Are My Deep Learning Systems Fair? An Empirical Study of Fixed-Seed Training

Shangshu Qian
Purdue University
West Lafayette, IN, USA
shangshu@purdue.edu

Hung Viet Pham
University of Waterloo
Vector Institute
hvpham@uwaterloo.ca

Thibaud Lutellier
University of Waterloo
Waterloo, ON, Canada
tlutelli@uwaterloo.ca

Zeou Hu
University of Waterloo
Vector Institute
z97hu@uwaterloo.ca

Jungwon Kim
Purdue University
West Lafayette, IN, USA
kim3241@purdue.edu

Lin Tan
Purdue University
West Lafayette, IN, USA
lintan@purdue.edu

Yaoliang Yu
University of Waterloo
Vector Institute
yaoliang.yu@uwaterloo.ca

Jiahao Chen*
J. P. Morgan AI Research
New York, NY, USA
jiahao@getparity.ai

Sameena Shah
J. P. Morgan AI Research
New York, NY, USA
sameena.shah@jpmorgan.com

Abstract

Deep learning (DL) systems have been gaining popularity in critical tasks such as credit evaluation and crime prediction. Such systems demand fairness. Recent work shows that DL software implementations introduce variance: identical DL training runs (i.e., identical network, data, configuration, software, and hardware) with a fixed seed produce different models. Such variance could make DL models and networks violate fairness compliance laws, resulting in negative social impact. In this paper, we conduct the first empirical study to quantify the impact of software implementation on the fairness and its variance of DL systems. Our study of 22 mitigation techniques and five baselines reveals up to 12.6% fairness variance across identical training runs with identical seeds. In addition, most debiasing algorithms have a negative impact on the model such as reducing model accuracy, increasing fairness variance, or increasing accuracy variance. Our literature survey shows that while fairness is gaining popularity in artificial intelligence (AI) related conferences, only 34.4% of the papers use multiple identical training runs to evaluate their approach, raising concerns about their results' validity. We call for better fairness evaluation and testing protocols to improve fairness and fairness variance of DL systems as well as DL research validity and reproducibility at large.

1 Introduction

DL systems, which consist of DL models and DL software (for example libraries such as TensorFlow) [71], are widely used; and enrusing their fairness is of paramount importance [47, 59, 97, 98]. Biases in DL systems have been identified as a major concern for civil rights by the U.S. government [27]. Yet, fairness testing for DL systems is still an open challenge.

Although the DL community is aware of the nondeterminism caused by using different random seeds [72, 90] , such algorithmic factors are not the sole source of DL nondeterminism. Even ***with the***

*Jiahao Chen has since moved to Parity Technologies.

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

same seed, e.g., the same initial weights, DL systems are nondeterministic, i.e., multiple training runs with the same network configuration, the same dataset, and the same seed, referred to as *fixed-seed identical training runs* (*FIT runs*), produce different models with different accuracies [71]. Such nondeterminism is caused by *software implementation*, e.g., software parallelism and floating-point computation of DL software (details in Appendix E.1).

Our previous work [71] shows that software implementation alone causes models' accuracies to vary by up to 2.9%, and most (83.8%) of the DL researchers and practitioners surveyed are unaware or unsure about the nondeterminism caused by software implementation. The small portion that is aware of such implemental variance underestimates it. The survey participants are researchers and practitioners with DL experience from universities and AI companies including Google, NVIDIA, Microsoft, and Facebook. These results suggest that even with fixed-seed identical training runs, one may still need to report error bars or use statistical tests.

Since the focus of the prior study [71] was on the variance of accuracy and training time only, it is unknown how DL software implementation affects the variance of *fairness*. Various debiasing algorithms [3, 47, 59, 98, 99] optimize DL models toward a specific fairness goal and it is unclear how software implementation affects these debiasing algorithms. In addition, unlike accuracy, which is a widely-accepted metric, dozens of fairness metrics exist [29, 35, 46, 60, 64, 99, 108]. It is unclear if software implementation causes different levels of variance for different fairness metrics.

In this paper, **we study the fairness variance caused by software implementation in training DL networks**. Overall, we found that fairness variance is large—up to 12.6% difference (27.3% versus 39.9%) across 16 fixed-seed identical training runs.

First, our results show that one training run may produce a fair model but another fixed-seed identical training run may generate an unfair one. For example, we reproduced previous work [47] that proposed a new network and a debiasing algorithm and we found that one FIT run generates a fair model (i.e., a model that does not discriminate based on gender) based on the threshold (20%) for the normalized disparate impact ($\overline{\text{DI}}$) metric [29, 64] ($\overline{\text{DI}}$ of 17.9%). A $\overline{\text{DI}}$ of 0% is the fairest, while a 100% $\overline{\text{DI}}$ is the least fair. However, a second FIT run produces an unfair model ($\overline{\text{DI}}$ of 20.4%, above the 20% threshold). This 20% threshold for $\overline{\text{DI}}$ has been used in ***a U.S. legal case*** to determine a system's fairness [9], where fairness variance could have changed the outcome.

Second, the variance of fairness imposes challenges for the valid evaluation of debiasing algorithms. For example, our experiments show that while a debiasing algorithm [107] reduces the bias amplification (BA) [99, 108] to 6.1% in one FIT run (compared with a BA score of 7.8% for its baseline), this technique *increases the bias amplification instead of reducing it* when we check the average of 16 FIT runs: an average BA score of 8.7% compared to an average BA score of 7.4% for the baseline. The Mann-Whitney U-test shows the results from the 16 runs are statistically significant. The observed variance across 16 FIT runs is solely caused by software implementation.

Finally, many mitigation techniques increase fairness variance or hurt accuracy. With Mann-Whitney U-test and Levene's test, our experiments show that a majority (68.0%) of the mitigation techniques reduce biases with a cost: they increase fairness variance, increase accuracy variance, or decrease model accuracy.

Since existing debiasing algorithms [47, 59] often do not measure the variance of biases, they could introduce risks. When such debiasing algorithms are applied to online learning, where DL models are trained continuously [53], the increased variance from debiasing algorithms could cause the resulting models' fairness to *miss* compliance requirements or service-level agreements. For example, Microsoft Tay [86] became a "racist chatbot" only after a few hours of deployment.

What's worse, even with fairness auditing before deployment, which is still *not* a standard practice in the industry [39], a model can be fair on pre-deployment data, but unfair on real-world data, due to the gap between training/test data and real-world data. The fairness variance puts the trained models at a greater risk of being unfair on real-world data. Theoretically, while debiasing techniques on average improve fairness on the training set, they increase the variance of fairness on the test set [2]. In practice, such gaps are the key reason that facial recognition systems [58, 76] have a racial bias.

In addition to quantifying fairness variance, **we conduct a survey of research papers** from top artificial intelligence (AI) conferences of recent years (2016–2020) to understand how researchers

handle the fairness variance caused by DL software implementations. Only 34.4% of the 32 papers evaluating or addressing DL fairness issues use multiple training runs in their evaluation.

To the best of our knowledge, we are the first to study software implementations' impact on DL fairness variance. The contributions are:

- A variance study of DL fairness of 27 techniques (22 mitigation techniques and five baselines). The experiments are performed on four popular datasets (CelebA, MS-COCO, imSitu, and CIFAR-10S) with three DL networks (ResNet-18, ResNet-50, and NIFR [47]), measured by seven popular bias metrics (Section 3).
  - **Finding 1**: Across 16 FIT runs, software implementation causes a maximum difference in bias metrics of up to 12.6% in the Demographic Parity (DP) metric and up to 11.8% in the $\overline{\text{DI}}$ metric. Yet, most (83.8%) researchers and practitioners are unaware or unsure about the DL nondeterminism caused by software implementation [71].
  - **Finding 2**: The effectiveness of the debiasing algorithms can be inaccurate with only one fixed-seed training run.
  - **Finding 3**: Most of (15 out of 22) the mitigation techniques tested increase at least one of the bias metrics.
  - **Finding 4**: In a majority (68.0%, or 70) of 103 experiments, debiasing algorithms have a negative impact on the model.
- A literature survey on the fairness research papers of top machine learning (ML), (NeurIPS/NIPS, ICML, AAAI, and FAccT), and computer vision (CVPR, ICCV, and ECCV) conferences (Section 4).
- Implications and suggestions for DL fairness researchers and practitioners (Section 5), including one should use FIT runs (setup details in Section 2.5) with statistical tests (Appendix B.3) to evaluate debiasing algorithms, when nondeterminism from DL software is unavoidable.

Data and code availability: Experiment data and artifact for reproducibility study are available in a public GitHub repository[1].

2 Experimental method

We investigate 27 techniques of DL fairness on four popular datasets with three networks by reproducing four previous works [47, 59, 98, 99] that fit our scope.

2.1 Scope

We focus on the impact of DL software implementation on the variance of group fairness [26] in classification tasks, where two orthogonal labels are assigned to each instance in the dataset, a *task label* for training a classifier, and a *protected label* that should not be discriminated against (e.g., gender). The set of instances with the same protected label forms a *protected group*.

Although the protected labels are not used in the classification task, the trained models could still be biased because the protected labels can be leaked into the prediction by proxy variables [3, 59, 98, 108].

2.2 Bias metrics

Various metrics exist to measure biases of ML algorithms. Yet, it is still an open problem to select these metrics as an optimization goal for real-world problems [49]. Therefore, we choose three fairness conditions and seven bias metrics derived from them. Fairness conditions include statistical parity [26] (SP), predictive equality [19] (PE), and equality of opportunity [35] (EOp). Seven bias metrics include demographic parity [15] (DP), normalized disparate impact [15] ($\overline{\text{DI}}$), statistical parity subgroup fairness [46] (SPSF), false positive subgroup fairness [46] (FPSF), equalized odds false positive [24] (EOFP), equalized odds true positive [35] (EOTP), and bias amplification [99, 108] (BA). For all bias metrics, 0 indicates the fairness condition is satisfied, i.e., a completely fair model, and 1 indicates a completely unfair model. Details of these metrics are in Appendix A.

[1] `https://github.com/lin-tan/fairness-variance/`

Table 1: Evaluated mitigation techniques and baselines

Task	Technique	Debiasing Algorithm	Network	Tgt
Multi-class classification in CIFAR-10S [99] (1)	S-Base	Baseline	ResNet-18	BA
	S-RS	Data Rebalancing		
	S-UC	Uniform confusion loss		
	S-GR	Gradient reversal		
	S-DD1	Sum of probabilities		
	S-DD2	Max of prob. w/ prior shift		
	S-DD3	Sum of prob. w/ prior shift		
	S-DD4	RBA debias		
	S-DI1	Domain independent, conditional		
	S-DI2	Domain independent, sum		
Multi-label classification in MS-COCO [98] (2)	C-Base	Baseline	ResNet-50	DP (BA*)
	C-R1	Data rebalancing, M/F = 1		
	C-R2	Data rebalancing, M/F < 2		
	C-R3	Data rebalancing, M/F < 3		
	C-A4	Adv training at 4th conv block		
	C-A5	Adv training at 5th conv block		
Multi-class classification in imSitu [98] (3)	I-Base	Baseline	ResNet-50	DP (BA*)
	I-R1	Data rebalancing, M/F = 1		
	I-R2	Data rebalancing, M/F < 2		
	I-R3	Data rebalancing, M/F < 3		
	I-A4	Adv training at 4th conv block		
	I-A5	Adv training at 5th conv block		
Binary classification in CelebA [59] (4)	A-Base	Baseline	ResNet-18	EO
	A-L2	L2 penalty		
	A-ALM	FairALM		
Binary classification in CelebA [47] (5)	N-Base	Baseline	NIFR	DP
	N-Flow	Conditional Flow		

Per-class and overall bias: Bias calculated on each class is called *per-class bias* and the bias for each model is the *overall bias*. For example, there are ten per-class biases in models trained on CIFAR-10 (one per class). The overall bias of the model is the average of all per-class biases.

Multi-(task)label classification: We treat each task label as an independent binary classification, and the overall bias value of the model is the average across all labels.

2.3 Reproduced studies

We conduct a literature survey of published papers from the recent five years (2016–2020) of the top ML (NeurIPS/NIPS, ICML, AAAI, and FAccT), and computer vision (CVPR, ICCV, and ECCV) conferences (Section 4). Out of the 32 papers relevant to the fairness of DL systems, only four [47, 59, 98, 99] have a working replication package so that we can reproduce their results. These papers study the fairness of DL systems for the five tasks described in Table 1 (expanded details in Appendix B.1). Overall, we replicate 27 different techniques from four state-of-the-art papers on five classification tasks. In the next section, we introduce each technique and detail our experiments.

2.4 Techniques

Table 1 lists the 27 techniques that we reproduce for our study. Column *Task* lists the tasks on which these techniques are applied. Column *Technique* lists the technique abbreviations we used. Column *Debiasing Algorithm* lists the debiasing algorithms used for each technique. Column *Network* lists the DL network used to train the models. Finally, column *Tgt* indicates which bias metric the corresponding debiasing algorithm is designed to reduce.

Five techniques (S-Base, C-Base, I-Base, A-Base, and N-Base) are baselines (i.e., do not use any debiasing algorithm), and the other 22 use debiasing algorithms. Techniques starting with "S" are from [99]; techniques starting with "C" and "I" are from [98] trained for tasks (2) and (3) respectively; techniques starting with "A" are from [59]; and techniques starting with "N" are from [47].

Apart from five baseline techniques, the remaining algorithms can be grouped into three categories including data rebalancing, fairness through blindness, and fairness through awareness. Details can be found in Appendix B.2.

Optimization target for tasks (2) and (3): The study [98] uses "bias amplification" as their optimization goal. However, current literature does not use this term consistently. The original study definition (noted as BA*) is different from the one proposed by Zhao et al. [108] (noted as BA) and BA* is more analogous to the DP metric. Therefore, we denote this optimization target as *DP (BA*)*.

2.5 Variance experiments setup

Following the prior work [71], we use *FIT runs* to measure the variance of trained model fairness. For each technique, all the training runs are executed with the same training data (also the original training/test split), hyper-parameters, and optimizers. With the fixed seed, all training runs also have the same order of data and the same initial weights. We perform 16 FIT runs with the same random seed for each technique, and then evaluate the fairness of the trained models using seven bias metrics.

Since there is no theoretical bound on the variance caused by DL software implementations, and each set of 16 FIT runs can only be viewed as a sample of the real distribution, we use statistical tests, including Mann-Whitney U-test, Cohen's d value, and Levene's test to ensure the statistical significance of our findings. Details of these statistical tests can be found in Appendix B.3.

3 Findings

We evaluate the fairness variance of 27 techniques (22 mitigation and five baseline) using seven bias metrics. These are 189 experiments (technique-metric pairs). To evaluate the impact of DL software implementation on different bias metrics, we perform 16 FIT runs for each of the 27 techniques, which produce 432 models. Each experiment generates one **bias value** per run (i.e., 16 bias values per experiment), which is a total of 3,024 bias values. The training time of the 189 experiments is 7,117 hours (**about 9.8 months**). Details of the hardware and software environment are in Appendix B.4.

To measure the difference across multiple FIT runs, we use two metrics: (1) the difference in percentage point as *MaxDiff* (maximum difference) and (2) the standard deviation in percentage point as *STDEV*. *MaxDiff* is the range of the bias values from a set of 16 FIT runs. For example, the *MaxDiff* of 16 runs, whose bias values are between 20.0%–30.0% inclusive, is 10.0%.

3.1 RQ1: How much variance of fairness does DL software implementation introduce?

Table 2 shows the fairness variance introduced by software implementation. Row *Technique* lists the techniques we focus on for this RQ. Row *Metric* shows the bias metrics used for evaluation. Row *Overall* shows the bias for the overall technique. Row *Per-class* shows the results for the class within each technique that has the most variance.

Table 2 shows that software implementation impacts all bias metrics. For example, we observed a maximum difference in DP with technique A-L2 of 12.6% (27.3%–39.9%), i.e., the most unfair model has 12.6% more gender bias measured by DP than the fairest model. Also, in technique S-GR, we observed a maximum difference in $\overline{\text{DI}}$ of 11.8% (21.3%–33.1%). Software implementation solely causes such variance, and thus cannot be eliminated by fixing the random seeds.

The per-class variance of fairness and single class fairness failure could be problematic if a class is privileged. For example, in a DL model for loan applications, the approval is the privileged class. A difference of 100% for $\overline{\text{DI}}$ means that in one run, the model is completely fair for all races (protected groups) to get approval. In another run, none of the applicants of a particular race gets approved, which is 100% bias against this minority group, even if the overall fairness variance is small.

Table 2: Maximum difference (MaxDiff) and standard deviation (STDEV) on the seven bias metrics for S-GR, A-L2, C-Base, and C-A5. All numbers are in percentage points.

Metric		DP	$\overline{\text{DI}}$	SPSF	FPSF	EOFP	EOTP	BA	DP	$\overline{\text{DI}}$	SPSF	FPSF	EOFP	EOTP	BA
Technique		**S-GR**							**A-L2**						
Overall (%)	**MaxDiff**	1.7	**11.8**	0.9	0.5	1.2	7.2	4.2	**12.6**	8.0	6.0	1.6	7.2	7.2	11.6
	STDEV	0.5	3.5	0.3	0.2	0.4	2.0	1.3	2.9	2.4	1.4	0.5	2.1	2.1	3.1
Per-class (%)	**MaxDiff**	7.3	50.1	3.6	2.5	5.7	34.6	19.3	12.6	31.1	6.0	3.5	14.3	14.3	15.9
	STDEV	1.9	12.7	1.0	0.7	1.4	8.1	5.0	2.9	8.8	1.4	1.2	4.8	4.8	4.2
Technique		**C-Base**							**C-A5**						
Overall (%)	**MaxDiff**	<0.1	2.6	<0.1	<0.1	<0.1	1.9	1.4	0.1	2.4	<0.1	<0.1	<0.1	2.6	3.1
	STDEV	<0.1	0.8	<0.1	<0.1	<0.1	0.6	0.4	<0.1	0.6	<0.1	<0.1	<0.1	0.6	0.8
Per-class (%)	**MaxDiff**	1.6	**100.0**	0.8	0.5	1.1	33.3	33.3	3.7	**100.0**	1.8	1.0	2.1	43.9	65.4
	STDEV	0.5	38.9	0.3	0.2	0.4	11.7	8.1	0.9	28.2	0.4	0.2	0.5	10.8	21.8

Finding 1: Across 16 FIT runs, software implementation causes a maximum difference of up to 12.6% in the DP bias metric, and up to 11.8% in the $\overline{\text{DI}}$ metric. A complete single-class fairness failure is possible, where one run produces a model with 0% bias while another with 100% bias.

Additional results of this RQ can be found in Appendix D.1. We discuss the implications of these results for researchers and practitioners in Section 5.

3.2 RQ2: How does software implementation affect debiasing algorithms?

To quantify the impact of *software implementation* on debiasing algorithms, we use the Mann-Whitney U-test and Levene's test to check whether the *average bias value* and the *variance of bias value* after debiasing are statistically different from those of its baseline respectively. We normalize each technique's bias values before Levene's test to eliminate the impact of the average value [5, 28, 77, 91].

For 22.1% (34 of 154) of the experiments (the comparison of 22 mitigation techniques with their baseline techniques evaluated using seven bias metrics), the mitigation techniques' bias values overlap with those of the baselines. For example, the range of EOTP bias values for the A-ALM debiasing is 19.6%–24.3%, while its baseline's (A-Base) range is 22.6%–24.7%.

For 9.1% (14 of 154) of the experiments, the mitigation technique's average bias metric is higher than the baseline, yet a single run's bias value is lower than a single run of the baseline. Thus, using a single run, one may incorrectly conclude that the mitigation technique is fairer, but on average, the mitigation technique is less fair. For example, *S-Base* (7.4% BA) is fairer than *S-GR* (8.7% BA) on average of 16 FIT runs ($p = 0.0011$ in U-test). However, one may obtain a fairer model with S-GR from one training run (e.g., BA of 6.1% for S-GR vs. 7.8% for S-Base). Therefore, DL fairness researchers should perform multiple runs and check statistical tests to ensure the effectiveness and validity of the debiasing algorithms. Our Levene's test results also confirm that the variance of bias values of *S-GR* is statistically higher than that of *S-Base* ($p = 0.0005$).

Finding 2: The effectiveness of the debiasing algorithms can be inaccurate with only one fixed-seed training run.

Since there is no consensus on which bias metric to use [19, 49], and some bias constraints cannot be satisfied simultaneously [50], we study if and how a debiasing algorithm reduces biases measured by one bias metric but increases biases measured by a different metric.

For the 154 experiments above, we conduct a series of Mann-Whitney U-tests on the bias values obtained. For each test that exhibits a statistically significant difference, we compute effect size using Cohen's d value between the two sets of bias values (baseline and mitigation) involved. The d values tell us whether each mitigation technique has a meaningful effect or not. Raw values before and after mitigation are available in the supplementary material.

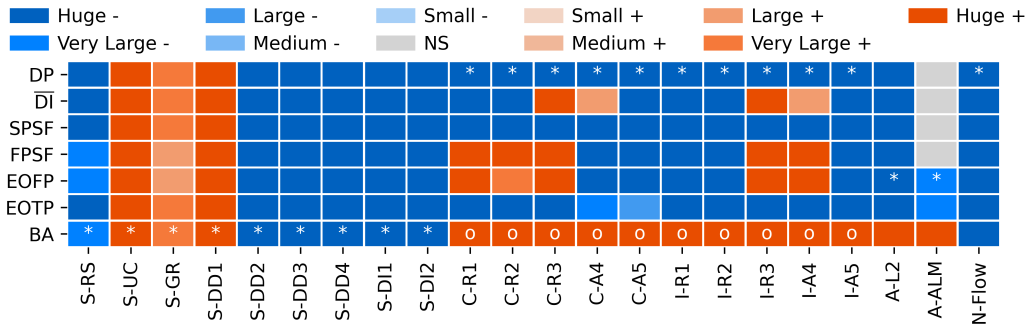

Figure 1: Effect size of *bias changes* between before and after mitigation application. Blue is a bias decrease (good), orange an increase (bad), and grey not statistically significant (NS).

Figure 1 shows the effect sizes of bias value between 22 mitigation techniques and their corresponding baselines. The x-axis and y-axis labels are the abbreviations for the techniques and bias metrics. If a mitigation technique statistically reduces the bias value on a bias metric (with respect to its baseline), the corresponding block is blue. Oppositely, the block is orange. The block is grey for mitigation techniques that do not induce a statistically significant difference from its baseline. The different shades of color show the standard grouping and interpretation of the effect size, such as "Huge" and "Very large". Figure 1's legend shows the color details. Blocks with a "*" are the bias metric that the mitigation technique is designed to reduce in the original research (e.g., BA for S-DI2 and DP for N-Flow). An "o" is a reminder that the optimization target of the mitigation techniques that begin with "C" and "I" is better captured by the DP metric (BA* in the original research, see Section 2.4).

Figure 1 shows that most (96 out of 103) mitigation techniques that induce a statistically significant improvement on the bias value have a huge effect (deep blue block). However, the vast majority (68.2%, 15 of 22) of mitigation techniques fail to reduce biases measured by all seven bias metrics. For example, A-L2 and A-ALM aim to improve bias metric EOFP. Although they succeeded, they also increase biases for BA.

This experiment shows that for most of the mitigation techniques, bias metrics may not be compatible with each other. Practitioners applying mitigation techniques must understand the techniques' optimization goal and use statistical tests.

Finding 3: Most of (15 out of 22) the mitigation techniques tested increase at least one of the bias metrics: all 15 of them fail to improve the newly introduced bias amplification metric.

3.3 RQ3: How does software implementation affect the cost of mitigation techniques?

The most known cost of mitigation techniques is the trade-off between fairness and model accuracy [25, 100]. However, with the software implementation taken into consideration, mitigation techniques could introduce new costs including more variance of model accuracy and fairness.

There are three dimensions to the cost of mitigation techniques: lower accuracy, higher accuracy variance, and higher fairness variance. We excluded the techniques that increase bias for all metrics compared to its baseline. For accuracy, we use U-test to check if the reduction is statistically significant. For the variance of both model accuracy and model bias, we use Levene's test.

Figure 2 shows the cost of mitigation techniques on fairness variance (Levene's test). Labels on x-axis and y-axis are the same as those in Figure 1. Blue and orange blocks respectively indicate lower and higher variance than the baseline. Grey blocks mean no statistically significant change in the variance of bias values, and "X" indicates mitigation technique fails to improve the bias metric tested. For example, the orange [N-Flow, DP] block shows that N-Flow generates models whose variance of DP bias values is higher than that of the baseline, and the comparison results are statistically significant.

To study the relationship among the three types of cost, we further examine the 154 experiments on mitigation techniques. Most of the experiments (103) reduce biases. The majority of them (70 out of 103) achieve bias reduction with a cost in one or more of the three dimensions. Eight experiments come with all three costs: lower accuracy, higher accuracy variance, and higher fairness variance, and are from A-L2 and A-ALM. However, there is no correlation between accuracy variance and bias

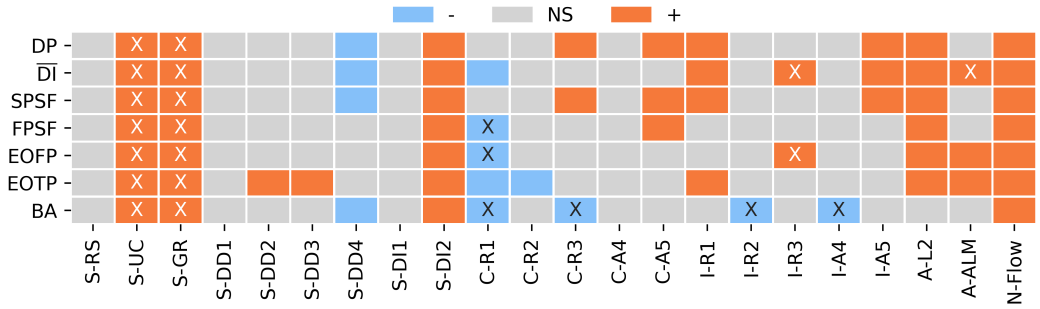

Figure 2: Cost of bias mitigation techniques on the *variance* of bias metrics after normalization (Levene's tests). Blue denotes a decrease in variance (good), orange an increase in variance (bad), and grey not statistically significant (NS).

variance. Interestingly, mitigation technique S-DD4 (RBA [108] debiasing) successfully reduces all of the seven bias metrics without sacrificing model accuracy or introducing more variance.

DL researchers and practitioners might overlook the cost of fairness variance when optimizing a model toward a particular fairness goal if they are unaware of the variance caused by DL software [71].

Finding 4: Out of 154 experiments with debiasing algorithms, 103 experiments show the applied algorithms reduce the bias values. In a majority (68.0%, or 70) of these 103 experiments, the debiasing algorithms have a negative impact on the model (36 increase the variance of bias value, 54 decrease the accuracy, and 15 increase the variance of accuracy, not exclusively).

4 Literature survey

We conduct a literature survey from the top ML conferences to evaluate the awareness of fairness variance of DL systems.

Paper selection criteria: We extract papers from the last five years (2016-2020) of the top ML (NeurIPS/NIPS, ICML, and AAAI), and computer vision (CVPR, ICCV, and ECCV) conferences. We also include FAccT and its predecessor FAT/ML workshop, an interdisciplinary venue of ML and fairness.

We only consider papers that contain fairness-related keywords (fair, bias, discriminate, aware, balance, disparity, opportunity, audit, gender, stereotype, race, amplification, ethic) in their titles using the Snowball Stemmer [73]. Then, we read the abstract and the paper and keep relevant papers involving the fairness of a classification task. Out of 24,186 papers, we extracted 274 papers studying fairness, including 32 dedicated to DL systems' fairness.

Paper survey results: While still taking only a small portion of accepted papers, the number of papers related to fairness has been increasing over the last five years. Researchers need to be aware of DL models' fairness variance caused by DL software implementations.

Finding 5: DL fairness is gaining more attention in the research community since 2017. The number of papers related to DL fairness in top ML and computer vision conferences has increased 750% since 2017 (two papers in 2017, 17 papers in 2020).

To understand whether researchers are aware of the fairness variance of DL training, we manually check whether the papers on the fairness of DL systems use multiple identical runs in their evaluation.

Finding 6: Of 32 papers relevant to the fairness of DL systems, only 34.4% (11 out of 32) of them use multiple identical runs to evaluate their proposed approaches.

A complete list of 32 papers surveyed in this section can be found in Appendix D.3.

5 Implications and suggestions

Research validity: When comparing the results of a proposed method (e.g., a new debiasing algorithm or a new DL network), it is necessary to run statistical tests to confirm any improvements.

Testing for fairness in online learning: When testing an online-learning software system for fairness (where models are continuously updated), such as identity-inference-based advertisement targeting systems [7, 70, 101], multiple identical runs are necessary, and statistical test results should be reported to measure the variance of bias values caused by the DL software implementation.

Research reproducibility: Given the increasing number of papers on fairness in AI [6, 32, 48, 49, 63, 65, 79, 105], software engineering [10, 14, 16, 94], and other conferences [2, 18, 52], reproducibility becomes increasingly important [4, 78]. Following prior efforts [43, 56, 72] that improve the reproducibility of DL methods, researchers may want to develop such guidance for fairness, e.g., considering multiple bias metrics, as variance levels are different for different bias metrics. Proper documentation of the software and hardware development environment and configuration parameters would also facilitate the reproduction of research results. As shown in Section 3.1, software implementation alone causes software fairness to vary by up to 12.6%, which is much higher than the 2.9% accuracy variance discovered by previous work [71].

Balancing hidden costs of debiasing: Since variance is a hidden cost for debiasing algorithms and could harm the fairness performance of online learning algorithms (Section 3.3), there is usually a trade-off between model accuracy, fairness, and variance. It is important that DL researchers and practitioners understand debiasing algorithms' costs, especially variance, and strike a balance among them.

Deterministic DL software for model tuning and debugging: Building deterministic DL libraries could be a research topic to improve model tuning and the debugging of DL systems. For a partial solution, one could use the deterministic mode (Appendix E.1) when it is provided by DL libraries.

Maximizing fairness through multiple runs: To take advantage of software variance, practitioners could train multiple times to select the fairest model. In this case, proper testing of the DL model is recommended to ensure the satisfaction of regulatory requirements, considering the pre-deployment and real-world data gap.

Selection of bias metrics: In most cases, a debiasing algorithm does not reduce bias on all targets (Section 3.2), and multiple fairness constraints could contradict each other [49, 50]. Therefore, developers and testers for DL systems should carefully select fairness requirements. Furthermore, optimizing on one bias metric may not be enough as it might make the model less fair on other metrics. DL researchers should consider the task and use the appropriate bias metric. In the strictest sense, one may need to use multiple bias metrics to ensure fairness across multiple dimensions.

6 Limitations and discussions

Limitation: It is known that gender bias exists in language embeddings [108]. We only study image classification tasks because they are a better fit for our scope of group fairness and bias metrics on image classification tasks are better defined. Also, we only empirically obtain the upper bound of fairness variance and did not include any theoretical proof. We leave both limitations as future work.

All 27 techniques reproduced are implemented in PyTorch instead of other popular DL frameworks such as TensorFlow and MXNet. We decide to reuse the implementation from the original authors as it is error-prone to reimplement all the techniques in another framework. Since modern DL frameworks share many implementation concepts (e.g., parallelism and auto-tuning, see Appendix E.1) as well as underlying libraries such as cuDNN and CUDA, they are likely to suffer from the same variance problem.

Deterministic mode of DL frameworks: A deterministic mode has been introduced to modern DL frameworks such as PyTorch and TensorFlow to tackle the nondeterminism from software implementations. However, at the time of writing, the deterministic mode cannot eliminate the fairness variance described in this paper (See Appendix E.2 for details). In addition, as the survey in our prior work [71] shows, over 80% of the respondents are unaware or unsure about the nondeterminism

caused by software implementations. DL researchers and practitioners would not use the deterministic mode if they are unaware of the fairness variance caused by DL software.

Negative social impact: With FIT runs to estimate the fairness variance of neural networks, our proposed approach could increase the power consumption and CO_2 emissions of DL model training. However, such additional evaluation is required for experimental validity and fairness compliance. In other words, the negative social impact of not performing multiple training runs includes disproportionally hurting certain races or genders. Our paper raises awareness of this issue and calls for new solutions to strike a balance. We are working on theoretically quantifying the upper bound of fairness variance and cheaper ways to estimate/bound variance.

7 Related work

Anecdotal evidence in fairness variance: Agrawal et al. [2] apply debiasing on the random forest classifiers and test for fairness variance with different random seeds and data splits for cross-validation. They find that debiasing algorithms could increase the fairness variance introduced by algorithmic nondeterminism. Different from their work, we focus on DL models and variance caused by DL software implementations, which cannot be removed by fixing random seeds.

Friedler et al. [30] investigate the stability of debiasing algorithms by measuring the standard deviation of fairness metrics over different dataset splits. However, they overlook the nondeterminism from the DL software implementations and only perform one training run for each split.

Fairness studies of ML systems: The fairness of ML and DL systems have gained attention and several studies pointed out fairness issues in datasets (e.g., CelebA [81], GitHub [41], and StackOverflow [61]) and major ML systems such as language models [13, 92, 95], face recognition [89], and sentiment analysis [88]. Such studies demonstrate that ML systems can be unfair and highlight the importance of fairness. Yet to the best of our knowledge, we are the first to study the impact of DL software implementation on fairness.

Bias metrics: Many metrics [6, 15, 19, 24, 26, 31, 35, 44, 46, 67, 82, 102, 103] have been proposed to measure biases of ML systems. Most of them can be applied to DL systems too. We evaluate fairness using seven of the most popular metrics and show that they are all impacted by DL implementation variance. None of the existing metrics considers such variance in their definition.

Debiasing algorithms: To mitigate biases of DL systems, many debiasing algorithms or bias resistant networks have been proposed and evaluated [10, 15, 16, 17, 45, 47, 54, 59, 66, 83, 98, 99, 106]. These techniques are important to improve the fairness of DL systems but their evaluation can be affected by fairness variance introduced by DL software implementations. Our study shows that the impact of some debiasing algorithms is not consistent during different identical runs.

8 Conclusion

This work studies the fairness variance of DL systems caused by software implementations. We examine five tasks on three networks and four datasets using 22 mitigation techniques. We find up to 12.6% difference in bias values across FIT runs and 68.0% of the experiments reduce model biases with cost in fairness variance, accuracy, or accuracy variance. Our paper survey shows that only 34.4% of the papers use multiple identical runs to evaluate the proposed approaches. We call for the awareness of the fairness variance of DL systems caused by software implementations.

Acknowledgments

The authors thank the anonymous reviewers for their constructive comments as well as the area chair and the senior area chair for overseeing the review process. This work has been partially supported by NSF 2006688 and a J.P. Morgan AI Faculty Research Award. We also thank NSERC for funding support. Any opinions, findings, and conclusions in this paper are those of the authors only and do not necessarily reflect the views of the sponsors.

References

[1] Tameem Adel, Isabel Valera, Zoubin Ghahramani, and Adrian Weller. 2019. One-network adversarial fairness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 2412–2420.

[2] Ashrya Agrawal, Florian Pfisterer, Bernd Bischl, Jiahao Chen, Srijan Sood, Sameena Shah, Francois Buet-Golfouse, Bilal A Mateen, and Sebastian J Vollmer. 2020. Debiasing classifiers: is reality at variance with expectation? (2020), 11 pages. arXiv:2011.02407

[3] Mohsan Alvi, Andrew Zisserman, and Christoffer Nellåker. 2019. Turning a blind eye: Explicit removal of biases and variation from deep neural network embeddings. In *Computer Vision (ECCV '18 Workshops)*. Springer, Cham, 556–572.

[4] Andrew L Beam, Arjun K Manrai, and Marzyeh Ghassemi. 2020. Challenges to the reproducibility of machine learning models in health care. *JAMA* 323, 4 (2020), 305–306.

[5] Arthur G Bedeian and Kevin W Mossholder. 2000. On the use of the coefficient of variation as a measure of diversity. *Organ. Res. Methods* 3, 3 (2000), 285–297.

[6] Omer Ben-Porat, Fedor Sandomirskiy, and Moshe Tennenholtz. 2021. Protecting the Protected Group: Circumventing Harmful Fairness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 5176–5184.

[7] Sebastian Benthall and Bruce D Haynes. 2019. Racial categories in machine learning. In *Conference on Fairness, Accountability, and Transparency (FAT* '19)*. ACM, New York, 289–298.

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

[9] Dan Biddle. 2006. *Adverse impact and test validation: A practitioner's guide to valid and defensible employment testing* (2 ed.). Routledge, London and New York.

[10] Sumon Biswas and Hridesh Rajan. 2020. Do the machine learning models on a crowd sourced platform exhibit bias? An empirical study on model fairness. In *28th Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE 2020)*. ACM, New York, 642–653.

[11] Emily Black, Samuel Yeom, and Matt Fredrikson. 2020. Fliptest: fairness testing via optimal transport. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. 111–121.

[12] Su Lin Blodgett and Brendan O'Connor. 2017. Racial disparity in natural language processing: A case study of social media african-american english. *arXiv preprint arXiv:1707.00061* (2017).

[13] Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in Neural Information Processing Systems (NIPS '16, Vol. 29)*. Curran Associates, Red Hook, NY, 4349–4357.

[14] Yuriy Brun and Alexandra Meliou. 2018. Software fairness. In *26th Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '18)*. ACM, New York, 754–759.

[15] Toon Calders and Sicco Verwer. 2010. Three naive Bayes approaches for discrimination-free classification. *Data Min. Knowl. Disc.* 21, 2 (2010), 277–292.

[16] Joymallya Chakraborty, Suvodeep Majumder, and Tim Menzies. 2021. Bias in Machine Learning Software: Why? How? What to Do?. In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (Athens, Greece) *(ESEC/FSE 2021)*. Association for Computing Machinery, New York, NY, USA, 429–440. https://doi.org/10.1145/3468264.3468537

[17] Joymallya Chakraborty, Suvodeep Majumder, Zhe Yu, and Tim Menzies. 2020. Fairway: a way to build fair ML software. In *28th Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '20)*. ACM, New York, 654–665.

[18] Jiahao Chen, Victor Storchan, and Eren Kurshan. 2021. Beyond Fairness Metrics: Roadblocks and Challenges for Ethical AI in Practice. *arXiv preprint arXiv:2108.06217* (2021).

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

[20] COCODataset. [n.d.]. *Common Objects in Context Terms of Use*. `https://cocodataset.org/#termsofuse` Last accessed 2021-05-25.

[21] Elliot Creager, David Madras, Jörn-Henrik Jacobsen, Marissa Weis, Kevin Swersky, Toniann Pitassi, and Richard Zemel. 2019. Flexibly fair representation learning by disentanglement. In *International Conference on Machine Learning*. PMLR, 1436–1445.

[22] Sunipa Dev, Tao Li, Jeff M Phillips, and Vivek Srikumar. 2020. On measuring and mitigating biased inferences of word embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 7659–7666.

[23] PyTorch Developers. [n.d.]. Torch.use_deterministic_algorithms. `https://pytorch.org/docs/stable/generated/torch.use_deterministic_algorithms.html#torch.use_deterministic_algorithms`

[24] Mengnan Du, Fan Yang, Na Zou, and Xia Hu. 2020. Fairness in deep learning: A computational perspective. *IEEE Intelligent Systems* (2020), 7 pages.

[25] Sanghamitra Dutta, Dennis Wei, Hazar Yueksel, Pin-Yu Chen, Sijia Liu, and Kush Varshney. 2020. Is there a trade-off between fairness and accuracy? A perspective using mismatched hypothesis testing. In *International Conference on Machine Learning (ICML '20, Vol. 119)*. PMLR, 2803–2813.

[26] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. 2012. Fairness through awareness. In *3rd Innovations in Theoretical Computer Science Conference (ITCS '12)*. ACM, New York, 214–226.

[27] Executive Office of the President. 2016. Big data: A report on algorithmic systems, opportunity, and civil rights.

[28] DS Faber and H Korn. 1991. Applicability of the coefficient of variation method for analyzing synaptic plasticity. *Biophys. J.* 60, 5 (1991), 1288–1294.

[29] Michael Feldman, Sorelle A Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. 2015. Certifying and removing disparate impact. In *21st SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '15)*. ACM, New York, 259–268.

[30] Sorelle A Friedler, Carlos Scheidegger, Suresh Venkatasubramanian, Sonam Choudhary, Evan P Hamilton, and Derek Roth. 2019. A comparative study of fairness-enhancing interventions in machine learning. In *Conference on Fairness, Accountability, and Transparency*. 329–338.

[31] Stephen Gillen, Christopher Jung, Michael Kearns, and Aaron Roth. 2018. Online learning with an unknown fairness metric. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. 2605–2614.

[32] Naman Goel, Alfonso Amayuelas, Amit Deshpande, and Amit Sharma. 2021. The Importance of Modeling Data Missingness in Algorithmic Fairness: A Causal Perspective. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 7564–7573.

[33] Aditya Grover, Jiaming Song, Ashish Kapoor, Kenneth Tran, Alekh Agarwal, Eric J Horvitz, and Stefano Ermon. 2019. Bias Correction of Learned Generative Models using Likelihood-Free Importance Weighting. *Advances in Neural Information Processing Systems* 32 (2019).

[34] Teng Guo, Feng Xia, Shihao Zhen, Xiaomei Bai, Dongyu Zhang, Zitao Liu, and Jiliang Tang. 2020. Graduate employment prediction with bias. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 670–677.

[35] Moritz Hardt, Eric Price, and Nati Srebro. 2016. Equality of opportunity in supervised learning. In *Advances in Neural Information Processing Systems (NIPS'16, Vol. 29)*. Curran Associates, Red Hook, NY, 3315–3323.

[36] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR '16)*. IEEE, New York, 770–778.

[37] Lisa Anne Hendricks, Kaylee Burns, Kate Saenko, Trevor Darrell, and Anna Rohrbach. 2018. Women also snowboard: Overcoming bias in captioning models. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 771–787.

[38] Lisa Anne Hendricks, Kaylee Burns, Kate Saenko, Trevor Darrell, and Anna Rohrbach. 2018. Women also Snowboard: Overcoming Bias in Captioning Models (Extended Abstract). arXiv:1807.00517 [cs.CV]

[39] Kenneth Holstein, Jennifer Wortman Vaughan, Hal Daumé III, Miro Dudik, and Hanna Wallach. 2019. Improving fairness in machine learning systems: What do industry practitioners need?. In *Proceedings of the 2019 CHI conference on human factors in computing systems*. 1–16.

[40] IBM. 2018. Pytorch-seq2seq/loss.py at F146087A9A271E9B50F46561E090324764B081FB. `https://tinyurl.com/ibm-seq2seq`

[41] Nasif Imtiaz, Justin Middleton, Joymallya Chakraborty, Neill Robson, Gina Bai, and Emerson Murphy-Hill. 2019. Investigating the effects of gender bias on GitHub. In *41st International Conference on Software Engineering (ICSE '19)*. IEEE/ACM, New York, 700–711.

[42] Jörn-Henrik Jacobsen, Arnold W M Smeulders, and Edouard Oyallon. 2018. *i*-RevNet: Deep invertible networks. In *6th International Conference on Learning Representations (ICLR '18)*. 11 pages.

[43] Christoph Jansen, Jonas Annuscheit, Bruno Schilling, Klaus Strohmenger, Michael Witt, Felix Bartusch, Christian Herta, Peter Hufnagl, and Dagmar Krefting. 2020. Curious Containers: A framework for computational reproducibility in life sciences with support for Deep Learning applications. *Future Gener. Comp. Sy.* 112 (2020), 209–227.

[44] Nathan Kallus and Angela Zhou. 2019. The fairness of risk scores beyond classification: Bipartite ranking and the xauc metric. *Advances in neural information processing systems* 32 (2019), 3438–3448.

[45] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. 2012. Fairness-aware classifier with prejudice remover regularizer. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD '12)*. Springer, Berlin Heidelberg, 35–50.

[46] Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. 2018. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness. In *35th International Conference on Machine Learning (ICML '18, Vol. 80)*. PMLR, Stockholm, 2564–2572.

[47] Thomas Kehrenberg, Myles Bartlett, Oliver Thomas, and Novi Quadrianto. 2020. Null-sampling for Interpretable and Fair Representations. In *Computer Vision (ECCV '20)*. Springer, Cham, 565–580.

[48] Hyemi Kim, Seungjae Shin, JoonHo Jang, Kyungwoo Song, Weonyoung Joo, Wanmo Kang, and Il-Chul Moon. 2021. Counterfactual Fairness with Disentangled Causal Effect Variational Autoencoder. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 8128–8136.

[49] Joon Sik Kim, Jiahao Chen, and Ameet Talwalkar. 2020. FACT: A diagnostic for group fairness trade-offs. In *37th International Conference on Machine Learning (PMLR, Vol. 119)*. PMLR, 5264–5274.

[50] Jon Kleinberg, Sendhil Mullainathan, and Manish Raghavan. 2017. Inherent trade-offs in the fair determination of risk scores. In *8th Innovations in Theoretical Computer Science Conference (ITCS '17) (LIPIcs, Vol. 67)*. Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 43:1–23.

[51] Alex Krizhevsky. 2009. *Learning multiple layers of features from tiny images*. Master's thesis. University of Toronto.

[52] Eren Kurshan, Hongda Shen, and Jiahao Chen. 2020. Towards Self-Regulating AI: Challenges and Opportunities of AI Model Governance in Financial Services. In *Proceedings of the First ACM International Conference on AI in Finance* (New York, New York) *(ICAIF '20)*. Association for Computing Machinery, New York, NY, USA, Article 49, 8 pages. `https://doi.org/10.1145/3383455.3422564`

[53] Gopiram Roshan Lal, Sahin Cem Geyik, and Krishnaram Kenthapadi. 2020. Fairness-Aware Online Personalization. (2020), 10 pages. arXiv:2007.15270

[54] Peizhao Li, Han Zhao, and Hongfu Liu. 2020. Deep fair clustering for visual learning. In *Conference on Computer Vision and Pattern Recognition (CVPR '20)*. IEEE/CVF, New York, 9070–9079.

[55] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft COCO: Common objects in context. In *Computer Vision (ECCV '14)*. Springer, Cham, 740–755.

[56] Chao Liu, Cuiyun Gao, Xin Xia, David Lo, John Grundy, and Xiaohu Yang. 2020. On the Replicability and Reproducibility of Deep Learning in Software Engineering. (2020). arXiv:2006.14244

[57] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2015. Deep Learning Face Attributes in the Wild. In *International Conference on Computer Vision (ICCV '15)*. IEEE, New York, 3730–3738.

[58] Steve Lohr. 2018. Facial recognition is accurate, if you're a white guy. *New York Times* 9 (2018), 8.

[59] Vishnu Suresh Lokhande, Aditya Kumar Akash, Sathya N Ravi, and Vikas Singh. 2020. FairALM: Augmented Lagrangian Method for Training Fair Models with Little Regret. In *Computer Vision (ECCV '20)*. Springer, Cham, 365–381.

[60] David Madras, Elliot Creager, Toniann Pitassi, and Richard Zemel. 2018. Learning Adversarially Fair and Transferable Representations. In *35th International Conference on Machine Learning (ICML '18, Vol. 80)*. PMLR, Stockholm, 3384–3393.

[61] Anna May, Johannes Wachs, and Anikó Hannák. 2019. Gender differences in participation and reward on Stack Overflow. *Empir. Softw. Eng.* 24, 4 (2019), 1997–2019.

[62] Daniel McDuff, Shuang Ma, Yale Song, and Ashish Kapoor. 2019. Characterizing bias in classifiers using generative models. In *Advances in Neural Information Processing Systems (NeurIPS '19, Vol. 32)*. Curran Associates, Red Hook, NY, 12 pages.

[63] Ninareh Mehrabi, Muhammad Naveed, Fred Morstatter, and Aram Galstyan. 2021. Exacerbating Algorithmic Bias through Fairness Attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 8930–8938.

[64] Aditya Krishna Menon and Robert C Williamson. 2018. The cost of fairness in binary classification. In *1st Conference on Fairness, Accountability and Transparency (FAT* '18, Vol. 81)*. PMLR, New York, 107–118.

[65] Blossom Metevier, Stephen Giguere, Sarah Brockman, Ari Kobren, Yuriy Brun, Emma Brunskill, and Philip S Thomas. 2019. Offline Contextual Bandits with High Probability Fairness Guarantees. *Advances in Neural Information Processing Systems* 32 (2019), 14922–14933.

[66] Shaobo Min, Hantao Yao, Hongtao Xie, Chaoqun Wang, Zheng-Jun Zha, and Yongdong Zhang. 2020. Domain-aware visual bias eliminating for generalized zero-shot learning. In *Conference on Computer Vision and Pattern Recognition (CVPR '20)*. IEEE/CVF, New York, 12664–12673.

[67] Debarghya Mukherjee, Mikhail Yurochkin, Moulinath Banerjee, and Yuekai Sun. 2020. Two simple ways to learn individual fairness metrics from data. In *International Conference on Machine Learning*. PMLR, 7097–7107.

[68] Razieh Nabi and Ilya Shpitser. 2018. Fair inference on outcomes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[69] Jun Hyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. 2020. Learning from Failure: De-biasing Classifier from Biased Classifier. In *Advances in Neural Information Processing Systems (NeurIPS '20, Vol. 33)*. Curran Associates, Red Hook, NY, 20673–20684.

[70] Claudia Perlich, Brian Dalessandro, Troy Raeder, Ori Stitelman, and Foster Provost. 2014. Machine learning for targeted display advertising: Transfer learning in action. *Mach. Learn.* 95, 1 (2014), 103–127.

[71] Hung Viet Pham, Shangshu Qian, Jiannan Wang, Thibaud Lutellier, Jonathan Rosenthal, Lin Tan, Yaoliang Yu, and Nachiappan Nagappan. 2020. Problems and Opportunities in Training Deep Learning Software Systems: An Analysis of Variance. In *35th International Conference on Automated Software Engineering (ASE '20)*. IEEE/ACM, New York, 771–783.

[72] Joelle Pineau, Philippe Vincent-Lamarre, Koustuv Sinha, Vincent Larivière, Alina Beygelzimer, Florence d'Alché Buc, Emily Fox, and Hugo Larochelle. 2020. Improving reproducibility in machine learning research (a report from the NeurIPS 2019 Reproducibility Program). (2020), 22 pages. arXiv:2003.12206

[73] Martin F Porter. 2001. Snowball: A language for stemming algorithms.

[74] Reid Pryzant, Richard Diehl Martinez, Nathan Dass, Sadao Kurohashi, Dan Jurafsky, and Diyi Yang. 2020. Automatically neutralizing subjective bias in text. In *Proceedings of the aaai conference on artificial intelligence*, Vol. 34. 480–489.

[75] Novi Quadrianto, Viktoriia Sharmanska, and Oliver Thomas. 2019. Discovering fair representations in the data domain. In *Conference on Computer Vision and Pattern Recognition (CVPR '19)*. IEEE/CVF, New York, 8227–8236.

[76] Inioluwa Deborah Raji, Timnit Gebru, Margaret Mitchell, Joy Buolamwini, Joonseok Lee, and Emily Denton. 2020. Saving face: Investigating the ethical concerns of facial recognition auditing. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. 145–151.

[77] George F Reed, Freyja Lynn, and Bruce D Meade. 2002. Use of coefficient of variation in assessing variability of quantitative assays. *Clin. Diagn. Lab. Immun.* 9, 6 (2002), 1235–1239.

[78] Félix Renard, Soulaimane Guedria, Noel De Palma, and Nicolas Vuillerme. 2020. Variability and reproducibility in deep learning for medical image segmentation. *Sci. Rep.* 10, 1 (2020), 1–16.

[79] Ashkan Rezaei, Anqi Liu, Omid Memarrast, and Brian D Ziebart. 2021. Robust Fairness Under Covariate Shift. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 9419–9427.

[80] Chris Rockwell and David F Fouhey. 2020. Full-body awareness from partial observations. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVII 16*. Springer, 522–539.

[81] Hee Jung Ryu, Hartwig Adam, and Margaret Mitchell. 2017. InclusiveFaceNet: Improving face attribute detection with race and gender diversity. In *Workshop on Fairness, Accountability, and Transparency in Machine Learning (FAT/ML '18)*. 6 pages. arXiv:1712.00193

[82] Sivan Sabato and Elad Yom-Tov. 2020. Bounding the fairness and accuracy of classifiers from population statistics. In *International Conference on Machine Learning*. PMLR, 8316–8325.

[83] Mhd Hasan Sarhan, Nassir Navab, Abouzar Eslami, and Shadi Albarqouni. 2020. Fairness by learning orthogonal disentangled representations. In *Computer Vision (ECCV '20)*. Springer, Cham, 746–761.

[84] Yash Savani, Colin White, and Naveen Sundar Govindarajulu. 2020. Intra-Processing Methods for Debiasing Neural Networks. In *Advances in Neural Information Processing Systems (NeurIPS '20, Vol. 33)*. Curran Associates, Red Hook, NY, 2798–2810.

[85] Shlomo S Sawilowsky. 2009. New effect size rules of thumb. *J. Mod. Appl. Stat. Methods* 8, 2 (2009), 597–599.

[86] Oscar Schwartz. 2019. In 2016 Microsoft's Racist Chatbot Revealed the Dangers of Online Conversation. *IEEE Spectrum* 11 (2019).

[87] Procheta Sen and Debasis Ganguly. 2020. Towards Socially Responsible AI: Cognitive Bias-Aware Multi-Objective Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 2685–2692.

[88] Judy Hanwen Shen, Lauren Fratamico, Iyad Rahwan, and Alexander M Rush. 2018. *Darling or babygirl?* Investigating stylistic bias in sentiment analysis. In *Workshop on Fairness, Accountability, and Transparency in Machine Learning (FAT/ML '18)*.

[89] Richa Singh, Akshay Agarwal, Maneet Singh, Shruti Nagpal, and Mayank Vatsa. 2020. On the robustness of face recognition algorithms against attacks and bias. In *Conference on Artificial Intelligence (AAAI '20, Vol. 34)*. AAAI, Palo Alto, CA, 13583–13589.

[90] Koustuv Sinha, Joelle Pineau, Jessica Forde, Rosemary Nan Ke, and Hugo Larochelle. 2020. NeurIPS 2019 reproducibility challenge. *ReScience C* 6, 2 (2020), 11.

[91] Jesper B Sørensen. 2002. The use and misuse of the coefficient of variation in organizational demography research. *Sociol. Method Res.* 30, 4 (2002), 475–491.

[92] Yi Chern Tan and L Elisa Celis. 2019. Assessing social and intersectional biases in contextualized word representations. In *Advances in Neural Information Processing Systems (NeurIPS '19, Vol. 32)*. Curran Associates, Red Hook, NY, 12 pages.

[93] Flickr Team. [n.d.]. *Flickr Terms of Use*. `https://www.flickr.com/creativecommons/` Last accessed 2021-05-25.

[94] Yuchi Tian, Ziyuan Zhong, Vicente Ordonez, Gail Kaiser, and Baishakhi Ray. 2020. Testing DNN image classifiers for confusion & bias errors. In *42nd International Conference on Software Engineering (ICSE '20)*. ACM/IEEE, New York, 1122–1134.

[95] Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. 2020. Investigating gender bias in language models using causal mediation analysis. In *Advances in Neural Information Processing Systems (NeurIPS '20, Vol. 33)*. Curran Associates, Red Hook, NY, 12388–12401.

[96] Christina Wadsworth, Francesca Vera, and Chris Piech. 2018. Achieving fairness through adversarial learning: an application to recidivism prediction. *arXiv preprint arXiv:1807.00199* (2018).

[97] Mei Wang and Weihong Deng. 2020. Mitigating bias in face recognition using skewness-aware reinforcement learning. In *Conference on Computer Vision and Pattern Recognition (CVPR '20)*. IEEE/CVF, New York, 9322–9331.

[98] Tianlu Wang, Jieyu Zhao, Mark Yatskar, Kai-Wei Chang, and Vicente Ordonez. 2019. Balanced datasets are not enough: Estimating and mitigating gender bias in deep image representations. In *International Conference on Computer Vision (ICCV '19)*. IEEE, New York, 5310–5319.

[99] Zeyu Wang, Klint Qinami, Ioannis Christos Karakozis, Kyle Genova, Prem Nair, Kenji Hata, and Olga Russakovsky. 2020. Towards fairness in visual recognition: Effective strategies for bias mitigation. In *Conference on Computer Vision and Pattern Recognition (CVPR '20)*. IEEE/CVF, New York, 8919–8928.

[100] Michael Wick, Swetasudha Panda, and Jean-Baptiste Tristan. 2019. Unlocking fairness: a trade-off revisited. In *Advances in Neural Information Processing Systems (NeurIPS '19, Vol. 32)*. Curran Associates, Red Hook, NY, 10 pages.

[101] Allison Woodruff, Sarah E Fox, Steven Rousso-Schindler, and Jeffrey Warshaw. 2018. A qualitative exploration of perceptions of algorithmic fairness. In *Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, 14 pages.

[102] Sirui Yao and Bert Huang. 2017. Beyond parity: fairness objectives for collaborative filtering. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 2925–2934.

[103] Sirui Yao and Bert Huang. 2017. New fairness metrics for recommendation that embrace differences. *arXiv preprint arXiv:1706.09838* (2017).

[104] Mark Yatskar, Luke Zettlemoyer, and Ali Farhadi. 2016. Situation recognition: Visual semantic role labeling for image understanding. In *Conference on Computer Vision and Pattern Recognition*. IEEE, New York, 5534–5542.

[105] Mikhail Yurochkin and Yuekai Sun. 2021. SenSeI: Sensitive Set Invariance for Enforcing Individual Fairness. In *ICLR 2021: The Ninth International Conference on Learning Representations*.

[106] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rogriguez, and Krishna P Gummadi. 2017. Fairness Constraints: Mechanisms for Fair Classification. In *International Conference on Artificial Intelligence and Statistics (AISTATS '17, Vol. 54)*. PMLR, Fort Lauderdale, 962–970.

[107] Brian Hu Zhang, Blake Lemoine, and Margaret Mitchell. 2018. Mitigating unwanted biases with adversarial learning. In *Conference on AI, Ethics, and Society (AIES '18)*. AAAI/ACM, New York, 335–340.

[108] Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2017. Men Also Like Shopping: Reducing Gender Bias Amplification using Corpus-level Constraints. In *Conference on Empirical Methods in Natural Language Processing (EMNLP '17)*. ACL, Stroudsburg, PA, 2979–2989.

A Bias Metrics

A.1 Symbols

For a binary classification task, we abbreviate the content in the confusion matrix as TP, FN, TN, and FP. Positive (P) instances include both TP and FN, while negative (N) instances include both TN and FP.

Prediction positive rate (PPR), *true positive rate* (TPR), and *false positive rate* (FPR) are derived from the confusion matrix and describe a binary classifier. PPR is the ratio of instances classified as positive to the instances in the whole dataset ($\frac{TP+FP}{P+N}$). TPR is the ratio of TP to the positive instances ($\frac{TP}{P}$). FPR is the ratio of FP to the negative instances ($\frac{FP}{N}$).

Each protected group g takes $\Pr[g]$ ($\frac{P_g+N_g}{P+N}$) of the dataset. We note the confusion matrix and the derived metrics for group g by adding an annotation to the terms (e.g., TP_g, FP_g, P_g, N_g, PPR_g, TPR_g, and FPR_g).

A.2 Selected Bias Metrics

Condition 1: Statistical Parity (SP) [26]: $PPR_0 = PPR_1$

- Demographic Parity [15]: **DP** $= |PPR_0 - PPR_1|$
- Normalized Disparate Impact [15]: $\overline{\textbf{DI}} = 1 - \min(\frac{PPR_0}{PPR_1}, \frac{PPR_1}{PPR_0})$
- Statistical Parity Subgroup Fairness [46]: **SPSF** $= \sum_{g \in G} \Pr[g] \times |PPR - PPR_g|$

Condition 2: Predictive Equality (PE) [19]: $FPR_0 = FPR_1$

- False Positive Subgroup Fairness [46]: **FPSF** $= \sum_{g \in G} \Pr[g] \times |FPR - FPR_g|$
- Equalized Odds (False Positive) [24]: **EOFP** $= |FPR_0 - FPR_1|$

Condition 3: Equality of Opportunity (EOp) [35]: $TPR_0 = TPR_1$

- Equalized Odds (True Positive) [35]: **EOTP** $= |TPR_0 - TPR_1|$

The last bias metric, bias amplification [99, 108] does not fit any existing fairness conditions.

- **BA** $= \left| \frac{TP_{\bar{g}} + FP_{\bar{g}}}{TP + FP} - \frac{P_{\bar{g}}}{P} \right|$, where $\bar{g} = \underset{g}{\operatorname{argmax}} \frac{P_g}{P}$

B Details of Reproduced Studies

B.1 Tasks

(1) Multi-class image classification in CIFAR-10S [99]: This study classifies images from the CIFAR-10S dataset into ten task labels. CIFAR-10S is designed to study fairness issues and debiasing algorithms. Its protected label is whether the images are color or grayscale. Nine different debiasing algorithms (and a baseline) have been evaluated with this dataset using the popular ResNet-18 network [36].

License: Either CIFAR-10 [51] or CIFAR-10S [99] datasets did not specify the type of license used.

(2) Multi-label classification in MS-COCO [98]: The MS-COCO dataset [55] contains images with several objects (i.e., several labels in each image). In their fairness study, Wang et al. [98] find that this dataset can leak gender information (i.e., gender is the protected label) and study five different debiasing algorithms (plus one baseline) on ResNet-50 [36] models trained on MS-COCO.

License: The annotations [20] released under Creative Commons Attribution 4.0 License, and the images from MS-COCO datasets must abide by Flicker Terms of Use [93].

(3) Multi-class classification in imSitu [98]: This task is the second task used for benchmarking debiasing algorithms in previous work [98]. The task labels in imSitu [104] are actions (e.g., cooking), and the protected label is gender. Wang et al. [98] compare the same five debiasing algorithms as for task (2) with the same network.

License: The images of the imSitu dataset is gathered from Google Image Search [104], and the annotations are labeled by Amazon Mechanical Turk [104]. The authors did not specify the type of license used.

(4) Binary classification in CelebA [59]: Lokhande et al. [59] propose a new debiasing algorithm (FairALM) and compare it to a baseline and another debiasing algorithm on a ResNet-18 model trained on the CelebA dataset [57]. CelebA contains faces of celebrities with several binary task labels and two protected labels (gender and youth). Task (4) uses the task label "attractive" and the protected label "gender".

License: The CelebA dataset is available for non-commercial research purposes only [57].

(5) Binary classification in CelebA with NIFR network [47]: For the final task, we reproduced another work [47] that uses the CelebA dataset. Key differences with task (4) are the use of NIFR network, different debiasing algorithms, and a different task label ("smile" instead of "attractive").

B.2 Debiasing Algorithms

Data rebalancing rebalances the training data to either be perfectly balanced (techniques S-RS, C-R1, I-R1) or within a certain ratio between the number of instances from each protected group (techniques C-R2, C-R3, I-R2, I-R3).

Fairness through blindness removes the protected label and creates an intermediate representation for each instance. Then a downstream classifier is trained to predict the task label on this representation. Adversarial training is the most popular debiasing algorithm within such category (techniques S-UC, S-GR, C-A4, C-A5, I-A4, I-A5). Recently, invertible neural networks [42] are also used to remove sensitive information (technique N-Flow).

Fairness through awareness consists of two categories: (1) the debiasing algorithm uses a loss function on protected labels to penalize the classifier's biases (techniques A-L2, A-ALM); and (2) the debiasing algorithm involves a fine-grained classifier that predicts the task label and the protected label of each instance. Before the final prediction on the task label, the predicted protected label is used to remove possible biases in the final prediction (techniques S-DD1, S-DD2, S-DD3, S-DD4, S-DI1, S-DI2). One concern of this debiasing algorithm is the waste of classification power to discriminate the boundaries between each protected group [99].

B.3 Statistical Tests

Mann-Whitney U-test: One-sided U-test is used in this paper to compare the mean of two sample distributions. U-test is chosen instead of T-test because it does not require normality in the sample distribution. For two experiments A and B, the null hypothesis is that the mean bias value of 16 runs in experiment A is similar to that in experiment B. We choose a significance level of 5%, i.e., if we get a p-value < 0.05, we are 95% confident that experiment A has a statistically different mean bias value than experiment B.

Cohen's d: In addition to the p-value from U-test, we use Cohen's d to quantify the impact of debiasing algorithms. We use the empirical interpretation of effect size [85] to represent Cohen's d values (e.g., Cohen's d of 0.8 represents a "large" impact while a 2.0 represents a "huge" impact).

Levene's test: We use Levene's test to compare the magnitude of the variance between experiment runs. For two experiments A and B, the null hypothesis is that the variance of the bias value from 16 runs of experiment A is similar to that of experiment B. The significance level is also 5%.

B.4 Hardware and Software

We conduct our experiments on a server with two Xeon Gold 5120 CPUs (56 cores in total) and 384 GB of memory. We use an RTX 2080Ti graphics card with 11GB of memory for each training run.

Table 3: Confusion matrix and bias metric values for a biased image classifier. “Sport” and “Cook” are the number of instances of real labels of “Sport" and “Cook” activity in the image. “Pred-Sport” and “Pred-Cook” are the number of instances of the predicted labels. "Male" and "Female" are the protected groups.

	Female (200)		**Male (200)**		**Total**
	Pred-Sport	Pred-Cook	Pred-Sport	Pred-Cook	
Sport	80	20	60	40	200
Cook	10	90	70	30	200
Total	90	110	130	70	400

Metric	DP	$\overline{\text{DI}}$	SPSF	FPSF	EOFP	EOTP	BA
Value (%)	20.0	33.6	10.0	20.0	40.0	40.0	10.1

We use CUDA 10.0 and cuDNN 7.6, TensorFlow 1.15, and PyTorch 1.7 DL libraries. Each run is isolated using Docker.

C Bias Metrics Running Example

We use the running example in Table 3 to explain SPSF, DP, and BA in detail. Table 3 shows the prediction results from a biased binary classifier and its bias values using the seven metrics. The classifier assigns labels “Sport” and “Cook” with respect to the activity in the image, which is a common classification task with DL [37]. For example, for the 200 images containing a woman, 80 images doing sport are classified as “Sport”, while another 20 of them are misclassified as “Cook”. The protected label is gender, i.e., whether the person inside the image is “Male” or “Female” (nonbinary genders exist but are not considered in this example). The ground-truth labels of the dataset are balanced, i.e., 100 instances for each task label in each protected group. Bias metrics are calculated on both task labels, and the average of the two is the bias of the model.

We choose to balance the protected group in the sample dataset because we would like to stress that even with a balanced dataset, the trained model could still be biased. With unbalanced groups, the bias could be even bigger [99, 108], and the biases from the best scenario are already worrisome.

(1) Demographic Parity (DP) [15]: DP is measured by the difference of the PPR between different protected classes. Without losing generality, we consider “Sport” the positive class in the binary classifier. Following the DP formula in Appendix A.2, for the “Sport” class, the PPR_{female} is 45.0% (90 / 200), and PPR_{male} is 65.0% (130 / 200), a difference of 20.0%, i.e., the male group is 20.0% more likely to be labeled as “Sport”. The DP bias metric for the “Sport” class is 20.0% $(110/200 - 70/200)$. Similarly, the DP for the “Cook” class is 20.0%. Therefore, the model has a DP bias of 20.0%, which is the average of the two per-class biases.

(2) Disparate Impact (DI, $\overline{\text{DI}}$) [15]: DI measures the ratio of PPR between protected groups. The less protected groups are discriminated against, the closer DI gets to 1. In this paper, we measure Normalized DI ($\overline{\text{DI}}$) as the difference between 1 and DI ($\overline{\text{DI}} = 1 - \text{DI}$) to be consistent with other bias metrics (i.e., the model is completely fair when $\overline{\text{DI}} = 0$, and unfair when $\overline{\text{DI}} = 1$).

(3) Statistical Parity Subgroup Fairness (SPSF) [46]: SPSF is also known as the equality of classification rates. The goal of SPSF is that all protected classes have the same PPR as the whole dataset.

PPR is 55.0% (220 / 400) for the dataset. PPR_{male} is 65.0% (130 / 200), 10.0% more likely to be predicted as positive. PPR_{female} is 45.0% (90 / 200), 10.0% less likely to be predicted positive. The difference in PPR for each group is both 10.0%, and each group makes up 50.0% of the instances (Pr$[\text{male}]$ and Pr$[\text{female}]$). Therefore, the SPSF bias for the “Sport” class is 10.0%. Similarly, the SPSF bias for the “Cook” class is also 10.0%. The overall SPSF bias is the weighted average of the two: 10.0%.

(3) False Positive Subgroup Fairness: (FPSF) [46] Similar to SPSF, FPSF requires each protected class to have the same false positive rate as the whole dataset. Compared with SPSF, this metric considers the accuracy of the classifier. It should be used with SPSF together to measure the fairness of the classifier.

(5) and (6) Equalized Odds (EO) [24, 35]: Equalized Odds was originally introduced in recidivism prediction. There are two variants of this metric. One requires the same TPR, and the other requires the same FPR across all protected classes. We name these two variants as Equalized Odds - True Positive (EOTP) and Equalized Odds - False Positive (EOFP) by their definition.

The female group has a TPR of 80% (80 / (80 + 20)), and the male group has a TPR of 60% (60 / (60 + 40)). The difference between TPR_{female} and TPR_{male} is 20%, which is a EOTP of 0.2. Similarly, the EOTP for "Cook" class is 0.6. And the classifier has a EOTP of 0.4.

Similar to the process in EOTP, the classifier has a EOFP of 0.4 on this dataset.

(7) Bias Amplification (BA) [99, 108]: Unlike the six bias metrics above, which only look at the outcome of a classifier, bias amplification separates the bias in the dataset and the bias caused by the algorithm. It assumes a pre-existing skew inside the dataset (e.g., gender bias), and measures how much the algorithm amplifies such skew.

The "Sport" class has 100 male ($\frac{P_{\text{male}}}{P} = 50.0\%$) and 100 female ($\frac{P_{\text{female}}}{P} = 50.0\%$) instances. In the prediction result, there are 90 female ($\frac{TP_{\text{female}}+FP_{\text{female}}}{TP+FP} = \frac{80+10}{140+80} = 40.9\%$) and 130 male ($\frac{TP_{\text{male}}+FP_{\text{male}}}{TP+FP} = \frac{60+70}{140+80} = 59.1\%$) instances. The prediction result for the "Sport" class is 9.1% more biased towards the male group than the dataset, and the model *amplifies bias* towards the male group by 9.1%, which is a BA of 9.1% for the "Sport" class.

Similarly, for the "Cook" class, the model *amplifies bias* toward the female group by 11.1% (BA of 11.1%). Therefore, the classifier has a BA of 10.1% (average of 9.1% and 11.1%), i.e., *amplifies bias* by 10.1%.

D Additional Results

D.1 RQ1

Table 4 shows the aggregated results for all of the 27 techniques evaluated with $\overline{\text{DI}}$ metric. This table, together with Table 2, shows that fairness variance is not tied to a particular debiasing algorithm nor a particular bias metric. A per-class absolute maximum difference of 100% in $\overline{\text{DI}}$ metric occurred for multiple techniques (C-Base, C-R3, C-A4, and C-A5).

D.2 RQ3

Figure 3 shows the cost of mitigation techniques on the model accuracy and the variance of accuracy. Mann-Whitney U-test confirms the cost in model accuracy and Levene's test confirms the cost in accuracy variance. The upper subgraph is the cost on model accuracy, and the lower subgraph is the cost on variance of model accuracy. A blue block indicates an increase in model accuracy or a decrease in the variance of model accuracy. Oppositely, an orange block is used. A grey block indicates no statistically significant change. An "X" inside the block indicates that the mitigation technique fails to improve any of the bias metrics tested.

D.3 Literature Survey

Table 5 shows the detailed results of the 32 papers related to DL fairness in our literature survey. Column "Paper" cites the each paper, and column "Use multiple runs" indicates whether one paper uses multiple training runs to ensure validity or not. The other three columns indicate whether a paper reports the average value ("Mean"), standard deviation ("STDEV"), and range ("Range"). A "Y" means "yes", a "N" means "no", and a "-" indicates the column is not applicable for the paper.

From the table, only 11 out of 32 papers use multiple training runs to evaluate the proposed approaches. All of the papers that use multiple runs report mean values but some of them only report the standard deviation instead of range. We believe it is important to report all three statistical values as the range

Table 4: Maximum differences of $\overline{\text{DI}}$ values for all techniques

Technique	Overall (%)		Per-class (%)	
	MaxDiff	STDEV	MaxDiff	STDEV
S-Base	2.0	0.5	7.0	1.9
S-RS	2.0	0.5	5.8	1.2
S-UC	5.1	1.7	18.4	4.6
S-GR	**11.8**	3.5	50.1	12.7
S-DD1	2.3	0.6	5.4	1.5
S-DD2	1.5	0.4	5.3	1.7
S-DD3	1.4	0.4	5.7	1.7
S-DD4	1.0	0.2	8.0	1.8
S-DI1	1.8	0.5	5.0	1.4
S-DI2	1.1	0.3	4.2	1.0
C-Base	2.6	0.8	**100.0**	38.9
C-R1	1.0	0.3	50.0	19.5
C-R2	2.8	0.8	62.5	21.1
C-R3	3.7	1.1	**100.0**	29.9
C-A4	3.7	0.8	**100.0**	22.8
C-A5	2.4	0.6	**100.0**	28.2
I-Base	0.8	0.2	58.9	22.6
I-R1	0.9	0.3	50.0	14.8
I-R2	0.7	0.2	30.0	8.1
I-R3	1.2	0.4	44.7	13.4
I-A4	0.8	0.2	40.0	14.5
I-A5	1.0	0.3	50.0	23.2
A-Base	1.7	0.5	5.8	1.8
A-L2	8.0	2.4	31.1	8.8
A-ALM	4.5	1.2	9.3	3.0
N-Base	0.0	0.0	0.0	0.0
N-Flow	2.5	0.6	3.6	0.9

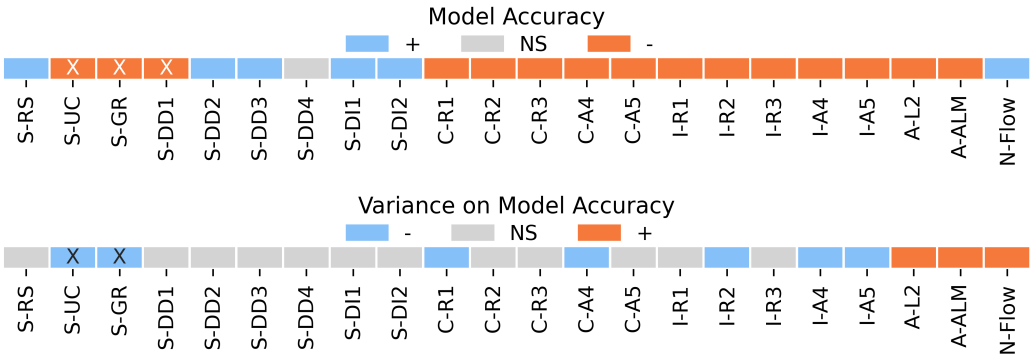

Figure 3: Cost of bias mitigation techniques on model accuracy and accuracy variance

could indicate the worst (or the best) case scenario, which is critical to the compliance requirements of the trained models.

E Discussion of DL Software Nondeterministic

E.1 Causes

As our prior work [71] pointed out, nondeterminism in DL software can be divided into two parts: algorithmic nondeterminism and implementation-level nondeterminism.

Table 5: Result of literature survey. "Y" and "N" under each column indicate whether the corresponding paper uses multiple training runs or reports statistical values.

Paper	Use multiple runs	Mean	STDEV	Range
Li et al. [54]	N	-	-	-
Min et al. [66]	N	-	-	-
Wang and Deng [97]	N	-	-	-
Wang et al. [99]	Y	Y	Y	Y
Quadrianto et al. [75]	Y	Y	Y	N
Wang et al. [98]	N	-	-	-
Lokhande et al. [59]	Y	Y	Y	Y
Sarhan et al. [83]	N	-	-	-
Rockwell and Fouhey [80]	N	-	-	-
Kehrenberg et al. [47]	N	-	-	-
Hendricks et al. [37]	N	-	-	-
Savani et al. [84]	Y	Y	N	Y
Vig et al. [95]	N	-	-	-
Nam et al. [69]	Y	Y	N	Y
Tan and Celis [92]	N	-	-	-
Grover et al. [33]	Y	Y	Y	Y
McDuff et al. [62]	Y	Y	N	Y
Creager et al. [21]	N	-	-	-
Pryzant et al. [74]	N	-	-	-
Guo et al. [34]	N	-	-	-
Dev et al. [22]	Y	Y	N	Y
Singh et al. [89]	N	-	-	-
Sen and Ganguly [87]	N	-	-	-
Adel et al. [1]	Y	Y	N	N
Nabi and Shpitser [68]	N	-	-	-
Black et al. [11]	Y	Y	Y	N
Ryu et al. [81]	N	-	-	-
Shen et al. [88]	N	-	-	-
Wadsworth et al. [96]	N	-	-	-
Hendricks et al. [38]	N	-	-	-
Blodgett and O'Connor [12]	N	-	-	-
Beutel et al. [8]	Y	Y	N	Y

Algorithmic nondeterminism includes nondeterministic DL layers such as dropout, weight initialization, data augmentation such as random cropping and rotation, and batch ordering [71].

With fixed-seed training, we can completely eliminate algorithmic nondeterminism and have exactly the same training data, dropout behavior, and initial weight.

Implementation-level nondeterminism has two causes [71]:

1. DL software selects primitive operations at runtime. For example, cuBLAS automatically selects computation kernels with respect to buffer availability, and DL frameworks such as PyTorch choose the fastest CUDA kernel based on benchmark results before every training run.

2. Float-point calculations are not associative. For faster training and pre-processing, data is handled by the DL system in parallel processes, and such processes finish the local task at a different speed, resulting in different orders of data fed into the network. Neural network training on GPUs is also highly parallelized. Thread scheduling results in nondeterministic floating-point reduction orders, causing different results across multiple identical training runs.

E.2 Deterministic Mode of DL Software

There is no easy way to eliminate implementation-level nondeterminism. One direction is to enable the deterministic mode provided by DL libraries such as PyTorch, which avoids certain nondeterministic primitives provided by NVIDIA under the hood. However, implementing a deterministic mode is difficult and may not be possible for all DL functions, since the underlying NVIDIA primitives are closed source and their mechanism remains highly opaque.

Although training with deterministic mode can produce identical results across multiple FIT runs, many common APIs (e.g., 25 APIs in PyTorch) still lack deterministic implementation at the time of writing. For example, "MaxPool3d" and "NLLLoss" do not have a deterministic implementation in PyTorch [23]. Popular networks such as "Seq2Seq" [40] will not be trained if PyTorch is under deterministic mode.

Overhead of deterministic mode: The deterministic mode hurts the training speed. Our experiments show that the PyTorch deterministic mode adds (i) on average 17.8% of training-time overhead on FIT runs for technique S-Base, and (ii) 20.8% overhead training a ResNet101 and 14.1% overhead training a DenseNet121 with FIT runs.