Balancing the Scales: Reinforcement Learning for Fair Classification

Leon Eshuijs[1], **Shihan Wang**[2], **Antske Fokkens**[1]
Vrije Universiteit Amsterdam[1], Utrecht University[2]
l.eshuijs@vu.nl, s.wang2@uu.nl, antske.fokkens@vu.nl

Abstract

Fairness in classification tasks has traditionally focused on bias removal from neural representations, but recent approaches have shifted towards algorithmic methods that embed fairness into the training process. These methods steer models towards fair performance, preventing potential elimination of valuable information that arises from representation manipulation. Reinforcement Learning (RL), with its ability to learn through interaction and adjust reward functions to encourage desired behaviors, presents a promising approach in this domain. In this paper, we conduct an exploratory evaluation of RL for addressing bias in imbalanced classification by scaling the reward function. We employ the contextual multi-armed bandit framework, adapt three popular RL algorithms, and conduct an extensive empirical evaluation of their relative strengths and limitations. Through this analysis, we contribute meaningful evidence to the ongoing debate between algorithmic and representational fairness approaches.[1]

Keywords: gender bias, reinforcement learning

1. Introduction

Issues of bias and fairness in Natural Language Processing have emerged as critical research priorities (Mehrabi et al., 2021). In classification algorithms, bias often stems directly from the training data leading to unfair outcomes between protected groups such as gender or race. To address this problem, previous work on fairness has focused on achieving *representational fairness*, so that the information of the protected groups is lost (Ravfogel et al., 2020; Haghighatkhah et al., 2022). However, recent work has demonstrated no meaningful correlation between representational fairness and *empirical fairness*, i.e. fairness on downstream tasks (Shen et al., 2022). To address empirical fairness directly, other work has explored the intersection of bias mitigation and class-imbalanced learning (Subramanian et al., 2021). Class-imbalanced learning approaches aim to achieve fair performance by balancing the training data via sampling or reweighing the loss function.

Various algorithmic approaches have been explored for addressing fairness in NLP tasks, including both traditional supervised learning methods and Reinforcement Learning (RL) frameworks. In NLP, Reinforcement Learning (RL) has already successfully been applied to various tasks, including syntactic parsing, conversational systems, and machine translation (Uc-Cetina et al., 2023). With regard to classification, a key distinction between the algorithms is that supervised learning is trained on binary labels, but RL is trained directly on the continuous value of each input, as illustrated in Figure 1. This makes reward shaping a natural mechanism for encoding fairness objectives, while RL's exploration strategies (e.g., UCB confidence bounds and ϵ-greedy strategies) can additionally encourage attention to underrepresented subgroups. In the context of classification, RL has been adapted to mitigate class imbalance by modifying the reward function for binary classification (Lin et al., 2020). However, implementations considering more complex imbalances have remained largely unexplored.

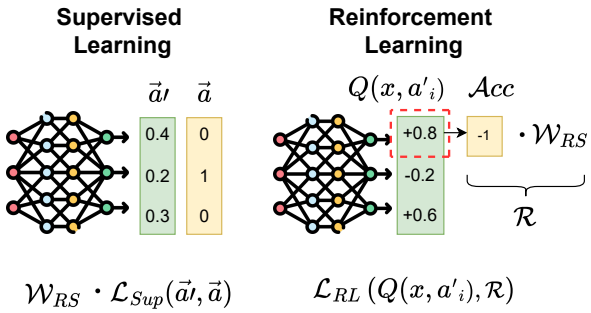

Figure 1: Overview of the classification setup with input vector x, and output class a for Reinforcement Learning and Supervised Learning, highlighting the place of the reward scaling matrix $\mathcal{W}_{RS}$

In this work, we conduct an exploratory evaluation of various algorithmic approaches that employ scaling mechanisms to address fairness among protected groups in text classification. We frame the fair classification task as a Contextual Multi-Armed Bandit (CMAB) problem. To mitigate bias, we scale the reward function to counteract imbalances among protected groups within each class. We employ three different types of RL methods,

[1]Our code is available at https://github.com/watermeleon/RL_for_imbalanced_classification

Proceedings of Identity-Aware AI (IAAI) @ LREC 2026, pages 32–46
16 May 2026.

each reflecting a key type of RL approach, and adapt them to our task, alongside supervised learning for comparison. Through extensive experiments, we investigate how these algorithms perform in terms of fairness and classification accuracy, conducting detailed ablation studies to examine their sensitivity to different reward scaling methods and data imbalances.

Experiments on two fair classification datasets demonstrate that our RL algorithms offer interesting trade-offs compared to existing baselines and that reward scaling provides a flexible tool to mitigate bias in classification. We systematically investigate how stable these approaches are under various class and subclass imbalances as well as various degrees of *representational fairness*. Our research makes the following contributions to advancing fairness in NLP:

1. We develop a framework to use various Reinforcement Learning (RL) techniques for fair classification, with a specific focus on addressing imbalances in protected groups.

2. We provide a systematic evaluation of diverse RL algorithms and reward scaling methods on textual datasets, including comprehensive ablation studies that analyze their behavior under varying class imbalances and scaling strategies. We find further evidence that algorithmic decision choices offer more substantial fairness gains than measures that tackle representational fairness.

3. Our findings reveal that LinUCB achieves strong results on binary datasets with minimal training (less than 2 epochs), while our MDP-derived algorithms perform better in the multi-class setting. Our scaled supervised implementation surpasses existing fairness methods for multi-class datasets.

2. Related Work

Bias Mitigation in NLP Research on mitigating bias can be divided into those that tackle the training data (Wang et al., 2019), those that attempt to remove bias from representations (Ravfogel et al., 2020; Haghighatkhah et al., 2022), and those that adjust the learning process (Elazar and Goldberg, 2018; Han et al., 2021). Within approaches that adjust the learning processes, we distinguish two main categories: those that add adversarial learners to ignore protected attributes (Wadsworth et al., 2018), and more closely to our work, approaches that adjust the loss function to emphasize performance on minority classes.

Prior work that modified the training setup to increase fairness used methods such as down/upsampling (Wang et al., 2019) and reweighting the loss function (Höfler et al., 2005; Lahoti et al., 2020). Han et al. (2022a) evaluate both down-sampling and loss reweighting on two datasets for fair text classification. Both techniques are applied to align training with different definitions of fairness. Downsampling using the Equal Opportunity fairness metric demonstrated impressive results. In this paper, we take the first step to explore whether reward scaling in reinforcement learning can improve fairness in classification.

Markov Decision Process (MDP) Early work by Wiering et al. (2011) casts classification as a sequential decision-making task, by introducing a classification variant of the MDP. In their setup agents manipulate memory cells to encode information by applying an action sequence on a single sample. They demonstrated competitive performance, but this remained limited to small tasks due to the computational complexity. Lin et al. (2020) extended this work, by introducing a variant of the classification MDP and applying a Deep Q-learning Network (DQN) to binary classification of images and texts. They focused on mitigating bias arising from class imbalance by scaling the rewards inversely proportional to the class frequency. However, in their setup the sequential component was taken over multiple data points, which assumes sequential dependency among data samples in the classification task.

Contextual Multi-Armed Bandit (CMAB) The RL framework CMAB offers a promising alternative because it considers the input as a sequence of independent states. We formalize our classification task as a CMAB problem, because this is consistent with the independence of data points in the commonly shuffled datasets. Dudík et al. (2014) use CMAB agents by modifying K-class classification as a K-armed bandit problem, where the agent receives a reward of 1 for correct and 0 for incorrect classification. Dimakopoulou et al. (2019) use this framework and modify different CMAB algorithms to balance exploration and exploitation and compare the original and modified agents on 300 classification datasets. However, their analysis focused on datasets with either limited classes, features, or observations. To the best of our knowledge, we are the first to extend reward scaling for fair multi-class classification or to apply reward scaling for classification with CMAB.

3. Methodology

In this section, we describe how we formalize our classification task as a CMAB. We introduce three RL methods and explain how we adapt them for fair classification.[1]

[1] Here we focus on the key idea of the algorithms and how we adapt them in the paper. More details can be

3.1. Contextual Multi-Armed Bandit

We formalize the multi-class classification task as a finite contextual multi-armed bandit (CMAB) problem. In each round t, an agent is presented with a context vector $x_t \in \mathbb{R}^d$. The agent chooses an action $a_t \in A$ from a fixed set of arms, based on the policy $a_t \sim \pi(x_t)$. After the action is taken, the environment returns a reward: $r_t \sim \mathcal{R}$. In a multi-class classification framework, the action space is the set of all possible classes, while the context vector is a representation of the input, e.g. a contextual text embedding (see Section 4.1 for more information). Within a finite number of rounds, the agent aims to learn the optimal policy to maximize the total reward. In other words, given a set of testing data, we aim to learn the optimal policy to maximize the selection of correct classes.

We extend the CMAB framework for fair classification by constructing a reward function that counters data imbalances. We assign a reward scale for each sensitive state (a, g), comprising the desired class a (e.g. occupation) and protected attribute g (e.g. gender). The total reward for a given prediction is calculated as $\mathcal{R}(a, a_{pred}, g) = \mathcal{A}cc(a, a_{pred}) \cdot \mathcal{W}(a, g)$. It comprises an accuracy term $\mathcal{A}cc$, and a reward scale matrix $\mathcal{W}$. Unlike previous work (Dudík et al., 2014), which defines the accuracy term as $\mathcal{A}cc \in \{0, 1\}$, we define it as $\mathcal{A}cc \in \{-1, 1\}$. This allows us to scale the reward for both correct (+1) and incorrect classifications (-1). We use the term *reward scale* to indicate that this approach adjusts the magnitude but not the sign of the reward. Section 3.3 presents various designs of the reward scale.

3.2. Reinforcement Learning Algorithms

We select three different RL algorithms and adapt them to learn optimal policies for fair classification in the formalized CMAB problem. These algorithms include one classical CMAB algorithm that addresses the linear relationship between the expected reward and the context, as well as two popular deep RL algorithms for MDP problems, Deep Q-Network (DQN) and Proximal Policy Optimization (PPO), which allow us to leverage non-linear approximations. The two deep RL algorithms are selected as they are representative of the two key types of deep RL approaches: value-based methods and policy gradient methods. By employing these three algorithms, we aim to investigate the application of diverse RL methods.

3.2.1. LinUCB

The classical CMAB algorithm, disjoint Linear UCB (LinUCB) (Li et al., 2010) assumes a linear relationship between the context embedding x_t and the reward $E[r_{t,a}|x_t] = x_t^\top \theta_a$. A benefit of disjoint LinUCB over other CMAB algorithms is that each class has a unique learnable weight vector θ_a, which makes it suitable for classification with many classes. In each round, the agent chooses the arm (i.e. class label) with the highest score $\hat{\theta}_a^\top x_t + \alpha\sqrt{x_t^\top A_a^{-1} x_t}$, based on the context vector x_t. This is a combination of the mean of the expected payoff, $\hat{\theta}_a^\top x_t$, and the standard deviation $\sqrt{x_t^\top A_a^{-1} x_t}$, weighted with parameter α to control the level of exploration. The weight vector of each arm is defined as $\hat{\theta}_{a_t} = A_{a_t}^{-1} b_{a_t}$. Here the covariant matrix A_{a_t} is calculated with the history of context vectors chosen by that arm, $A_a = \lambda I_d + \sum_{s=1}^{t-1} x_s x_s^\top$. The vector b_a is the mean context vector of the arm weighted by the obtained rewards, $b_{a_t} = \sum_{s=1}^{t} r_{s,a_t} x_{s,a_t}$.

3.2.2. DQN$_{\text{bandit}}$

To adapt the MDP algorithms for a CMAB problem, our CMAB implementation is congruent with a one-step MDP, where each initial state is sampled from the existing set of context $s_1 \in X$, and each second state is a terminal state. In DQN (Mnih et al., 2015), the agent learns a Q-function, parameterized by ϕ, to estimate the return for each state-action pair. According to the Bellman equation (Bellman, 1957), the optimal Q-value, Q^*, of two sequential states are linked by:

$$Q^*(s, a) = \mathbb{E}_\pi[r_t + \gamma \max_{a'} Q^*(s_{t+1}, a')] \quad (1)$$

In our case (the one-step MDP), each next state is the terminal state, after which there is no reward, thus we obtain, $Q_\phi(s_{t+1}, \cdot) = 0$, and $G_t = r_t$. The parameters of ϕ are optimized using the mean-squared error between the current Q-value, $Q_\phi(s_t, a_t)$, and the updated value provided in Equation 1. The updated value is computed as $r_t + \gamma \max_{a'} Q_\phi(s_{t+1}, a')$, but since the next state is always the terminal state it reduces to r_t. We finalize the adaptation of DQN for the CMAB by casting the states as context vectors, obtaining the loss function:

$$L_{DQN}(\phi) = \mathbb{E}_{(x_t, a_t, r) \in B}\left[(r - Q_\phi(x_t, a_t))^2\right]$$

The network is updated by sampling a minibatch of tuples B from the replay buffer. The DQN_{bandit} enables exploration using an ϵ-greedy policy for selecting actions.

3.2.3. PPO$_{\text{bandit}}$

Different from DQN, in Proximal Policy Optimization (PPO) (Schulman et al., 2017), the policy π (parameterized by θ) is directly optimized under

found in the appendix and original papers.

the objective of selecting the best action. The general objective in policy gradient methods is to maximize: $\mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T}\nabla_\theta \log \pi_\theta(a_t|s_t)\cdot A_t\right]$. The advantage A_t is computed as $A_t = Q_\pi(s_t, a_t) - V_\phi(s_t)$, where a critic network V_ϕ is used to estimate the state value. PPO ensures the policy does not deviate too far during an update, by scaling the advantage with the probability ratio, $r_t(\theta)$. This ratio is clipped to create a conservative lower bound to control the policy's change at each step. The actor's objective function is thus defined as:

$$L_{actor}(\theta) = \mathbb{E}_t\left[\min(r_t(\theta)A_t, \text{clip}_\epsilon(r_t(\theta))A_t)\right]$$

To adapt PPO for CMAB, the sequential component is removed and the state s_t is replaced by the context vector x_t. For the actor loss, the advantage changes and is calculated as $A_t = r_t - V_\phi(x_t)$. The return again reduces to the reward, thereby simplifying the critic loss to:

$$L_{critic}(\phi) = \mathbb{E}_t\left[(V_\phi(x_t) - r_t)^2\right]$$

Lastly, the final loss of the PPO_{bandit} agent contains a penalty that maximizes the policy's entropy of the context vector to encourage exploration.

3.3. Reward Scales

Below we describe four different implementations of reward scaling to mitigate imbalances of protected groups. For context, we use the profession classification dataset, BiasBios, where reward scaling tackles the sub-class imbalance of the protected attribute gender. To illustrate the influence of various reward scales Figure 2 shows the scales of a balanced (Professor) and an imbalanced (Nurse) class for the protected groups with attribute gender.

For the first method, we cast the work of Lin et al. (2020) into our reward scaling framework and extend it to the multi-class classification setting. Therefore we reduce the reward for the majority by scaling it with the imbalanced ratio $\rho^a_{imb} = \frac{|D^a_{min}|}{|D^a_{maj}|}$, which is the ratio between the number of samples of the minority and majority class in class a.

$$\mathcal{W}_{\rho+}(a, g) = \begin{cases} 1 & \text{if } g \text{ is minority in } a \\ \rho^a_{imb} & \text{if } g \text{ is majority in } a \end{cases}$$

Figure 2 demonstrates that $\mathcal{W}_{\rho+}(x)$ scales with a reverse of the bias within a class, however, compared to a balanced class, the reward scale of the majority is very low. Therefore, we propose a second design that keeps the scales of the majority group in the imbalanced class equal to the scales of the balanced class. Thus, we set the majority value at 1 and only increase the minority value, based on the inversed imbalanced ratio.

$$\mathcal{W}_{\rho-}(a, g) = \begin{cases} (\rho^a_{imb})^{-1} & \text{if } g \text{ is minority in } a \\ 1 & \text{if } g \text{ is majority in } a \end{cases}$$

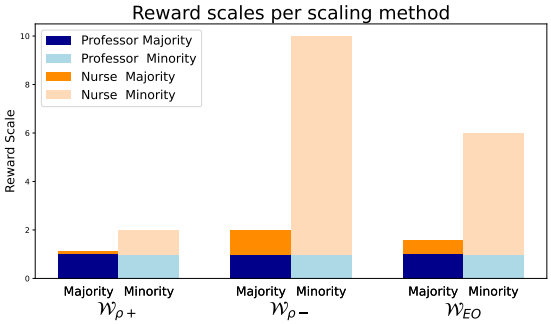

Figure 2: Reward scales for the professions with different gender imbalances *Professor* (50/50) and *Nurse* (90/10) using the different scaling functions.

The third implementation adopts the Equal Opportunity (EO) formalization used by Han et al. (2022a). Contrary to the previous two methods it ensures the average weights per class remain equal, providing an improved theoretical fairness among classes. The EO objective is achieved by aggregating the loss per sensitive state and then scaling it. However, our work scales per instance, thus we convert the EO objective to instance-specific weights and obtain:[2]

$$\mathcal{W}_{EO}(a, g) = \frac{1}{2}\frac{1}{P(g|a)}$$

Lastly, we also employ the Inverse Probability Weighting (IPW) technique (Höfler et al., 2005). Full fairness across classes and protected groups is obtained by scaling with the joint probability, resulting in:

$$\mathcal{W}^{IPW}(a, g) = \frac{1}{P(a, g)}$$

3.4. Loss Reweighting

Parallel to reward scaling in RL is (instance) reweighing in supervised learning (Han et al., 2022a; Lahoti et al., 2020), here loss reweighting for clarity. Loss reweighting has been a popular technique for imbalanced datasets, where the loss of each data sample is scaled to mitigate the class imbalance, traditionally using the IPW (Höfler et al., 2005). The weighted cross-entropy loss using the true probability p, predicted probability q:

$$L^{CE} = -\sum_{x,g}\sum_{a}\mathcal{W}(a, g)p(a|x)\log q(a|x)$$

We implement supervised learning with loss reweighing for comparison and highlight the connection between loss reweighing and reward scaling.

[2]Details can be found in the Appendix.

4. Experiments

4.1. Dataset

The BiasBios (De-Arteaga et al., 2019) consists of 393,423 biographies labeled with one of 28 professions, and a binary gender label. Following De-Arteaga et al. (2019), the data is randomly split according to 65% training, 25% testing, and 10% for validating. The dataset contains two imbalances: varying frequencies of the professions and a difference in gender percentage for each class.

Following Ravfogel et al. (2020) and Han et al. (2022a) we also evaluate on the Emoji (Elazar and Goldberg, 2018) sentiment analysis task of Twitter data (Blodgett et al., 2016). The task involves binary sentiment classification evaluation with race as the protected attribute, approximated through the provided labels Standard American English (SAE) and African American English (AAE). As per Han et al. (2021), the dataset is composed of Happy (40% AAE, 10% SAE), and Sad: (10% AAE, 40% SAE). We use the same train, dev, and test splits of 100k/8k/8k instances, respectively.

Context Vectors Each textual data sample is embedded into a context vector via a pretrained encoder, enabling the algorithms for classification. Following Ravfogel et al. (2020) we use the same fixed pretrained encoder for each dataset. For the BiasBios dataset, each biography is encoded using the `[CLS]` output of the uncased BERT-base model (Devlin et al., 2019). For the Emoji dataset, we use the DeepMoji encoder (Felbo et al., 2017), which has been demonstrated to capture a diverse range of moods and demographic information.

4.2. Metrics

Following prior work, we evaluate performance using accuracy and fairness using the True Positive Ratio (TPR) gap (De-Arteaga et al., 2019; Ravfogel et al., 2020). The TPR gap of a class $a \in A$ is calculated as: $TPR^a_{gap} = TPR^a_g - TPR^a_{\sim g}$, where g and $\sim g$ represent the two options for the sensitive states. The global TPR metric, GAP, is then calculated as the root mean square of the individual metrics:

$$GAP = \sqrt{\frac{1}{|A|}\sum_{a\in A}\left(TPR^a_{gap}\right)^2} \qquad (2)$$

To quantify performance and fairness as a single metric we use the *Distance To the Optimum* (DTO) introduced in Han et al. (2022a). DTO combines the metrics (accuracy, 1-GAP) as dimensions of evaluation space and computes the Euclidean distance between the achieved and Utopian point. The smaller the distance to the Utopian point (lower DTO), the better. We report the DTO with the Utopian accuracy and GAP as the best values across all evaluated models.

While accuracy measures the overall performance and GAP the disparity among protected groups within a class, these metrics do not capture imbalance performance across classes. Therefore we also evaluate our algorithms using the macro-averaged F1 metric to detect if minority classes are ignored. All metrics are scaled by 100 for ease of reading and all metrics are represented in the tables as the mean $\pm$ std over 5 random seeds, except DTO which is taken over the mean.

4.3. Hyperparameters and Models

Each algorithm uses the same classifier architecture, except LinUCB, which has a custom set of learnable parameters. The classifier has one hidden layer MLP. All models are trained for 10 epochs, except LinUCB, which achieved optimal performance within 2 epochs. All models are evaluated on the validation set after 50k iterations to account for different convergence speeds of models. The best model throughout training and across hyperparameters is selected using DTO. We apply hyperparameter optimization on both datasets for each of the algorithms.[3]

4.4. Comparison Models

Besides the supervised implementation in Section 3.4, abbreviated to **Sup**, we also compare our models against two embedding debiasing models, one of which we use later in Section 5.5. For clarity and brevity we leave results of further baselines to the appendix. **INLP** (Ravfogel et al., 2020) debiases embeddings by iteratively training classifiers to predict the protected attribute, it then removes this information from the embedding using a projection of the classifier's nullspace. **MP** (Haghighatkhah et al., 2022) simplifies the INLP setup by using a single Mean Projection (MP) between the representation of each class's protected groups. We implement these existing methods with the same training settings for fair comparison. Notably, we highlight how Supervised $\mathcal{W}^{EO}$ is theoretically equal to instance reweighing in Han et al. (2022a), but our implementation achieves significantly higher performance.

5. Results and Analysis

This section presents the experimental results of the different RL algorithms for fair classification. We begin by analyzing the different reward scaling functions, after which we show the performance of

[3] Details can be found in the supplementary material.

Algo		Accuracy ↑	GAP ↓	F1
SUP	$\mathcal{W}_{\rho+}$	79.3 ± 0.1	7.9 ± 0.3	69.3 ± 0.3
	$\mathcal{W}_{\rho-}$	79.8 ± 0.3	6.9 ± 0.2	**71.8 ± 0.6**
	$\mathcal{W}_{EO}$	**80.1 ± 0.2**	7.1 ± 0.5	71.7 ± 0.5
	$\mathcal{W}_{IPW}$	72.1 ± 0.7	**6.1 ± 0.3**	64.8 ± 0.8
PPO	$\mathcal{W}_{\rho+}$	74.6 ± 0.7	9.9 ± 0.8	49.7 ± 2.2
	$\mathcal{W}_{\rho-}$	78.8 ± 0.1	**8.4 ± 0.6**	64.7 ± 0.8
	$\mathcal{W}_{EO}$	**79.2 ± 0.2**	8.5 ± 0.2	**66.0 ± 0.8**
	$\mathcal{W}_{IPW}$	45.8 ± 6.9	10.5 ± 0.9	45.3 ± 5.8
DQN	$\mathcal{W}_{\rho+}$	76.2 ± 1.1	10.4 ± 0.7	57.2 ± 4.8
	$\mathcal{W}_{\rho-}$	**79.3 ± 0.1**	11.1 ± 0.6	65.8 ± 1.4
	$\mathcal{W}_{EO}$	79.2 ± 0.1	**10.1 ± 0.4**	**66.4 ± 0.2**
	$\mathcal{W}_{IPW}$	74.6 ± 0.3	12.8 ± 0.2	56.6 ± 0.3
LinUCB	$\mathcal{W}_{\rho+}$	72.8 ± 0.1	12.0 ± 0.5	54.6 ± 0.9
	$\mathcal{W}_{\rho-}$	74.1 ± 0.4	11.6 ± 0.5	59.3 ± 1.7
	$\mathcal{W}_{EO}$	**74.6 ± 0.2**	12.2 ± 0.5	**59.8 ± 1.1**
	$\mathcal{W}_{IPW}$	37.3 ± 2.5	**10.3 ± 0.7**	35.4 ± 1.0

Table 1: Results with different reward scaling on BiasBios for the various algorithms

our best model on the two datasets and compare them against existing baselines. We then evaluate the behavior of different algorithms under various imbalance ratios and examine their robustness to various degrees of *representational fairness*.

5.1. Reward Function Impact

Table 1 presents the results of implementing different reward scales (discussed in Section 3.3) across four algorithms: supervised learning (SUP), PPO, DQN, and LinUCB.

The results consistently demonstrate that the imbalance ratio ρ yields substantial gains in fairness and accuracy when applied to increase the reward for the minority class ($\mathcal{W}_{\rho-}$) as opposed to decreasing the reward for the majority class ($\mathcal{W}_{\rho+}$). This effect is particularly pronounced for reinforcement learning algorithms, with PPO showing the most significant performance gap between these two scaling approaches. This suggests that RL algorithms might struggle to perform optimally under low reward scenarios. Scaling with the joint probability of class and protected attribute ($\mathcal{W}_{IPW}$) confirms our hypothesis that this approach is too unstable across all algorithms. While it occasionally achieves the best fairness scores (lowest GAP), this comes at a substantial cost to accuracy and F1, particularly for PPO and LinUCB where performance drops drastically.

The difference between $\mathcal{W}_{EO}$ and $\mathcal{W}_{\rho-}$ is minimal across all four algorithms, as expected from their similar reward scales depicted in Figure 2. Notably, $\mathcal{W}_{\rho-}$ achieves better fairness (lower GAP) than $\mathcal{W}_{EO}$ in most cases, while $\mathcal{W}_{EO}$ generally yields higher accuracy. Despite this slight fairness advantage of $\mathcal{W}_{\rho-}$, we choose $\mathcal{W}_{EO}$ for our experiments due to its stronger theoretical foundation.

5.2. Baseline Comparison

The results of our models on the two dataset compared to the baseline models are summarized in Table 2 and demonstrate several key findings. We apply our models with the best performing scaling (EO), see Section 5.1. On the multi-class BiasBios dataset, our deep RL algorithms (PPO and DQN) achieve competitive performance to our scaled supervised approach. Notably, PPO outperforms DQN in fairness, as indicated by PPO's lower GAP score. In contrast on the Emoji dataset, the classical CMAB algorithm LinUCB excels, achieving one of the best performance-fairness trade-offs, as indicated by the low DTO score. Our experiments suggest that LinUCB may exhibit improved generalization for tasks with few classes, but its performance might deteriorate as the number of classes increases, potentially due to the constraints of its linear classifier.

Notably, the F1 score for the deep RL algorithms is considerably lower than the baselines on BiasBios. Further analysis of per-class metrics reveals that while the F1 for most classes was on par with the supervised setup, both deep RL algorithms failed to recall two of the very sparse classes while performing well on all others.

Contrary to Han et al. (2022a) who found that loss scaling with EO offers minimal benefits,[4] our implementation demonstrates it is a powerful technique outperforming baselines on BiasBios.

Moreover, although RL algorithms have a reputation for being compute-intensive, the training time estimations in Table 2 show that our PPO implementation is fast for both datasets, and LinUCB is fast for a lower number of classes. LinUCB's long training time on BiasBios suggests computation speed is bottle-necked by the class-dependent arm calculations, which could be further parallelized. While we made initial efficiency improvements, we leave further improvements for future work but estimate computational overhead is an implementation challenge rather than an algorithmic limitation.

5.3. Scaling Impact per RL agent

We investigate the influence of reward scaling on our models by training them with and without scaling. Table 3 presents the results on BiasBios as the mean performance without scaling and the change in metrics when EO scaling is applied.

Without reward scaling the three RL algorithms achieve similar accuracy to the supervised ap-

[4]The EO scaled supervised implementation of Han et al. (2022a) achieves an Accuracy of 75.7 and GAP of 13.9

Algorithm	BiasBios (28 Classes) Accuracy ↑	GAP ↓	DTO ↓	F1 ↑	Time ↓	Emoji (2 Classes) Accuracy ↑	GAP ↓	DTO ↓	Time ↓
Sup	81.0 ± 0.1	16.4 ± 0.5	9.3	73.8 ± 0.3	1.0	72.3 ± 0.1	38.1 ± 0.6	28.3	1.0
INLP	80.2 ± 0.6	9.7 ± 0.4	2.8	71.7 ± 1.4	50.1	63.5 ± 3.6	24.1 ± 5.4	18.6	3.6
MP	**81.1 ± 0.1**	13.9 ± 0.6	6.8	**74.0 ± 0.2**	2.6	71.8 ± 0.3	17.1 ± 1.0	8.1	2.3
Sup^{EO}	80.1 ± 0.2	**7.1 ± 0.5**	**1.0**	71.7 ± 0.5	1.0	**75.5** ± 0.1	11.4 ± 1.1	1.4	1.0
LinUCB^{EO}	74.6 ± 0.2	12.2 ± 0.5	8.3	59.8 ± 1.1	31.9	75.3 ± 0.2	10.4 ± 0.7	**0.5**	2.8
DQN^{EO}_{bandit}	79.2 ± 0.1	10.1 ± 0.4	3.6	66.4 ± 0.2	57.4	70.8 ± 0.8	**10.0 ± 1.0**	4.8	30.2
PPO^{EO}_{bandit}	79.2 ± 0.2	8.5 ± 0.2	2.4	66.0 ± 0.8	2.9	75.4 ± 0.1	14.4 ± 0.6	4.4	3.0

Table 2: Results on the BiasBios and Emojis classification datasets for our models (in grey) and the baselines. Metrics provided as mean ± std over 5 random seeds, except DTO which is computed over the mean Accuracy, and GAP, and Time which is the relative time compared to the supervised baseline (first row).

Algo	Accuracy ↑	GAP ↓	F1 ↑
Sup	**81.0** (- 0.9)	16.4 (- 9.3)	**73.8** (- 2.1)
LinUCB	78.4 (- 3.8)	15.5 (- 3.3)	67.3 (- 7.5)
DQN_{bandit}	80.1 (- 0.9)	**13.7** (- 3.6)	66.5 (- 0.1)
PPO_{bandit}	79.7 (- 0.5)	14.4 (- 5.9)	67.5 (- 1.5)

Table 3: Results on the Bias dataset **without** reward scaling, presented as mean and difference from the case without EO, where red (worse), blue (better).

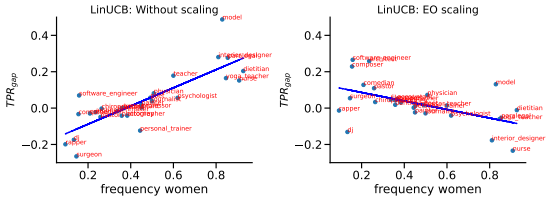

Figure 3: TPR gap plotted against the gender distribution per profession for LinUCB. Left without reward scaling and right with EO reward scaling

proach but at the cost of a lower F1 score. As mentioned above, the RL algorithms fail on two very sparse classes, which explains the drop in GAP and F1. Failing to classify any instances of a class correctly results in a TPR gap of 0 for that class, since the result is "fair" among both genders.

The EO reward scale significantly reduces the GAP of all implementations, at the cost of a slight decrease in Accuracy and F1 for most models. However, on LinUCB the scaling causes a large performance reduction with only a small GAP reduction, suggesting that scaling hinders the performance more than it improves the fairness.

Analysis of LinUCB's performance and fairness per group (Figure 3) sheds more light on why the performance drops. Without scaling, LinUCB's performance follows a predictable positive correlation with gender imbalance, favoring the majority group. However, with reward scaling this trend inverts, leading the model to perform better for minority groups. This suggests LinUCB is oversensitive to scaling on the BiasBios dataset, causing it to overcompensate and penalize the majority group.

5.4. Sensitivity to Imbalance

We investigate each model's sensitivity to subclass imbalance by training them on the Emoji dataset across a range of stereotyping ratios. A stereotyping ratio represents the proportion of the AAE and SAE samples in each class. For example, a stereotyping ratio of 0.2 means the data is distributed as Happy (20% AAE, 80% SAE), Sad (80% AAE, 20% SAE).

Figure 4 reveals a strong inverse relationship between LinUCB's fairness and the stereotyping ratio. Although the stereotypical ratios are symmetric at the value of 0.5 the fairness of LinUCB is asymmetric. Thus there is a residual representation bias in the data that is not addressed by the reward scaling. In contrast, the supervised approach maintains mostly stable fairness, except for the most extreme ratios. Interestingly, LinUCB reveals a reverse pattern in best and worst fairness.

The relatively low accuracy of DQN and poor performance on fairness of PPO are consistent across ratios. However, PPO does have the most constant fairness and performance across stereotyping ratios, indicating good training stability.

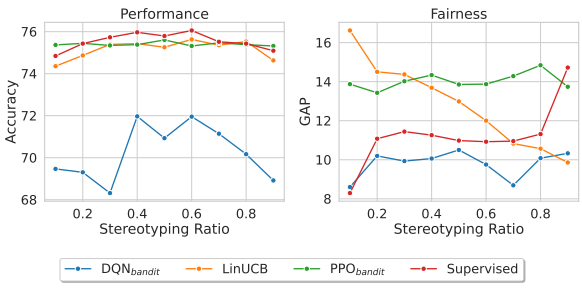

Figure 4: Performance (Accuracy) and Fairness (GAP) on the Emoji dataset using different stereotyping ratios. All models use the scaling of $\mathcal{W}^{EO}$.

Algo+$\mathcal{W}^{EO}$	Explicit Gender Info		MP-Debiased	
	Accuracy ↑	**GAP** ↓	**Accuracy** ↑	**GAP** ↓
Sup	**80.2** (+ 0.1)	**7.2** (+ 0.1)	**80.0** (- 0.1)	**7.4** (+ 0.3)
LinUCB	74.5 (+ 0.1)	11.7 (- 0.5)	74.3 (- 0.3)	11.5 (- 0.7)
DQN_{bandit}	79.2 (+ 0.0)	10.0 (- 0.1)	79.0 (- 0.2)	8.6 (- 1.5)
PPO_{bandit}	79.3 (+ 0.1)	8.7 (+ 0.2)	79.2 (+ 0.0)	9.7 (+ 1.2)

Table 4: Results on the BiasBios with added gender info (left) and MP-debiased (right), presented as mean, and difference without change: red (worse), blue (better).

5.5. Signal strength vs. Scaling

We now examine how the strength of the protected information affects the efficacy of reward scaling. We focus on two scenarios that modify the gender signal in the representations: 1) adding explicit gender information, thus increasing the gender signal strength 2) debiasing the embeddings using MP, which reduces it. Table 4 presents the mean and relative difference compared to the results without the specified modification.

Providing the model with gender information increases the overall accuracy. However, the impact on fairness, as indicated by the GAP score varies among algorithms. The GAP score increases for the two algorithms with the lowest GAPs (Sup, PPO) and decreases for the two with the highest GAP (LinUCB, DQN). Only the algorithms that perform worse on fairness benefited from access to protected attribute.

Removing the bias with MP reduces the test accuracy for nearly all algorithms, indicating some useful information is removed. Again, the modification increased relatively low GAP scores and decreased relatively high scores. As such, changing to a representation with relatively low bias helps LinUCB and DQN, whereas Sup and PPO that already achieved better fairness mainly see their overall performance hindered.

Analysis representational fairness Notably, the differences in Table 4 are relatively small and hardly ever surpass the standard deviation provided in Table 2. This suggests that while the strength of the protected information influences performance and fairness, the impact might be less pronounced than the choice of algorithmic design. Moreover, Table 2 demonstrated that the GAP score of all four scaled methods is lower than that of MP. Thus representational fairness appears to have a less significant effect for downstream fairness than algorithmic methods such as reward scaling.

6. Comparative Analysis

To inform trade-offs for method selection in fair classification tasks, we analyze and summarize the key differences here.

Performance and Computational Trade-offs LinUCB shows fast training, saturating within two epochs with only one hyperparameter (α), but training time scales poorly with class count—from 3× as much as the supervised system on binary Emoji to 32× on 28-class BiasBios (Table 2). PPO_{bandit} maintains consistent training time (3× for both datasets). Without scaling, DQN demonstrates one of the best accuracy-fairness trade-offs (Table 3). Experiments on the top 8 BiasBios classes reveal that DQN achieves the best accuracy and GAP without scaling, suggesting natural sub-class imbalance handling given sufficient samples. This contrasts with LinUCB and PPO_{bandit}, which require scaling for comparable performance. However, LinUCB is overly sensitive to scaling and can overshoot, hurting majority class performance more than it helps minority groups (Figure 4), while PPO_{bandit} maintains stable fairness across different stereotyping ratios. Beyond these practical considerations, the CMAB reward formulation also opens up the possibility of optimizing non-differentiable fairness objectives (e.g., the TPR gap) directly, a promising direction for future work.

Method Selection Guidelines For binary classification with computational constraints, LinUCB provides good speed and performance. DQN_{bandit} may be suitable when the right scaling factors are not known and when moderate fairness improvements are acceptable without extensive tuning, given its good performance without scaling. PPO_{bandit} works well for multi-class scenarios where training stability matters. Supervised learning with EO scaling remains practical when both efficiency and good performance are needed, consistently performing well while being straightforward to implement. Future work is needed to determine which of the limitations we found in our RL systems are inherent to the method and which can be resolved by minor modifications.

7. Conclusion

This paper introduces a novel approach to fair classification using the Contextual Multi-Armed Bandit (CMAB) framework and explores various Reinforcement Learning (RL) algorithms. Our findings demonstrate the potential of different RL algorithms for this task and the efficacy of reward scaling in mitigating imbalances of protected groups. Concerning representational fairness, our experiments provided further evidence that the signal strength of the protected attribute had minimal impact compared to scaling methods.

We believe the proposed framework presents a promising approach to leverage RL algorithms for fair classification, opening up new research avenues. We encourage future work to extend our framework by exploiting different RL characteristics, such as model updates for MDP algorithms based on non-differentiable fairness metrics.

Limitations

Important limitations of this work can be divided into two sections: 1) Limitations of the dataset and data requirements of our models 2) Limitations specific to our algorithms and experiments, independent of the data.

Data limitation Firstly, all datasets considered in this study used English text, which restricts the analysis and might miss other types of biases related to different linguistic and cultural contexts. Secondly, the protected groups evaluated in this study simplified to binary labels, which excludes people who do not fall into this category such as non-binary individuals and the multidimensional nature of ethnicity.

Our reward scaling approach also requires these labels for classification. Although our setup could easily be extended to cases with more labels, it would be interesting to see fair classification with protected attributes as continuous values. But the lack of good benchmarks restricts the evaluation of such cases.

Algorithmic Limitation Firstly, our paper used two deep RL MDP algorithms and one linear classical CMAB agent. We recognize that while linear agents have a significant focus in the CMAB literature, the fast field includes options with non-linear algorithms that could also be applied to this task. The choice of LinUCB does not represent the state-of-the-art, but rather a classical high-performance implementation.
Second, the various hyperparameters limit the extent of general statements about each algorithm. We have documented our hyperparameter search and training methods in the supplementary materials which are available in the appendix, to ensure the interpretability of our experiments, but our results only demonstrate the capabilities of our best implementation. Moreover, the use of DTO to select the best model throughout training fails to account for potential trade-offs between fairness and accuracy at different points in training. For example, on the Emoji dataset, PPO underperformed in Fairness and DQN in accuracy. However, it is possible that at another pointing training with a higher DTO score, the trade-off between fairness and accuracy was reversed.

Ethics Statement

The application of the paper was to improve fairness among protected groups in classification. However, no algorithm is able to obtain perfect fairness and remove the bias perfectly. Therefore applications of the mentioned debiasing methods should always strongly take the mentioned limitations into account. Moreover, the current experiments are limited to specific datasets and real-world use cases may be different. Careful evaluation and testing system behavior in the intended setting with input from experts who can judge the consequences of remaining bias is essential.

Acknowledgments

This research was partially funded by the Hybrid Intelligence Center, a 10-year program funded by the Dutch Ministry of Education, Culture and Science through the Netherlands Organisation for Scientific Research, https://hybrid-intelligence-centre.nl.

8. Bibliographical References

Carlos Aguirre, Kuleen Sasse, Isabel Cachola, and Mark Dredze. 2024. Selecting shots for demographic fairness in few-shot learning with large language models. In *Proceedings of the Third Workshop on NLP for Positive Impact*, pages 50–67.

Richard Bellman. 1957. *Dynamic Programming*, 1 edition. Princeton University Press, Princeton, NJ, USA.

Maria De-Arteaga, Alexey Romanov, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, and Adam Tauman Kalai. 2019. Bias in bios: A case study of semantic representation bias in a high-stakes setting. In *proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 120–128.

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

Maria Dimakopoulou, Zhengyuan Zhou, Susan Athey, and Guido Imbens. 2019. Balanced linear contextual bandits. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3445–3453.

Miroslav Dudík, Dumitru Erhan, John Langford, and Lihong Li. 2014. Doubly robust policy evaluation and optimization. *Statistical Science*, 29(4):485–511.

Yanai Elazar and Yoav Goldberg. 2018. Adversarial removal of demographic attributes from text data. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 11–21.

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1615–1625.

Pantea Haghighatkhah, Antske Fokkens, Pia Sommerauer, Bettina Speckmann, and Kevin Verbeek. 2022. Better hit the nail on the head than beat around the bush: Removing protected attributes with a single projection. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8395–8416.

Xudong Han, Timothy Baldwin, and Trevor Cohn. 2021. Diverse adversaries for mitigating bias in training. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2760–2765.

Xudong Han, Timothy Baldwin, and Trevor Cohn. 2022a. Balancing out bias: Achieving fairness through balanced training. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11335–11350.

Xudong Han, Aili Shen, Yitong Li, Lea Frermann, Timothy Baldwin, and Trevor Cohn. 2022b. Fairlib: A unified framework for assessing and improving fairness. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 60–71.

Michael Höfler, Hildegard Pfister, Roselind Lieb, and Hans-Ulrich Wittchen. 2005. The use of weights to account for non-response and drop-out. *Social psychiatry and psychiatric epidemiology*, 40:291–299.

Preethi Lahoti, Alex Beutel, Jilin Chen, Kang Lee, Flavien Prost, Nithum Thain, Xuezhi Wang, and Ed Chi. 2020. Fairness without demographics through adversarially reweighted learning. *Advances in neural information processing systems*, 33:728–740.

Lihong Li, Wei Chu, John Langford, and Robert E Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670.

Enlu Lin, Qiong Chen, and Xiaoming Qi. 2020. Deep reinforcement learning for imbalanced classification. *Applied Intelligence*, 50:2488–2502.

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

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. 2020. Null it out: Guarding protected attributes by iterative nullspace projection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7237–7256.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Aili Shen, Xudong Han, Trevor Cohn, Timothy Baldwin, and Lea Frermann. 2022. Does representational fairness imply empirical fairness? In *Findings of the Association for Computational Linguistics: AACL-IJCNLP 2022*, pages 81–95.

Jack Sherman and Winifred J Morrison. 1950. Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *The Annals of Mathematical Statistics*, 21(1):124–127.

Shivashankar Subramanian, Afshin Rahimi, Timothy Baldwin, Trevor Cohn, and Lea Frermann. 2021. Fairness-aware class imbalanced learning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2045–2051.

Victor Uc-Cetina, Nicolas Navarro-Guerrero, Anabel Martin-Gonzalez, Cornelius Weber, and Stefan Wermter. 2023. Survey on reinforcement learning for language processing. *Artificial Intelligence Review*, 56(2):1543–1575.

Christina Wadsworth, Francesca Vera, and Chris Piech. 2018. Achieving fairness through adversarial learning: an application to recidivism prediction. In *Proceedings of the FAT/ML Workshop*.

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 *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5310–5319.

Marco A Wiering, Hado Van Hasselt, Auke-Dirk Pietersma, and Lambert Schomaker. 2011. Reinforcement learning algorithms for solving classification problems. In *2011 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pages 91–96. IEEE.

9. Language Resource References

Su Lin Blodgett, Lisa Green, and Brendan O'Connor. 2016. Demographic Dialectal Variation in Social Media: A Case Study of African-American English. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1119–1130.

De-Arteaga, Maria and Romanov, Alexey and Wallach, Hanna and Chayes, Jennifer and Borgs, Christian and Chouldechova, Alexandra and Geyik, Sahin and Kenthapadi, Krishnaram and Kalai, Adam Tauman. 2019. *Bias in bios: A case study of semantic representation bias in a high-stakes setting*.

10. Appendix A: Reproducibility

10.1. Data Analysis

Because the BiasBios dataset needs to be scraped online, we provide the full composition of the BiasBios dataset split up in profession and gender in Table 5.

Profession	Female	Male
Professor	53290	64820
Physician	19579	18986
Attorney	12494	20113
Photographer	8689	15635
Journalist	9873	10077
Nurse	17236	1735
Psychologist	11385	6910
Teacher	9768	6428
Dentist	5153	9326
Surgeon	1972	11301
Architect	2398	7715
Painter	3543	4193
Model	6214	1288
Poet	3441	3570
Filmmaker	2310	4699
Software Engineer	1089	5817
Accountant	2081	3571
Composer	918	4682
Dietitian	3689	289
Comedian	592	2207
Chiropractor	690	1908
Pastor	609	1923
Paralegal	1499	268
Yoga Teacher	1406	257
Dj	211	1274
Interior Designer	1183	280
Personal Trainer	654	778
Rapper	136	1271

Table 5: Class and gender composition of the BiasBios dataset

10.2. Model Selection

Selecting the best model throughout training or across hyperparameters is strongly dependent on the selection metric. To balance fairness and performance we use the proposed method of Han et al. (2022a), and select using DTO. The full equation of DTO is provided below, where the obtained metrics are determined by the point $(Acc, (1 - GAP))$, and the utopian metrics are $(Acc^{utop}, (1 - GAP^{utop}))$.

$$DTO = \sqrt{(Acc^{utop} - Acc)^2 + ((1 - GAP^{utop}) - (1 - GAP))^2}$$

The best training timestep according to DTO is determined with utopian values (1,1), and the best hyperparameters setting utopian values as the best metric values during training (i.e. the highest performance and fairness each individually obtained, which do not necessarily belonging to the same algorithm).

The reported DTO values in table X and Y are obtained using the best performance and accuracy method as: [performance, fairness] BiasBios 28C = [0.811, 0.929], BiasBios 8C = [0.868, 0.978], Moji=[0.756, 0.900]

10.3. Hyperparameters

The architecture of the neural network for each algorithm is fixed and consists of 2 layers MLP. For the critic in PPO the architecture is the same except for the output size which is 1. Hyperparameter optimization is applied for each of the parameters of the algorithms using grid search. Table 7 shows the ranges and the best values.

Related work implementations Following previous work (Ravfogel et al., 2020; Han et al., 2022b), we use INLP and MP in a post hoc manner to the features extracted from the last hidden layer of the

	Type	Dimensions
Layer 1	Linear	$n\_features \times 128$
Layer 2	Linear	$128 \times n\_actions$
Activation	ReLU	
Optimizer	Adam	

Table 6: Neural Network Architecture

supervised model and train a logistic classifier for the final classification. For our MP debiasing experiments in section 5.5 we use MP to debias the context vectors before training, instead of poshoc on the hidden layer of the trained network.

11. Appendix B: Algorithms

11.1. Single-Step Markov Decision Process

To formalize how the policy-gradient methods such as PPO relate to the Contextual Multi-Armed Bandit framework, we define below the single-step Markov Decision Process. An MDP is defined by the tuple (S, A, P, R, γ), and our single-step variant contains only two states $S = \{s_1, s_2\}$. The initial state is sampled each time from the environment and for our classification setup is part of the set of context embeddings, $s_1 \in \{x_j\}$. To ensure data samples are treated independently the second state is always the terminal state $s_2 = s_{terminal}$. The action space is equal to the number of classes: $A = C = \{c_1, c_2, .., c_{28}\}$. The reward function R is equal to that of the CMAB and is defined in section 3.1. Lastly, each trajectory is defined as $\tau = \{s_1, a_1, s_{terminal}\}$ and both the transition probability, P, and the discount factor γ are irrelevant since each action results in the terminal state.

11.2. LinUCB

The full algorithm of LinUCB from Li et al. (2010), used in the paper is shown in Algorithm

Algorithm 1 LinUCB Algorithm

Require: Context features $x_{t,a}$ for context at time t and arm $a \in \mathcal{A}$, exploration parameter α.
  Initialize A_a and b_a for each arm $a \in \mathcal{A}$
  for each sample t **do**
    for each arm a **do**
      $\hat{\theta}_{a_t} = A_{a_t}^{-1} b_{a_t}$
      $p_{t,a} = \hat{\theta}_a^\top x_{t,a} + \alpha\sqrt{x_{t,a}^\top A_a^{-1} x_{t,a}}$
    end for
    Choose arm $a_t = \arg\max_{a \in \mathcal{A}}(p_{t,a})$, and observe real-valued payoff r_t
    Update $A_{a_t} \leftarrow A_{a_t} + x_{t,a_t} x_{t,a_t}^\top$
    Update $b_{a_t} \leftarrow b_{a_t} + r_t x_{t,a_t}$
  end for

11.3. Equal Opportunity Weights

Where Han et al. (2022a) used EO for supervised learning, their implementation achieved this objective by grouping the loss per class and then averaging over them. In this section, we see how we can use this to obtain the weights for each data sample based on the class a and protected attribute g. For two protected groups g_1 and g_2 in class a , let C_1 and C_2 be the number of samples for g_1 and g_2, and $\mathcal{W}_1$ and $\mathcal{W}_2$, be the weights. To get a statement of the weights with EO for each sensitive state, (a, g), we need two axioms.

Axiom 1. The weight scale ratio between the two protected groups of a class should be inversely proportional to their probability in the dataset:

$$\mathcal{W}_1 \cdot C_1 = \mathcal{W}_2 \cdot C_2$$

Axiom 2: To ensure fairness across classes, the average weight per profession should be a fixed value B so that:

$$\frac{1}{C_1 + C_2}\left(\mathcal{W}_1 \cdot C_1 + \mathcal{W}_2 \cdot C_2\right) = B$$

Combining these two axioms we obtain the formulation:

$$\mathcal{W}_2 = \frac{B}{2}\frac{(C_1 + C_2)}{C_2}$$

$$\mathcal{W}_2 = \frac{B}{2}\frac{1}{P(C_2)}$$

For the multi-class classification task the average reward scale, B, should be 1, and the probability is conditional on the class a, obtaining the final W_{EO} equation:

$$\mathcal{W}_{EO}(g, y) = \frac{1}{2}\frac{1}{P(g|a)}$$

12. Appendix C: Ablation Experiments

Here we add our experiments that did not make the main paper.

12.1. Analysis: Model and Data Efficiency

An important aspect for evaluation is related to the data and computational of each algorithm. For ease of comparison, all algorithms except LinUCB were trained for 10 epochs. However, DQN and PPO each reuse the seen data in a different way to deal with the data sparsity of standard RL

Algorithm	Parameter	Min	Max	Best	
				BiasBios	Emoji
PPO	lr (actor)	3.0×10^{-4}	1.0×10^{-6}	1.0×10^{-4}	3.0×10^{-5}
	lr (critic)	1.0×10^{-3}	1.0×10^{-5}	1.0×10^{-3}	1.0×10^{-4}
	Batch size	64	512	512	512
	Entropy c_2	0.01	0.1	0.2	0.1
	ϵ-clip	0.05	0.3	0.1	0.3
Supervised	lr	1.0×10^{-3}	1.0×10^{-6}	3.0×10^{-4}	1.0×10^{-3}
	Batch size	64	512	128	512
DQN	lr	3.0×10^{-4}	1.0×10^{-6}	3.0×10^{-6}	3.0×10^{-4}
	Batch size	32	256	256	32
	Eps_end	0.001	0.1	0.1	0.01
	Eps decay	0.5	1.0	0.5	0.5
LinUCB	α	0.1	3.0	1.5	2.5

Table 7: Hyperparameter ranges and best values for different algorithms. For PPO the "Entropy c_2" refers to the coefficient of the entropy in the loss.

settings. DQN is updated using a replay-buffer from which it samples a minibatch of N triplets (s, a, r) for each iteration. In contrast, PPO collects N samples during the observation phase after which it updates the model with this batch K_epoch number of times. Lastly, LinUCB achieves optimal results after 1 epoch but is constrained by the computations of its weight matrices, which require the inverse of a square matrix with dimension $n\_features$. For computational efficiency, we use the Sherman–Morrison formula which updates the previous computed inverse with a rank one update (Sherman and Morrison, 1950)

The time complexities in Table 2, demonstrate that PPO is closest to supervised learning and that DQN takes significantly more time since it needs to sample from the buffer at each iteration. Notably, LinUCB is strongly dependent on the number of classes, reducing its relative efficiency from 32 to 3 times that of Supervised Learning. The bottleneck here is that it needs to compute an upper confidence bound for each class.
Another important feature is the sensitivity to hyperparameters. PPO and DQN are sensitive to several hyperparameters that determine the level of its exploration, such as DQN's mini-batch size or exploration parameter, or PPO's entropy and clipping coefficients. LinUCB is easiest to implement in this regard and does not require any neural network hyperparameters, but only one exploration parameter α, see section 11.2.

13. Appendix D: Full result for experiments

To distinguish the sensitivity of gender imbalance and data-sparsity we also run experiments with a subset of the data, following Aguirre et al. (2024), and select only the professions that have at least 1000 samples for both genders in the test set, resulting in 8 professions.

13.1. BiasBios: training performance over time

In Reinforcement Learning literature it is common to provide the performance of an algorithm throughout training for evaluation. Therefore we provide the evaluation accuracy of our four algorithms in Figure 5

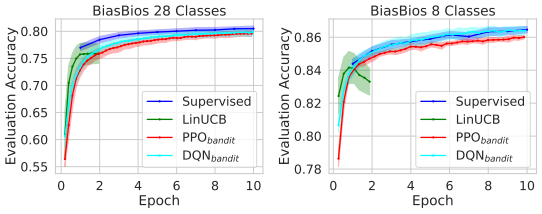

Figure 5: Evaluation accuracy of the different algorithms the full 28 classes and the 8 class subset of the Bias in Bios dataset

13.2. BiasBios: Recall per profession

As a further analysis of the lacking F1 score of the RL algorithms compared to the supervised implementation, we provide the Recall scores as a

percentage of the class. Since class 21, Professor appears significantly more often than the most common class after it, we leave it out for clarity.

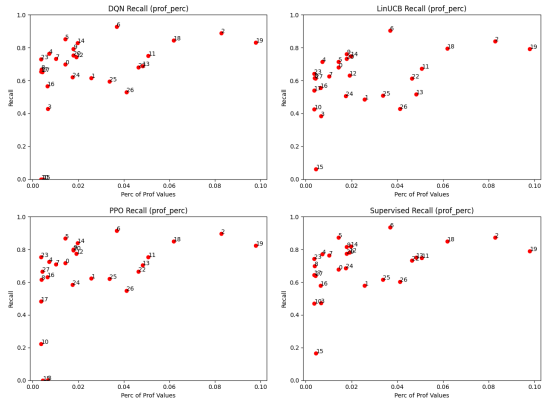

Figure 6: Recall of each class on the BiasBios dataset for the four algorithm implementations

13.3. Full tables: Baselines on BiasBios and Emoji

In Section 5.2, we presented a focused comparison between different RL algorithms to analyze their relative strengths and weaknesses. For a broader evaluation context, we provide additional baseline comparisons in this appendix. While these supplementary results offer valuable insights, we chose to separate them from the main results section to maintain clarity in our analysis of RL approaches. Specifically, we present here the performance of BTEO and DAdv models, which complement our primary findings. DAdv, (Han et al., 2021) removes sensitive information from the embeddings by applying adversarial training using diverse adversaries. Lastly, BTEO (Han et al., 2022a) subsample the dataset to establish equal opportunity. To this end we offer these additional comparisons in this section to contextualize our method's thatcapabilities within the broader landscape of available approaches. Note while Table 2 provides comparative results, it should not be interpreted as a comprehensive benchmark against state-of-the-art performance.

Table 8 demonstrates that supervised learning with scaling still obtains the best performance and fairness on BiasBios, and that PPO and DQN are comparable to BTEO and DAdv. On the Emoji dataset, BTEO and LinUCB achieve best overall performance, with BTEO obtaining 0.1 % higher accuracy.

13.4. Full tables: BiasBios (28C and 8C)

Some of our results in section 5.3 are presented as the mean only. The full results of our algorithms as the mean and std over the five seeds is provided in the tables here. Table 9 shows the performance of our algorithms with and without reward scaling on the BiasBios dataset with the 28 and 8 classes.

13.5. Full tables: four reward scaling methods

The results from reward scaling using the four described scales and our four algorithms are shown in Table 10.

13.6. Full results: Explicit gender information and Ensemble techniques

This section includes the full results of Section 5.5, after adding the gender information explicitely and after removing it with MP. The results are presented as mean and standard deviation over 5 seeds in Table 11 and Table 12

Algorithm	**BiasBios** (28 Classes) Accuracy ↑	GAP ↓	DTO ↓	F1 ↑	Time ↓	**Emoji** (2 Classes) Accuracy ↑	GAP ↓	DTO ↓	Time ↓
Sup	81.0 ± 0.1	16.4 ± 0.5	9.3	73.8 ± 0.3	1.0	72.3 ± 0.1	38.1 ± 0.6	28.3	1.0
INLP	80.2 ± 0.6	9.7 ± 0.4	2.8	71.7 ± 1.4	50.1	63.5 ± 3.6	24.1 ± 5.4	18.6	3.6
MP	**81.1 ± 0.1**	13.9 ± 0.6	6.8	**74.0 ± 0.2**	2.6	71.8 ± 0.3	17.1 ± 1.0	8.1	2.3
BTEO	79.2 ± 0.3	8.4 ± 0.6	2.3	68.1 ± 0.4	1.7	75.4 ± 0.1	10.4 ± 1.0	**0.4**	0.8
DAdv	80.8 ± 0.2	8.5 ± 0.6	1.4	72.9 ± 0.4	4.8	**75.6 ± 0.3**	11.6 ± 1.7	1.6	5.7
Sup^{EO}	80.1 ± 0.2	**7.1 ± 0.5**	**1.0**	71.7 ± 0.5	1.0	75.5 ± 0.1	11.4 ± 1.1	1.4	1.0
LinUCB^{EO}	74.6 ± 0.2	12.2 ± 0.5	8.3	59.8 ± 1.1	31.9	75.3 ± 0.2	10.4 ± 0.7	0.5	2.8
DQN$^{EO}_{bandit}$	79.2 ± 0.1	10.1 ± 0.4	3.6	66.4 ± 0.2	57.4	70.8 ± 0.8	**10.0 ± 1.0**	4.8	30.2
PPO$^{EO}_{bandit}$	79.2 ± 0.2	8.5 ± 0.2	2.4	66.0 ± 0.8	2.9	75.4 ± 0.1	14.4 ± 0.6	4.4	3.0

Table 8: Results on the BiasBios and Emojis classification datasets for our own models (in grey) and the baselines. Metrics are provided as mean ± std over 5 random seeds, except DTO which is computed over the mean Accuracy, and GAP, and Time which is the relative time compared to the supervised baseline (first row).

Algorithm	28 Classes Accuracy ↑	GAP↓	DTO ↓	F1↑	8 Classes Accuracy ↑	GAP ↓	DTO ↓	F1↑
Sup	**81.0 ± 0.1**	16.4 ± 0.5	10.0	**73.8 ± 0.3**	**86.8 ± 0.1**	8.3 ± 0.7	6.2	**82.7 ± 0.1**
LinUCB	78.4 ± 0.1	15.5 ± 0.3	9.6	67.3 ± 0.4	85.3 ± 0.2	**7.6 ± 0.3**	5.8	80.6 ± 0.2
DQN$_{bandit}$	80.1 ± 0.2	**13.7 ± 0.3**	**7.2**	66.5 ± 1.3	86.5 ± 0.2	**7.6 ± 0.3**	**5.5**	82.2 ± 0.2
PPO$_{bandit}$	79.7 ± 0.5	14.4 ± 0.7	8.0	67.5 ± 2.0	86.0 ± 0.2	8.7 ± 0.4	6.7	81.6 ± 0.2
Sup^{EO}	**80.1 ± 0.2**	**7.1 ± 0.5**	**1.1**	**71.7 ± 0.5**	**86.3 ± 0.2**	2.4 ± 0.1	**0.6**	**82.0 ± 0.2**
LinUCB^{EO}	74.6 ± 0.2	12.2 ± 0.5	9.6	59.8 ±1.1	83.4 ± 0.2	7.6 ± 0.3	6.8	77.6 ± 0.3
DQN$^{EO}_{bandit}$	79.2 ± 0.1	10.1 ± 0.4	3.9	66.4 ± 0.2	86.2 ± 0.1	**2.2 ± 0.2**	0.7	81.6 ± 0.2
PPO$^{EO}_{bandit}$	79.2 ± 0.2	8.5 ± 0.2	2.7	66.0 ± 0.8	85.8 ± 0.1	2.8 ± 0.6	1.3	81.4 ± 0.2

Table 9: Results on the BiasBios dataset for the full dataset (28 classes) and a subset of the most common professions (8 classes). The first rows use a constant reward scale, and the last four (in grey) use the EO reward scale

Algo		Accuracy ↑	GAP ↓	F1
SUP	$\mathcal{W}_{\rho+}$	79.3 ± 0.1	7.9 ± 0.3	69.3 ± 0.3
	$\mathcal{W}_{\rho-}$	79.8 ± 0.3	6.9 ± 0.2	**71.8 ± 0.6**
	$\mathcal{W}_{EO}$	**80.1 ± 0.2**	7.1 ± 0.5	71.7 ± 0.5
	$\mathcal{W}_{IPW}$	72.1 ± 0.7	**6.1 ± 0.3**	64.8 ± 0.8
PPO	$\mathcal{W}_{\rho+}$	74.6 ± 0.7	9.9 ± 0.8	49.7 ± 2.2
	$\mathcal{W}_{\rho-}$	78.8 ± 0.1	**8.4 ± 0.6**	64.7 ± 0.8
	$\mathcal{W}_{EO}$	**79.2 ± 0.2**	8.5 ± 0.2	**66.0 ± 0.8**
	$\mathcal{W}_{IPW}$	45.8 ± 6.9	10.5 ± 0.9	45.3 ± 5.8
DQN	$\mathcal{W}_{\rho+}$	76.2 ± 1.1	10.4 ± 0.7	57.2 ± 4.8
	$\mathcal{W}_{\rho-}$	**79.3 ± 0.1**	11.1 ± 0.6	65.8 ± 1.4
	$\mathcal{W}_{EO}$	79.2 ± 0.1	**10.1 ± 0.4**	**66.4 ± 0.2**
	$\mathcal{W}_{IPW}$	74.6 ± 0.3	12.8 ± 0.2	56.6 ± 0.3
LinUCB	$\mathcal{W}_{\rho+}$	72.8 ± 0.1	12.0 ± 0.5	54.6 ± 0.9
	$\mathcal{W}_{\rho-}$	74.1 ± 0.4	11.6 ± 0.5	59.3 ± 1.7
	$\mathcal{W}_{EO}$	**74.6 ± 0.2**	12.2 ± 0.5	**59.8 ± 1.1**
	$\mathcal{W}_{IPW}$	37.3 ± 2.5	**10.3 ± 0.7**	35.4 ± 1.0

Table 10: Results with different reward scaling on BiasBios for various algorithms

Algo + *g*	Accuracy ↑	GAP ↓	F1 ↑
Sup^{EO}	**80.2 ± 0.2**	**7.2 ± 0.5**	**71.9 ± 0.7**
LinUCB^{EO}	74.5 ± 0.2	11.7 ± 0.5	59.6 ± 0.8
DQN$^{EO}_{bandit}$	79.2 ± 0.2	10.0 ± 0.5	66.1 ± 0.4
PPO$^{EO}_{bandit}$	79.3 ± 0.1	8.7 ± 0.3	66.1 ± 0.6

Table 11: Results on the BiasBios dataset with explicit gender information added to the context.

Algo + *MP*	Accuracy ↑	GAP ↓	F1 ↑
Sup^{EO}	80.0 ± 0.2	7.4 ± 0.4	71.9 ± 0.3
LinUCB^{EO}	74.3 ± 0.4	11.5 ± 0.1	59.4 ± 1.0
DQN$^{EO}_{bandit}$	79.0 ± 0.2	8.6 ± 0.3	65.8 ± 0.6
PPO$^{EO}_{bandit}$	79.2 ± 0.2	9.7 ± 0.6	66.8 ± 1.4

Table 12: Performance on the BiasBios dataset, using MP debiased embeddings