RepMix: Representation Mixing for Robust Attribution of Synthesized Images [Supplementary Materials]

Tu Bui[1], Ning Yu[2], and John Collomosse[1,3]

[1] University of Surrey t.v.bui@surrey.ac.uk
[2] Salesforce Research ning.yu@salesforce.com
[3] Adobe Research collomos@adobe.com

1 More details of the Attribution88 benchmark

1.1 Training GAN models

To compose the Attribution88 benchmark, we need to train 77 GAN models for 7 GAN architectures and 11 semantics. Tab. 1 shows the source repositories that we employed to train those GAN models. We use the recommended settings. Specifically, Stylegan3 [7] has configuration stylegan3-t, batch size 32, and 20M training iterations. Stylegan2 [9] has configuration config-f, batch size 32, and 10M training iterations. Stylegan [8] settings are the same as Stylegan2 except using configuration config-a. Progan [6] has batch size 32 and 12M training iterations. Cramergan [2] has a g-resnet5 backbone, 256 dof-dim, gradient penalty of 10.0 and 150k training iterations. MMDgan [3] also has a g-resnet5 backbone, 16 dof-dim, gradient penalty of 1.0 and 150k training iterations. SNgan [11] has resnet backbone, batch size 32, hinge loss, and 100k training iterations. To avoid mode collapse, we checkpoint regularly and manually examine the quality and diversity of the output sampled images.

Despite our training efforts, several models are hard to train. We therefore use publicly released models if available. Specifically, there are pretrained Progan [6] models[4] for all the 10 LSUN semantics, which are used in Attribution88. For CelebA face where there does not exist an official pretrained Progan model, we use an unofficial version on tensor-hub[5]. For computational efficiency and eco-friendly validation, all synthesized images are downscaled to 128×128 if its public-release corresponding models were trained for higher resolutions. The diversity in our GAN model collection makes Attribution88 more practical and challenging. Example images are shown in Fig. 1.

[4] https://drive.google.com/drive/folders/15hvzxt_XxuokSmj0uO4xxMTMWVc0cIMU
[5] https://tfhub.dev/google/progan-128/1

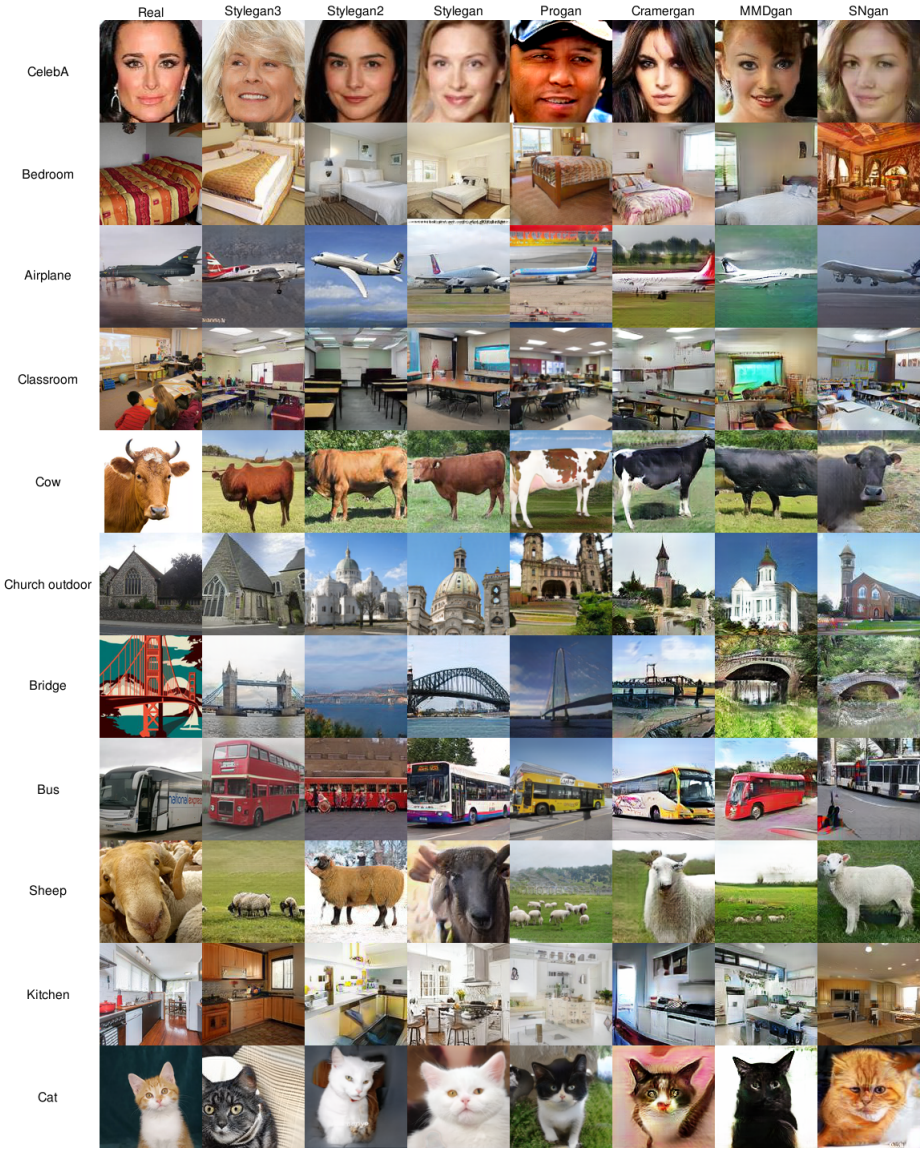

Fig. 1. Image examples of each source/semantic categories in Attribution88 benchmark.

1.2 Data cleaning

We describe in Sec.3 of the main paper our cleaning procedure for Attribution88. The goal is to remove images with artifacts without reducing the level of diversity in the generated data. Removing images with artifacts is achieved by selecting the synthesized images that are perceptually closest to a real one, while diversity

Table 1. Code repositories used in our GAN model training for Attribution88 benchmark

Architecture	Codebase
Stylegan3	https://github.com/NVlabs/stylegan3
Stylegan2, Stylegan	https://github.com/NVlabs/stylegan2
Progan	https://github.com/tkarras/progressive_growing_of_gans
Cramergan, MMDgan	https://github.com/mbinkowski/MMD-GAN
SNgan	https://github.com/pfnet-research/sngan_projection

(a) Random (b) Yu *et al*. [13] (c) Proposed

Fig. 2. Data cleaning examples for Stylegan2 model source, LSUN Bedroom semantics. (a) random images without cleaning - high diversity but often contain artifacts; (b) data cleaning using Yu *et al*. method - high quality but lack diversity; (c) proposed method - balancing diversity and quality.

is maintained via clustering. We note that Yu *et al*. [13] also performed data cleaning, however their method sacrifices diversity for quality, leading to many similar images. Fig. 2 compares our data cleaning method with Yu *et al*. on Stylegan2 bedroom images. It is worth noting that, although models from the Stylegan family [8, 9, 7] have an internal truncation mechanism to enhance the overall quality of the generated images, artifacts are still present in some of them (Fig. 2a). Our cleaning method filters out the images with artifacts at a neglectable cost of only 1.5 degradation in FID, as opposed to 8.8x degradation in Yu *et al*. approach.

1.3 Perturbations

Fig. 3 shows the effect of 19 ImageNet-C perturbations [5] on an example synthesized image. These transformations further complicate the attribution task as it deteriorates fine-level features on the image. Fig. 4 illustrates the mean pixel and mean spectrum of several sources and semantics, before and after random perturbations. Unique GAN patterns are visually observable on the mean spectrum of the clean images, which are successfully exploited for image attribution

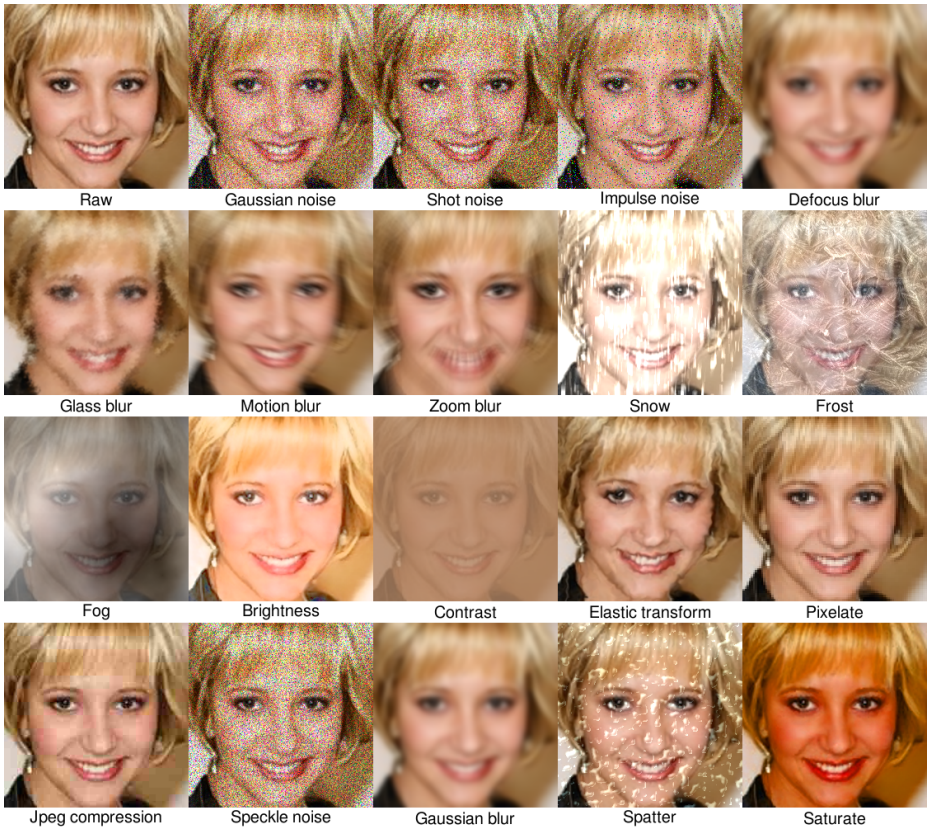

Fig. 3. Effects of different perturbations on a Stylegan3 CelebA generated image. All perturbations have strength 3 (out of 5). Best view in color when zoom in.

in DCT-CNN [4]. However, random perturbations make it difficult to recognize and attribute these patterns to the GAN architectures.

CelebA versus the other semantics. Fig. 4 also shows the difference between CelebA faces and the other semantics in Attribution88 benchmark. The visible mean pixel images regardless of generator sources demonstrate that face images are more aligned than the other semantic objects. This demonstrates the needs of validating an image attribution algorithm (and even an image synthesis algorithm) on multiple semantic sets, which is a design feature of Attribution88. The inclusion of CelebA along with other LSUN semantics also adds diversity to our benchmark.

2 Baseline implementation

Tab. 2 shows the code repositories that we employed to train and evaluate the baseline models. We follow the recommended settings and only change the data

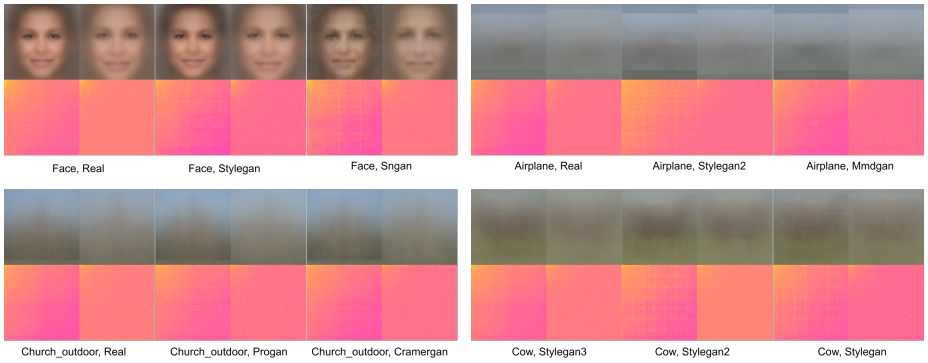

Fig. 4. Mean images and spectrums of several generator classes and semantics. For each inset, top: mean image before (left) and after (right) random perturbation, bottom: corresponding mean of DCT images.

Table 2. Code repositories of the baseline methods reported in Tab.1 of the main paper.

Baseline	Codebase
Yu *et al*. [13], Eigen Face [12]	https://github.com/ningyu1991/GANFingerprints
DCT-CNN [4], PRNU [10]	https://github.com/RUB-SysSec/GANDCTAnalysis
Reverse Eng. [1]	https://github.com/vishal3477/Reverse_Engineering_GMs

augmentation strategy (*i.e*. ImageNet-C [5]) for fair comparisons with our proposed method. For Yu *et al*. [13] approach, we also re-implement their method using Pytorch instead of Tensorflow and add an extra 256-D FC layer and report the improved performance in Tab.1 of the main paper. For DCT-CNN [4], we observe that the training of their recommended architecture, a 1-layer multinomial regression model, collapses due to the complexity of Attribution88 benchmark. We therefore use a Resnet50 backbone (same as RepMix) instead.

3 Further results

3.1 More backbones

Tab. 3 shows performance of RepMix on more advanced backbones, Resnet101 and Densenet121, on the ablated set (c.f. Tab.3 of the main paper). We achieve slightly better performance at the cost of more expensive models.

3.2 Beta distribution in the RepMix layer

Fig. 5 shows our experiment with the beta distribution in RepMix layer on the ablated set of Attribution88 benchmark. RepMix is robust to β value of 0.25

Table 3. RepMix with Resnet101 and Densenet121 backbones on the ablated set.

Backbone	Detection Acc. ⇑	Attribution Acc. ⇑	Attribution NMI ⇑
Resnet101	**0.9523**	0.7430	0.5546
Densenet121	0.9504	**0.7474**	**0.5673**

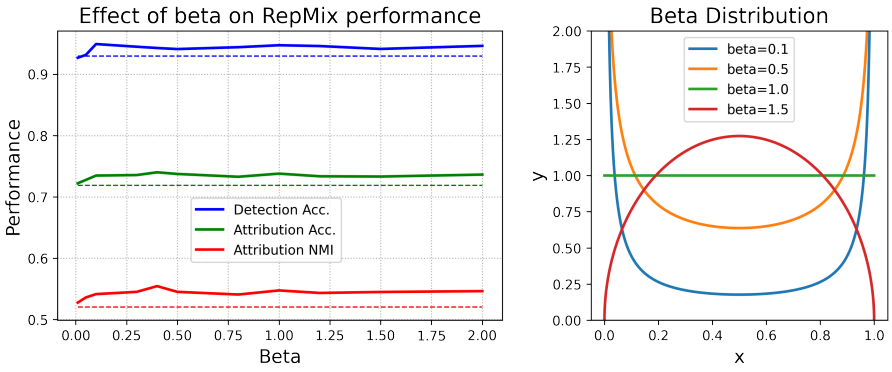

Fig. 5. Effects of beta on RepMix. Dash lines refer to the no-mixing baselines.

and above, performing favorably even at uniform distribution ($\beta = 1.0$). We argue that the RepMix layer is positioned close to the loss layer in our network, resulting in a stronger regularization and less dependence on the value of the mixing ratio.

3.3 Confusion matrix

Fig. 6 shows in details the contribution of each source class's performance towards the overall attribution accuracy of RepMix and other baselines. It is consistent across all deep-learning methods that attribution performance is high for Real, Progan and SNgan, probably because of the unique data distribution (Real) or distinct GAN architecture (Progan, SNgan). The source of confusion mainly comes from attributing images from the Stylegan family and Cramergan/MMDgan because of the similarities in design of these GAN architectures.

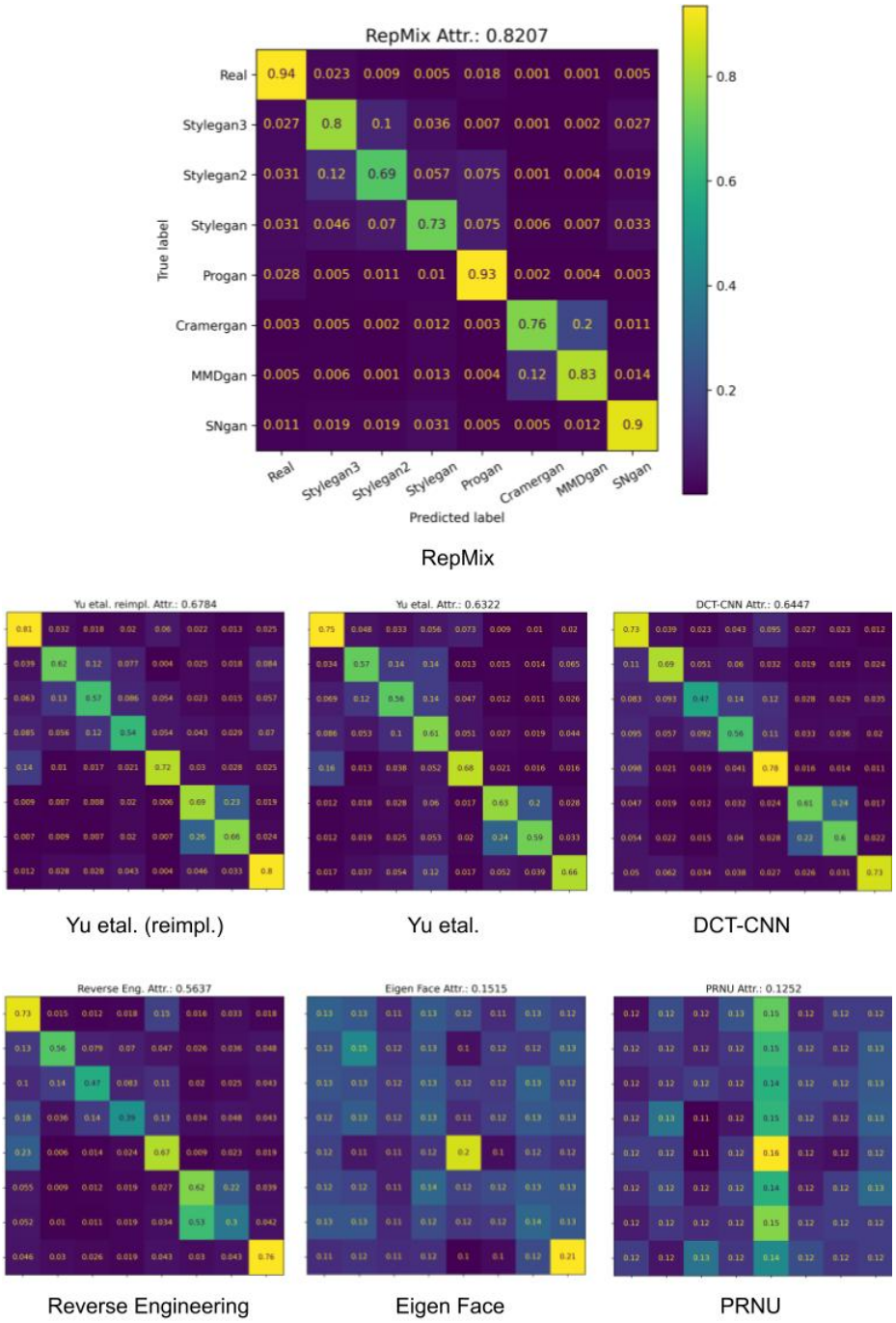

Fig. 6. Confusion matrices of RepMix and all baselines on Attribution88.

References

1. Asnani, V., Yin, X., Hassner, T., Liu, X.: Reverse engineering of generative models: Inferring model hyperparameters from generated images. arXiv preprint arXiv:2106.07873 (2021)
2. Bellemare, M.G., Danihelka, I., Dabney, W., Mohamed, S., Lakshminarayanan, B., Hoyer, S., Munos, R.: The cramer distance as a solution to biased wasserstein gradients. arXiv preprint arXiv:1705.10743 (2017)
3. Bińkowski, M., Sutherland, D.J., Arbel, M., Gretton, A.: Demystifying mmd gans. In: Proc. ICLR (2018)
4. Frank, J., Eisenhofer, T., Schönherr, L., Fischer, A., Kolossa, D., Holz, T.: Leveraging frequency analysis for deep fake image recognition. In: Proc. ICML. pp. 3247–3258. PMLR (2020)
5. Hendrycks, D., Dietterich, T.: Benchmarking neural network robustness to common corruptions and perturbations. In: Proc. ICLR (2018)
6. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of gans for improved quality, stability, and variation. In: Proc. ICLR (2018)
7. Karras, T., Aittala, M., Laine, S., Härkönen, E., Hellsten, J., Lehtinen, J., Aila, T.: Alias-free generative adversarial networks. NeurIPS **34** (2021)
8. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: Proc. CVPR. pp. 4401–4410 (2019)
9. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. In: Proc. CVPR. pp. 8110–8119 (2020)
10. Marra, F., Gragnaniello, D., Verdoliva, L., Poggi, G.: Do gans leave artificial fingerprints? In: Proc. MIPR. pp. 506–511. IEEE (2019)
11. Miyato, T., Kataoka, T., Koyama, M., Yoshida, Y.: Spectral normalization for generative adversarial networks. In: International Conference on Learning Representations (2018)
12. Sirovich, L., Kirby, M.: Low-dimensional procedure for the characterization of human faces. Josa a **4**(3), 519–524 (1987)
13. Yu, N., Davis, L.S., Fritz, M.: Attributing fake images to gans: Learning and analyzing gan fingerprints. In: Proc. ICCV. pp. 7556–7566 (2019)