction. Some techniques like reweighing and retraining with loss functions are used to improve fairness through awareness setting [71, 29, 168]. However, for many real-word applications, it is difficult to obtain sensitive attributes due to various reasons such as privacy or legal restrictions. Fairness through unawareness, in contrary, ignores sensitive attribute during training and prediction process. However, it has been shown that simply removing sensitive features is not effective in mitigating biases. This is because biases can still be present in the data, hidden in correlations between non-sensitive features, and may be implicitly learned by the model [41]. Addressing fairness issues without sensitive attributes is challenging. There are limited researches focus on fairness problems without sensitive attributes. For example, Gupta et al [58] constructed proxy groups using variables that are not sensitive but are likely correlated to sensitive attributes based on prior knowledge. For another example, Lahoti et al. proposed adversarial reweighted learning that leverages the notion of computationally-identifiable errors to achieve Max-Min fairness without sensitive attributes [86]. Yan et al. proposed pre-processing approach to cluster the data and used obtained groups as the proxy [161].

Moreover, during the process of machine learning model development, especially for neural networks, biases can introduce and propagate at multiple stages. First, neural networks are trained using a large-scale dataset which might contains biased data. This can occur due to the imbalances data, and as a result, the model learns and potentially reinforces existing biases present in the training data. Second, in the training stage, biases can be remembered by the complex neuron connections and propagate through the prediction process. The fairness testing can be applied from different aspects, e.g., data testing [21], model testing [8, 47, 150, 160], framework testing [114, 156] and component testing [97]. In this dissertation, we focus on model fairness testing. It consider the model as the testing component and reveal discrimination based on the prediction behaviors.

Additionally, the fairness testing with regards to model level can be performed in white, gray and black box manners. In the setting of white-box testing, we has full access to the model's internal structure, including its architecture and parameters. This allows for a detailed examination of how the internals of the model contribute to discrimination. For example, the tool ADF utilizes gradient information to generate individual discriminatory instances to test individual fairness [176]. It focuses on discriminatory instances near the decision boundary and use gradient to search for more neighboring idis. In the setting of gre-box testing, wehas particular knowledge of the model's internals. Finally, black box testing involves evaluating the model without any knowledge of its internal working. Testers can only access to the model's inputs and outputs. This level of testing is particularly relevant in real-world scenarios where access to a model's internal architecture may not be feasible. For example, Tramer *et al.* proposed an unwarranted associations framework to detect unfair, discriminatory or offensive user treatment in data-driven applications [148] in a black-box manner. Each of these testing offers unique insights into the fairness of neural networks.

2.5 Fairness Improvement and Bias Mitigation

Once a model is found to be biased, a natural question is how to mitigate such bias. The existing fairness improvement methods can be categorized into three groups according to when they are applied, i.e., pre-processing, in-processing and post-processing.

Pre-processing methods aim to reduce the discrimination and bias in the training data so as to improve the fairness of the trained model. Among the many pre-processing methods [71, 44, 27, 167], the first typical method is reweighing [71]. It works by assigning different weights to training samples in order to reduce the effect of data biases. In particular, lower weights are assigned to favored inputs which have a higher chance of being predicted with the favorable label and higher weights are assigned to deprived inputs. Another typical method is Disparate Impact Remover [44]. It is based on the disparate impact metric which compares the proportion of individuals that are predicted with the favorable label for an unprivileged group and a privileged group. It modifies the values of the non-protected attribute to remove the bias from the training dataset.

In-processing methods modify the model in different ways to mitigate the bias in the model predictions [29, 168, 73, 5, 6, 76]. For instance, Celis *et al.* proposed a classification technique with fairness constraints which develops a meta-algorithm that captures the desired metrics of group fairness using convex fairness constraints (with strong theoretical guarantees) [29]. Then it uses the constraints as an additional loss function for training the neural network. Zhang *et al.* proposed adversarial debiasing method [168]. It modifies the original model by including backward feedback for predicting the protected attribute. It maximizes the predictors' ability for classification while minimizing the adversary's ability to predict the protected attribute to mitigate the bias.

The post-processing methods modify the prediction results instead of the inputs or the model. For instance, Kamiran *et al.* proposed Reject Option Classification method [72]. It assigns favorable labels to unprivileged instances and unfavorable labels to privileged instances around the decision boundary with the highest uncertainty. The Equalized Odds method solves a linear program to find probabilities with which to change the output labels, so as to optimize equalized odds on protected attributes [61].

In terms of bias mitigation with respect to NLP models on textual data, common approaches include modifying the training data through data augmentation, manipulating word or sentence representations and other in-processing methods.

Data Augmentation methods aim to mitigate biases by augmenting original data by some specific techniques, e.g., counterfactual demographics generation or gender swapping [84, 180]. Other existing works focus on word embeddings [49]. For example, Bolukbasi *et al.* mitigated gender bias in Word2Vec [19]. Lastly, typical in-processing methods for bias mitigation include retraining or fine-tuning the given model with loss constraints, and modifying the model structure or training process. For example, Guo *et al.* proposed an automatic method to mitigate the bias for masked language models through fine-tuning with distribution alignment loss on biased prompts [57]. Reinforcement learning is also included to mitigate biases of LMs. Liu *et al.* proposed a reinforcement learning framework for mitigating political biases without having access to the training data or requiring the model to be retrained [91]. However, with the language models become more and more complex and harder to access, more black-box techniques are desired.

Chapter 3

Interpretability Testing

3.1 Introduction

Neural networks are getting increasingly popular thanks to their exceptional performance in solving many real-world problems, such as face recognition [149], language translation [89] and self-driving cars [18]. At the same time, they are shown to be vulnerable to attacks, difficult to debug and subject to fairness issues [55, 181, 107].

In order to understand where fairness issues are coming from, we first need some way of explaining neural networks' decisions. For instance, in scenarios where a neural network is employed to determine the approval or rejection of a bank loan application, it becomes problematic if we cannot provide a clear rationale for why one application from a female applicant is rejected while a similar application from a male applicant is approved. In such cases, stakeholders are keen to minimize instances where contentious decisions cannot be adequately explained. In other words, there is a need to test neural networks in terms of the explainability of their decisions. In this chapter, we present an approach that aim to not only generate simple models that are capable of accurately explaining most of the decisions of a neural networks, but also allow us to compare the explanability of two models. To the best of our knowledge, this is a testing problem which is yet to be addressed.

Since we aim to explain decisions from users' perspectives, the need is closely related yet different from existing studies on neural network interpretability. It is based on recent studies on what are considered human-understandable. These studies are typically conducted through human studies [95, 116, 85]. The conclusion is that comparing different models for explanation, human understanding goes only as far as simple models such as decision trees and linear functions. The consequence is that when we explain a neural network's decisions, we are limited to simple linear functions (e.g., "your loan is rejected before your annual income is below 50K whereas hers is more than 50K") or simple decision trees (e.g., "your loan is rejected because your annual income is below 50K and you have an existing loan, whereas although her annual income is similar, she does not have an existing loan"). Note that this need (of providing a human-understandable explanation of neural network decisions) is different from the need addressed by works which aim to provide interpretation on the inner working of neural networks [79, 87], such as the studies in [137, 164, 56] which provide hints on how a neural network makes certain decision through highlighting certain pixels of the input picture or certain words of an input text.

In this work, we develop an approach and a software toolkit to address the problem. We first propose multiple measurements on how easy it is to explain a user-provided neural network's decisions based on the results of the above-mentioned user studies [95, 94, 116, 85]. Intuitively, we define the measurements based on how often the prediction results can be explained consistently using a human-understandable model and how simple the model is. These measurements allow us to compare different neural networks in terms of how explainable their decisions are. Afterwards, we develop multiple algorithms that allow us to automatically evaluate and compare neural networks in terms of the measurements. Given a neural network, our algorithms systematically explore the input space and measure the percentage of inputs on which the prediction results can be consistently explained with a certain simple model. We remark that our approach can be regarded as a generalization of the recent work on evaluating fairness [7, 176] (where fairness is defined in terms of individual discrimination). That is, a fairness issue is a special case of a decision explainability problem, i.e., the only possible explanation is based on the sensitive features.

Our approach has been implemented as a self-contained toolkit (open-source at [171]). We conduct multiple experiments on neural network models trained based on benchmark datasets to evaluate the relevance of our approach and the effectiveness of our algorithms. Our experiment results show that, unsurprisingly, the neural network models almost always have a low decision explainability. One of the reasons seems to be the existence of adversarial samples. We further perform experiments to check whether models obtained through robust training have improved decision explainability. The results suggest that it is not always the case. Lastly, we check whether we can improve the decision explainability of a model by retraining it using samples generated by our algorithms without reducing accuracy significantly, and the results are affirmative.

This work is based on the published paper [174]. The remaining sections of this chapter are organized as follows. In Section 3.2, we define our problem and provide multiple measurements of the decision explainability of neural networks. In Section 3.3, we present our algorithms for evaluating neural networks based on our measurements. In Section 3.4, we evaluate our approach through multiple experiments. Lastly, we show related work in Section 3.5 and conclude in Section 3.6.

3.2 Problem Definition

In this work, our goal is to develop systematic methods to measure how easy it is to explain the decision of neural networks in a human-understandable way. To define the problem properly, we must answer questions such as: what are considered human-understandable and what quantitative measurements do we use? In the following, we first review existing related literature and then define the problem.

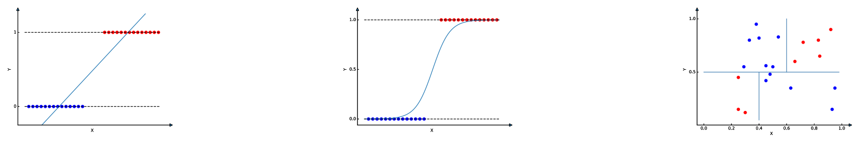
3.2.1 What is human-understandable?

There are multiple human studies on what is considered human-understandable. For example, it is observed that human beings prefer explanations that are both simple and highly probable [95] and often develop simplified understanding of complex systems by ignoring low-probabilistic cases [116]. One study compares the effectiveness of different forms of explanations by measuring human response time [85]. The result shows that simulation was the fastest, followed by verification and then counterfactual reasoning. The counterfactual reasoning task also has the lowest accuracy.

Recent studies [28] found that human-understandable models include linear regression, logistic regression, and decision trees. The linearity of the learned relationship makes human-understanding easy, especially for the model with monotonous constraints [28].

Logistic regression is an extension of linear regression, typically for classification problems. Instead of fitting a straight line or a hyperplane, it outputs the prediction probabilities between 0 and 1 for different classes. While linear regression and logical regression are limited to linear relationships, decision trees can be used to represent non-linear relationships to some extent (i.e., some Boolean combinations of them). A decision tree model splits the data multiple times according to certain atomic propositions on the input features.

Figure 3.1 shows one sample model for each kind. The task is to classify the blue and red dots. Given the task, a linear regression, shown in Figure 3.1(a), is a best-fit straight line which aims to separate the dots based on predicted value; a logistic regression, shown in Figure 3.1(b), takes the weighted sum of the inputs and applies an activation function such as Sigmoid to generate classifications;



(a) Linear regression (b) Logistic regression (c) Decision tree

Figure 3.1: Human-understandable models

and a decision tree, shown in Figure 3.1(c), separates the input space into multiple regions and each of the regions is assigned with one predicted classification label.

Intuitively, our idea of defining and measuring how easy it is to explain the decisions of a neural network model is based on measuring how often the prediction results can be "explained" consistently using a human-understandable model. Existing user studies suggest that humans are capable of understanding simple models such as the kind of decision trees that we focus on [22]. Thus, by measuring how often the neural network's decisions can be explained by such a simple model, we get a measure on how often people can understand the neural network's decisions.

In this work, we focus on decision trees over linear regressions for the following reasons. First, as samples end up in distinct groups according to a decision tree model, it is arguably easier to understand than points on lines or multi-dimensional hyperplanes as in linear regression models. Secondly, the tree structures of decision trees have a natural visualization and humans can reason about its decision-making process following its hierarchical structure. Thirdly, decision trees are reasonably expressive (e.g., they are more expressive than linear regressions and they can capture certain non-linear relationships) and often achieve more accurate prediction results on real-world tasks [179, 75].

3.2.2 Learning Decision Trees

Note that our approach is not approximating a given neural network using decision tree, but quantitatively evaluating neural network explainability based on decision tree (or any other simple approximation model) which works as a proxy. So our approach relies on methods for generating decision trees. In the literature, there are well-studied algorithms for generating decision trees [140, 32, 75]. In this work, we adopt the existing method called the classification and regression trees (CART [22]) to construct decision trees. CART relies on the learning samples which are a set of historical data with assigned classes for all observations to learn decision trees. It raises a sequence of yes/no questions, each of which splits the learning samples into two partitions. In other words, given a set of labeled feature vectors, the algorithm identifies one atomic proposition each time for splitting the domain of each feature. The atomic proposition is identified greedily by minimizing the Gini score, which is defined in [40]. In the following, we provide a formal definition of decision trees and refer the readers to the existing literature on how to learn decision trees.

Definition 1 (Decision trees) *Given a set of atomic propositions $\psi_1, \psi_2, \cdots, \psi_m$, a decision tree is a binary tree where each node has two outgoing edges, one labeled with a proposition ψ_i and the other labeled with $\neg\psi_i$.* □

A sample decision tree is shown in Figure 3.2. If a decision tree has height n, there are a maximum of 2^n leaf nodes. Each leaf node is associated with a prediction. Given a sample, the prediction by the tree can be identified by navigating from the root of the tree and following the edges according to the truth value of the proposition associated with the node, until a leaf node is reached.

The above definition is parameterized with a set of propositions. The choice of propositions would obviously be related to whether the decision tree is considered human-understandable. In this work, we restrict propositions to the form $f \geq d$ where f is a feature and d is a constant, e.g., $income \geq 50K$. We choose Boolean proposition for linear relationship instead of continuous in the range, e.g., $income \in [50K, 60K]$, because the former proposition always splits the space into two partitions which is easier for human reasoning and two such

Algorithm 1 *buildTree*(LD, KF, K) where *LD* is a set of labeled data and *KF* is a set of features

```
1: let $node$ be a new node
2: if K = 0 or all samples in $LD$ have the same label then
3:     return $node$
4: else
5:     identify a proposition $\psi$ of the form $f \geq d$ such that $gini_\psi$ is minimum
       for all $f \in KF$ and $d$
6:     let $left$ be the samples in $LD$ that violates $\psi$
7:     let $right$ be the samples in $LD$ that satisfies $\psi$
8:     let left child of $node$ be $buildTree(left, KF, K-1)$ and right child be
       $buildTree(right, KF, K-1)$
9: end if
```

propositions work the same as continuous in the range. We leave it to future work to consider propositions relating to multiple features.

The details of the decision tree building algorithm are shown in Algorithm 1. We limit the height of the tree to be maximum K, which is an input parameter, along with a set of labeled samples *LD* and features KF. Note that, instead of letting CART algorithm do the feature selection, we use KF to define features used to approximating the neural network. In this setting, we support evaluating how easy it is to explain neural network's decisions using any user-provided feature set and also ignore the possible effect of sensitive features, e.g., gender or race. In the base case, if $K = 0$, we simply return a new node, which will be a leaf of the decision tree. Otherwise, we identify a proposition ψ at line 5 (by trying all feature f and constant d in f's domain). Afterwards, the set of labeled samples is split into two, i.e., *right* contains those which satisfy ψ, and $left$ contains the rest. A perfect split is one that all samples in $left$ have the same label and all samples in *right* have the same label. In such a case, the Gini score is 0. Since we exhaustively search for the ψ which has the minimum score (assuming that there are finitely many d values) for each level of the decision tree, the generated decision tree is guaranteed to have better accuracy on *LD* than any other decision trees. This is stated in the following theorem.

Theorem 1 *If Algorithm 1 returns a decision tree D with accuracy ϕ, that does* *not exist a decision tree D with the same height and accuracy ϕ' such that $\phi' > \phi$ [22] .*

3.2.3 Defining Measurements

In the following, we define multiple measurements that can be used to quantify how easy it is to explain the decisions of a neural network and also how easy it is for human to understand neural network's decisions. Note that we evaluate on the predictions of the neural network instead of the ground truth labels.

Definition 2 (K-explainability) *A model N's decisions are K-explainable with respect to a set of samples T iff there exists a decision tree D such that height(D) $\leq K$ and $D(i) = N(i)$ for all $i \in T$, where $N(i)$ is the prediction on i by N and $D(i)$ is the prediction on i by D.* □

Intuitively, a model N's decisions are K-explainable if and only if its decisions can be consistently explained using a decision tree with a height no more than K. In the above definition, the decision explainability is parameterized with a limit on the height of the decision tree K. It is known [75] that the higher the decision tree is, the less understandable it is. Thus, we must take the height of the decision tree into account. Furthermore, the above is defined with respect to a set of samples T, i.e., the set of samples on which we evaluate the model's decision explainability. In this work, we focus on two sets. One is the training set, i.e., the decision explainability of a model is tested and measured against those samples in the training set. The other is the entire input space, i.e., the decision explainability of a model is tested against any sample, including those that are yet to be seen. The latter is clearly more demanding than the former. Given the above definition, our problem is to test whether a model N's decisions are K-explainable. We show an algorithm to solve the problem in Section 3.3.3.

In the following, we define K, ϕ-explainability which is a slightly relaxed notion of decision explainability. That is, instead of demanding that a neural network is completely consistent with a decision tree with a limited height (which is often unlikely in practice as we show in Section 6.5), we require that the explaining model must reach a certain level of consistency with the neural network model in terms of the decisions.

Definition 3 (K,ϕ-explainability) *Let ϕ be a percentage. A model N's decisions are K,ϕ-explainable with respect to a set of samples T if and only if there exists a decision tree D such that $height(D) \leq K$ and the fidelity of D with respect to N is no less than ϕ where fidelity is defined as follows.*

$$Fidelity = \frac{\sum_{i \in T} Sign(N(i) = D(i))}{\#T} \tag{3.1}$$

where $Sign(b)$ is a function which returns 1 if b is true.

Intuitively, a model N's decisions are K,ϕ-explainable if and only if its decisions can be explained consistently using a decision tree with a height no more than K in most of the cases (defined by ϕ). The value of ϕ provides a quantification of the explainability. Note that we similarly distinguish two cases based on what T refers to. One is the training set and the other is the entire input space. Note that in the latter case, since there may be infinitely many samples, measuring the fidelity is non-trivial. To evaluate whether a model's decisions are K,ϕ-explainable, we would like to have an algorithm which can systematically measure the fidelity of the best decision tree with a height no more than K. We present such an algorithm in Section 3.3.3.

With the above-defined measurements, we can then compare the decision explainability of two neural network models N_1 and N_2. We say that N_1 is more explainable than N_2 if N_1's decisions are K_1-explainable, N_2's decisions are only K_2-explainable and $K_1 < K_2$; or N_1's decisions are K,ϕ_1-explainable, N_2's decisions are only K,ϕ_2-explainable and $\phi_1 > \phi_2$.

Algorithm 2 *isExplainableTrainSet* (K,T,N)

```
1: label $T$ using $N$ to obtain a labeled dataset $LD$
2: sets = all combinations of $K$ features
3: for each set of features $KF$ in sets do
4:     $D = buildTree(LD, KF, K)$
5:     if $\forall\, x$ in $LD$, $D(x) = N(x)$ then
6:         return $true, D$
7:     else
8:         return $false, None$
9:     end if
10: end for
```

3.3 Testing Decision Explainability

In this section, we answer the following question: how do we measure the decision explainability of a user-provided neural network model? We present four algorithms according to the two measurements defined in Section 3.2.3.

3.3.1 K-Explainability Testing against Training Set

In the following, we present algorithms for testing whether a neural network's decisions are K-explainable. We distinguish two cases. One is that the decision explainability is defined with respect to the training set. The other is that it is defined with respect to the entire input space. The former is solved using Algorithm 2.

Intuitively, the goal of Algorithm 2 is to test whether there exists a decision tree of height K which perfectly explains every prediction of the neural network. It takes the neural network model N and the training set T as well as K as inputs. At line 1, we label each feature vector in T with the corresponding prediction of the neural network N. Note that our goal is to explain N's decisions and thus the ground truth labels of T are irrelevant here.

At line 2, we identify all combinations of K-features. Suppose each input feature vector has a total of M features. Then there are C_M^K different combinations of K features. The loop from line 3 to 10 tries every combination one by one. Note that this loop can be easily parallelized. For each combination,

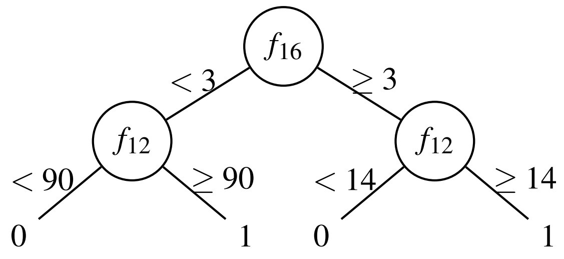


Figure 3.2: A sample decision tree on the Bank dataset

Algorithm 1 is applied to generate the 'best' decision tree using the given set of features. If perfect accuracy is achieved, i.e., every feature vector is classified with a label that is consistent with that of N, a decision tree satisfying Definition 2 with respect to the training set is identified and thus the neural network model's decisions are K-explainable. Furthermore, the generated decision tree is reported as evidence of the model's decision explainability. The complexity of the testing is $\Theta(C_M^K)$, where M is the dimension of feature vectors and K is the depth of decision trees. We remark that the complexity can be high when M is large and we experiment heuristics in Section 6.5 which may significantly reduce the complexity.

The following theorem states the soundness and completeness of the algorithm, whose correctness can be easily established based on Theorem 1.

Theorem 2 *A neural network N's decisions are K- explainable with respect to the training set if and only if Algorithm 2 returns true.* □

Example 3.3.1 *We use the* ***Bank*** *[110] dataset to illustrate Algorithm 2. Each sample in the dataset has 16 features. Assume K is 2 and there are C_{16}^2 different combinations of 2 features. Next, each combination is tested. Let us take the combination of f_{12} (with domain $0-99$) and f_{16} (with domain $0-3$) as an example. The decision tree generated at line 4 is shown in Figure 3.2.*

Based on the decision tree, we have 3 propositions. ψ_1 means $f_{16} < 3$, ψ_2 means $f_{12} < 90$ and ψ_3 means $f_{12} \geq 14$. We then travel through all samples in the training set to see whether the decision tree is completely consistent with the neural network. If we find a sample x such that the prediction of the neu-

ral network and decision tree is different, the neural network is considered not explainable according to this decision tree. That is, the decisions of the neural network cannot be explained entirely based on the values of the feature f_{12} and f_{16}. The following is such an example, whose prediction according to the neural network is 1 whereas its prediction according to the decision tree is 0.

$$x = [3,7,0,2,0,1,0,0,0,8,4,81,1,1,0,0]$$

3.3.2 K-Explainability Testing against All Inputs

It may be insufficient to explain a model's decisions over the training set only, as a future sample might not fit in the explanation. In the following, we present an approach to test the explainability of a model N's decisions against all possible inputs including the unseen ones. In this setting, applying Algorithm 2, which requires us to exhaustively enumerate all possible inputs, is infeasible due to the enormous input space.

We use a best-effort approach to solve the problem. That is, we first sample (uniformly in the entire input sample space) a certain number of feature vectors and their corresponding labels (i.e., predictions made by N). Note that our algorithms can be readily adopted to a prior distribution (such as a normal distribution) that is provided. We remark that it is hard to define the actual feature distributions and we view the problem of identifying the actual feature distribution as a problem that is orthogonal to our problem. After sampling, we learn a decision tree that is best consistent with the samples and then test whether the decision tree is consistent throughout the sample space. The details of the algorithm are shown in Algorithm 3.

It takes N as well as K as inputs. At line 1, we first generate a set of feature vectors as X from the entire input space randomly. Note that our approach can be easily modified to sample according to the data distribution of the training set if it is provided (or extracted using existing approaches [136, 178]). Then at line 2, we label all feature vectors in X with the prediction of N to obtain a set of labeled samples *LD*. Afterwards, the same as in Algorithm 2, we identify all combinations of *K*-features at line 3. Then we test every combination in the loop from line 4 to 9. For each combination, Algorithm 1 is applied to build the 'best' decision tree. If the decision tree fails to achieve 100% accuracy on *LD*, we proceed to try the next combination. Otherwise, we further test whether the decision tree is applicable to the entire input space using Algorithm 4.

Algorithm 4 is inspired by existing approaches on adversarial perturbation [55, 83] with a different objective function. That is, the goal is to apply perturbation to samples in *LD* so as to find a 'counterexample', i.e., one such that the prediction of *N* and that of the decision tree are different. First, we cluster those samples in *LD* using a standard clustering algorithm k-means at line 1, where #*c* is the size of clusters [92]. Note that the reason for clustering is that we can diversify the search for counterexamples. Afterwards, we obtain seed samples from each cluster in a round-robin fashion at line 2. For each seed sample, we apply a gradient-based algorithm to search for counterexamples iteratively (see loop at line 4 to 11). The gradient is commonly used to generate adversarial samples [119, 109]. The intuition is to perturb the original input according to the direction of the gradient so that the prediction of the neural network has the maximum change. During each iteration of the loop, we calculate the gradient of the loss function on the input x as $\mathrm{J}(\mathrm{x})$ and calculate sign of the gradient as $sign(\mathrm{J}(\mathrm{x}))$. Here, the *sign* function is used to extract the sign of the real gradient. Then we perturb the seed sample x with an amount $step\_size * grad$, where $step\_size$ is used to determine the perturbation degree in each time and *grad* is the gradient.

Note that Algorithm 4 is a best-effort approach as we limit the number of iterations at line 4. If we find a counterexample, the algorithm returns $false$; otherwise, it returns $true$. Only if Algorithm 4 fails to find a counterexample, Algorithm 3 reports $true$ with D at line 7.

Example 3.3.2 *We illustrate how Algorithm 3 works using the **Bank** dataset*

Algorithm 3 $isExplainableAllInput\,(K,N)$

```
1:  generate sample set $X$ from input space randomly
2:  label $X$ using $N$ to obtain a labeled dataset $LD$
3:  $sets$ = all combinations of $K$ features
4:  for set $KF$ in $sets$ do
5:      $D = buildTree\,(KF,LD,K)$
6:      if $\forall$ $x$ in $LD$, $D(x) = N(x)$ and $attackFail\,(LD,D)$ then
7:          return $true, D$
8:      end if
9:  end for
10: return $false, None$
```

Algorithm 4 $attackFail\,(LD,D)$ where #c is constant

```
1:  clusters = KMeans($LD$, #$c$)
2:  for $i$ from 0 to $p\_num$ do
3:      Get seed x from clusters in a round-robin fashion
4:      for iter from 0 to max_iter do
5:          $grad = sign(\mathrm{J}(\mathrm{x}))$
6:          $x' = x + step\_size * grad$
7:          if $x'$ is valid and $N(x') \neq D(x)$ then
8:              return $false$
9:          end if
10:         $x = x'$
11:     end for
12: end for
13: return $true$
```

example based on the feature combination (f_{12}, f_{16}). We first generate 5000 samples randomly and generate a decision tree D which is consistent with the neural network predictions. We then apply Algorithm 4 to test whether D is consistent throughout the input space. Assume that we take the following x as a seed.

$$x : [5,10,1,1,0,19,1,0,0,5,4,81,1,1,0,0]$$

Intuitively, the goal is to perturb x such that its label changes and hopefully becomes different from that of D. The perturbation is guided by the gradient (so that the perturbation would cause a maximum change in the prediction). That is, we calculate the sign of x's gradient as follows.

$$grad : [1,1,1,-1,1,1,-1,1,1,-1,1,0,-1,-1,-1,0]$$

Algorithm 5 *explainabilityTrainSet* (K, T, N, ϕ)

```
1: label $T$ using $N$ to obtain a labeled dataset $LD$
2: sets = all combinations of $K$ features
3: for each set of features $KF$ in sets do
4:     $D = buildTree(LD, KF, K)$
5:     if accuracy of $D$ is no less than $\phi$ then
6:         return $true, D$
7:     end if
8: end for
9: return $false, None$
```

Next, we perturb x by updating each feature f_i to be f_i + $step\_size * grad[i]$ where $step\_size$ is 1 in this example. The result is the following sample.

$$x' : [6, 11, 2, 0, 1, 20, 0, 1, 1, 4, 5, 81, 1, 0, 0, 0]$$

Note that a clipping function is applied to make sure the perturbed feature value remains in its domain. The predictions of this sample by N and D are 0 and 1 respectively. As a result, D fails to be taken as an explanation of the neural network N.

3.3.3 K, ϕ-Explainability Testing

As discussed above, due to the gap between the expressiveness of decision trees and neural networks, there is likely no decision tree (with a height limit such as K) that can be used to perfectly explain the decisions of the neural network. It is thus more useful to quantify the explainability by testing K, ϕ-explainability of the given neural network in practice. In the following, we similarly distinguish two cases, i.e., the training set and the entire input space. The former is addressed using Algorithm 5. The latter is addressed using Algorithm 6.

Algorithm 5 is similar to Algorithm 2. The only difference is that once Algorithm 1 is applied to generate the 'best' decision tree, the fidelity of the decision tree is compared with the given threshold ϕ. If a decision tree with a fidelity no less than ϕ is identified, the algorithm returns $true$ together with the decision tree. The following theorem states the soundness and completeness of the algorithm, whose correctness can be easily established based on Theorem 1.

Theorem 3 *A neural network N's decisions are K,ϕ-explainable with respect to the training set if and only if Algorithm 5 returns true.* □

Example 3.3.3 *Assume that we are testing whether a neural network trained on the* ***Bank*** *dataset is $2, 90\%$-explainable. Further, assume that we are testing the combination of f_{12} and f_{16}. Applying Algorithm 1, we obtain the decision tree shown in Figure 3.2, which has an accuracy of 94.92% (calculated based on LD). As a result, Algorithm 5 returns true. That is, we can explain 90% of the decisions of the neural network (over the training set) using a simple explanation based on the decision tree.*

Testing whether a model's decision is K,ϕ-explainable with respect to all inputs is challenging as it is infeasible to exhaustively enumerate all inputs and check whether there exists a decision tree with prediction accuracy more than ϕ. We thus take a best-effort approach similar to that of Algorithm 3. That is, we uniformly sample a certain number of samples and build a candidate decision tree. Afterwards, we evaluate whether this candidate has a fidelity of ϕ throughout the input space. We formulate the latter problem as a hypothesis evaluation problem and solve it systematically using hypothesis testing (so that we have a certain level of statistical confidence in the evaluation result [154]). That is, given the candidate decision tree D and the threshold ϕ, we formulate two hypotheses. One is that the fidelity of D is no less than ϕ and the other is that it is less than ϕ. We then keep sampling randomly (in an i.i.d. manner [33]) from the entire input space until one of the hypotheses reaches a certain level of statistical confidence.

The details of the algorithm used to evaluate whether a candidate decision tree achieves a fidelity of ϕ against all inputs are shown in Algorithm 6. It is designed based on the sequential probability ratio test (SPRT) algorithm proposed in [155]. SPRT decides whether to accept a hypothesis after evaluating every

Algorithm 6 $explainabilityAllInput\,(K,N,\phi)$

```
1: generate sample set $X$ from input space randomly
2: label $X$ using $N$ to obtain a labeled dataset $LD$
3: $sets$ = all combinations of $K$ features
4: for set $KF$ in $sets$ do
5:     $D = buildTree\,(KF, LD, K)$
6:     if $SPRT\,(N,D,\phi)$ then
7:         return $true, D$
8:     end if
9: end for
10: return $false, None$
```

Algorithm 7 $SPRT\,(N,D,\phi)$

```
1: $p_0 = \phi - \sigma$
2: $p_1 = max(0.99, \phi + \sigma)$
3: stop = False
4: while not stop do
5:     generate sample x from input space randomly
6:     $n$ = size of totally detected samples $X$
7:     $s$ = size of $x \in X | N(x) = D(x)$
8:     $sprt\_ratio$ = calculate_sprt($s, n, p_0, p_1$)
9:     if $sprt\_ratio \geq \frac{1-\beta}{\alpha}$ then
10:         return accept
11:     end if
12:     if $sprt\_ratio \leq \frac{\beta}{1-\alpha}$ then
13:         return deny
14:     end if
15: end while
```

new sample. The SPRT algorithm is parameterized by α, β and indifference region σ, which intuitively define the error bounds. According to ϕ and σ, we calculate p_0 and p_1 in line 1 and 2. During the loop from line 4 to 15, we keep generating new samples (i.e., by randomly generating values for each feature from its domain). Then we detect whether labels predicted by N and D are the same and update n and s respectively (see in line 6 and 7). Then we calculate the SPRT probability ratio at line 8 using the following formula.

$$sprt\_ratio = \frac{p_1^s (1-p_1)^{n-s}}{p_0^s (1-p_0)^{n-s}} \tag{3.2}$$

From line 9 to 14, we compare the SPRT ratio with the acceptance/rejection bounds. The test accepts the hypothesis if the SPRT ratio is larger than or equal to the acceptance bound or denies it if the SPRT ratio is less than or equal to the rejection bound. The algorithm stops whenever a hypothesis is accepted at line 10 or denied at line 13. Note that this algorithm is guaranteed to terminate and the probability of accepting the wrong hypothesis is bounded [155].

Example 3.3.4 *Let us use the* ***Bank*** *dataset and the feature combination of* f_{12} *and* f_{16} *as an example. The candidate decision tree is shown in Figure 3.2. To check whether this decision tree is able to explain the decisions on 90% of the inputs throughout the input space, we apply the above-mentioned algorithm. In our experiments, we set* $\alpha=\beta=0.05$, $\sigma=0.05$ *and* $\phi=0.90$. *Then* $accept\_bound$ *and* $deny\_bound$ *are 19.0 and 0.053 respectively. After detecting 28 samples, all 28 samples have consistent prediction labels and the calculated* $sprt\_ratio$ *is 22.52 by formula 3.2. Since* $pr \geq accpet\_bound$, *we accept the hypothesis that the probability of consistency within the entire input space can reach 90%.* □

3.4 Implementation and Evaluation

Our approach has been implemented as a self-contained software toolkit based on Tensorflow [1] and scikit-learn [118]. It is implemented with a total of about 4K lines of Python code. Our experiments are based on the following datasets which have been used as evaluation subjects in multiple previous studies [7, 176].

- Adult Income [128]: The Adult Income dataset was published in 1996. The prediction task is to determine whether the income of an adult is above $50,000 annually based on his/her personal information. The dataset contains 32561 samples, each of which has 13 features.
- Bank Marketing: The dataset came from a Portuguese banking institution and is used to train models for predicting whether the client would subscribe a

term deposit based on his/her information. The size of the dataset is more than 45,000 and each record contains 16 attributes.

- German Credit [64]: This is a small dataset with 600 samples, each of which has 20 features. The task is to assess an individual's credit based on personal and financial records.

For each dataset, we train a neural network using the exact same configuration as reported in the previous studies [66, 176]. The details are shown in Table 3.1. All these neural networks contain six layers. Each hidden layer contains 64, 32, 16, 8 and 4 units. The output layer contains 2 (number of predict classes) units. ReLU is used as the active function. Lastly, the Softmax function is used to output prediction probabilities. Although we focus on feedforward neural networks on classification tasks in our study, our algorithms work for more complex neural networks such as convolutional neural networks (CNNs) in general. Furthermore, we focus on neural networks trained on tabular feature vectors instead of complex data such as images and texts. This is because, in order to explain the decisions of neural networks trained on images and texts, we must additionally address the challenge of identifying high-level human-understandable features before we can learn the decision trees. Note that high-level feature extraction for such complicated data is itself an active research field [88, 37].

In the following, we report the evaluation results which are conducted to answer multiple research questions (RQ). Note that Algorithm 4 and Algorithm 7 require multiple hyper-parameters, whose values are either determined empirically (such as the number of clusters and the max number of iterations for perturbation in Algorithm 4) or adopted from standard practice (such as α and β for hypothesis testing). The details are shown in Table 3.2. Here *p_num* is set to be 1000 for Adult Income and Bank data and 600 for Credit data due to its small size. All experiments are conducted on a laptop running macOS (10.15.6) with 16 GB memory. Each experiment is set with a timeout of 1000 hours. All

Table 3.1: Dataset and Models of Experiments

Dataset	**Model**	**Accuracy**	**attributes**
Adult Income	Six-layer Fully-connected NN	86.13%	13
Bank Marketing	Six-layer Fully-connected NN	91.62%	16
German Credit	Six-layer Fully-connected NN	100%	20

Table 3.2: Parameters of the experiments

Parameter	**Value**	**Description**
c_num	4	cluster count
max_iter	10	max number of iterations for perturbation
$step\_size$	1	step size of perturbation
p_num	1000,600	number of seed instances for perturbation
α, β	0.05	error probability bounds
σ	0.05	indifference region

models and experiment details are available online at [171].

RQ1: Are our algorithms efficient on testing the models' decision explainability? This question is designed to evaluate the efficiency of our algorithms, particularly the effect of the design parameters such as K and ϕ. To answer this question, we systematically apply our algorithms to the above-mentioned models and measure the results including efficiency. That is, we test each model against K-explainability for different K against the training set and the entire input space; and test each model against K, ϕ-explainability with different K and ϕ. The results of testing K-explainability are summarized in Table 3.3.

The third column shows the testing results and time taken for all three models with respect to different K on the training set. For all models, we can evaluate the models' K-explainability with a K value of 2 or 3. Note that in this setting, the optimal decision tree on the training set is generated based on all instances in the training set for each combination of K features, which is time-consuming. As a result, we can test 4 or 5-explainability only on the model trained based on the Credit dataset. In our experiments, more than 99% of the time is spent on generating the decision trees. One way to reduce time consumption is to paral-

Table 3.3: K-explainability testing results

Dataset	**K**	**Testing training set**		**Testing all inputs**	
		Result	Time (min)	Result	Time (min)
Adult Income	2	No	370.01	No	33.68
	3	No	12485.24	No	223.27
	4	No	T/O	No	685.57
	5	No	T/O	No	2110.25
Bank	2	No	5538.1	No	45.6
	3	No	51647.40	No	384.63
	4	No	T/O	No	1980.77
	5	No	T/O	No	6149.42
Credit	2	No	0.48	No	241.77
	3	No	7.22	No	1470.45
	4	No	33.12	No	6456.2
	5	No	121.17	No	28250.87

lelize the generation of decision trees for different combinations of features.

The fourth column shows the testing results and time taken for all inputs. We sample a threshold number of instances to build a candidate decision tree. The results here are based on training candidate models with 5000 random instances. Note that given the relatively small number of instances for building the candidate tree, we can finish testing up to 5-explainability for all models before timeout and the number of random instances may have an effect on the quality of the candidate models. The number 5000 is determined empirically based on the experiment results shown in Table 3.4. It shows the time taken of generating one decision tree as well as the maximum accuracy ϕ achieved by the candidate model on the Adult Income dataset. It can be observed that the accuracy achieved by the decision tree often maximizes when the number of instances is 5000. Further increasing the number of instances increases the time proportionally without increasing the accuracy. In multiple cases, the accuracy even drops. Thus, we set the 5000 as a threshold to learn candidate decision trees in testing against all inputs.

The results of testing K, ϕ-explainability against different K and ϕ as well as the training set or all inputs are summarized in Table 3.5. ϕ is set to be 5 values,

Table 3.4: Training time with different #instances

K	#instances	T_{dt}(s)	$max(\phi)$
2	2000	4.2	93%
	5000	25.74	93%
	10000	112.11	93%
3	2000	9.06	95%
	5000	46.66	96%
	10000	133.21	95%
4	2000	15.17	95%
	5000	57.34	97%
	10000	140.94	95%
5	2000	21.32	96%
	5000	98.15	97%
	10000	202.06	97%

i.e., 70%, 80%, 90%, 95%, and 99%. In the third column, we show the testing results and time taken for evaluating the fidelity of all decision trees against the training set. Note that we can still measure prediction accuracy on decision trees generated before timeout. The fourth column shows the results and time taken on testing K, ϕ-explainability against all inputs.

We note that the execution time is dominated by the time required for learning the decision trees. Since we consider all feature combinations with different size K, the number of all possible feature sets would be large especially with regards to some training set with high-dimensional feature vectors. In the case of testing K, ϕ-explainability against all inputs, it is strongly related to the number of feature combinations and the number of samples we use to train the candidate decision trees.

One practical way to reduce the complexity is thus to heuristically select a subset of the features, i.e., those which are likely correlated to the neural network's decision. For example, SHapley Additive explanation uses Shapely values to compute the contributions of the features [98]. Based on the Shapely values, we can focus on the features with high contribution to generate the decision trees. In the following, we conduct experiments to evaluate this idea by focusing on the top 6 contributing features based on Shapely value ranking.

Table 3.5: K, ϕ-explainability testing result

Dataset	K	Testing against training set										Testing against all inputs									
		70%		80%		90%		95%		99%		70%		80%		90%		95%		99%	
		Re.	Time	Re.	Time	Re.	Time	Re.	Time	Re.	Time	Re.	Time	Re.	Time	Re.	Time	Re.	Time	Re.	Time
Adult Income	2	Yes	4.75	Yes	4.75	No	370.02	No	370.02	No	370.02	Yes	0.43	Yes	6.75	Yes	33.78	No	33.83	No	33.79
	3	Yes	43.69	Yes	43.69	No	12485.33	No	12485.33	No	12485.33	Yes	1.75	Yes	44.70	Yes	223.42	Yes	223.42	No	223.38
	4	Yes	T/O	Yes	T/O	Yes	T/O	No	T/O	No	T/O	Yes	0.96	Yes	137.14	Yes	411.43	Yes	480.02	No	685.69
	5	Yes	T/O	Yes	T/O	Yes	T/O	No	T/O	No	T/O	Yes	1.64	Yes	422.08	Yes	1266.25	Yes	1582.82	No	2110.37
Bank	2	Yes	46.15	Yes	46.15	Yes	46.15	No	5538.14	No	5538.14	Yes	0.38	Yes	0.38	Yes	38.89	No	45.78	No	45.74
	3	Yes	92.23	Yes	92.23	Yes	92.23	Yes	92.23	No	17523.23	Yes	0.69	Yes	1.67	Yes	192.39	Yes	373.27	No	384.78
	4	Yes	T/O	Yes	T/O	Yes	T/O	Yes	T/O	No	T/O	Yes	1.09	Yes	1.09	Yes	990.47	Yes	1723.41	No	1980.91
	5	Yes	T/O	Yes	T/O	Yes	T/O	Yes	T/O	No	T/O	Yes	1.41	Yes	1.41	Yes	3647.79	Yes	5780.61	No	6149.59
Credit	2	Yes	0.48	No	0.48	No	0.48	No	0.48	No	0.48	Yes	26.69	No	241.79	No	241.79	No	241.79	No	241.79
	3	Yes	0.63	No	7.22	No	7.22	No	7.22	No	7.22	Yes	242.50	No	1470.48	No	1470.48	No	1470.48	No	1470.48
	4	Yes	0.68	No	33.17	No	33.17	No	33.17	No	33.17	Yes	1700.34	No	6456.23	No	6456.23	No	6456.23	No	6456.22
	5	Yes	0.78	No	121.30	No	121.30	No	121.30	No	121.30	Yes	8394.73	No	28250.90	No	28250.90	No	28250.90	No	28250.89

Table 3.6: K, ϕ-explainability testing result based on Shapley values

Dataset	K	Testing against training set									
		70%		80%		90%		95%		99%	
		Re.	Time	Re.	Time	Re.	Time	Re.	Time	Re.	Time
Adult Income	2	Yes	5.04	Yes	5.04	No	37.24	No	37.24	No	37.24
	3	Yes	5.34	Yes	5.34	No	41.86	No	41.86	No	41.86
	4	Yes	6.87	Yes	6.87	Yes	33.55	No	70.87	No	70.87
	5	Yes	5.11	Yes	5.11	Yes	5.11	No	29.25	No	29.25
Bank	2	Yes	4.16	Yes	4.16	Yes	41.82	No	41.82	No	41.82
	3	Yes	5.31	Yes	5.31	Yes	46.05	Yes	46.05	No	46.05
	4	Yes	5.94	Yes	5.94	Yes	29.63	Yes	44.31	No	44.31
	5	Yes	6.24	Yes	6.24	Yes	6.24	Yes	27.11	No	27.11
Credit	2	Yes	1.53	No	19.06	No	19.06	No	19.06	No	19.06
	3	Yes	5.45	No	15.77	No	15.77	No	15.77	No	15.77
	4	Yes	2.96	No	20.28	No	20.28	No	20.28	No	20.28
	5	Yes	4.03	No	11.22	No	11.22	No	11.22	No	11.22

That is, we only consider the decision trees with the selected features and test each model against k, ϕ-explainability against the training set and the entire input space. The reason why we select 6 features is that the maximum K in our experiments is 5. The testing results are summarized in Table 3.6.

We can observe that, although we focus on the top 6 features only, the testing results (i.e., whether each model satisfies the corresponding explainability metric) remain exactly the same as the results in Table 3.5. The time cost however decreases significantly, especially for the first two neural networks trained on Adult Income dataset and Bank dataset. When K is set as 4 or 5, the time cost reduces from T/O to no more than 71 minutes. Furthermore, the execution time for most tests is far less than 1 hour. In general, our algorithms is efficient on testing the neural networks' explainability.

RQ2: Are existing models' decisions explainable? This question aims to apply our approach to study decision explainability of neural network models. To answer the question, we investigate the decision explainability of the models using our approach based on the results shown in Table 3.3 and Table 3.5.

In terms of K-explainability, it can be observed that every model is found to be un-explainable, whether considering only those in the training set or all inputs. In other words, no matter what combination of K-features used to explain the decisions of the neural network, there are always counterexamples, i.e., instances which are predicted differently by the decision tree and the neural network. In particular, the adversarial sampling approach adopted in Algorithm 4 is proved to be effective in identifying such counterexamples. These results show that indeed it is unlikely that we can always explain the decisions of a neural network using a decision tree. Thus, the goal should perhaps be minimizing the percentage of such un-explainable cases.

In terms of K, ϕ-explainability test, the results shown in Table 3.5 are mixed. We have several observations. First, comparing the results with different K values, the bigger the K is, the bigger a ϕ can be achieved. This is intuitively reasonable since it is easier to explain the prediction of a neural network with a more complicated decision tree. Second, it is not necessarily easier to explain the instances in the training set than to explain all input instances. Note that there are instances where a model fails a K, ϕ-explainability test on the training set but passes the test on the entire input space. For instance, the models trained on Adult Income dataset pass the test with higher ϕ against all inputs than against the training set. In a close investigation, we discover that this training sets are highly imbalanced, e.g., samples with one label are significantly more than samples within the other labels. Neural networks trained on such an imbalanced dataset are known to produce imbalanced predictions, e.g., the majority of predictions on random samples are the same label. The majority of the randomly generated samples of Adult Income dataset are predicted as label 1. Such imbalanced predictions are much easier to explain, i.e., it can be explained with a simple decision tree which always generates the same prediction (e.g., "everyone's application is rejected"). This is confirmed in our experiments, most of the randomly generated samples are predicted with the same label. As a result, even with a K value of 3, a ϕ of 95% can be achieved. Furthermore, in our experiments, we test K, ϕ-explainability against all inputs using the SPRT algorithms, since it is infeasible to enumerate all inputs. The SPRT algorithm provides only statistically results with a bounded range of errors, which is more "relaxed" compared to testing against all samples in the training set.

Lastly, it can be seen that the results vary across different models. For instance, the neural network trained on the Credit dataset has the lowest decision explainability. The fidelity of the learned decision trees with respect to the training set and all inputs is less than 80% no matter what K value is used. The highest ϕ achieved by the neural network against all inputs is only 70%. Our interpretation of the result is as follows. Because this dataset is very small, the model is less robust compared with the models trained on the other dataset. In other words, its predictions on unseen instances are rather random and thus hard to explain. In general, existing neural networks' decisions have low explainability.

RQ3: Are robust models' decision more explainable? One of the observations in the above experiments is that Algorithm 4 is often successful in finding instances which are un-explainable by the decision tree with adversarial sampling. The above experiment results seem to suggest that the lack of robustness often makes a model's decision un-explainable. This question is thus designed to test this hypothesis, i.e., with the help of robust training, we aim to see whether more robust models' decisions are more explainable. That is, whether it becomes harder to find 'counterexamples' that have different predictions of the neural network and the decision tree after retraining. Here, we use the technique called FGSM [55] to compute adversarial perturbations and retrain the model. Note that the label of the samples generated through adversarial perturbation is the same as the original sample.

To answer this question, we test the decision explainability of the retrained models using our approach and check whether a higher ϕ can be achieved. After robust training, the level of ϕ on the training set remains almost identical to that without robust training. We thus focus on K,ϕ-explainability testing against all inputs. The results are shown in Table 3.7, where 7 values of ϕ (i.e., 70%, 75%, 80%, 85%, 90%, 95% and 99%) are tested. We highlight improved results in green and worsened results in red.

Compared to the corresponding entries in Table 3.5, we observe that among the 12 cases (4 different K values on three models), 4 results show improvement and 5 results show worse decision explainability after robust training. The two models trained on the Adult Income and Bank dataset become less explainable. For instance, the model trained on Adult Income without robust training is $2,90\%$-explainable against all inputs and it is only $2,85\%$-explainable after training. Our hypothesis is that the two models trained on Adult Income and Bank dataset are able to achieve high K,ϕ-explainability because the model makes simplified predictions as the result of imbalanced training data. After robust training, the training data (i.e., the original data plus those generated through adversarial perturbation) become relatively more balanced. As a result, the neural network model makes more complicated predictions, and thus its K,ϕ-explainability decreases. On the contrarily, the model trained on the Credit dataset becomes much more explainable after robust training. This can be explained by the fact that the Credit dataset, although small, is more balanced, and in such a case, robust training actually improves decision explainability. We acknowledge that this hypothesis needs to be evaluated with a large number of models to be conclusive. So, existing neural networks' decisions are not necessarily easier to explain after robust training.

RQ4: Can we improve model decision explainability using our testing results? Many practical scenarios would prefer models whose decisions can be explained.

Table 3.7: Results after robust training

Dataset	K	ϕ in testing against all inputs 70%	75%	80%	85%	90%	95%	99%
Adult Income	2	Yes	Yes	Yes	Yes	No	No	No
	3	Yes	Yes	Yes	Yes	Yes	No	No
	4	Yes	Yes	Yes	Yes	Yes	No	No
	5	Yes	Yes	Yes	Yes	Yes	No	No
Bank	2	Yes	Yes	Yes	Yes	Yes	No	No
	3	Yes	Yes	Yes	Yes	Yes	No	No
	4	Yes	Yes	Yes	Yes	Yes	Yes	No
	5	Yes	Yes	Yes	Yes	Yes	Yes	No
Credit	2	Yes	Yes	Yes	Yes	No	No	No
	3	Yes	Yes	Yes	Yes	No	No	No
	4	Yes	Yes	Yes	Yes	No	No	No
	5	Yes	Yes	Yes	Yes	No	No	No

This *RQ* thus aims to see whether our approach could help improve model decision explainability. The idea is to check whether a neural network that fails to reach a certain level of K,ϕ-explainability can be improved through retraining with un-explainable adversarial samples identified using our approach. That is, we label those adversarial samples generated by Algorithm 4 with the labels of corresponding seed instances (i.e., the labels of the samples in training set predicted by the neural network). Then we retrain the neural network with these additional samples and apply our approach to test the decision explainability of the retrained models. Note that the premise condition of Algorithm 4 is that all samples x in the original labeled dataset LD satisfy $D(x) = N(x)$ (as shown in line 6 at Algorithm 3). That is, all seed instances x in Algorithm 4 have the same predictions by the neural network and decision tree. Here, we assume the generated adversarial sample x' has the same ground truth label with the original seed instance x. The testing results as well as the accuracy of retrained models are shown in Table 3.8. We highlight improved results in green as well. Note that 6 results show improvement after retraining and none shows worsened decision explainability.

We observe that all three models' decisions become more explainable after

Table 3.8: Results after training with 'counterexamples'

Dataset	**Accuracy**	**K**	**ϕ in testing against all inputs**						
			70%	75%	80%	85%	90%	95%	99%
Adult Income	84.91%	2	Yes	Yes	Yes	Yes	Yes	Yes	No
		3	Yes	Yes	Yes	Yes	Yes	Yes	No
		4	Yes	Yes	Yes	Yes	Yes	Yes	No
		5	Yes	Yes	Yes	Yes	Yes	Yes	No
Bank	91.53%	2	Yes	Yes	Yes	Yes	Yes	Yes	No
		3	Yes	Yes	Yes	Yes	Yes	Yes	No
		4	Yes	Yes	Yes	Yes	Yes	Yes	No
		5	Yes	Yes	Yes	Yes	Yes	Yes	No
Credit	90.3%	2	Yes	Yes	Yes	Yes	No	No	No
		3	Yes	Yes	Yes	Yes	No	No	No
		4	Yes	Yes	Yes	Yes	No	No	No
		5	Yes	Yes	Yes	Yes	No	No	No

retraining. For instance, the models trained on Adult Income and Bank dataset are both $2, 90\%$-explainable against all inputs before retraining. After retraining, both of the two models achieve $2, 95\%$-explainable. The model trained on Credit can achieve ϕ as 85% for any given K after retraining. The results thus proof that our testing algorithms could be potentially used to improve model decision explainability, by paying a relatively small price in terms of accuracy. Although the improvement in the model trained on Credit is more substantial, the accuracy drops mostly from 100% to 90.3% compared with the models trained on Adult Income and Bank dataset. This result suggests that higher explainability may come at the cost of prediction accuracy. More precise trade-offs between the neural network's explainability and accuracy need further experiments. In nutshell, decisions of the neural networks retrained with samples generated by our algorithms are easier to explain.

Threats to validity In the experiment, only 3 datasets are applied to evaluate the effectiveness of our approach. This could be further improved with more models and datasets as well as protected features. Furthermore, in the experiments, we focus on feed-forward neural networks only. Our approach can be potentially adopted for other neural network models. Lastly, the datasets are all tabular data. For complicated models such as RNN for text, the task is likely more complicated.

3.5 Related Work

The term explainability or interpretability has been used to refer to multiple different things. For instance, Murdoch *et al.* attempted to define interpretability in the context of machine learning and placed it as a part of the generic data science life cycle [111]. They defined interpretable machine learning as the use of machine learning models for the extraction of relevant knowledge in data. Montavon *et al.* defined interpretation as the mapping of an abstract concept into a domain that the human can make sense of. Examples of interpretable domains are images or texts [108]. Models from domains with abstract vector spaces are deemed to be un-interpretable. The notion of explainability or interpretability in the above studies is informal and not measurable, i.e., there is no way to quantitatively measure or compare models in terms of explainability or interpretability.

The term interpretability is often associated with studies which aim to provide hints on how neural networks work internally, such as studies on local/global interpretability of neural networks. Studies on local interpretability focus on investigating neural network prediction on one or a specific set of samples. One of the popular methods is the saliency map approach [166]. The idea is to identify specific parts of an input sample that contribute the most to the prediction of the network [146, 35] or the activity of a specific layer in the network [179]. In the case of image classifier interpretation, it is useful to know which parts of the input activate certain filters for the prediction and how each part contributes to the prediction score at the pixel level [98] or at the object level [179]. In [80], Kim *et al.* proposed TCAV which quantitatively tests the contribution of user-defined concepts. The difference between these approaches and ours is that these approaches focus on explaining prediction on one particular input, whereas we aim to provide a way of measuring neural networks' decision explainability as a whole.

Studies on the global interpretability of neural networks aim to come up with models that are simple enough to be human-understandable and yet expressive enough to predict the predictions of the neural network (at least in most of the cases). Such candidates include regression models [133], decision trees [93], decision rules [87]. In [87], Lakkaraju *et al.* proposed interpretable decision sets, i.e., a framework for building high-accurate predictive models, yet also interpretable. They capture the interpretability of a decision set by defining four natural metrics: size, length, cover and overlap. They also set metrics to measure the accuracy of each rule. While these approaches share the same idea of using simple models to mimic neural networks, they do not provide ways of testing and measuring the degree of decision explainability.

This work is also related to work on testing neural networks. Unlike traditional software systems that have clear and controllable logic, the lack of interpretability of neural networks makes system testing difficult. In [119], Pei *et al.* introduced the concept of neuron coverage for measuring testing coverage of a neural network. They considered a neuron to be activated if its output is higher than a threshold value and unactivated otherwise. They generated neurons' activation status and a set of test inputs based on the number of neurons activated by the inputs and propose a number of alliterative coverage metrics. In terms of fairness testing, Kusner *et al.* introduced a causal approach to address fairness. They leveraged the causal framework to model the relationship between protected attributes [84]. This work can work as an assisting tool to fairness testing. It can analyze whether the model's decisions are relevant to sensitive features. If the explainable decision tree with high fidelity contains propositions with certain sensitive attributes, the model is likely unfair. Once this work helps understanding predictions and behaviors of neural networks, it might help with further improvement as well. To the best of our knowledge, this is the first approach on testing neural networks' decision explainability.

3.6 Summary

In this work, we propose multiple definitions of neural network interpretability and develop algorithms to systematically test the decision explainability of neural networks. We define decision explainability based on measuring its fidelity against decision trees with a height limit. We remark that this is an initial attempt at testing model decision explainability and there is much to be done in the future.

Chapter 4

Subtle Group Discrimination Identification

4.1 Introduction

Machine learning models, especially neural networks, are becoming ubiquitous in various real-life applications. For example, they are used in medical diagnosis [82], self-driving cars [18] and criminal sentencing [10]. Meanwhile, discrimination has been discovered in many applications [52, 130, 150, 165]. For instance, machine learning models were used to predict recidivism risk for suspected criminals by computing the likelihood of committing a future crime [10]. Analysis results show that the prediction model was more likely to mislabel black defendants as high recidivism risk and mislabel white defendants as low risk. To minimize such ethical risks, it is crucial to systematically test the fairness of machine learning models, especially neural networks where such issues are typically 'hidden' due to the lack of interpretability [119, 147].

In this work, we aim to develop a tool to systematically test the given machine learning models with regards to fairness and reveal the discrimination with explainable way. Specifically, we develop an effective tool to systematically *test* a given machine learning model against such hidden *s*ubtle *g*roup

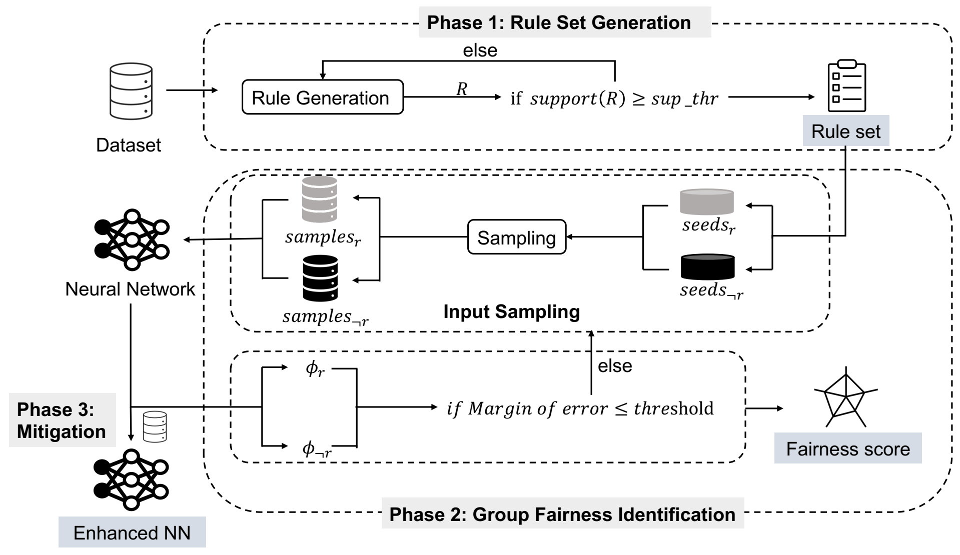


Figure 4.1: An Overview of TESTSGD.

*d*iscrimination, namely TESTSGD. Here, we only consider group fairness which is also know as group discrimination. *Group discrimination*, which characterizes a model's discrimination against a certain group (whose sensitive features[1] satisfy certain conditions) has been widely studied [47, 165, 148, 81]. However, compared to testing of individual discrimination which focuses on individual discriminatory instances, testing a machine learning model against group discrimination imposes new challenges. First, it is highly non-trivial to effectively enumerate all combinations of sensitive features (especially when the sensitive features have multiple or even continuous values). Second, *group discrimination can be hidden, i.e., there might be 'subtle' group discrimination against those groups whose sensitive features satisfy certain unknown conditions, e.g., male-white of certain age group*. While a prior work [76] similarly addresses discrimination against subgroups defined over conjunctions of protected features in the learning phase, we propose an automatic testing approach to systematically identify such subgroups using interpretable rules and measure such discrimination before model deployment.

An overview of TESTSGD is shown in Figure 4.1, which consists of three main phases: 1) candidate rule set generation, 2) group fairness identification,

[1]we use "feature"/"attribute" interchangeably

and 3) discrimination mitigation. In the first phase, TESTSGD will automatically generate a candidate set of rules concerning multiple sensitive features. Building on the insights gained from the interpretability testing work as shown in Chapter 3, we aim to explain the conditions of group discrimination in a human-understandable way. So we utilize rule sets to express the hidden group discrimination here. Note that we only consider frequent rule set with sufficient support (which characterize a sufficiently large group). In the second phase, the rule set R effectively partitions the samples into two groups, i.e., $samples_r$ which satisfies the rules and $samples_{\neg r}$ which does not. The key intuition behind is to develop effective criteria to automatically mine interpretable rules which are practical and relevant in the real-world applications. Then we measure if the model suffers from group discrimination (against the groups partitioned by the rule set) by measuring the group fairness score. Note that, solely relying on the training samples might not be enough to accurately measure such a score. We thus propose to apply a standard data augmentation method, i.e., imposing minor perturbation on the available seed samples to generate new samples, and obtain an accurate estimation of the group fairness score (with bounded errors). The testing results of the first two phases are thus the identified subtle group discrimination (characterized by the rules) and their corresponding group fairness score (with bounded errors). For example, we test the model trained on the Crime [125] dataset which predicts whether the violent crimes per population in a specific community is high. The interpretable rule set found by TESTSGD shows that it discriminates against communities in which the percentage of Caucasian population is lower than 80% and the percentage of females who are divorced is higher than 40%, with a 60.7% group fairness score, i.e., it is 60.7% more likely to predict high crime rate for such a community. In the last phase (optional depending whether the identified discrimination is considered to be harmful), TESTSGD leverages the testing results to mitigate the identified subtle group discrimination. That is, to improve group fairness, we generate new samples according to the condition under which discrimination exists and retrain the original model.

TESTSGD is implemented as an open-source software [175]. We evaluate our TESTSGD on 8 models trained on widely adopted datasets including both structured data and text data. The experimental results show TESTSGD is effective in identifying and measuring subtle group discrimination. The results also show that *subtle group discrimination does exist in all of these 8 models and sometimes to a surprising level which has never been revealed before*. For instance, the model trained on the COMPAS [10] dataset is much less likely to predict Hispanic males older than 40 years old as criminals with high recidivism risk. Furthermore, our experiments show that the testing-guided discrimination mitigation is useful. That is, we can mitigate identified subtle group discrimination for all models without sacrificing the accuracy.

In a nutshell, we summarize our main contributions as follows.

- We propose a method to automatically generate an interpretable rule set to identify subtle group discrimination in neural networks, applicable for both structured and text data;
- We develop a theoretical bound for accurately sampling and estimating the group fairness score against two groups.
- We show that we can generate samples systematically based on the interpretable rule set to mitigate subtle group discrimination.

This work is based on the published paper [175]. The remaining sections of this work are organized as follows. In Section 4.2, we provide the input type definition, fairness definitions and define our research problem. In Section 4.3, we describe the details of our approach. In Section 4.4, we show our experiment setting and evaluation results. Lastly, we show related works in Section 4.5 and conclude in Section 4.6.

4.2 Background

Our goal is to develop a black-box method to identify subtle group discrimination in a user-provided neural network model. Our method supports neural networks trained on two different kinds of data, i.e., structure data and text data. Our method does not require the inner details of the neural network. That is, the neural network is viewed as a function $M : R^p \to R^q$ which maps an input $x \in R^p$ to an output $y \in R^q$. Furthermore, we focus on deep feed-forward neural networks and recurrent neural networks.

4.2.1 Input Type

First of all, we define two different data, i.e., structure data, text data, and their corresponding sensitive features which are used to evaluate the discrimination of the neural networks.

A sample of structured dataset is composed of a set of features, i.e., a feature vector. A feature can be categorical (i.e., with a fixed set of values) or continuous (i.e., with a certain value range). We define the structure data and the corresponding sensitive features as follows.

Definition 4 (Structured Data) *A structured data x contains N features* $\{x_1, x_2, \cdots, x_N\}$*, where* $\forall x_i, x_i \in L_i$*, where* L_i *is a set of feature values. We write* $S = \{s_1, s_2, \cdots, s_n\}$ *to denote the set of sensitive features in x, where* $n < N$*.*

The text data is composed of a set of tokens. We define the sensitive feature of text data based on the presence of sensitive terms. Note that there could be different categories of sensitive terms, e.g., terms referring to race, religion, or ethnicity. We define the text data and the corresponding sensitive features as follows.

Definition 5 (Text Data) *A text data x contains a sequence of tokens* $\{x_1, x_2, \cdots, x_N\}$*. We write* $S = \{s_1, s_2, \cdots, s_n\}$ *to denote a set of categories of sensitive* *terms, where $n < N$ and T to denote a set of sensitive terms $\{t_1, t_2, \cdots, t_k\}$, where $t_j \in s_i$ for some i, for all $j \in [1, k]$ and $t_j \in x$.*

4.2.2 Group Fairness Definitions

To define our problem, we define fairness and the concept of group fairness score. There are multiple definitions of fairness [41, 69, 25, 47, 81, 165]. In this work, we focus on group fairness for its relevance in many network applications.

Group fairness, also known as statistical fairness, focuses on sensitive groups such as ethnic minority and the parity across different groups based on some statistical measurements [47, 81, 16, 165, 148]. A classifier satisfies this metric if the samples in the sensitive group have a positive classification probability or true positive probability that is similar with or equal to that of the insensitive group.

In the following, we provide a formal definition of group fairness based on positive classification rate measurement.

Definition 6 *Let M be a neural network model; l be a (favorable) prediction; and ξ be a positive constant. Let G be a group identified by certain condition ϕ on sensitive features S. G can be defined as a set of samples $\{x | x \vDash \phi\}$, where $x \vDash \phi$ means x satisfies condition ϕ. We say M satisfies group fairness, with respect to ξ and G, if and only if*

$$|\, P(M(x) = l \,|\, x \in G) - P(M(x) = l \,|\, x \notin G)\,| \leq \xi \tag{4.1}$$

Note that, in some cases, the model may be fair overall but unfair under some specific 'subtle' conditions. For example, the model is fair considering gender attribute if it approves half of the loans from female or male applicants. However, when we consider both gender and race, the model may show discrimination. For example, it approves loans for far less a percentage of Hispanic female individual, compared to the remaining group. In this setting, we say that the model discriminates against Hispanic females (if we show that the testing results have sufficient statistical confidence).

4.2.3 Problem Definition

Our problem is to develop a systematic method for identifying subtle group discrimination. That is, given a neural network model M (as well as a constant threshold ξ, we aim to generate a condition ϕ such that M is unfair with respect to the group identified by ϕ. The condition ϕ must satisfy the following conditions. (1) It must be constituted by variables representing sensitive features. (2) It must be human-interpretable, so that our analysis result can be presented for human decision making. (3) It must identify a group of non-trivial size. In addition, our method must support both structured data as well as text data. Furthermore, we would like our method to generate results with certain correctness guarantee, e.g., the chance of reporting non-existing discrimination is low.

Inspired by rule-based models, which are widely used to learn interpretable models [87, 126], we generate ϕ in the form of rules (a.k.a. constraints) which are understandable by human beings and also concrete enough to show model prediction differences between different groups. The rules are constituted by the input features, without relying on any latent variables or representations. We define ϕ to be the conjunction of one or more rules, each of which is constituted by only one sensitive feature. Furthermore, to limit the search space as well as to make sure the generated rules are interpretable, we limit each rule on continuous features to be of the form of a linear inequality, e.g., $age \geq 30$ is a possible rule but $age\ is\ multiples\ of\ 7$ is not.

In order to make sure that the discrimination that we discover is highly likely in the actual system, we propose a sampling based approach to estimate the probability of predicting certain label within a given group. Such a method allows us to generate an estimation with certain level of statistical confidence, i.e., with a bound on the error. Note that it is not straightforward to adopt existing techniques such as hypothesis testing [154, 155]. This is because the group fairness score is the difference between two estimations (i.e., one for the individual in the group and the other for those not in the group). We solve this problem by establishing a conservative error bound on the difference based on the error bounds for the two estimations.

4.3 Methodology

In this section, we describe the details of our approach. There are two main steps, i.e., learning a rule set and identifying group discrimination based on the learned rule set. The inputs for our method include a machine learning model M, its training set D, and a set of sensitive features S. The output is the subtle group discrimination represented as a rule set characterizing the discriminated group and the corresponding group fairness score.

4.3.1 Generating Frequent Rule Sets

To identify discrimination against certain group, we first need a way of characterizing a group. In this work, we characterize the groups based on a set of rules, each of which constrains one sensitive feature. In the following, we present how we generate rules for sensitive features of both structured data and text data.

In terms of categorical features x_i in structured data such as gender or race, in general, a rule can be defined as a subset of the possible feature values L_i. For instance, given the sensitive feature of race which has five values, i.e., Caucasian, Black, Hispanic, Asian and other-race, a rule can be any set containing one to four of these five values. In total, we have 30 rules. For continuous features x_i in structure data such as age or percentage, there may be too many possible values to enumerate, i.e., the domain of L_i is too large. Thus we apply techniques such as binning to turn continuous features into categorical features. Here, we divide the original value range into K intervals with equal width. Then

Table 4.1: Identity Sensitive Terms

Sensitive Features	Identity Terms
gender	lesbian, gay, bisexual, transgender, trans, queer, lgbt, lgbtq, homosexual, straight, heterosexual, male, female, nonbinary
race	african, african american, black, white, european, hispanic, latino, latina, latinx, mexican, canadian, american, asian, indian, middle eastern, chinese, japanese
religion	christian, muslim, jewish, buddhist, catholic, protestant, sikh, taoist, atheist
age	old, older, young, younger, teenage, millennial, middle aged, elderly

we consider each interval as a single value and consider a set containing adjacent values as a rule. We set K as 10 in our experiments. For example, we divide age attribute ranging from 0 to 100 into 10 equal intervals.

For textual dataset, defining rules is not that straightforward. In this work, we define the rules based on the presence of sensitive terms T (refer to Definition 5). For each sensitive category s, we define a rule which intuitively means that the text sample contains a term t, where $t \in s$. In this work, we use a set of 48 terms created in [38] as the sensitive terms which can be classified into 4 categories, i.e., gender, race, religion and age. The sensitive terms are shown in Table 4.1. For example, when we consider the *gender* feature for text dataset, there are 14 sensitive terms and thus 14 rules are generated. Notably, for simplicity, we consider both gender related terms such as 'male' and sexuality related terms such as 'lesbian' in one category and named it as 'gender'.

Once a set of rules are identified, we then characterize a group based on a rule set. Each element of a rule set is a rule constraining one sensitive feature. Intuitively, a rule set partitions the input space of M into two disjoint groups, i.e., those who satisfy all the rules in the rule set and the rest. If these two groups have a significant different probability of being predicted favorably by the model M, we successfully identify a subtle discrimination.

Note that a naive approach is to enumerate all possible rules based on one sensitive feature and combine them arbitrarily. Such an approach is both infeasible and undesirable. First, there can be enormous combinations of the rules. Second, not all rule sets are interesting. For instance, a rule set may be $\{age \geq 100, gender = Male\}$. A discrimination found against the group identified by this rule set is likely to be due to the limited data. Furthermore, the discrimination is perhaps not as concerning as discrimination against groups that represent a sizable population.

We thus only consider frequent rule sets among all possible combinations of rules. A frequent rule set is a set of rules that are satisfied by a group with a size more than certain threshold. Formally, given a rule set R, the *support* for R is the frequency of the number of samples that satisfy all rules in rule set R. Given a *support threshold* θ (i.e., a percentage), we say that R is *frequent* if its *support* is no less than θ. In the following, we present how to identify a set of frequent rule sets for structured and text data.

For each sensitive feature s, let R^s be the set of rules concerning s. A rule set R is composed of rules for each sensitive feature, i.e., $R = \{r^{s_1}, r^{s_2}, ..., r^{s_n}\}$ where $r^{s_i} \in \{R^{s_i} \cup \varnothing\}$ and $R \neq \varnothing$. R is frequent if and only if $support(R) \geq \theta$ where $support(R)$ is defined as follows.

$$\text{support}(R) = \frac{\#\{d \in D | \forall r \in R.\ d \vDash r\}}{\#D} \tag{4.2}$$

where $\#X$ of a set X is the number of elements in X; and $d \vDash r$ means that d satisfies r.

Example 4.3.1 *Consider the structured dataset* ***Adult Income*** *[128]. It has three sensitive features, i.e., gender, race, and age. Each feature has a set of values. The following constitutes a rule set*

$$\{gender = Male, race = White, 40 \leq age < 60\}$$

Rule sets for text data are defined differently. Recall that each rule is a proposition on whether the text contains certain sensitive term. Formally, given the set of the categories of sensitive terms S, a rule set R is then a set of sensitive terms $\{r_1, r_2, \cdots, r_m\}$, where $r_k \in s_i$ for some i, for all $k \in [1, m]$ and $m \leq n$. The support of R is defined as follows.

$$\text{support}(R) = \frac{\#\{d \in D | \forall r \in R.\ contains(d, s_r)\}}{\#D} \tag{4.3}$$

where s_r is the sensitive category referring to r and $contains(d, s_r)$ is a proposition which is true if and only if d contains at least one term in the category s_r.

Example 4.3.2 *Consider the text dataset Wikipedia Talk Pages [159]. We have two categories of sensitive terms, e.g., gender and race. For each category, we have a set of corresponding sensitive terms as shown in Table 4.1. The following constitutes a rule set*

$$\{\text{“}bisexual\text{”}, \text{“}Caucasian\text{”}\}$$

Algorithm 8 shows the exact steps in generating all possible rule sets. At line 1, we first initialize a dictionary single_rules and an empty set rules_sets. During the loop from line 2 to 5, we generate all possible 1-feature rules for each sensitive feature as discussed above. At line 6, we generate a set of all 1-feature rules. Then, we generate all possible rule sets at line 7. Lastly, at line 8, we only keep those rule sets that have a support value no less than θ.

In general, given a dataset with K sensitive features, and at most N rules for each sensitive feature, the number of rule sets is N^K in the worse case. For example, we have 2 gender-related single rules, 5 race-related 1-feature rules and 10 age-related 1-feature rules, there are 17 rule sets when considering one sensitive attribute, 80 rule sets when considering two sensitive attributes and

Algorithm 8 *FrequentRuleSets*(D,S,sup_thr) where D is the training set, S is the sensitive attributes, θ is the support threshold

```
1: $single\_rules \leftarrow \{\}$, $rule\_sets \leftarrow \varnothing$
2: for each $s$ in $S$ do
3:     $rules \leftarrow \{r_1, r_2, ...\}$
4:     $single\_rule[s] = rules$
5: end for
6: $all\_single\_rules \leftarrow \{single\_rule[s] \cup \varnothing\}$ for all $s \in S$
7: $rule\_sets \leftarrow combinations(all\_single\_rules)$
8: $all\_rule\_sets \leftarrow \{R\ for\ R\ in\ rule\_sets\ if\ support(D,R) \geq \theta\}$
9: return $all\_rule\_sets$
```

100 rule sets when considering all sensitive attributes. So in total, there are 197 possible rule sets.

4.3.2 Identifying Group Fairness

For each group identified by a rule set, we then measure the discrimination against the group. That is, we aim to compute the probability of predicting certain label by M on those samples in the group, and that on those samples not in the group, and measure the difference. The score is the group fairness score, which varies from 0 (i.e., no difference) to 1 (i.e., completely different). Formally,

Definition 7 (Group fairness score) *Let R be a rule set. Let l be a (favorable) label. The group fairness score with respect to R and l is $|prob(R,l) - prob(\neg R,l)|$, where $prob(R,l)$ is the probability of predicting l given samples satisfying R, $\neg R$ identifies samples not satisfying R.*

We remark that this definition is similar to the CV score [25] and multivariate group discrimination score [47]. However, the former is limited to binary input types and the latter is limited to categorical input types. In comparison, our fairness score supports both structured data and text data.

Example 4.3.3 *Take a model trained on the (structured) Adult Income dataset as an example. The model predicts whether the income of an adult is above*

$50,000 annually, i.e., "True" means above the threshold and "False" means otherwise. Assume the rule set is

$$\{gender = Male, race = White, 40 \leq age < 60\}$$

Assume that the model predicts 28% of individuals in this group with "True", and 10% of the remaining population with "True". The model's group fairness score, with respect to the rule set and the prediction, is 18%.

Given a rule set, measuring the group fairness score requires us to measure $prob(R,l)$ and $prob(\neg R,l)$, which is non-trivial since exhaustively enumerating all samples is infeasible due to the enormous input space. On the other hand, measuring it based on a limited number of samples may yield inaccurate results. In the following, we propose an approach to compute group fairness scores with a statistical confidence guarantee. Formally, we would like to measure the group fairness score f within a margin of error ε under a certain confidence δ, such that $prob(|f - \hat{f}| > \varepsilon) < 1 - \delta$, where $\hat{f}$ is the real group fairness score over all possible samples.

Algorithm 9 shows how we measure the group fairness score. We maintain two sets of samples, i.e., $samples_r$ which contains samples satisfying R and $samples_{\neg r}$ which contains samples not satisfying R. At line 1, we set both $samples_r$ and $samples_{\neg r}$ to be empty, error margin ε to be infinity and the number of generated samples as 0. During the loop from line 2 to 16, we keep generating samples and calculating group fairness score until the error margin ε is no more than the given error threshold $error\_thr$. From line 3 to line 6, we generate new samples for $samples_r$ and $samples_{\neg r}$ respectively using a function $Sample$. We remark that the generated samples should follow the original data distribution (i.e., that of the training dataset). We present details on how we sample on structured and text dataset in the next subsection.

At line 7, we increase num by 1. After generating a sufficient number of

Algorithm 9 *GroupFairnessScore*$(D, M, R, sample\_thr, error\_thr)$ where D is the training dataset; M is the machine learning model; R is a rule set, *sample_thr* is the number of generated inputs threshold; *error_thr* is error margin threshold

```
1: $samples_r \leftarrow \varnothing$, $samples_{\neg r} \leftarrow \varnothing$, $\varepsilon \leftarrow +\infty$, $num \leftarrow 0$
2: while $\varepsilon > error\_thr$ do
3:     $x \leftarrow Sample(D, R)$
4:     $x' \leftarrow Sample(D, \neg R)$
5:     $samples_r \leftarrow samples_r \cup x$
6:     $samples_{\neg r} \leftarrow samples_{\neg r} \cup x'$
7:     $num \leftarrow num + 1$
8:     if $num > sample\_thr$ then
9:         $\phi_r \leftarrow \#\{i \in samples_r | M(i) = l\}/num$
10:        $\phi_{\neg r} \leftarrow \#\{i \in samples_{\neg r} | M(i) = l\}/num$
11:        $\varepsilon \leftarrow z \times \sqrt{\frac{\phi_r(1-\phi_r)}{num}} + z \times \sqrt{\frac{\phi_{\neg r}(1-\phi_{\neg r})}{num}}$
12:        if $\varepsilon \leq error\_thr$ then
13:            break
14:        end if
15:    end if
16: end while
17: return $f \leftarrow |\phi_r - \phi_{\neg r}|$
```

samples, we check the error margin ε from line 9 to 15. We first calculate the probability of predicting l at line 9 and 10 for two sets of samples. Then at line 11, we calculate the error margin ε on the group fairness score. We explain why it is calculated this way below. If ε is less than or equal to *error_thr*, the stopping criteria is satisfied (as in line 12 and 13). Lastly, at line 17, we return the absolute difference between ϕ_r and $\phi_{\neg r}$ as the group fairness score.

In the above algorithm, we estimate the error margin of the group fairness score based on an estimation of $prob(R, l)$ and $prob(\neg R, l)$. The complication is that both $prob(R, l)$ and $prob(\neg R, l)$ carry certain error margin, which may magnify the error margin for the group fairness score. In the following, we prove that line 11 in the above algorithm allows us to conservatively estimate the error margin of the group fairness score.

Theorem 4 *Assume that ϕ_r satisfies the following*

$$prob(|\phi_r - \hat{\phi}_r| > \varepsilon_r) < 1 - \delta_r \tag{4.4}$$

where ε_r and δ_r are constants. Similarly, $\phi_{\neg r}$ satisfies the following.

$$prob(|\phi_{\neg r} - \hat{\phi}_{\neg r}| > \varepsilon_{\neg r}) < 1 - \delta_{\neg r} \tag{4.5}$$

Then the following is satisfied.

$$prob(|f - \hat{f}| > \varepsilon_r + \varepsilon_{\neg r}) < 1 - \delta_r \delta_{\neg r} \tag{4.6}$$

Proof: *Since $prob(|\phi_r - \hat{\phi}_r| > \varepsilon_r) < 1 - \delta_r$ and $prob(|\phi_{\neg r} - \hat{\phi_{\neg r}}| > \varepsilon_{\neg r}) < 1 - \delta_{\neg r}$, we have:*

$$prob(|\phi_r - \hat{\phi}_r| \le \varepsilon_r) \ge \delta_r$$

$$prob(|\phi_{\neg r} - \hat{\phi}_{\neg r}| \le \varepsilon_{\neg r}) \ge \delta_{\neg r}$$

Hence

$$prob(|(\phi_r - \hat{\phi}_r) - (\phi_{\neg r} - \hat{\phi}_{\neg r})| \le \varepsilon_r + \varepsilon_{\neg r}) \ge$$

$$prob(|\phi_r - \hat{\phi}_r| \le \varepsilon_r) \cdot prob(|\phi_{\neg r} - \hat{\phi}_{\neg r}| \le \varepsilon_{\neg r}) \ge \delta_r \delta_{\neg r}$$

and

$$prob(|(\phi_r - \hat{\phi}_r) - (\phi_{\neg r} - \hat{\phi_{\neg r}})| > \varepsilon_r + \varepsilon_{\neg r}) < 1 - \delta_r \delta_{\neg r}$$

$$prob(|(\phi_r - \phi_{\neg r}) - (\hat{\phi}_r - \hat{\phi}_{\neg r})| > \varepsilon_r + \varepsilon_{\neg r}) < 1 - \delta_r \delta_{\neg r}$$

According to Definition 7, group fairness score $f = \phi_r - \phi_{\neg r}$. Thus

$$prob(|f - \hat{f}| > \varepsilon_r + \varepsilon_{\neg r}) < 1 - \delta_r \delta_{\neg r}$$

The above theorem provides a theoretical guarantee on the statistical confidence for the group fairness score estimation. That is, based on the Equation 4.6, the fairness level for fairness score f is $\delta_r \delta_{\neg r}$ and the margin of error is the sum of two margin of errors as $\varepsilon_r + \varepsilon_{\neg r}$. Each ε is calculated by:

$$\varepsilon = z \times \sqrt{\frac{\phi(1-\phi)}{num}} \tag{4.7}$$

where z is the value from the standard normal distribution for a certain confidence level δ (e.g., for a 95% confidence level, $z = 1.96$). So the final margin of error for fairness score f is shown in line 11 of Algorithm 9. Based on the result, we derive the stopping criteria, as shown in line 12 and 13 of Algorithm 9.

The above shows how we compute the group fairness score for one rule set. Given multiple rule sets, we systematically compute the fairness score for each rule set with Algorithm 9, and then rank the rule sets according to the resultant group fairness score. If the group fairness score of certain rule set is more than a given tolerable threshold, we report that discrimination is identified.

Example 4.3.4 *Take a model trained on the (structured) Adult Income dataset as an example. We fixed the confidence level to 95% and the corresponding z-value is 1.96. We set the sampling threshold sample_thr as 1000 and the error of margin threshold error_thr as 0.05. We are given a rule set*

$$\{gender = Male, race = White, 40 \leq age < 60\}$$

First, we sample 1000 inputs as $samples_r$ using Sample function that represents white males who are older than 40 but younger than 60. Then we sample another 1000 inputs as $samples_{\neg r}$ using Sample function that represents the rest individuals. We observe that 283 samples in $samples_r$ are labeled as "True", while only 91 samples in $samples_{\neg r}$ are labeled as "True". So ϕ_r is 28.3% and $\phi_{\neg r}$ is 9.1%. According to Algorithm 9, ε_r is 0.028 and $\varepsilon_{\neg r}$ is 0.018. So the margin of error ε for fairness score is 0.046. Since ε is less than 0.05, we stop sampling. Finally, the group fairness score is computed as 19.2% with 90.25% confidence.

4.3.3 Input Sampling

As discussed above, Algorithm 9 requires us to sample inputs with a distribution which is similar to the data distribution of the training dataset. As shown in [54], modern machine learning models mostly rely on the i.i.d. assumptions. That is, the training and test set are assumed to be generated independently from an identical distribution. It is more likely for machine learning models to predict identically distributed data correctly.

While it is impossible to know the actual data distribution, we aim to generate additional samples from a distribution as close as possible to the distribution of the training set. For structured data, instead of generating feature vectors randomly, we generate new samples by adding tiny perturbations on original samples uniformly. The perturbation is added to one randomly selected non-sensitive attribute with randomly selected perturbation direction and the perturbation size is 1 for integer variables or 0.01 for decimal variables. Formally, given the rule set R, we first search a seed instance from the dataset D as $seed = \{x_1, x_2, \cdots, x_N\}$, where $\forall r \in R.\ seed \vDash r$. Then we randomly select a non-sensitive attribute x_k, where $x_k \notin S$. We perturb x_k as $x'_k = x_k + dir \cdot s\_pert$, where $dir \in [-1, 1]$ and s_pert is the perturbation size.

For text data, we generate new samples by replacing sensitive terms with a different term in the same sensitive term category. For example, when we test the machine learning model trained on the Wikipedia Talk Pages dataset, given a rule set {"*gay*"}, we need to generate additional comments containing the term "gay". First, we search all comments containing gender-related sensitive terms such as "lesbian" and "bisexual", as defined in Table 4.1. Then we replace these terms in the original comments with the term "gay" to generate new comments. That is, we can generate "I am a gay" from an original comment "I am a lesbian". The reason why we use text replacement instead of text perturbation, as in the case of structured data, is that perturbing texts with synonyms (as proposed in [132] for adversarial attacks) is ineffective to generate the texts in the desired group. Our text generation method also has the benefit of mitigating the influence of data imbalance which may cause unintended bias [38]. Formally, given the rule set $R = \{r_1, r_2, \cdots, r_m\}$, we first search a seed instance from the dataset D as $seed = \{x_1, x_2, \cdots, x_N\}$, where $\forall r \in R.\ contains(seed, s_r)$, where s_r is the sensitive category referring to r and $contains(d, s_r)$ is a proposition which is true if and only if d contains at least one term in the category s_r. Then we replace the term x_i to term r_j, for all $r_j \in R$ and $x_i \in s_{r_j}$.

4.4 Implementation and Evaluation

We have implemented TESTSGD as a self-contained software toolkit based on Tensorflow [1] with about 6K lines of Python code.

Experiment Subjects Our experiments are based on 8 models trained with the following benchmark datasets. These datasets have been widely used as evaluation subjects in multiple previous studies on fairness [176, 177, 47, 38, 130, 99]. Expect for the first three datasets which are mentioned in the previous work at Chapter 3, we consider three more tabular datasets and two more textual datasets in this work.

- Adult Income [128]: The dataset contains more than 30,000 samples and is used to predict whether the income of an adult is above \$50,000 annually. The attributes *gender*, *race* and *age* are sensitive attributes.
- Bank Marketing [110]: The dataset contains 45,000+ samples and is used to train models for predicting whether the client would subscribe a term deposit. Its sensitive attribute is *age*.
- German Credit [64]: This is a small dataset with 600 samples. The task is to assess an individual's credit. The sensitive attributes are *gender* and *age*.
- COMPAS [10]: This dataset contains 7,000+ samples. The task is to predict whether the recidivism risk score for an individual is high. The sensitive

attributes are *gender*, *race* and *age*.

- Crime [125]: This dataset contains almost 2,000 data for communities within the US. The task is to predict whether the violent crimes per population in a specific community is high. Since this dataset records population statistics, their sensitive features are shown in multiple attributes with percentage values. Here, we extract all gender/race/age related attributes to learn rule sets.
- Law School [11]: This dataset has more than 20,000 application records and is used to predict whether a student passes the bar exam. The attributes, *race* and *gender* are sensitive attributes.
- Wiki Talk Pages [159]: This is a textual dataset containing more than 100,000 Wikipedia TalkPage comments. The task is to predict whether a given comment is toxic.
- IMDB [100]: IMDB dataset contains 50,000 movie reviews. The task is to predict whether a given sentence is a positive review.

For the first six structured datasets, we train a six-layer feed-forward neural network using the exact same configuration as reported in the previous studies [176, 177]. For the last two textual datasets, we train a convolutional neural network (CNN) combined with long short-term memory (LSTM). The details of trained models are shown in Table 4.3. The accuracy of the trained models is expectedly similar to what is reported in the previous studies. Table 4.2 shows the value of parameters used in our experiment to run TESTSGD. All experiments are conducted on a server running Ubuntu 1804 with 1 Intel Core 3.10GHz CPU, 32GB memory and 2 NVIDIA GV102 GPU. To mitigate the effect of randomness, all the results are the average of 3 runs.

We aim to answer multiple research questions as follows.

RQ1: Is our method effective in identifying subtle group discrimination of a given machine learning model? To answer the question, we systematically ap-

Table 4.2: Parameters of the Experiments

Parameters	Value	Discription
θ	5%	support threshold
sample_thr	1000	sampling threshold
δ	95%	confidence level
error_thr	0.05	error margin threshold
z	1.96	z value
s_pert	1	perturbation size for integer variables
s_pert	0.01	perturbation size for decimal variables

Table 4.3: Dataset and Models of Experiments

Dataset	Model	Accuracy
Adult Income	Six-layer Fully-connected NN	86.13%
Bank Marketing	Six-layer Fully-connected NN	91.62%
German Credit	Six-layer Fully-connected NN	100%
COMPAS	Six-layer Fully-connected NN	78.99%
Crime	Six-layer Fully-connected NN	92.52%
Law School	Six-layer Fully-connected NN	95.19%
Wikipedia Talk Pages	CNN Long Short-term memory network	93.89%
IMDB	CNN Long Short-term memory network	86.68%

ply our approach to the above-mentioned models and measure the results. The results are summarized in Table 4.4. It shows results on the six models trained on structured data and results on the two models trained on text data. These four columns show datasets, rule sets, group fairness scores and model accuracies respectively. The favorable label is "True", the meaning of which can be found in the above introduction on the corresponding dataset. Note that for each model, we rank the identified subtle discrimination according to the group fairness score and we report the top 3 worst discrimination only.

We can observe that subtle discrimination does exist in these models, which were never revealed in the previous studies [176, 177, 47, 38, 99]. For instance, the model trained on the Bank Marketing dataset predicts only 3.3% of the clients who are older than 10 but younger than 90 would subscribe a term deposit, whilst predicting 41.5% of clients older than 90 would subscribe a term deposit. All of the top 3 testing results all show the model discriminates against young clients. We remark that although this is unfair according to the definition,

Table 4.4: Rule Sets and Fairness Scores for Neural Networks

Dataset	top 1		top 2		top 3	
	Rule Set	**Fairness Score** $(\phi_r, \phi_{\neg r})$	**Rule Set**	**Fairness Score** $(\phi_r, \phi_{\neg r})$	**Rule Set**	**Fairness Score** $(\phi_r, \phi_{\neg r})$
Adult Income	gender=male, 40≤age<80, race=White or Asian-Pac-Islander	20.2% (29.9%, 9.7%)	gender=male, 40≤age<70, race=White or Amer-Indian-Eskimo	19.4% (28.9%, 9.5%)	gender=male, 40≤age<80, race=White, Asian-Pac-Islander or Amer-Indian-Eskimo	18.4% (26.9%, 8.5%)
Bank Marketing	10 ≤ age <90	38.2% (3.3%, 41.5%)	10 ≤ age <70	22.8% (26.6%, 3.8%)	10 ≤ age <60	20.5% (4.7%, 25.2%)
German Credit	gendre = female, 60≤age<70	21.9% (72.25%, 50.63%)	gender = female, 60≤age<80	21.8% (70.5%, 48.7%)	gender = male, 40≤age<80	15.5% (52.6%, 47.1%)
COMPAS	gender = male, age≥40, race = Hispanic or other race	62.4% (20.7%, 83.1%)	gender = male, 40≤age<60, race = Hispanic or other race	62.3% (20.3%, 82.6%)	gender = male, 50≤age<60, race = Hispanic	62.3% (19.5%, 81.8%)
Law School	gender = male, race = Black	13.4% (86.3%, 99.7%)	gender = male, race = Asian or Black	11.5% (88.4%, 99.9%)	gender = female, race = Asian or Black	10.5% (89.4%, 99.9%)
Crime	pct(white)≤0.8, pct(female divorce)≥0.4	60.7% (83.8%, 23.2%)	pct(white)≤0.8, pct(female divorce)≥0.5	59.6% (87.0%, 27.4%)	pct(white)≤0.6, pct(female divorce)≥0.5	59.5% (94.6%, 35.1%)
Wiki Talk Pages	"gay", "taoist"	6.5% (13.0%, 6.5)	"gay", "protestant"	5.4% (12.9%, 7.5%)	"gay", "african american"	5.1% (12.5%, 7.4%)
IMDB	"european", "yong"	6.6% (56.0%, 49.4%)	"white", "older"	6.6% (59.1%, 52.6%)	"lgbtq"	6.5% (7.5%, 14.0%)

it may have its underlying reasons and it is still up to human experts to decide whether it is actual discrimination.

For the models trained on the Adult Income dataset, German Credit dataset and the Law School dataset, they show relatively mild discrimination. In contrast, the model trained on the COMPAS dataset shows severe discrimination, with a fairness score of 62.4%. That is, for Hispanic or other race male individuals who are older than 40, the model is much less likely to predict the recidivism risk as high. For the remaining individuals, the model predicts 83.1% of them have high recidivism risk. Top 2 and top 3 test results also show severe discrimination against older Hispanic or other race male individuals. Similarly, the model trained on the Crime dataset also shows high discrimination. Different from the first five structured datasets, samples in this dataset have 10 different sensitive features, each of which is a decimal ranging from 0.0 to 1.0 representing the percentage of certain population. As shown in the top 1 testing result, when the percentage of divorced females is above 40% and the percentage of Caucasian is below 80%, the model is much more likely to predict that the violent crimes per population in this community is high. All testing results on the model trained on Crime dataset suggest that the model discriminates against communities with high percentage of divorced females and low percentage of Caucasian.

In Table 4.4, the last two rows show the results on models trained on the text data. In general, we observe that the models trained on text dataset show less discrimination. The maximum fairness score for the model trained on the

Wikipedia Talk Pages dataset is 6.5%. That is, the model predicts 13.0% of comments containing both "gay" and "taoist" as toxic. For other comments (i.e., those without one of these two terms or both), the model predicts only 6.5% of them as toxic. Top 2 and top 3 testing results show that the model discriminates against comments containing both "gay" and "protestant" and comments containing both "gay" and "african american" respectively. The model trained on the IMDB dataset shows a similar level of discrimination. It is more likely to predict reviewers containing "european" and "young" and reviews containing "white" and "older" as positive. It also shows discrimination against reviews containing "lgbtq". Our conjecture on why the level of discrimination is considerably lower on these models is that each sample in these text datasets often has many features and as a result, the influence of each term (including sensitive terms) is distributed.

RQ2: Is our method efficient? To answer this question, we measure the amount of time required to identify the subtle discrimination for each model. The total execution time and the numbers of tested rule sets are shown in Table 4.5. For all models, the time required to identify the subtle discrimination is less than 20 hours. Furthermore, models trained on structured dataset take considerably less time than those trained on text dataset. That is, models trained on the Adult Income, Bank Marketing, German Credit, COMPAS and Law School take less than 16 minutes. One exception is the model trained on the Crime dataset that takes more than 8 hours. The main reason is that it has a large number of rule sets, due to a large number of sensitive features (i.e., 10), all of which are continuous features. In contrast, both models trained on text dataset take more than 9 hours to finish. The main reason is that generating additional samples for such dataset takes much more time in general. We remark that the sampling procedure can be easily parallelized and thus we could significantly reduce the time if it is an issue.

Note that the support threshold θ is set to be 5% in all the above experiments.

Table 4.5: Time Taken to Identify the subtle discrimination

Dataset	Time (seconds)	#rule set
Adult Income	869.35	880
Bank Marketing	141.52	34
German Credit	104.85	53
COMPAS	908.5	1590
Law School	18.46	17
Crime	29150.01	13282
Wiki Talk Pages	34982.28	732
IMDB	69125.16	876

Intuitively, it means that each rule must be relevant to 5% of the population (although the rule set, which is a conjunction of multiple rules, may impact a smaller population). This hyper-parameter largely determines how many rule sets that we must examine and thus may have an impact on the execution time. We thus conduct additional experiments with different θ values, ranging from 1% to 50%, to evaluate the effect of θ on the execution time and the results. The results on two models, i.e., the model on Law School and the model on COMPAS, are detailed in Table 4.6.

The table shows the execution time, the number of rule sets and the worst group fairness score. We can observe that, the larger a θ we set, the fewer rule sets, the less execution time and the smaller group fairness score in general. If the threshold θ is too low, e.g., 1%, we spend a lot of time on testing a huge number of rule sets, which may not be interesting (one such example is $\{gender = Male, age \geq 100\}$). In contrast, if the threshold θ is too high, e.g., 20% or 50%, there may only exists few or even none rule set (as in the case of the model trained on the COMPAS dataset).

We note that different θ may result in different discrimination being identified. For the model trained on Law School, the rule set shows that the model discriminates against black or Asian males the most when θ is 5%. However, when we set θ to be 1%, the model is shown to discriminate against black male individuals the most. For the model trained on the COMPAS dataset, the model

Table 4.6: Effect of Different θ

Dataset	θ	**Time (seconds)**	**#rule sets**	**Rule set**	**Fairness Score**
Law School	1%	46.71	59	gender=male, race=Black	14.6%
	5%	18.46	17	gender=male, race=Asian or Black	9.6%
	10%	17.83	16	gender=male, race=Asian or White	1.0%
	20%	17.83	16	gender=male, race=Asian or White	0.9%
	50%	5.27	2	gender=male, race=other race	0.3%
COMPAS	1%	1175.79	2063	gender=male, age≥40 race=Hispanic or other race	57.7%
	5%	908.5	1590	gender=male, age≥40 race=Hispanic or other race	57.7%
	10%	676.74	1180	gender=male, age≥20 race=Hispanic or other race	43.9%
	20%	0	0	NULL	NULL
	50%	0	0	NULL	NULL

discriminates against Hispanic or other race males who is older than 40 years old most when we set θ to be 5%. However, when we set θ higher (i.e., 10%), the age range is expanded to be over 20 years in the identified rule set. Such a result is expected as a large θ requires us to find discrimination against a large group. What is considered to be a reasonable value for θ is a complicated question, which should probably be answered by lawmakers.

RQ3: Can we mitigate subtle discrimination using our testing results? To further show the usefulness of our approach, we evaluate whether we can mitigate the identified subtle discrimination using our testing results. The idea is to mitigate the discrimination by retraining. We remark that there are alternative approaches for improving fairness as well [76, 131]. Note that we generate additional instances satisfying the rule set with the sampling approach described in Section 4.3.3. We only select those generated instances with the opposite label. For example, the model trained on COMPAS is more likely to predict

Table 4.7: Discrimination Mitigation for Neural Networks

Dataset	Rule Set	Before accuracy	Before Fairness Score $(\phi_r, \phi_{\neg r})$	After accuracy	After Fairness Score $(\phi_r, \phi_{\neg r})$
Adult Income	gender=male, $40 \leq age < 80$, race=White or Asian-Pac-Islander	86.1%	20.20% (29.9%, 9.7%)	86.2%	10.1% (18.9%, 8.8%)
Bank Marketing	$10 \leq$ age <90	91.6%	38.2% (3.3%, 41.5%)	90.6%	5.4% (6.9%, 12.3%)
German Credit	gendre = female, $60 \leq age < 70$	100.0%	21.9% (72.25%, 50.63%)	100.0%	7.3% (45.9%, 53.2%)
COMPAS	gender = male, age$\geq$40, race = Hispanic or other race	79.0%	62.4% (20.7%, 83.1%)	78.5%	4.2% (80.9%, 85.1%)
Law School	gender = male, race = Black	75.2%	13.36% (86.3%, 99.7%)	95.1%	7.5% (92.3%, 99.8%)
Crime	FemalePctDiv $\geq$0.4, racePctWhite$\leq$0.8	93.9%	60.7% (83.8%, 23.2%)	98.1%	51.4% (90.6%, 39.2%)
Wiki Talk Pages	"gay", "taoist"	93.9%	6.50% (13.0%, 6.5)	95.5%	0.4% (8.4%, 8.0%)
IMDB	"european", "yong"	86.7%	6.6% (56.0%, 49.4%)	84.0%	3.3% (43.7%, 40.4%)

elderly males who are Hispanic or other race with "False" label. We can use the *Sample* function to generate instances satisfying the condition that are labeled as "True" according to the original model. Afterward, we retrain the original model with these additional instances and testing the subtle discrimination with respect to the same rule set to see the improvement. Note that we gradually increase the number of additional instances from 50 to 10% of original dataset size to achieve the lowest fairness score without decreasing the accuracy of the retrained model.

We only consider the top 1 worst rule sets to mitigate the discrimination. The results are shown in Table 4.7 for six models trained on additional structured data and two models retrained on additional textual data. We can observe that all models show reduced subtle discrimination and almost the same accuracy. The fairness scores for retrained models on Adult Income, German Credit and Law School decrease by about half. For the most improvement, the model retrained on the COMPAS dataset shows much less subtle discrimination as the fairness score decreases by more than 10 times, i.e., from 57.7% to 4.2%. The fairness score of the model trained on the Crime dataset decreases from 60.7% to 51.4%. Relatively, the fairness improvement is not obvious. We believe that it is due to its many continuous sensitive features and the large number of features (i.e., each input contains more than 100 attributes). That is, it would require a lot more additional data to improve fairness. In terms of CNN models, the fairness score decreases from from 6.5% to 0.4% for the model retrained on Wikipedia Talk Pages and decreases from 6.6% to 3.3% for the model retrained on IMDB dataset.

We proposed the data-augmentation approach in this work to improve the fairness merely as an example to show the usefulness of our approach. There are also methods which target reducing algorithmic bias as well which are often referred to as in-processing methods. In face, our results can work with those approaches as well, e.g., by training with additional loss functions based on the rule sets identified using our method.

Comparison with Baselines We identify the following two baselines from literature which can potentially identify similar group discrimination as our work. **1) THEMIS [47]** calculates group discrimination scores over combinations of multiple features (subgroups) by measuring the difference between the maximum and minimum frequencies of two subgroups on randomly generated samples. Those subgroups can then be regarded as identified discrimination if the score is higher than a threshold. **2) FairFictPlay [76]** proposed an in-processing algorithm aiming to improve subgroup fairness. The subgroups are identified with user-provided constraints in the form of conjunctions of Boolean attributes, linear threshold functions, or bounded degree polynomial threshold functions over multiple protected features.

In Table 6.1, we show the identified group discrimination with TESTSGD, Themis and FairFictPlay respectively, along with the fairness scores, on the same models trained on structured data (similarly to Table 4.3). We set the timeout as 24 hours. Note that FairFictPlay uses complex linear functions on all the protected features (which are hard to interpret) to define discriminatory subgroups, thus we do not show the exact concrete linear functions in the table. We have the following observations. 1) Compared to FairFictPlay, TESTSGD identifies discrimination with higher scores (more discriminating) while being

Table 4.8: Comparisons Between TESTSGD , Themis and FairFictPlay. '-' means timeout.

Dataset	TESTSGD		Themis		FairFictPlay	
	Rule Set	Fairness Score	Sensitive Attributes' values for Max/Min Proportion	Fairness Score	Subgroup	Fairness Score
Adult Income	Gender=male, 40≤age<80, race=White or Asian-Pac_islander	20.2%	[gender=Female, 60≤age<60, race=Asian-pac_islander] - [gender=Male, 10≤age<20, race=White]	26.6%	Linear Threshold Funciton	13.9%
Bank Markating	10≤age<90	38.2%	[60≤age<70] - [10≤age<20]	8..4%	Linear Threshold Funciton	7.6%
German Credit	gender=feamle, 60≤age<70	21.9%	[gender=Female, 80≤age<90] - [gender=Male, 10≤age<20]	17.1%	Linear Threshold Funciton	7.0%
COMPAS	gender=male, age≥40, race=Hispanic or other race	62.4%	[gender=Female, 10≤age<20, race=Native American] - [gender=Male, 60≤age<70, race=other race]	67.30%	Linear Threshold Funciton	22.4%
Law School	gender=male, race=Asian or Black	11.5%	[gender=Male, race=White] - [gender=Female, race=Black]	13.5%	Linear Threshold Funciton	3.7%
Crime	FemalePctDiv≥0.4, racePctWhite≤0.8	60.7%	-	-	Linear Threshold Funciton	38.8%

interpretable. Moreover, TESTSGD automatically identifies the discriminated subgroups without any prior knowledge. 2) Similar to TESTSGD, Themis is able to identify discriminated subgroups automatically. However, Themis identifies two subgroups which are maximally different (in terms of being predicted favorably) while TESTSGD identifies subgroups which are predicted different from the rest. These two approaches thus produce results that are complementary to each other. Note that Themis does not support text data.

Threats to validity We mentioned that our error guarantees rely on the i.i.d. assumption and it is challenging and still open question how to guarantee that the assumption is satisfied in practice. In this work, we try to approximate the original dataset's distribution by adding perturbation on one attribute systematically, which is the approach adopted in [176, 150] and many other machine learning literature [132].

Our approach aims to explain why and under which conditions the discrimination shows in a human-understandable way. In comparison with baselines, **THEMIS** is not designed to test against subtle discrimination. To compare wich our approach, we further show the discrimination conditions of **THEMIS** based on the testing results. As for the baseline **FairFictPlay**, we omit the exact linear threshold functions since they are complicated.

4.5 Related Work

Many existing works attempted to test discrimination according to different fairness definitions and measurements [41, 25]. In [44], Feldman *et al.* provide a fairness definition which is measured according to demographic parity of model predictions. It measures how well the sensitive class can be predicted based on classification accuracy. In [61], Hardt *et al.* present an alternate definition of fairness based on demographic parity. It requires a decision to be independent of the sensitive attribute. In [84], Kusner *et al.* define counterfactual discrimination which focuses on single decisions towards an individual. A prediction is counterfactual fair if it is the same in the actual group and a different demographic group. In [47], Galhotra *et al.* propose causal discrimination to measure the fraction of inputs for which model causally discriminates. This definition is similar to counterfactual fairness, but it takes instances of discrimination into account. In [76], Kearns *et al.* proposed an in-processing algorithm aiming to improve the fairness of given subgroups, where subgroups are defined as conjunctions of attributes, linear threshold functions, or bounded degree polynomial threshold functions over multiple protected features. Most existing works [47, 81, 16] use positive classification rate as fairness measurement.

Subsequently, many works focus on individual discrimination to generate individual discriminatory instances [176, 177, 7, 66]. They tried to generated instances which are classified differently after changing sensitive attributes. In [7], Agarwal *et al.* present an automated testing approach to generate test inputs to find individual discrimination. In [130], Ruoss *et al.* propose a fairness representation framework to generalize individual fairness to multiple notions. It learns a mapping from similar individuals to latent representations. However, the testing on individual discrimination cannot provide a statistical measurement of fairness.

Some other existing works attempted to test model discrimination with fairness score measurements. In [148], Tramer *et al.* propose an unwarranted associations framework to detect unfair, discriminatory or offensive user treatment in data-driven applications. It identifies discrimination according to multiple metrics including the CV score, related ratio and associations between outputs and sensitive attributes. In [81], Kleinberg *et al.* also test multiple discrimination scores and compare different fairness metrics. In [47], Galhotra *et al.* propose a tool called THEMIS to measure software discrimination. It tests discrimination with two fairness definitions, i.e., group discrimination score and causal discrimination score. In [4], Adebayo *et al.* try to determine the relative significance of a model's inputs in determining the outcomes and use it to assess the discriminatory extent of the model.

Some prior work has been done on fairness for text classification tasks as well. In [17], Blodgett *et al.* discuss the impact of unfair natural language in NLP and show how statistical discrimination arises in processing applications. In [19], Bolukbasi *et al.* show gender bias in the world embedding and provide a methodology for modifying an embedding to remove gender bias. In [38], Dixon *et al.* measure discrimination using a set of common demographic identity terms and propose a method to mitigate the unintended bias by balancing the training data.

Compared with all the above-mentioned existing works, we provide further fairness testing. Instead of measuring the overall discrimination, our approach systematically identifies and measures subtle discrimination. That is, we not only measure statistical discrimination with a confidence guarantee but also offer interpretable rule sets to represent subtle discrimination.

This work is remotely related to works on applying rule-based models for model explanation. In [163], Yang *et al.* present an algorithm for building probabilistic rule lists with logical IF-THEN structure. In [87], Lakkaraju *et al.* propose interpretable decision sets to interpret model predictions with high accuracy and high interpretation. Our work leverage such rule-based interpretable structure to present subtle discrimination in models.

4.6 Summary

In this work, we focus on testing neural network models against subtle group discrimination and propose a framework to systematically identify interpretable subtle group discrimination based on group fairness measurement with a certain confidence. Our extensive evaluation demonstrates that subtle group discrimination in neural networks is common to a surprising level. We also show that it is possible to mitigate such discrimination by utilizing our testing results to generate more data for retraining.

Chapter 5

Adaptive Fairness improvement

5.1 Introduction

While these neural networks often have high accuracy in these classification tasks, some concerning fairness issues have been observed as well [130, 23, 30, 46, 16]. To address these fairness issues, many methods and tools have been proposed to detect discrimination in neural networks systematically [47, 176, 9, 150, 99]), and more relevantly, to improve the fairness of neural networks [44, 71, 168, 73, 29, 5, 61, 72, 123, 175]. While various fairness improvement methods have been proposed and try to mitigate discrimination in different aspects, e.g., data processing, model retraining or modifying prediction scores, not all of them are effective in a certain task. In this work, we aim to evaluate existing fairness improvement methods and propose a novel method to choose the 'right' fairness improvement method based on causality analysis.

In general, existing fairness improving methods can be classified into three categories according to when the method is applied, e.g., pre-processing, in-processing and post-processing methods. Pre-processing methods [71, 44, 27, 167] aim to reduce the bias in the training data so as to reduce the bias of model predictions; in-processing methods [29, 168, 73, 5, 6, 76] focus on the model and the training process; and post-processing methods [61, 72, 123] modify the prediction results directly rather than the training data or the model.

However, fairness improving is a complicated task and it is not always clear which method should be applied. As shown in Section 6.3.4, different fairness improving methods perform significantly differently on different models (which is consistent with the partial results reported in [16, 30, 46]). More importantly, applying the 'wrong' method would not only lead to a huge loss in accuracy (e.g., the accuracy of the model trained on the COMPAS dataset drops by 35% after applying the Reject Option post-processing method), but also lead to worsened fairness. For instance, out of 90 cases (i.e., combinations of model, protected attribute and fairness improving method) that we examined in Section 6.3.4, 20% of them result in worsened fairness. Furthermore, a fairness improving method may be effective with respect to one protected attribute whilst being harmful with respect to another protected attribute. For instance, the fairness of the model trained on Adult Income dataset improves by around 4% with respect to the *gender* attribute after applying the Equalized Odds post-processing method and worsens by 20% with respect to the *race* attribute. Given that many of the fairness improving methods require significant effort and computing resource, it is infeasible to try all of them and identify the best performing one. It is thus important to have a systematic way of identifying the 'right' method efficiently.

So then, we propose the method to choose the 'right' fairness improving method based on causality analysis. Intuitively, the idea is to conduct causality analysis so as to understand the causes of the discrimination, i.e., whether a certain number of input attributes or hidden neurons are highly responsible for the unfairness. Formally, we use the probability of high causal effects and Coefficient of Variation to characterize the distribution of the causal effects. Depending on the result of the causality analysis, we then choose the fairness improving method accordingly. For instance, if a small number of input attributes bare most of the responsibility for unfairness, a pre-processing method such as [71, 44] would be the choice, whereas an in-processing method would be the choice if some neurons are highly responsible. Our approach is designed based on the results of an empirical study which evaluates 9 fairness improving methods (i.e., 2 pre-processing methods, 4 in-processing methods and 3 post-processing methods) on 4 different benchmark datasets with respect to different fairness metrics. Our approach is systematically evaluated with the same models. The results show that our selected processing approach is the optimal choice to improve group fairness in all cases and the optimal choice to reduce individual discrimination in most cases.

This work is based on the published paper [172]. The remaining sections of this work are organized as follows. In Section 5.2, we review relevant background. In Section 5.3, we present results from our empirical study which motivates our approach. In Section 5.4, we present our adaptive fairness improving method. In Section 5.5, we evaluate our approach. Lastly we discuss related works in Section 5.6 and conclude in Section 5.7.

5.2 Background

In the following, we review relevant background on fairness and existing fairness improving methods.

5.2.1 Fairness Definitions

In the literature, there are multiple definitions of fairness [41, 69, 25, 47, 81, 165]. What is common across different definitions is that to define fairness, we must first identify a set of protected attributes (a.k.a. sensitive attributes). Commonly recognized protected attributes instance race, sex, age and religion. Note that different models may have different protected attributes.

As in section 4.2.2, we focus on two popular definitions of fairness, i.e., group fairness and individual discrimination. In the following, we formally define these two fairness categories as well as the corresponding fairness scores, i.e., metrics that are used to quantify the degree of unfairness.

Group fairness, which is also known as statistical fairness, is the primary focus of this work as well as many existing studies [170, 62, 77, 145, 14]. Classic measurements for group fairness include positive classification rate and true positive rate. Given a model, we can calculate a model's degree of unfairness using a typical measurement, i.e., Statistical Parity Difference (SPD) [25][1]. Referring to the basic definition of group fairness at Definition 6, we show the formal definition of Statistical Parity Difference in the following.

Definition 8 (Statistical Parity Difference) *Let Y be the predicted output of the neural network N; l be a (favorable) prediction and F be a protected attribute. Statistical Parity Difference is the difference in the probability of favorable outcomes between the unprivileged and privileged groups where the unprivileged/privileged groups are defined based on the value of the protected attribute.*

$$|P(Y=l \mid F=0) - P(Y=l \mid F=1)| \tag{5.1}$$

Note that the above definition only considers a single binary protected attribute, which is sometimes insufficient. The following metric, called Group Discrimination Score (GDS), extends SPD to measure fairness based on multiple protected attributes.

Definition 9 (Group Discrimination Score) *Let N be a neural network; Y be the predicted output of the neural network; l be a (favorable) prediction, and F be a set of (one or more) protected attributes. Let θ (and θ') be an arbitrary valuation of the protected attributes F. Let X_θ be the set of inputs whose F-attribute values are θ. Let P_θ be $P(N(x)=l \mid x \in X_\theta)$. The multivariate group discrimination with respect to protected attributes F is the maximum difference between any P_θ and $P_{\theta'}$.*

[1]There are also alternative similar measures such as Disparate Impact [165] that we omit in this study.

Example 5.2.1 *Consider the structured dataset Adult Income [128]. It has two protected attributes, i.e., gender, and race. Each attribute has a set of two values, i.e., Female or Male for gender, and White or non-White for race. As a result, there are 4 possible θ, i.e., (Male, White), (Female, White), (Male, non-White) and (Female, non-White). The probabilities of an individual who is predicted to have a high-income level (i.e., more than 50K) with respect to these four θ is 14.4%, 39.6%, 9.0% and 28.5% respectively. The GDS of the model is thus 30.6%.*

Individual discrimination is another concept which is often applied in fairness analysis. It focuses on specific pairs of individuals. Intuitively, individual discrimination occurs when two individuals that differ by only certain protected attribute(s) are predicted with different labels. An individual whose label changes once its protected attribute(s) changes is referred to as an individual discriminatory instance. This notion is widely used to search discriminatory instances which differ only in those sensitive characteristics [176, 177, 150]. There are also plenty of works on learning models which are more likely to avoid individual discrimination [130]. The formal definition of individual discriminatory instance is shown in the following.

Definition 10 (Individual Discriminatory Instance) *Let F be a set of (one or more) protected attributes; and N be a neural network. x is an individual discriminatory instance if there exists an instance x' such that the following conditions are satisfied.*

- $\forall q \notin F.\ x_q = x'_q$
- $N(x) \neq N(x')$

The above definition is often adopted in fairness testing, i.e., works on searching or generating individual discriminatory instances [176, 150]. In addition, there are proposals on learning models which are more likely to avoid individual discriminatory [130].

Given a model, we can measure its fairness according to individual discrimination by measuring the percentage of individual discriminatory instances in a set of instances (which can be the test set or a set generated to simulate unseen samples), formally called Causal Discrimination Score (CDS).

Definition 11 (Causal Discrimination Score) *Let N be a neural network; F be a set of protected attributes. The causal discrimination score of N with respect to protected attributes F, is the fraction of inputs which are individual discrimination instances.*

5.2.2 Fairness Improving Methods

Many methods have been proposed to improve the fairness of neural networks [71, 44, 27, 167, 29, 168, 73, 5, 6, 76, 61, 123, 72]. They can be categorized into three groups according to when they are applied, i.e., pre-processing, in-processing and post-processing.

Pre-processing methods aim to reduce the discrimination and bias in the training data so as to improve the fairness of the trained model. Among the many pre-processing methods [71, 44, 27, 167], we focus on the following two representatives in this work.

- Reweighing (RW) [71] works by assigning different weights to training samples in order to reduce the effect of data biases. In particular, lower weights are assigned to favored inputs which have a higher chance of being predicted with the favorable label and higher weights are assigned to deprived inputs.
- Disparate Impact Remover (DIR) [44] is based on the disparate impact metric which compares the proportion of individuals that are predicted with the favorable label for an unprivileged group and a privileged group. It modifies the values of the non-protected attribute to remove the bias from the training dataset.

In-processing methods modify the model in different ways to mitigate the bias in the model predictions [29, 168, 73, 5, 6, 76]. We focus on the following representative in-processing methods in this work.

- Classification with fairness constraints (META) [29] develops a meta-algorithm which captures the desired metrics of group fairness (e.g., GDS), using convex fairness constraints (with strong theoretical guarantees) and then using the constraints as an additional loss function for training the neural network.
- Adversarial debiasing (AD) [168] modifies the original model by including backward feedback for predicting the protected attribute. It maximizes the predictors' ability for classification while minimizing the adversary's ability to predict the protected attribute to mitigate the bias.
- Prejudice remover regularizer (PR) [73] focuses on the indirect prejudge. It uses regularizers to compute and restrict the effect of the protected attributes.
- Exponential gradient reduction (GR) [5] reduces the fair classification problem to a sequence of cost-sensitive classification problems, whose solutions yield a randomized classifier with the lowest empirical error subject to the desired constraints.

Post-processing methods modify the prediction results instead of the inputs or the model. We consider three representative processing algorithms in this work.

- Equalized Odds (EO) [61] solves a linear program to find probabilities with which to change the output labels, so as to optimize equalized odds on protected attributes.
- Calibrated Equalized Odds (CEO) [123] optimizes over calibrated classifier score outputs to find probabilities with which to change output labels with an equalized odds objective.

- Reject Option Classification (RO) [72] assigns favorable labels to unprivileged instances and unfavorable labels to privileged instances around the decision boundary with the highest uncertainty.

5.3 An Empirical Study

In this section, we present an empirical study which aims to compare the performance of different fairness improving methods on different models, different protected attributes or attribute combinations.

5.3.1 Experimental Setup

Datasets Our experiments are based on 4 models trained with the following benchmark datasets: Census Income [128], German Credit [64], Bank Marketing [110] and COMPAS [10]. These datasets have been used as the evaluation subjects in multiple previous studies [176, 47, 38, 130, 99, 170].

- Adult Income: The prediction task of this dataset is to determine whether the income of an adult is above $50,000 annually. The dataset contains more than 30,000 samples. The attributes *gender*, *race* are protected attributes.
- German Credit: This is a small dataset with 600 samples. The task is to assess an individual's credit based on personal and financial records. The attributes *gender* and *age* are protected attributes.
- Bank Marketing: The dataset contains more than 45,000 samples and is used to train models for predicting whether the client would subscribe a term deposit. Its only sensitive attribute is *age*.
- COMPAS: The dataset contains more than 7,000 samples and is used to predict whether the recidivism risk score for an individual is high. The attributes *gender*, *race* are protected attributes.

Table 5.1: Dataset Privileged Groups Definition

Dataset	**protected Attribute**	**Privileged Group**	**Favorable Class**

Adult Income	gender
	race
		gender=Male
		race=Caucasian
			income>50K

German Credit	gender age	gender=Male age>30	good credit
Bank Marketing	age	age>30	Yes

COMPAS	gender
	race
		gender=Female
		race=Caucasian
			no recidivism

In our experiment, we define privileged and unprivileged groups based on the default setting in [14]. The details of the privileged group definitions and favorable class are summarised at Table 5.1. Altogether, we have a total of 10 model-attribute combinations. Our implementation of the 9 fairness improving methods is based on the AIF360 implementation [14]. Each implementation is manually reviewed and tested through standard practice.

Model Training Our models are feed-forward neural networks, which are shown to be highly accurate and efficient in these real-world classification problems [67, 169, 3]. All these neural networks contain five hidden layers, each of which contains 64, 32, 16, 8 and 4 units. The output layer contains 2 (number of predict classes) units. For each dataset, we split the data into 70% training data and 30% test data. All experiments are conducted on a server running Ubuntu 1804 operating system with 1 Intel Core 3.10GHz CPU, 32GB memory and 2 NVIDIA GV102 GPU. To mitigate the effect of randomness, whenever relevant, we set the same random seed for each test. The trained models reach standard state-of-the-art accuracy. The trained results including the corresponding fairness scores are shown in Table 5.2. Note that SPD is the probability difference between the unprivileged and privileged groups which is defined on a single protected attribute and thus it is irrelevant if multiple protected attributes are considered simultaneously.

Table 5.2: Neural Networks in Experiments

Dataset	Protected Attribute	SPD	GDS	CDS	Accuracy
Adult Income	gender	0.249	0.249	0.103	81.7%
	race	0.119	0.119	0.117	
	gender+race	-	0.306	0.179	
German Credit	gender	0.031	0.031	0.078	63.3%
	age	0.095	0.095	0.15	
	gender+age	-	0.133	0.172	
Bank	age	0.047	0.047	0.014	90.0%
COMPAS	gender	0.227	0.227	0.076	72.7%
	race	0.151	0.151	0.028	
	gender+race	-	0.301	0.083	

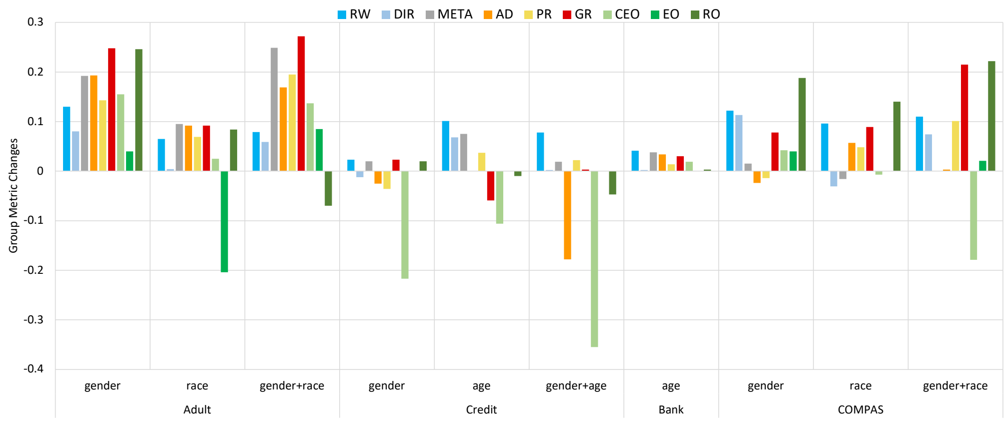


Figure 5.1: Group Fairness Improvement of Models with respect to Different Protected Attributes

5.3.2 Evaluation Results

In the following, we present the results of the empirical study, which aims to answer the following research questions.

RQ1: Do the fairness improving methods always improve group fairness? To answer the question, we systematically apply all fairness improving methods on all the model-attribute combinations and measure the effectiveness of the fairness improving methods. We measure the group fairness improvement as follows. SPD is adopted if a single protected attribute is relevant and GDS is adopted if multiple protected attributes are considered at the same time. Note that GDS is the same as SPD with respect to a single protected attribute.

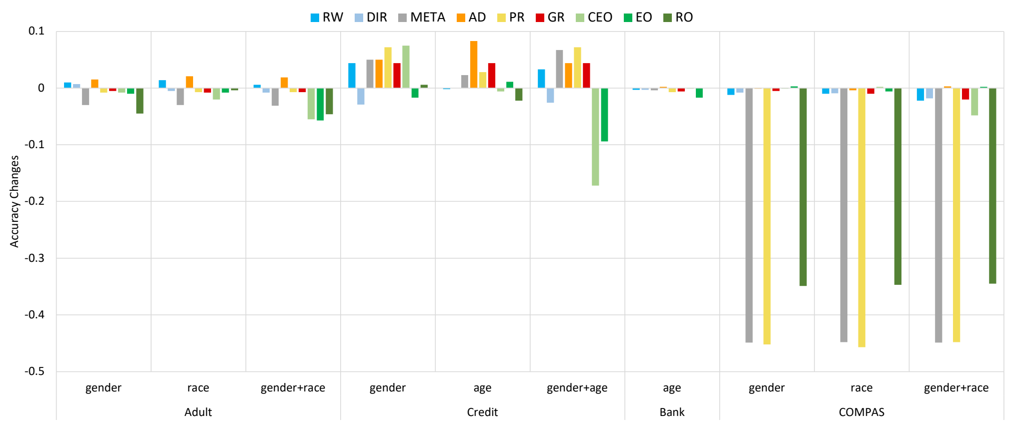


Figure 5.2: Accuracy Changes of Models with respect to Different Protected Attributes After Processing

The results are shown in Figure 5.1, where there is one bar for each model-attribute combination and for each fairness improving method, i.e., a total of 9 bars for each model-attribute combination (e.g., *Adult-gender*) and 90 bars in total. A positive value means improved fairness and a negative value means worsened fairness. This bar is shown in 9 different colors for the nine different methods.

First of all, to our surprise, the fairness improving methods are not always helpful in terms of improving fairness. As shown in Figure 5.1, while many methods have a positive effect in many cases, *there are many instances where applying fairness improving method results in worsened fairness, sometimes quite significantly.* This is shown as the colorful bar before the zero line, which accounts for a total of 18 cases (i.e., 20%). Most of those cases are for in-processing and post-processing methods.

Furthermore, the performance of the methods varies significantly across different models and protected attributes. Table 5.3 shows a summary on which method achieves the most fairness improvement for each model-attribute combination and it can be observed that different winners are there for different model-attribute combinations. Further analysis shows the performance of the fairness improving methods vary across many dimensions. First, the performance of the same method varies significantly on different models. For in-

Table 5.3: Best Method for Group Fairness Improvement

Dataset	Protected Attribute	Group Fairness	Absolute Change
Adult Income	gender	GR	0.248
	race	META	0.095
	gender, race	GR	0.272
German Credit	gender	RW	0.023
	age	RW	0.101
	gender, age	RW	0.078
Bank	age	RW	0.041
COMPAS	gender	RO	0.188
	race	RO	0.14
	gender, race	RO	0.222

stance, while the post-processing method CEO works effectively for the neural network trained on Adult Income dataset, it is ineffective for the model trained on German Credit dataset. Secondly, the performance of the methods varies across different attributes in the same model. For instance, the post-processing method EO improves the group fairness with respect to *gender* attribute effectively but leads to worse group fairness with respect to *race* attribute for the neural network trained on Adult Income dataset.

Moreover, even the processing methods in the same category behave differently on the same model-attribute combination. In terms of in-processing methods, RW is much more effective than DIR. All models' group fairness can be improved by RW, whereas DIR is ineffective with respect to *Credit-gender* and *COMPAS-race*. For in-processing methods, GR is most effective in improving group fairness for all model-attribute combinations except *Credit-age*. The performance of Post-processing methods varies significantly. For example, the post-processing method RO is much more effective in improving the group fairness for the neural network trained on COMPAS dataset than CEO and EO.

We have some conjectures on why fairness improvement approaches may have different effects on different models and different model-attribute combinations. The main reason is that these methods improve fairness based on certain metrics which may be subtly different from common notions of fairness such as

SPG, GDS and CDS. For instance, CEO focuses on reducing False Positive Rate difference in particular, which sometimes translates to fairness measured using SPG/GDS/CDS (as for the Adult Income dataset) and sometimes not. For the different performances on different model-attribute combinations, there may be two reasons. The first is that the discrimination against different attributes in the model may be very different (see in Table 5.2 and observed in [16]). The second possible reason is that the reasons of the discrimination against different attributes may be different, e.g., biased training data or biased models.

So, existing fairness improving methods are not always effective in improving group fairness and thus they must be applied with caution.

RQ2: What is the cost on accuracy when applying existing fairness improving methods? The results are shown in Figure 5.2, where there is similarly one bar for each model-attribute combination and for each fairness improving method. A positive value indicates an increased accuracy and a negative value indicates a decreased accuracy.

First of all, we observe that some of the fairness improving methods may indeed incur a significant loss of accuracy. This is most observable on META, PR, CEO, EO and RO. Especially for the neural network trained on the COMPAS dataset, the accuracy drops more than 40% after applying META, PR or RO. The average loss of accuracy is around 13% after processing by META and 12% after processing by RO. To our surprise, some of the fairness improving methods result in improved accuracy in some cases. This is most observable in some in-processing methods. Especially for the neural network trained on the German Credit dataset, the accuracy increases after applying all four in-processing methods. It should be noted however most of these in-processing methods have a less or harmful effect in terms of group fairness improvement in these cases. For example, while the accuracy increases by 4% after applying GR on *Credit-age*, the SPD fairness score worsens by 6%.

The accuracy reduction varies across not only different model-attribute com-

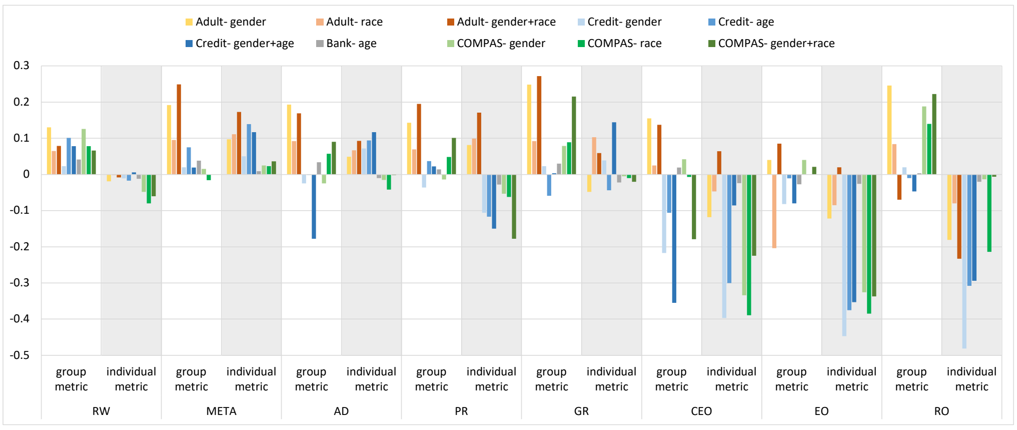


Figure 5.3: Comparison between group fairness improvement and individual discrimination reduction

binations, but also different methods across different categories. Compared fairness improving methods from different categories, the pre-processing methods have an overall mild impact on the model accuracy. In terms of the most effective pre-processing method RW, it is effective on group fairness improvement with respect to all model-attribute combinations and scarifies little accuracy. In terms of the most effective in-processing method GR, it is effective on group fairness improvement with respect to all model-attribute combinations except *Credit-age* (although sometimes with minimal fairness improvement). Among them, 7 neural networks get lower accuracy after processing. But the accuracy drops less than 1% in average. In terms of the post-processing method RO, it is effective on group fairness improvement with respect to 7 model-attribute but 5 neural networks get lower accuracy after processing. Especially for the neural network trained on COMPAS dataset, the accuracy drops more than 30%, which is unacceptable.

In a nutshell, existing fairness improving methods may incur a significant loss in accuracy.

RQ3: Do the fairness improving methods perform differently for improving group fairness and reducing individual discrimination? Almost all existing fairness improving methods focus on group fairness (whilst some fairness testing approaches focus on individual discrimination for some reason). Thus we are curious about whether the existing fairness improving methods can reduce individual discrimination as well. To answer this question, we compare the CDS change against the group fairness metric change achieved by the same method. The idea is to check whether the changes are consistent, i.e., whether an improvement in group fairness leads to a reduction in individual discrimination and vice versa. Note that, by the default setting in [14], the DIR pre-processing method removes all protected attributes, which makes individual discrimination irrelevant, and thus is not considered in this experiment.

The results are shown in Figure 5.3, where the CDS change is placed next to the fairness metric change for each fairness improving method. First of all, the group fairness improvement and individual discrimination reduction are inconsistent. A method improving the group fairness effectively might have none or even harmful effect on individual fairness. This is most observable on RW and RO. The pre-processing method RW is effective on group fairness improvement for all models but lead to more individual discrimination for 8 model-attribute combinations. After applying the post-processing method RO, the individual discrimination worsens for all model-attribute combinations.

Furthermore, only the in-processing methods consistently reduce individual discrimination. In terms of META method, it increases the group fairness and reduces the individual discrimination at the same time for 8 model-attribute combinations. The method AD reduces the individual discrimination with respect to all protected attributes in Adult Income dataset and German Credit dataset. Especially for the neural network trained on Adult Income dataset, all in-processing methods improve the individual fairness effectively. By contrary, all post-processing methods have harmful effect on individual discrimination. on average, the CDS worsens by around 19% after applying CEO, worsens by 24% after applying EO and worsens by more than 18% with RO.

So, existing methods are less effective in reducing individual discrimination.

5.4 An Adaptive Approach

Our empirical study shows that the performance of fairness improving methods varies significantly across different models, i.e., sometimes resulting in worsened fairness and/or reduced accuracy. We thus need a systematic way of choosing the right method. Our proposal is an adaptive approach based on causality analysis. Intuitively, causality analysis measures the "responsibility" of each neuron and input attributes towards the unfairness, and depending on whether the most responsible neurons are in the hidden layers or at the input layer, as well as whether a small number of them are significantly more responsible than the rest. Then we choose the fairness improving method accordingly. In the following, we present the details of our approach.

5.4.1 Causality Analysis

Causality analysis aims to identify the presence of causal relationships among events. Furthermore, it can be used to quantify the causal influence of an event on another event. To conduct causality analysis on neural networks, we first adopt the approach in [31, 145], and treat neural networks as Structured Causal Models (SCM). Formally,

Definition 12 (Structure Causal Model) *A Structure Causal Model consists of a set of endogenous variables X and a set of exogenous variables U connected by a set of functions F that determine the values of the variables in X based on the values of the variables in U. The neural network corresponding SCM can be represented as a 4-tuple Model $M(X,U,F,P_U)$, where P_U is the probability of distribution over U.*

For the neural network, the endogenous variables V are observed variables, e.g., attributes or neurons. The exogenous variables are the unobserved random variables, e.g., noise, and P_U is the possible distribution of the exogenous variables. Trivially, an SCM can be represented by a directed graphical model $G = (X,E)$, where the nodes X is a set of endogenous variables and the edges E are the causal mechanism. Specifically, each node of the graph correspond to a variable. An edge from one node to another indicates a causal relationship between the corresponding variables, where the tail of the arrow represents the case and the head represents the effect.

Based on SCM, the causal effect of a certain event can be computed as the difference between potential outcomes under different treatments. In this work, we adopt the Average Causal Effect (ACE) as the measurement of the causal effect [31, 145][2]. The formal definitions of ACE are shown below (where it is assumed that the input endogenous variables are not correlated to each other).

Definition 13 (Average Causal Effect) *The ACE of a given endogenous variable x with value α on output y can be measured as:*

$$ACE^{y}_{do(x=\alpha)} = \mathbb{E}\left[y \mid do\left(x=\alpha\right)\right] - baseline_x \tag{5.2}$$

where $\mathbb{E}\left[y \mid do\left(x_i=\alpha\right)\right]$ represents the interventional expectation which is the expected value of y when the random variable x is set to α; and $baseline_x$ is the average ACE of x on y, i.e., $\mathbb{E}_x\left[\mathbb{E}_y\left[y \mid do\left(x=\alpha\right)\right]\right]$[3].

Following the recent work reported in [145], we apply ACE to capture the causal influence on model fairness. That is, the y in Equation 5.2 should be a measure of the model unfairness, i.e., SPR, GDS or CDS. For simplicity, we denote it as y_{fair}.

In order to analyze the causal effect on fairness, we analyze two possible causal effects, i.e., the relationship between input attributes to unfairness, and the relationship between the hidden neurons to unfairness. In this work, we make use of the average interventional expectation to approximate the ACE of variable x to y_{fair}. Formally, $ACE^{y_{fair}}_{do(x=\alpha)}$ represents the ACE of variable x under

[2]There are alternative ones such as the gradient of causal attribution [121] which work slightly differently.

[3]or alternatively it can be $\mathbb{E}\left[y \mid do\left(x=\hat{x}\right)\right]$ where $\hat{x}$ is the selected significant point.

value α to the fairness property y_{fair}. One complication is that each input attribute or neuron has many possible values and we must consider all the possible values in computing the ACE. Our remedy is to consider the average Interventional Expectation (AIE).

Definition 14 (Average Interventional Expectation) *Let x be the given endogenous variable, y_{fair} be the fairness property and val_set_x be a set of values of variable x. The average interventional expectation is the mean of expected values of y_{fair} when x is set to be each value α:*

$$AIE_x^{y_{fair}} = \frac{\sum_{\alpha \in val\_set_x} \mathbb{E}[y_{fair} \mid do(x = \alpha)]}{\#(val\_set_x)} \tag{5.3}$$

For the input features with categorical values, we intervene the feature with every possible value based on the training dataset. For the hidden neurons with continuous value, intervening it with every possible value might be consuming. We thus intervene the neurons as follows which is adopted in [31] as well. That is, we assume the "intervener" is equally likely to perturb variable x to any value α within the input range, so that $\alpha \sim U(min_x, max_x)$, where min_x and max_x are the minimum and maximum input values of x. In practice, min_x and max_x can be obtained by observing the value of the input attribute or neuron given all the training samples and the val_set_x is generated by partitioning the range $[min_x, max_x]$ uniformly into a fixed number of intervals. Note that if a specific distribution of the interventions is given, it can be used to generate the intervention values instead.

The details of causality analysis on the hidden neurons are shown in Algorithm 10. Given a neural network N, a set of inputs D (i.e., the training set), a hidden neuron n and the function for measuring the desired fairness score $fair\_metric$, we systematically measure the AIE with neuron intervention. At line 1 and line 2, we set min to the minimum output of n and max to the maximum output of n. Then we generate a set of evenly spaced numbers

Algorithm 10 $CausalityNeuron(N,D,n,fair\_metric)$ where N is the neural network, D is the dataset used to measure causal effect, n is a hidden neuron in N and $fair\_metric$ is the function measuring the fairness score based on the desired fairness metric

```
1: $min$ := minimum output of neuron $n$
2: $max$ := maximum output of neuron $n$
3: $val\_set = generate\_vals(min, max, num\_interval)$
4: for $\alpha$ in $val\_set$ do
5:     $ie \leftarrow \{\}$
6:     $y_{fair} = fair\_metric(N, D|do(n = \alpha))$
7:     $ie \leftarrow ie \cup y_{fair}$
8: end for
9: return $mean(ie)$
```

Algorithm 11 $CausalityAttribute(N,D,f,fair\_metric)$ where N is the neural network, D is the dataset used to measure causal effect, f is an input attribute and $fair\_metric$ is the function of measuring the fairness score based on the desired fairness metric

```
1: $val\_set$ := the set of all possible values of attribute $f$
2: for $\alpha$ in $val\_set$ do
3:     $ie \leftarrow \{\}$
4:     $y_{fair} = fair\_metric(N, D|do(f = \alpha))$
5:     $ie \leftarrow ie \cup y_{fair}$
6: end for
7: return $mean(ie)$
```

within the domain of the neuron output $[min, max]$ as *val_set* through function *generate_vals* at line 3. The input parameter *num_interval* decides how many intervals are there. From line 4 to 8, we calculate the AIE with each perturbing value α. In each round, we first set *ie* as an empty set at line 5 and then calculate the fairness score y_{fair} whilst fixing the value of neuron n as α. At line 9, we return the mean of all Interventional Expectation as the AIE.

Algorithm 11 similarly conducts causality analysis on the input attributes. The only difference is that we perform the intervention on the given attribute f at line 4 with all possible values of the attribute.

5.4.2 Adaptive Fairness Improvement

Once we compute the causal effect of each neuron and each input attribute on fairness (i.e., responsibility for unfairness), we can then adaptively select the fairness improving methods. For example, if the causal effects of input attributes are relatively high, the unfairness is more likely to be related to the input attributes and likely to be eliminated by pre-processing methods. Similarly, if the interior neurons in the neural network have high causal effects on the fairness property, in-processing methods might be a suitable choice for fairness improvement.

Formally, to properly compare the casual effects of neurons and input attributes, we first normalize it with respect to a baseline $\overline{y_{fair}}$, which is the fairness score based on the desired fairness metric without any intervention. The baseline $\overline{y_{fair}}$ can be SPD, GDS and CDS as discussed previously.

We define the causal effects higher than the basic fairness property as high causal effects and vice versa. In other words, only the variable with a causal effect higher than the basic fairness property has the positive causality to unfairness. That is, we only consider those neurons and attributes with a causal effect higher than $\overline{y_{fair}}$ as responsible. Next, we measure the proposition of input attributes and neurons that are considered responsible. Given the set of causal effects of all attribute AIE_f and the set of causal effects of all neurons AIE_n, we formally denote the proportion of high causality attributes as P_f and the proportion of high causality neurons as P_n and define them as follows.

$$P_f = P(AIE > \overline{y_{fair}} \mid AIE \in AIE_f) \tag{5.4}$$

$$P_n = P(AIE > \overline{y_{fair}} \mid AIE \in AIE_n) \tag{5.5}$$

Furthermore, we measure the distribution of the “responsibility” among the input attributes and neurons, since it intuitively has an impact on which fairness improving method should be chosen. For instance, if all input attributes have similar responsibility for unfairness, it is likely hard to pre-process the inputs so as to eliminate the discrimination. Similarly, if all neurons are equally responsible for unfairness, it is complex to improve the fairness by focusing on a few neurons as in [145]. Formally, we use the Coefficient of Variation (CV) to capture the distribution of the causal effects. CV is used to measure the dispersion of data points around the mean. It represents the ratio of the standard deviation to the mean which indicates the degree of variation. In this setting, the larger the CV, the more uneven the distribution of causal effects. We denote the CV of attributes as CV_f and the CV of neurons as CV_n.

The details of how to select fairness improving methods are shown in Algorithm 12. If both the proportion of responsible attributes and responsible neurons is less than a proportion threshold P_thres, few input attributes and neurons are to be blamed for the unfairness. As a result, it is unlikely pre-processing (which focuses on input attributes) or in-processing (which focuses on the hidden neurons) is effective, and thus we choose to apply the post-processing methods. In practice, we set the threshold P_thres to be 10%. Otherwise, there are sufficient number of input attributes or neurons that are responsible for unfairness, we then select to apply a pre-processing method if $CV_f > CV_n$, i.e., the distribution of causal effects is more uneven in the input attributes which means that some of the input attributes are more responsible. Otherwise, an in-processing method is chosen. For pre-processing methods, RW is preferred over DIR, as RW is also feasible to individual fairness metrics. For in-processing methods and post-processing methods, we choose the method with the best improvement and least accuracy cost.

Example 5.4.1 *For the neural network trained on Adult Income dataset, assume that the protected attribute is the "gender" attribute. According to the above discussion, we use the group fairness metric SPD to calculate the causal effects of attributes and neurons. The causality analysis result is shown in Figure 5.4, where each dot represents the AIE of either an input attribute or a hidden neuron. We mark the causal effects of input attributes with black dots and mark the causal effects of hidden neurons in different layers with different*

Algorithm 12 $AdaptiveImprove(P_n, P_f, CV_n, CV_f)$

```
1: if $P_f \leq P\_thres$ and $P_n \leq P\_thres$ then
2:     return post-processing methods
3: else
4:     if $CV_f > CV_n$ then
5:         return pre-processing methods
6:     else
7:         return in-processing methods
8:     end if
9: end if
```

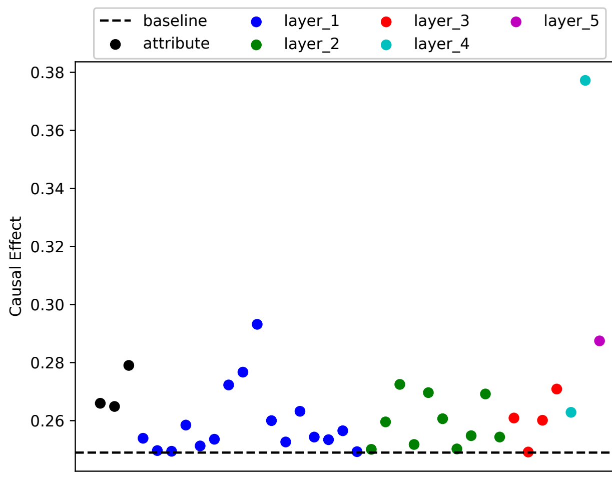


Figure 5.4: Causality analysis result of *Adult-gender*

colors. The dotted line marks the baseline $\overline{y_{fair}}$ which is 0.249. There are 3 (i.e., 25%) attributes with causal effects higher than the baseline and 33 (i.e., 26.6%) neurons with causal effects higher than the baseline. As the proportion of responsible input attributes and neurons satisfy the threshold, we then calculate the CV values of those responsible attributes and neurons. The CV_f of these 3 attributes is 0.041 and the CV_n of these 33 neurons is 0.152. Since $CV_n > CV_f$, we choose to apply in-processing methods so as to improve the model's group fairness.

5.5 Implementation and Evaluation

In this section, we evaluate the performance of our adaptive approach systematically to answer multiple research questions. Note that the same datasets, models, and the configuration from Section 6.3.4 are used in this section.

RQ1: How are the "responsibility" distributed among the neurons and input attributes? To answer this question, we show the probability of high causal effects and CV of these causal effects for both the hidden neurons and input attributes in Table 5.4 and Table 5.5. The first column is the training dataset and the second column shows the corresponding protected attribute(s) in each dataset. Then we show the probability of attributes with high causal effects P_f, the probability of neurons with high causal effects P_n, CV on highly causal attributes CV_f and CV on highly causal neurons CV_n. It can be observed that the distribution of responsibility varies significantly across different model-attribute combinations, which potentially explains why only some fairness improving methods are effective sometimes.

Table 5.4 shows the distribution of high causal effects based on group fairness metrics, e.g., SPD for single protected attributes and GDS for multivariate protected attributes. Based on algorithm 12, the selected processing categories are shown in the last column. For all attribute(s) in Adult Income dataset, the probabilities of high causal effects are higher than 10% and CV_n scores are higher than CV_f scores. So we decide to apply pre-processing methods to this model to improve the group fairness for all attributes. For the neural network trained on German Credit dataset with respect to all attributes, we conclude to apply pre-processing methods. For example, with respect to *age* attribute, both the proportion and the CV of high causal neurons are lower than the two of high causal attributes. Similarly, based on the distribution of high causal effects, we conclude to apply pre-processing to the neural network trained on Bank dataset and the neural network trained on COMPAS dataset with respect to *gender* and

Table 5.4: Distribution of high causal effects with Group Fairness

Dataset	**Protected Attribute**	P_f	P_n	CV_f	CV_n	**Processing**
Adult Income	gender	25.0%	26.6%	0.041	0.152	in-processing
	race	16.6%	28.2%	0.104	0.215	in-processing
	gender+race	27.3%	26.7%	0.095	0.163	in-processing
German Credit	gender	73.7%	46.0%	0.339	0.323	pre-processing
	age	21.1%	9.6%	0.160	0.096	pre-processing
	gender+age	77.8%	53.2%	0.269	0.235	pre-processing
Bank	age	33.3%	37.9%	0.183	0.142	pre-processing
COMPAS	gender	63.6%	43.5%	0.052	0.045	pre-processing
	race	36.4%	19.4%	0.056	0.034	pre-processing
	gender+race	60.0%	86.3%	0.0018	0.002	in-processing

race attributes. With respect to *gender+race* attribute in COMPAS dataset, as the CV of neurons is higher, we conclude to apply in-processing methods.

Table 5.5 show the distribution of high causal effects based on individual fairness metrics, e.g., CDS. The selected processing categories are shown in the last column. Similarly, Algorithm 12 decides to apply in-processing methods for all model-attribute combinations, expect *Credit-gender* and *Bank-age*. We can observe that the proportion of high causal effects of attributes might be 0% in some cases, e.g., *COMPAS-gender* and *COMPAS-race*, which means no attribute is responsible for individual discrimination.

Note that, post-processing methods are selected only if both the proportions of responsible neurons/attributes are low, as it often has a significant negative impact on model performance (so that it is impossible to improve fairness through pre-processing or in-processing). In our experiments, however, all the neural networks have sufficiently many responsible neurons/attributes, so no post-processing method is adopted.

RQ2: Are we always able to identify the best performing fairness improvement method? To answer this question, we compare our adaptive approach against the best performing pre-processing, in-processing and post-processing method in four ways.

- One is the group fairness improvement, which is shown in Figure 5.5(a).

Table 5.5: Distribution of high causal effects with Individual Discrimination

Dataset	Protected Attribute	P_f	P_n	CV_f	CV_n	Processing
Adult Income	gender	75.0%	58.8%	0.033	0.058	in-processing
	race	75.0%	38.7%	0.128	0.141	in-processing
	gender+race	63.3%	46.8%	0.091	0.105	in-processing
German Credit	gender	94.7%	70.2%	0.114	0.096	pre-processing
	age	63.2%	29.0%	0.041	0.053	in-processing
	gender+age	83.3%	10.3%	0..061	0.066	in-processing
Bank	age	40.0%	50.8%	0.076	0.047	pre-processing
COMPAS	gender	0%	15.3%	-	0.026	in-processing
	race	0%	21.0%	-	0.133	in-processing
	gender+race	30%	39.5%	0.075	0.1	in-processing

- One is the group fairness improvement minus the accuracy loss, which is shown in Figure 5.5(b).
- One is the individual discrimination reduction, which is shown in Figure 5.6(a).
- One is the individual discrimination reduction minus the accuracy loss, which is shown in Figure 5.6(b).

As shown in Figure 5.5(a), if we focus on group fairness improvement only, our approach achieves the best performance for 7 out of 10 cases, e.g., *e.g., all attributes in Adult Income dataset, all attributes in German Credit dataset the attribute in Bank dataset*. Although for the neural network trained on Compas dataset, our adaptive approach does not have the best fairness improvement. If we consider at the same time the accuracy loss, as shown in Figure 5.5(b), our approach performs the best in all of the cases. Note that while the post-processing method RO often improves the group fairness significantly, the accuracy often drops significantly (e.g., more than 30% after processing with respect to all protected attributes for the COMPAS dataset, which is clearly unacceptable). In fact, according to our experiments, post-processing should rarely be the choice if we would be maintain high-accuracy. The results shown in Figure 5.5(b) clearly suggests that our approach is able to improve fairness effectively whilst maintaining a high accuracy.

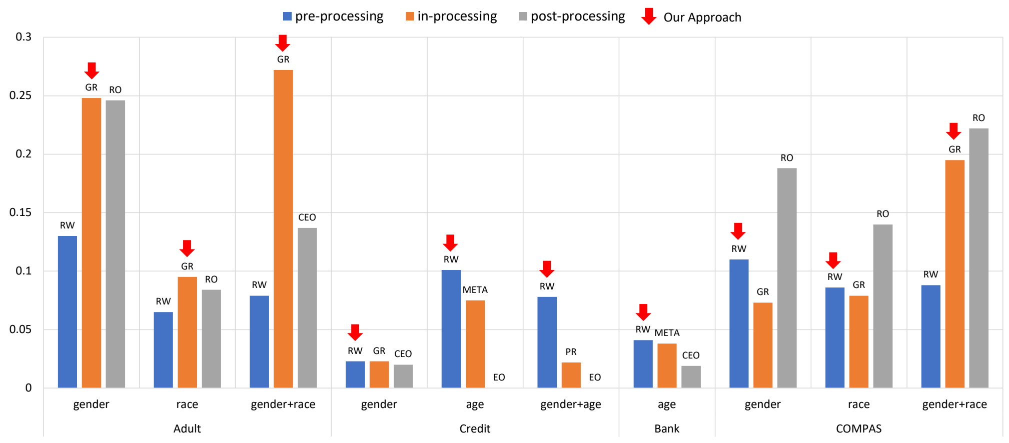
In Figure 5.6(a), we show the comparison between our approach and the existing approaches in terms of reducing individual discrimination. We can observe that only the in-processing methods can reduce the individual discrimination effectively. In fact, our Adaptive Processing Algorithm 12 almost always selects to apply in-processing methods, except for *Credit-gender* and *Bank-age*. After applying the in-processing method RW, the CDS remains almost the same with respect to *Credit-gender* but worsens by around 2% with respect to *Bank-age*. Taking accuracy loss into account at the same time, we show the individual discrimination reduction minus the accuracy lost in Figure 5.6(b). Our approach performs best in 8 out of 10 cases, except for the two cases where RW is selected for *Credit-gender* and *Bank-age*. One potential reason why this is the case is that existing pre-processing methods are not designed for reducing individual discrimination and as a result, even if a small number of input attributes are indeed responsible for the unfairness, existing pre-processing methods such as RW are not able to remove biases in the training set effectively. This calls for research into alternative pre-processing methods for reducing individual discrimination.

It is worth noting that with our approach, we always (10 out 10) achieve improved group fairness and almost always (9 out 10) achieve reduced individual discrimination, whist achieving a low accuracy loss.

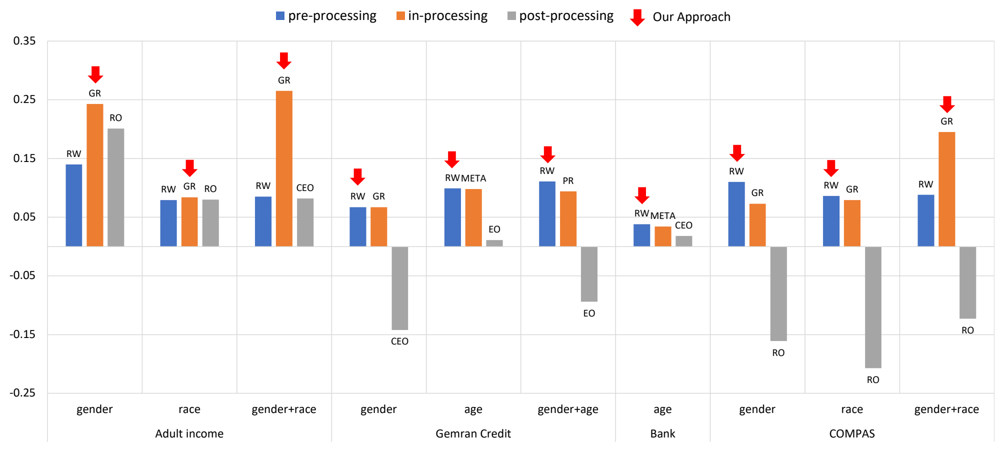
RQ3: What are the time overhead for causality analysis? The time spent on causality analysis is summarised in Table 5.6. Note that the time is the additional time a user has to spend on applying our method before applying the selected fairness improving method. The time required for causality analysis is always less than 10 minutes.

Threats to Validity:

Limited model structures We currently support feed-forward neural networks (for tabular data) and convolutional neural networks (for images). It is possible to extend our method to support deep learning architectures such as RNN (for text data) by extending causality analysis to handle feedback loops. We focus on feed-forward NN as existing studies on fairness largely focus on tabular



(a) *Group Fairness Improvement*



(b) *Group Fairness Improvement - Accuracy Loss*

Figure 5.5: Our Approach vs SOTA on Group Fairness

data [176, 47, 38, 130, 99, 170].

Limited fairness metrics We only use SPD and GDS metrics for group fairness and CDS metric for individual fairness. We focus on GPD and GDS as they are the primary focuses of existing works [9, 14, 62, 77, 145, 170]. Given that GPD and GDS are similar with other metrics which consider positive classification rate like Disparate Impact, our method could work for other notions of fairness as well.

Causal effect measurement ACE is commonly used to evaluate causality [31, 145]. According to [31], alternative measurements like integrated gradients and gradients of causal effect [121] might suffer from sensitivity and induce causal effects by other input features.

Distributional shift in the data Our approach might be affected by distributional

Table 5.6: Time overhead for causality analysis

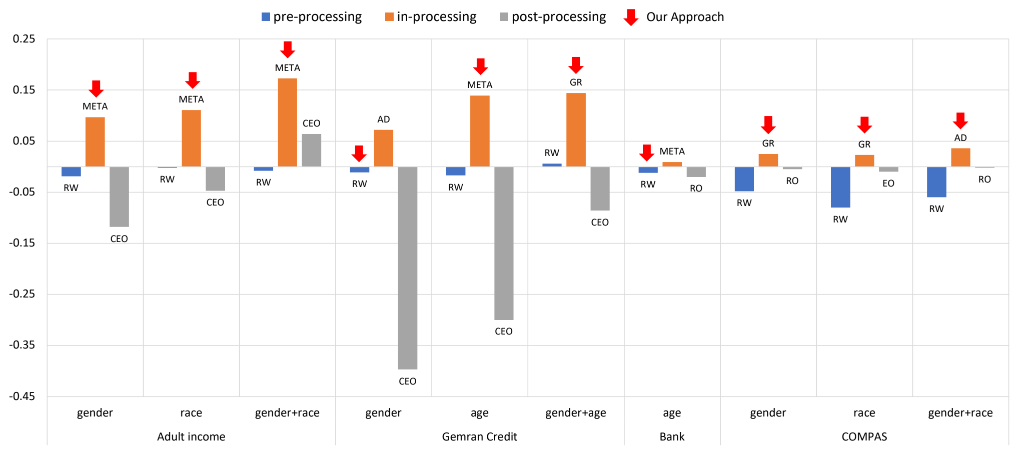
Dataset	Protected Attribute	Time(s)
Adult Income	gender	495.26
	race	504.72
	gender, race	553.42
German Credit	gender	107.79
	age	116.56
	gender, age	221.72
Bank	age	550.52
COMPAS	gender	106.37
	race	152.42
	gender, race	162.19

shifts in the data. We evaluate the stability of our approach against slight distributional shifts on Adult Income dataset. Firstly, following [46], we randomly split train/test set 10 times and then evaluate whther the method selected by our approach is the best one for each of the 10 test sets. Secondly, following [150], we evaluate our approach using data generated by perturbation. In both conditions, the results confirm that is the case. It shows perhaps that our approach is robust to such levels of distributional shift.

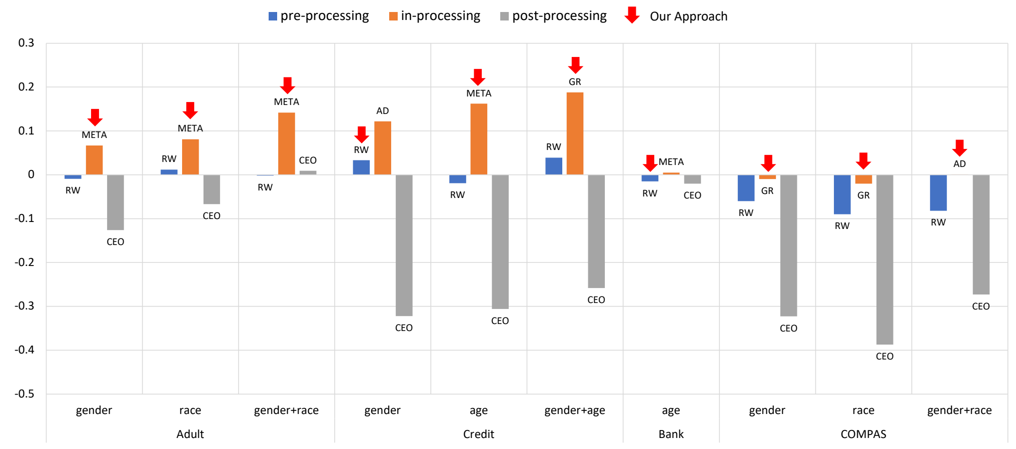
5.6 Related Work

This work is related to research on fairness improving methods, fairness testing and fairness verification methods as well as broadly various studies on fairness. Besides those mentioned in the previous sections, we summarize other related works below.

Fairness Testing and Verification Some existing works attempted to test model discrimination with fairness score measurements. In [148], Tramer *et al.* propose an unwarranted associations framework to detect unfair, discriminatory or offensive user treatment in data-driven applications. It identifies discrimination according to multiple metrics including the CV score, related ratio and associa-



(a) *Individual Discrimination Reduction*



(b) *Individual Discrimination Reduction - Accuracy Loss*

Figure 5.6: Our Approach vs SOTA on Individual Discrimination

tions between outputs and protected attributes. In [81], Kleinberg *et al.* also test multiple discrimination scores and compare different fairness metrics. In [47], Galhotra *et al.* propose a tool called THEMIS to measure software discrimination. It tests discrimination with two fairness definitions, i.e., group discrimination score and causal discrimination score. It measures these two scores for different software instances with respect to race and gender separately. Their approach generates additional testing samples by selecting random values from the domain for all attributes. In [4], Adebayo *et al.* try to determine the relative significance of a model's inputs in determining the outcomes and use it to assess the discriminatory extent of the model. In [51], Ghosh *et al.* verify different fairness measures of the learning process with respect to underlying data distribution.

Empirical Studies of Fairness Chakraborty *et al.* empirically research on the effectiveness and efficiency of existing fairness improvement methods based on group fairness metrics [30]. Friedler *et al.* work on an empirical study to compare the effects of different fairness improvement methods [46]. In [16], Biswas *et al.* focus on an empirical evaluation of fairness and mitigation on 8 different real-world machine learning models. They apply 7 mitigation techniques to these models and analyzed the fairness, mitigation results, and impacts on performance. They also present different trade-off choices of fairness mitigation decisions. Zhang *et al.* discuss how key aspects of machine learning systems, such as attribute set and training data, affect fairness in [170]. Kearns *et al.* test the effectiveness and measure the trade-offs between rich subgroup fairness and accuracy in [77]. In [39], Dodge *et al.* propose four types of programmatically generated explanations to understand fairness in machine learning systems.

5.7 Summary

In this paper, we empirically evaluate 9 fairness improving methods on 4 real world dataset and 90 model-attribute combinations with 3 fairness metric. Our evaluation shows that existing fairness improving methods are not always effective in improving group fairness and are often not effective in reducing individual discrimination. Motivated by the empirical study, we propose a light weight approach to choose the the optimal fairness improving method adaptively based on causality analysis. That is, we identify on the distribution of "responsible" attribute and neurons and choose the methods accordingly. The results show that our approach is effective in choosing the optimal improvement method.

Chapter 6

Non-Intrusive Bias Mitigation

6.1 Introduction

Since the emergence of transformer-based models, like GPT and BERT, the evolution of Large Language Models (LLMs) has taken impressive strides. In recent years, state-of-art LLMs, such as GPT-3.5 and GPT-4 have shown outstanding performance from understanding complex textual context to a generating texts that emulate human speech. These LLMs have demonstrated their potential across a broad range of fields, including healthcare, education, and business decision-making [135, 74, 101]. In healthcare, for instance, LLMs are used to decipher intricate medical literature and assist in disease diagnostics. Meanwhile, within the domain of education, they serve as personalized tutors, aiding in diverse subjects. Meanwhile, these LLMs may also exist some potential problems such as biases have been increasingly concerned [15, 138, 2]. In this work, we extend our focus from fairness testing in deep neural networks applied to tabular data to language models applied to textual data. We aim to introduce a novel bias mitigation approach that does not require access to the internal workings of language models, enabling debiasing from an external perspective.

As LLMs are trained on extensive data corpus, it is inevitable to encompass various forms of biases, e.g., gender, religion, race and politics biases. Various existing studies have demonstrated that biases are present in different parts of LLMs, including word-level representation [102, 96, 26, 134], sentence encoders [105] and downstream tasks [138, 2]. For instance, public word embeddings have been found to exist a strong correlation between gender-related words and career-related attributes [26]. Then, various works on mitigating bias on language models have been proposed. Common approaches include modifying the training data through data augmentation [84, 180], manipulating word embeddings or sentence representation [49, 120, 106, 158], causality-based bias repair [144] and other in-processing methods [63, 91]. However, as training LLMs is increasingly beyond the capacity of ordinary users or small business owners, many of these methods become impractical. Most in-processing methods replying on model retraining or fine-tuning become too costly. Even pre-processing methods such as Counterfactual Data Augmentation [96, 68] still need an additional phase of pre-training. Moreover, due to the latent nature of biases embedded and propagated through complex connections in language models, they often appear in various ways. It is thus challenging to produce a LLM-based production that avoids biases (e.g., accordingly to the regulatory requirements) when we only have API access to the LLMs.

To tackle the challenges, we propose a novel bias mitigation approach, which treats the language model as a black box. That is, we do not access or modify the internals of the given models. We propose a debias adapter based on the the sequence representation output from language models, which aims to mitigate bias in a parameter-efficient manner. The adapter is a multi-layer feed-forward neural network. Compared to bias mitigation through complete model fine-tuning, our approach focuses only on fine-tuning a limited number of parameters in the adapter and downstream tasks. This makes our approach much more efficient. For example, when debiasing a pre-trained 'bert-base-cased' model, our approach requires modifications to only 0.17% of the whole model's parameters.

Moreover, our approach provides statistical confidence in the achieved fairness during debias training. The ability to provide such statistical confidence is essential for many applications that need satisfying regulatory requirements on fairness (e.g., according to General Data Protection Regulation (GDPR), Algorithmic Accountability Act (US AAA) and Artificial Intelligence Act (EU AIA)). In details, we formulate the bias problem as a hypothesis evaluation problem and calculate the loss function on bias using hypothesis testing results, i.e., *z-score*. That is, given a bias threshold ϕ, the null hypothesis is: model's bias is less than ϕ. The alternative hypothesis is the opposite condition. The z-score is used to evaluate the statistical likelihood of the null hypothesis being true. If the calculate *z-score* is close to 0 or negative, it suggests the model's bias degree is not significantly different from the threshold ϕ or is less than ϕ. Then, we assume the null hypothesis is true. As such, at each step of debiasing process, the bias loss satisfies a certain level of statistical confidence.

For experimental evaluation, we focus on the hate speech classification problems and experiment on pre-trained BERT models. Note that, our approach works for other downstream tasks and can be easily generalized to other LLMs, such as GPT, ELMo, RoBERTa, DistilBERT and ALBERT due to its non-intrusive character. We consider identity biases and support both group bias mitigation and individual bias mitigation.

We evaluate our approach on 5 hate speech classification problems. The results show that our approach is effective in mitigating both group bias and individual bias across all models. Furthermore, the accuracy drops for all models are relatively negligible, i.e., under 3%. We compare our results with three state-of-the-art baselines that apply in a similar setting, two for group bias mitigation and one for individual bias mitigation. It shows that our approach consistently achieves the most effective results while maintaining accuracy across all models.

In the remainder of this work, we initially delineate the definition of textual data, bias definitions and our problem definition in Section 6.2. Then we introduce our adapter construction and bias mitigation method in Section 6.3. We show our experiment setup in Section 6.4 and analysis our results in Section 6.5. Lastly, we discuss related works in Section 6.6 and conclude in Section 6.7.

6.2 Background

In this section, we present necessary background on textual data with identities and bias definitions. Lastly, we define our research problem.

6.2.1 Textual Data

First of all, we define textual data and corresponding identity attributes.

As we focus on semantic classification in this work, the inputs of semantic classification system are complete sentences, each is compoased of a set of terms. As a growing number of studies have paid attention on the identity related biases in natural language models [151, 38, 20], we consider identities as sensitive attributes in this work. We define the sensitive attributes of text data based on the presence of identity terms. Inspired by previous work [151], we classified identity attributes into 8 categories, including *male, female, homosexual (gay or lesbian), christian, jewish, muslim, black and white*. These identities are further organized into three categories: Gender, Religion, and Race. Notably, for simplicity, we also include sexual identity, such as *homosexual* within the Gender category.

There exists dataset which is labled with identities by annotators manually. For example, the dataset published by the Jiasaw Unintended Bias in Toxicity Classification Challenge on Kaggle label the identity values based on the fraction of annotators who believed a comment fit the identity mentioned. Based on this dataset, we extract identity terms according to each identity category. Then, once a given comment exists certain identity terms, we believe the comment can be labeled with the corresponding identity.

In this work, the define the text data based on the previous given Definition 5. As we focus on identity bias, the sensitive terms should be identity terms and the categories of sensitive terms should be identity attributes.

Formally, we define the text data x with identity s as follows.

Definition 15 (Text Data with Identity) *A text data x contains a sequence of terms $\{x_1, x_2, \cdots, x_N\}$. We write $S = \{s_1, s_2, \cdots, s_n\}$ to denote a set of identity attributes and write T to denote a set of identity terms $\{t_1, t_2, \cdots, t_k\}$, where $t_j \in s_i$ for some i, for all $j \in [1, k]$ and $t_j \in x$.*

For example, if a comment x *"I am a gay"* exists the identity term t *"gay"*, the comment can be labeled with identity attribute s *homosexual*, where $s \in S$ and $t \in s$. This comment can be further classified in the Gender category.

6.2.2 Bias Definition

Numerous definitions of bias exist within the scope of textual data for language models. Existing definitions can be broadly divided into two popular categories, i.e., intrinsic bias and extrinsic bias. Intrinsic bias refers to bias inherent in the representations, e.g., word embeddings [96, 63, 26, 26] and sentence representation [105, 112, 113]. Extrinsic bias refers to bias in downstream tasks, e.g., demographic bias on prediction classes, disparity in false positive rate and equality of opportunity [142, 91, 68]. Due to the complexity of LLMs, measuring intrinsic bias is not necessary to reflect bias in downstream tasks [53]. In this work, we focus on extrinsic bias. To be specific, we consider both group bias and individual bias in this work.

Group bias: When considering group bias, we focus on the inputs which are wrongly predicted as unprivileged labels. That is, we focus on the False Negative Rate or False Positive Rate depending on the settings. For example, if the semantic classification is trained to predict whether the Twitter comments are hate speech, the positive prediction is the unprivileged label 'hate' and the negative prediction is the privileged label 'non-hate'. Then, we only consider False Positive Rate in this setting. In the following, we adopt the term "False Unprivileged Rate (FUR)" to refer to the false prediction rate on unprivileged labels. We calculate the group bias degree by computing the difference between the overall FUR and FUR on specific identity across all groups (samples labeled by different identities). The bias metric is based on average-violations [162] of true positive/negative rate, which is also known as *equality of opportunity* [60, 143, 50]. Such defined bias is highly correlated to the regulations such as EU AIA. The definitions are shown below.

Definition 16 (Group Bias Degree) *Given the classification model M, an identity attribute s, the group bias degree of M with regards to identity s is:*

$$\delta^s = FUR_s - FUR \tag{6.1}$$

where FUR_s is the FUR of samples labeled with identity s and FUR is the overall demographic FUR. So the overall bias degree of M is :

$$\delta_g = \frac{1}{\#S}\sum_{s\in S}|\delta^s| \tag{6.2}$$

where S is the set of considering identity attributes and #S is the size of the attribute set.

Informally, δ^s calculates the difference between FUR on inputs with identity s and the overall FUR. A lower δ^s means less bias of M with regards to identity s. For Equation 6.2, δ_g calculates δ^s across all groups labeled with identity attribute s and compute the average. A lower δ_g means better consistency in FUR with respect to different identities and less bias of M. We believe that achieving higher consistency in FUR is essential for mitigating potential bias so as to improve fairness in language classification systems.

Individual Bias: According to Definition 15, we treat the identity attributes s as sensitive attributes. So given the semantic classification model M and a set of identity terms T as mentioned in Definition 15, x is an individual discriminatory instance (idi) if there exists an instance x' such that the following conditions are satisfied.

- $\forall t \notin T. x_t = x'_t$
- $M(x) \neq M(x')$

That is, if there exists an instance x' where the identity terms are changed, but all other tokens remain the same as in x, and the prediction of model M changes, then x and x' are considered as a pair of individual discriminatory instances. The above definition is similar with Definition 10 found in Section 5.2.1. However, in this context, we specify idis with a focus on textual data. The operation of modifying identity terms of original samples are similar with Counterfactual Data Augmentation [96, 68], while the latter only considering two identities, i.e., *female* and *male*. More details of individual discriminatory instances generation are discussed in Section 6.3.4. We quantify the individual bias degree with the following definition, which is also similar with Causal Discrimination Score as defined in Definition 11 found in Section 5.2.1. However, in this work, we use symbol δ_i to denote the bias degree.

Definition 17 (Individual Bias Degree) *Given the classification model M, a set of identity attributes S, the individual bias degree of M is:*

$$\delta_i = \frac{\#idi}{\#samples} \tag{6.3}$$

where $\#idi$ *is the number of detected individual discriminatory instances (idis) and* $\#samples$ *is the size of testing samples.*

A lower δ_i means less individual bias of M.

6.2.3 Problem Definition

In the following, we define our research problem and briefly discuss how we aim to solve the problem.

Our aim is to develop a systematic method for mitigating extrinsic bias with regards to both group fairness and individual fairness. The proposed debiasing method must operates under the following assumptions:

- Non-intrusive: it does not access or modify the internals of a given language model.
- Fairness with statistical confidence: it can provide a theoretical guarantee at all stages of the debiasing process.
- Maintaining accuracy: it improves fairness with little cost on accuracy.

6.3 Methodology

In this section, we describe how our debiasing method works. First, we provide an overview of our debias strategy and describe the proposed debias adapter, which is designed to mitigate bias from an external standpoint, eliminating the need to access or modify the internals of language models. Then, we describe how we design a general loss function that works for both group fairness and individual fairness, as well as provides statistical confidence on the achieved fairness.

6.3.1 Overview

The overview of our debias strategy is shown in Figure 6.1. First, inspired by the idea of adapters for the transformer architecture [65, 122], we introduce the debias adapter which aims to mitigate bias in a parameter-efficient way. The debias adapter is added after the given language model and right before

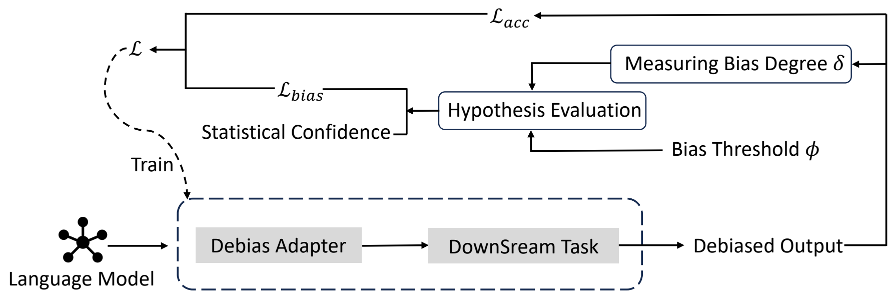


Figure 6.1: Overview of the Proposed Debias Strategy

the downstream task, i.e., hate speech classifier in this work. We adopt such non-intrusive adapter for several reasons. Primarily, as state-of-the-art language models grow increasingly complex, the process of debiasing through fine-tuning or internal adapters becomes not only costly but also practically impossible, especially when the internal architecture is not accessible, e.g., for GPT-3.5 and GPT-4. Hence, we aim to treat the whole encoding model, such as BERT, as black-box encoder. In this setup, we can access only the sequence representation output, e.g., through an online API, and not any multi-head attention or feed-forward layer within the transformer layers. Then, based on the prediction output of the classifier, we first calculate the loss on accuracy as $\mathscr{L}_{acc}$. Second, we measure the bias degree, denoted as δ, according to our definitions. Then, we formulate the bias problem as a hypothesis evaluation problem, so we can provide statistical confidence on the fairness in each step of training. The null hypothesis and alternative hypothesis are defined based on a pre-defined bias threshold ϕ. According to the evaluation results, we calculate the loss on bias degree as $\mathscr{L}_{bias}$. Lastly, we combine $\mathscr{L}_{acc}$ and $\mathscr{L}_{bias}$ to train debias adapter and classifier.

In detail, the overview of the debias adapter is shown in Figure 6.2. The adapter is a multi-layer feed-forward neural network with two bottlenecks. We limit the number of parameters at bottleneck to 8% of the parameters of the original output. We employ skip-connection internally and nonlinearity connection, i.e., GELU, between layers in the adapter module.

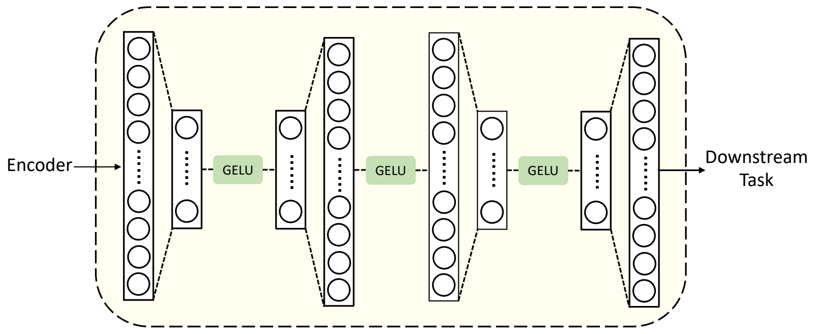


Figure 6.2: Overview of Debias Adapter

6.3.2 Mitigating Bias with Statistical Confidence

In this section, we first discuss how we design the loss function to mitigate bias without significantly compromising accuracy. Then, we discuss how to provide statistical confidence during the debias training.

First, to avoid compromising accuracy in favor of fairness, we combine the main task objective and the fairness constraint.

$$\text{minimize } \mathscr{L}(M(x), y) \text{ s.t. } \mathscr{L}_{\text{bias}} \leq v$$

where $\mathscr{L}(M(x), y)$ is the loss of the main task for input x, i.e., classification loss, $\mathscr{L}_{\text{bias}}$ is the loss on bias and v is a slack variable, which stands for the maximum value of bias loss. A common approach to handle this constrained optimization problem is using the Lagrangian function.

$$\mathscr{L} = \mathscr{L}_{acc} + \lambda \mathscr{L}_{bias} \tag{6.4}$$

where $\mathscr{L}_{acc}$ is the loss for the main classification task and λ is the user-defined weight on bias loss. The higher the weight, the more attention is paid to bias mitigation. The loss on accuracy $\mathscr{L}_{acc}$ can be any existing loss function applied to classification problems.

Second, to provide statistical confidence on the fairness of the model during the debias training, we formulate the bias problem as a hypothesis evaluation problem and calculate the bias loss using hypothesis testing (thereby having a certain level of statistical confidence on the loss value). Hypothesis testing is a statistical procedure used to evaluate the validity of a claim or hypothesis about a population based on observed samples. For example, we initially set a null hypothesis (H_0) and an alternative hypothesis (H_1), along with a defined significance level that outlines the decision-making criteria. Hypothesis testing evaluates whether there is a substantial discrepancy between a sample proportion and a hypothesized population proportion. This reference allows us to gauge the likelihood of the null hypothesis being true. In machine learning, it is commonly used to evaluate a model's performance based on a sample of data.

Specifically, we apply Z-test to evaluate the null hypothesis. The Z-test is a widely used hypothesis test when the sample size is large and the population standard deviation is known or estimated. It is widely employed in various fields, such as medicine, social sciences, and business, helping derive valuable insights from data and facilitating informed decision-making. When implementing a Z-test, we compute the test statistic, commonly referred to as the z-score. Then we compare the calculated z-score to the critical z-value based on the significance level. If z-score exceeds the critical value, the null hypothesis is rejected. The smaller the z-score, the less likely to reject the null hypothesis.

The formula for calculating the z-score in a one-proportion Z-test is shown below.

$$\textit{z-score} = \frac{\sqrt{n}\,(\hat{p} - p_0)}{\sqrt{p_0\,(1 - p_0)}} \tag{6.5}$$

where $\hat{p}$ is the sample proportion, p_0 is the population proportion and n is the sample size.

In this work, we aim to test whether the bias metric satisfies a given regulation, e.g., $\delta \leq \phi$. Here, δ is the bias score as defined in Section 6.2.2 and ϕ is self-defined threshold. So in *z-score* calculation, $\hat{p} = \delta$ and $p_0 = \phi$.

As the z-test we apply is one-tail test, the critical value is determined based on the tail of the distribution in which the alternative hypothesis is located. Given a user-provided significance level, e.g., 0.05, we can systematically compute the critical value, e.g., 1.64. Once the calculated z-score is larger than 1.64, the null hypothesis is rejected with statistical significance.

Then, we scale the z-score to a range between 0 and 1 as $\mathscr{L}_{bias}$. We apply the typical normalization function with clipping as shown below.

$$\mathscr{L}_{bias} = clipped\_ReLU\left(\frac{z\text{-}score - Z_{min}}{Z_{max} - Z_{min}}\right) \tag{6.6}$$

where, $Z_{min} = z\text{-}score(0)$ and $Z_{max} = z\text{-}score(\delta_{orig} + \tau)$ (τ is a self-defined constant which represents the offset added on the baseline bias degree).

Considering that we have defined the maximum value for the *z-score* to be achieved when the bias degree reaches $\delta_{orig} + \tau$, it is possible that higher bias scores may push the normalized *z-score* beyond 1. To manage this potential issue, we employ a *clipped_ReLU* function [24] on the normalized *z-score*. This function restricts the resulting value within the range $(-\infty, 1]$, effectively limiting the normalization output between 0 and 1. This step is crucial to the accurate calculation of the bias loss.

To ensure the bias loss satisfies a certain confidence level in each batch during debias training, we further calculate the sample size it requires. Ideally, we always have a confidential conclusion on each *z-test score* as well as $\mathscr{L}_{bias}$. To guarantee the *z-score* satisfies a certain confidence level, we use a sample size calculation formula as shown in Equation 6.7 to fix the batch size when considering two types of errors, i.e., Type I error α and Type II error β. Given a null hypothesis H_0, α represents the probability that a test wrongly rejects a true null hypothesis, i.e., $P(reject\ H_0 \mid H_0\ is\ True)$ and β represents the probability that does not reject a false null hypothesis, i.e., $P(do\ not\ reject\ H_0 \mid H_0\ is\ False)$. With Type II error β, Power can be defined as $1 - \beta$ which represents the probability that a test correctly rejects a false null hypothesis, i.e., $P(reject\ H_0 \mid H_0\ is\ False)$. The calculation of batch size which is also known as sample size is shown below.

$$n = \left(\frac{z_\alpha + z_{1-\beta}}{\hat{p} - p_0}\right)^2 p_0 (1 - p_0) \tag{6.7}$$

where Z_α and $Z_{1-\beta}$ are the critical values from the standard normal distribution (Z-distribution) at significance level α and Power $1-\beta$ respectively. $\hat{p} - p_0$ is the desired level of precision (i.e. the margin of error) which control the maximum deviation allowed.

In the following, we show how we derive the Equation 6.7 in detail.

Theorem 5 *Given the null hypothesis H_0 and the alternative hypothesis H_1, according to the definition of Type I error α, Power $1-\beta$ and the definition of z-score, the following two equations are satisfied.*

$$\alpha = P\left(\frac{\sqrt{n}\,(\hat{p} - p_0)}{\sqrt{p_0(1-p_0)}} \geqslant z_\alpha \mid H_0\right)$$

$$1 - \beta = P\left(\frac{\sqrt{n}\,(\hat{p} - p_0)}{\sqrt{p_0(1-p_0)}} \geqslant z_{1-\beta} \mid H_1\right)$$

So we get:

$$\frac{\sqrt{n}\,(\hat{p} - p_0)}{\sqrt{p_0(1-p_0)}} \geqslant z_\alpha + z_{1-\beta}$$

Hence,

$$\hat{p} - p_0 \geqslant \left(z_\alpha + z_{1-\beta}\right)\sqrt{\frac{p_0(1-p_0}{n}}$$

$$\sqrt{n} \geqslant \left(z_\alpha + z_{1-\beta}\right)\frac{\sqrt{p_0(1-p_0)}}{\hat{p} - p_0}$$

Thus,

$$n = \left(\frac{z_\alpha + z_{1-\beta}}{\hat{p} - p_0}\right)^2 p_0 (1 - p_0)$$

Similarly, we can also consider one type of error only, i.e., Type I error α. If so, the sample size is calculated by the following equation.

$$n = \left(\frac{z_\alpha}{\hat{p} - p_0}\right)^2 p_0 (1 - p_0) \tag{6.8}$$

For example, we set the bias degree threshold ϕ (i.e., p_0) as 10%, the margin of error $\hat{p} - p_0$ as 0.05, the significance level α as 95% (z_α is 1.65) and the power $1-\beta$ as 0.80 ($z_{1-\beta}$ is 0.84). According to Equation 6.7, the calculated sample size n is 223. Similarly, if only Type I error is considered, the sample size n is 98, according to Equation 6.8.

6.3.3 Mitigating Group bias

In the following, we show how our method can be used to improve group fairness based on a black-box LLM. Although the bias loss $\mathscr{L}_{bias}$ is differentiable over δ (i.e. $\hat{p}$), the bias score δ_g involving indicator variables is non-differentiable. This characteristic poses challenges to the direct application of hypothesis testing discussed in Section 6.3.2.

The null hypothesis and the alternative hypothesis are shown below:

- $H_0 : \delta_g < \phi$
- $H_1 : \delta_g \geq \phi$

where ϕ is the self-defined group bias threshold.

According to the Definition 16, δ_g is the average difference between FUR_s and FUR, where FUR is a constant. In the following, we provide a differentiable way of FUR_s and δ_g calculation.

Given a group of samples G, the classification model M, a set of identity attributes S, a label set with all privileged labels L^+ and a label set with all unprivileged labels L^-:

$$FUR_s = \frac{\sum_{x_i \in G_s^+} \mathbb{S}(\theta(\mathscr{L}(M(x_i), L^+) - \mathscr{L}(M(x_i), L^-)))}{\#G_s^+} \tag{6.9}$$

$$\delta_g = \frac{1}{\#S} \sum_{s \in S} |FUR_s - FUR| \tag{6.10}$$

where G^+ denotes the subgroup of samples with group-truth privileged labels from G and G_s^+ denotes samples which are labeled with identity s among G^+. θ is a constant, $\mathbb{S}$ is the Sigmoid active function, x_i is the i'th sample from group G_s^+, and $\mathscr{L}$ can be any loss function for classification problems. In our experiments, we apply BCELoss by default.

In detail, $\mathscr{L}(M(x_i), L^+)$ calculates the loss between classification model's prediction and privileged labels and $\mathscr{L}(M(x_i), L^-)$ calculates the loss for unprivileged labels. Once the loss for privileged labels is larger than it for unprivileged labels, the prediction output of language model M on instance x_i is unprivileged label. So if $\mathscr{L}(M(x_i), L^+) - \mathscr{L}(M(x_i), L^-) > 0$, x_i is a False Unprivileged instance. Then, we use Sigmoid function to map positive values to 1 and negative values to 0. To streamline the following explanation, we introduce the term if_FU to denote $\mathbb{S}(\theta(\mathscr{L}(M(x_i), L^+) - \mathscr{L}(M(x_i), L^-)))$. If if_FU is 1, the sample x_i is a False Unprivileged instance. After traveling all samples in G_s^+, we calculate the sum of if_FU as the value of False Unprivileged (FU). We compute FUR through dividing FU by the size of group G_s^+ as shown in Equation 6.9. Then we average the absolute difference between FUR_s and FUR for different identity s as δ_g at Equation 6.10.

Overall, we utilize a one-tailed z-test with a significance level, e.g., 0.05. Consequently, based on the bias loss $\mathscr{L}_{bias}$ calculated from the bias score δg, it is able to determine with 95% confidence whether δ_g is truly less than the threshold ϕ. This gives us a statistically robust means to challenge the validity of the null hypothesis.

6.3.4 Mitigating Individual bias

In the following, we show how our method can be used to improve individual fairness of a black-box LLM. However, like the scenario with group fairness, the straightforward application of hypothesis testing proves to be unfeasible due to the inherent non-differentiable nature of the bias measurement.

The null hypothesis and the alternative hypothesis are shown below:

- $H_0 : \delta_i \leq \phi$
- $H_1 : \delta_i > \phi$

where ϕ is the self-defined individual bias threshold.

According to the definition of individual bias degree as Definition 17, we provide a differentiable way of δ_i calculation.

Given a group of samples as G, a classification model M, a set of identity attributes S, a label set with all 'True' labels L^{true} and a label set with all 'False' labels L^{false}:

$$\begin{aligned}\delta_i = \frac{1}{\#G} \sum_{x_i \in G} 2(\mathbb{S}(\theta(\sum_{s \in S} \text{ReLU}(\mathscr{L}(M(x_i^s), L^{true}) - \\ \mathscr{L}(M(x_i^s), L^{false})))) - 0.5)\end{aligned} \tag{6.11}$$

where, θ is a constant, $\mathbb{S}$ is the Sigmoid active function, x_i is the original i'th sample from group G, x_i^s is perturbed sample with identity s based on x_i, and $\mathscr{L}$ is the loss function for classification problems. Note that L^{true} and L^{false} is the predicted labels on the original samples x_i, i.e., $L^{true} = M(x_i)$ and $L^{false} = 1 - M(x_i)$.

In detail, the term $\mathscr{L}(M(x_i^s), L^{true})$ represents the loss between model's prediction of perturbed sample x_i^s and the predicted label of the original sample x_i. The term $\mathscr{L}(M(x_i^s), L^{false})$ represents the loss obtain between model's prediction on x_i^s and the opposite predicted label. Once $\mathscr{L}(M(x_i^s), L^{true}) - \mathscr{L}(M(x_i^s), L^{false}) > 0$, the predicted label of x_i^s differs from the original predicted label. Then we use ReLU function to clip negative values. To streamline the explanation, we introduce the term is_diff to denotes $\text{ReLU}(\mathscr{L}(M(x_i^s), L^{true}) - \mathscr{L}(M(x_i^s), L^{false}))$. If is_diff is positive, sample x_i^s and x_i is a pair of idis. Then we travel all perturbed samples x_i^s and sum up is_diff. If $\sum_{s=1}^{S} is\_diff$ is positive, the original sample x_i is an idi. Then we use formula $2(\mathbb{S}(\theta(\sum_{s=1}^{S} is\_diff)) - 0.5)$ to map the output into 1 and 0, e.g., 1 means x_i is an idi and 0 means the opposite. After traversing through all the samples in G, we compute the total count of idi. This sum is then divided by the total number of samples $\#G$, yielding the individual bias degree δ_i.

When mitigating individual bias, we are required to generate idis. We search for potential idis by data masking which is a mature technique on textual data, especially for gender attributes [68, 68]. That is, we generate a new identity version based on the original text by turning all included identity terms into the terms of the new identity within the same category. For example, given an original text with identity *male*, we need to generate additional instances with other identities within Gender category, e.g., *female* and *homosexual*. In detail, we turn all *male* related terms into *male* or *homosexual* related terms. For instance, we generate the instance "I am a mother" from an original comment "I am a father" when considering *female* identity. For gender-related terms (not including sexuality terms), i.e., *female, male*, we use 'ALL' term set in previous work [68] which contains 341 terms for *male* and *female* version respectively. For other identities, we identify the term sets based on our identity term sets.

Note that, for text containing multiple identities, we only consider one identity at one time and we only turn terms within the same category. For instance, given the comment "I am a mother and a Christian follower", we can generate the instance "I am a mother and a Muslim follower" considering Religion category and generate the instance "I am a father and a Christian follower" when considering Gender category.

Similar to the case for group bias mitigation, we utilize a one-tailed z-test with a significance level, e.g., 0.05. This enables us to maintain 95% confidence in the validity of the null hypothesis on individual bias.

6.4 Experiment Setup

In this section, we conduct multiple experiments to evaluate the relevance of our approach.

Dataset: Our experiments are based on 5 models trained with the following benchmark datasets:

- Hate speech dataset from a white supremacist forum *(WhiteForumHate)* [36] : A total number of 10,568 sentences have been extracted from Stormfront, a white supremacist forum. Those sentences have been manually labelled as containing hate speech or not, according to certain annotation guidelines. Among them, 961 sentences are labeled as hate speech.
- Hate speech dataset from Twitter posts *(TwitterHate)* [36, 157] : We combine these two datasets as one, as they both consider hate speech from Twitter in the form of racist and sexist. It provides annotations for a publicly available corpus for more than 16k tweets by amateur and expert annotators. According to their given data with tweet IDs and labels at Github [1], there are 13,180 tweets in total. Among them, 4304 tweets are annotated as ”Sexism”, 2068 tweets are annotated as ”Racism” and 48 tweets are annotated as ”Both”.
- Hate speech dataset from Gab posts *(GabHate)* [124] : It is a large-scale hate speech intervention dataset collected from Gab [2]. All the posts are manually labeled as hate or non-hate speech by Mechanical Turk workers, so they can also be used for the hate speech detection task. It contains 31,914 posts in total, containing 13,999 hate speech.
- Hate speech dataset from Reddit posts *(RedditHate)* [124] : All posts in this dataset are collected from Reddit [3]. Similar with hate speech dataset from Gab, all posts are manually labeled as hate or non-hate speech. It contains 17,557 posts in total, containing 5,257 hate speech.

[1] `https://github.com/zeeraktalat/hatespeech`
[2] `https://gab.com`
[3] `https://www.reddit.com`

Table 6.1: Metrics of Baseline BERT Models w.r.t. Overall Performance and Individual Identity Categories

Bias Type	**Dataset**	**Accuracy**	**F1**	δ	**Gender** (δ^{male}, δ^{female}, $\delta^{homosexual}$)	**Religion** ($\delta^{christian}$, δ^{muslim}, δ^{jewish})	**Race** (δ^{black}, δ^{white})
Group Bias	WhiteForumHate	91.22%	0.9506	0.0749	0.0448 (0.0288, 0.0348, 0.0708)	0.1116 (-0.0403, 0.2097, 0.0847)	0.0651 (0.0887, 0.0416)
	TwitterHate	91.88%	0.9155	0.2369	0.0671 (0.0265, 0.1019, -0.0729)	0.6771 (0.4271, 0.9271, None)	0.0514 (-0.0329, 0.07)
	GabHate	87.34%	0.8585	0.1485	0.1362 (0.0491, 0.1131, 0.2466)	0.117 (0.1322, 0.2046, -0.0141)	0.2143 (0.3485, 0.0802)
	RedditHate	88.80%	0.8098	0.1060	0.0336 (0.027, 0.0595, 0.0144)	0.2063 (0.0306, 0.5154, -0.0729)	0.064 (0.0876, 0.0405)
	GabTwitterHate	77.93%	0.8264	0.0876	0.0563 (-0.0281, -0.1072, -0.0336)	0.1067 (0.0097, 0.3062, -0.0043)	0.1057 (0.1179, -0.0936)
Individual Bias	WhiteForumHate	91.22%	0.9506	0.1962	0.1369	0.2364	0.1599
	TwitterHate	91.88%	0.9155	0.1414	0.1176	0.1096	0.1148
	GabHate	87.34%	0.8585	0.3057	0.1051	0.1708	0.3718
	RedditHate	88.80%	0.8098	0.1021	0.0336	0.2037	0.2319
	GabTwitterHate	77.93%	0.8264	0.2361	0.0903	0.2388	0.2072

- Hate and offensive speech from Twitter and Gab posts *(TwitterGabHate)* [104] : This dataset is collected from previous studies on hate speech from Twitter [45] and Gab [103]. It contains 20,148 posts in total, which includes 12,334 hate speech.

Models: We fine-tune BERT on downstream prediction tasks. [4] We apply BERT base cased (BERT-base-cased) pre-trained model from Pytorch library [78, 117]. BERT-base-cased models contain 12 attention layers, 768 hidden, 12 heads, 110M parameters and 30522 cased vocabulary size. The model is trained on an NVIDIA-SMI 520.61.05. The hyper-parameters for fine-tuning the pre-trained BERT models are inspired by previous work [68]. By default setting, we set the dropout rate as 0.5, the maximum sequence length as 128, batch size as 32 and learning rate as 2e-5. The model is optimised with Adam. As BERT fine-tuning tends to overfit quickly, we set the epoch as 3.

The basic metrics on the baseline models are shown in Table 6.1. We present the performance of model in terms of accuracy, F1 score. Additionally, we include δ_g and δ_i in the fifth column, representing the degree of group bias and individual bias respectively. Then, we proceed to further evaluate the bias degree for each identity category, including Gender, Religion, and Race in the last three columns. Concerning group bias, we present the group bias degree δ_g with

[4]Our approach works for other LLMs such as ChatGPT as well. We do not evaluate it because (1) it is costly stlll, and (2) ChatGPT does not provide an API for downstream task fine-tuning yet.

Table 6.2: Metrics after Debiasing with Regards to Group Bias

Dataset	**Accuracy**	**F1**	δ_g	**Gender** (δ^{male}, δ^{female}, $\delta^{homosexual}$)	**Religion** ($\delta^{christian}$, δ^{muslim}, δ^{jewish})	**Race** (δ^{black}, δ^{white})
WhiteForumHate	91.34%	0.9525	0.0203	0.0087 (0.0035, 0.0098, -0.0127)	0.0355 (-0.0127, -0.0127, 0.081)	0.0151 (0.0088, 0.0214)
TwitterHate	89.10%	0.8839	0.1183	0.0512 (0.0038, 0.0616 0.0884)	0.2500 (0.4217, -0.0783, None)	0.0871 (-0.0383, 0.136)
GabHate	87.61%	0.8553	0.0753	0.0282 (0.0198, 0.0247, 0.0399)	0.0531 (-0.0364, 0.08 0.0429)	0.1794 (0.3403, 0.0186)
RedditHate	89.26%	0.8115	0.0372	0.0244 (0.0095, 0.0021, 0.0616)	0.0631 (0.0129, 0.1204, -0.0561)	0.0176 (0.018, 0.0173)
GabTwitterHate	77.78%	0.8220	0.0667	0.0374 (-0.0028, -0.0026, -0.1069)	0.0840 (0.0041, -0.0096, 0.2383)	0.0847 (0.0959, -0.0734)

respect to each identity category, as well as the corresponding δ^s for each individual identity s. A positive δ^s means larger FUR compared with the baseline FUR and vice versa. We can observe that, the group bias degree varies significantly when considering different identity categories. In most cases, the group bias degree on Religion category is the highest. Among total 40 identities s, 29 of them show positive δ^s. That is, when the textual sample contains identity information, the model is more likely to wrongly predict it as hate speech. Concerning individual bias, the bias degree δ_i for each identity category represents the percentage of idis observed when only perturbing the corresponding identity terms. We can observe that, the bias degree varies significantly when considering different identity categories. Most of the models exhibit higher individual bias on Religion and Race categories. For instance, in the case of model fine-tuned on the *GabHate* dataset, δ_i is notably high at 0.3718 when considering Race category, but only 0.1051 when considering Gender category.

Other hyper-parameters: For *z-test*, we set Type I error α as 0.05 and Type II error β as 0.2 by default setting. The margin of error $\hat{p} - p_0$ is set to 0.05. We determined the bias threshold, denoted as ϕ or p_0, based on the bias degree observed in the training data of the baseline model. By default, we set the bias threshold at 10%. However, if the original bias degree is below 10%, indicating a relatively low level of bias, we adjust the threshold to 5%. Specifically, for group bias mitigation, we set the threshold at 10% for the baseline model fine-tuned on the *TwitterHate* and *GabHate* datasets, while for the remaining datasets, the threshold is set at 5%. For individual bias mitigation, we set the threshold at 10% for all baseline models, except the model fine-tuned on the *RedditHate* dataset. The weight λ assigned to the bias loss in Equation 6.4 is determined empirically. Specifically, for individual bias mitigation, it falls within the range of 0.1 to 1. For group bias mitigation, it is set between 1 and 1.5. During debiasing, we determined the batch size automatically based on the guidelines outlined in Section 6.3.2 and the learning rate is set to 1e-3. We train the debias adapter along with fine-tuning the downstream classifier within 20 epochs using the Adam optimizer.

6.5 Implementation and Evaluation

In this section, we evaluate the effectiveness of our debiasing approach systematically to answer multiple research questions. In our experiments, we only consider debias adapters that yield significant mitigation on the bias degree, with minimal impact on accuracy. For all models, we set an upper bound for an acceptable decline in accuracy as 3%. Consequently, we present debiasing results that achieve the maximum reduction in bias degrees while simultaneously preserving acceptable accuracy levels.

RQ1: How effective the debias adapter is to mitigate group bias? To answer this question, we show the group bias degree as well as performance metrics after debiasing in Table 6.2. In our experiments, we train the debias adapters and downstream classifiers with Loss Function 6.4 and parameters shown in Equation 6.9 and Equation 6.10.

In Table 6.2, the second and third columns show the model performance metrics with respect to accuracy and F1-score. The fourth column presents the debiased group bias degree, denoted as δ_g. Additionally, we evaluate this group bias degree for each identity category, with details provided in the final three columns. The group bias degree δ_g for each identity category is presented in the first row of each dataset. Then, in the second row within brackets, we display

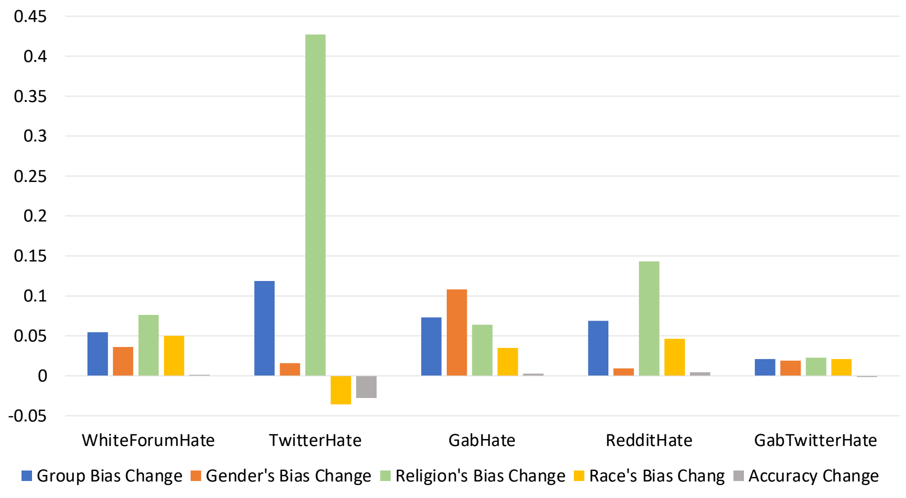


Figure 6.3: Group Bias 'δ_g' Changes and Accuracy Changes

the bias degrees δ^s for individual identities s. A positive δ^s signifies a higher FUR compared to the overall FUR and vice versa.

In general, the bias degree δ_g shows a reduction by more than 20% across all models. In certain instances, the mitigation in group bias degree exceeds 70%. For example, with the model fine-tuned on the *WhiteForumHate* dataset, the group bias degree decreases from 0.0749 to 0.0203 after debiasing. Similarly, for the model fine-tuned on the *RedditHate* dataset, δ_g reduces from 0.1060 to 0.0372 by 65% after debiasing. Across all models, accuracy and F1-score remain relatively stable after debiasing, with three models even recording an increase in accuracy. In a closer observation, the Religion and Race identity categories continue to exhibit a higher group bias compared to the Gender identity category after debiasing. Out of a total of 40 identities, 27 δ^s values are positive, suggesting that models have a tendency to misclassify samples associated with these identities as hate speeches.

To provide an in-depth analysis, we visualize the changes in group bias degree and accuracy compared to the baseline model in Figure 6.3. For each model, we show the overall changes in group bias (i.e., δ_g changes), changes in group bias for Gender, Religion, and Race identity categories, as well as accuracy changes. A value above 0 signifies an improvement. That is, regarding

Table 6.3: Debiasing with Regards to Individual Bias

Dataset	Accuracy	F1	δ_i	Gender	Religion	Race
WhiteForumHate	90.84%	0.9501	0.0827	0.0726	0.1273	0.0502
TwitterHate	91.50%	0.9121	0.1184	0.1126	0.0351	0.0492
GabHate	86.30%	0.8491	0.2239	0.0474	0.0841	0.3276
RedditHate	89.35%	0.8148	0.0812	0.0197	0.0926	0.2342
GabTwitterHate	76.44%	0.8252	0.2148	0.1153	0.2028	0.1585

biases, a bar above 0 denotes an improvement in group fairness (i.e., decreased δ_g). For accuracy changes, a bar above 0 indicates increased accuracy and vice versa. Generally, our approach successfully mitigates group bias for almost all identity categories concurrently. Except for a minor increase in the group bias for the Race category (by 0.03) for the model fine-tuned on *TwitterHate* dataset, all bias degrees exhibit improvements. Particularly noteworthy is the significant improvement on the Religion category across all models. Furthermore, the accuracy drops for all models are almost negligible when compared with the improvements in fairness.

RQ2: How effective the debias adapter is to mitigate individual bias? To answer this question, we show the individual bias degree as well as performance metrics after debiasing in Table 6.3. In our experiments, we train the debias adapters and downstream classifier with Loss Function 6.4 and parameters shown in Equation 6.11.

In Table 6.3, the second and third columns display the model performance metrics with respect to accuracy and F1-score. The fourth column presents the debiased individual bias degree δ_i. Similar with RQ1, we evaluate the individual bias degree for each identity category, with details provided in the final three columns.

In general, the individual bias degree improves by more than 20% across all models. Particularly, for the model fine-tuned on the *WhiteForumHate* dataset, δ_i decreases by 58% from 0.1962 to 0.0827. Notably, after debiasing, accuracy and F1-scores stay consistent for all models, with the accuracy of one model

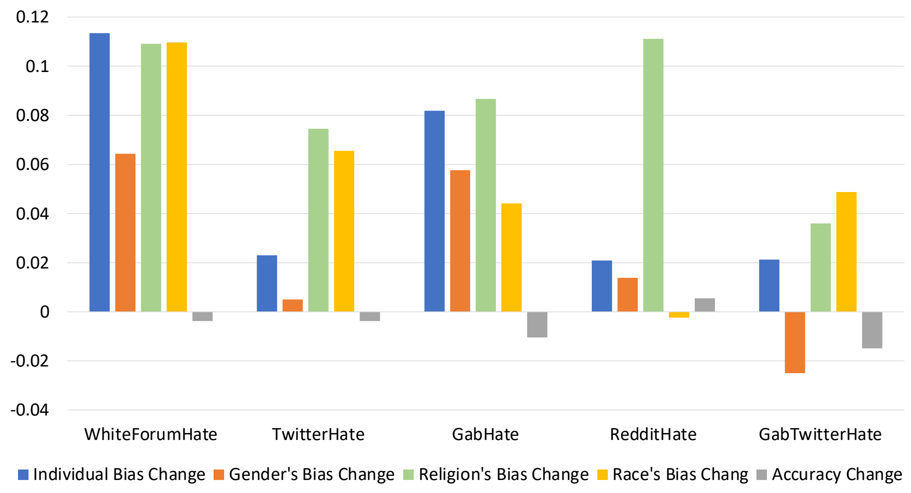


Figure 6.4: Individual Bias 'δ_i' and Accuracy Changes

even showing an increase. The individual bias δ_i still varies across different identity categories after debiasing. Most of the models exhibit higher individual bias on Religion and Race categories.

In Figure 6.4, we visualize the changes in individual bias degree and accuracy compared to the baseline model. For each model, we show the overall changes in individual bias (i.e., δ_i), changes in individual bias for Gender, Religion and Race identity categories, as well as accuracy changes. As in previous discussion, a value above 0 denotes an improvement. The results demonstrate successful mitigation of individual bias across nearly all identity categories concurrently. Except for the Race category in *RedditHate* dataset and Gender category in *GabTwitterHate* dataset, all individual bias δ_i decrease after debiasing. Specifically, for the model fine-tuned on the *WhiteForumHate* dataset, the individual bias for all categories decreases by more than 0.06. Remarkably, a significant reduction in individual bias is observed within the Religion category across all models. Furthermore, the accuracy drops for all models are relatively minor, remaining under 2% after debiasing.

RQ3: How much weight we should place on fairness? To answer this question, we analyze the changes in bias degree and accuracy after debiasing, while adjusting parameter λ. The parameter λ, as denoted in Equation 6.4, serves as the weighting factor determining the influence of bias loss. A larger value assigns more weight to the bias loss. In our experiments, we adjust λ to control the balance between achieving bias reduction and maintaing accuracy.

During the process of mitigating group bias, we noticed that adjusting the value of λ did not yield an obvious trade-off between reducing group bias and experiencing a decrease in accuracy. Consequently, we empirically determined the optimal value of λ, with the objective of minimizing δ_g while maintaining an acceptable level of accuracy. This absence of significant trade-off might be explained by the fact that the group bias degree is based on FUR and thus there is a strong correlation between group bias loss and accuracy loss.

In the context of individual bias mitigation, we noted a clear trade-off between the reduction of bias and the drop in accuracy. As a result, determining the optimum value of λ for these convex curves becomes straightforward through golden section search [115]. Figure 6.5 visualizes how the individual bias degree δ_i changes with the increase in λ. As observed, δ_i decreases as λ increases from 0 to 1.5. Notably, in certain scenarios where λ is sufficiently large, δ_i drops to 0.

In Figure 6.6, we provide a more detailed visualization of the changes in 'Improved δ_i - Acc Drop', which represents the improvement in individual bias degree minus the accuracy drop, as we increase the value of λ. Notably, the value exhibits a steady increase initially, followed by a sudden drop when λ reaches a high value. One should note that, for the model fine-tuned on *White-ForumHate* dataset, the data is highly unbalanced, with over 90% of the data labeled as positive. This imbalance means that even if the debiased model inappropriately predicted all samples as positive, the accuracy could still achieve 88.0%. Consequently, the value 'Improved δ_i - Acc Drop' hits its peak when λ equals 1.4.

RQ4: How effective is our approach compared with related works? To answer this question, we compare the efficacy of our methods with that of other related

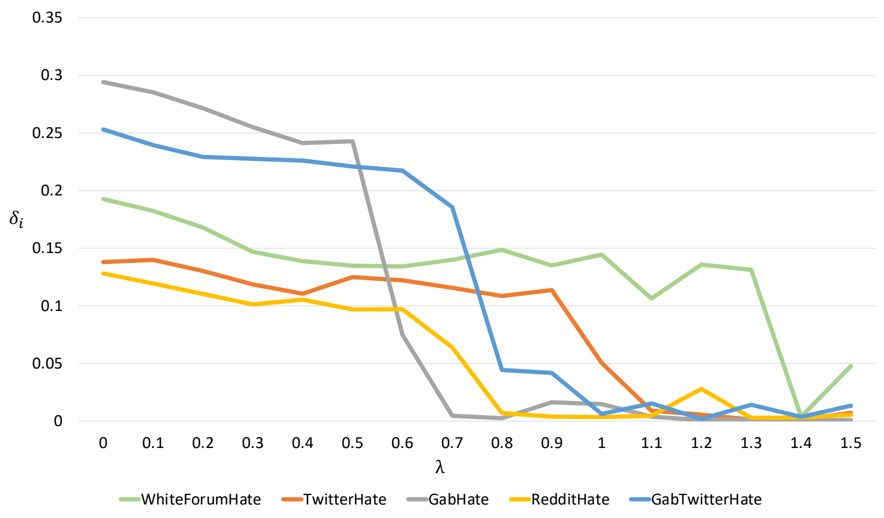


Figure 6.5: 'δ_i' Changes with Increasing λ

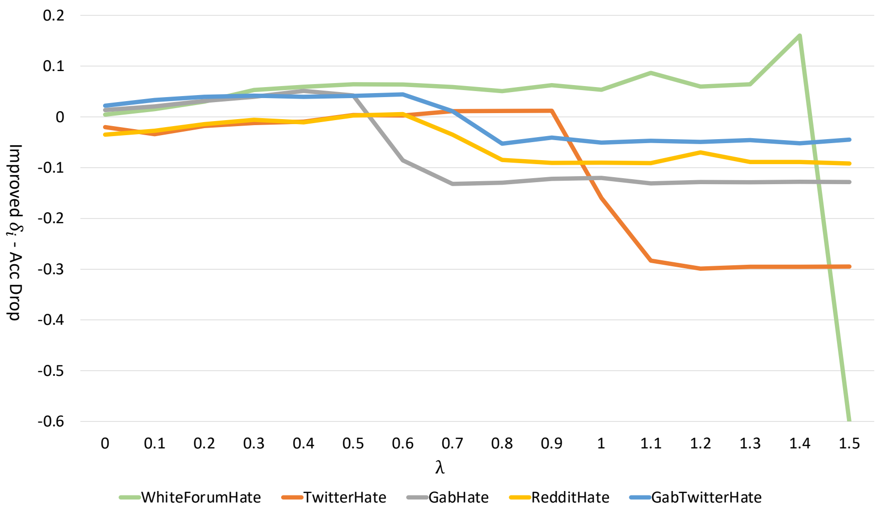


Figure 6.6: 'Improved δ_i - Acc Drop' Changes with Increasing λ

works. As our approach is essentially a post-processing method, which does not access or modify the internals of language models, we exclusively consider post-processing debiasing techniques as related works. In other words, the compared method cannot gain access to or modify the internals of baseline BERT models. Overall, we take into account three post-hoc debiasing techniques as baselines, two aimed at group bias mitigation and one targeting individual bias mitigation.

For group bias mitigation, we consider two well-established techniques. The first one is 'Calibrated Equalized Odds' (CEO) [123], which mitigates biases by optimizing calibrated classifier score outputs. It searches for probabilities with which to change output labels with an equalized odds objective. The second baseline is ‘Reject Option Classification’ (ROC) [72], a method that assigns favorable labels to unprivileged instances and unfavorable labels to privileged instances around the decision boundary with the highest uncertainty. The implementations of these bias mitigation techniques in our work are based on the AIF360 framework [14]. As both methods mitigate bias through pre-defined privileged and unprivileged groups based on tabular data, we extend the implementation to textual data and define the unprivileged group according to the baseline group bias δ^s for each individual identity s. Accordingly, samples associated with identities exhibiting the highest δ^s are classified as the unprivileged group. For example, in the *WhiteForumHate* dataset, the unprivileged group comprises *homosexual, muslim, black*, while the privileged group includes the remaining identities.

The group bias degree and accuracy changes are shown in Table 6.4. We denote a decrease in value by the symbol ‘$\downarrow$’, hence, a positive $\delta_g \downarrow$ signifies an improvement in group fairness. Conversely, the symbol ‘$\uparrow$’ denotes an increase in value, so a positive ‘Acc $\uparrow$’ indicates an increase in fairness and vice versa. We can observe that CEO and ROC methods do not consistently mitigate group bias across all models. In some instances, these methods even negatively affect group fairness. Moreover, almost all models exhibit a drop in accuracy after debiasing. To evaluate the balance between debiasing effect and accuracy compromise, we visualize the ‘improved δ_g - Acc Drop’ values in Figure 6.7. Notably, our method shows the most effective group bias mitigation while maintaining accuracy across all models.

For individual bias mitigation, we compare our results with models that have been retrained using idis, which is commonly used to mitigate individual bias in existing works [176, 7]. Note that, similar to our setting, the comparison method cannot modify the baseline BERT models. Consequently, it only trains on the debias adapter and classifier subsequent to the BERT encoder. In our experiments, we generate idis from the training data and use these along with

Table 6.4: Group Bias Degree Changes After Debiasing by Related Works and Ours

Dataset	CEO		ROC		Ours	
	$\delta_g\downarrow$	Acc↑	$\delta_g\downarrow$	Acc↑	$\delta_g\downarrow$	Acc↑
WhiteForumHate	0.0005	0.03%	0.0713	-3.24%	0.0546	0.12%
TwitterHate	-0.1011	-0.25%	-0.0058	-23.59%	0.1186	-2.78%
GabHate	0.0399	-0.58%	0.0523	-0.45%	0.0732	0.27%
RedditHate	0.0458	-0.89%	0.0059	-2.20%	0.0687	0.46%
GabTwitterHate	-0.0051	-0.51%	0.0139	-3.19%	0.0209	-0.15%

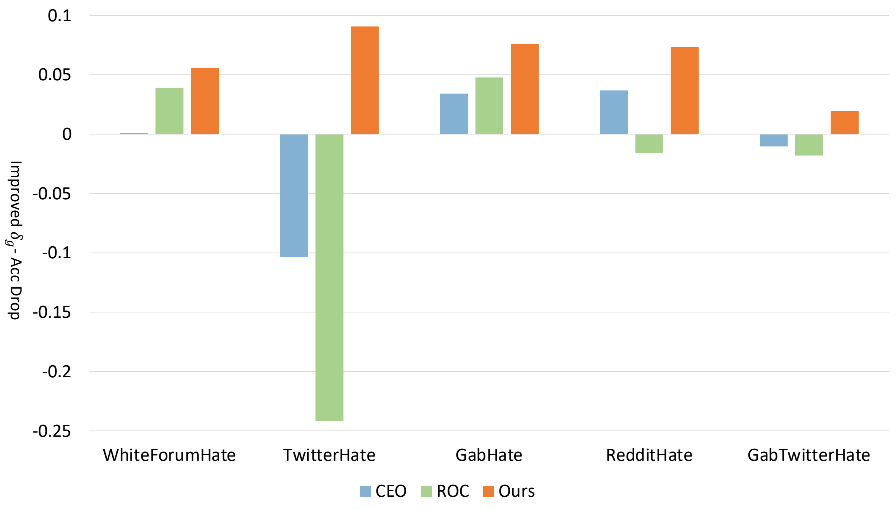


Figure 6.7: 'Improved δ_g - Acc Drop' Changes

the original training data to retrain debias adapters and classifiers.

We present the changes in individual bias degree and accuracy in Table 6.5. We can observe that while retraining with idi improves individual fairness for two models, it leads to an obvious decrease in accuracy for almost all models. In Figure 6.8, we visualize the 'improved δ_i - Acc Drop' values. In alignment with the previous discussion, our method consistently displays the most effective mitigation on individual bias, while maintaining accuracy across all models.

6.6 Related Work

This work is related to research on fairness testing and verification, bias mitigation methods as well as broadly various studies on textual bias. Besides those mentioned in the previous sections, we summarize other related works below.

Table 6.5: Individual Bias Degree Changes After Debiasing by Related Works and Ours

Dataset	**Train with idis**		**Ours**	
	$\delta_i \downarrow$	Acc$\uparrow$	$\delta_i \downarrow$	Acc$\uparrow$
WhiteForumHate	0.0686	0.09%	0.1135	-0.37%
TwitterHate	-0.0888	-1.24%	0.0230	-0.38%
GabHate	-0.0041	-3.51%	0.0818	-1.04%
RedditHate	-0.0632	-2.92%	0.0209	0.55%
GabTwitterHate	0.0224	-1.85%	0.0213	-1.49%

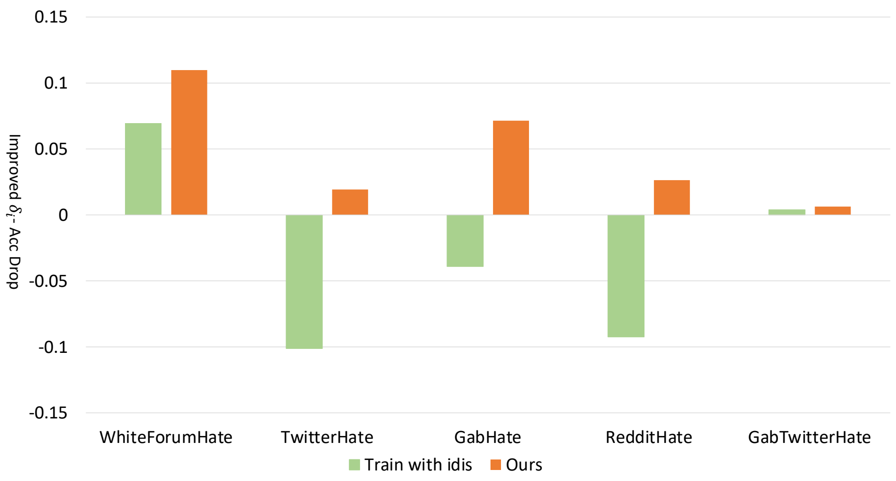


Figure 6.8: 'Improved δ_i - Acc Drop' Changes

Fairness Testing and Verification There exist various works trying to detect bias in NLP models in different angles. In the word embedding association test (WEAT), various studies show the stereotypical biases in word level [102, 26, 134]. Lu et al. show that NLP systems exists more correlations on gender/occupation stereotypes such as he/doctor [96] and Aylin et al. demonstrate the tight association between gender-related words and career or family attributes in GloVe and word2vec word embeddings [26]. Sheng et al. also show that there are more negative associations of the black population for context related to occupation [134]. Liang et al. demonstrate that certain harmful tokens, such as "terrorists" and "murder," are more likely to associate with specific classes, such as Muslims [90]. Then, May et al. extend WEAT to sentence embedding association test (SEAT) and test social biases on sentence encoders, e.g., ELMo and BERT [105]. There are also works on testing pre-trained LLMs to explore the extrinsic bias. Soares et al. study how group fairness varies across different fine-tuned LMs for binary classification [138]. Marta et al. try to explore whether language models' architectures, when trained with the same data, influence the level of gender bias [34]. They also propose interpretability analysis to interpret gender bias in neural machine translation models. Abid et al. focus on GPT-3 to investigate religious bias, i.e., anti-Muslim bias, in various tasks [2]. Jentzsch et al. analyze gender bias in BERT models trained on sentiment rating dataset [68].

For fairness verification, we try to have a statistical confident conclusion in every single batch. There also exist other works verifying fairness metrics. Bastani et al. contribute a novel approach for probabilistic verification of fairness properties in machine learning systems, leveraging concentration bounds and logical specifications [13].

Bias Mitigation To mitigate LM biases, common approaches include modifying the training data through data augmentation, manipulating word or sentence representations and other in-processing methods.

For Data Augmentation, Kusner et al. first propose counterfactual fairness, which treated different samples equally in actual and counterfactual demographics groups [84]. Zhao et al. mitigate gender bias by augmenting original data with gender swapping and retrain the model [180]. Some existing works focus on word embeddings [49]. Bolukbasi et al. mitigate gender bias in Word2Vec [19]. Mikolov et al. propose an improved method called GN-GLoVe, which separate the GloVe [120] embedding space into neutral and gender dimensions [106]. Then they train the model with a modified loss function to obtain gender-neutral embeddings. He et al. focus on gender biases and propose a bias mitigation technique that creates informative and fair contextualized representations based on natural language reference data [63]. Lastly, typical in-processing methods for bias mitigation include retraining or fine-tuning the given model with loss constraints, and modifying the model structure or training process. Guo et al. focus on gender and racial bias and propose an automatic method to mitigate the bias for masked language models through fine-tuning with distribution alignment loss on biased prompts [57]. Zhang et al. use adversarial networks to prevent the discriminator from identifying gender in an analogy completion task [168]. Reinforcement learning is also included to mitigate biases of LMs. Liu et al. propose a reinforcement learning framework for mitigating political biases without having access to the training data or requiring the model to be retrained [91]. Subramanian et al. consider 'gerrymandering' groups and propose approaches to predict outputs without the influence of social identities encoded in hidden representation. It focuses on *equality of opportunity* [162], i.e., average-violations of true positive rate as bias metric.

6.7 Summary

In this work, we introduce a non-intrusive bias mitigation strategy that carries three significant benefits. First, it does not access or modify the internals of language models and debiasing in a parameter-efficient way. This method is particularly valuable in the current context, where most LLMs are becoming increasingly complex and are not intrinsically accessible. Second, it provides a theoretical guarantee at each step during debiasing. Third, our approach improves fairness at little cost of accuracy.

Chapter 7

Conclusion and Future Work

In this chapter, we conclude the dissertation, first with some conclusions and then with some ideas for intriguing future work.

7.1 Conclusions

At the beginning of this dissertation, we elaborated on the background of state-of-the-art machine learning models, including neural networks and LLMs as well as the potential issues arise. Then, we aimed to improve people's trust in machine learning models in two aspects: interpretability and fairness. Specifically, we proposed multiple approaches and toolkits to systematically test model's interpretability, fairness and also proposed novel methods to mitigate discrimination with regard to both tabular data and textual data. The contributions of this dissertation are as follows.

First of all, we advanced the field of neural network interpretability by proposing multiple definitions and developing algorithms to systematically evaluate interpretability. Our approach hinged on measuring fidelity against decision tree with a predefined height limit, providing a framework for assessing model interpretability. The decision trees allowed us to provide reasonable expression where humans can reason about the decision making process naturally. We also evaluated the relationship between robustness and interpretability. lastly, we proposed a method to improve the decision explainability of a model through retraining.

Second, we narrowed the interpretability problem into a specific case, i.e., fairness. We noticed that group discrimination can manifest in subtle ways, where it may be hidden and not immediately apparent. This can occur when certain groups, such as males of a specific age range or individuals with specific combinations of sensitive features like race and gender, experience discrimination based on unknown and nuanced conditions. So, in the second work, we focused on testing neural networks for these subtle group discriminations. We introduced a framework to systematically identify interpretable instances of such discrimination. Specifically, the identified subtle discrimination comes with a fairness score which is supported by a specified confidence level. Our extensive evaluation uncovered the existence of subtle group discrimination in neural networks, emphasizing the critical need for mitigation strategies. By leveraging our testing results to augment training data, we demonstrated the potential for mitigating these subtle discriminations.

Thirdly, we focused on fairness improvement methods and undertaked a thorough empirical evaluation of nine fairness-improving methods across four real-world datasets and a diverse set of model-attribute combinations, employing three distinct fairness metrics. Notably, our findings revealed that existing methods do not consistently enhance group fairness and often fall short of reducing individual discrimination. Motivated by this empirical insight, we proposed a lightweight, causality-based approach to dynamically select the most suitable fairness improvement method. By evaluating the distribution of ”responsible” attributes and neurons, we achieved notable success in optimizing fairness improvements.

Lastly, with the significant development of LLMs, we observed that it is necessary to test and mitigate biases with regard to language models. Here, we introduced a non-intrusive bias mitigation strategy, which operated without the need to access or modify the internal workings of language models. This advantage presented a parameter-efficient and practical solution for increasingly complex language models. It also provided theoretical guarantees at each debiasing step, ensuring a principled approach to fairness enhancement. Additionally, our method substantially improved fairness with minimal impact on accuracy, underscoring its potential as a powerful tool in the pursuit of equitable AI systems.

7.2 Future Work

Our experimental results demonstrate that our proposed methods perform well in terms of both effectiveness and efficiency. In future work, we want to investigate the following directions.

Fairness testing and bias mitigation on LLMs First, In our research work, we focus on extrinsic bias, which is calculated according to downstream tasks. However, it is necessary to develop a tool to systematically test both extrinsic bias and intrinsic bias within the LLMs. Typically, extrinsic bias refers to biases that are introduced into the model's output and intrinsic bias refers to biases that are inherent in the architecture or embedded in the tokens. Testing both extrinsic and intrinsic biases provides a more comprehensive assessment of a model's behavior. It allows for a nuanced understanding of the sources of bias and how they manifest. It is also essential for building AI systems that align with ethical regulations. Additionally, like our third research work, where we evaluate the reason of discrimination to select the optimal fairness improvement method, different types of biases may require different mitigation approaches within LLMs. Especially, when the LLMs become more complex and hard to access for normal users, it is imperative to develop more technologies to reduce different types of biases without accessing the internals of the model or the training data.

Fairness testing for more applications Our research works primarily focus on tabular data in the context of deep neural networks and textual data in the context of language models. However, our examination is limited to classification problems, where fairness concerns have been widely defined. Fairness problems, as a critical non-functional property, should be considered across a wider array of systems, such as recommendation systems, speech recognition systems, text generation systems, etc. For these diverse systems, it becomes imperative to propose more comprehensive definitions of fairness and develop tailored tools. It enables us to create a fairer and safer environment for the development of machine learning technologies.

List of References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, pages 265–283, 2016.

[2] Abubakar Abid, Maheen Farooqi, and James Zou. Persistent anti-muslim bias in large language models. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pages 298–306, 2021.

[3] Oludare Isaac Abiodun, Aman Jantan, Abiodun Esther Omolara, Kemi Victoria Dada, Nachaat AbdElatif Mohamed, and Humaira Arshad. State-of-the-art in artificial neural network applications: A survey. *Heliyon*, 4(11):e00938, 2018.

[4] Julius Adebayo and Lalana Kagal. Iterative orthogonal feature projection for diagnosing bias in black-box models. *CoRR*, abs/1611.04967, 2016.

[5] Alekh Agarwal, Alina Beygelzimer, Miroslav Dudík, John Langford, and Hanna Wallach. A reductions approach to fair classification. In *International Conference on Machine Learning*, pages 60–69. PMLR, 2018.

[6] Alekh Agarwal, Miroslav Dudík, and Zhiwei Steven Wu. Fair regression: Quantitative definitions and reduction-based algorithms. In *International Conference on Machine Learning*, pages 120–129. PMLR, 2019.

[7] Aniya Aggarwal, Pranay Lohia, Seema Nagar, Kuntal Dey, and Diptikalyan Saha. Automated test generation to detect individual discrimination in AI models. *CoRR*, abs/1809.03260, 2018.

[8] Aniya Aggarwal, Pranay Lohia, Seema Nagar, Kuntal Dey, and Diptikalyan Saha. Black box fairness testing of machine learning models. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 625–635, 2019.

[9] Rico Angell, Brittany Johnson, Yuriy Brun, and Alexandra Meliou. Themis: Automatically testing software for discrimination. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 871–875, 2018.

[10] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. Machine bias: There's software used across the country to predict future criminals. and it's biased against blacks. *ProPublica*, 2016. `https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing`.

[11] Lisa C Anthony and Mei Liu. Analysis of differential prediction of law school performance by racial/ethnic subgroups based on the 1996-1998 entering law school classes. lsac research report series. 2003.

[12] Muhammad Hilmi Asyrofi, Imam Nur Bani Yusuf, Hong Jin Kang, Ferdian Thung, Zhou Yang, and David Lo. Biasfinder: Metamorphic test generation to uncover bias for sentiment analysis systems. *CoRR*, abs/2102.01859, 2021.

[13] Osbert Bastani, Xin Zhang, and Armando Solar-Lezama. Probabilistic verification of fairness properties via concentration. *Proceedings of the ACM on Programming Languages*, 3(OOPSLA):1–27, 2019.

[14] Rachel KE Bellamy, Kuntal Dey, Michael Hind, Samuel C Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Lohia, Jacquelyn Martino, Sameep Mehta, Aleksandra Mojsilović, et al. Ai fairness 360: An extensible toolkit for detecting and mitigating algorithmic bias. *IBM Journal of Research and Development*, 63(4/5):4–1, 2019.

[15] Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 610–623, 2021.

[16] Sumon Biswas and Hridesh Rajan. Do the machine learning models on a crowd sourced platform exhibit bias? an empirical study on model fairness. In *Proceedings of the 28th ACM joint meeting on European software engineering conference and symposium on the foundations of software engineering*, pages 642–653, 2020.

[17] Su Lin Blodgett and Brendan O'Connor. Racial disparity in natural language processing: A case study of social media african-american english. *CoRR*, abs/1707.00061, 2017.

[18] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars. *CoRR*, abs/1604.07316, 2016.

[19] Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. *Advances in neural information processing systems*, 29, 2016.

[20] Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. Nuanced metrics for measuring unintended bias with real data for text classification. In *Companion proceedings of the 2019 world wide web conference*, pages 491–500, 2019.

[21] Eric Breck, Neoklis Polyzotis, Sudip Roy, Steven Whang, and Martin Zinkevich. Data validation for machine learning. In *MLSys*, 2019.

[22] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.

[23] Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on fairness, accountability and transparency*, pages 77–91. PMLR, 2018.

[24] Zhaowei Cai, Xiaodong He, Jian Sun, and Nuno Vasconcelos. Deep learning with low precision by half-wave gaussian quantization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5918–5926, 2017.

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

[26] Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186, 2017.

[27] Flavio Calmon, Dennis Wei, Bhanukiran Vinzamuri, Karthikeyan Natesan Ramamurthy, and Kush R Varshney. Optimized pre-processing for discrimination prevention. *Advances in neural information processing systems*, 30, 2017.

[28] Diogo V Carvalho, Eduardo M Pereira, and Jaime S Cardoso. Machine learning interpretability: A survey on methods and metrics. *Electronics*, 8(8):832, 2019.

[29] L Elisa Celis, Lingxiao Huang, Vijay Keswani, and Nisheeth K Vishnoi. Classification with fairness constraints: A meta-algorithm with provable guarantees. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 319–328, 2019.

[30] Joymallya Chakraborty, Tianpei Xia, Fahmid M. Fahid, and Tim Menzies. Software engineering for fairness: A case study with hyperparameter optimization. *CoRR*, abs/1905.05786, 2019.

[31] Aditya Chattopadhyay, Piyushi Manupriya, Anirban Sarkar, and Vineeth N Balasubramanian. Neural network attributions: A causal perspective. In *International Conference on Machine Learning*, pages 981–990. PMLR, 2019.

[32] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[33] Aaron Clauset. A brief primer on probability distributions. In *Santa Fe Institute*, 2011.

[34] Marta R Costa-jussà, Carlos Escolano, Christine Basta, Javier Ferrando, Roser Batlle, and Ksenia Kharitonova. Interpreting gender bias in neural machine translation: Multilingual architecture matters. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11855–11863, 2022.

[35] Piotr Dabkowski and Yarin Gal. Real time image saliency for black box classifiers. In *Advances in Neural Information Processing Systems*, pages 6967–6976, 2017.

[36] Ona de Gibert, Naiara Perez, Aitor García-Pablos, and Montse Cuadros. Hate Speech Dataset from a White Supremacy Forum. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 11–20, Brussels, Belgium, October 2018. Association for Computational Linguistics.

[37] Deepti Deshwal, Pardeep Sangwan, and Divya Kumar. Feature extraction methods in language identification: a survey. *Wireless Personal Communications*, 107(4):2071–2103, 2019.

[38] Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. Measuring and mitigating unintended bias in text classification. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pages 67–73, 2018.

[39] Jonathan Dodge, Q Vera Liao, Yunfeng Zhang, Rachel KE Bellamy, and Casey Dugan. Explaining models: an empirical study of how explanations impact fairness judgment. In *Proceedings of the 24th international conference on intelligent user interfaces*, pages 275–285, 2019.

[40] Weymark John A Donaldson David. A single-parameter generalization of the gini indices of inequality. *Journal of economic Theory*, 22(1):67–86, 1980.

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

[42] Jeroen Eggermont, Joost N Kok, and Walter A Kosters. Genetic programming for data classification: Partitioning the search space. In *Proceedings of the 2004 ACM symposium on Applied computing*, pages 1001–1005, 2004.

[43] Gregory J Feist. Quantity, quality, and depth of research as influences on scientific eminence: Is quantity most important? *Creativity Research Journal*, 10(4):325–335, 1997.

[44] Michael Feldman, Sorelle A Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and removing disparate impact. In *proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 259–268, 2015.

[45] Paula Fortuna and Sérgio Nunes. A survey on automatic detection of hate speech in text. *ACM Computing Surveys (CSUR)*, 51(4):1–30, 2018.

[46] Sorelle A Friedler, Carlos Scheidegger, Suresh Venkatasubramanian, Sonam Choudhary, Evan P Hamilton, and Derek Roth. A comparative study of fairness-enhancing interventions in machine learning. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 329–338, 2019.

[47] Sainyam Galhotra, Yuriy Brun, and Alexandra Meliou. Fairness testing: testing software for discrimination. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, pages 498–510, 2017.

[48] Raul Vicente Garcia, Lukasz Wandzik, Louisa Grabner, and Joerg Krueger. The harms of demographic bias in deep face recognition research. In *2019 International Conference on Biometrics (ICB)*, pages 1–6. IEEE, 2019.

[49] Nikhil Garg, Londa Schiebinger, Dan Jurafsky, and James Zou. Word embeddings quantify 100 years of gender and ethnic stereotypes. *Proceedings of the National Academy of Sciences*, 115(16):E3635–E3644, 2018.

[50] Andrew Gaut, Tony Sun, Shirlyn Tang, Yuxin Huang, Jing Qian, Mai ElSherief, Jieyu Zhao, Diba Mirza, Elizabeth Belding, Kai-Wei Chang, and William Yang Wang. Towards understanding gender bias in relation extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2943–2953, Online, July 2020. Association for Computational Linguistics.

[51] Bishwamittra Ghosh, Debabrota Basu, and Kuldeep S Meel. Justicia: A stochastic sat approach to formally verify fairness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7554–7563, 2021.

[52] Milena A Gianfrancesco, Suzanne Tamang, Jinoos Yazdany, and Gabriela Schmajuk. Potential biases in machine learning algorithms using electronic health record data. *JAMA internal medicine*, 178(11):1544–1547, 2018.

[53] Seraphina Goldfarb-Tarrant, Rebecca Marchant, Ricardo Muñoz Sánchez, Mugdha Pandya, and Adam Lopez. Intrinsic bias metrics do

not correlate with application bias. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 1926–1940. Association for Computational Linguistics, 2021.

[54] Ian J. Goodfellow. A research agenda: Dynamic models to defend against correlated attacks. *CoRR*, abs/1903.06293, 2019.

[55] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[56] Pei Guo, Connor Anderson, Kolten Pearson, and Ryan Farrell. Neural network interpretation via fine grained textual summarization. *CoRR*, abs/1805.08969, 2018.

[57] Yue Guo, Yi Yang, and Ahmed Abbasi. Auto-debias: Debiasing masked language models with automated biased prompts. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1012–1023, 2022.

[58] Maya R. Gupta, Andrew Cotter, Mahdi Milani Fard, and Serena Lutong Wang. Proxy fairness. *CoRR*, abs/1806.11212, 2018.

[59] Rami Haffar, Ashneet Khandpur Singh, Josep Domingo-Ferrer, and Najeeb Jebreel. Measuring fairness in machine learning models via counterfactual examples. In Vicenç Torra and Yasuo Narukawa, editors, *Modeling Decisions for Artificial Intelligence - 19th International Conference, MDAI 2022, Sant Cugat, Spain, August 30 - September 2, 2022, Proceedings*, volume 13408 of *Lecture Notes in Computer Science*, pages 119–131. Springer, 2022.

[60] Moritz Hardt, Eric Price, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

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

[62] Galen Harrison, Julia Hanson, Christine Jacinto, Julio Ramirez, and Blase Ur. An empirical study on the perceived fairness of realistic, imperfect machine learning models. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*, pages 392–402, 2020.

[63] Jacqueline He, Mengzhou Xia, Christiane Fellbaum, and Danqi Chen. MABEL: Attenuating gender bias using textual entailment data. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9681–9702, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[64] Hans Hofmann. German credit dataset. 1994. `https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data)`.

[65] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019.

[66] Marianne Huchard, Christian Kästner, and Gordon Fraser. Proceedings of the 33rd acm/ieee international conference on automated software engineering (ase 2018). In *ASE: Automated Software Engineering*. ACM Press, 2018.

[67] Anil K Jain, Robert P. W. Duin, and Jianchang Mao. Statistical pattern recognition: A review. *IEEE Transactions on pattern analysis and machine intelligence*, 22(1):4–37, 2000.

[68] Sophie Jentzsch and Cigdem Turan. Gender bias in bert-measuring and analysing biases through sentiment rating in a realistic downstream classification task. In *Proceedings of the 4th Workshop on Gender Bias in Natural Language Processing (GeBNLP)*, pages 184–199, 2022.

[69] Matthew Joseph, Michael Kearns, Jamie H Morgenstern, and Aaron Roth. Fairness in learning: Classic and contextual bandits. *Advances in neural information processing systems*, 29, 2016.

[70] Kristy N Kamarck. Diversity, inclusion, and equal opportunity in the armed services: Background and issues for congress. 2017.

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

[72] Faisal Kamiran, Asim Karim, and Xiangliang Zhang. Decision theory for discrimination-aware classification. In *2012 IEEE 12th International Conference on Data Mining*, pages 924–929. IEEE, 2012.

[73] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. Fairness-aware classifier with prejudice remover regularizer. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 35–50. Springer, 2012.

[74] Enkelejda Kasneci, Kathrin Seßler, Stefan Küchemann, Maria Bannert, Daryna Dementieva, Frank Fischer, Urs Gasser, Georg Groh, Stephan Günnemann, Eyke Hüllermeier, et al. Chatgpt for good? on opportunities

and challenges of large language models for education. *Learning and Individual Differences*, 103:102274, 2023.

[75] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in neural information processing systems*, pages 3146–3154, 2017.

[76] Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness. In *International Conference on Machine Learning*, pages 2564–2572. PMLR, 2018.

[77] Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. An empirical study of rich subgroup fairness for machine learning. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 100–109, 2019.

[78] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186, 2019.

[79] Been Kim, Cynthia Rudin, and Julie A Shah. The bayesian case model: A generative approach for case-based reasoning and prototype classification. In *Advances in neural information processing systems*, pages 1952–1960, 2014.

[80] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, pages 2668–2677, 2018.

[81] Jon M. Kleinberg, Sendhil Mullainathan, and Manish Raghavan. Inherent trade-offs in the fair determination of risk scores. In *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*, volume 67 of *LIPIcs*, pages 43:1–43:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

[82] Igor Kononenko. Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in medicine*, 23(1):89–109, 2001.

[83] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *Artificial intelligence safety and security*, pages 99–112. Chapman and Hall/CRC, 2018.

[84] Matt J Kusner, Joshua Loftus, Chris Russell, and Ricardo Silva. Counterfactual fairness. *Advances in neural information processing systems*, 30, 2017.

[85] Isaac Lage, Emily Chen, Jeffrey He, Menaka Narayanan, Been Kim, Sam Gershman, and Finale Doshi-Velez. An evaluation of the human-interpretability of explanation. *CoRR*, abs/1902.00006, 2019.

[86] Preethi Lahoti, Alex Beutel, Jilin Chen, Kang Lee, Flavien Prost, Nithum Thain, Xuezhi Wang, and Ed H. Chi. Fairness without demographics through adversarially reweighted learning. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[87] Himabindu Lakkaraju, Stephen H Bach, and Jure Leskovec. Interpretable decision sets: A joint framework for description and prediction. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1675–1684, 2016.

[88] Afshan Latif, Aqsa Rasheed, Umer Sajid, Jameel Ahmed, Nouman Ali, Naeem Iqbal Ratyal, Bushra Zafar, Saadat Hanif Dar, Muhammad Sajid, and Tehmina Khalil. Content-based image retrieval and feature extraction: a comprehensive review. *Mathematical Problems in Engineering*, 2019, 2019.

[89] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, 2020.

[90] Paul Pu Liang, Chiyu Wu, Louis-Philippe Morency, and Ruslan Salakhutdinov. Towards understanding and mitigating social biases in language models. In *International Conference on Machine Learning*, pages 6565–6576. PMLR, 2021.

[91] Ruibo Liu, Chenyan Jia, Jason Wei, Guangxuan Xu, Lili Wang, and Soroush Vosoughi. Mitigating political bias in language models through reinforced calibration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14857–14866, 2021.

[92] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[93] Wei-Yin Loh. Classification and regression trees. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(1):14–23, 2011.

[94] Tania Lombrozo. The structure and function of explanations. *Trends in cognitive sciences*, 10(10):464–470, 2006.

[95] Tania Lombrozo. Simplicity and probability in causal explanation. *Cognitive psychology*, 55(3):232–257, 2007.

[96] Kaiji Lu, Piotr Mardziel, Fangjing Wu, Preetam Amancharla, and Anupam Datta. Gender bias in neural natural language processing. *Logic, Language, and Security: Essays Dedicated to Andre Scedrov on the Occasion of His 65th Birthday*, pages 189–202, 2020.

[97] Qinghua Lu, Liming Zhu, Xiwei Xu, Jon Whittle, David Douglas, and Conrad Sanderson. Software engineering for responsible ai: An empirical study and operationalised patterns. In *Proceedings of the 44th International Conference on Software Engineering: Software Engineering in Practice*, pages 241–242, 2022.

[98] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Proceedings of the 31st international conference on neural information processing systems*, pages 4768–4777, 2017.

[99] Pingchuan Ma, Shuai Wang, and Jin Liu. Metamorphic testing and certified mitigation of fairness violations in nlp models. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI). 458–465*, 2020.

[100] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[101] Spyros Makridakis. The forthcoming artificial intelligence (ai) revolution: Its impact on society and firms. *Futures*, 90:46–60, 2017.

[102] Thomas Manzini, Yao Chong Lim, Alan W. Black, and Yulia Tsvetkov. Black is to criminal as caucasian is to police: Detecting and removing multiclass bias in word embeddings. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 615–621. Association for Computational Linguistics, 2019.

[103] Binny Mathew, Ritam Dutt, Pawan Goyal, and Animesh Mukherjee. Spread of hate speech in online social media. In *Proceedings of the 10th ACM conference on web science*, pages 173–182, 2019.

[104] Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. Hatexplain: A benchmark dataset for explainable hate speech detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14867–14875, 2021.

[105] Chandler May, Alex Wang, Shikha Bordia, Samuel Bowman, and Rachel Rudinger. On measuring social biases in sentence encoders. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 622–628, 2019.

[106] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[107] Mazin Abed Mohammed, Mohd Khanapi Abd Ghani, Net al Arunkumar, Raed Ibraheem Hamed, Salama A Mostafa, Mohamad Khir Abdullah, and MA Burhanuddin. Decision support system for nasopharyngeal carcinoma discrimination from endoscopic images using artificial neural network. *The Journal of Supercomputing*, 76(2):1086–1104, 2020.

[108] Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15, 2018.

[109] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582, 2016.

[110] Sérgio Moro, Paulo Cortez, and Paulo Rita. A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems*, 62:22–31, 2014. `https://archive.ics.uci.edu/ml/datasets/bank+marketing`.

[111] W. James Murdoch, Chandan Singh, Karl Kumbier, Reza Abbasi-Asl, and Bin Yu. Definitions, methods, and applications in interpretable machine learning. *Proceedings of the National Academy of Sciences*, 116(44):22071–22080, 2019.

[112] Moin Nadeem, Anna Bethke, and Siva Reddy. StereoSet: Measuring stereotypical bias in pretrained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5356–5371, Online, August 2021. Association for Computational Linguistics.

[113] Nikita Nangia, Clara Vania, Rasika Bhalerao, and Samuel R. Bowman. CrowS-pairs: A challenge dataset for measuring social biases in masked language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1953–1967, Online, November 2020. Association for Computational Linguistics.

[114] Mahdi Nejadgholi and Jinqiu Yang. A study of oracle approximations in testing deep learning libraries. In *2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 785–796. IEEE, 2019.

[115] Jorge Nocedal and Stephen J Wright. *Numerical optimization*. Springer, 1999.

[116] Mike Oaksford and Nick Chater. The probabilistic approach to human reasoning. *Trends in cognitive sciences*, 5(8):349–357, 2001.

[117] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[118] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[119] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. Deepxplore: Automated whitebox testing of deep learning systems. In *proceedings of the 26th Symposium on Operating Systems Principles*, pages 1–18, 2017.

[120] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[121] Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of causal inference: foundations and learning algorithms*. The MIT Press, 2017.

[122] Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. Adapterfusion: Non-destructive task composition for transfer learning. In *16th Conference of the European Chapter of the Associationfor Computational Linguistics, EACL 2021*, pages 487–503. Association for Computational Linguistics (ACL), 2021.

[123] Geoff Pleiss, Manish Raghavan, Felix Wu, Jon Kleinberg, and Kilian Q Weinberger. On fairness and calibration. *Advances in neural information processing systems*, 30, 2017.

[124] Jing Qian, Anna Bethke, Yinyin Liu, Elizabeth Belding, and William Yang Wang. A benchmark dataset for learning to intervene in online hate speech. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4755–4764, Hong Kong, China, November 2019. Association for Computational Linguistics.

[125] Michael Redmond. Communities and crime dataset. 2009. `http://archive.ics.uci.edu/ml//datasets/Communities+and+Crime).`

[126] Ronald L Rivest. Learning decision lists. *Machine learning*, 2(3):229–246, 1987.

[127] Joseph P Robinson, Gennady Livitz, Yann Henon, Can Qin, Yun Fu, and Samson Timoner. Face recognition: too bias, or not too bias? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–1, 2020.

[128] Barry Becker Ronny Kohavi. Data mining and visualization. 1996. `https://archive.ics.uci.edu/ml/datasets/adult`.

[129] Pradeep Kumar Roy, Sarabjeet Singh Chowdhary, and Rocky Bhatia. A machine learning approach for automation of resume recommendation system. *Procedia Computer Science*, 167:2318–2327, 2020.

[130] Anian Ruoss, Mislav Balunovic, Marc Fischer, and Martin Vechev. Learning certified individually fair representations. *Advances in neural information processing systems*, 33:7584–7596, 2020.

[131] Babak Salimi, Luke Rodriguez, Bill Howe, and Dan Suciu. Interventional fairness: Causal database repair for algorithmic fairness. In *Proceedings of the 2019 International Conference on Management of Data*, pages 793–810, 2019.

[132] Motoki Sato, Jun Suzuki, Hiroyuki Shindo, and Yuji Matsumoto. Interpretable adversarial perturbation in input embedding space for text. 2018.

[133] Holger Schielzeth. Simple means to improve the interpretability of regression coefficients. *Methods in Ecology and Evolution*, 1(2):103–113, 2010.

[134] Emily Sheng, Kai-Wei Chang, Prem Natarajan, and Nanyun Peng. The woman worked as a babysitter: On biases in language generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3407–3412, 2019.

[135] Benjamin Shickel, Patrick James Tighe, Azra Bihorac, and Parisa Rashidi. Deep ehr: a survey of recent advances in deep learning techniques for electronic health record (ehr) analysis. *IEEE journal of biomedical and health informatics*, 22(5):1589–1604, 2017.

[136] Haim Shore. Approximating an unknown distribution when distribution information is extremely limited. *Communications in Statistics-Simulation and Computation*, 27(2):501–523, 1998.

[137] K Simonyan, A Vedaldi, and A Zisserman. Deep inside convolutional networks: visualising image classification models and saliency maps. In *Proceedings of the International Conference on Learning Representations (ICLR)*. ICLR, 2014.

[138] Ioana Baldini Soares, Dennis Wei, Karthikeyan Natesan Ramamurthy, Moninder Singh, and Mikhail Yurochkin. Your fairness may vary: Pretrained language model fairness in toxic text classification. In *Annual Meeting of the Association for Computational Linguistics*, 2022.

[139] Nisha Srinivas, Karl Ricanek, Dana Michalski, David S Bolme, and Michael King. Face recognition algorithm bias: Performance differences on images of children and adults. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 0–0, 2019.

[140] Dan Steinberg and Phillip Colla. Cart: classification and regression trees. *The top ten algorithms in data mining*, 9:179, 2009.

[141] Vitomir Štruc and Nikola Pavešić. Gabor-based kernel partial-least-squares discrimination features for face recognition. *Informatica*, 20(1):115–138, 2009.

[142] Shivashankar Subramanian, Xudong Han, Timothy Baldwin, Trevor Cohn, and Lea Frermann. Evaluating debiasing techniques for intersectional biases. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2492–2498, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[143] Shivashankar Subramanian, Xudong Han, Timothy Baldwin, Trevor Cohn, and Lea Frermann. Evaluating debiasing techniques for intersectional biases. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2492–2498, 2021.

[144] Bing Sun, Jun Sun, Long H Pham, and Jie Shi. Causality-based neural network repair. In *Proceedings of the 44th International Conference on Software Engineering*, pages 338–349, 2022.

[145] Bing Sun, Jun Sun, Long H Pham, and Jie Shi. Causality-based neural network repair. In *Proceedings of the 44th International Conference on Software Engineering*, pages 338–349, 2022.

[146] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017.

[147] Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In *Proceedings of the 40th international conference on software engineering*, pages 303–314, 2018.

[148] Florian Tramer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, Jean-Pierre Hubaux, Mathias Humbert, Ari Juels, and Huang Lin. Fairtest: Discovering unwarranted associations in data-driven applications. In *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 401–416. IEEE, 2017.

[149] Luan Tran, Xi Yin, and Xiaoming Liu. Disentangled representation learning gan for pose-invariant face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1415–1424, 2017.

[150] Sakshi Udeshi, Pryanshu Arora, and Sudipta Chattopadhyay. Automated directed fairness testing. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, pages 98–108, 2018.

[151] Ameya Vaidya, Feng Mai, and Yue Ning. Empirical analysis of multi-task learning for reducing identity bias in toxic comment detection. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 14, pages 683–693, 2020.

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

[153] Sahil Verma and Julia Rubin. Fairness definitions explained. In *2018 ieee/acm international workshop on software fairness (fairware)*, pages 1–7. IEEE, 2018.

[154] Abraham Wald. Sequential tests of statistical hypotheses. *The annals of mathematical statistics*, 16(2):117–186, 1945.

[155] Abraham Wald and Jacob Wolfowitz. Optimum character of the sequential probability ratio test. *The Annals of Mathematical Statistics*, pages 326–339, 1948.

[156] Jiannan Wang, Thibaud Lutellier, Shangshu Qian, Hung Viet Pham, and Lin Tan. Eagle: creating equivalent graphs to test deep learning libraries. In *Proceedings of the 44th International Conference on Software Engineering*, pages 798–810, 2022.

[157] Zeerak Waseem and Dirk Hovy. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California, June 2016. Association for Computational Linguistics.

[158] Kellie Webster, Xuezhi Wang, Ian Tenney, Alex Beutel, Emily Pitler, Ellie Pavlick, Jilin Chen, and Slav Petrov. Measuring and reducing gendered correlations in pre-trained models. *CoRR*, abs/2010.06032, 2020.

[159] Ellery Wulczyn, Nithum Thain, and Lucas Dixon. Ex machina: Personal attacks seen at scale. In *Proceedings of the 26th international conference on world wide web*, pages 1391–1399, 2017.

[160] Wentao Xie and Peng Wu. Fairness testing of machine learning models using deep reinforcement learning. In *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 121–128. IEEE, 2020.

[161] Shen Yan, Hsien-Te Kao, and Emilio Ferrara. Fair class balancing: Enhancing model fairness without observing sensitive attributes. In Mathieu d'Aquin, Stefan Dietze, Claudia Hauff, Edward Curry, and Philippe

Cudré-Mauroux, editors, *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 1715–1724. ACM, 2020.

[162] Forest Yang, Mouhamadou Cisse, and Sanmi Koyejo. Fairness with overlapping groups; a probabilistic perspective. *Advances in neural information processing systems*, 33:4067–4078, 2020.

[163] Hongyu Yang, Cynthia Rudin, and Margo Seltzer. Scalable bayesian rule lists. In *International Conference on Machine Learning*, pages 3921–3930. PMLR, 2017.

[164] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. *ICML Deep Learning Workshop*, 2015.

[165] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rogriguez, and Krishna P Gummadi. Fairness constraints: Mechanisms for fair classification. In *Artificial Intelligence and Statistics*, pages 962–970. PMLR, 2017.

[166] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

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

[168] Brian Hu Zhang, Blake Lemoine, and Margaret Mitchell. Mitigating unwanted biases with adversarial learning. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pages 335–340, 2018.

[169] Guoqiang Peter Zhang. Neural networks for classification: a survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 30(4):451–462, 2000.

[170] Jie M Zhang and Mark Harman. "ignorance and prejudice" in software fairness. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, pages 1436–1447. IEEE, 2021.

[171] Mengdi Zhang. Github repository. 2021. `https://github.com/zhangmengling/NN_interpretability.git`.

[172] Mengdi Zhang and Jun Sun. Adaptive fairness improvement based on causality analysis. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 6–17, 2022.

[173] Mengdi Zhang and Jun Sun. Non-intrusive mitigation of biases in language models with statistical confidence. 2023.

[174] Mengdi Zhang, Jun Sun, and Jingyi Wang. Which neural network makes more explainable decisions? an approach towards measuring explainability. *Automated Software Engineering*, 29(2):39, 2022.

[175] Mengdi Zhang, Jun Sun, Jingyi Wang, and Bing Sun. Testsgd: Interpretable testing of neural networks against subtle group discrimination. *ACM Transactions on Software Engineering and Methodology*, 2022.

[176] Peixin Zhang, Jingyi Wang, Jun Sun, Guoliang Dong, Xinyu Wang, Xingen Wang, Jin Song Dong, and Ting Dai. White-box fairness testing through adversarial sampling. *Proceedings of the 42rd International Conference on Software Engineering (ICSE 2020), Seoul, South Korea*, 2020.

[177] Peixin Zhang, Jingyi Wang, Jun Sun, Xinyu Wang, Guoliang Dong, Xingen Wang, Ting Dai, and Jin Song Dong. Automatic fairness testing of neural classifiers through adversarial sampling. *IEEE Transactions on Software Engineering*, 2021.

[178] Peng Zhang, Yuexian Hou, and Dawei Song. Approximating true relevance distribution from a mixture model based on irrelevance data. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 107–114, 2009.

[179] Quanshi Zhang, Yu Yang, Haotian Ma, and Ying Nian Wu. Interpreting cnns via decision trees. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6261–6270, 2019.

[180] Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. Gender bias in coreference resolution: Evaluation and debiasing methods. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 15–20, 2018.

[181] Haoti Zhong, Cong Liao, Anna Cinzia Squicciarini, Sencun Zhu, and David Miller. Backdoor embedding in convolutional neural network models via invisible perturbation. In *Proceedings of the Tenth ACM Conference on Data and Application Security and Privacy*, pages 97–108, 2020.