Generator Attribution With Biased Data

Daniel Delijani, Boston University
deljani@bu.edu

1 Introduction

Image generation models learn to produce novel, realistic-looking images to mock the images it is provided in its training data. This general class of models is capable of fantastic feats. Most popularly, image generation models such as those utilized by OpenAI's DALL-E [1] can generate photorealistic images based on a user's prompt. While models such as this can be massively useful to society as they not only allow people to create art, pictures for presentations, or even new designs for products, they also are mainstream mediums which have increased the public's interest in AI. However, the quality and ease-of-use of these models can also lead to negative consequences. Most notably, these models can be used to develop photo-realistic propaganda. Additionally, with the rise of social media and thus the rapid transfer of information from anonymized sources to large audiences, the scale at which this propaganda can produce negative impact is massive. Therefore, there is a large necessity to develop techniques to combat these models' potential for widespread deception. There exists methods to attribute synthetic media to generators, or generally detect synthetic media [2, 3] which have been proven to work very well using popular computer vision architectures including VGG16 and ResNet50 along with transfer learning. However, existing literature applying these techniques to this problem all assume an equal distribution of generator (image generation model) to subclass (what is in the image, e.g. horse) mapping in the training and validation datasets. In this paper, we compare a variety of techniques for attributing an image to an image generation model given a biased dataset, including a novel approach using adversarial debiasing.

2 Problem Definition

For this problem, we want to accurately predict which of the following image generation models is responsible for generating an image: Guided Diffusion [4], Latent Diffusion [5], LSGM [6], StyleGAN2 [7], StyleGAN3 [8], or Taming-Transformers [9]. The distribution of generator-subclass mapping in our training dataset is shown in the following figure:

Generation Model	Horse	Face	Animal Face	Bed	Church	Cat	Car
Guided Diffusion	7001	0	0	6999	0	0	0
Latent Diffusion	0	14000	0	7000	7000	0	0
LSGM	0	8480	0	0	0	0	0
StyleGAN2	7000	7000	0	0	7000	7000	7000
StyleGAN3	0	14000	7000	0	0	0	0
Taming-Transformers	0	14000	0	0	0	3440	0

Our testing dataset consists of entirely military vehicles produced by the image generation models. The number of images per generator in the test dataset is as follows:

Generation Model	Military Vehicle
Guided Diffusion	29
Latent Diffusion	30
LSGM	30
StyleGAN2	29
StyleGAN3	29
Taming-Transformers	29

We would like to develop a model that will most accurately predict the generation model on the test dataset when trained with the training dataset, which is not only biased but also does not contain the subject of the test data, military vehicles.

3 Techniques

3.1 Baseline

For this approach, we utilize a single-headed CNN architecture. It uses Resnet50, replacing its last fully connected classification layer with a fully connected classification layer with 6 output neurons and softmax activation. We initialize weights randomly. Additionally, we use cross entropy loss to update weights using a batch size of 4, along with the Adam optimizer with a learning rate of .001. Additionally, the training pipeline as well as the test pipeline go through some preprocessing. Firstly, images are resized to (224, 224, 3). Then, images are normalized across all dimensions to mean value .5 and standard deviation .5. Additionally, the model is trained for 25 epochs.

3.2 Weighted Sampling

We use the same architecture, hyperparameters, and preprocessing as the baseline model for this model. However, we use a weighted random sampling method when generating batches. The probability of an image i being sampled for a given batch is given by the following function:

$$Pr(i) = \frac{1}{(K) * (C_i)}$$

K: the number of classes (6 in this case)
C_i: the number of images in image i's class

Additionally, the weighted sampler replaces an image in the training dataset after including it in its training for a given epoch, therefore allowing the model to sample the same image multiple times in the same epoch. This weighted sampler is formulated so that the expected value for the number of samples per class that are sampled in a given epoch are equally distributed among the classes. This, in theory, will help with debiasing the model by removing its incentive to predict classes with more training examples with higher probability.

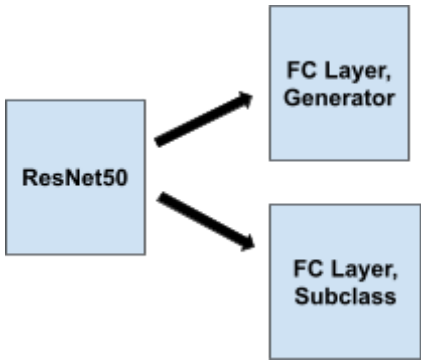
3.3 Fine Tuning

We use the baseline model from 3.1 as a pre-trained model to begin training with. On the test dataset with military vehicles, we conduct a .80-.20 train-test split, leading to 143 images in the train set and 37 for the test set. Using the same training parameters as the other models, we fine-tune the parameters in the classification layer for 75

epochs on the train set. We use stochastic gradient descent with cross entropy loss and a learning rate of .001 with a momentum of .9. We also use the same image preprocessing as baseline.

3.4 Adversarial Debiasing

Lastly, we utilize an adversarial debiasing approach to debias the classifier. In this method, we utilize an approach similar to [10]. We utilize a multi-headed model, where both heads are fully connected classification layers with softmax activation. Head 1 is designed for predicting the generator, and therefore has 6 neurons. Head 2 is designed for predicting the subclass, and so it has 7 neurons. The neurons previous to the two heads are shared, both using non-pretrained ResNet50 architecture to remain consistent with previous approaches. Additionally, both heads utilize cross entropy loss.



We use the same preprocessing steps as the previous models, and also use 25 epochs.

In terms of training, we use a three-step process for each batch.

1. Update weights in fully connected subclass layer to *minimize* the subclass loss function.
2. Update weights in the resnet50 layers and the fully connected generator layer to *minimize* the generator loss function.
3. Update weights in the resnet50 layers to *maximize* the subclass loss function.

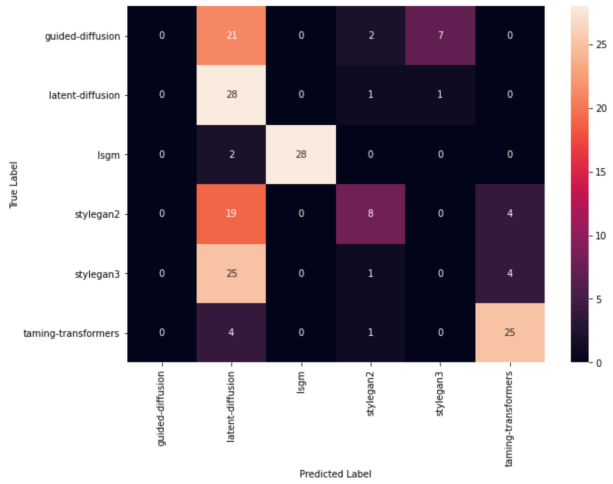
This three-step process is designed such that the parameters present in the resnet50 layers are penalized for learning subclass features; therefore, they will theoretically focus their learning on the generator task.

4 Results

For each technique, we evaluate the precision, recall, and F1-score for each generator, along with a confusion matrix to visualize how the bias of the model behaves under each model architecture. Then, we compare the accuracies of the models along with their average precision, recall, and F1-score.

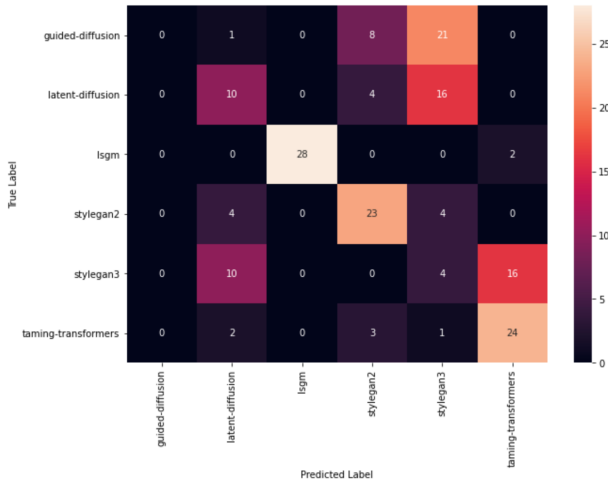
4.1 Baseline

Generation Model	Precision	Recall	F1-Score	Support
Guided Diffusion	0.00	0.00	0.00	29
Latent Diffusion	0.28	0.93	0.43	30
LSGM	1.00	0.93	0.97	30
StyleGAN2	0.62	0.26	0.36	29
StyleGAN3	0.00	0.00	0.00	29
Taming-Transformers	0.76	0.83	0.79	29
Average	0.44	0.49	0.43	176



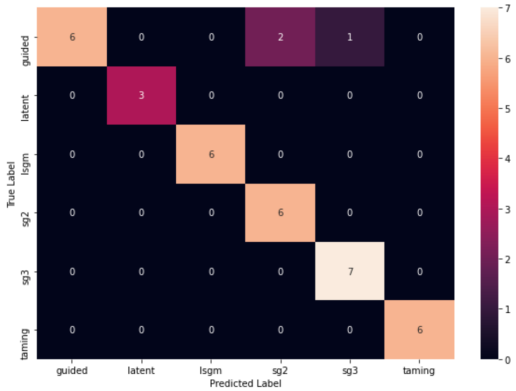
4.2 Weighted Sampling

Generation Model	Precision	Recall	F1-Score	Support
Guided Diffusion	0.00	0.00	0.00	29
Latent Diffusion	0.37	0.33	0.35	30
LSGM	1.00	0.93	0.97	30
StyleGAN2	0.61	0.74	0.67	29
StyleGAN3	0.09	0.13	0.11	29
Taming-Trans formers	0.57	0.80	0.67	29
Average	0.44	0.49	0.46	176



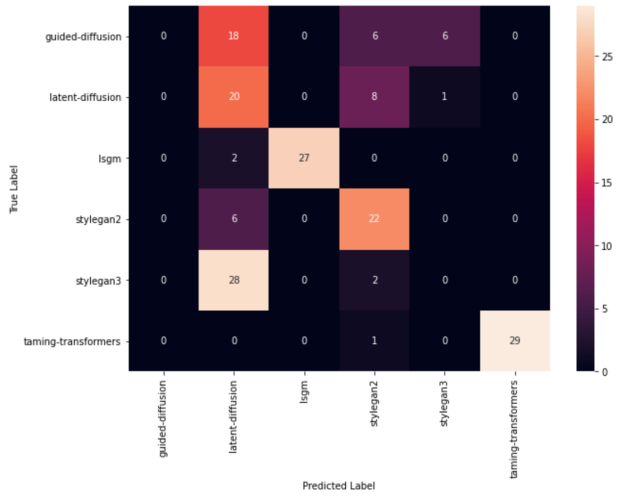
3.3 Fine Tuning

Generation Model	Precision	Recall	F1-Score	Support
Guided Diffusion	1.00	0.67	0.80	9
Latent Diffusion	1.00	1.00	1.00	3
LSGM	1.00	1.00	1.00	6
StyleGAN2	0.75	1.00	0.86	6
StyleGAN3	0.88	1.00	0.93	7
Taming-Trans formers	1.00	1.00	1.00	6
Average	0.94	0.94	0.93	37



3.4 Adversarial Debiasing

Generation Model	Precision	Recall	F1-Score	Support
Guided Diffusion	0.00	0.00	0.00	29
Latent Diffusion	0.28	0.70	0.40	30
LSGM	1.00	0.93	0.97	30
StyleGAN2	0.55	0.72	0.63	29
StyleGAN3	0.00	0.00	0.00	29
Taming-Trans formers	1.00	0.97	0.98	29
Average	0.47	0.55	0.50	176



3.5 Results Summary

Technique	Average Precision	Average Recall	Average F1-Score	Accuracy
Baseline	0.44	0.49	0.43	0.49
Weighted Sampling	0.44	0.49	0.46	0.49
Fine Tuning	0.94	0.94	0.93	0.91
Adversarial Debiasing	0.47	0.55	0.50	0.56

Based on our methodology, the following approaches ordered in accuracy for predicting generators with the availability of biased data is the following:

1. Fine Tuning
2. Adversarial Debiasing
3. Baseline/Weighted Sampling

While these results favor the fine-tuning approach, this approach is not always practical. When developing a model to predict a generator, images used at deployment may differ from those presented during training. The requirement to develop a dataset for each subclass, then fine-tune the model for it in order to achieve an effective model requires a lot of work.

Thus the most effective practical approach is the adversarial debiased model.

5 Summary

In this work, we present multiple methodologies for attributing synthetic images to a generation model given an imbalanced dataset: a baseline model, a model utilizing weighted sampling, a model with fine-tuning, and an adversarial debiasing approach. We show that the most robust approach for developing a model to perform well on a new subclass of images is to fine-tune a new dataset containing the new subclass; however, the most effective model for practical use is an adversarial debiasing approach.

References

[1] Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., ... & Sutskever, I. (2021, July). Zero-shot text-to-image generation. In International Conference on Machine Learning (pp. 8821-8831). PMLR.

[2] Kumar, N., Pranav, P., Nirney, V., & Geetha, V. (2021, September). Deepfake Image Detection using CNNs and Transfer Learning. In 2021 International Conference on Computing, Communication and Green Engineering (CCGE) (pp. 1-6). IEEE.

[3] Chang, X., Wu, J., Yang, T., & Feng, G. (2020, July). Deepfake face image detection based on improved VGG convolutional neural network. In 2020 39th chinese control conference (CCC) (pp. 7252-7256). IEEE.

[4] Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., ... & Chen, M. (2021). Glide: Towards photorealistic image generation and editing with text-guided diffusion models. arXiv preprint arXiv:2112.10741.

[5] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 10684-10695).

[6] Vahdat, A., Kreis, K., & Kautz, J. (2021). Score-based generative modeling in latent space. Advances in Neural Information Processing Systems, 34, 11287-11302.

[7] Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., & Aila, T. (2020). Analyzing and improving the image quality of stylegan. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 8110-8119).

[8] Karras, T., Aittala, M., Laine, S., Härkönen, E., Hellsten, J., Lehtinen, J., & Aila, T. (2021). Alias-free generative adversarial networks. Advances in Neural Information Processing Systems, 34, 852-863.

[9] Esser, P., Rombach, R., & Ommer, B. (2021). Taming transformers for high-resolution image synthesis. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 12873-12883).

[10] Correa, R., Jeong, J. J., Patel, B., Trivedi, H., Gichoya, J. W., & Banerjee, I. (2021). Two-step adversarial debiasing with partial learning--medical image case-studies. arXiv preprint arXiv:2111.08711.