FFB: A FAIR FAIRNESS BENCHMARK FOR IN-PROCESSING GROUP FAIRNESS METHODS

Xiaotian Han[1]* **Jianfeng Chi**[2] **Yu Chen**[3] **Qifan Wang**[2] **Han Zhao**[4] **Na Zou**[5] **Xia Hu**[6]
[1]Texas A&M University [2]Meta AI [3]Anytime AI [4]UIUC [5]University of Houston
[6]Rice University
han@tamu.edu {jianfengchi, wqfcr}@meta.com ychen@anytime-ai.com
hanzhao@illinois.edu nzou2@uh.edu xia.hu@rice.edu

ABSTRACT

This paper introduces the Fair Fairness Benchmark (FFB), a benchmarking framework for in-processing group fairness methods. Ensuring fairness in machine learning is important for ethical compliance. However, there exist challenges in comparing and developing fairness methods due to inconsistencies in experimental settings, lack of accessible algorithmic implementations, and limited extensibility of current fairness packages and tools. To address these issues, we introduce an open-source standardized benchmark for evaluating in-processing group fairness methods and provide a comprehensive analysis of state-of-the-art methods to ensure different notions of group fairness. This work offers the following key contributions: the provision of flexible, extensible, minimalistic, and research-oriented open-source code; the establishment of unified fairness method benchmarking pipelines; and extensive benchmarking, which yields key insights from $\mathbf{45,079}$ experiments, $\mathbf{14,428}$ GPU hours. We believe that our work will significantly facilitate the growth and development of the fairness research community. The benchmark is available at https://github.com/ahxt/fair_fairness_benchmark.

1 INTRODUCTION

Machine learning models trained on biased data have been found to perpetuate and even exacerbate the bias against historically underrepresented and disadvantaged demographic groups when deployed (Mehrabi et al., 2021; Pessach & Shmueli, 2022; Caton & Haas, 2020; Wan et al., 2023). As a result, concerns about fairness have gained significant attention, especially as applications of these models expand to high-stakes domains such as criminal justice, hiring process, and credit scoring. To mitigate such algorithmic bias, a variety of fairness criteria and algorithms have been proposed, which impose statistical constraints on the model to ensure equitable treatment under the respective fairness notions (Hsu et al., 2022; Chai et al., 2022; Reddy et al., 2021b). However, a fair and objective comparison between the proposed and existing algorithms to enforce fairness can be difficult due to the following reasons:

- *Hard to compare the performance of two objectives: utility*[1] and fairness. Often, there is a trade-off between these two objectives (McNamara et al., 2019; Zhao & Gordon, 2022; Xian et al., 2023). Besides, the instability of the fairness performance of those methods during the training process can complicate the pursuit of an optimal balance (Xian et al., 2023).
- *Inconsistent experimental settings (Madras et al., 2018; Louizos et al., 2016)*: Fairness methods can also be hindered by variations in dataset preprocessing and the use of different backbones. These discrepancies can lead to inconsistent (Table 1) and unfair comparison, further complicating the comparison of methods.
- *Stable and customizable implementation of commonly used fairness methods are not accessible.* The fairness methods are often implemented in different programming languages and frameworks, complicating the reproduction and the comparative analysis of various fairness approaches.

*This work was done while the first author was an intern at Meta.

[1] We use utility to represent the performance of the downstream tasks, which is commonly used in fairness community (Dwork et al., 2012; Li & Liu, 2022; Baumann et al., 2022).

- *Current fairness packages (Bellamy et al., 2018; Bird et al., 2020) and tools often suffer from a lack of extensibility.* This can make it difficult for researchers and practitioners to build upon existing methods and develop new ones.

This work aims to facilitate the growth and development of the fairness research community by addressing existing challenges, promoting more accessible and reproducible methods for fairness implementation, and establishing efficient benchmarking techniques. To achieve these goals, we develop a standardized benchmark for evaluating and comparing in-process group fairness methods, which we make open-source. We also conduct comprehensive experiments and analysis of fairness methods. The **major contributions** of our benchmark are summarized as follows:

- **Extensible, Minimalistic, and Research-oriented Open-Source Code**: We offer open-source implementation for all preprocessing, methods, metrics, and training results, thus facilitating other researchers to utilize and build upon this work. The Fair Fairness Benchmark is publicly available, making it easy for researchers and practitioners to use and contribute to.
- **Unified Fairness Method Benchmarking Pipelines**: Our benchmark includes a unified fairness method development and evaluation pipeline, with three key components: First, we provide a thorough statistical and experimental analysis of widely-used fairness datasets and identity some widely-used datasets unsuitable for studying fairness issues; second, we standardize preprocessing for these datasets, ensuring consistent, comparable evaluations; lastly, we present a range of bias mitigation algorithms and comprehensive evaluation metrics for group fairness.
- **Comprehensive Benchmarking and Detailed Obsevations**: We conduct comprehensive experiments on 14 datasets (each with two sensitive attributes), 6 utility metrics and 9 fairness metrics. We run a total of $45,079$ experiments. Our experiments yield the following key insights: ① Not all widely used fairness datasets stably exhibit fairness issues. *(Important for evaluation)*. ② The HSIC achieves the best utility-fairness trade-off among the tested methods. *(Not a popular baseline before)* ③ Utility-fairness trade-offs are generally controllable by the hyperparameter. *(Despite adversarial debiasing method)* ④ Stopping training while learning rate is lower enough is effective. *(Practical strategy)*.

Scope of this Work. We focus on the problem of in-processing group fairness, which is defined as discrimination against demographic groups. We consider the fairness methods in the context of binary classification and binary sensitive attributes.

Notation. The dataset is denoted as $\{(\mathbf{x}_i, s_i, y_i)_{i=1}^N\}$, where N is the number of samples. For the sample i in the dataset, $\mathbf{x}_i \in \mathbb{R}^d$ is non-sensitive attributes, $s_i \in \{0,1\}$ represents the binary sensitive attribute, and $y_i \in \{0,1\}$ is the label of the downstream task. We use $\hat{y} \in \{0,1\}$ to denote the predicted label of the downstream task, which is obtained by thresholding the output of a machine learning model $f(\mathbf{x}) : \mathbb{R}^d \rightarrow [0,1]$ with trainable parameter θ. Accordingly, we use X, Y, S and $\hat{Y}$ to denote the random variables.

2 Why is this Benchmark Needed?

Ensuring fairness in algorithmic predictions is crucial in the field of machine learning. However, fairly comparing the current fairness method is challenging due to inconsistencies in data pre-processing and a lack of flexibility in existing fairness packages. This section aims to analyze the critical issue of current fairness methods and discuss the urgent need for a benchmark to address these challenges.

Current fairness packages lack flexibility for researchers. AIF360 (Bellamy et al., 2018) and FairLearn (Bird et al., 2020) are two well-known fairness packages that have successfully mitigated bias in machine learning algorithms. As popular open-source Python packages, they provide practitioners with toolkits for detecting and mitigating bias in their models and evaluating model fairness. AIF360 is a comprehensive fairness toolkit offering a variety of algorithms and metrics for addressing bias in machine learning models. FairLearn also provides multiple algorithms and fairness metrics for assessing model performance. While both packages are highly recognized within the fair machine learning community, they may not give researchers the desired flexibility for research purposes. We provide a more in-depth comparison between AIF360, FairLearn, and FFB in Appendix A.

Inconsistent experimental setting leads to unfair comparison. Prior research has often experienced inconsistencies in data pre-processing and train test split, which has led to divergent performance

Table 2: Fairness definitions and metrics used in the experiments. Appendix B lists a more comprehensive list of fairness metrics and their details. Their simple code implementations are at this url.

Abbreviation	Name	Formal Definition
`dp` (Dwork et al., 2012)	Demographic/Statistical Parity	$P(\hat{Y} \mid S=0) = P(\hat{Y} \mid S=1)$
`prule` (Zafar et al., 2017)	p-Rule	$P(\hat{Y}=1 \mid S=1)/P(\hat{Y}=1 \mid S=0)\vert \leq p/100$
`ppv` (Chouldechova, 2017)	Predictive Parity Value Parity	$P(Y=1 \mid \hat{Y}, S=0) = P(Y=1 \mid \hat{Y}, S=1)$
`bnegc` (Kleinberg et al., 2016)	Balance for Negative Class	$\mathbb{E}[f(X) \mid Y=0, S=0] = \mathbb{E}[f(X) \mid Y=0, S=1]$
`bposc` (Kleinberg et al., 2016)	Balance for Positive Class	$\mathbb{E}[f(X) \mid Y=1, S=0] = \mathbb{E}[f(X) \mid Y=1, S=1]$
`eopp` (Hardt et al., 2016)	Equality of Opportunity	$P(\hat{Y} \mid S=0, Y=1) = P(\hat{Y} \mid S=1, Y=1)$
`eodd` (Hardt et al., 2016)	Equalized Odds	$P(\hat{Y} \mid S=0, Y=y) = P(\hat{Y} \mid S=1, Y=y), y \in \{0,1\}$
`abcc` (Han et al., 2023)	Area Between CDF Curves	$\int_0^1 \lvert F_0(x) - F_1(x)\rvert \, \mathrm{d}x$

results that hinder comparison and reproducibility. Minor variations in data preparation and dataset split can significantly impact the performance of machine learning algorithms. Due to these issues, the reported accuracy of tabular data (`Adult`,`German`) varies in different papers (Madras et al., 2018; Zemel et al., 2013; Edwards & Storkey, 2015; Feldman et al., 2015; Louizos et al., 2016) shown in Table 1. To tackle these issues, we propose a standardized and consistent data pre-processing and split approach, including data normalization, outlier removal, and the implementation of a uniform train-test split ratio.

Sufficient and in-depth analysis of fairness methods is urgently needed. The current fairness method lacks a comprehensive comparison, such as the training curves and stability, the influence of the utility-fairness trade-off parameters, and how to select the best utility-fairness trade-offs during the training process. Our benchmark addresses these shortcomings by offering more in-depth and multifaceted analysis. To present a more thorough understanding of fairness methods, we investigate the training stability, model performance under various fairness constraints, and the selection of best-performing models.

Table 1: The reported accuracy of tabular data varies in different papers.

Paper	`Adult`	`German`
Madras et al. (2018)	≈ 0.85	—
Zemel et al. (2013)	≈ 0.70	≈ 0.69
Edwards & Storkey (2015)	≈ 0.83	—
Feldman et al. (2015)	≈ 0.68	≈ 0.69
Louizos et al. (2016)	≈ 0.82	≈ 0.78
Δ_{max}	0.17	0.09
ΔPercentage	20%	13%

3 FFB: Fair Fairness Benchmark

To overcome the above limitations of the previous methods, we introduce the Fair Fairness Benchmark (FFB), a extensible, minimalistic, and research-oriented fairness benchmark package. Compare to other fairness packages, FFB codebase has the following **main characteristics**:

- **Extensible**: We provide the source code for fairness methods implementation, allowing researchers to modify, extend, and tailor these methods to suit their specific requirements and implement new ideas upon our code. This fosters a more customizable approach to developing fairness methods.
- **Minimalistic**: We focus on delivering core fairness methods and allowing users to understand the fundamental techniques comprehensively without being overwhelmed by unnecessary complexity. This approach ensures that users can easily implement and integrate our methods into their existing workflows while maintaining a solid grasp of the underlying concepts.
- **Research-oriented**: We include benchmark datasets and evaluation metrics that facilitate assessing fairness methods. This simplifies the research process, allowing researchers to quickly compare and analyze the effectiveness of different methods in various scenarios.

3.1 Group Fairness Metrics

To provide a comprehensive comparison for bias mitigating methods, we consider multiple fairness metrics, including demographic parity, p-rule, equality of opportunity, equalized odds, the area between CDF curves, *etc.* Table 2 presents the fairness metrics used in Section 4 and Section 5.

Table 3: The summary of the benchmarking datasets. The #nFeat/#cFeat is the number of the numerical/categorical features and the #allFeat is the total number of the features after our preprocessing. The $y_0 : y_1/s_0 : s_1$ is the ratio of two classes of the target label and the sensitive attributes. More details about datasets are presented in Appendix D. The dataset loading codes are at this url.

Dataset	Task	SensAttr	#Instances	#nFeat	#cFeat	#allFeat	$y_0 : y_1$	$s_0 : s_1$ (1st)	$s_0 : s_1$ (2nd)
Adult (Kohavi & Becker, 1996)	income	gender, race	$45,222$	7	5	101	$1:0.33$	$1:2.08$	$1:9.20$
German (Dua & Graff, 2017)	credit	gender, age	$1,000$	13	6	58	$1:2.33$	$1:2.23$	$1:4.26$
KDDCensus (Dua & Graff, 2017)	income	gender, race	$292,550$	32	8	509	$1:14.76$	$1:0.92$	$1:8.14$
COMPAS (Larson et al., 2016)	recidivism	age	$6,172$	400	5	405	$1:0.83$	$1:4.25$	—
Bank (Dua & Graff, 2017)	credit	gender, race	$41,188$	10	9	62	$1:0.13$	$1:37.58$	$1:37.58$
ACS-I (Ding et al., 2021)	income	gender, race	$195,665$	8	1	908	$1:0.70$	$1:0.89$	$1:1.62$
ACS-E (Ding et al., 2021)	employment	gender, race	$378,817$	15	0	187	$1:0.84$	$1:1.03$	$1:1.59$
ACS-P (Ding et al., 2021)	public	gender, race	$138,554$	18	0	1696	$1:0.58$	$1:1.27$	$1:1.31$
ACS-M (Ding et al., 2021)	mobility	gender, race	$80,329$	20	0	2678	$1:3.26$	$1:0.95$	$1:1.32$
ACS-T (Ding et al., 2021)	traveltime	gender, race	$172,508$	15	0	1567	$1:0.94$	$1:0.89$	$1:1.61$
CelebA-A (Liu et al., 2015)	attractive	gender, age	$202,599$	—	—	48×48	$1:0.95$	$1:1.40$	$1:0.29$
CelebA-W (Liu et al., 2015)	wavy hair	gender, age	$202,599$	—	—	48×48	$1:2.13$	$1:1.40$	$1:0.29$
CelebA-S (Liu et al., 2015)	smiling	gender, age	$202,599$	—	—	48×48	$1:1.07$	$1:1.40$	$1:0.29$
UTKFace (Zhang et al., 2017)	age	gender, race	$23,705$	—	—	48×48	$1:1.15$	$1:1.10$	$1:1.35$

3.2 Benchmarking Datasets

To provide a comprehensive comparison of fairness methods, we adopted multiple commonly-used fairness datasets (Le Quy et al., 2022) for our experiments, including tabular and image datasets. Table 3 summarizes the datasets used. they are publicly available and can be downloaded from the corresponding websites. We also present the number of features and the ratio of the target label and sensitive attributes. For example, the target label ratio of KDDCensus is $1 : 14.76$, which is an extremely unbalanced dataset.

3.3 Data Preprocessing

To ensure a fair comparison and maintain the reproducibility of the fairness approach, we adhere to a conventional data preprocessing strategy. We apply standard normalization for numerical features while employing one-hot encoding to process the categorical features. We also split the data into training and test sets with random seeds. We use the training set to train the model and the test set to evaluate the model's performance. We use the same data preprocessing strategy for all the datasets.

3.4 Benchmarking Fairness Methods

In this section, we introduce the benchmarking methodology employed in our experiments. The benchmarking methods can be classified into three categories: surrogate loss, independence constraints, and adversarial debiasing techniques. In this paper, we focus on in-processing methods for fairness primarily due to the following: 1) They intervene directly in the learning algorithm to ensure fairness, which provides a more nuanced and effective approach to mitigating bias; 2) The emergence of more in-processing techniques designed in deep neural networks calls for systematic comparison; 3) In-processing methods are susceptible to information leakage since they do not require sensitive attributes during inference. In particular, we consider the following three types of in-processing methods. **Gap Regularization** (Chuang & Mroueh, 2020) simplifies optimization by offering a smooth approximation to real loss functions, which are often non-convex or difficult to optimize directly. This approach includes DiffDP, DiffEodd, and DiffEopp. **Independence** introduces fairness constraint into the optimization process to minimize the impact of protected attributes on model predictions while maintaining performance. This approach includes PRemover (Kamishima et al., 2012) and HSIC (Li et al., 2019). **Adversarial Learning**[2] minimizes the utility loss while preventing the adversary from accurately predicting the protected attributes. This approach includes

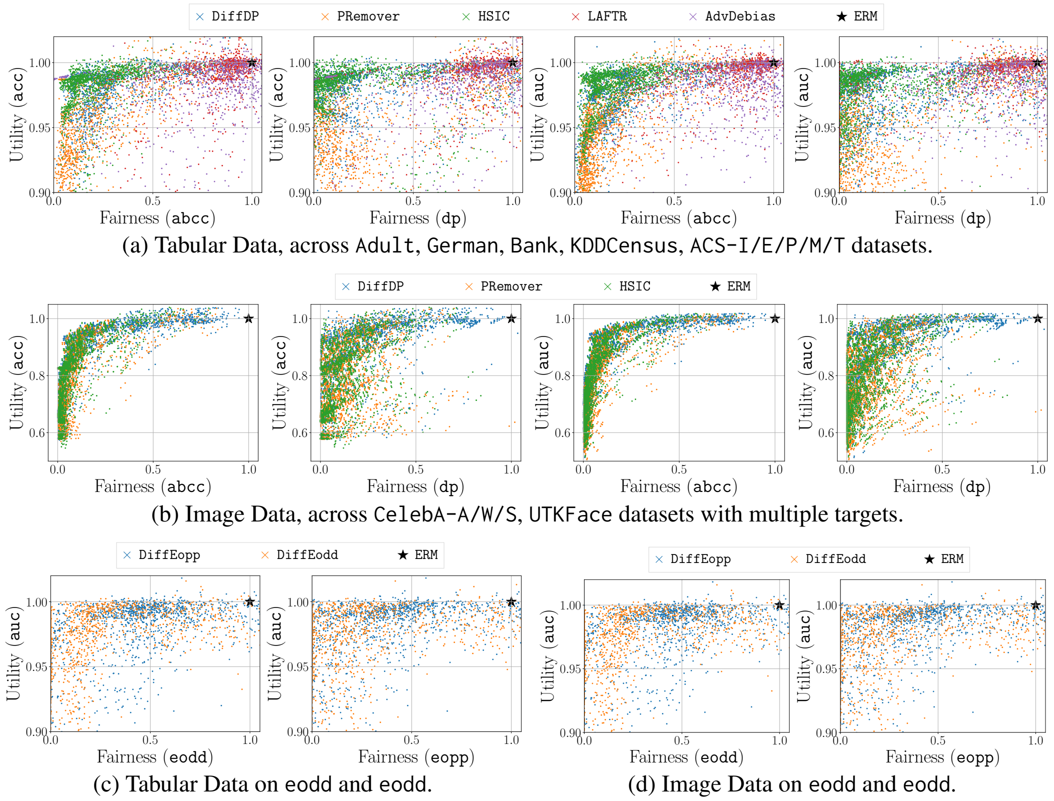
[2] For adversarial learning methods, we use gradient reversal (Ganin et al., 2016) for better training stability.

Table 4: Bias examination for all datasets. We identify the biased dataset specified with sensitive attributes with the reported utility and fairness performance of ERM. ~~Numbers~~ (e.g., ~~1.35±1.17~~) mean that the bias is too small, indicating is not suitable for fairness assessment. This only reflects our own recommendation and fairness metrics should be appropriately chosen for different applications. The biased datasets are marked with ✓, while the unbiased datasets are marked with ✗. The ~~✓~~ indicates that the bias exists but with a large standard deviation. The results are based on 10 trials with varying data splits and training seeds, to ensure reliable outcomes. The table is generated from **910** runs.

Dataset	SenAttr	Utility				Fairness					Recommend?
		acc	auc	ap	f1	dp	abcc	prule	eodd	eopp	
~~Bank~~	~~Age~~	91.17±0.57	94.05±0.19	62.41±1.83	54.31±11.14	10.88±4.27	10.64±1.63	44.48±5.95	10.71±5.68	6.16±4.90	~~✓~~
~~German~~	~~Gender~~	75.42±2.03	78.55±1.97	89.21±1.06	83.02±1.54	7.36±4.35	4.89±1.77	90.47±5.74	14.45±11.55	2.74±1.76	~~✓~~
	~~Age~~	75.19±2.16	77.21±2.60	88.28±1.74	82.90±1.62	12.20±6.02	10.01±1.63	84.44±7.17	17.97±10.71	8.26±6.12	~~✓~~
Adult	Gender	85.35±0.34	91.06±0.34	78.50±0.72	66.78±0.75	16.67±0.69	18.36±0.71	32.54±2.62	14.16±3.12	7.93±2.88	✓
	Race	85.21±0.27	91.10±0.16	78.65±0.35	66.85±0.46	12.23±0.72	12.59±0.60	41.54±2.68	13.12±2.65	8.81±2.93	✓
COMPAS	Gender	67.07±0.80	72.56±0.74	67.99±0.93	59.77±2.27	13.43±2.48	5.80±1.12	65.12±8.25	19.67±6.02	11.54±4.73	✓
	Race	67.13±1.06	72.98±0.59	68.24±0.72	60.58±3.06	16.83±3.48	8.15±1.12	61.83±4.56	29.03±6.66	20.05±3.95	✓
KDDCensus	Gender	94.88±0.48	94.03±0.04	99.55±0.00	97.32±0.24	3.61±1.60	5.20±0.35	96.35±1.62	14.97±7.11	~~0.77±0.37~~	✓
	~~Race~~	94.49±0.78	94.40±0.08	99.57±0.01	97.13±0.38	~~1.35±1.17~~	3.31±0.15	~~98.64±1.18~~	6.56±5.83	~~0.28±0.25~~	✗
ACS-I	Gender	82.30±0.12	90.28±0.09	86.02±0.15	77.91±0.19	9.10±0.31	8.27±0.24	79.01±0.65	3.38±0.61	~~1.75±0.41~~	✓
	Race	82.40±0.09	90.40±0.09	86.17±0.14	78.11±0.11	9.81±0.39	7.71±0.29	77.24±0.82	9.72±0.73	6.21±0.48	✓
~~ACS-E~~	~~Gender~~	81.63±0.12	88.95±0.08	83.12±0.12	81.31±0.16	~~0.60±0.20~~	~~0.56±0.12~~	~~98.87±0.37~~	10.77±0.20	~~0.90±0.18~~	✗
	~~Race~~	81.99±0.16	90.00±0.13	85.58±0.24	81.38±0.11	~~1.42±0.35~~	~~0.99±0.11~~	~~97.29±0.62~~	3.48±0.82	~~2.19±0.45~~	✗
~~ACS-P~~	~~Gender~~	71.92±0.18	75.25±0.16	67.23±0.22	52.93±0.71	~~2.09±0.64~~	~~2.35±0.16~~	91.26±2.52	~~2.30±1.32~~	~~1.52±1.00~~	✗
	~~Race~~	71.70±0.22	75.00±0.31	67.01±0.28	52.06±0.54	~~0.48±0.32~~	~~1.98±0.20~~	~~97.87±1.36~~	4.63±0.38	4.03±0.72	✗
~~ACS-M~~	~~Gender~~	76.81±0.32	72.85±0.36	88.40±0.22	86.54±0.31	~~0.18±0.17~~	~~0.84±0.12~~	~~99.80±0.19~~	~~0.45±0.51~~	~~0.08±0.09~~	✗
	~~Race~~	76.98±0.65	73.23±0.61	88.53±0.27	86.70±0.04	~~0.11±0.17~~	~~1.15±0.17~~	~~99.88±0.18~~	~~0.82±1.13~~	~~0.18±0.29~~	✗
ACS-T	Gender	66.36±0.22	73.54±0.18	71.59±0.14	66.51±0.31	8.60±0.45	5.02±0.28	84.65±0.74	12.90±0.82	5.72±0.44	✓
	Race	66.45±0.20	73.64±0.17	71.67±0.19	66.26±0.47	9.62±0.67	6.07±0.22	83.09±0.92	15.19±1.24	6.50±0.99	✓
CelebA-A	Gender	78.19±0.44	86.67±0.53	86.66±0.64	79.17±0.48	52.39±1.27	37.67±0.98	30.42±1.23	70.84±3.15	35.53±1.82	✓
	Age	78.19±0.44	86.67±0.53	86.66±0.64	79.17±0.47	41.90±1.03	31.15±1.17	33.43±1.71	37.42±2.26	18.83±1.94	✓
CelebA-W	Gender	82.50±0.76	88.38±0.86	80.38±1.57	70.14±1.64	33.92±1.35	29.52±1.12	16.89±2.01	52.71±4.28	39.62±3.49	✓
	Age	82.50±0.76	88.38±0.86	80.38±1.57	70.14±1.64	10.27±0.71	10.61±0.47	64.59±2.35	10.63±2.18	6.48±1.94	✓
CelebA-S	Gender	89.95±3.40	96.51±1.84	96.67±1.80	89.08±6.79	14.09±1.08	13.02±1.46	72.76±3.44	6.99±1.16	6.51±1.06	✓
	Age	89.95±3.40	96.51±1.84	96.67±1.80	89.08±6.79	5.91±0.75	5.59±0.70	88.21±2.50	6.46±1.06	~~0.82±0.80~~	✓
UTKFace	Gender	83.34±0.71	91.78±0.57	91.36±0.67	81.66±0.85	25.68±1.88	20.57±1.23	54.61±2.54	28.66±4.12	17.16±2.63	✓
	Race	83.34±0.71	91.78±0.57	91.36±0.67	81.66±0.85	23.25±1.71	18.99±1.22	59.67±2.63	23.07±3.64	16.68±2.70	✓
Jigsaw	Gender	68.10±0.69	74.81±0.82	74.02±0.92	65.82±2.27	8.17±3.35	4.41±1.34	83.11±5.39	13.35±6.49	4.90±3.94	✓
	Race	68.10±0.69	74.81±0.82	74.02±0.92	65.82±2.27	4.56±2.28	~~1.74±0.52~~	90.25±4.85	7.92±3.95	3.13±2.26	✓

AdvDebias (Zhang et al., 2018; Louppe et al., 2017; Beutel et al., 2017; Edwards & Storkey, 2015; Adel et al., 2019) and LAFTR (Madras et al., 2018). The fairness methods are present as follows:

- **ERM** is a standard machine learning method that minimizes the empirical risk of the training data. It is a common baseline for fairness methods.
- **DiffDP, DiffEopp, DiffEodd** are the gap regularization methods for demographic parity, equalized opportunity, and equalized odds (Chuang & Mroueh, 2020). As these fairness definitions cannot be optimized directly, gap regularization is alternative loss and differentiable and can be optimized using gradient descent.
- **PRemover** (Kamishima et al., 2012) (PrejudiceRemover) minimizes the mutual information between the prediction accuracy and the sensitive attributes.
- **HSIC** (Gretton et al., 2005; Baharlouei et al., 2020; Li et al., 2019) minimizes the Hilbert-Schmidt Independence Criterion between the prediction accuracy and the sensitive attributes.
- **AdvDebias** (Zhang et al., 2018; Louppe et al., 2017; Beutel et al., 2017; Edwards & Storkey, 2015; Adel et al., 2019) (adversarial debiasing) maximize a classifier for prediction ability and simultaneously minimizes an adversary to predict sensitive attributes from the predictions.
- **LAFTR** (Madras et al., 2018) is a fair representation learning method that aims to learn an intermediate representation that minimizes the classification loss, reconstruction error, and the adversary's ability to predict the sensitive attributes from the representation.



(a) Tabular Data, across Adult, German, Bank, KDDCensus, ACS-I/E/P/M/T datasets.

(b) Image Data, across CelebA-A/W/S, UTKFace datasets with multiple targets.

(c) Tabular Data on eodd and eodd.

(d) Image Data on eodd and eodd.

Figure 1: The utility-fairness trade-offs of current fairness methods – DiffDP, PRemover, HSIC, LAFTR, and AdvDebias. To plot the fairness and utility performance in one figure, for each dataset, we normalize the utility (acc,auc) and fairness (abcc, dp) based on the performance of ERM, which is denoted as the point $(1.0, 1.0)$. The figures clearly show that utility-fairness exhibits trader-offs. These figures are generated from a total of $\mathbf{27568}$ runs.

4 Bias Examination for Widely Used Fairness Benchmark Datasets

The presence of bias in the current widely used dataset is not well examined and investigated as it should be, even though such bias can significantly influence the assessment of fairness methods. As such, our work endeavors to delve deeper into this issue by exploring the inherent bias in the widely used dataset. We aim to assess the suitability of these datasets for fairness evaluation critically.

① **Not all widely used fairness datasets stably exhibit fairness issues.** We systematically identify datasets that not only have demonstrated bias but are also prevalently used in fairness research. We found that in some cases, the bias in these datasets is either not consistently present or its manifestation varies significantly. This finding indicates that relying on these datasets for fairness analysis might not always provide stable or reliable results, suggesting the need for more rigorous dataset selection or bias evaluation methodologies in future fairness studies. The biased datasets are marked with ✓ while unbiased ones are with ✗. The ✓✗ indicates that the bias exists but with a large standard deviation.

5 Benchmarking Current Fairness Methods

This section presents comprehensive experiments to benchmark the performance of existing in-processing group fairness methods.

5.1 How the Bias Mitigating Methods Perform on Utility-Fairness Trade-offs?

In this section, we present the results of experiments conducted to assess the performance of existing in-processing fairness methods in terms of the utility-fairness trade-offs. We analyze how well these methods balance optimizing utility and ensuring fairness in decision-making. To accurately reflect the performance of the different methods, we aggregate the performance across different datasets in

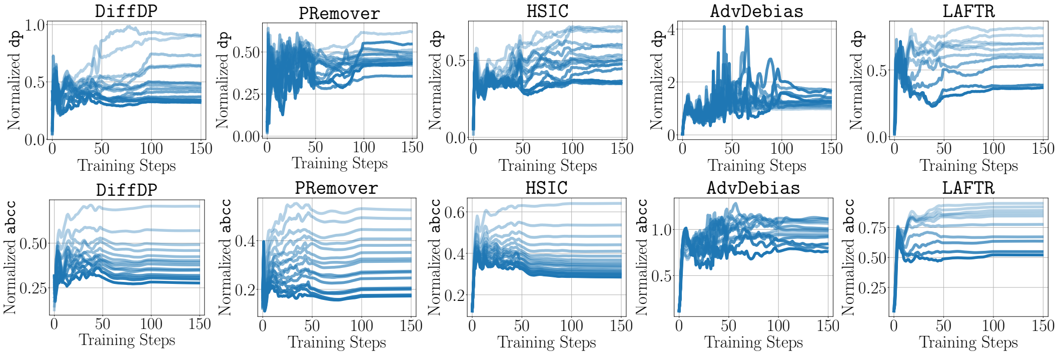


Figure 2: The fairness performance with varying fairness control hyperparameters. The intensity of the color represents the size of the control parameters. In most cases, the **larger** value of control parameters yields better fairness performance, while small ones have worse fairness performance. These figures are generated from **13110** runs of experiments.

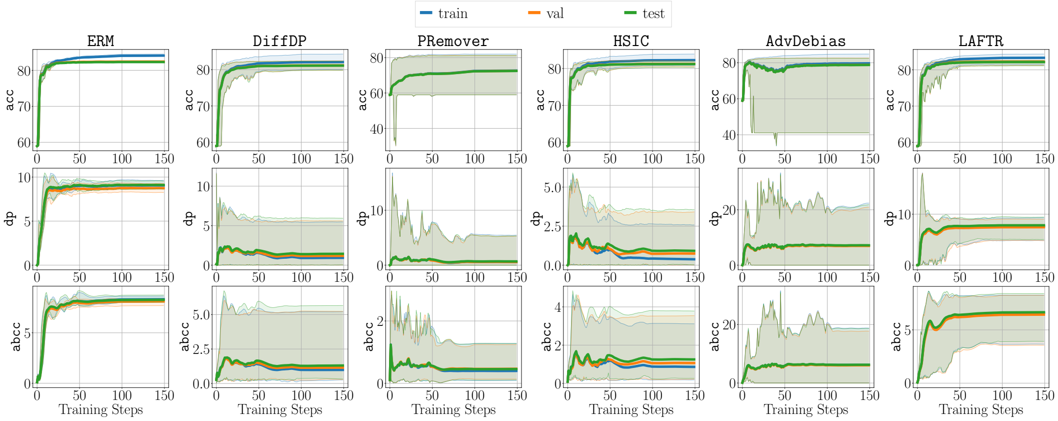


Figure 3: The training curves on tabular dataset. The training curves for fairness metrics typically have larger standard deviation than utility performance, showing the instability of fairness performance.

one figure. To do so, we normalize the utility (acc,auc) and fairness (abcc, dp) performance based on the performance of the ERM. From the results, we make the following major observations:

② **The utility-fairness performance of the current fairness method exhibits trade-offs.** We first present the utility-fairness trade-offs of the existing in-processing fairness methods in Figure 1. We conduct experiments using various in-processing fairness methods and analyze the ability to adjust the trade-offs to cater to specific needs while maintaining a balance between accuracy and fairness.

③ **The HSIC method achieves the best utility-fairness trade-off overall.** **The HSIC method consistently excels in balancing utility and fairness, outperforming other approaches across our tests.** This method, depicted in green in Figure 1, shows particular effectiveness when applied to tabular data. It exhibits a significant ability to improve fairness measures without compromising the precision of utility, maintaining high accuracy in predictions. This quality affirms the robustness of the HSIC method in preserving utility-fairness equilibrium under various conditions. However, when this method is applied to image data, it exhibits a relative performance decline, showing lower fairness and utility scores.

④ **Adversarial debiasing methods exhibit instability.** As illustrated in Figure 1, the utility-fairness points representing the AdvDebias method are scattered randomly across the figures, failing to depict a consistent trade-off pattern. This randomness suggests an inherent instability in adversarial debiasing methods. This inconsistency is further demonstrated in subsequent experiments, where the training curves reveal that these methods are difficult to control effectively.

5.2 Can the Utility-fairness Trade-offs be Controlled?

Hereby we investigate the extent to which the utility-fairness trade-offs can be controlled. We conduct experiments using various in-processing fairness methods and analyze the ability of trade-offs.

⑤ **The utility-fairness trade-offs are generally controllable.** The intensity of color in Figure 2 corresponds to the size of the control parameters. With the exception of adversarial debiasing, we find that the performance of most methods can be modulated effectively through the adjustment of hyperparameters. Specifically, larger control hyperparameters tend to yield lower `dp` and `abcc` values, indicating enhanced fairness. This implies that the trade-offs between utility and fairness can be actively managed in most cases, providing a crucial degree of flexibility in fairness-oriented data processing tasks. In comparison, the adversarial debiasing method (`AdvDebias`) is hard to control.

5.3 How do Utility and Fairness Performance Change During Training Process?

In this section, we thoroughly examine the training curves of existing in-processing fairness methods, which are not sufficiently explored in previous studies. We conduct a series of experiments to track the evolution of utility and fairness performance throughout the training process and evaluate the impact of these dynamics on the final results. We presented the training curves in Figures 3 and 4 for tabular data and image data, respectively.

⑥ **The training curves of utility are stable, while those of fairness are unstable.** Results on both tabular and image data show that the standard deviation for fairness metrics is significantly larger than that for utility metrics. Among the fairness methods, `LAFTR` shows the most stable fairness performance. Even though the value of fairness metrics is small, the large standard deviation still suggests unstable fairness performance. The results indicate a future direction focused on enhancing fairness training stability.

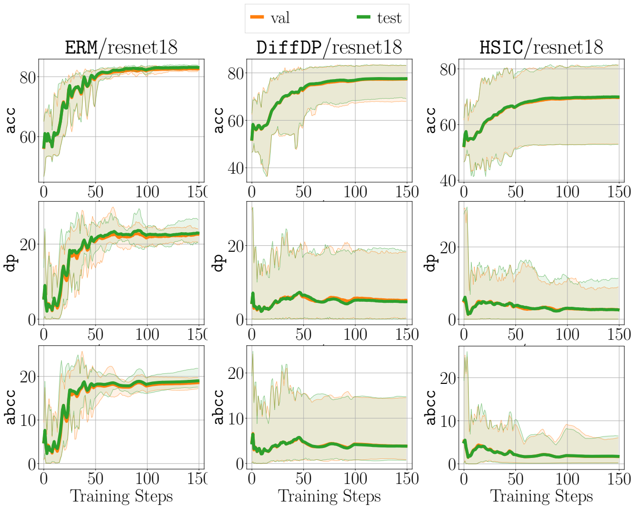


Figure 4: The training curves on image dataset. The results are similar to tabular dataset that training curves for fairness metrics typically have larger standard deviation than utility performance.

⑦ **Stopping training while the learning rate is lower enough proves effective.** In our work, we halt the model training when the learning rate diminishes to a value less than $1e^{-5}$, which is decayed by multiplying by 0.1 every 50 training steps. The utilization of learning rate decay to halt training results in stable fairness metrics, thereby affirming its efficacy and reasonableness.

5.4 Experiment on Text Data

In this section, we conducted experiments on text data, specifically on comment toxicity classification using the `Jigsaw` toxic comment dataset (Jigsaw, 2018). The task is to predict whether a comment is toxic or not. Part of this dataset is labeled with identity attributes, such as gender and race. In our study, we regard race and gender and race as sensitive attributes. We follow the setting in (Chuang & Mroueh, 2020) to transform each comment into a vector using BERT (Devlin et al., 2019). Subsequently, we employ an MLP to predict based on the encoded vector. We present the result on the utility-fairness trade-offs in Figure 5, training process in Figure 6, and controllability of trade-offs in Appendix H.

From these results, we observe a similar trend of utility-fairness trade-offs in text data as in tabular and image data in most cases, but there are some slight differences. We made the following observations: 1) Utility and fairness typically exhibit trade-offs. 2) The training stability has an opposite trend compared to tabular/image data. The training curves for utility are unstable, while those for fairness are relatively unstable for most bias mitigation methods.

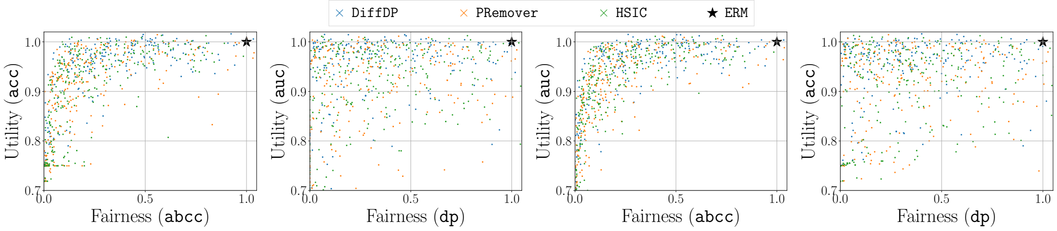


Figure 5: The utility-fairness trade-offs of current fairness methods – DiffDP, PRemover, and HSIC on text data. To plot the fairness and utility performance in one figure, for each dataset, we normalize the utility (acc,auc) and fairness (abcc, dp) based on the performance of ERM, which is denoted as the point $(1.0, 1.0)$. The figures show that utility-fairness exhibits trader-offs.

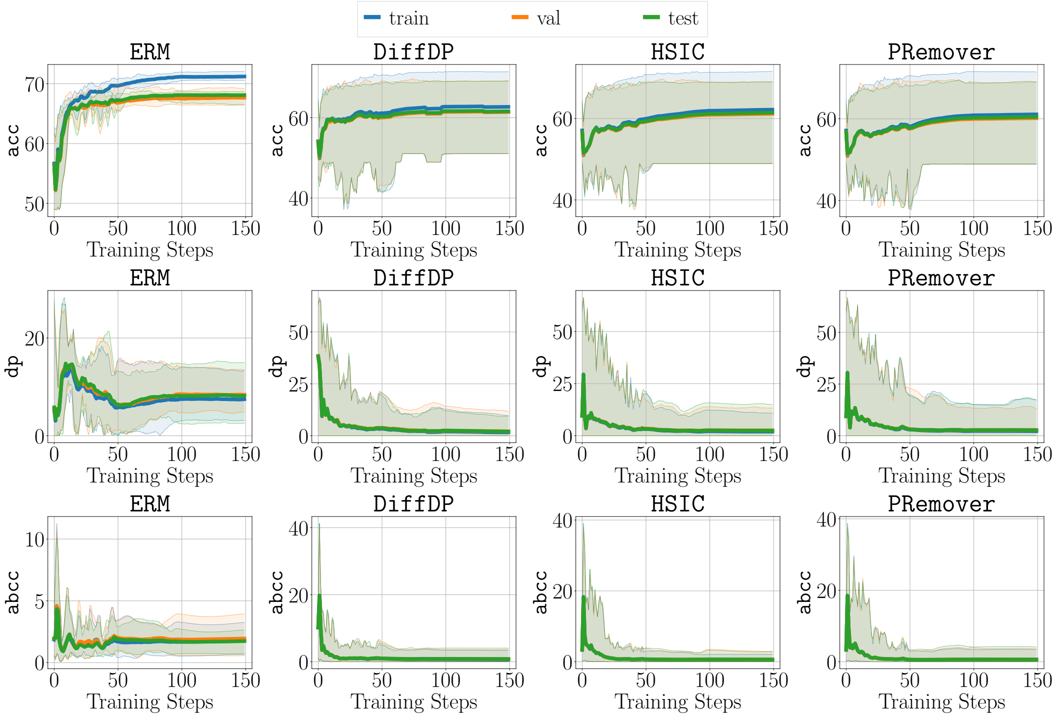


Figure 6: The training curves on text dataset. The training curves for fairness metrics typically have a larger standard deviation than utility performance, showing the instability of fairness performance.

6 Discussions

This paper introduces Fair Fairness Benchmark (FFB) to benchmark the in-processing group fairness models, offering extensible, minimalistic, and research-oriented open-source code, as well as comprehensive experiments.

Social Impact. Our benchmark, with its extensible, minimalistic, and research-oriented open-source code, is designed to facilitate researchers and practitioners to explore and implement fairness methods. Standardized dataset preprocessing and reference baseline implementation will help reduce inconsistencies and make fairness more accessible, especially for beginners in the field. Ultimately, our work aims to stimulate future research on fairness and foster the development of fairness models.

Limitations. When the ground truth is biased, relying solely on traditional accuracy can be problematic, as mentioned in (Wick et al., 2019; Zajko, 2022; Ruggieri et al., 2023). We employ multiple metrics (accuracy, average precision, AUC) for evaluating "utility" instead of just relying on accuracy, aiming to alleviate this problem to some extent.

Future work. Our plan for subsequent phases of this work involves extending the scope of the FFB to include a wider range of in-processing group fairness methods. Moreover, we intend to incorporate additional definitions of fairness into our evaluations. Exploring methods to measure unbiased accuracy is a promising direction for future research in fairness.

Acknowledgement

We thank the anonymous reviewers for their constructive suggestions and fruitful discussion. Han Zhao is supported in part by a research grant from the IBM-IL Discovery Accelerator Institute (IIDAI). This work is also supported in part by NSF IIS 1900990, 1939716, 2239257, and 1750074.

References

Tameem Adel, Isabel Valera, Zoubin Ghahramani, and Adrian Weller. One-network adversarial fairness. In Proceedings of the AAAI, pp. 2412–2420, 2019.

Wael Alghamdi, Hsiang Hsu, Haewon Jeong, Hao Wang, Peter Michalak, Shahab Asoodeh, and Flavio Calmon. Beyond adult and compas: Fair multi-class prediction via information projection. Advances in Neural Information Processing Systems, 35:38747–38760, 2022.

Sina Baharlouei, Maher Nouiehed, Ahmad Beirami, and Meisam Razaviyayn. Rényi fair inference. In International Conference on Learning Representations, 2020. URL https://openreview.net/forum?id=HkgsUJrtDB.

Joachim Baumann, Anikó Hannák, and Christoph Heitz. Enforcing group fairness in algorithmic decision making: Utility maximization under sufficiency. In Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency, pp. 2315–2326, 2022.

Rachel K. E. Bellamy, Kuntal Dey, Michael Hind, Samuel C. Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Lohia, Jacquelyn Martino, Sameep Mehta, Aleksandra Mojsilovic, Seema Nagar, Karthikeyan Natesan Ramamurthy, John Richards, Diptikalyan Saha, Prasanna Sattigeri, Moninder Singh, Kush R. Varshney, and Yunfeng Zhang. AI Fairness 360: An extensible toolkit for detecting, understanding, and mitigating unwanted algorithmic bias, October 2018. URL https://arxiv.org/abs/1810.01943.

Alex Beutel, Jilin Chen, Zhe Zhao, and Ed H Chi. Data decisions and theoretical implications when adversarially learning fair representations. arXiv preprint arXiv:1707.00075, 2017.

Sarah Bird, Miro Dudík, Richard Edgar, Brandon Horn, Roman Lutz, Vanessa Milan, Mehrnoosh Sameki, Hanna Wallach, and Kathleen Walker. Fairlearn: A toolkit for assessing and improving fairness in ai. Technical Report MSR-TR-2020-32, Microsoft, May 2020. URL https://www.microsoft.com/en-us/research/publication/fairlearn-a-toolkit-for-assessing-and-improving-fairness-in-ai/.

Maarten Buyl and Tijl De Bie. Optimal transport of classifiers to fairness. In Advances in Neural Information Processing Systems, 2022.

Flavio Calmon, Dennis Wei, Bhanukiran Vinzamuri, Karthikeyan Natesan Ramamurthy, and Kush R Varshney. Optimized pre-processing for discrimination prevention. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), Advances in Neural Information Processing Systems, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/9a49a25d845a483fae4be7e341368e36-Paper.pdf.

Simon Caton and Christian Haas. Fairness in machine learning: A survey. arXiv preprint arXiv:2010.04053, 2020.

Junyi Chai, Taeuk Jang, and Xiaoqian Wang. Fairness without demographics through knowledge distillation. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), Advances in Neural Information Processing Systems, 2022. URL https://openreview.net/forum?id=8gjwWnN5pfy.

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

Ching-Yao Chuang and Youssef Mroueh. Fair mixup: Fairness via interpolation. In ICLR, 2020.

Sam Corbett-Davies, Emma Pierson, Avi Feller, Sharad Goel, and Aziz Huq. Algorithmic decision making and the cost of fairness. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '17, pp. 797–806, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450348874. doi: 10.1145/3097983.3098095. URL https://doi.org/10.1145/3097983.3098095.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In NAACL-HLT (1), 2019.

Jimmy Ba Diederik P. Kingma. Adam: a method for stochastic optimization. CoRR, abs/1412.6980, 2014.

Frances Ding, Moritz Hardt, John Miller, and Ludwig Schmidt. Retiring adult: New datasets for fair machine learning. NeurIPS, 2021.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In Conference on Innovations in Theoretical Computer Science (ITCS), 2012.

Harrison Edwards and Amos Storkey. Censoring representations with an adversary. arXiv preprint arXiv:1511.05897, 2015.

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, pp. 259–268, 2015.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. Domain-adversarial training of neural networks. Journal of Machine Learning Research, 17(59):1–35, 2016. URL http://jmlr.org/papers/v17/15-239.html.

Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. In International conference on algorithmic learning theory, pp. 63–77. Springer, 2005.

Xiaotian Han, Zhimeng Jiang, Hongye Jin, Zirui Liu, Na Zou, Qifan Wang, and Xia Hu. Retiring δDP: New distribution-level metrics for demographic parity. Transactions on Machine Learning Research, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=LjDFIWWVVa.

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

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016.

Brian Hsu, Rahul Mazumder, Preetam Nandy, and Kinjal Basu. Pushing the limits of fairness impossibility: Who's the fairest of them all? In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), Advances in Neural Information Processing Systems, 2022. URL https://openreview.net/forum?id=bot35zOudq.

Ray Jiang, Aldo Pacchiano, Tom Stepleton, Heinrich Jiang, and Silvia Chiappa. Wasserstein fair classification. In Uncertainty in Artificial Intelligence, pp. 862–872. PMLR, 2020.

Jigsaw. Jigsaw unintended bias in toxicity classification, 2018. URL https://www.kaggle.com/c/jigsaw-unintended-bias-in-toxicity-classification.

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

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. Springer, 2012.

Jon Kleinberg, Sendhil Mullainathan, and Manish Raghavan. Inherent trade-offs in the fair determination of risk scores. arXiv preprint arXiv:1609.05807, 2016.

Ronny Kohavi and Barry Becker. Uci adult data set. UCI Meachine Learning Repository, 5, 1996.

Jeff Larson, Surya Mattu, Lauren Kirchner, and Julia Angwin. Propublica compas analysis—data and analysis for 'machine bias'. https://github.com/propublica/compas-analysis, 2016. Accessed: 2023-03-13.

Tai Le Quy, Arjun Roy, Vasileios Iosifidis, Wenbin Zhang, and Eirini Ntoutsi. A survey on datasets for fairness-aware machine learning. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 12(3):e1452, 2022.

Peizhao Li and Hongfu Liu. Achieving fairness at no utility cost via data reweighing with influence. In International Conference on Machine Learning, pp. 12917–12930. PMLR, 2022.

Zhu Li, Adrian Perez-Suay, Gustau Camps-Valls, and Dino Sejdinovic. Kernel dependence regularizers and gaussian processes with applications to algorithmic fairness. arXiv preprint arXiv:1911.04322, 2019.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In Proceedings of International Conference on Computer Vision (ICCV), December 2015.

Christos Louizos, Kevin Swersky, Yujia Li, Max Welling, and Richard S Zemel. The variational fair autoencoder. In International Conference on Learning Representations (ICLR), 2016.

Gilles Louppe, Michael Kagan, and Kyle Cranmer. Learning to pivot with adversarial networks. NeurIPS, 30, 2017.

David Madras, Elliot Creager, Toniann Pitassi, and Richard Zemel. Learning adversarially fair and transferable representations. International Conference on Machine Learning, 2018.

Daniel McNamara, Cheng Soon Ong, and Robert C Williamson. Costs and benefits of fair representation learning. In Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society, pp. 263–270, 2019.

Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. ACM Computing Surveys (CSUR), 54(6):1–35, 2021.

Anay Mehrotra and Nisheeth Vishnoi. Fair ranking with noisy protected attributes. Advances in Neural Information Processing Systems, 35:31711–31725, 2022.

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. In Advances in neural information processing systems, pp. 8026–8037, 2019.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12:2825–2830, 2011.

Dana Pessach and Erez Shmueli. A review on fairness in machine learning. ACM Computing Surveys (CSUR), 55(3):1–44, 2022.

Charan Reddy, Deepak Sharma, Soroush Mehri, Adriana Romero Soriano, Samira Shabanian, and Sina Honari. Benchmarking bias mitigation algorithms in representation learning through fairness metrics. In J. Vanschoren and S. Yeung (eds.), Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks, volume 1, 2021a. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/file/2723d092b63885e0d7c260cc007e8b9d-Paper-round1.pdf.

Charan Reddy, Deepak Sharma, Soroush Mehri, Adriana Romero-Soriano, Samira Shabanian, and Sina Honari. Benchmarking bias mitigation algorithms in representation learning through fairness metrics. In Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1), 2021b. URL https://openreview.net/forum?id=OTnqQUEwPKu.

Salvatore Ruggieri, Jose M Alvarez, Andrea Pugnana, Franco Turini, et al. Can we trust fair-ai? In Proceedings of the AAAI Conference on Artificial Intelligence, pp. 15421–15430, 2023.

Saeed Sharifi-Malvajerdi, Michael Kearns, and Aaron Roth. Average individual fairness: Algorithms, generalization and experiments. Advances in neural information processing systems, 32, 2019.

Changjian Shui, Gezheng Xu, Qi Chen, Jiaqi Li, Charles X Ling, Tal Arbel, Boyu Wang, and Christian Gagné. On learning fairness and accuracy on multiple subgroups. Advances in Neural Information Processing Systems, 35:34121–34135, 2022.

Mingyang Wan, Daochen Zha, Ninghao Liu, and Na Zou. In-processing modeling techniques for machine learning fairness: A survey. ACM Transactions on Knowledge Discovery from Data, 17 (3):1–27, 2023.

Michael Wick, Jean-Baptiste Tristan, et al. Unlocking fairness: a trade-off revisited. Advances in neural information processing systems, 32, 2019.

Ruicheng Xian, Lang Yin, and Han Zhao. Fair and optimal classification via post-processing. In International Conference on Machine Learning, 2023.

Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rogriguez, and Krishna P. Gummadi. Fairness constraints: Mechanisms for fair classification. In International Conference on Artificial Intelligence and Statistics (AISTATS), 2017.

Mike Zajko. Artificial intelligence, algorithms, and social inequality: Sociological contributions to contemporary debates. Sociology Compass, 16(3):e12962, 2022.

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

Brian Hu Zhang, Blake Lemoine, and Margaret Mitchell. Mitigating unwanted biases with adversarial learning. In AAAI/ACM Conference on AI, Ethics, and Society (AIES), 2018.

Guanhua Zhang, Yihua Zhang, Yang Zhang, Wenqi Fan, Qing Li, Sijia Liu, and Shiyu Chang. Fairness reprogramming. arXiv preprint arXiv:2209.10222, 2022.

Zhifei Zhang, Yang Song, and Hairong Qi. Age progression/regression by conditional adversarial autoencoder. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2017.

Han Zhao and Geoffrey J. Gordon. Inherent tradeoffs in learning fair representations. Journal of Machine Learning Research, 23(57):1–26, 2022. URL http://jmlr.org/papers/v23/21-1427.html.

Appendix

Table of Contents

A RELATED WORK

Algorithmic Fairness Fairness in machine learning has garnered considerable attention in recent years. The goal of fairness in machine learning is to ensure that the machine learning models are fair and unbiased towards an individual or group. Thus, fairness in machine learning can be divided into t categories: group fairness (Dwork et al., 2012; Hardt et al., 2016; Corbett-Davies et al., 2017; Zafar et al., 2017; Madras et al., 2018) and individual fairness (Dwork et al., 2012; Sharifi-Malvajerdi et al., 2019). Group fairness aims to ensure that the machine learning models are fair to different groups of people, while individual fairness aims to "similar individuals should be treated similarly." To mitigate fairness and bias problems in machine learning models, bias mitigation methods can be divided into three categories: pre-processing (Kamiran & Calders, 2012; Calmon et al., 2017), in-processing (Kamishima et al., 2012; Zhang et al., 2018; Madras et al., 2018; Zhang et al., 2022; Buyl & De Bie, 2022; Alghamdi et al., 2022; Shui et al., 2022; Mehrotra & Vishnoi, 2022), and post-processing (Hardt et al., 2016; Jiang et al., 2020). Given that the group fairness metrics are widely adopted in real-world applications and the emergence of more in-processing techniques designed in deep neural network models, we focus on benchmarking in-processing methods for group fairness for neural network models for tabular and image data.

Fairness Packages and Benchmarks There are many fairness packages in the literature. Among them, AIF360 (Bellamy et al., 2018) and FairLearn (Bird et al., 2020) are the two most widely used Python packages that provide a set of metrics and algorithms to measure and mitigate bias in machine learning models. They provide a set of metrics to measure the bias of machine learning models, including disparate impact, statistical parity difference, and equal opportunity difference, and a set of algorithms to mitigate the bias of machine learning models. However, both AIF360 and FairLearn implement the bias mitigation algorithms using Scikit-learn (Pedregosa et al., 2011) API design (e.g., the use of `fit()` function) with complicated class inheritance, making the understanding and direct modification of implementation difficult. In comparison, our benchmark decouples the implementation of different bias mitigation algorithms using Pytorch-style (Paszke et al., 2019) training scripts and provides a unified fairness evaluation interface for a comprehensive list of group fairness metrics. One recently proposed benchmark (Reddy et al., 2021a) also aims to benchmark bias mitigation algorithms. However, their benchmark only includes adversarial learning methods and datasets (e.g., a synthetic dataset and CI-MNIST) without fairness implications and uses dp and eodd as the fairness metrics. In contrast, our benchmark is more comprehensive in terms of algorithms, datasets, and fairness evaluation metrics.

B Details of the Group Fairness

In this section, we provide the details of the group fairness. We first introduce the definition of group fairness. Then, we introduce the existing group fairness metrics and algorithms.

- `dp` (Demographic Parity or Statistical Parity) (Zemel et al., 2013). A classifier satisfies demographic parity if the predicted outcome $\hat{Y}$ is independent of the sensitive attribute S, i.e., $P(\hat{Y} \mid S = 0) = P(\hat{Y} \mid S = 1)$.
- `prule` (Zafar et al., 2017). A classifier satisfies $p\%$-rule if the ratio between the probability of subjects having a certain sensitive attribute value assigned the positive decision outcome and the probability of subjects not having that value also assigned the positive outcome should be no less than p/100, i.e., $|P(\hat{Y} = 1 \mid S = 1)/P(\hat{Y} = 1 \mid S = 0)| \leq p/100$.
- `eopp` (Equality of Opportunity) (Hardt et al., 2016). A classifier satisfies equalized opportunity if the predicted outcome Y is independent of the sensitive attribute S when the label $Y = 1$, i.e., $P(\hat{Y} \mid S = 0, Y = 1) = P(\hat{Y} \mid S = 1, Y = 1)$.
- `eodd` (Equalized Odds) (Hardt et al., 2016). A classifier satisfies equalized odds if the predicted outcome Y is independent of the sensitive attribute S conditioned on the label Y, i.e., $P(\hat{Y} \mid S = 0, Y = y) = P(\hat{Y} \mid S = 1, Y = y), y \in \{0, 1\}$.
- `acc` (Accuracy Parity). A classifier satisfies accuracy parity if the error rates of different sensitive attribute values are the same, i.e., $P(\hat{Y} \neq Y \mid S = 0) = P(\hat{Y} \neq Y \mid S = 1), y \in \{0, 1\}$.
- `aucp` (ROC AUC Parity). A classifier satisfies ROC AUC parity if its area under the receiver operating characteristic curve of w.r.t. different sensitive attribute values are the same.
- `ppv` (Predictive Parity Value Parity) A classifier satisfies predictive parity value parity if the probability of a subject with a positive predictive value belonging to the positive class w.r.t. different sensitive attribute values are the same, i.e., $P(Y = 1 \mid \hat{Y}, S = 0) = P(Y = 1 \mid \hat{Y}, S = 1)$.
- `bnegc` (Balance for Negative Class). A classifier satisfies balance for the negative class if the average predicted probability of a subject belonging to the negative class is the same w.r.t. different sensitive attribute values, i.e., $\mathbb{E}[f(X) \mid Y = 0, S = 0] = \mathbb{E}[f(X) \mid Y = 0, S = 1]$.
- `bposc` (Balance for Positive Class). A classifier satisfies balance for the negative class if the average predicted probability of a subject belonging to the positive class is the same w.r.t. different sensitive attribute values, i.e., $\mathbb{E}[f(X) \mid Y = 1, S = 0] = \mathbb{E}[f(X) \mid Y = 1, S = 1]$.
- `abcc` (Area Between Cumulative density function Curves) (Han et al., 2023) is proposed to precisely measure the violation of demographic parity at the distribution level. The new fairness metrics directly measure the difference between the distributions of the prediction probability for different demographic groups

C Experimental Setting

For tabular datasets, we use a two-layer Multi-layer Perceptron with 256 neurons each for all datasets. We use different bath sizes for different datasets based on the total number of instances of each dataset. For image datasets, we use various neural networks (such as ResNet-18 (He et al., 2016) and ResNet-152) as the backbone. We don't use weight decay for all datasets and all fairness methods. We use linear learning rate decay for all datasets and all fairness methods. We use Adam (Diederik P. Kingma, 2014) as the optimizer with a learning rate of 0.001 for both tabular and image data. As these objectives, utility and fairness, often present trade-offs, it can be challenging to determine when to stop model training, and this issue is rarely discussed in the previous literature. In this work, we adopted a straightforward stopping strategy *based on our experience*. We employ a linear decay strategy for the learning rate, halving it every 50 training steps. The model training is stopped when the learning rate decreases to a value below $1e^{-5}$.

D Details of the Benchmarking Datasets

In this section, we provide the details of the benchmarking datasets. We first introduce the benchmarking datasets. Then, we introduce the data preprocessing and data splitting.

- **Tabular Datasets**
  - **`Adult`**[3] (Kohavi & Becker, 1996). The Adult dataset is widely used in machine learning and data mining research. It contains 1994 U.S. census data. The task of the dataset is to predict whether a person makes over $50K a year, given an individual's demographic and financial information. The dataset includes sensitive information such as age and gender. In the literature, gender is mostly used as the (binary) sensitive attribute to evaluate group fairness.
  - **`COMPAS`**[4] (Larson et al., 2016). The COMPAS dataset contains records of criminal defendants, and it is used to predict whether the defendant will recidivate within two years. The dataset includes attributes related to the defendant, such as their criminal history, and demographic information, such as gender and race.
  - **`German`**[5] (Dua & Graff, 2017). The German Credit dataset contains information on individuals who applied for credit at a German bank, including their financial status, credit history, and demographic information (e.g., gender and age). It is used to predict whether an individual should receive a positive or negative credit risk rating.
  - **`Bank`**[6] (Dua & Graff, 2017). The bank marketing dataset is used to analyze the effectiveness of marketing strategies of a Portuguese banking institution by predicting if the client will subscribe to a term deposit. The input variables of the dataset include the bank client's personal information and other bank marketing activities related to the client. Age was studied as the sensitive attribute in (Zafar et al., 2017).
  - **`KDDCensus`**[7] (Dua & Graff, 2017). Similar to the Adult dataset, the task of the KDD Census dataset is to predict whether the individual's income is above $50k with more instances. The sensitive attributes are gender and race.
  - **`ACS-I/E/P/M/T`**[8] (Ding et al., 2021). The ACS dataset provides several prediction tasks (e.g., predict whether an individual's income is above $50K or whether an individual is employed). It is constructed from the American Community Survey (ACS) Public Use Microdata Sample (PUMS). All tasks contain features for race, gender, and other task-related features.
- **Image Datasets**
  - **`CelebA-A/W/S`**[9] (Liu et al., 2015). The CelebFaces Attributes dataset comprises 20k face images from 10k celebrities. Each image is annotated with 40 binary labels indicating specific facial attributes such as gender, hair color, and age.
  - **`UTKFace`**[10] (Zhang et al., 2017). The UTKFace dataset is a large-scale face dataset that contains over 20k face images of people from different ethnicities and ages. The images are annotated with age, gender, and ethnicity information.
- **Text Datasets**
  - **`Jigsaw`**[11] (Jigsaw, 2018). The CelebFaces Attributes dataset comprises 20k face images from 10k celebrities. Each image is annotated with 40 binary labels indicating specific facial attributes such as gender, hair color, and age.

[3] https://archive.ics.uci.edu/ml/datasets/adult

[4] https://github.com/propublica/compas-analysis

[5] https://archive.ics.uci.edu/dataset/144/statlog+german+credit+data

[6] https://archive.ics.uci.edu/dataset/222/bank+marketing

[7] https://archive.ics.uci.edu/ml/datasets/Census-Income+(KDD)

[8] https://github.com/zykls/folktables

[9] https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html

[10] https://susanqq.github.io/UTKFace/

[11] https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge

E Detailed Experimental Settings

We present the details of the experimental setting in Tables 5 to 7. Table 5 presents the common hyperparameters used by both Tabular and Image datasets, including an initial learning rate of 0.01, Adam as the optimizer, zero weight decay, StepLR as the scheduler with a step of 50, a gamma value of 0.1, and 150 training steps in total. Table 6 presents the range of control hyperparameters used for various fairness methods. Each method has a unique range of these parameters. Table 7 indicates the batch sizes chosen for various datasets during training, ranging from 32 for the `German` and `COMPAS` datasets to a large 4096 for the `KDDCensus` and `ACS-I/E/P/M/T` datasets, with `CelebA-A/W/S` and `UTKFace` datasets using a batch size of 128, which are determined by the number of instances of the datasets.

Table 5: Common Hyper-parameters.

Dataset	Initial LR	Optimizer	Weight Decay	Scheduler	StepLR_step	StepLR_gamma	Training steps
Tabular	0.01	Adam	0.0	StepLR	50	0.1	150
Image	0.01	Adam	0.0	StepLR	50	0.1	150

Table 6: The fairness control hyperparameter selections.

Dataset	Control hyperparameter
`DiffDP`	$0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0, 2.5, 3.0, 3.5, 4$
`DiffEodd`	$0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0, 2.5, 3.0, 3.5, 4$
`DiffEopp`	$0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0, 2.5, 3.0, 3.5, 4$
`PRemover`	$0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.40, 0.45, 0.50, 0.6, 0.7, 0.8, 0.9, 1.0$
`HSIC`	$50, 100, 150, 200, 250, 300, 350, 400, 450, 500, 600, 700, 800, 900, 1000$
`AdvDebias`	$0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0, 2.5, 3.0, 3.5, 4$
`LAFTR`	$0.1, 0.2, 0.3, 0.4, 0.5, 1, 2, 3, 4, 5$

Table 7: The batch size for different datasets during the training.

Dataset	`Bank`	`German`	`Adult`	`COMPAS`	`KDDCensus`	`ACS-I/E/P/M/T`	`CelebA-A/W/S`	`UTKFace`
Batch Size	1024	32	1024	32	4096	4096	128	128

F More Experiment Results on Adult

In this appendix, we present the experimental results on Adult datasets. The same experiment results are presented in http://TODO.

F.1 Utility-Fairness Trade-offs

We plot the utility-fairness trade-offs for the Adult dataset with gender as the sensitive attribute and present the results in Figures 7 and 8.

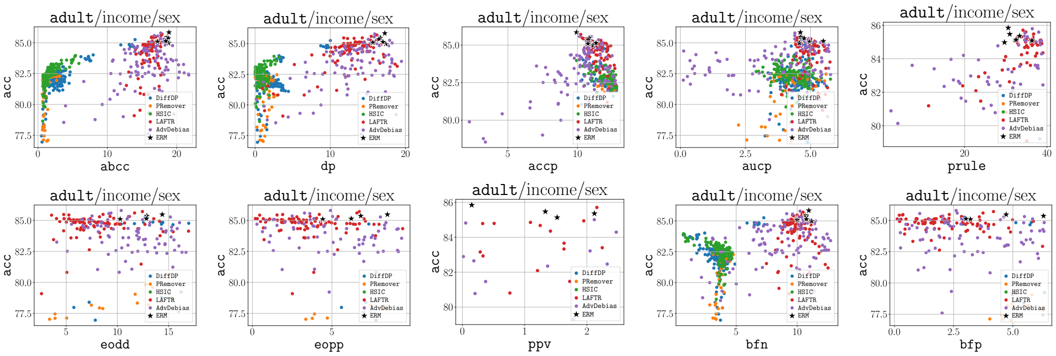


Figure 7: The Utility-Fairness Trade-offs with acc as utility metric.

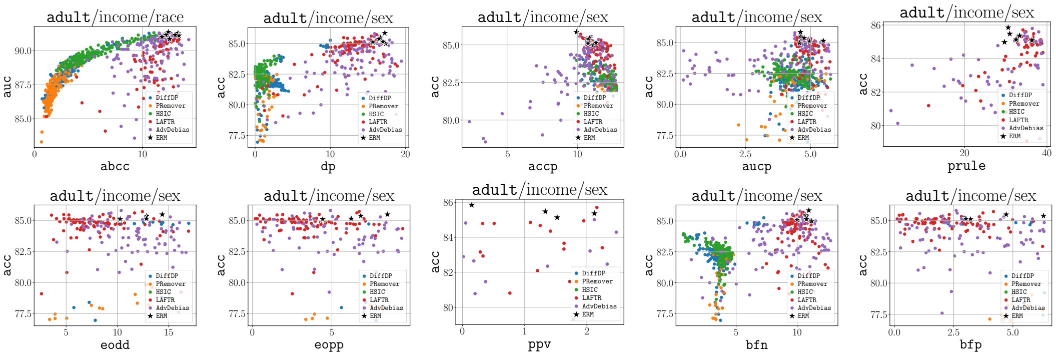


Figure 8: The Utility-Fairness Trade-offs with acc as utility metric.

F.2 Training Curves and Hyperparameters for Controlling Fairness

We plot the utility and fairness training curves for varying fairness control hyperparameters on the Adult dataset, and present the results in Figure 9. The intensity of the color represents the size of the control parameters. In most cases, the **larger** value of control parameters yields better fairness performance, while small ones have worse fairness performance.

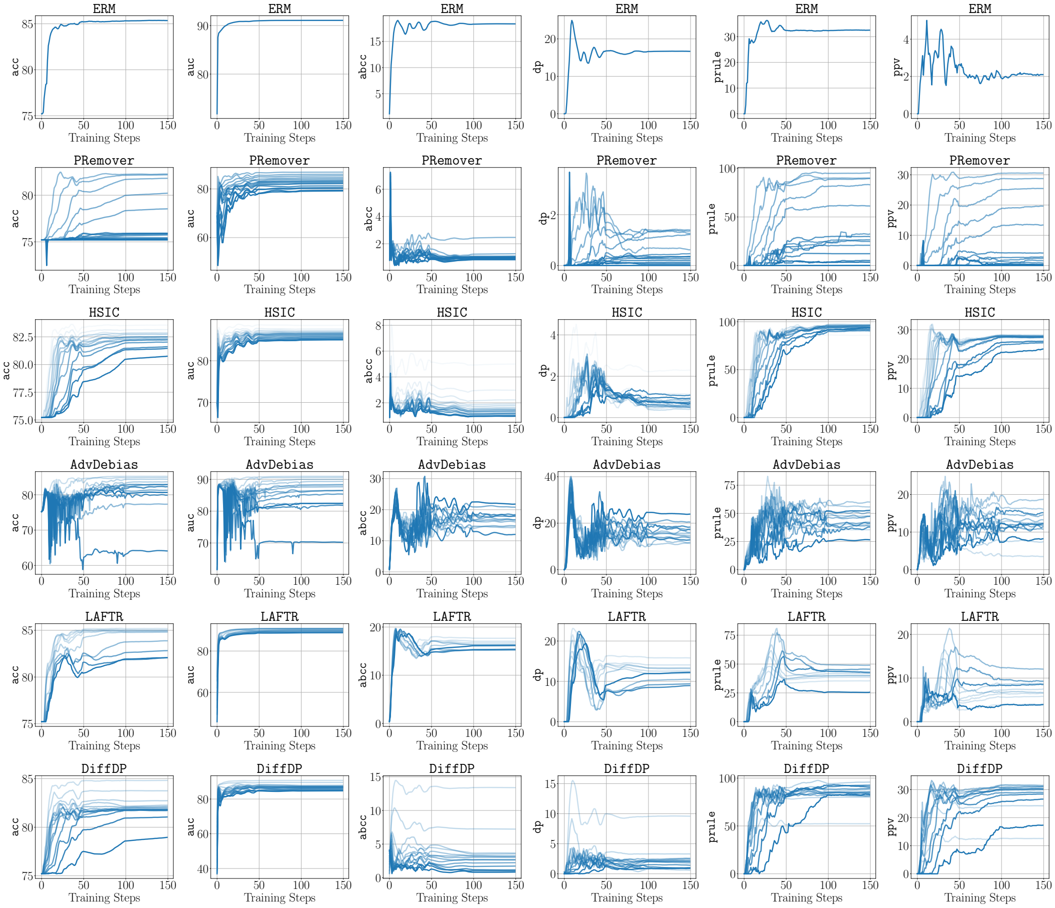


Figure 9: Hyperparameters for Controlling Fairness on `Adult` dataset.

G More Experiment Results on CelebA-A

In this appendix, we present the experimental results on the CelebA-A dataset.

G.1 Utility-Fairness Trade-offs

We plot the utility-fairness trade-offs for the CelebA-A dataset with gender as the sensitive attribute and present the results in Figure 10. The results show the utility-fairness trade-offs in CelebA-A dataset.

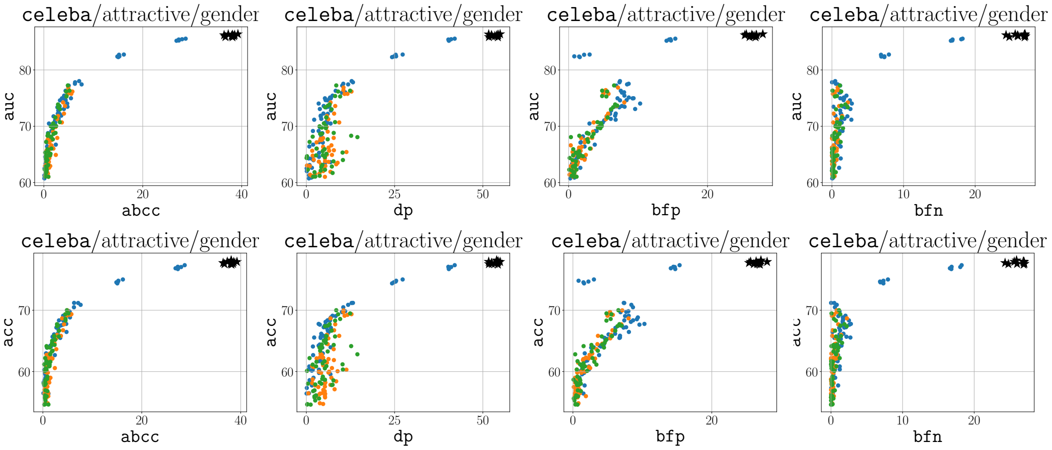


Figure 10: The Utility-Fairness Trade-offs

G.2 Training Curves and Hyperparameters for Controlling Fairness

We plot the utility and fairness training curves for varying fairness control hyperparameters on the CelebA-A dataset, and present the results in Figure 11. The intensity of the color represents the size of the control parameters. In most cases, the **larger** value of control parameters yields better fairness performance, while small ones have worse fairness performance.

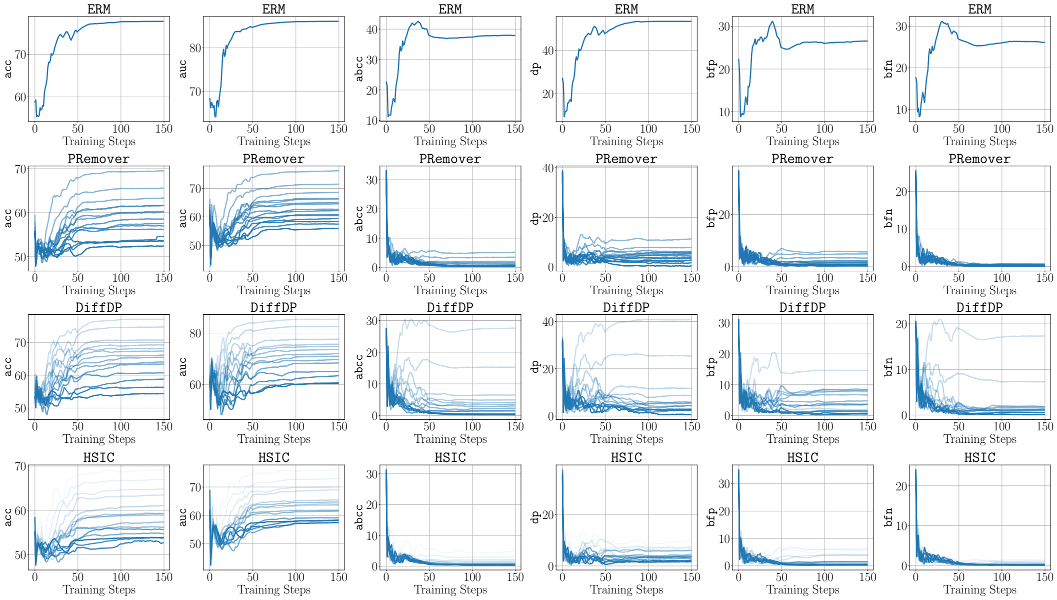


Figure 11: Training Curves and Hyperparameters for Controlling Fairness one CelebA-A dataset.

H More Experiment Results on Jigsaw

We present the results on fairness performance using various fairness control hyperparameters. We observed that the fairness metrics abcc and dp cannot be influenced by the fairness control hyperparameters, whereas prule can.

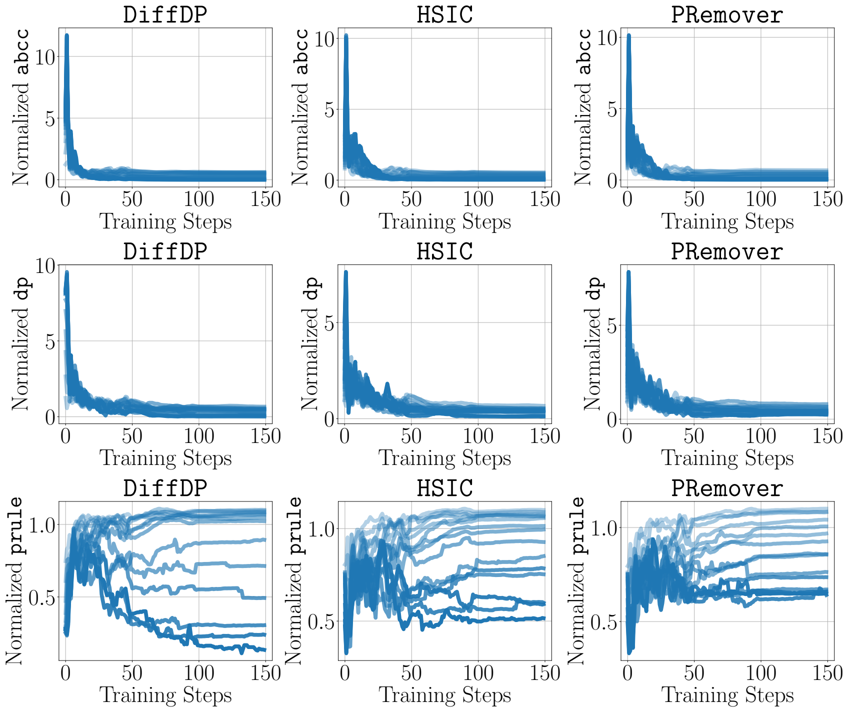


Figure 12: The fairness performance with varying fairness control hyperparameters on text data. The intensity of the color represents the size of the control parameters. In most cases, the **larger** value of control parameters yields better fairness performance, while small ones have worse fairness performance.

H.1 How does Model Size Influence Fairness Performance?

We conducted experiments to explore the influence of model size on fairness performance. We use various neural networks with the number of neural network trainable parameters spanning from 11.6M to 126.9M.[12] The results are presented in Figure 13.

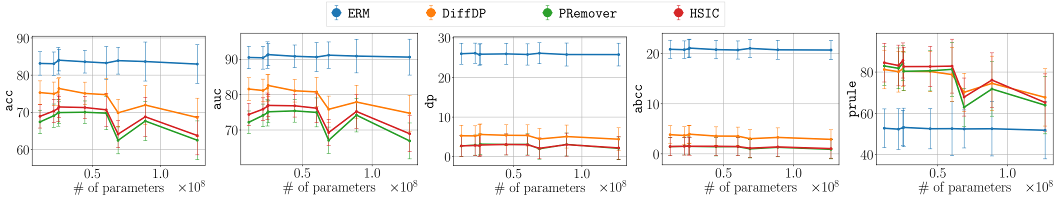


Figure 13: The performance with varying sizes of neural networks. The x-axis is the number of model parameters. There is no clear relationship between model size and performance.

⑧ **The architecture does not significantly influence fairness performance.** An increase in neural network parameters does not yield significant changes in utility or fairness performance. This suggests larger models do not naturally mitigate that dataset bias. The results observed from the DiffDP method indicate a potential correlation between utility performance and bias: fairness performance may deteriorate as utility increases, exhibiting trade-offs between them across different models.

[12] ResNet-18 (11.6M), ResNet-34 (21.8M), ResNet-50 (25.6M), ResNet-101 (44.5M), ResNet-152 (60.2M), ResNext-50 (25.0M), ResNext101 (88.8M), wide_ResNet-50 (68.9M), and wide_ResNet101 (126.9M).

I Implementation Comparison with AIF360 and FairLearn

In this section, we provide the implementations of adversarial debiasing in AIF360, FairLearn, and FFB to demonstrate FFB are extensible, minimalistic, and research-oriented compared to existing fairness packages. Algorithm 1, Algorithm 2, and Algorithm 3 show the implementation of adversarial debiasing in AIF360, FairLearn, and FFB, respectively.

We can see that both AIF360 and FairLearn use Scikit-learn API design (e.g., the use of `fit()` function), whereas FFB use Pytorch-style implementation, which provides a unified data preprocessing pipeline and fairness evaluation interface in a single script. Thus, researchers using FFB can use the bias mitigation method and reproduce the experimental results using one line of command. Additionally, AIF360 and FairLearn use complicated class inheritance (e.g., `AdversarialFairnessClassifier` in FairLearn inherents `AdversarialFairness` and `ClassifierMixin`), `AdversarialDebiasing` in AIF360 inherents `Transformer`), and other external dependencies (e.g., `AdversarialFairness` in FairLearn uses `backendEngine_` to implement the training step), making the implementation hard to read. This makes researchers hard to understand and re-implement the bias mitigation methods.

Algorithm 1 `AdvDebias` in AIF360

```
class AdversarialDebiasing(Transformer):

    def __init__(self):
        ...

    def _classifier_model(self, features, features_dim, keep_prob):
        ... # deine classifier

    def _adversary_model(self, pred_logits, true_labels):
        ... # deine adversary model

    def predict(self, dataset):
        ...

    def fit(self, dataset):

        with tf.variable_scope(self.scope_name):

            # tf graph construction
            ...

            self.sess.run(tf.global_variables_initializer())
            self.sess.run(tf.local_variables_initializer())


            for epoch in range(self.num_epochs):
                # training
                for i in range(num_train_samples//self.batch_size):
                    batch_feed_dict = {self.features_ph: batch_features,
                                     self.true_labels_ph: batch_labels,
                                     self.protected_attributes_ph: batch_protected_attributes,
                                     self.keep_prob: 0.8}
                    if self.debias:
                        _, _, pred_labels_loss_value, pred_protected_attributes_loss_vale = self.sess.run([
                                     classifier_minimizer,
                                     adversary_minimizer,
                                     pred_labels_loss,
                                     pred_protected_attributes_loss], feed_dict=batch_feed_dict)
                        if i % 200 == 0:
                            ... # logging
                    else:
                        _, pred_labels_loss_value = self.sess.run(
                            [classifier_minimizer,
                             pred_labels_loss], feed_dict=batch_feed_dict)
                        if i % 200 == 0:
                            ... # logging
```

Algorithm 2 AdvDebias in FairLearn

```
class _AdversarialFairness(BaseEstimator):

    def __init__(self):
        ...

    def __setup(self, X, Y, A):
        ...

    def fit(self, X, y, *, sensitive_features=None):
        X, Y, A = self._validate_input(X, y, sensitive_features, reinitialize=True)

        ...

        for epoch in range(epochs):
            batch_slice = slice(
                    batch * batch_size,
                    min((batch + 1) * batch_size, X.shape[0]),
                )
            (LP, LA) = self.backendEngine_.train_step(
                X[batch_slice], Y[batch_slice], A[batch_slice]
            )
            predictor_losses.append(LP)
            adversary_losses.append(LA)

            ...

    def partial_fit(self, X, y, *, sensitive_features=None):
        ...

    def decision_function(self, X):
        ...

    def predict(self, X):
        ...

    def _validate_input(self, X, Y, A, reinitialize=False):
        ...

    def _validate_backend(self):
        ...

    def _set_predictor_function(self):
        ...

class AdversarialFairnessClassifier(_AdversarialFairness, ClassifierMixin):
    def __init__(self):
        """Initialize model by setting the predictor loss and function."""
        self._estimator_type = "classifier"
        super(AdversarialFairnessClassifier, self).__init__()
```

Algorithm 3 AdvDebias in FFB

```
class Adversary(nn.Module):
    ...

class MLP(nn.Module):
    ...

def test(model, test_loader, criterion, device, args=None):
    ...

def train(clf, adv, data_loader, clf_criterion, adv_criterion, clf_optimizer, adv_optimizer):
    ...

if __name__ == ’__main__’:
    parser = argparse.ArgumentParser()
    parser.add_argument(’--dataset’, type=str, default="adult”)
    parser.add_argument(’--model’, type=str, default="MLP”)
    ...
    args = parser.parse_args()

    # Dataset selection
    if args.dataset == "adult":
        X, y, s = load_adult_data(sensitive_attribute=args.sensitive_attr)
    elif args.dataset == "compas":
        X, y, s = load_compas_data( sensitive_attribute=args.sensitive_attr)
    ...

    # Unified Dataset preprocessing (e.g., train/test split, )
    ...

    # define network architecture, etc. optimizer
    clf = MLP(n_features=n_features, num_classes=1, mlp_layers=[512, 256, 64]).to(device)
    clf_criterion = nn.BCELoss()
    clf_optimizer = optim.Adam( clf.parameters(), lr=args.lr)

    adv = Adversary( n_sensitive = 1 ).to(device)
    adv_criterion = nn.BCELoss(reduction="mean”)
    adv_optimizer = optim.Adam(adv.parameters(), lr=args.lr)


    for epoch in range(1, args.num_epochs+1):
        # begin training
        train(clf, adv, train_loader, clf_criterion, adv_criterion, clf_optimizer, adv_optimizer)

        if epoch % args.logging_steps == 0 or epoch == args.num_epochs:
            test_metrics = test(model=clf, test_loader=test_loader, criterion=clf_criterion, device=device)
            # logging metrics
            ...
```