DiffInject: Revisiting Debias via Synthetic Data Generation using Diffusion-based Style Injection

Supplementary Material

A. Related Work

Previous Debiasing Methods. Prior approaches to debiasing have employed supervised training using explicitly defined bias labels [14, 28, 29]. These methods extracted bias features and attributes from datasets under the assumption that bias labels were explicitly predefined. Recent works has sought to tackle biases without depending solely on pre-established bias labels. Instead, these strategies aim to reduce human intervention through techniques such as augmentation and re-weighting of properties. LfF [23] pinpoints bias-conflict samples by deploying two concurrently trained and updated models, f_D and f_B, with the debiased model f_D adjusting CE loss based on a relative difficulty score. Rebias [4] strives to mitigate bias by disentangling and interchanging features within the latent space. A^2 [2] leveraged StyleGAN [13] to produce augmented bias-conflict samples through a few-shot adaptation method [25]. AmpliBias [16] employed FastGAN for few-shot learning in generating synthetic bias-conflict samples. Yet, no further exploration on diffusion models were undergone in context of debiasing.

Content injection using diffusion models Numerous methods has been introduced to enhance controllability in image generation through diffusion-based models [6, 10, 24]. Recent works have explored the incorporation of either text guidance [1, 8, 11, 17, 27] or structure guidance [21, 22, 31] as a method of content injection. However, these approaches typically rely on textual descriptions or structure maps as conditioning inputs. Concurrently, alternative methodologies [15, 18] propose the utilization of reference images for image editing guidance. In contrast, InjectFusion [12] explores a novel approach by leveraging the latent space of a frozen, pretrained diffusion model as a means of content injection from a reference image. We further investigate the exploitation of the semantic latent space as a source of control to generate synthetic images, aiming to mitigate biases in classification tasks.

B. Implementation Details

We provide further details in implementation settings as the following.

B.1. Training ADM with P2-weighting

We train the diffusion model by setting $T = 1000$ for all experiments. We train our model with a fixed size of 32×32 images for CMNIST and CCIFAR-10, and 256×256 for BFFHQ and Dogs & Cats. Note that images for CMNIST are resized to facilitate the implementation of P2-weighting [5].

B.2. Injecting Biased Contents

The parameter t_{edit} is empirically defined such that $LPIPS(x, P_{t_{\text{edit}}}) = 0.33$, while t_{boost} is fixed as 200. We set content injection ratio γ as 0.9, 0.3, 0.7, and 0.2 for CMNIST, CCIFAR-10, BFFHQ, and Dogs & Cats, respectively. We apply local content injection for CMNIST, BFFHQ, and Dogs & Cats, and global content injection for CCIFAR-10. Bias-conflict ratio is set as 0.6 for BFFHQ and Dogs & Cats, and 0.1 for CMNIST and CCIFAR-10. Ablation studies on bias-conflict ratio can be explored in future work.

InjectFusion [12] takes approximately 7-10 seconds (2-3 seconds for computing inversion for each of the two images and applying content injection, respectively) per generated sample for BFFHQ and Dogs & Cats, and 90 seconds (30 seconds for computing inversion for each of the two images and applying content injection, respectively) per generated sample for CMNIST and CCIFAR-10, based on NVIDIA A100 and NVIDIA H100 GPUs. We use multiprocessing to accelerate the content injection process.

B.3. Training Unbiased Classifier

We implement the preprocessing techniques described in DisEnt [20]: We apply random crop and horizontal flip transformations for CCIFAR-10 and BFFHQ, and apply normalization with the mean of (0.4914, 0.4822, 0.4465) and standard deviation of (0.2023, 0.1994, 0.2010) for each channel. We do not implement any augmentations for CMNIST and Dogs & Cats. We use cross entropy loss as our loss function, and use Adam optimizer with the learning rate of 0.001 for CMNIST and CCIFAR-10, and 0.0001 for BFFHQ and Dogs & Cats.

B.4. Additional Generated Synthetic Images

In this section, we provide additional samples generated from DiffInject. Figure 4 includes synthetic samples for CMNIST and CCIFAR-10. Figure 5 and Figure 6 includes generated samples for BFFHQ and Dogs & Cats, respectively. Each figure consists of three columns representing, from left to right, the original samples from the dataset, top-k loss samples, and synthetic samples generated from DiffInject, respectively.

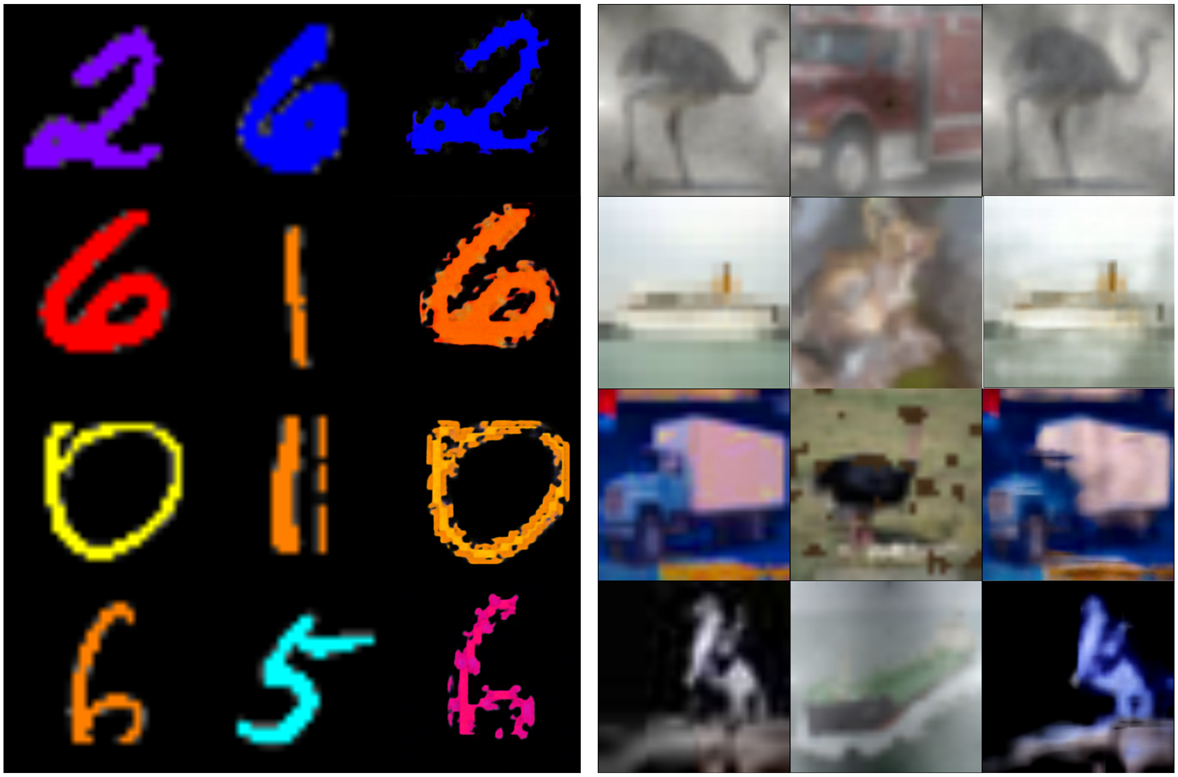

Figure 4. Examples of generated bias-conflict samples with DiffInject for CMNIST and CCIFAR-10 dataset. The three columns represent samples from the original dataset, top-k loss samples and generated samples, respectively.

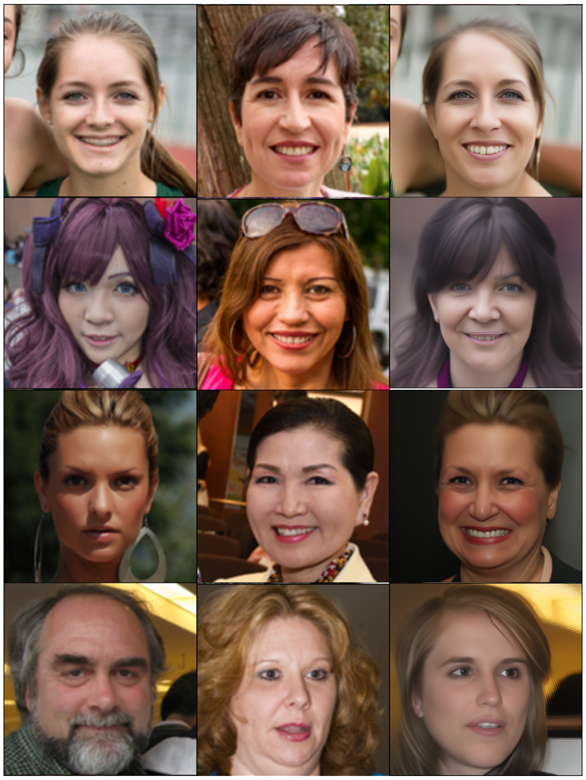

Figure 5. Examples of generated bias-conflict samples with DiffInject for BFFHQ dataset. The three columns represent samples from the original dataset, top-k loss samples and generated samples, respectively.

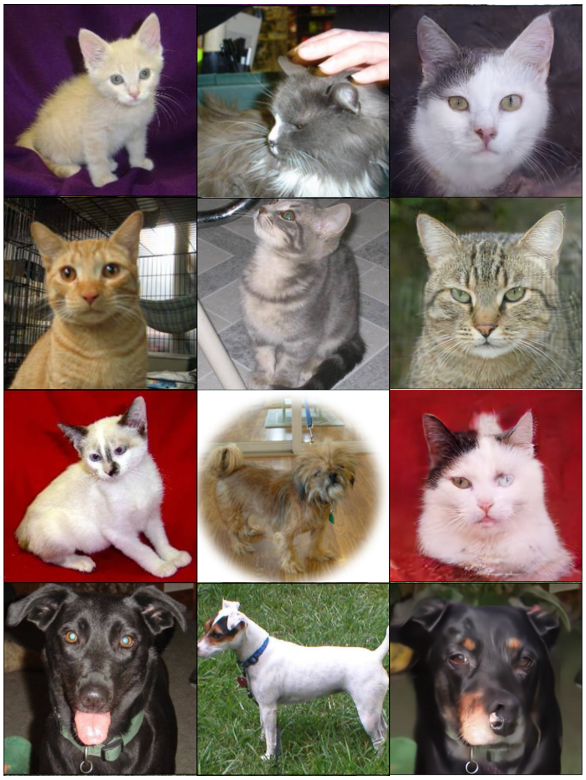

Figure 6. Examples of generated bias-conflict samples with DiffInject for Dogs & Cats dataset. The three columns represent samples from the original dataset, top-k loss samples and generated samples, respectively.