Appendix For Debiasing Pretrained Generative Models by Uniformly Sampling Semantic Attributes

Anonymous Author(s)
Affiliation
Address
email

A Appendix

A.1 Corrections

We unfortunately had an error in Definition 3, and in Figure 5.

A.1.1 Correction for Definition 3

We had an error in the subscripts of the summations in Definition 3. The statement should have been:

Definition ($\mathbb{P}_E^\lambda$)**.** *Define* $\mathbb{P}_E^\lambda = \sum_{i=1}^{|\mathcal{Y}|} \lambda_i E_{:,i}$ *as a distribution over* $\mathcal{Y}$ *determined by prediction-conditional error matrix* E *and* $\lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_{|\mathcal{Y}|}\}$, $\lambda_i \in \mathbb{R}_{\geq 0}$, $\sum_{i=1}^{|\mathcal{Y}|} \lambda_i = 1$.

A.1.2 Correction for Figure 5

We incorrectly transposed the Antimode and Mode Polarity Sampling results in this figure. Corrected figure is shown below.

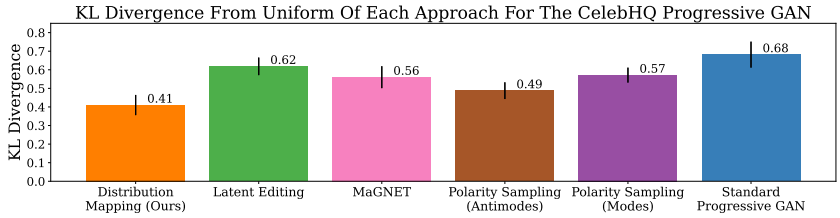

Figure 1: KL Divergence between the distribution over the semantic space for the output of each method (*lower is better*).

A.2 Calculating E for The Generated Distribution

The error rates reported for a classifier C_ϕ are typically reported on the distribution on the distribution of fit's training data, $\mathbb{P}_{training}$. However, the distribution $\mathbb{P}_{G_\theta}$ of the generative model G_θ may differ from the training distribution. Additionally, rather than reporting $P(\mathbf{y}|\hat{\mathbf{y}})$, often times the error rates are given in a confusion matrix $C_{\hat{\mathbf{y}}|\mathbf{y}}$ where $C_{\hat{\mathbf{y}}|\mathbf{y}}[i,j] = P(\hat{\mathbf{y}}|\mathbf{y})$. Thankfully, we can construct the error rate matrix E for the generative distribution $\mathbb{P}_{G_\theta}$ under the simplifying assumption that the difference between $\mathbb{P}_{G_\theta}$ and $\mathbb{P}_{training}$ can be explained as a label shift [1, 3].

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

By Bayes' Theorem, we know that

$$P(\mathbf{y}|\hat{\mathbf{y}}) = P(\hat{\mathbf{y}}|\mathbf{y})\frac{P(\mathbf{y})}{P(\hat{\mathbf{y}})}.$$

Under the label shift assumption, $P(\hat{\mathbf{y}}|\mathbf{y})$ stays the same between $\mathbb{P}_{training}$ and $\mathbb{P}_{G_\theta}$. Additionally, $P(\mathbf{y})$ can be calculated for $\mathbb{P}_{G_\theta}$ under label shift [1, 3]. Lastly, $P(\hat{\mathbf{y}})$ can be approximated for $\mathbb{P}_{G_\theta}$ by finding the proportion predicted for each class on a large sample from the generative model. Thus, E can be calculated as:

$$E = C_{\hat{\mathbf{y}}|\mathbf{y}}\frac{P_{G_\theta}(\mathbf{y})}{P_{G_\theta}(\hat{\mathbf{y}})}.$$

A.3 Distribution of Races Generated By Progressive GAN

We show the two best performing methods' distributions on Progressive GAN, along with the distribution of the unmodified ProgressiveGAN, over the `Race` attribute.

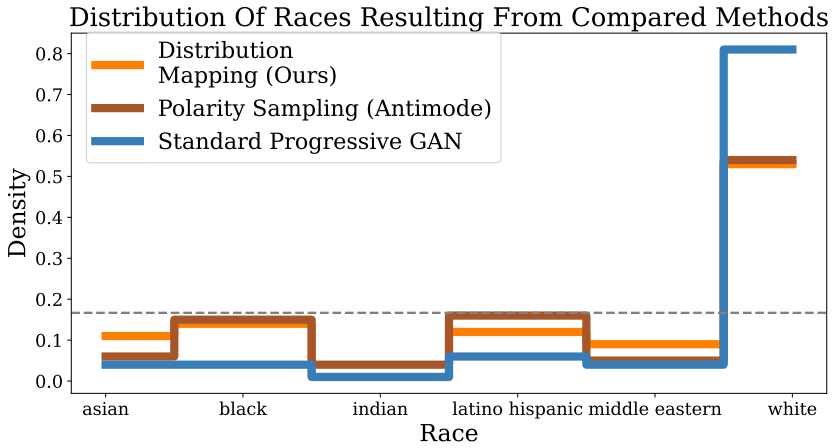

Figure 2: Distribution of our approach, Polarity Antimode Sampling (next best), and the standard generator.

A.4 Implementation Details

Ground Truth Shape Classifier

```
----------------------------------------------------------------
      Layer (type)               Output Shape
================================================================
            Conv2d-1         [-1, 32, 16, 16]
              ReLU-2         [-1, 32, 16, 16]
            Conv2d-3           [-1, 64, 8, 8]
              ReLU-4           [-1, 64, 8, 8]
            Conv2d-5          [-1, 128, 4, 4]
              ReLU-6          [-1, 128, 4, 4]
            Conv2d-7          [-1, 256, 2, 2]
              ReLU-8          [-1, 256, 2, 2]
            Conv2d-9            [-1, 2, 1, 1]
================================================================
```

Encoder for Shapes VAE

```
----------------------------------------------------------------
```

```
        Layer (type)               Output Shape
================================================================
            Conv2d-1           [-1, 32, 16, 16]
              ReLU-2           [-1, 32, 16, 16]
            Conv2d-3             [-1, 64, 8, 8]
              ReLU-4             [-1, 64, 8, 8]
            Conv2d-5            [-1, 128, 4, 4]
              ReLU-6            [-1, 128, 4, 4]
            Conv2d-7            [-1, 256, 2, 2]
              ReLU-8            [-1, 256, 2, 2]
            Conv2d-9          [-1, code_dim, 1, 1]
================================================================
```

Decoder for Shapes VAE

```
----------------------------------------------------------------
        Layer (type)               Output Shape
================================================================
   ConvTranspose2d-1            [-1, 256, 2, 2]
             ReLU-2            [-1, 256, 2, 2]
   ConvTranspose2d-3            [-1, 128, 8, 8]
             ReLU-4            [-1, 128, 8, 8]
   ConvTranspose2d-5           [-1, 64, 16, 16]
             ReLU-6           [-1, 64, 16, 16]
   ConvTranspose2d-7           [-1, 32, 32, 32]
             ReLU-8           [-1, 32, 32, 32]
   ConvTranspose2d-9             [-1, 3, 64, 64]
          Sigmoid-10             [-1, 3, 64, 64]
================================================================
```

Biased Age Classifier (Note: Target value was normalized age, made binary after)

```
----------------------------------------------------------------
        Layer (type)               Output Shape
================================================================
            Conv2d-1           [-1, 2, 32, 32]
       BatchNorm2d-2           [-1, 2, 32, 32]
         LeakyReLU-3           [-1, 2, 32, 32]
           Dropout-4           [-1, 2, 32, 32]
            Conv2d-5           [-1, 4, 16, 16]
       BatchNorm2d-6           [-1, 4, 16, 16]
         LeakyReLU-7           [-1, 4, 16, 16]
           Dropout-8           [-1, 4, 16, 16]
            Conv2d-9             [-1, 8, 8, 8]
      BatchNorm2d-10             [-1, 8, 8, 8]
        LeakyReLU-11             [-1, 8, 8, 8]
          Dropout-12             [-1, 8, 8, 8]
          Flatten-13                 [-1, 512]
           Linear-14                  [-1, 64]
        LeakyReLU-15                  [-1, 64]
           Linear-16                   [-1, 1]
          Sigmoid-17                   [-1, 1]
================================================================
```

Ground Truth Age Classifier (Note: Target value was normalized age; made binary after)

```
----------------------------------------------------------------
        Layer (type)               Output Shape
================================================================
            Conv2d-1           [-1, 8, 32, 32]
```

```
        BatchNorm2d-2           [-1, 8, 32, 32]
          LeakyReLU-3           [-1, 8, 32, 32]
             Dropout-4           [-1, 8, 32, 32]
              Conv2d-5          [-1, 16, 16, 16]
        BatchNorm2d-6          [-1, 16, 16, 16]
          LeakyReLU-7          [-1, 16, 16, 16]
             Dropout-8          [-1, 16, 16, 16]
              Conv2d-9            [-1, 32, 8, 8]
       BatchNorm2d-10            [-1, 32, 8, 8]
         LeakyReLU-11            [-1, 32, 8, 8]
            Dropout-12            [-1, 32, 8, 8]
            Flatten-13                 [-1, 2048]
             Linear-14                   [-1, 64]
         LeakyReLU-15                   [-1, 64]
             Linear-16                    [-1, 1]
            Sigmoid-17                    [-1, 1]
================================================================
```

The distribution mapper used default architecture of SDV's CTGAN [1] version 0.6.0, except for in the ProgressiveGAN experiment where `embedding_dim =512, generator_dim =(512,512)` were passed as arguments.

For the networks we trained, we utilized the Adam optimizer [2] with learning rate between 0.002 and 0.0001.

The linear classifier utilized Scikit-Learn's LinearSVC (for latent editing) and RidgeClassifier for the biased Shapes classifier.

A.5 Proof of Lemma 1

Proof. First, note that if $1^{|\mathcal{Y}|} \in Cone(E)$, then likewise $\frac{1}{\mathcal{Y}}1^{|\mathcal{Y}|} \in Cone(E)$.

Let $\mathbf{z}' \sim \mathbb{P}_{z|C_\phi=i}$; i.e., z is a draw from the distribution of noise such that the classifiers prediction of the generated sample corresponding to z' is group i.

Let $(C' \circ G_\theta)_* \mathbb{P}_{z|C_\phi=i}$ be the pushforward distribution of the perfect classifier C''s output when conditioned on the generator's output of draws from $\mathbb{P}_{z|C_\phi=i}$. Then, $(C' \circ G_\theta)_* \mathbb{P}_{z|C_\phi=i} = [Pr(\mathbf{y}=1|C_\theta=i), Pr(\mathbf{y}=2|C_\theta=i), \ldots, Pr(\mathbf{y}=N|C_\theta=i)] = E_{:,i}$. Thus, $Cone(\{(C' \circ G_\theta)_* \mathbb{P}_{z|C_\phi=i}, \ldots, (C' \circ G_\theta)_* \mathbb{P}_{z|C_\phi=|\mathcal{Y}|}\}) = Cone(E)$. Therefor, following from above, $\frac{1}{\mathcal{Y}}1^{|\mathcal{Y}|} \in Cone(\{(C' \circ G_\theta)_* \mathbb{P}_{z|C_\phi=i}, \ldots, (C' \circ G_\theta)_* \mathbb{P}_{z|C_\phi=|\mathcal{Y}|}\})$. This means that $\exists \lambda_1, \lambda_2, \ldots, \lambda_{|\mathcal{Y}}$ s.t. $\lambda_1 (C' \circ G_\theta)_* \mathbb{P}_{z|C_\phi=i} + \cdots + \lambda_{|\mathcal{Y}|} (C' \circ G_\theta)_* \mathbb{P}_{z|C_\phi=|\mathcal{Y}|} = [\frac{1}{|\mathcal{Y}|}, \ldots, \frac{1}{|\mathcal{Y}|}] = Unif\mathcal{Y}$. This is equivalent to saying that $C'(G_\theta(\mathbf{z})) \sim Unif(\mathcal{Y})$ for $\mathbf{z} \sim \sum_{i=1}^{|\mathcal{Y}|} \lambda_i \mathbb{P}_{z|C_\phi=i} = \mathbb{Q}^\lambda$. Thus, by definition $\mathbb{Q}^\lambda$ is a Fair Noise Distribution.

□

A.6 Proof of Lemma 2

Proof. Note that the sign of the coefficient of the cross product $E_{:,1} \times E_{:,2}$ is $P(\mathbf{y}=1|\hat{\mathbf{y}}=1)P(\mathbf{y}=2|\hat{\mathbf{y}}=2) - P(\mathbf{y}=1|\hat{\mathbf{y}}=2)P(\mathbf{y}=2|\hat{\mathbf{y}}=1)$. Also note that $E_{:,1} \times [0.5, 0.5]$ is $0.5P(\mathbf{y}=1|\hat{\mathbf{y}}=1) - 0.5P(\mathbf{y}=2|\hat{\mathbf{y}}=1)$.

Additionally, $P(\mathbf{y}=1|\hat{\mathbf{y}}=1)P(\mathbf{y}=2|\hat{\mathbf{y}}=2) > P(\mathbf{y}=1|\hat{\mathbf{y}}=1)0.5 > 0$, and $0 < P(\mathbf{y}=1|\hat{\mathbf{y}}=2)P(\mathbf{y}=2|\hat{\mathbf{y}}=1) < 0.5P(\mathbf{y}=2|\hat{\mathbf{y}}=1)$. Thus, the coefficient of $E_{:,1} \times E_{:,2}$ is greater than $E_{:,1} \times [0.5, 0.5]$, while there signs are equal. This implies that $[0.5, 0.5]$ is in between $E_{:,1}$ and $E_{:,2}$. Thus, $[0.5, 0.5] \in cone(E)$. The rest of the proof follows directly from Lemma 1. □

[1] `https://sdv.dev/SDV/user_guides/single_table/ctgan.html#how-to-modify-the-ctgan-hyperparameters`

A.7 Proof of Proposition 1

Proof. Note that $\mathbb{P}_E^{\lambda}$ has density $[\sum_i \lambda_i Pr(\mathbf{y}=1|\hat{\mathbf{y}}=i), \ldots, \sum_i \lambda_i Pr(\mathbf{y}=N|\hat{\mathbf{y}}=i)]$. For ease of notation let us refer to $\sum_i \lambda_i Pr(\mathbf{y}=m|\hat{\mathbf{y}}=i)$ as r_m^{λ}.

Then,

$$\begin{aligned} KL\{\mathbb{P}_E^{\lambda}||Unif(\mathcal{Y})\} &= \sum_m r_m^{\lambda} \log\big(\frac{r_m^{\lambda}}{u}\big) \\ &= \sum_m \Big(r_m^{\lambda}\log(r_m^{\lambda}) - r_m^{\lambda}\log(\frac{1}{|\mathcal{Y}|})\Big) \\ &= \sum_m r_m^{\lambda}\log(r_m^{\lambda}) - \sum_m r_m^{\lambda}\log(\frac{1}{|\mathcal{Y}|}) \end{aligned}$$

Note that $\log\big(\frac{1}{N}\big)$ is constant for each term in the second summation. Thus,

$$\begin{aligned} &= \sum_m r_m^{\lambda}\log(r_m^{\lambda}) - \log\big(\frac{1}{N}\big)\sum r_m^{\lambda} \\ &= \sum_m r_m^{\lambda}\log(r_m^{\lambda}) - \log\big(\frac{1}{N}\big), \end{aligned}$$

As $\log\big(\frac{1}{N}\big)$ does not depend on r_m^{λ},

$$\begin{aligned} \operatorname*{argmin}_{\lambda} KL\{\mathbb{P}_E^{\lambda}||Unif(\mathcal{Y})\} &= \operatorname*{argmin}_{\lambda} \sum_m r_m^{\lambda}\log(r_m^{\lambda}) \\ &= \operatorname*{argmin}_{\lambda} -H(\mathbb{P}_E^{\lambda}) \\ &= \operatorname*{argmax}_{\lambda} H(\mathbb{P}_E^{\lambda}) \end{aligned}$$

□

A.8 Proof of Proposition 2

Proof.

$$\begin{aligned} (C'\circ G_\theta)_*\mathbb{P}_{z|C_\phi=i} =& [Pr(\mathbf{y}=1|C_\theta=i), Pr(\mathbf{y}=2|C_\theta=i), \ldots, Pr(\mathbf{y}=N|C_\theta=i)] \\ \implies \sum_i \lambda_i (C'\circ G_\theta)_*\mathbb{P}_{z|C_\phi=i} =& [\sum_i \lambda Pr(\mathbf{y}=1|C_\theta=i), \sum_i Pr(\mathbf{y}=2|C_\theta=i), \\ & \ldots, \sum_i \lambda Pr(\mathbf{y}=N|C_\theta=i)] \\ \implies \mathbb{P}_E =& \mathbb{Q}^{\lambda} \end{aligned}$$

□

A.9 Proof of Theorem 1

The first statement follows directly from Proposition 1 and Proposition 2.

If $C_\phi = C'$, then $\{\frac{E_{:,1}}{|E_{:,1}|}, \ldots, \frac{E_{:,|\mathcal{Y}|}}{|E_{:,|\mathcal{Y}|}|}\}$ forms a standard basis of $\mathbb{R}^{|\mathcal{Y}|}$, and therefor $1^{|\mathcal{Y}|}$ is is in $Cone(E)$. Thus, $\mathcal{Q}^{\lambda *}$ is a Fair Noise Distribution by Lemma 1.

References

[1] S. Garg, Y. Wu, S. Balakrishnan, and Z. Lipton. A Unified View of Label Shift Estimation. In *Advances in Neural Information Processing Systems*, volume 33, pages 3290–3300. Curran Associates, Inc., 2020.

[2] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization, Jan. 2017. arXiv:1412.6980 [cs].

[3] T. Sipka, M. Sulc, and J. Matas. The Hitchhiker's Guide to Prior-Shift Adaptation. In *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2031–2039, Waikoloa, HI, USA, Jan. 2022. IEEE.