Uncovering and Mitigating Algorithmic Bias through Learned Latent Structure

Alexander Amini[1,3*], **Ava Soleimany**[2*], **Wilko Schwarting**[1,3], **Sangeeta Bhatia**[3], **and Daniela Rus**[1,3]
[1] Computer Science and Artificial Intelligence Lab, Massachusetts Institute of Technology
[2] Biophysics Program, Harvard University
[3] Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology
[*] Denotes co-first authors
{amini, asolei, wilkos, sbhatia, rus}@mit.edu

Abstract

Recent research has highlighted the vulnerabilities of modern machine learning based systems to bias, especially for segments of society that are under-represented in training data. In this work, we develop a novel, tunable algorithm for mitigating the hidden, and potentially unknown, biases within training data. Our algorithm fuses the original learning task with a variational autoencoder to learn the latent structure within the dataset and then adaptively uses the learned latent distributions to re-weight the importance of certain data points while training. While our method is generalizable across various data modalities and learning tasks, in this work we use our algorithm to address the issue of racial and gender bias in facial detection systems. We evaluate our algorithm on the Pilot Parliaments Benchmark (PPB), a dataset specifically designed to evaluate biases in computer vision systems, and demonstrate increased overall performance as well as decreased categorical bias with our debiasing approach.

1 Introduction

Machine learning (ML) systems are increasingly making decisions that impact the daily lives of individuals and society in general. For example, ML and artificial intelligence (AI) are already being used to determine if a human is eligible to receive a loan (Khandani, Kim, and Lo 2010), how long a criminal should spend in prison (Berk, Sorenson, and Barnes 2016), the order in which a person is presented the news (Nalisnick et al. 2016), or even diagnoses and treatments for medical patients (Mazurowski et al. 2008).

The development and deployment of fair and unbiased AI systems is crucial to prevent any unintended side effects and to ensure the long-term acceptance of these algorithms (Miller 2015; Courtland 2018). Even the seemingly simple task of facial recognition (Zafeiriou, Zhang, and Zhang 2015) has been shown to be subject to extreme amounts of algorithmic bias among select demographics (Buolamwini and Gebru 2018). For example, (Klare et al. 2012) analyzed the face detection system used by the US law enforcement and discovered significantly lower accuracy among dark women between the age of 18-30 years old. This is especially concerning since these facial recognition systems are often not deployed in isolation but rather as part of a larger surveillance or criminal detection pipeline (Abdullah et al. 2017).

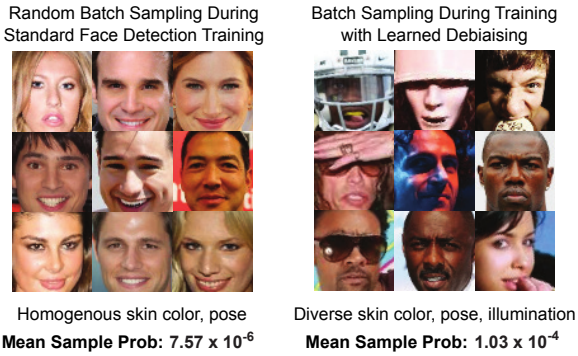

Figure 1: **Batches sampled for training without (left) and with (right) learned debiasing.** The proposed algorithm identifies, in an unsupervised manner, under-represented parts of training data and subsequently increases their respective sampling probability. The resulting batch (right) from the CelebA dataset shows increased diversity in features such as skin color, illumination, and occlusions.

While deep learning based systems have been shown to achieve state-of-the-art performance on many of these tasks, it has also been demonstrated that algorithms trained with biased data lead to algorithmic discrimination (Bolukbasi et al. 2016; Caliskan, Bryson, and Narayanan 2017). Recently, benchmarks quantifying discrimination (Kilbertus et al. 2017; Hardt et al. 2016) and even datasets designed to evaluate the fairness of these algorithms (Buolamwini and Gebru 2018) have emerged. However, the problem of severely imbalanced training datasets and the question of how to integrate debiasing capabilities into AI algorithms still remain largely unsolved.

In this paper, we tackle the challenge of integrating debiasing capabilities directly into a model training process that adapts automatically and without supervision to the shortcomings of the training data. Our approach features an end-to-end deep learning algorithm that simultaneously learns the desired task (e.g., facial detection) as well as the underlying latent structure of the training data. Learning the latent distributions in an unsupervised manner enables us to uncover hidden or implicit biases within the training data. Our algorithm, which is built on top of a variational autoencoder

(VAE), is capable of identifying under-represented examples in the training dataset and subsequently increases the probability at which the learning algorithm samples these data points (Fig. 1).

We demonstrate how our algorithm can be used to debias a facial detection system trained on a biased dataset and to provide interpretations of the learned latent variables which our algorithm actively debiases against. Finally, we compare the performance of our debiased model to a standard deep learning classifier by evaluating racial and gender bias on the Pilot Parliaments Benchmark (PPB) dataset (Buolamwini and Gebru 2018).

The key contributions of this paper can be summarized as:

1. A novel, tunable debiasing algorithm which utilizes learned latent variables to adjust the respective sampling probabilities of individual data points while training; and
2. A semi-supervised model for simultaneously learning a debiased classifier as well as the underlying latent variables governing the given classes; and
3. Analysis of our method for facial detection with biased training data, and evaluation on the PPB dataset to measure algorithmic fairness across race and gender.

The remainder of this paper is structured as follows: we summarize the related work in Sec. 2, formulate the model and debiasing algorithm in Sec. 3, describe our experimental results in Sec. 4, and provide concluding remarks in Sec. 5.

2 Related Work

Interventions that seek to introduce fairness into machine learning pipelines generally fall into one of three categories: those that use data pre-processing before training, in-processing during training, and post-processing after training. Several pre-processing and in-processing methods rely on new, artificially generated debiased data (Calmon et al. 2017) or resampling (More 2016). However, these approaches have largely focused on class imbalances, rather than variability within a class, and fail to use any information about the structure of the underlying latent features. Learning the latent structure of data has a long standing history in machine learning, including Expectation-Maximization (Bailey, Elkan, and others 1994), topic modelling (Blei 2012), latent-SVM (Felzenszwalb, McAllester, and Ramanan 2008), and more recently, variational autoencoders (VAE) (Kingma and Welling 2013; Rezende, Mohamed, and Wierstra 2014). The presented work uses a novel VAE-based approach for resampling based on the data's underlying latent structure, debiases automatically during training, and does not require any data pre-processing or annotation prior to training or testing.

Resampling for class imbalance: Resampling approaches have largely focused on addressing class imbalances (More 2016; Zhou and Liu 2006), as opposed to biases within individual classes. For example, duplicating instances of the minority class as in (Lu, Guo, and Feldkamp 1998) has been used as pre-processing steps for mitigating class imbalance, yet is not capable of running adaptively during training itself. Further, applying these approaches to debiasing variabilities *within* a class would require *a priori* knowledge of the latent structure to the data, which necessitates manual annotation of the desired features. On the other hand, our approach debiases variability within a class automatically during training and learns the latent structure from scratch in an unsupervised manner.

Generating debiased data: Recent approaches have utilized generative models (Sattigeri et al. 2018) and data transformations (Calmon et al. 2017) to generate training data that is more 'fair' than the original dataset. For example, (Sattigeri et al. 2018) used a generative adversarial network (GAN) to output a reconstructed dataset similar to the input but more fair with respect to certain attributes. Preprocessing data transformations that mitigate discrimination, as in (Calmon et al. 2017), have also been proposed, yet such methods are not learned adaptively during training nor do they provide realistic training examples. In contrast to these works, we do not rely on artificially generated data, but rather use a resampled, more representative subset of the original dataset for debiasing.

Clustering to identify bias: Supervised learning approaches have also been used to characterize biases in imbalanced data sets. Specifically, k-means clustering has been employed to identify clusters in the input data prior to training and to inform resampling the training data into a smaller set of representative examples (Nguyen, Bouzerdoum, and Phung 2008). However, this method does not extend to high dimensional data like images or to cases where there is no notion of a data 'cluster', and relies on significant preprocessing. Our proposed approach overcomes these limitations by learning the latent structure using a variational approach.

3 Methodology

Problem Setup

Consider the problem of binary classification in which we are presented with a set of paired training data samples $\mathcal{D}_{train} = \{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\}_{i=1}^{n}$ consisting of features $\boldsymbol{x} \in \mathbb{R}^m$ and labels $\boldsymbol{y} \in \mathbb{R}^d$. Our goal is to find a functional mapping $f : \boldsymbol{X} \rightarrow \boldsymbol{Y}$ parameterized by $\boldsymbol{\theta}$ which minimizes a certain loss $\mathcal{L}(\boldsymbol{\theta})$ over our entire training dataset. In other words, we seek to solve the following optimization problem:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_i(\boldsymbol{\theta}) \tag{1}$$

Given a new test example, $(\boldsymbol{x}, \boldsymbol{y})$, our classifier should ideally output $\hat{\boldsymbol{y}} = f_{\boldsymbol{\theta}}(\boldsymbol{x})$ where $\hat{\boldsymbol{y}}$ is "close" to $\boldsymbol{y}$, with the notion of closeness being defined from the original loss function. Now, assume that each datapoint also has an associated continuous latent vector $z \in \mathbb{R}^k$ which captures the hidden, sensitive features of the sample (Zemel et al. 2013). We can formalize the notion of a biased classifier as follows:

Definition 1 *A classifier, $f_{\boldsymbol{\theta}}(x)$, is **biased** if its decision changes after being exposed to additional sensitive feature inputs. In other words, a classifier is fair with respect to a set of latent features, z, if: $f_{\boldsymbol{\theta}}(x) = f_{\boldsymbol{\theta}}(x, z)$.*

For example, when deciding if an image contains a face or not, the skin color, gender, or even age of the individual

are all underlying latent variables and should not impact the classifier's decision.

To ensure fairness of a classifier across these various latent variables, the dataset should contain roughly uniform samples over the latent space. In other words, the training distribution itself should not be biased to overrepresent a certain category while under-representing others. Note that this is different than claiming that our dataset should be balanced with respect to the classes (i.e., include roughly the same number of faces as non-faces in the dataset). Namely, we are saying that *within* a single class the unobserved latent variables should also be balanced. This would promote the notion that all instances of a single class will be treated fairly by the classifier such that even if a latent variable was changed to the opposite extreme (e.g., skin tone from light to dark) the accuracy of the classifier would not be changed.

Furthermore, given a labeled test set across the space of sensitive latent variables, z, we can measure the bias of the classifier by computing its accuracy across each of the sensitive categories (e.g. skin tone). While the overall accuracy of the classifier is the mean accuracy over all sensitive categories, the bias is the variance in accuracies across all realizations of these categories (e.g., light vs. dark faces). For example, if a classifier performs equally well no matter the realization of a specific latent variable (e.g., skin tone), it will have zero variance in accuracy, and thus be called unbiased with respect to that variable. On the other hand, if some realizations of the latent variable cause the classifier to perform better or worse, the variance in the accuracies will increase, and thus, so will the overall bias of said classifier.

While it is possible to use a set of human defined sensitive variables to ensure fair representation during training, this requires time intensive manned annotation of each variable over the entire dataset. Additionally, this approach is subject to potential human bias in the selection of which variables are deemed sensitive or not. In this work, we address this problem by learning the latent variables of the class in an entirely unsupervised manner and proceed to use these learned variables to adaptively resample the dataset while training. In the following subsection, we will outline the architecture used to learn the latent variables.

Learning Latent Structure with Variational Autoencoders

In this work, we *learn* the latent variables of the class in an entirely unsupervised manner and proceed to use these to adaptively resample the dataset while training. To accomplish this, we propose an extension of the variational autoencoder (VAE) network architecture: a debiasing-VAE (DB-VAE). The encoder portion of the VAE learns an approximation $q_\phi(z|x)$ of the true distribution of the latent variables given a data point. As opposed to classical VAE architectures, we also introduce d additional output variables where $\hat{y} \in \mathbb{R}^d$. With k latent variables and d output variables, the encoder outputs $2k+d$ activations corresponding to $\boldsymbol{\mu} \in \mathbb{R}^k$, $\boldsymbol{\Sigma} = Diag[\sigma^2] \succ 0$, which are used to define the distribution of z, and the d-dimensional output, $\hat{\boldsymbol{y}}$.

Note that, in order to still learn our original supervised learning task we assign and explicitly supervise the d output variables. This, in turn, transforms our traditional VAE model from an entirely unsupervised model to a semi-supervsied model, where some latent variables are implicitly learned by trying to reconstruct the input and the others are explicitly supervised for a specific task (e.g. classification). For example, if we originally wanted to train a binary classifier (i.e., $\hat{\boldsymbol{y}} \in \{0, 1\}$), our DB-VAE model would learn a latent encoding of k latent variables (i.e., $\{z_i\}_{i\in\{1,k\}}$) as well as a single variable specifically for classification: $z_0 = \hat{y}$.

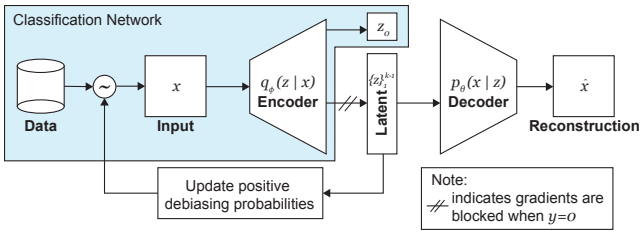

Figure 2: **Debiasing Variational Autoencoder.** Architecture of the semi-supervised DB-VAE for binary classification (blue region). The unsupervised latent variables are used to adaptively resample the dataset while training.

A decoder network mirroring the encoder is then used to reconstruct the input back from the latent space by approximating $p_\theta(x|z)$. VAEs utilize reparameterization to differentiate the outputs through a sampling step, where we sample $\epsilon \sim \mathcal{N}(0, (I))$ and compute $\boldsymbol{z} = \boldsymbol{\mu}(x) + \boldsymbol{\Sigma}^{\frac{1}{2}}(x) \circ \epsilon$. This decoded reconstruction enables unsupervised learning of the latent variables during training, and is thus necessary for automated debiasing of the data during training.

We train the network end-to-end using backpropagation with a three component loss function comprising of a supervised latent loss, a reconstruction loss, and a latent loss for the unsupervised variables. For a binary classification task, for example, the supervised loss $\mathcal{L}_y(y, \hat{y})$ is given by the cross-entropy loss; the reconstruction loss $\mathcal{L}_x(x, \hat{x})$ is given by the L_p norm between the input and the reconstructed output; and the latent loss $\mathcal{L}_{KL}(\mu, \sigma)$ is given by the Kullback-Liebler (*KL*) divergence. Finally, the total loss is a weighted combination of these three losses:

$$\mathcal{L}_{TOTAL} = c_1 \underbrace{\left[\sum_{i\in\{0,1\}} y_i \log\left(\frac{1}{\hat{y}_i}\right)\right]}_{\mathcal{L}_y(y,\hat{y})} + c_2 \underbrace{\left[\|x - \hat{x}\|_p\right]}_{\mathcal{L}_x(x,\hat{x})} + c_3 \underbrace{\left[\frac{1}{2}\sum_{j=0}^{k-1}(\sigma_j + \mu_j^2 - 1 - \log(\sigma_j))\right]}_{\mathcal{L}_{KL}(\mu,\sigma)} \quad (2)$$

where c_1, c_2, c_3 are the weighting coefficients to impact the relative importance of each of the individual loss functions. For comparison, the baseline model used for the desired task has a similar architecture as the DB-VAE, without the unsupervised latent variables and decoder network, and would be trained according to only the supervised loss function.

Note that special care needs to be taken when feeding training examples from classes which you do not want to debias. For example, in the facial detection problem, we primarily care about ensuring that our positive dataset of faces is fair and unbiased, and less about debiasing the negative example where there is no face present. For these negative samples, the gradients from the decoder and latent space should be stopped and not backpropogated. This effectively means that, for these classes, we only train the encoder to improve the supervised loss.

Algorithm for Automated Debiasing

In this section, we present the algorithm for adaptive re-sampling of the training data based on the latent structure learned by our DB-VAE model. By dropping over-represented regions of the latent space according to their frequency of occurrence, we increase the probability of selecting rarer data for training. This is done adaptively as the latent variables themselves are being learned during training. Thus, our debiasing approach accounts for the complete distribution of the underlying features in the training data.

The training dataset is fed through the encoder network, which provides an estimate $\mathcal{Q}(\boldsymbol{z}|X)$ of the latent distribution. We seek to increase the relative frequency of rare data points by increased sampling of under-represented regions of the latent space. To do so, we approximate the distribution of the latent space with a histogram $\hat{\mathcal{Q}}(\boldsymbol{z}|X)$ with dimensionality defined by the number of latent variables, k. To circumvent the high-dimensionality of the histogram when the latent space becomes increasingly complex, we simplify further and use independent histograms to approximate the joint distribution. Specifically, we define an independent histogram, $\hat{\mathcal{Q}}_i(\boldsymbol{z}_i|X)$, for each latent variable $\boldsymbol{z}_i$:

$$\hat{\mathcal{Q}}(\boldsymbol{z}|X) \propto \prod_i \hat{\mathcal{Q}}_i(\boldsymbol{z}_i|X) \tag{3}$$

This allows us to neatly approximate $\mathcal{Q}(\boldsymbol{z}|X)$ based on the frequency distribution of each of the learned latent variables. Finally, we introduce a single parameter, α, to tune the degree of debiasing introduced during training. We define the probability distribution of selecting a datapoint $\boldsymbol{x}$ as $\mathcal{W}(\boldsymbol{z}(\boldsymbol{x})|X)$, parameterized by the debiasing parameter α:

$$\mathcal{W}(\boldsymbol{z}(\boldsymbol{x})|X) \propto \prod_i \frac{1}{\hat{\mathcal{Q}}_i(\boldsymbol{z}_i(\boldsymbol{x})|X) + \alpha} \tag{4}$$

We provide pseudocode for training the DB-VAE in Algorithm 1. At every epoch all inputs x from the original dataset X are propagated through the model to evaluate the corresponding latent variables $z(x)$. The histograms $\hat{Q}_i(z_i(x)|X)$ are updated accordingly. During training, a new batch is drawn by keeping inputs, x, from the original dataset, X, with likelihood $W(z(x)|X)$. Training on the debiased data batch now forces the classifier into a choice of parameters that work better in rare cases without strong deterioration of performance for common training examples. Most importantly, the debiasing is not manually specified beforehand but instead based on *learned* latent variables.

Algorithm 1 Adaptive re-sampling for automated debiasing of the DB-VAE architecture

Require: Training data $\{X, Y\}$, batch size b
Initialize weights $\{\phi, \theta\}$
for each epoch, E_t **do**
  Sample $z \sim q_\phi(z|X)$
  Update $\hat{\mathcal{Q}}_i(z_i(x)|X)$
  $\mathcal{W}(z(x)|X) \leftarrow \prod_i \frac{1}{\hat{\mathcal{Q}}_i(z_i(x)|X)+\alpha}$
  while $iter < \frac{n}{b}$ **do**
    Sample $\boldsymbol{x}_{batch} \sim \mathcal{W}(z(x)|X)$
    $L(\phi, \theta) \leftarrow \frac{1}{b} \sum_{i \in \boldsymbol{x}_{batch}} \mathcal{L}_i(\phi, \theta)$
    Update: $[w \leftarrow w - \eta \nabla_{\phi,\theta} \mathcal{L}(\phi, \theta)]_{w \in \{\phi,\theta\}}$
  end while
end for

Intuitively, the parameter α as tuning the degree of debiasing. As $\alpha \to 0$, the subsampled training set will tend towards uniform over the latent variables z. As $\alpha \to \infty$, the subsampled training set will tend towards a random uniform sample of the original training dataset (i.e., no debiasing).

4 Experiments

To validate our debiasing algorithm on a real-world problem with significant social impact, we learn a debiased facial detector using potentially biased training data. Here we define the facial detection problem, describe the datasets used, and outline model training, debiasing, and evaluation.

For the facial detection problem, we are given a set of paired training data samples $\mathcal{D}_{train} = \{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\}_{i=1}^n$, where $\boldsymbol{x}^{(i)}$ are the raw pixel values of an image patch and $\boldsymbol{y}^{(i)} \in \{0, 1\}$ are their respective labels, indicating the presence of a face. Our goal is to ensure that the set of positive examples used to train a facial detection classifier is fair and unbiased. The positive training data may potentially be biased with respect to certain attributes such as skin tone, in that particular instances of those attributes may appear more or less frequently than other instances. Thus, in our experiments, we train a full DB-VAE model to learn the latent structure underlying the positive (face) images and use the adaptive resampling approach outlined in Algorithm 1 to debias the model with respect to facial features. For negative examples, we only train the encoder portion of our network, as described in Section 3. We evaluate the performance of our debiased models relative to standard, biased classifiers on the PPB dataset and provide estimates of the precision and bias of each model as performance metrics.

Datasets

We train our classifiers on a dataset of $n = 4 \times 10^5$ images, consisting of 2×10^5 positive (images of faces) and negative (images of non-faces) examples, split 80% and 20% into training and validation sets, respectively. Positive examples were taken from the CelebA dataset (Liu et al. 2015) and cropped to a square based on the annotated face bounding box. Negative examples were taken from the ImageNet dataset (Deng et al. 2009), from a wide variety of non-human

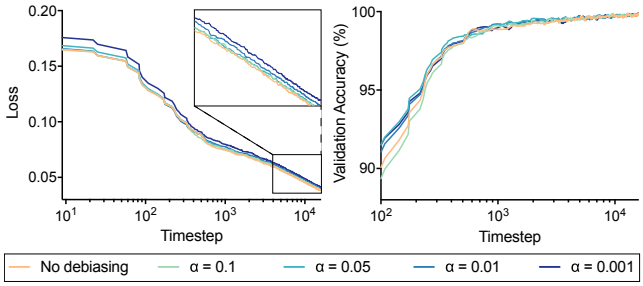

Figure 3: **Loss evolution and validation accuracy.** Convergence of the total loss on the training set (left) and classification accuracy on the validation set (right) for models with varying degrees of debiasing.

categories. All images were resized to 64×64.

After training, we evaluate our debiasing algorithm on the PPB test dataset (Buolamwini and Gebru 2018), which consists of images of 1270 male and female parliamentarians from various African and European countries. Images are consistent in pose, illumination, and facial expression, and the dataset exhibits parity in both skin tone and gender. The gender of each face is annotated with the sex-based "Male" and "Female" labels. Skin tone annotations are based on the Fitzpatrick skin type classification system (Fitzpatrick 1988), with each image labeled as "Lighter" or "Darker".

Training the Models

For the classical facial detection task, we train a convolutional neural network, with four sequential convolutional layers (5×5 filters with 2×2 strides) for feature extraction. Final classification is done with an additional two fully connected layers with 1000 and 1 hidden neurons in each layer respectively. All layers in the network use ReLU activation and batch normalization (Ioffe and Szegedy 2015). Our DB-VAE architecture shares this same classification network for the encoder, except for the final fully connected layer which now outputs an additional k latent variables for a total of $2k + 1$ activations. A decoder, which mirrors the encoder with 2 fully connected layers and 4 de-convolutional layers, is then used to reconstruct the original input image. We train our models by minimizing the empirical training loss as defined in Eq. 2 with $L2$ reconstruction loss.

In our experiments, we additionally block all gradients from the decoder network when $y = 0$, i.e., for negative examples, as we only want to debias for positive face examples. In addition to training the standard classification network with no debiasing, we trained DB-VAE models with varying degrees of debiasing, defined by the parameter α, for 50 epochs and evaluated their performance on the validation set. Models were re-trained from scratch 5 times each for added statistical robustness of results.

Automated Debiasing of Facial Detection Systems

We explore the output of the debiasing algorithm and provide extensive evaluation of our learned models on the PPB dataset. We consider the resampling probabilities, $\mathcal{W}(\boldsymbol{z}(\boldsymbol{x})|X)$, that arise from learning a debiased model. As shown in Fig. 4A, as the probability of resampling increases, the number of data points within the corresponding bin decreases, suggesting that those images more likely to be resampled are those characterized by 'rare' features.

Indeed, as the probability of resampling increases, the corresponding images become more diverse, as evidenced by the four sample faces from each frequency bin in Fig. 4A. This observation is further validated by considering the ten faces in the training data with the lowest and highest resampling probabilities (Fig. 4B,C respectively). The ten faces with the lowest resampling probability appear quite uniform, with consistent skin tone, hair color, forward gaze, and background color. In contrast, the ten faces with the highest resampling probability display rarer features such as headwear or eyewear, tilted gaze, shadowing, and darker skin. Taken together, these results imply that our algorithm identifies and then actively resamples those data points with rarer, more diverse features based on a learned latent representation.

We observed that the DB-VAE is able to learn facial features such as skin tone, presence of hair, and azimuth, as well as other features such as gender and age by slowly perturbing the value of a single latent variable and and feeding the resulting encoding through the decoder (Fig. 5A). This supports the hypothesis that our DB-VAE algorithm is capable of debiasing against such features since the resampling probabilities are directly defined based on the probability distributions of *individual* learned latent variables (Alg. 1).

To evaluate the performance of our debiasing approach, we utilized classification accuracy (positive predictive value) as a metric, and tested our models on the PPB dataset. For this evaluation, we extracted patches from each image using sliding windows of varying dimension, and fed these extracted image patches to our trained models. We output a positive match of a face if the classifier identifies a face in

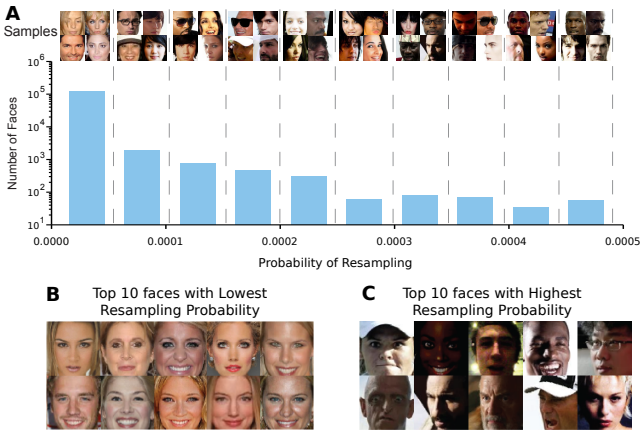

Figure 4: **Sampling probabilities over the training dataset.** Histogram over resampling probabilities showing four samples from each bin (**A**). The top ten faces with the lowest (**B**) and highest (**C**) probabilities of being sampled.

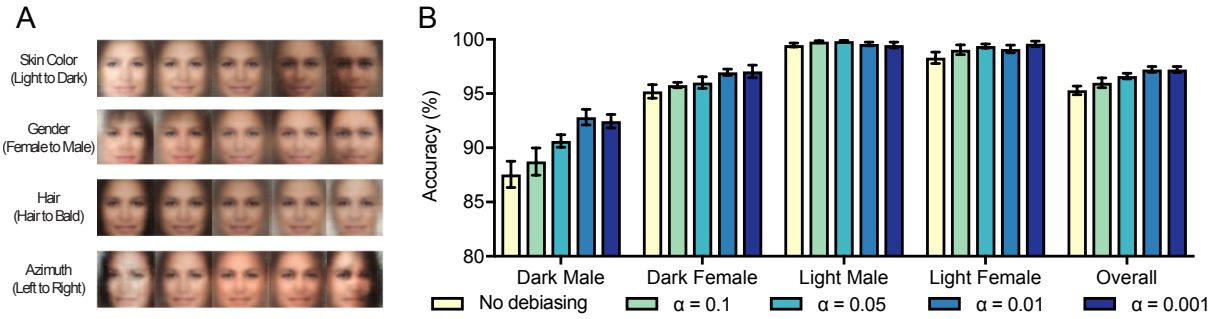

Figure 5: **Increased performance and decreased categorical bias with DB-VAE.** The model learns latent features such as skin color, gender, hair (**A**) and demonstrates increased performance and decreased categorical bias with learned debiasing (**B**).

any one of the subpatches within the image.

To demonstrate debiasing against specific latent features, we quantified classification performance on individual demographics. Specifically, we considered skin tone (light/dark) and gender (male/female). We denote $\mathcal{A}$ as the set of classification accuracies of a model on each of the four intersectional classes. We compared the accuracy of models trained with and without debiasing on both individual demographics (race/gender) and the PPB dataset as a whole, and provide results on the effect of the debiasing parameter α on performance (Fig. 5). Recall that no debiasing corresponds to the limit $\alpha \to \infty$, where we uniformly sample over the *original training set* without learning the latent variables. Conversely, $\alpha \to 0$, corresponds to sampling from a uniform distribution over *the latent space*. Error bars (standard error of the mean) are provided to visualize statistical significance of differences between the trained models.

As shown in Fig. 5, greater debiasing power (decrease α) significantly increased classification accuracy on "Dark Male" subjects, consistent with the hypothesis that adaptive resampling of rare instances (e.g., dark faces) in the training data results in less algorithmic discrimination. This suggests that our algorithm can debias for a qualitative feature like skin tone, which has significant social implications for its utility in improving fairness in facial detection systems.

In contrast to the trend observed with dark male faces, the classification accuracy on "Light Male" faces remained nearly constant for both the biased and debiased models. Additionally, the accuracy on light male subjects was higher than the three other groups, consistent with (Buolamwini and Gebru 2018). This suggests that our debiasing algorithm does not sacrifice performance on categories which already have high precision. Importantly, the high, near constant accuracy suggests that an arbitrary classification model trained on the CelebA dataset may be biased towards light male subjects, and further supports the need for approaches that seek to reduce such biases.

Although the DB-VAE improved accuracy on dark males significantly, it never reached the accuracy of light males. Despite the fact that we debias our training data with respect to latent variables such as skin tone, there are inherently fewer examples of dark male faces in our data. Our model is simply limited by infrequency of these examples but we note that increasing the overall size of our training dataset may further mitigate this effect.

We summarize the key trends in overall performance with DB-VAE in Table 1. As confirmed by Fig. 5, the overall precision, $\mathbb{E}[\mathcal{A}]$, increased with increased debiasing power (decreasing α). Additionally, we observed a decrease in the variance in accuracy between categories, indicative of decreased bias with greater debiasing. Together, these results suggest effective debiasing with DB-VAE.

Table 1: **Accuracy and bias on PPB test dataset.**

	$\mathbb{E}[\mathcal{A}]$ (Precision)	$Var[\mathcal{A}]$ (Measure of Bias)
No Debiasing	95.13	28.84
$\alpha = 0.1$	95.84	25.43
$\alpha = 0.05$	96.47	18.08
$\alpha = 0.01$	97.13	9.49
$\alpha = 0.001$	**97.36**	**9.43**

5 Conclusion

In this paper, we propose a novel, tunable debiasing algorithm to adjust the respective sampling probabilities of individual data points while training. By learning the underlying latent variables in an entirely unsupervised manner, we can scale our approach to large datasets and debias for latent features without ever hand labeling them in our training set.

We apply our approach to facial detection to promote algorithmic fairness by reducing hidden biases within training data. Given a biased training dataset, our debiased models show increased classification accuracy and decreased categorical bias across race and gender, compared to standard classifiers. Finally, we provide a concrete algorithm for debiasing as well as an open source implementation of our model.

The development and deployment of fair and unbiased AI systems is crucial to prevent unintended discrimination and to ensure the long-term acceptance of these algorithms. We envision that the proposed approach will serve as an additional tool to promote systematic, algorithmic fairness of modern AI systems.