COM505 DIP Project

GAN Fingerprinting

- Kausik N. (COE17B010)
- Parth Patel (COE17B020)
- Hrishikesh V. (CED17I001)
- Mahidhar R. (CED17I010)

Abstract

Recent advances in Generative Adversarial Networks (GANs) have shown increasing success in generating photorealistic images. But they also raise challenges to visual forensics and model attribution.

Figure 1: *Examples of Photorealistic GAN-Generated Faces.*

We present a study of learning GAN fingerprints towards image attribution and using them to classify an image as real or GAN-generated.

For GAN-generated images, we further identify their sources.

Background

For model fingerprinting, each GAN model is characterized by many parameters:

- Training dataset distribution
- Network architecture
- loss design, optimization strategy, and
- hyperparameter settings.

Because of the non-convexity of the objective function and the instability of adversarial equilibrium between the generator and discriminator in GANs, the values of model weights are sensitive to their random initializations and do not converge to the same values during each training. This indicates that even though two well-trained GAN models may perform equivalently, they generate high quality images differently. This suggests the **existence and uniqueness** of GAN fingerprints.

We define the model fingerprint per GAN instance as a reference vector, such that it consistently interacts with all its generated images. As far as image fingerprinting goes, GAN-generated images are the outcomes of a large number of fixed filtering and non-linear processes, which generate common and stable patterns within the same GAN instances but are distinct across different GAN instances. That suggests the existence of image fingerprints and attributability towards their GAN sources.

We introduce the fingerprint per image as a feature vector encoded from that image.

What is GAN?

The main focus for GAN (Generative Adversarial Networks) is to generate data from scratch, mostly images but other domains including music have been done. But the scope of application is far bigger than this.

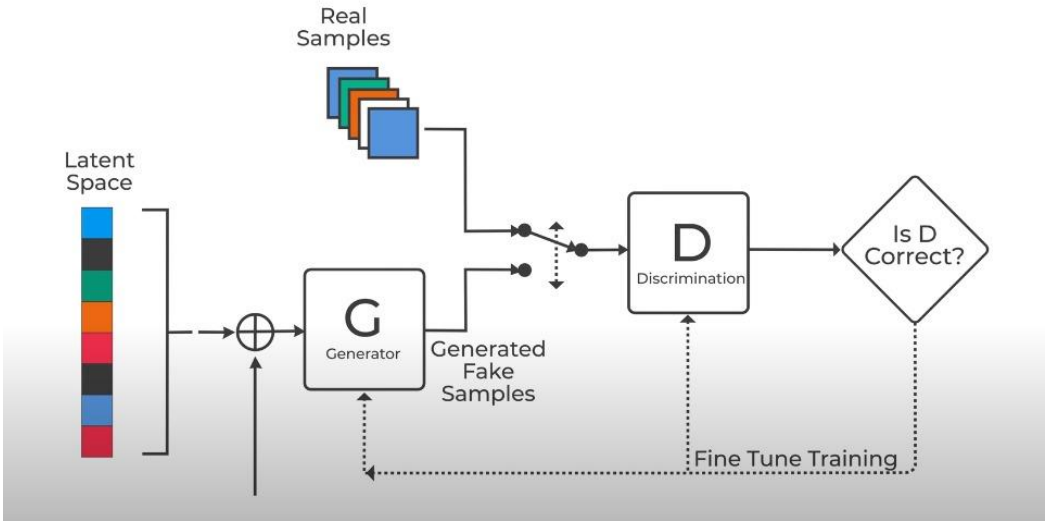


Figure 2: General GAN model

GAN composes of two deep networks:

1. the generator, and
2. the discriminator (as seen from the diagram above).

The Generator generates fake samples of data (be it an image, audio, etc.) and tries to fool the Discriminator. The Discriminator, on the other hand, tries to distinguish between the real and fake samples. The Generator and the Discriminator are both Neural Networks and they both run in

competition with each other in the training phase. The steps are repeated several times and in this, the Generator and Discriminator get better and better in their respective jobs after each repetition.

Analogy

Generators are counterfeiters who produce fake money and Discriminator are police who detects fake money.

Initially, counterfeiters bad at generating fake money and police always catches them.

Over time, counterfeiters get better at producing fake money that police finds difficult to detect and police gets better at detecting fake money.

Eventually, counterfeiters are forced to make perfect replicas of real money.

Applications of GAN

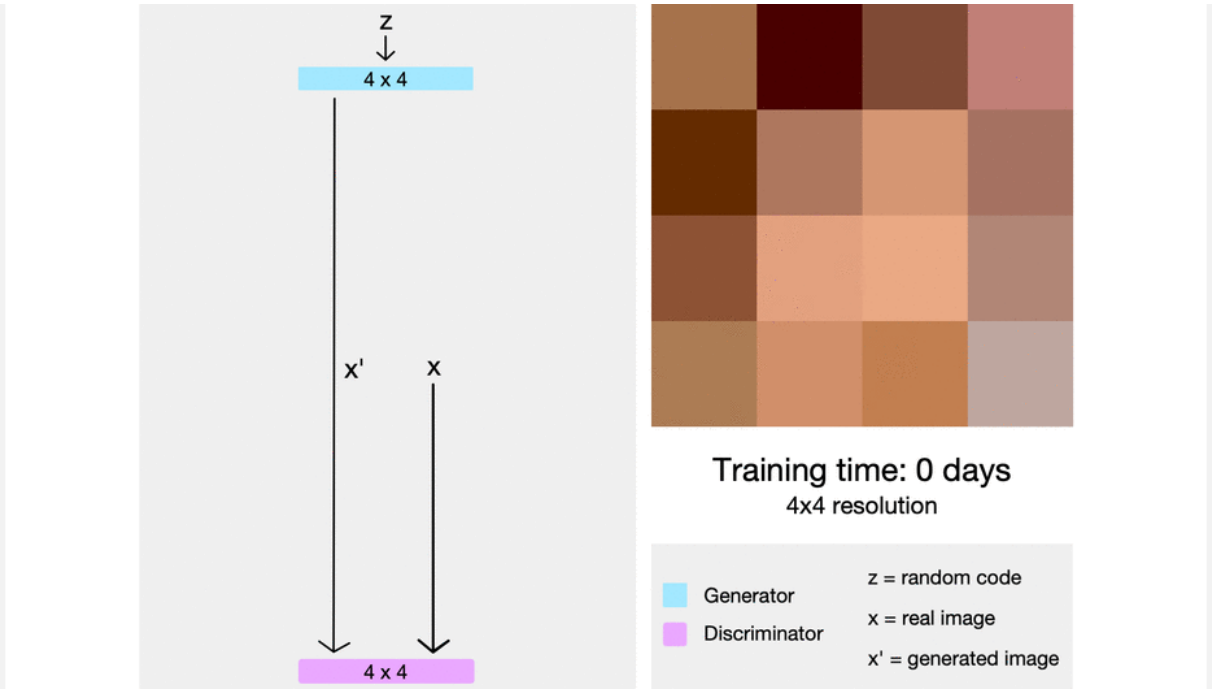
We will divide these applications into the following areas:

- Generate Examples for Image Datasets
- Generate Photographs of Human Faces
- Generate Realistic Photographs
- Generate Cartoon Characters
- Image-to-Image Translation
- Text-to-Image Translation
- Semantic-Image-to-Photo Translation
- Face Frontal View Generation
- Generate New Human Poses
- Photos to Emojis
- Photograph Editing
- Face Aging
- Photo Blending
- Super Resolution
- Photo Inpainting
- Clothing Translation
- Video Prediction
- 3D Object Generation

Different GAN models

ProGAN

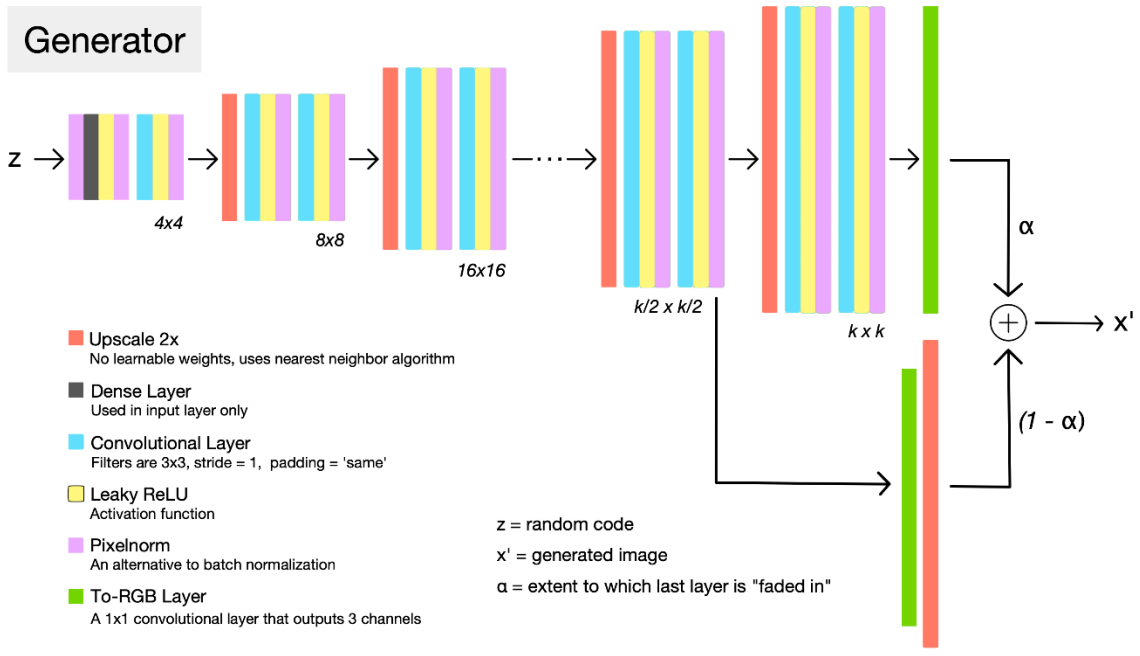
Also known as growing GAN. Instead of attempting to train all layers of the generator and discriminator at once — as is normally done — the team gradually *grew* their GAN, one layer at a time, to handle progressively higher resolution versions of the images.



This is a GIF image: https://towardsdatascience.com/progan-how-nvidia-generated-images-of-unprecedented-quality-51c98ec2cbd2

Figure 3: *The ProGAN starts out generating very low resolution images. When training stabilizes, a new layer is added and the resolution is doubled. This continues until the output reaches the desired resolution. By progressively growing the networks in this fashion, high-level structure is learned first, and training is stabilized.*

ProGAN architecture



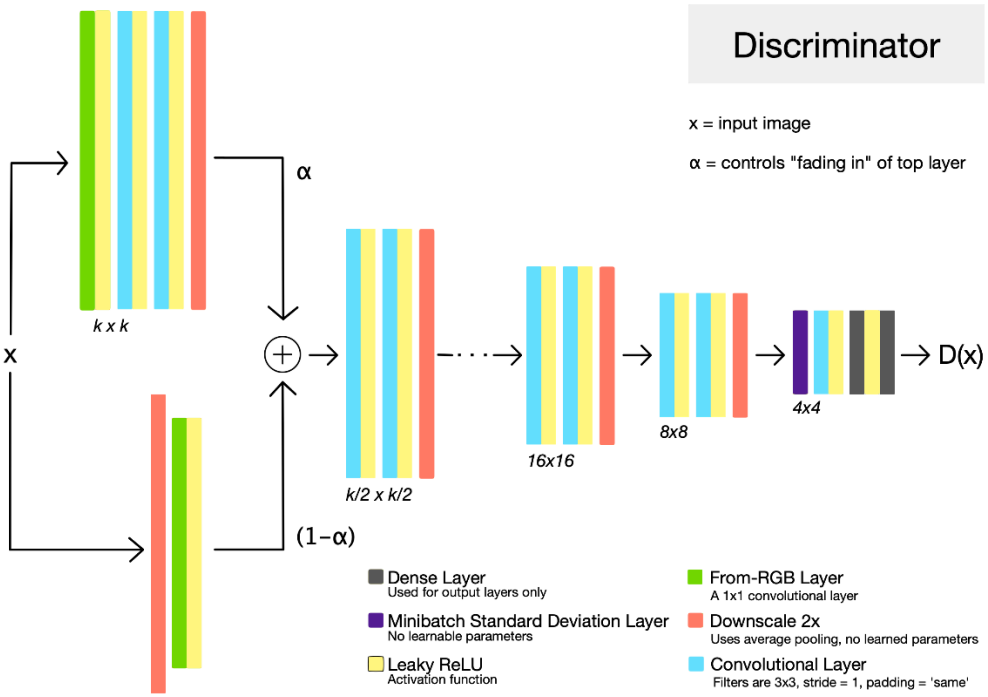


Figure 4: *The output of ProGAN when we smoothly interpolate over the input vector, z.*

This is a GIF image: https://towardsdatascience.com/progan-how-nvidia-generated-images-of-unprecedented-quality-51c98ec2cbd2

SNGAN

Full Form: Spectral Normalization for Generative Adversarial Networks.

Architecture:

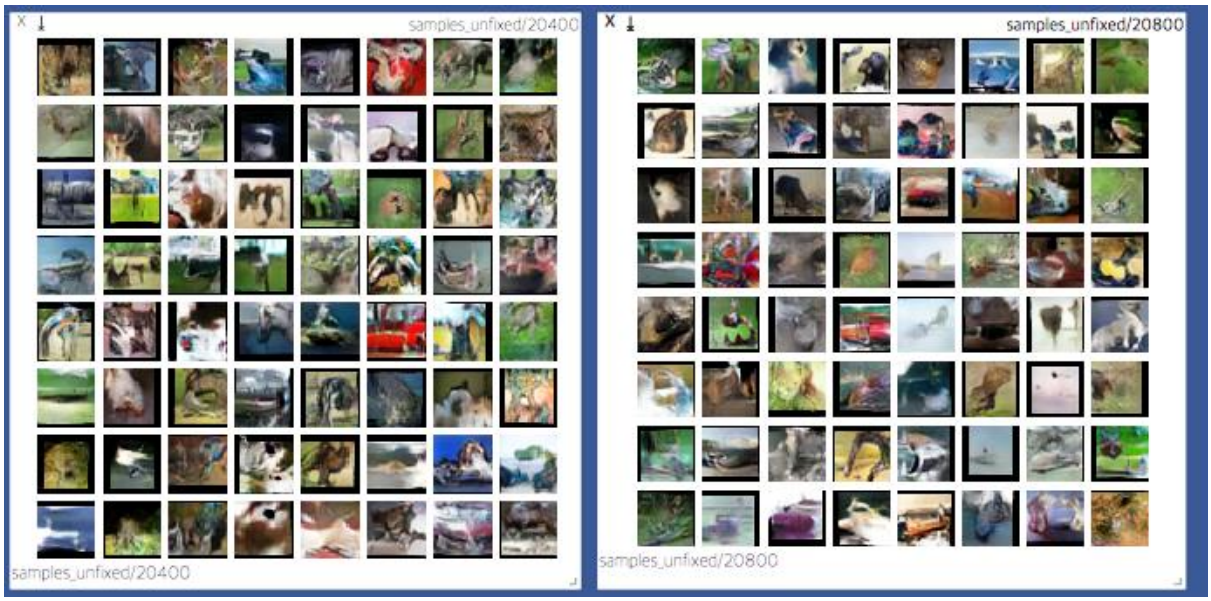
$z \in \mathbb{R}^{128} \sim \mathcal{N}(0, I)$
dense $\rightarrow M_g \times M_g \times 512$
4×4, stride=2 deconv. BN 256 ReLU
4×4, stride=2 deconv. BN 128 ReLU
4×4, stride=2 deconv. BN 64 ReLU
3×3, stride=1 conv. 3 Tanh

(a) Generator, $M_g = 4$ for SVHN and CIFAR10, and $M_g = 6$ for STL-10

RGB image $x \in \mathbb{R}^{M \times M \times 3}$
3×3, stride=1 conv 64 lReLU 4×4, stride=2 conv 64 lReLU
3×3, stride=1 conv 128 lReLU 4×4, stride=2 conv 128 lReLU
3×3, stride=1 conv 256 lReLU 4×4, stride=2 conv 256 lReLU
3×3, stride=1 conv. 512 lReLU
dense $\rightarrow$ 1

(b) Discriminator, $M = 32$ for SVHN and CIFAR10, and $M = 48$ for STL-10

Results



CramerGAN

Another recent member in the GAN family!

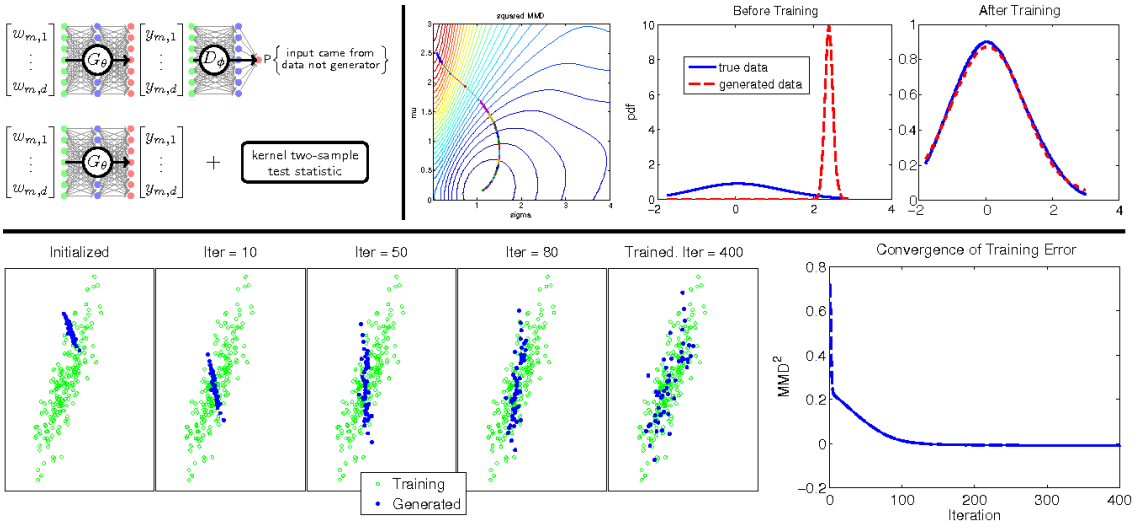
Cramer GAN starts by outlining an issue with the popular WGAN. It claims that there are three properties that a probability divergence should satisfy:

- Sum invariance
- Scale sensitivity
- Unbiased sample gradients

Figure 3: Generated right halves of the faces for WGAN-GP (left) and Cramér GAN (right). The given left halves are from CelebA 64x64 validation set (Liu et al., 2015).

MMD GAN

Maximum Mean Discrepancy GAN or MMD GAN is, you guessed it, an improvement of GMMN. Their major contributions come in the form of not using static Gaussian kernels to calculate the MMD, and instead use adversarial techniques to learn kernels. It combines ideas from the original GAN and GMMN papers to create a hybrid of the ideas of the two. The benefits it claims are an increase in performance and run time.



Key Take-Aways

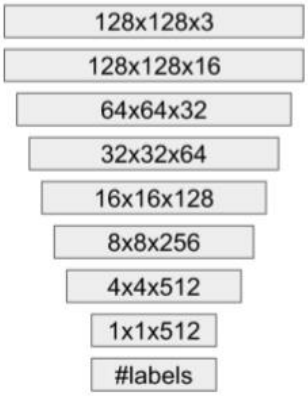
- Iteration on GMMN: Adversarial learned kernels for estimating MMD

Attribution network (VisNET)

Similar to the authorship attribution task in natural language processing we train an attribution classifier that can predict the source of an image: real or from a GAN model.

Approach:

Training a deep convolutional neural network supervised by image-source pairs {(I,y)}
where I ∼ **I** is sampled from an image set and
y ∈ Y is the source ground truth belonging to a finite set.
That set is composed of pre-trained GAN instances plus the real world.



Above Figure depicts an overview of our attribution network.

We implicitly represent:

- **Image fingerprints** as the final classifier features (the 1 × 1 × 512 tensor before the final fully connected layer) and
- **GAN model fingerprints** as the corresponding classifier parameters (the 1×1×512weighttensorofthefinalfullyconnectedlayer).

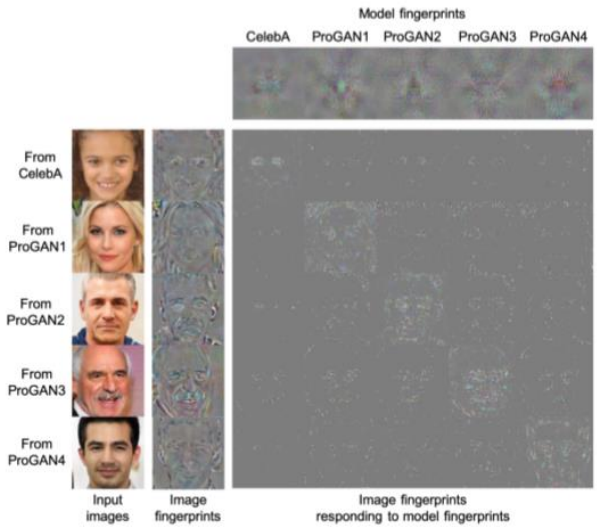


Figure 7. Visualization of model and image fingerprint samples. Their pairwise interactions are shown as the confusion matrix.

Dataset Testing

A Dataset of 5 Real images (see below), 5 ProGAN generated images and 5 SNGAN generated images was generated. The Test Dataset was then classified using classifier visNet pretrained model.
Here we can see that for 1 image, the prediction is wrong.

```
# Real
Real_Imgs_Path = '/content/drive/My Drive/DIP Project/RealDataset/'
n_Real = 5

# ProGAN
seed_ProGAN = 1
n_ProGAN = 5

# SNGAN
seed_SNGAN = 1
n_SNGAN = 5
```

Figure 5: *Dataset Generation Parameters*

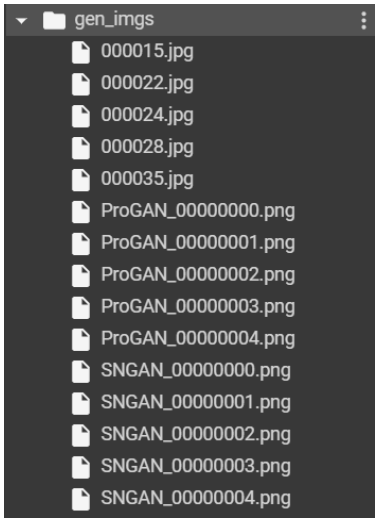


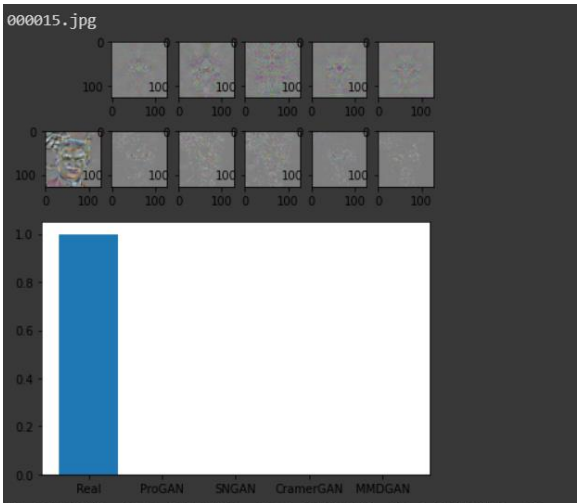
Figure 6: *Generated Test Dataset*

Real Images in Test dataset

Image Name	Target Class	Best Class predicted
000015.jpg	Real	CelebA_real_data
000022.jpg	Real	CelebA_real_data
000024.jpg	Real	CelebA_real_data
000028.jpg	Real	CelebA_real_data
000035.jpg	Real	CramerGAN_generated_data

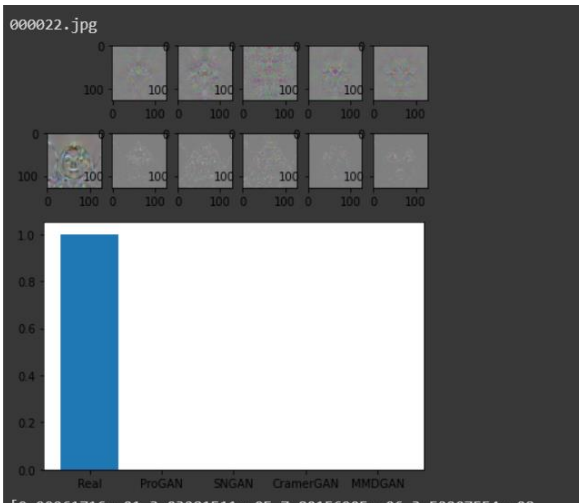
Below are 5 real images which were taken along with their classification.

000015.jpg



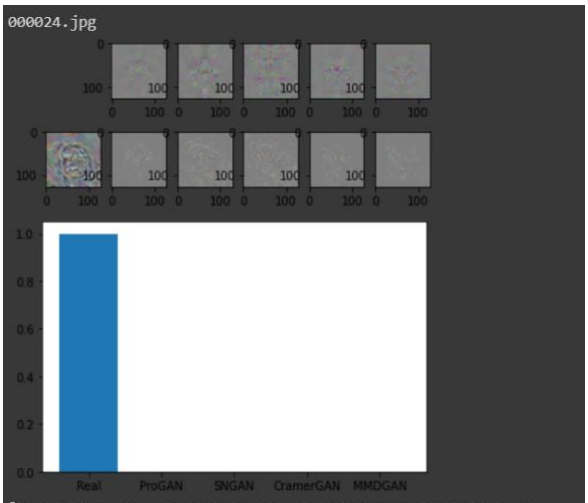

```
[9.99731848e-01 2.25207606e-04 3.19633409e-08 4.28170433e-05
 9.50978298e-08]
Best Class: CelebA_real_data
Target Class: Real
```

000022.jpg



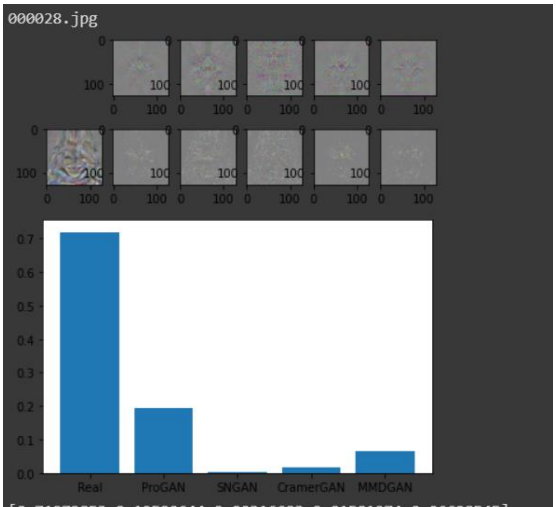

```
[9.99961716e-01 3.03281511e-05 7.80156905e-06 3.52207554e-08
 1.19079466e-07]
Best Class: CelebA_real_data
Target Class: Real
```

000024.jpg



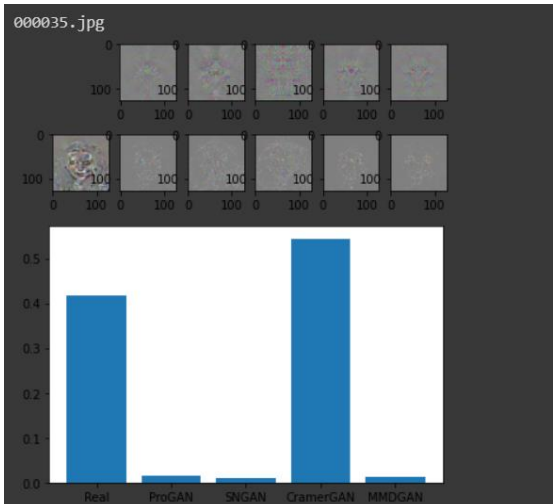

```
[9.98993863e-01 5.57512267e-05 5.94609611e-04 9.77798198e-05
 2.57996267e-04]
Best Class: CelebA_real_data
Target Class: Real
```

000028.jpg




```
[0.71972853 0.19500644 0.00316683 0.01581274 0.06628545]
Best Class: CelebA_real_data
Target Class: Real
```

000035.jpg

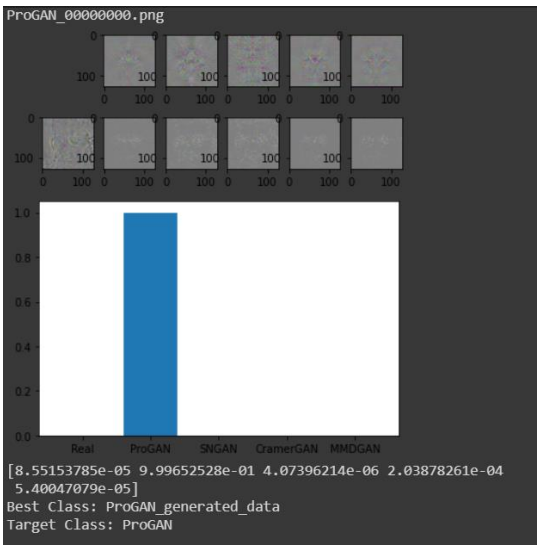


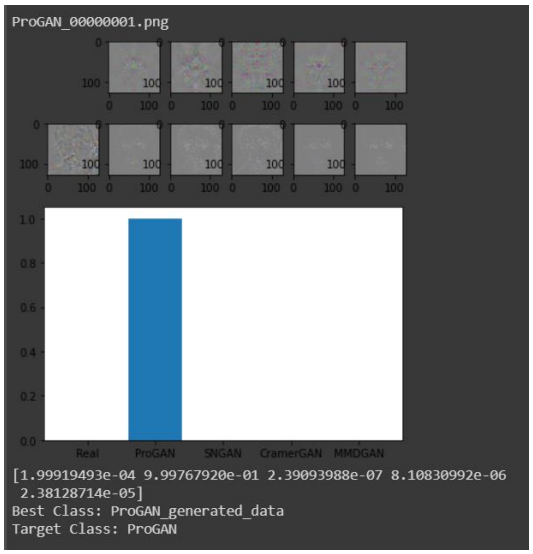

```
[0.41775713 0.01652481 0.00888118 0.54425791 0.01257896]
Best Class: CramerGAN_generated_data
Target Class: Real
```

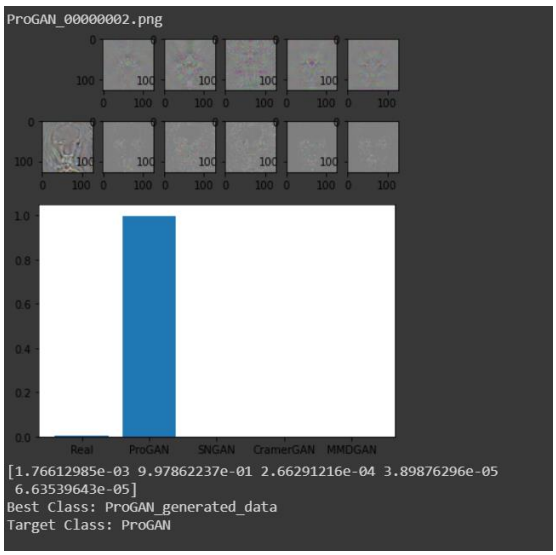
ProGAN generated Images in Test dataset

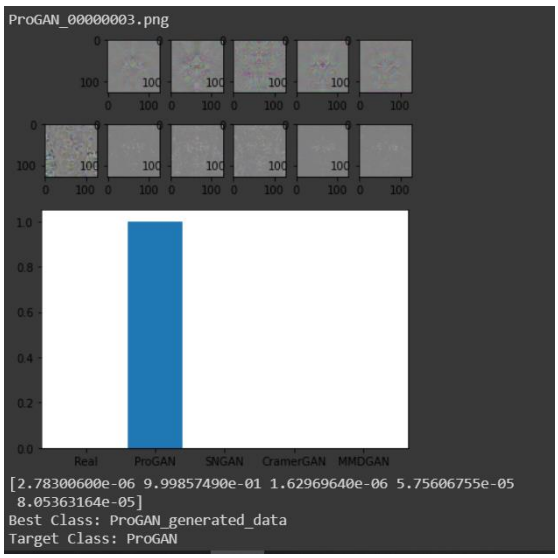
Image Name	Target Class	Best Class predicted
ProGAN_00000000.png	ProGAN	ProGAN_generated_data
ProGAN_00000001.png	ProGAN	ProGAN_generated_data
ProGAN_00000002.png	ProGAN	ProGAN_generated_data
ProGAN_00000003.png	ProGAN	ProGAN_generated_data
ProGAN_00000004.png	ProGAN	ProGAN_generated_data

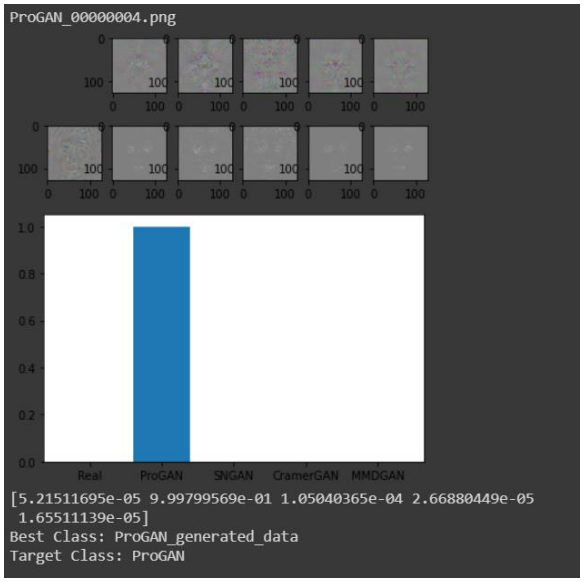
Below are 5 ProGAN generated images which were taken along with their classification.







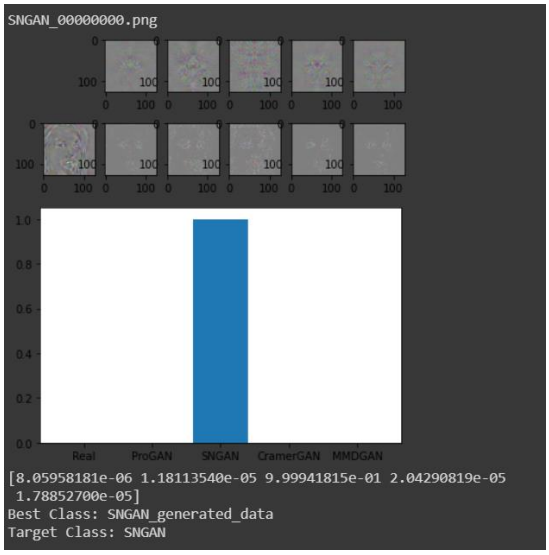


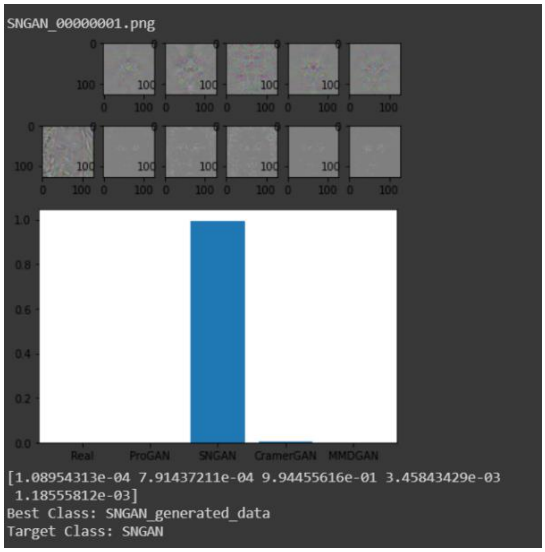


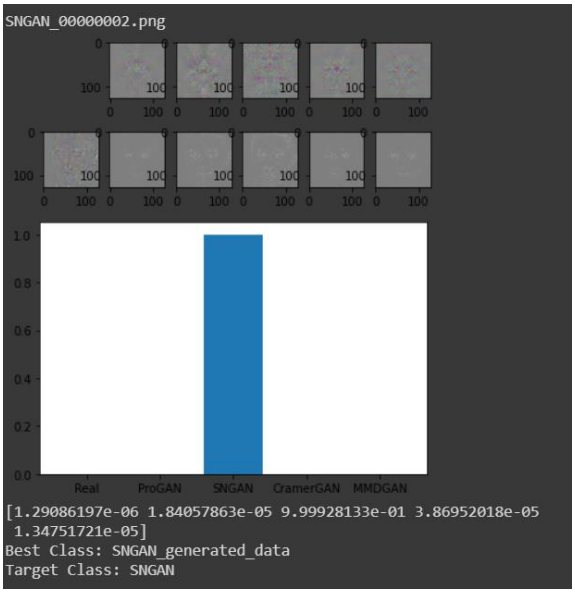
SNGAN generated Images in Test dataset

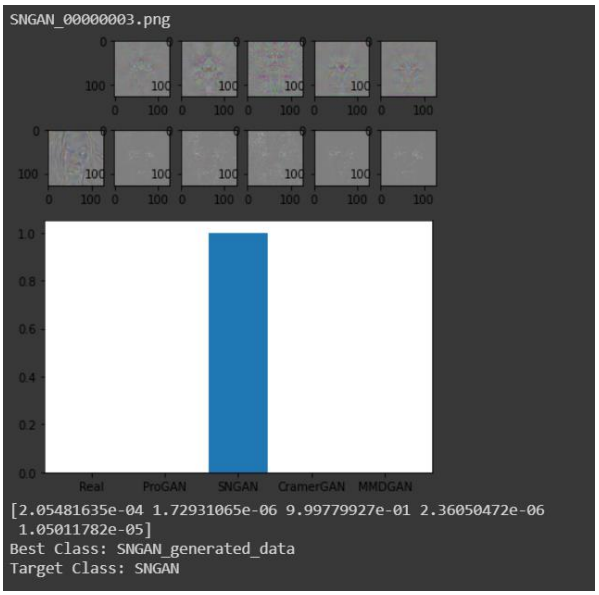
Image Name	Target Class	Best Class predicted
SNGAN_00000000.png	SNGAN	SNGAN_generated_data
SNGAN_00000001.png	SNGAN	SNGAN_generated_data
SNGAN_00000002.png	SNGAN	SNGAN_generated_data
SNGAN_00000003.png	SNGAN	SNGAN_generated_data
SNGAN_00000004.png	SNGAN	SNGAN_generated_data

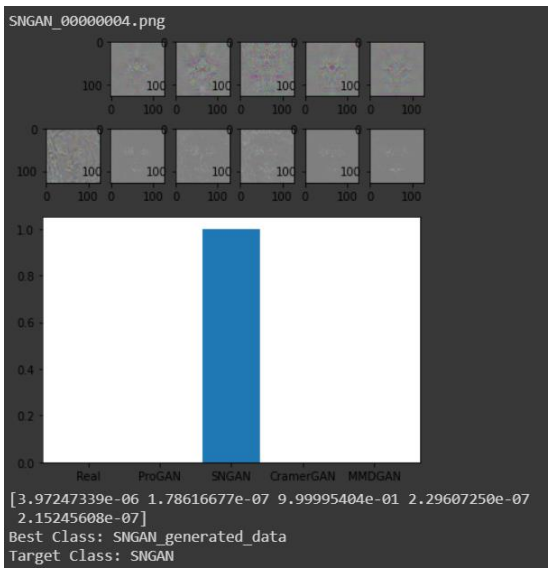
Below are 5 ProGAN generated images which were taken along with their classification.











Accuracy

Accuracy when 15 images in test dataset

```
Accuracy: 0.9333333333333333
Correct Predictions: 14 / 15
```

Accuracy when 150 images in test dataset

```
Accuracy: 0.92
Correct Predictions: 138 / 150
```

Improvement

The accuracy can be further improved by correcting the prediction in case of wrong predictions.
In the previous testing, we observed that one image got wrongly predicted. (refer real images table)

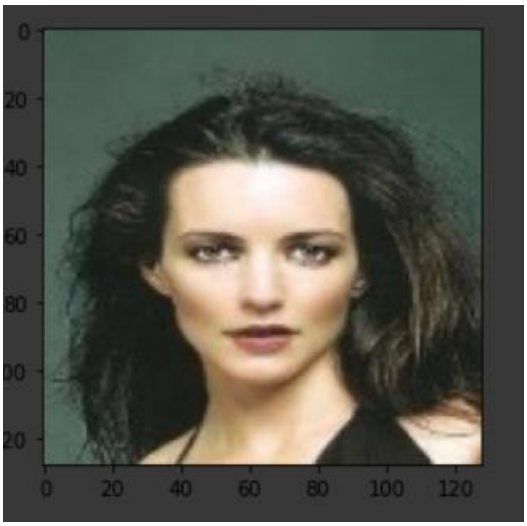


Figure 7: *Real Image Predicted as CramerGAN Generated*

So, to reduce such wrong predictions, we are employing an image augmentation based approach.

We take an input image, and apply some augmentations to it to generate more images but we choose the augmentations such that the inherent fingerprint data does not get destroyed.

E.g. we generate rotated versions of the input image for some angles and also a vertically flipped image.

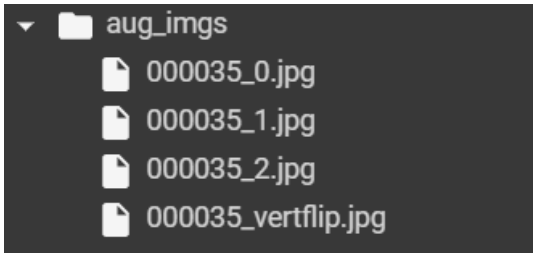


Figure 8: *Augmented Images Generated*

Then we predict for all the augmented images and assign the class which is most frequent in the predictions of augmented images of the input image.

```
Class Counts:
 dict_items([('CelebA_real_data', 4)])


Best Class: CelebA_real_data
```

Here, the once wrongly predicted image has now been correctly predicted.
Hence, this augmentation approach can be employed to further improve the accuracy of the model without retraining or changing the model.

References

[1] https://github.com/ningyu1991/GANFingerprints
[2] https://www.geeksforgeeks.org/generative-adversarial-network-gan/
[3] https://medium.com/@jonathan_hui/gan-whats-generative-adversarial-networks-and-its-a
[4] https://www.youtube.com/watch?v=Sw9r8CL98N0
[5] https://towardsdatascience.com/progan-how-nvidia-generated-images-of-unprecedented-quality-51c98ec2cbd2
[6] https://towardsdatascience.com/gan-objective-functions-gans-and-their-variations-ad77340bce3c
[7] https://machinelearningmastery.com/impressive-applications-of-generative-adversarial-networks/