A Model- and Data-Agnostic Debiasing System for Achieving Equalized Odds

Thomas Pinkava, Jack McFarland, Afra Mashhadi

Computing and Software Systems
University of Washington
Bothell, WA, USA
pinkavat@uw.edu, jackmc@uw.edu, mashhadi@uw.edu

Abstract

As reliance on Machine Learning (ML) systems in real-world decision-making processes grows, ensuring these systems are free of bias against sensitive demographic groups is of increasing importance. Existing techniques for automatically debiasing ML models generally require access to either the models' internal architectures, the models' training datasets, or both. In this paper we outline the reasons why such requirements are disadvantageous, and present an alternative novel debiasing system that is both data- and model-agnostic. We implement this system as a Reinforcement Learning Agent and through extensive experiments show that we can debias a variety of target ML model architectures over three benchmark datasets. Our results show performance comparable to data- and/or model-gnostic state-of-the-art debiasers.

Introduction

Machine-Learning (ML) systems are being increasingly used in real-world decision-making processes, such as employment offers, credit evaluations, or predictions of criminal recidivism. Researchers have often found that these decision-making algorithms exhibit bias towards sensitive demographic groups, and have raised arguments in favour of reducing this bias since Machine Learning was far less prevalent than it is today (Bolukbasi et al. 2016; Calmon et al. 2017; Berk et al. 2021).

Achieving fairness in machine learning models has attracted increasing attention in the past decade, with many debiasing technologies being developed. We provide examples below, along with a taxonomy of these systems that reveals two drawbacks, namely, that existing debiasing techniques require either access to the underlying architecture of the model being debiased or to its training data. In real-world applications, however, models and data can be distributed, unwieldy, proprietary, or subject to privacy concerns. We therefore ask the following research questions:

- RQ1: Can we create a debiasing system that is data/model-agnostic?
- RQ2: Can such a debiasing system achieve group fairness comparable to state-of-the-art techniques?

To address these research questions, we propose a novel debiasing technique that is *model* and *data* **agnostic**; that is, our proposed system operates without access to the target model's underlying architecture (i.e., for a *black-box* target) or regard for the data on which said target was trained.

We implement this system as a Reinforcement Learning Agent[1]. To assess the generalizability of our system, we provide empirical results over three benchmark datasets and comparisons with state-of-the-art debiasing algorithms. Our results show that our debiasing technique can achieve bias reduction close to the results of the state-of-the-art data- and model-gnostic algorithms, while remaining data- and model-agnostic.

Background

Existing Debiasing Techniques

Barocas et al. (Barocas, Hardt, and Narayanan 2019) describe three types of existing debiasing techniques, based on where they appear in the Machine Learning pipeline:

- Preprocessing-based techniques operate on the model's input data. Such a system might learn a transformation for said data (Calmon et al. 2017), or might weight (Mehrabi et al. 2021; Kamiran and Calders 2012) or prune (Li et al. 2022; Lum and Johndrow 2016; Kamiran and Calders 2012) features most responsible for bias. Representation Learning, wherein the data's representation is chosen to minimize bias, is also fruitful (Locatello et al. 2019; Zemel et al. 2013; Madras et al. 2018).
- In-training techniques act during the model's training process. Such a system modifies the model's cost-sensitivity (Agarwal et al. 2018; Liao and Naghizadeh 2023) or adds an adversarial model to recover sensitive attributes from predictions (Yang et al. 2023; Zhang, Lemoine, and Mitchell 2018). Fairness-aware hyperparameter optimization (F.Cruz et al. 2021) and AutoML (Nguyen, Biswas, and Rajan 2023) can also reduce bias. If the target model uses latent-space representations or embeddings, debiasers can modify these (Zhang et al. 2023; Zeng et al. 2022; Bolukbasi et al. 2016).
- Post-processing techniques are applied after training, and come in two types. The former type attaches a model

[1] https://github.com/pinkavat/rl_debiaser

to the target's output, which modifies the model's predictions to reduce bias (Alabdulmohsin and Lucic 2022; Kamiran, Karim, and Zhang 2012; Wei, Ramamurthy, and Calmon 2021). The latter type modifies the parameters of the target model itself (Woodworth et al. 2017; Fish, Kun, and Lelkes 2016).

Drawbacks of Existing Techniques

The above three classes of debiasing techniques each entail certain drawbacks.

Pre-process approaches require forethought; as they must be applied before the model is trained, fairness must be a consideration from the beginning of the modeling process. As a corollary, if the model is later found to be biased, to recover fairness using pre-process techniques would require a complete retraining of the model. Additionally, the model cannot be so debiased without access to the training data, which may be restricted due to security, ownership, privacy, or resource-management concerns.

In-process techniques likewise require the consideration of the training party, and the modifications to the training process, unless well-defined, well-publicized, and generalizable, are likely to be practically inapplicable, unfamiliar to the trainer, and/or expensive. Debiasing a trained model with in-process techniques also entails retraining.

Post-process techniques place the least burden on the training party, and are therefore suitable for post-factum application to models found to exhibit bias. However, they too have drawbacks: techniques that modify the model's architecture require a white-box target, which may not be possible if the model is proprietary, secret, or otherwise inaccessible. Post-processing the model's output incurs a complexity cost, as the post-processor is an additional step in the model's inference process. Such output transformations, being unaware of data and model, are also susceptible to poor performance (Woodworth et al. 2017).

Desiderata for a New System

The shortcomings discussed above reveal a hitherto-unfilled niche in machine debiasing, which we can circumscribe with the following desiderata:

1. *Data-Agnosticism*: To avoid the need to access data that may be private, secret, proprietary, etc., the debiasing system should not require access to the target model's training data. Ideally, no provision should need to be made for what the target's featurespace actually corresponds to.
2. *Model-Agnosticism*: To avoid the need to perform expensive, complex, or impossible modifications to the target model, the system should not be able to read from or write to the target model's structure, weights, architecture, etc. By extension this implies that the system need not be concerned with said structure, weights, architecture, etc.
3. *Applicable After Training*: To avoid the need for a fairness-aware training party, the system should be usable on targets that have already been trained.
4. *No complexity increase*: We do not wish to increase model size or execution time, or to add unwieldy components; we may also wish to debias models that are lightweight. We therefore want a system that adds no components to the target model.
5. *Single-pass post-train debiasing*: We do not wish to burden the end user of the model with the responsibility of keeping it unbiased. The debiasing system must therefore be a task that, once accomplished, produces a debiased model, and not an ongoing process.

Measuring Fairness

Group fairness in ML is defined by the relationship between predictions and sensitive demographic attributes. Debiasing a model automatically, therefore, entails minimizing some metric derived from the relationship between the model's input, the sensitive attributes therein, and the model's output. Fairness research employs a great variety of these (Tang, Zhang, and Zhang 2023). As our work is primarily concerned with prototyping a novel system we will not be delving into the active debate surrounding the relative merits of the panoply of available metrics.

The metric we chose is Equalized Odds (Hardt, Price, and Srebro 2016), which is satisfied if the predictor's output is independent of the sensitive attribute(s), but still conditional on the ground truth:

$$\forall a \in A,\ \ y, \hat{y} \in Y: \quad P_{\hat{Y}|AY}(\hat{y}|a, y) = P_{\hat{Y}|Y}(\hat{y}|y) \quad (1)$$

Where A is the sensitive attribute, Y is the ground truth label, and $\hat{Y}$ is the model's predicted label.

Since we are training a machine learning model to optimize for bias reduction, it is necessary to convert the Equalized Odds metric into a scalar loss, the *Equalized Odds Violation* (**EOV**). In the binary classification case, we compute this by taking the difference between the True Positive Rate (**TPR**) and False Positive Rate (**FPR**) of the sensitive and nonsensitive groups, and selecting the maximum (i.e. worst) violation:

$$\begin{aligned} &\max_y \left| P(\hat{Y} = 1|A = a, Y = y) - P(\hat{Y} = 1|A = b, Y = y) \right| \\ &= \max(|\mathbf{TPR}_a - \mathbf{TPR}_b|, \quad |\mathbf{FPR}_a - \mathbf{FPR}_b|) \end{aligned} \quad (2)$$

The Equalized Odds Violation will be in the range $[0.0, 1.0]$, where the closer a model scores to zero, the less biased (i.e., fairer) we consider it to be.

Noise-Injection Debiasing

The aforementioned desiderata provide the mode by which our system must debias its targets. Since we cannot control the target's parameters directly, and must deploy our system only after the target has been trained, we are limited to providing the target with data and observing its output. We modify the target's structure by providing it with new training data that has been synthesized for the purpose.

This interaction mode has already been explored to some degree; existing work has used data adversarially generated to maximize the target's exposure of bias as a means of measurement (Xiao et al. 2023), and as a means of debiasing the specific domain of face-attribute classification (Zeng et al. 2022). We aim to go further, to produce the data required to debias any target on-the-fly.

The simplest form such a system might take would be feedback-free where the biased target data drawn from a random distribution is fed into the target model without monitoring its output and impact of the data on its fairness. We tested such a system by drawing synthetic training data from a parameterized Gaussian distribution and feeding it into the target, without responding to the target's output at all. Our results are included below. Such a system, however, is not capable of learning, and does not make use of all the information available; we thus expect it not to perform as well as one that was capable of learning.

A learning system would observe the target model's output and predict its reaction to different synthetic data, and thus tailor the perturbed data to maximize bias reduction. We describe this novel system next.

Debiasing with an RL Agent

To generate synthetic training data that will effectively debias any target, we require a learning system. Such a system is inherently an ML model whose training process includes the training process of another ML model (the target). As a result, we encountered practical difficulties when attempting to implement a direct approach (in which the synthesizer generates data, the data is fed to the target, and the bias reduction thereby obtained is backpropagated through both models) in the TensorFlow and PyTorch ML libraries. We therefore require an alternative ML paradigm that permits 'air-gapping' between the training processes of the target and the debiaser.

One possible such paradigm is Reinforcement Learning (**RL**), in which an *agent* (the model) makes *observations* about an *environment*, then generates *actions*, which are given to the environment, producing *rewards*. The agent attempts to maximize these rewards by adjusting its actions, and thus learns through a trial-and-error process (Kaelbling, Littman, and Moore 1996). RL is useful to our work for a simple reason: it does not concern itself with the nature of the environment, only with its input and output; we can therefore 'hide' our target model's training process inside the environment. Our debiaser model becomes an Agent, the synthetic training data it produces become analogous to actions, and the measurement of the target's bias after training on that data is the reward. The observations are measures of the target's current bias and the difference in bias from its previous state incurred by the agent's previous action.

RL models come in many varieties; the type we chose as ensuring a sufficient airgap between target and debiaser was an Actor-Critic architecture (Sutton et al. 1999). In this setup, the Agent is divided into two ML models. The first, the *Actor*, models an *action policy*: a mapping from observations to actions, which provides the mechanism for the Agent's decision-making. The second, the *Critic*, models the *Q-function*: a mapping from observation/action pairs to the expected reward for taking said action given said observation.

In order to generate synthetic data items (that is, tensors of floats) from a bias measure, we need an agent capable of operating in a continuous observation and action space. We draw our groundwork and practical architecture from the work of Lillicrap et al. (Lillicrap et al. 2019), who extend Deterministic Policy Gradient learning (Silver et al. 2014) into the continuous action space. Their work contains three refinements common in RL Agents, which we added to our own code, as both stabilize the Agent's training process:

1. The *Replay Buffer*: The Actor and Critic are trained on observation/action/reward/next observation tuples, so instead of generating these individually, passing them through the models individually, and running the model's optimizers, the tuples are placed in a FIFO queue of fixed size. At each training step, a sample of several tuples is drawn from the queue at random, and the Actor and Critic are trained thereupon. The buffer also provides a larger air-gap between the Target and Agent training processes.
2. *Secondary Actor and Critic*: To mitigate the volatility of the estimated future state, the future-reward prediction is performed by a second Actor and Critic, who are not trained, but whose weights are soft-copied (their weights and the weights of the main Actor and Critic are mixed in some hyperparametric proportion) at each training step.
3. *Stochastic Exploration*: The Actor-Critic Agent alone is entirely deterministic; what variation it experiences in its exploration is due entirely to its starting weights. This is not desirable – the Agent should be capable of exploration – so we implement exploration thus: once the Actor has chosen an action from an observation, we add a Gaussian noise sample to each feature of the synthetic datum. The result is stored as the action, as if the Actor had created it. The Gaussian distribution was used because it is simple and versatile; future work might investigate alternative noise distributions (Lillicrap et al. use an Ornstein-Uhlenbeck process). The distribution has a mean of zero and a hyperparametrized standard deviation.

Algorithm 1 shows the debiasing process; Figure 1 shows the debiaser's datapath. Note that the datapath includes the target dataset as part of the EOV-measuring mechanism; the system as a whole is not fully data-agnostic, though the debiaser agent is, As we will discuss below, true data-agnosticity can be achieved if the target system exposes its own bias measure.

Experimental Method

Target Model Architecture

To substantiate the claim that the system is *model-agnostic*, we trained four different ML models to act as biased targets: one Logistic Regression classifier and three Multilayer Perceptrons. The perceptrons were given varying architectures: the 'normal' configuration had one hidden layer of 50 neurons, the 'wide' configuration had one hidden layer of 100

Algorithm 1: RL Debiasing Procedure (adapted from Lillicrap et al. (Lillicrap et al. 2019))

procedure DEBIAS(T, E, S, N, ρ, σ, γ, τ)
Create critic networks $Q(b, \Delta b, a|\theta^Q)$ and Q', and actor networks $\mu(b, \Delta b|\theta^\mu)$ and μ'.
Randomly initialize weights θ^Q and θ^μ.
Copy to secondary networks $\theta^{Q'} \leftarrow \theta^Q, \theta^{\mu'} \leftarrow \theta^\mu$.
Initialize a ring buffer R of length ρ.
for episode = 1 to E **do**
Reset target model T to its initial state.
Clear ring buffer R.
Measure initial bias b_1 of T; $\Delta b_1 \leftarrow 0$
for step i=1 to S **do**
Action $a_i = \mu(b_i, \Delta b_i|\theta^\mu) + \mathcal{N}(0, \sigma)$.
Train target model T on a_i.
Measure bias b'_i of T; $\Delta b'_i \leftarrow (b'_i - b_i)$.
Store $(b_i, \Delta b_i, a_i, b'_i, \Delta b'_i)$ in buffer R.
Sample a batch of N items $(b_n, \Delta b_n, a_n, b'_n, \Delta b'_n)$ from buffer R.
Set $y_n = b'_n + \gamma Q'(b'_n, \Delta b'_n, \mu'(b'_n, \Delta b'_n))$ for $n \in N$
Update critic by minimizing the loss $L = \frac{1}{N}\sum_n (y_n - Q(b_n, \Delta b_n, a_n))^2$
Update the actor policy using the sampled policy gradient:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N} \sum_n \nabla_a Q(b_n, \Delta b_n, a|\theta^Q)|_{a=\mu(b_n, \Delta b_n)}$$
$$\nabla_{\theta^\mu} \mu(b_n, \Delta b_n|\theta^\mu)|_{b_n, \Delta b_n}$$

Copy weights to the secondary networks:

$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$$
$$\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$$

end for
end for
end procedure

neurons, and the 'deep' configuration had two hidden layers of 50 neurons apiece. Each model had a Sigmoid output layer to perform binary classification, and each used a PyTorch library Adam optimizer with a learning rate of 10^{-5}. Each model was given between 10 and 30 epochs (depending on accuracy convergence) of pre-training on the target dataset before being fed into the debiasing system. This pre-trained state is given as the *initial state* in the results section, with accuracy comparable to the baselines listed on the UC Irvine Machine Learning Repository (Kelly, Longjohn, and Nottingham 1994).

Datasets

To demonstrate *data-agnosticism*, we trained our target models on three standard fairness-research demographic tabular datasets, in order to have state-of-the-art benchmarks for comparison:

- The Adult dataset (Becker and Kohavi 1996), a set of cleaned US census data from 1996, where the task is to predict whether a given individual has an income above a certain threshold. The sensitive attribute used here was sex, to align with the benchmarks used.
- The German Credit dataset (Hofmann 1994), demographic data from German loan applicants. The task here classifies individuals into 'good' or 'bad' credit risks. The sensitive attribute used was sex, to align with the benchmarks.
- The COMPAS dataset (Angwin et al. 2013), a set of demographic data from Florida defendants. The task here is to predict whether a given defendant will reoffend in two years' time. For the sensitive attribute we used race (which we binarized as African-American vs. Non-African-American), as the sex attribute was highly skewed.

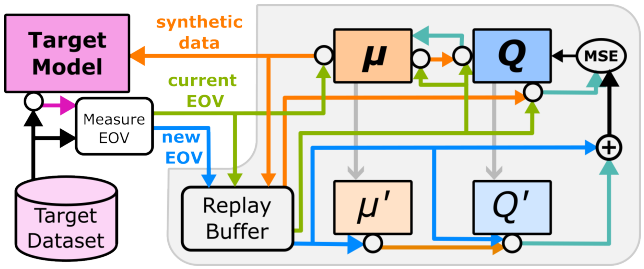

Figure 1: System Datapath Outline. Bubbles attached to each network denote a forward pass. Q and μ are the Critic and Actor networks, respectively, and Q' and μ' are the soft-copied secondary networks used in discounted future-state prediction.

Hyperparameter Optimization

To find performant values for the hyperparameters (Given in the parameter list of Algorithm 1), we trained a normal MLP target on a subset of the Adult dataset and employed a heuristic-guided Monte Carlo search process. This yielded a step count S of 100000, a sample size N of 1024, a memory size ρ of 4096, an exploration standard deviation σ of 8, a lookahead factor γ of 0.99, and a softcopy factor τ of 0.001. Actor and Critic MLPs were constructed from a 400-neuron hidden layer followed by a 300-neuron hidden layer, with Pytorch library Adam optimizers. The learning rate of the Actor was set at 10^{-7}, while the Critic's was set at 10^{-6}.

We do not believe these hyperparameters are optimal, as the possibility space is large and the interaction between hyperparameters is not well understood. However, as we will demonstrate even with the aforementioned hyperparameters we achieve a high fairness and accuracy (low trade-off) on the benchmarked datasets.

Experimental Procedure

For each pairing of dataset and target architecture, we ran three learning trials. In each trial, the debiasing system was given 4 episodes (hyperparameter E in Algorithm 1) to debias its target. Within the episode, the target's bias was measured using a training subset of the relevant dataset. Before

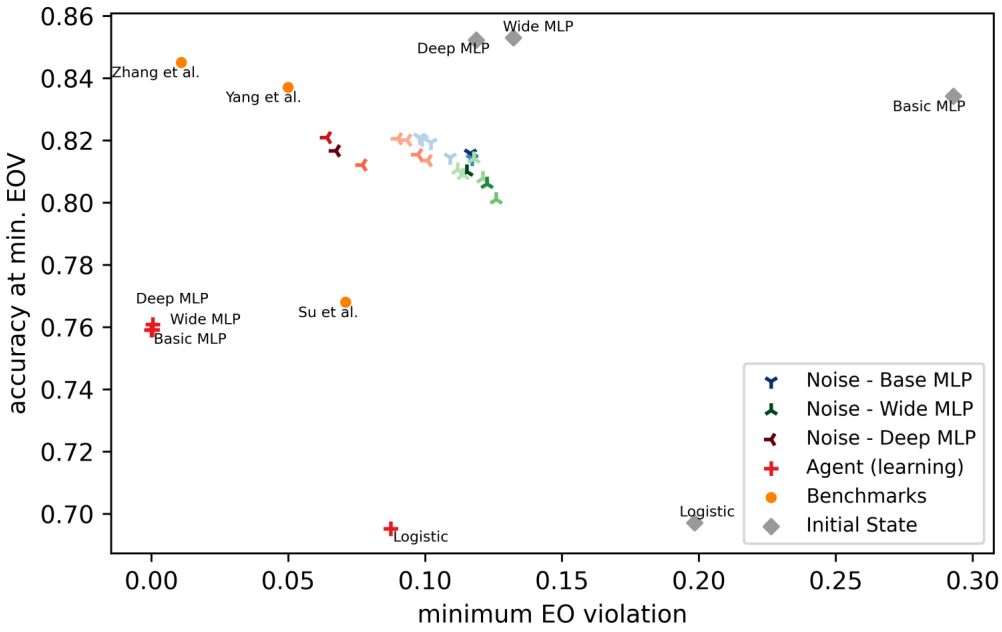

Figure 2: Results for the Adult Task

the process and at the end of each episode, the target's bias was measured using a withheld testing subset. The target's classification accuracy was also measured.

For non-learning noise-injection trials, the same pattern was followed. The standard deviation for the Gaussian noise injected was set at doubling intervals between 0.1 and 8.0, and four episodes of injection were run for each value, for each dataset/target pair. No noise-injection trials were run on the logistic regression target due to resource constraints.

Trials were run on two separate computers equipped with M1-Pro 10-core CPUs with 16 and 32 GB of RAM respectively; as the debiasing process is highly sequential, no advantage is gained from GPU parallelization; indeed, initial trials run on a server with 4 NVIDIA RTX 2080 Ti GPUs proved to be far slower than M1-Pro CPU execution on the laptops. Each episode took on average approximately 40 minutes to execute.

Results

Figures 2-3 show the results obtained from our experiment, giving the minimum (best) EOV achieved and the classification accuracy of the target at that point.

These results suggest that it is indeed possible to automatically debias an ML model without regard to its architecture or its data, and, what is more, to obtain performance approaching that of state-of-the-art model- and data-gnostic debiasers.

For the Adult task (Fig. 2), the learning agent reduces the Equalized Odds Violation below that of the state-of-the-art benchmarks for the perceptron targets. The logistic target's EOV reduction is also close to SOTA techniques. The system loses a similar amount of target accuracy as the work of Su et al., but lags behind the better accuracies of Zhang et al. and Yang et al. The pure noise-injector achieves slight EOV improvements for the Wide and Deep MLP targets, and a major EOV improvement for the Basic MLP target, but does not outperform the learning system, though it achieves higher classification accuracy.

The German task results (Fig. 3) exhibits similar characteristics. That is the learning-agent EOV reduction on all target architectures, including logistic regression, falls in the middle of the SOTA benchmarks. The accuracy loss is likewise good, with accuracy at minimum EOV only slightly below the initial state. The noise-injection failed to achieve performant results.

The COMPAS task results (Fig. 4) are the weakest. The learning agent reduced the EOV to almost zero, but at the cost of unacceptable accuracy loss. For the logistic target, the system had little effect. The feature of note here is that the pure-noise injection achieved similar results to the learning process for the Basic MLP target, and had significantly better classification accuracy for the Wide MLP target.

In all cases, we observe a reduction in accuracy associated with an improvement in EOV. These reductions are in some cases slight, and in some cases major. They stem from a systemic problem know as *fairness-accuracy tradeoff* (Berk et al. 2017; Teodorescu and Yao 2021; Menon and Williamson 2017; Kenfack et al. 2021); that is, a classifier that must make allowances for fairness cannot reliably be as accurate as one that does not. Some debiasing techniques make this tradeoff an explicit, controllable value; others devise entire constrained optimization processes to eke out as much utility as possible (Grgić-Hlača et al. 2018). Our system makes no provision for accuracy at all, rather optimizing only for minimal Equalized Odds Violation. The tar-

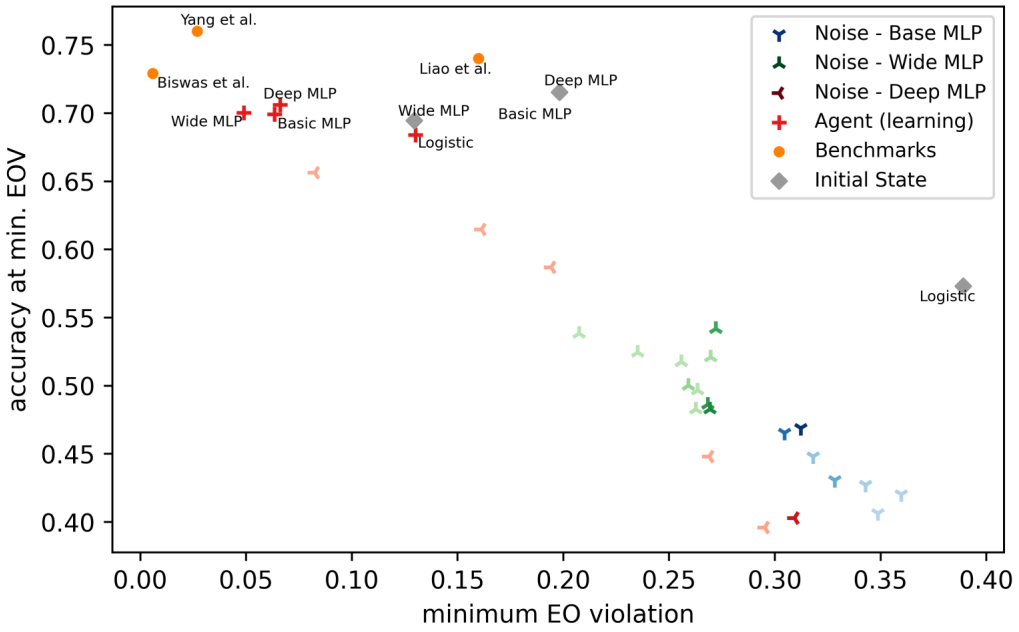

Figure 3: Results for the German Credit Task

gets remain accurate because their (black-box) optimizers are (presumably) optimizing for the target's own accuracy. The tradeoff may also explain why a deliberate improvement in EOV by a learning agent results in a lower accuracy than pure-noise in some cases; as the EOV improves, the accuracy must drop.

Discussion

Implications

A real-world application of our new system can expect to gain the practical advantages previously discussed. Its being model-agnostic means we can apply our technique to any ML model to whom we can pass new training data, and whose bias we can measure, without needing to modify or observe the internal structure of said model. The system's data-agnosticity means we need not care about the dataset involved; we need not tailor our system to suit particular distributions, nor obtain access to data that may be private, secret, or proprietary. The post-training application of our technique means we can debias existing, pre-trained models; we do not have to intervene in the training process, nor trust the training party to have a vested interest in debiasing. Additionally, the system need only be applied once; it adds no complexity to the target, which ensures that the target model's prediction does not become slower or more expensive.

Practical implementation and deployment of our work would mean that already-existing ML models could be debiased after their training completes. As a result, bias reduction could be provided as a service – or enforced as a standard – without the need to distribute responsibility for fairness among ML model builders. The in-process benchmarks against which we compare, however, still retain higher accuracies than our own system. Without knowing to what degree our work may be extended we are as yet unwilling to suggest that our system constitutes a replacement for considered in-training bias mitigation in critical decision-making systems.

Our work also has a broader theoretical implication for the use of RL Agents in ML. We modified a measurable property (fairness) of a target model using only synthetic training data. It may be that our system could modify other properties of an ML model as well; all that is needful is a means to measure the property in question.

Theoretical Limitations

Our system as a whole falls slightly short of total data-agnosticity. We cannot evade the need for some means of measuring the target model's bias; since that bias depends on the model's classification of data with respect to said data's sensitive attributes, it cannot be computed without some access to that data. Our system used two subsets (training and testing) of each target's dataset to perform its bias calculations, and we cannot therefore claim that it is truly data-agnostic.

However, the algorithmic core of the system – the Reinforcement-Learning Agent – is data-agnostic; it requires only the data's shape and the bias measurements. If the bias measurements were provided by the target model through some opaque mechanism, the system would attain true data-agnosticity. Such a mechanism would of course also require data-access, but, if the measurement were done target-side, would address some of the reasons we wish the debiaser to be data-agnostic in the first place, namely, that the dataset might be secret, sensitive, or unwieldy.

Another theoretical problem concerns model-agnosticity.

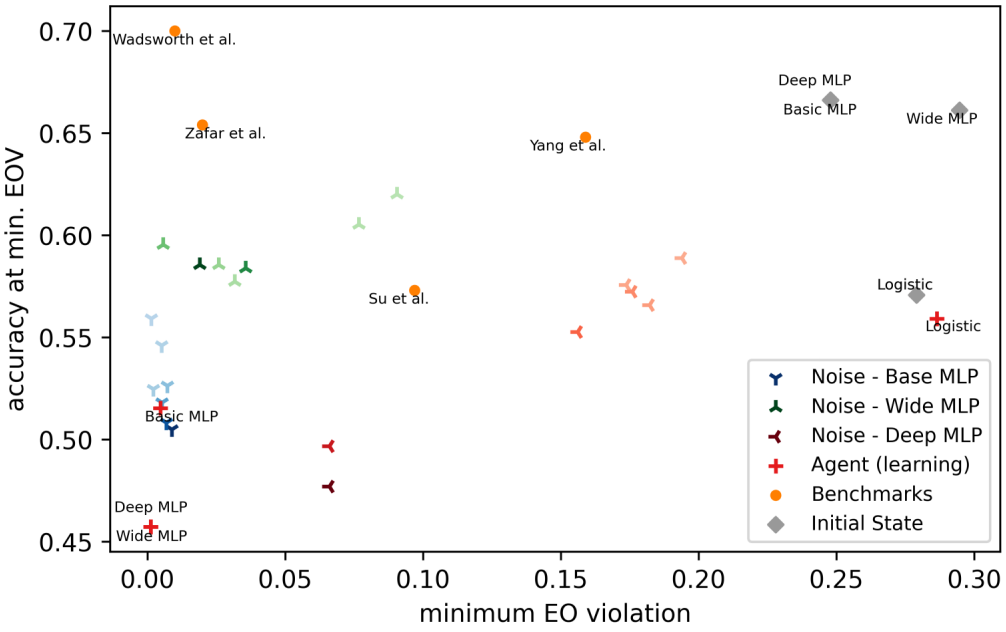

Figure 4: Results for the COMPAS Task

Our experiments were performed only on ML models that could accept further training data after being trained; this is not true of all ML models (for instance, Decision Trees). Our system might be able to debias these models also, through a baroque relaxation of the data-agnosticity requirement wherein the target is entirely retrained at each step with the synthetic debiasing data appended to its dataset. Such a procedure could potentially be expensive.

Practical Difficulties

One main practical problem with debiasing using our system is the drop in target accuracy incurred by the fairness-accuracy tradeoff (discussed above). While the tradeoff cannot be entirely avoided, it can be finessed in several ways; for instance, an in-process system with access to the loss function can control the relative cost weighting of accuracy and fairness with a high degree of precision. Our system, without such fine control, may be more subject to the depredations of the tradeoff.

Another practical difficulty encountered was execution time. Our system is necessarily sequential, as the target's reaction to any given action depends on its state which is in turn directly modified by said action. The debiasing Agent can therefore only furnish the target with one synthetic datum at a time. ML libraries are written with parallelization as a goal, the understanding being that large batches of data can be drawn and passed through models simultaneously. As our system cannot do this, we cannot take advantage of GPU execution (indeed, bussing the data back and forth seems to incur a significant time penalty) and must run our system entirely on the CPU. The debiasing process is therefore slow, and this is likely to be exacerbated by large, complex target models, though we did not test any such.

Future Work

The viability of the prototype opens many avenues for future investigation. In addition to expanded target- and dataset- (and data modality) compatibility testing, it would be worthwhile to determine if the Agent could be re-used: synthetic data created to debias one model trained on a dataset might also debias a different model trained on the same data.

The prototype would also undoubtedly benefit from more concerted hyperparameter optimization (certain hyperparameters, such as the Agent's memory size and Actor and Critic internal architectures, were barely touched during the search process). Architectural overhauls may also prove a fruitful pursuit; as it stands, the Agent learns to generate synthetic data directly, which is then perturbed by its noise generator; what if the Agent were instead, say, to learn per-feature parameters for a generative noise distribution?

Another mechanism to add might be deliberate consideration for the fairness-accuracy tradeoff as discussed above. Additionally, we made a strong assumption by creating experimental target models which optimized themselves for accuracy only. We may wish to apply our system to models that are optimizing for other desirable properties (say, privacy).

Potential alternative applications of our general system might also be considered. In a Federated Learning setting, for instance, the Agent might be inserted as an 'impostor client'. It would receive the aggregate model from the FL server, measure the aggregate's fairness using its own data supply (with accuracy limited, of course, by whether the client's data is a representative sample) and then generate an entirely synthetic client model, with ersatz weights, to be sent back to the server to reduce the server's bias. Such a

system might have certain drawbacks to overcome: the feedback loop would perhaps be infeasibly slow (which opens another avenue for research, that of extracting information about the server's bias from as little feedback as possible), and the server might flag the synthetics as outliers to be culled (which leads into the field of *data poisoning*).

Acknowledgments

This work was supported by generous support from the US National Science Foundation (NSF) award **IIS-2304213**.

References

Agarwal, A.; Beygelzimer, A.; Dudík, M.; Langford, J.; and Wallach, H. 2018. A Reductions Approach to Fair Classification. ArXiv:1803.02453 [cs].

Alabdulmohsin, I.; and Lucic, M. 2022. A Near-Optimal Algorithm for Debiasing Trained Machine Learning Models. ArXiv:2106.12887 [cs, stat].

Angwin, J.; Larson, J.; Mattu, S.; and Kirchner, L. 2013. How We Analyzed the COMPAS Recidivism Algorithm. Https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm.

Barocas, S.; Hardt, M.; and Narayanan, A. 2019. *Fairness and Machine Learning: Limitations and Opportunities*. fairmlbook.org.

Becker, B.; and Kohavi, R. 1996. Adult. Published: UCI Machine Learning Repository.

Berk, R.; Heidari, H.; Jabbari, S.; Kearns, M.; and Roth, A. 2017. Fairness in Criminal Justice Risk Assessments: The State of the Art. ArXiv:1703.09207 [stat].

Berk, R.; Heidari, H.; Jabbari, S.; Kearns, M.; and Roth, A. 2021. Fairness in criminal justice risk assessments: The state of the art. *Sociological Methods & Research*, 50(1): 3–44.

Bolukbasi, T.; Chang, K.-W.; Zou, J. Y.; Saligrama, V.; and Kalai, A. T. 2016. Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Calmon, F.; Wei, D.; Vinzamuri, B.; Natesan Ramamurthy, K.; and Varshney, K. R. 2017. Optimized Pre-Processing for Discrimination Prevention. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

F.Cruz, A.; Saleiro, P.; Belém, C.; Soares, C.; and Bizarro, P. 2021. Promoting Fairness through Hyperparameter Optimization. In *2021 IEEE International Conference on Data Mining (ICDM)*, 1036–1041. ISSN: 2374-8486.

Fish, B.; Kun, J.; and Lelkes, A. D. 2016. A Confidence-Based Approach for Balancing Fairness and Accuracy. ArXiv:1601.05764 [cs].

Grgić-Hlača, N.; Zafar, M. B.; Gummadi, K. P.; and Weller, A. 2018. Beyond Distributive Fairness in Algorithmic Decision Making: Feature Selection for Procedurally Fair Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1). Number: 1.

Hardt, M.; Price, E.; and Srebro, N. 2016. Equality of Opportunity in Supervised Learning. ArXiv:1610.02413 [cs].

Hofmann, H. 1994. Statlog (German Credit Data). Published: UCI Machine Learning Repository. https://archive.ics.uci.edu/dataset/144.

Kaelbling, L. P.; Littman, M. L.; and Moore, A. W. 1996. Reinforcement Learning: A Survey. ArXiv:cs/9605103.

Kamiran, F.; and Calders, T. 2012. Data preprocessing techniques for classification without discrimination. *Knowledge and Information Systems*, 33(1): 1–33.

Kamiran, F.; Karim, A.; and Zhang, X. 2012. Decision Theory for Discrimination-Aware Classification. In *2012 IEEE 12th International Conference on Data Mining*, 924–929. ISSN: 2374-8486.

Kelly, M.; Longjohn, R.; and Nottingham, K. 1994. The UCI Machine Learning Repository. Https://archive.ics.uci.edu.

Kenfack, P. J.; Khan, A. M.; Kazmi, S. A.; Hussain, R.; Oracevic, A.; and Khattak, A. M. 2021. Impact of Model Ensemble On the Fairness of Classifiers in Machine Learning. In *2021 International Conference on Applied Artificial Intelligence (ICAPAI)*, 1–6.

Li, Y.; Meng, L.; Chen, L.; Yu, L.; Wu, D.; Zhou, Y.; and Xu, B. 2022. Training Data Debugging for the Fairness of Machine Learning Software. In *2022 IEEE/ACM 44th International Conference on Software Engineering (ICSE)*, 2215–2227. ISSN: 1558-1225.

Liao, Y.; and Naghizadeh, P. 2023. Social Bias Meets Data Bias: The Impacts of Labeling and Measurement Errors on Fairness Criteria. ArXiv:2206.00137 [cs].

Lillicrap, T. P.; Hunt, J. J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; and Wierstra, D. 2019. Continuous control with deep reinforcement learning. ArXiv:1509.02971 [cs, stat].

Locatello, F.; Abbati, G.; Rainforth, T.; Bauer, S.; Schölkopf, B.; and Bachem, O. 2019. On the Fairness of Disentangled Representations. ArXiv:1905.13662 [cs, stat].

Lum, K.; and Johndrow, J. 2016. A statistical framework for fair predictive algorithms. ArXiv:1610.08077 [cs, stat].

Madras, D.; Creager, E.; Pitassi, T.; and Zemel, R. 2018. Learning Adversarially Fair and Transferable Representations. ArXiv:1802.06309 [cs, stat].

Mehrabi, N.; Gupta, U.; Morstatter, F.; Steeg, G. V.; and Galstyan, A. 2021. Attributing Fair Decisions with Attention Interventions. ArXiv:2109.03952 [cs].

Menon, A. K.; and Williamson, R. C. 2017. The cost of fairness in classification. ArXiv:1705.09055 [cs].

Nguyen, G.; Biswas, S.; and Rajan, H. 2023. Fix Fairness, Don't Ruin Accuracy: Performance Aware Fairness Repair using AutoML. ArXiv:2306.09297 [cs].

Silver, D.; Lever, G.; Heess, N.; Degris, T.; Wierstra, D.; and Riedmiller, M. 2014. Deterministic Policy Gradient Algorithms. In *Proceedings of the 31st International Conference on Machine Learning*, 387–395. PMLR. ISSN: 1938-7228.

Sutton, R. S.; McAllester, D.; Singh, S.; and Mansour, Y. 1999. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *Advances in Neural Information Processing Systems*, volume 12. MIT Press.

Tang, Z.; Zhang, J.; and Zhang, K. 2023. What-Is and How-To for Fairness in Machine Learning: A Survey, Reflection, and Perspective. *ACM Computing Surveys*. Just Accepted.

Teodorescu, M. H.; and Yao, X. 2021. Machine Learning Fairness is Computationally Difficult and Algorithmically Unsatisfactorily Solved. In *2021 IEEE High Performance Extreme Computing Conference (HPEC)*, 1–8. ISSN: 2643-1971.

Wei, D.; Ramamurthy, K. N.; and Calmon, F. d. P. 2021. Optimized Score Transformation for Consistent Fair Classification. ArXiv:1906.00066 [cs, math, stat].

Woodworth, B.; Gunasekar, S.; Ohannessian, M. I.; and Srebro, N. 2017. Learning Non-Discriminatory Predictors. ArXiv:1702.06081 [cs].

Xiao, Y.; Liu, A.; Li, T.; and Liu, X. 2023. Latent Imitator: Generating Natural Individual Discriminatory Instances for Black-Box Fairness Testing. ArXiv:2305.11602 [cs].

Yang, J.; Soltan, A. A. S.; Eyre, D. W.; Yang, Y.; and Clifton, D. A. 2023. An adversarial training framework for mitigating algorithmic biases in clinical machine learning. *npj Digital Medicine*, 6(1): 1–10. Number: 1 Publisher: Nature Publishing Group.

Zemel, R.; Wu, Y.; Swersky, K.; Pitassi, T.; and Dwork, C. 2013. Learning Fair Representations. In *Proceedings of the 30th International Conference on Machine Learning*, 325–333. PMLR. ISSN: 1938-7228.

Zeng, H.; Yue, Z.; Shang, L.; Zhang, Y.; and Wang, D. 2022. Boosting Demographic Fairness of Face Attribute Classifiers via Latent Adversarial Representations. In *2022 IEEE International Conference on Big Data (Big Data)*, 1588–1593.

Zhang, B. H.; Lemoine, B.; and Mitchell, M. 2018. Mitigating Unwanted Biases with Adversarial Learning. ArXiv:1801.07593 [cs].

Zhang, J.; Wang, L.; Su, D.; Huang, Y.; Cao, C. C.; and Chen, L. 2023. Model Debiasing via Gradient-based Explanation on Representation. ArXiv:2305.12178 [cs].