Supplementary Material

A. Proof of Theorem 1

Proof. Since $p_{\text{bias}}(\mathbf{x}|\mathbf{z}=k)$ and $p_{\text{bias}}(\mathbf{x}|\mathbf{z}=k')$ have disjoint supports for $k \neq k'$, we know that for all $\mathbf{x}$, there exists a deterministic mapping $f : \mathcal{X} \to \mathcal{Z}$ such that $p_{\text{bias}}(\mathbf{x}|\mathbf{z}=f(\mathbf{x})) > 0$.

Further, for all $\tilde{\mathbf{x}} \notin f^{-1}(\mathbf{z})$:

$$p_{\text{bias}}(\tilde{\mathbf{x}}|\mathbf{z}=f(\mathbf{x})) = 0; \tag{9}$$
$$p_{\text{ref}}(\tilde{\mathbf{x}}|\mathbf{z}=f(\mathbf{x})) = 0. \tag{10}$$

Combining Eqs. 21,22 above with the assumption in Eq. 20, we can simplify the density ratios as:

$$\frac{p_{\text{bias}}(\mathbf{x})}{p_{\text{ref}}(\mathbf{x})} = \frac{\int_{\mathbf{z}} p_{\text{bias}}(\mathbf{x}|\mathbf{z})p_{\text{bias}}(\mathbf{z})\mathrm{d}\mathbf{z}}{\int_{\mathbf{z}} p_{\text{ref}}(\mathbf{x}|\mathbf{z})p_{\text{ref}}(\mathbf{z})\mathrm{d}\mathbf{z}} \tag{11}$$
$$= \frac{p_{\text{bias}}(\mathbf{x}|f(\mathbf{x}))p_{\text{bias}}(f(\mathbf{x}))}{p_{\text{ref}}(\mathbf{x}|f(\mathbf{x}))p_{\text{ref}}(f(\mathbf{x}))} \text{ (using Eqs. 21,22)} \tag{12}$$
$$= \frac{p_{\text{bias}}(f(\mathbf{x}))}{p_{\text{ref}}(f(\mathbf{x}))} \text{ (using Eq. 20)} \tag{13}$$
$$= b(f(\mathbf{x})). \tag{14}$$

From Eq. 5 and Eq. 23, the Bayes optimal classifier c^* can hence be expressed as:

$$c^*(Y=1|\mathbf{x}) = \frac{1}{\gamma b(f(\mathbf{x}))+1}. \tag{15}$$

The optimal cross-entropy loss of a binary classifier c for density ratio estimation (DRE) can then be expressed as:

$$NCE(c^*) = \frac{1}{\gamma+1}\mathbb{E}_{p_{\text{ref}}(\mathbf{x})}[\log c^*(Y=1|\mathbf{x})] + \frac{\gamma}{\gamma+1}\mathbb{E}_{p_{\text{bias}}(\mathbf{x})}[\log c^*(Y=0|\mathbf{x})] \tag{16}$$
$$= \mathbb{E}_{p_{\text{ref}}(\mathbf{x})}\left[\log \frac{1}{\gamma b(f(\mathbf{x}))+1}\right] + \frac{\gamma}{\gamma+1}\mathbb{E}_{p_{\text{bias}}(\mathbf{x})}\left[\log \frac{\gamma b(f(\mathbf{x}))}{\gamma b(f(\mathbf{x}))+1}\right] \text{ (using Eq. 24)} \tag{17}$$
$$= \frac{1}{\gamma+1}\mathbb{E}_{p_{\text{ref}}(\mathbf{z})}\mathbb{E}_{p_{\text{ref}}(\mathbf{x}|\mathbf{z})}\left[\log \frac{1}{\gamma b(f(\mathbf{x}))+1}\right] + \frac{\gamma}{\gamma+1}\mathbb{E}_{p_{\text{bias}}(\mathbf{z})}\mathbb{E}_{p_{\text{bias}}(\mathbf{x}|\mathbf{z})}\left[\log \frac{\gamma b(f(\mathbf{x}))}{\gamma b(f(\mathbf{x}))+1}\right] \tag{18}$$
$$= \frac{1}{\gamma+1}\mathbb{E}_{p_{\text{ref}}(\mathbf{z})}\left[\log \frac{1}{\gamma b(\mathbf{z})+1}\right] + \frac{\gamma}{\gamma+1}\mathbb{E}_{p_{\text{bias}}(\mathbf{z})}\left[\log \frac{\gamma b(\mathbf{z})}{\gamma b(\mathbf{z})+1}\right] \text{ (using Eqs. 21,22).} \tag{19}$$

□

The performance of Algorithm 1 critically depends on the quality of estimated density ratios, which in turn is dictated by the training of the binary classifier itself.

To analyze the conditions under which we can learn optimal ratios via a binary classifier, we need a more refined characterization of the dataset bias.

In order to do so, we consider data distributions p_{bias} and p_{ref} that admit an (unknown) many-to-one, deterministic mapping $f : \mathcal{X} \to \mathcal{Z}$ of the input variables $\mathbf{x}$ onto a set of bias variables $Z \in \mathcal{Z}$ such that the conditional distributions over $\mathbf{x}$ match:

$$p_{\text{bias}}(\mathbf{x}|Z=f(\mathbf{x})) = p_{\text{ref}}(\mathbf{x}|Z=f(\mathbf{x})). \tag{20}$$

Since f is assumed to be many-to-one and letting $\mathbf{z} = f(\mathbf{x})$, we also have for all $\tilde{\mathbf{x}} \notin f^{-1}(\mathbf{z})$:

$$p_{\text{bias}}(\tilde{\mathbf{x}}|\mathbf{z}) = 0; \tag{21}$$
$$p_{\text{ref}}(\tilde{\mathbf{x}}|\mathbf{z}) = 0. \tag{22}$$

For example, as we shall see in our experiments, $\mathbf{x}$ can correspond to images, whereas Z represents a subgroup defined via unobserved sensitive bias factors for each image, such as gender, race, etc. that leads to dataset bias.

With the assumption in Eq. 20, we can simplify the density ratios as:

$$\frac{p_{\text{bias}}(\mathbf{x})}{p_{\text{ref}}(\mathbf{x})} = \frac{\int p_{\text{bias}}(\mathbf{x}|\mathbf{z})p_{\text{bias}}(\mathbf{z})\mathrm{d}\mathbf{z}}{\int p_{\text{ref}}(\mathbf{x}|\mathbf{z})p_{\text{ref}}(\mathbf{z})\mathrm{d}\mathbf{z}} = \frac{\cancel{p_{\text{bias}}(\mathbf{x}|f(\mathbf{x}))}p_{\text{bias}}(f(\mathbf{x}))}{\cancel{p_{\text{ref}}(\mathbf{x}|f(\mathbf{x}))}p_{\text{ref}}(f(\mathbf{x}))} := b(f(\mathbf{x})). \tag{23}$$

where $\mathbf{z} = f(\mathbf{x})$ as before and we simplified the integrals based on the assumption that f exists and is many-to-one. From Eq. 5 and Eq. 23, the Bayes optimal classifier c^* can hence be expressed as:

$$c^*(Y = 1|\mathbf{x}) = \frac{\gamma b(f(\mathbf{x}))}{\gamma b(f(\mathbf{x})) + 1}. \tag{24}$$

The optimal negative cross-entropy of a binary classifier c for density ratio estimation (DRE) can then be expressed as:

$$NCE(c^*) := \mathbb{E}_{p_{\text{ref}}(\mathbf{x})}[\log c^*(Y = 1|\mathbf{x})] + \mathbb{E}_{p_{\text{bias}}(\mathbf{x})}[\log c^*(Y = 0|\mathbf{x})] \tag{25}$$

$$= \mathbb{E}_{p_{\text{ref}}(\mathbf{x})}\left[\log \frac{\gamma b(f(\mathbf{x}))}{\gamma b(f(\mathbf{x})) + 1}\right] + \mathbb{E}_{p_{\text{bias}}(\mathbf{x})}\left[\log \frac{1}{\gamma b(f(\mathbf{x})) + 1}\right] \quad \text{(using Eq. 24)} \tag{26}$$

$$= \mathbb{E}_{p_{\text{ref}}(\mathbf{z})}\mathbb{E}_{p_{\text{ref}}(\mathbf{x}|\mathbf{z})}\left[\log \frac{\gamma b(f(\mathbf{x}))}{\gamma b(f(\mathbf{x})) + 1}\right] + \mathbb{E}_{p_{\text{bias}}(\mathbf{z})}\mathbb{E}_{p_{\text{bias}}(\mathbf{x}|\mathbf{z})}\left[\log \frac{1}{\gamma b(f(\mathbf{x})) + 1}\right] \tag{27}$$

$$= \mathbb{E}_{p_{\text{ref}}(\mathbf{z})}\left[\log \frac{\gamma b(\mathbf{z})}{\gamma b(\mathbf{z}) + 1}\right] + \mathbb{E}_{p_{\text{bias}}(\mathbf{z})}\left[\log \frac{1}{\gamma b(\mathbf{z}) + 1}\right] \quad \text{(using Eqs. 21,22).} \tag{28}$$

B. Dataset Details

B.1. Dataset Construction Procedure

We construct such dataset splits from the full CelebA training set using the following procedure. We initially fix our dataset size to be roughly 135K out of the total 162K based on the total number of females present in the data. Then for each level of `bias`, we partition 1/4 of males and 1/4 of females into $\mathcal{D}_{\text{ref}}$ to achieve the 50-50 ratio. The remaining number of examples are used for $\mathcal{D}_{\text{bias}}$, where the number of males and females are adjusted to match the desired level of `bias` (e.g. 0.9). Finally at each level of reference dataset size `perc`, we discard the appropriate fraction of datapoints from both the male and female category in $\mathcal{D}_{\text{ref}}$. For example, for `perc` = 0.5, we discard half the number of females and half the number of males from $\mathcal{D}_{\text{ref}}$.

B.2. FID Calculation

As noted Sections 2.3 and 6, the FID metric may exhibit a relative preference for models trained on larger datasets in order to maximize perceptual sample quality, at the expense of propagating or amplifying existing dataset bias. In order to obtain an estimate of sample quality that would also incorporate a notion of fairness across sensitive attribute classes, we pre-computed the relevant FID statistics on a "balanced" construction of the CelebA dataset that matches our reference dataset p_{ref}. That is, we used all train/validation/test splits of the data such that: (1) for single-attribute, there were 50-50 portions of males and females; and (2) for multi-attribute, there were even proportions of examples across all 4 classes (females with black hair, females without black hair, males with black hair, males without black hair). We report "balanced" FID numbers on these pre-computed statistics throughout the paper.

C. Architecture and Hyperparameter Configurations

We used PyTorch (Paszke et al., 2017) for all our experiments. Our overall experimental framework involved three different kinds of models which we describe below.

C.1. Attribute Classifier

We use the same architecture and hyperparameters for both the single- and multi-attribute classifiers. Both are variants of ResNet-18 where the output number of classes correspond to the dataset split (e.g. 2 classes for single-attribute, 4 classes for the multi-attribute experiment).

Architecture. We provide the architectural details in Table 2 below:

Name	Component
conv1	7×7 conv, 64 filters. stride 2
Residual Block 1	3×3 max pool, stride 2
Residual Block 2	$\begin{bmatrix} 3 \times 3 \text{ conv, } 128 \text{ filters} \\ 3 \times 3 \text{ conv, } 128 \text{ filters} \end{bmatrix} \times 2$
Residual Block 3	$\begin{bmatrix} 3 \times 3 \text{ conv, } 256 \text{ filters} \\ 3 \times 3 \text{ conv, } 256 \text{ filters} \end{bmatrix} \times 2$
Residual Block 4	$\begin{bmatrix} 3 \times 3 \text{ conv, } 512 \text{ filters} \\ 3 \times 3 \text{ conv, } 512 \text{ filters} \end{bmatrix} \times 2$
Output Layer	7×7 average pool stride 1, fully-connected, softmax

Table 2. ResNet-18 architecture adapted for attribute classifier.

Hyperparameters. During training, we use a batch size of 64 and the Adam optimizer with learning rate = 0.001. The classifiers learn relatively quickly for both scenarios and we only needed to train for 10 epochs. We used early stopping with the validation set in CelebA to determine the best model to use for downstream evaluation.

C.2. Density Ratio Classifier

Architecture. We provide the architectural details in Table 2.

Name	Component
conv1	7×7 conv, 64 filters. stride 2
Residual Block 1	3×3 max pool, stride 2
Residual Block 2	$\begin{bmatrix} 3 \times 3 \text{ conv, } 128 \text{ filters} \\ 3 \times 3 \text{ conv, } 128 \text{ filters} \end{bmatrix} \times 2$
Residual Block 3	$\begin{bmatrix} 3 \times 3 \text{ conv, } 256 \text{ filters} \\ 3 \times 3 \text{ conv, } 256 \text{ filters} \end{bmatrix} \times 2$
Residual Block 4	$\begin{bmatrix} 3 \times 3 \text{ conv, } 512 \text{ filters} \\ 3 \times 3 \text{ conv, } 512 \text{ filters} \end{bmatrix} \times 2$
Output Layer	7×7 average pool stride 1, fully-connected, softmax

Table 3. ResNet-18 architecture adapted for attribute classifier.

Hyperparameters. We also use a batch size of 64, the Adam optimizer with learning rate = 0.0001, and a total of 15 epochs to train the density ratio estimate classifier.

Experimental Details. We note a few steps we had to take during the training and validation procedure. Because of the imbalance in both (a) unbalanced/balanced dataset sizes and (b) gender ratios, we found that a naive training procedure encouraged the classifier to predict all data points as belonging to the biased, unbalanced dataset. To prevent this phenomenon from occuring, two minor modifications were necessary:

1. We *balance* the distribution between the two datasets in each minibatch: that is, we ensure that the classifier sees equal numbers of data points from the balanced ($y = 1$) and unbalanced ($y = 0$) datasets for each batch. This provides enough signal for the classifier to learn meaningful density ratios, as opposed to a trivial mapping of all points to the larger dataset.

2. We apply a similar balancing technique when testing against the validation set. However, instead of balancing the minibatch, we weight the contribution of the losses from the balanced and unbalanced datasets. Specifically, the loss is computed as:

$$\mathcal{L} = \frac{1}{2}\left(\frac{\text{acc}_{\text{pos}}}{\text{n}_{\text{pos}}} + \frac{\text{acc}_{\text{neg}}}{\text{n}_{\text{neg}}}\right)$$

where the subscript `pos` denotes examples from the balanced dataset ($y = 1$) and `neg` denote examples from the unbalanced dataset ($y = 0$).

C.3. BigGAN

Architecture. The architectural details for the BigGAN are provided in Table 4.

Generator	Discriminator
$1 \times 1 \times 2ch$ Noise	$64 \times 64 \times 3$ Image
Linear $1 \times 1 \times 16ch \rightarrow 1 \times 1 \times 16ch$ ResBlock up $16ch \rightarrow 16ch$	ResBlock down $1ch \rightarrow 2ch$ Non-Local Block (64×64)
ResBlock up $16ch \rightarrow 8ch$	ResBlock down $2ch \rightarrow 4ch$
ResBlock up $8ch \rightarrow 4ch$	ResBlock down $4ch \rightarrow 8ch$
ResBlock up $4ch \rightarrow 2ch$	ResBlock down $8ch \rightarrow 16ch$
Non-Local Block (64×64) ResBlock up $4ch \rightarrow 2ch$	ResBlock down $16ch \rightarrow 16ch$ ResBlock $16ch \rightarrow 16ch$
BatchNorm, ReLU, 3×3 Conv $1ch \rightarrow 3$	ReLU, Global sum pooling
Tanh	Linear $\rightarrow 1$

Table 4. Architecture for the generator and discriminator. Notation: ch refers to the channel width multiplier, which is 64 for 64×64 CelebA images. ResBlock up refers to a Generator Residual Block in which the input is passed through a ReLU activation followed by two 3×3 convolutional layers with a ReLU activation in between. ResBlock down refers to a Discriminator Residual Block in which the input is passed through two 3×3 convolution layers with a ReLU activation in between, and then downsampled. Upsampling is performed via nearest neighbor interpolation, whereas downsampling is performed via mean pooling. "ResBlock up/down $n \rightarrow m$" indicates a ResBlock with n input channels and m output channels.

Hyperparameters. We sweep over a batch size of $\{16, 32, 64, 128\}$, and the Adam optimizer with learning rate $= 0.0002$, and $\beta_1 = 0, \beta_2 = 0.99$. We train the model by taking 4 discriminator gradient steps per generator step. Because the BigGAN was originally designed for scaling up class-conditional image generation, we fix all conditioning labels for the unconditional baselines (`imp-weight, equi-weight`) to the zero vector.

Additionally, we investigate the role of *flattening* in the density ratios used to train the generative model. As in (Grover et al., 2019), flattening the density ratios via a power scaling parameter $\alpha \geq 0$ is defined as:

$$\mathbb{E}_{\mathbf{x} \sim p_{\text{ref}}[\ell(\mathbf{x}, \theta)]} \approx \frac{1}{T} \sum_{i=1}^{T} w(\mathbf{x}_i)^{\alpha} \ell(\mathbf{x}_i, \theta)$$

where $\mathbf{x}_i \sim p_{\text{bias}}$. We perform a hyperparameter sweep over $\alpha = \{0.5, 1.0, 1.5\}$, while noting that $\alpha = 0$ is equivalent to the `equi-weight` baseline (no reweighting).

D. Density Ratio Classifier Analysis

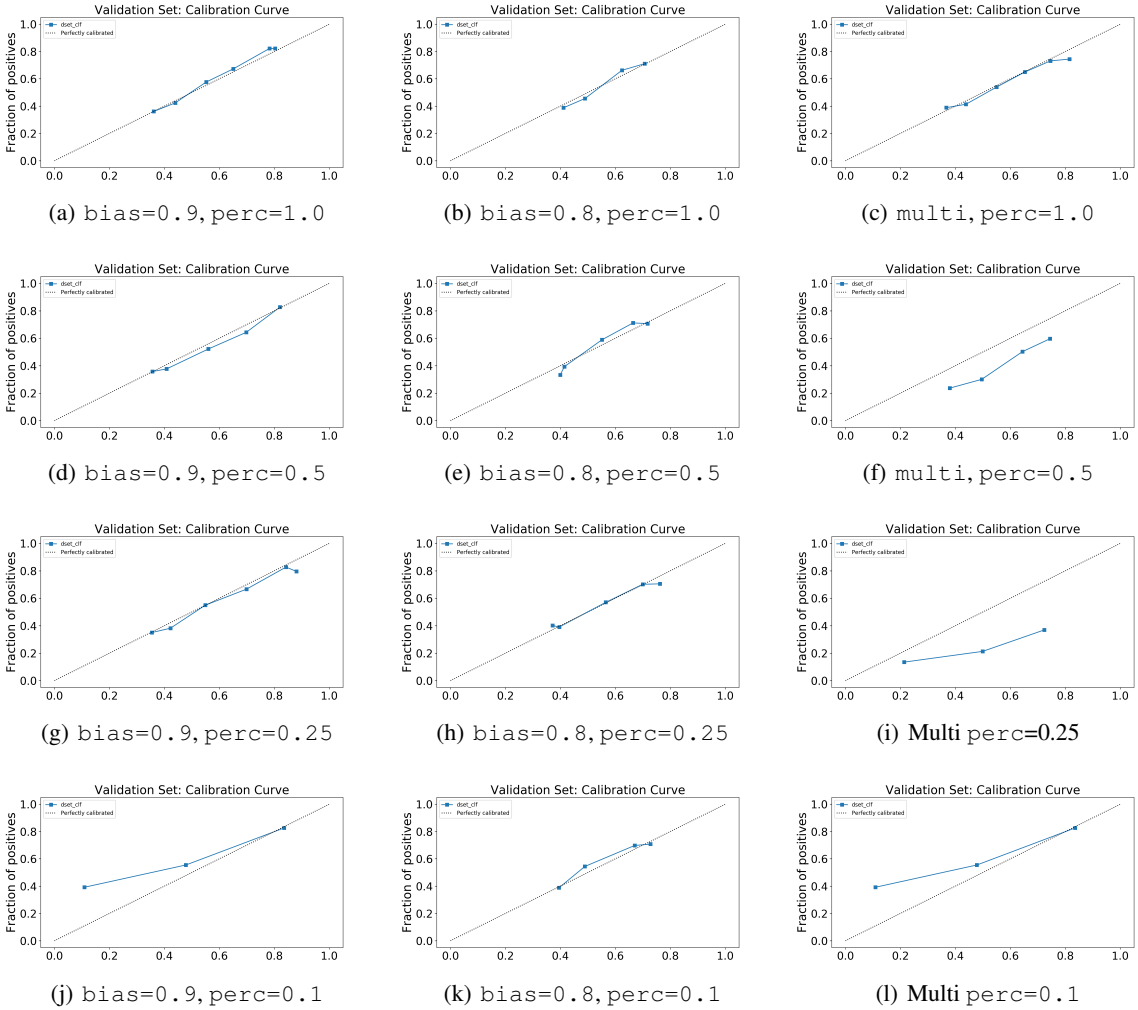
In Figure 5, we show the calibration curves for the density ratio classifiers for each of the $\mathcal{D}_{\text{ref}}$ dataset sizes across all levels of bias. As evident from the plots, most classifiers are already calibrated and did not require any post-training recalibration.

E. Fairness Discrepancy Metric

In this section, we motivate the fairness discrepancy metric and elaborate upon its construction. Recall from Equation 2 that the metric is as follows for the sensitive attributes $\mathbf{u}$:

$$f(p_{\text{ref}}, p_\theta) = |\mathbb{E}_{p_{\text{ref}}}[p(\mathbf{u}|\mathbf{x})] - \mathbb{E}_{p_\theta}[p(\mathbf{u}|\mathbf{x})]|_2.$$

To gain further insight into what the metric is capturing, we rewrite the joint distribution of the sensitive attributes $\mathbf{u}$ and our data $\mathbf{x}$: (1) $p_{\text{ref}}(\mathbf{u}, \mathbf{x}) = p(\mathbf{u}|\mathbf{x})p_{\text{ref}}(\mathbf{x})$ and (2) $p_\theta(\mathbf{u}, \mathbf{x}) = p(\mathbf{u}|\mathbf{x})p_\theta(\mathbf{x})$. Then, marginalizing out $\mathbf{x}$ and only looking at



(a) `bias=0.9, perc=1.0` (b) `bias=0.8, perc=1.0` (c) `multi, perc=1.0`

(d) `bias=0.9, perc=0.5` (e) `bias=0.8, perc=0.5` (f) `multi, perc=0.5`

(g) `bias=0.9, perc=0.25` (h) `bias=0.8, perc=0.25` (i) Multi `perc`=0.25

(j) `bias=0.9, perc=0.1` (k) `bias=0.8, perc=0.1` (l) Multi `perc=0.1`

Figure 5. Calibration curves

the distribution of $\mathbf{u}$, we get that $p(\mathbf{u}) = \int p(\mathbf{u}, \mathbf{x})dx = \int p(\mathbf{u}|\mathbf{x})p(\mathbf{x})dx = \mathbb{E}_p(\mathbf{x})p(\mathbf{u}|\mathbf{x})$. Thus the fairness discrepancy metric is $|p_{\text{ref}}(\mathbf{u}) - p_\theta(\mathbf{u})|_2$.

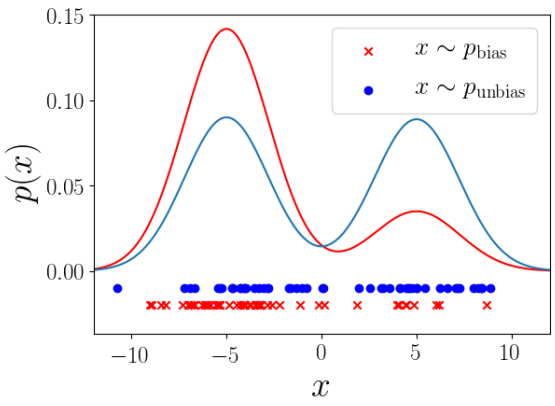
This derivation is informative because it allows us to relate the fairness discrepancy metric to the behavior of the (oracle) attribute classifier. Suppose we use a deterministic classifier $p(\mathbf{u}|\mathbf{x})$ as in the paper: that is, we threshold at 0.5 to label all examples with $p(\mathbf{u}|\mathbf{x}) > 0.5$ as $\mathbf{u} = 1$ (e.g. male), and $p(\mathbf{u}|\mathbf{x}) \leq 0.5$ as $\mathbf{u} = 0$ (e.g. female). In this setting, the fairness discrepancy metric simply becomes the ℓ_2 distance in proportions of different populations between the true (reference) dataset and the generated examples.

It is easy to see that if we use a probabilistic classifier (without thresholding), we can obtain similar distributional discrepancies between the true (reference) data distribution and the distribution learned by p_θ such as the empirical KL.

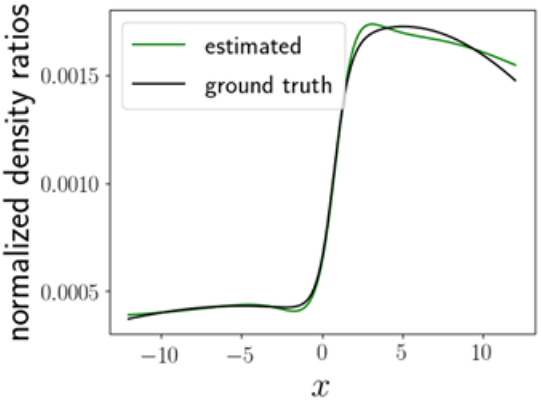
F. Additional Results

F.1. Toy Example with Gaussian Mixture Models

We demonstrate the benefits of our reweighting technique through a toy Gaussian mixture model example. In Figure 6(a), the reference distribution is shown in blue and the biased distribution in red. The blue distribution is an equi-weighted mixture of 2 Gaussians (reference), while the red distribution is a non-uniform weighted mixture of 2 Gaussians (biased).

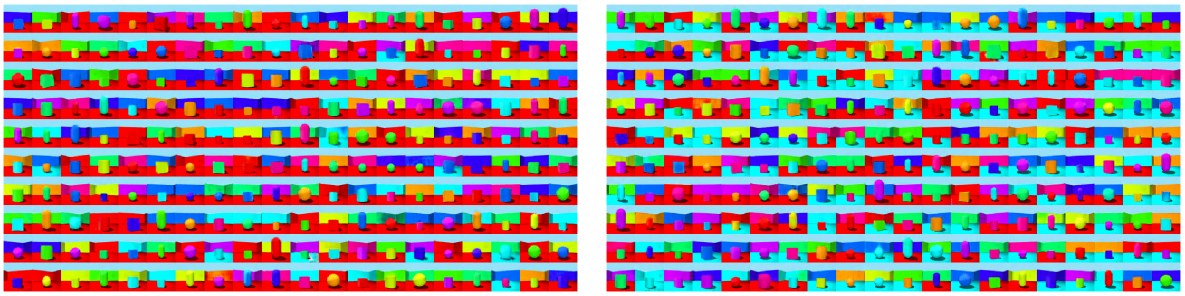


(a) Biased and Reference Distributions



(b) Density Ratios

Figure 6. (a) Comparison between two biased (non-uniform weighted mixture, shown in blue) and reference (equi-weighted Gaussian mixture, shown in red). After the optimal density ratios are estimated using a two-layer MLP, we observe that the estimated density ratios are extremely similar to the ratios output by the Bayes optimal classifier, as desired.



(a) Baseline samples

(b) Samples after reweighting

Figure 7. Results from the Shapes3D dataset. After restricting the possible floor colors to red or blue and using a biased dataset of `bias=0.9`, we find that the samples obtained after importance reweighting (b) are considerably more balanced than those without reweighting (a), as desired.

The weights are 0.9 and 0.1 for the two Gaussians in the biased case. We trained a two layer multi-layer perceptron (MLP) (with tanh activations) to estimate density ratios based on 1000 samples drawn from the two distributions. We then compare the Bayes optimal and estimated density ratios in Figure 6(b), and observe that the estimated density ratios closely trace the ratios output by the Bayes optimal classifier.

F.2. Shapes3D Dataset

For this experiment, we used the Shapes3D dataset (Burgess & Kim, 2018) which is comprised of 480,000 images of shapes with six underlying attributes. We chose a random attribute (floor color), restricted it to two possible instantiations (red vs. blue), and then applied Algorithm 1 in the main text for `bias`=0.9 for this setting. Training on the large biased dataset (containing excess of red floors) induces an average fairness discrepancy of 0.468 as shown in Figure 7(a). In contrast, applying the importance-weighting correction on the large biased dataset enabled us to train models that yielded an average fairness discrepancy of 0.002 as shown in Figure 7(b).

F.3. Downstream Classification Task

We note that although it is difficult to directly compare our model to supervised baselines such as FairGAN (Xu et al., 2018) and FairnessGAN (Sattigeri et al., 2019) due to the unsupervised nature of our work, we conduct further evaluations on a relevant downstream task classification task, adapted to a fairness setting.

In this task, we augment a biased dataset (165K exmaples) with a "fair" dataset (135K examples) generated by a pre-trained GAN to use for training a classifier, then evaluate the classifier's performance on a held-out dataset of true examples. We train a conditional GAN using the AC-GAN objective (Odena et al., 2017), where the conditioning is on an arbitrary downstream attribute of interest (e.g., we consider the attractiveness attribute of CelebA as in (Sattigeri et al., 2019)). Our goal is to learn a fair classifier trained to predict the attribute of interest in a way that is fair with respect to gender, the sensitive attribute.

As an evaluation metric, we use the *demographic parity distance* (Δ_{dp}), denoted as the absolute difference in demographic parity between two classifiers f and g:

$$\Delta_{\text{dp}} = |f_{\text{dp}} - g_{\text{dp}}|$$

We consider 2 AC-GAN variants: (1) `equi-weight` trained on $\mathcal{D}_{\text{bias}} \cup \mathcal{D}_{\text{ref}}$; and (2) `imp-weight`, which reweights the loss by the density ratio estimates. The classifier is trained on both real and generated images for both AC-GAN variants, with the labels given by the conditioned attractiveness values for the respective generations. The classifier is then asked to predict attractiveness for the CelebA test set.

As shown in Table 5, we find that the classifier trained on both real data and synthetic data generated by our `imp-weight` AC-GAN achieved a much lower Δ_{dp} than the `equi-weight` baseline, demonstrating that our method achieves a higher demographic parity with respect to the sensitive attribute, despite the fact that we did not explicitly use labels during training.

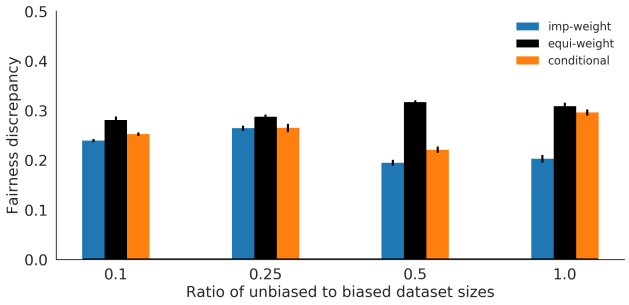
Model	**Accuracy**	**NLL**	Δ_{dp}
Baseline classifier, no data augmentation	79%	0.7964	0.038
`equi-weight`	**79%**	0.7902	0.032
`imp-weight` (ours)	75%	**0.7564**	**0.002**

Table 5. For the CelebA dataset, classifier accuracy, negative log-likelihood, and Δ_{dp} across `bias` = 0.9 and `perc`=1.0 on the downstream classification task. Our importance-weighting method learns a fair classifier that achieves a lower Δ_{dp}, as desired, albeit with a slight reduction in accuracy.

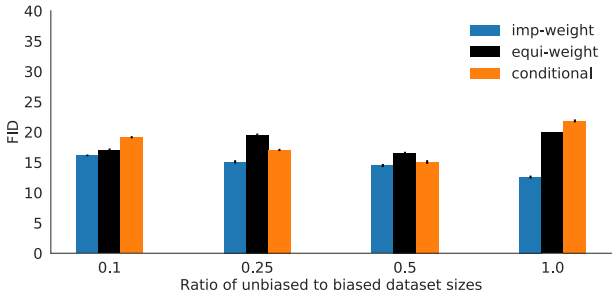
F.4. Single-Attribute Experiment

The results for the single-attribute split for `bias=0.8` are shown in Figure 8.

(a) Samples generated via importance reweighting. Faces above orange line classified as female (55/100) while rest as male.



(b) Fairness Discrepancy



(c) FID

Figure 8. Single Attribute Dataset Bias Mitigation for `bias`=0.8. Standard error in (b) and (c) over 10 independent evaluation sets of 10,000 samples each drawn from the models. Lower fairness discrepancy and FID is better. We find that on average, `imp-weight` outperforms the `equi-weight` baseline by 23.9% and the `conditional` baseline by 12.2% across all reference dataset sizes for bias mitigation.

G. Additional generated samples

Additional samples for other experimental configuration are displayed in the following pages.

(a) `equi-weight`

(b) `conditional`

(c) `imp-weight`

Figure 9. Additional samples of `bias=0.9`, across different methods. All samples shown are from the scenario where $|\mathcal{D}_{\text{ref}}| = |\mathcal{D}_{\text{bias}}|$.

(a) `equi-weight`

(b) `conditional`

(c) `imp-weight`

Figure 10. Additional samples of `bias=0.8`, across different methods. All samples shown are from the scenario where $|\mathcal{D}_{\text{ref}}| = |\mathcal{D}_{\text{bias}}|$.

(a) `equi-weight`

(b) `conditional`

(c) `imp-weight`

Figure 11. Additional samples of the multi-attribute experiment, across different methods. All samples shown are from the scenario where $|\mathcal{D}_{\text{ref}}| = |\mathcal{D}_{\text{bias}}|$.