Generating Photo & Cartoon Faces Using Generative Adversarial Network (GAN)

Shaily Sarker (ID: 17-35951-3)[a], *Kowshik Chakraborty (ID: 18-36200-1)*[a], *Saiful Islam (ID:18-37404-1)*[a]

[a] *Department of Computer Sciences, American International University-Bangladesh*

Abstract

Generative adversarial networks (GANs) are often utilized in modeling highly complex distributions for real-world data, especially images. GAN is about creating, like drawing a portrait or composing a symphony. This can be hard compared to other deep learning fields. It's much easier to spot a Monet painting than painting one, by computers or by people. But it brings us closer to understanding intelligence. GAN is an algorithmic architecture that uses two neural networks, pitting one against the opposite (thus the "adversarial") so as to get new, synthetic instances of information that may pass for real data. They're used widely in image generation, video generation, and voice generation. During a traditional GAN, the generator network would obtain a random "latent" vector as its input, and using multiple transposed convolutions, would alter that vector into a picture that might appear authentic to the discriminator. This latent vector are often thought of as a tensor representation of the image to the network. Generative Adversarial Networks [d1] are used for the task of image generation and have achieved impressive results. There's always a challenge to coach networks that generate large-scale images since they have a tendency to be huge and training needs lots of information.

Keywords: *GAN, NN, Generator Model, Discriminator Model, Image generation*

1. Introduction

A generative adversarial network (GAN) may be a quite AI algorithm. It's a sort of neural spec for generative modeling problems. The goal of a generative model is to review a set of coaching examples and learn the probability distribution that generated them. Generative Adversarial Networks (GANs) are then able to generate more examples from the estimated probability distribution. Generative modeling is an unsupervised learning task in machine learning that involves automatically discovering and learning the

regularities or patterns in computer file in such the way that the model may be wont to generate or output new examples that plausibly could are drawn from the first dataset.

Actually, GANs are an architecture for automatically training a generative model by treating the unsupervised problem as supervised and using both a generative and a discriminative model GANs are an inventive way of coaching a generative model by framing the matter as a supervised learning problem with two sub-models: the generator model that we train to come up with new examples, and also the discriminator model that tries to classify examples as either real (from the domain) or fake (generated). The 2 models are trained together in a very game, adversarial, until the discriminator model is fooled about half the time, meaning the generator model is generating plausible examples.

GANs are an exciting and rapidly changing field, delivering on the promise of generative models in their ability to come up with realistic examples across a spread of problem domains, most notably in image-to-image translation tasks like translating photos of summer to winter or day to nighttime, and in generating photorealistic photos of objects, scenes, and folks that even humans cannot tell are fake. Here, we used the dataset from the link "https://www.kaggle.com/srrrrr/photo2cartoon". Actually, this dataset could be a combination of photos and cartoon images. There have two directories within the "photo2cartoon" dataset. Training GANs involves training both a generator network and a discriminator network. The method involves both real data drawn from a dataset and pretend data created continuously by the generator throughout the training process. The discriminator is trained very like the other classifier defined by a deep neural network. By using that dataset, we've got develop GAN. First we selected variety of real images from the training set. Then we generated variety of faux images. This was done by sampling random noise vectors and creating images from them using the generator. At last, we trained the discriminator for one or more epochs using both fake and real. Training GANs involves training both a generator network and a discriminator network. The method involves both real data drawn from a dataset and pretend data created continuously by the generator throughout the training process.

The discriminator is trained very like the other classifier defined by a deep neural network. When the discriminator classifies this fake data then the discriminator is trained to assign this data to the "fake" class. The back-propagation algorithm makes it possible to use the derivatives of the discriminator's output with relation to the discriminator's input to coach the generator. The generator is trained to fool the discriminator, in other words, to form the discriminator assign its input to the "real" class. The training process for the

discriminator is thus much the identical as for the other binary classifier with the exception that the information for the “fake” class comes from a distribution that changes constantly because the generator learns instead of from a hard and fast distribution. the training process for the generator is somewhat unique, because it's not given specific targets for its output, but rather simply given an award for producing outputs that fool its (constantly changing) opponent.

2. Literature Review

Goodfellow et al. [1] suggested a Generative Adversarial Network (GAN) which is very strong model for modeling and generating images in an unsupervised way. The network had generator and discriminator that works adversarial to learn the generalization method of the given dataset. Large output image was created by Denton et al. [2] from interpolating reconstructed latent variables with GAN generator using a Laplacian pyramid. But in spite of great result of GAN, small difference in parameters and structure choice were likely to result in irregular, poor outcome and this made hard to train in purpose of real world applications. Though there are some limitations of GAN, researchers are working with various kinds of GAN. Deep Convolutional GAN (DCGAN) is the most famous network which is introduced by Radford et al. [3]. DCGAN improved the overall quality of generated images by adapting Convolutional Neural Network (CNN) into GAN architecture. It also gave an instruction of building a GAN network, making it easier to select between parameters. Larsen et al. [4], at the same time, was working with VAE/GAN and they developed it by modifying Vibrational Auto Encoder (VAE) [5] into GAN to maximize the advantages of both models. Discriminator of the GAN was used in that case for calculating feature-wise loss for training decoder of VAE. This thing increased the performance remarkably compared to the one with pixel-wise loss.

3. Proposed Model

A generative adversarial network (GAN) may be a machine learning (ML) model. By using two neural networks compete with one another to become more accurate in their predictions. These two neural networks are: a Generator and a Discriminator. The generator generates a "fake" sample given a random vector/matrix, and also the discriminator attempts to detect whether a given sample is "real" (picked from the training data) or "fake" (generated by the generator). Training happens in tandem: we train the discriminator for some epochs, then train the generator for some epochs, and repeat. This

manner both the generator and therefore the discriminator regain at doing their jobs. The GAN sets up a supervised learning problem so as to try to unsupervised learning, generates fake / random-looking data, and tries to work out if a sample is generated fake data or real data. GANs have very specific use cases and it is difficult to know these use cases when getting started.

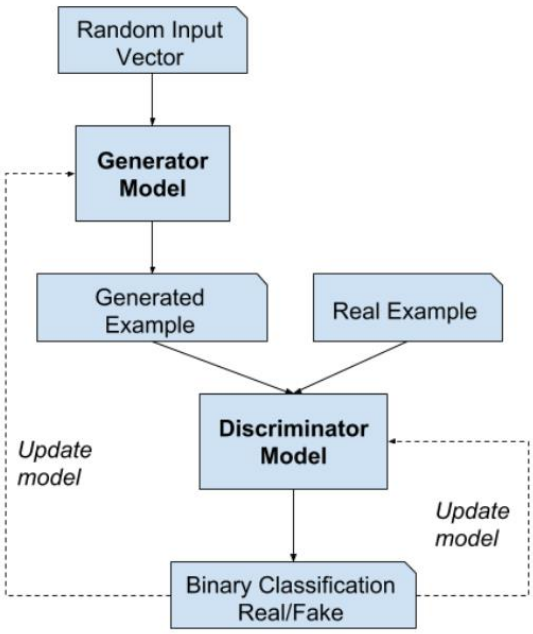

Figure 01: Generative Adversarial Network (GAN) Working Process

If we'd like to use a generative adversarial network (GAN) in an exceedingly dataset, then we are going to follow below steps-

- Step 1 — Select variety of real images from the training set.
- Step 2 — Generate variety of faux images. This can be done by sampling random noise vectors and creating images from them using the generator.
- Step 3 — Train the discriminator for one or more epochs using both fake and real images.

Generative Adversarial Networks (GANs) are then ready to generate more examples from the estimated probability distribution. Actually, The GAN sets up a supervised learning problem so as to try and do unsupervised learning, generates fake / random-looking data,

and tries to see if a sample is generated fake data or real data. We all know that supervised learning is usually ready to achieve greater than human accuracy after the training process is complete, and thus has been integrated into many products and services. Generative models supported deep learning are common, but GANs are among the foremost successful generative models (especially in terms of their ability to get realistic high-resolution images). GANs are successfully applied to a good kind of tasks (mostly in research settings) but still present unique challenges and research opportunities because they're supported scientific theory while most other approaches to generative modeling are supported optimization. The Generative Adversarial Network, or GAN, is an architecture that produces effective use of huge, unlabeled datasets to coach a picture generator model via a picture discriminator model. The discriminator model will be used as a start line for developing a classifier model in some cases.

4. Result:

In the code, we used 500 epochs to get better result. The result at first was not up to the mark. But the result became better after running epoch to epoch and at 500 epochs, we got a better result. Though the output result is not exactly same as the real one, but it is nearly closes to real image. The accuracy of detecting real image is approximately 96.32% and accuracy of detecting fake image is approximately 2.13%. That means the discriminator thought the fake images as the real images. The generator model succeeded to generate high quality images. But the discriminator also detected real images almost correctly. So, generator and discriminator model were trained well and gave a perfect result.

Figures are given below:

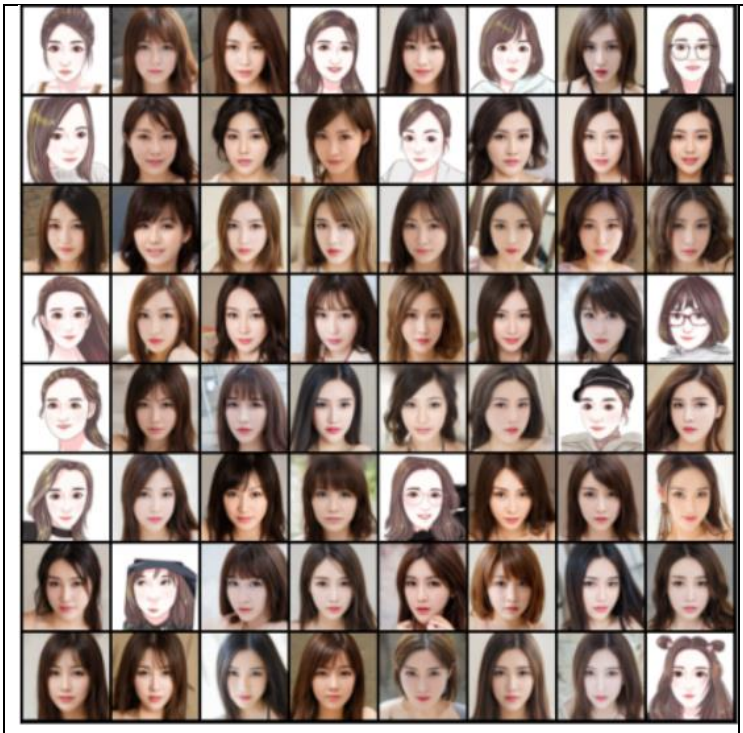

Figure 02: Real Images (From Dataset)

```
Epoch [489/500], loss_g: 4.4285, loss_d: 0.1406, real_score: 0.9358, fake_score: 0.0556
Saving generated-images-0489.png

Epoch [490/500], loss_g: 3.1849, loss_d: 0.1422, real_score: 0.8883, fake_score: 0.0195
Saving generated-images-0490.png

Epoch [491/500], loss_g: 4.3958, loss_d: 0.0793, real_score: 0.9727, fake_score: 0.0487
Saving generated-images-0491.png

Epoch [492/500], loss_g: 3.7642, loss_d: 0.0809, real_score: 0.9453, fake_score: 0.0225
Saving generated-images-0492.png

Epoch [493/500], loss_g: 3.4973, loss_d: 0.0611, real_score: 0.9710, fake_score: 0.0304
Saving generated-images-0493.png

Epoch [494/500], loss_g: 4.3228, loss_d: 0.0797, real_score: 0.9625, fake_score: 0.0386
Saving generated-images-0494.png

Epoch [495/500], loss_g: 3.0139, loss_d: 0.1441, real_score: 0.9041, fake_score: 0.0337
Saving generated-images-0495.png

Epoch [496/500], loss_g: 3.7364, loss_d: 0.0455, real_score: 0.9767, fake_score: 0.0213
Saving generated-images-0496.png

Epoch [497/500], loss_g: 5.9260, loss_d: 0.1779, real_score: 0.9930, fake_score: 0.1475
Saving generated-images-0497.png

Epoch [498/500], loss_g: 4.2561, loss_d: 0.1207, real_score: 0.9258, fake_score: 0.0370
Saving generated-images-0498.png

Epoch [499/500], loss_g: 3.7688, loss_d: 0.0546, real_score: 0.9689, fake_score: 0.0223
Saving generated-images-0499.png

Epoch [500/500], loss_g: 5.0251, loss_d: 0.0602, real_score: 0.9632, fake_score: 0.0213
Saving generated-images-0500.png
```

Figure 03: Accuracy(Real and Fake Score)

Figure 04: Generated Images (At 300 Epochs)

Figure 05: Generated Images (At 500 Epochs)

5. Discussion

Two neural network models were used for this GAN project, Generator and Discriminator. The generator generates "fake" images from a random vector/matrix, and also the discriminator finds whether a given sample image is "real" (taken from the training data) or "fake" (generated by the generator). Both generator and discriminator were trained for few epochs. At first, the generator was trained for few epochs (creating fake images) and faux images were passed to the discriminator. Real examples were also passed within the discriminator model in order that the model can compare fake images with real images and may detect whether the generated images were real or not. Basically, our main idea was to create the discriminator fool. It implies that the discriminator won't be able to detect fake images after some epochs (Quality of pretend images will bounce back from time to time by training generator model); it'll think about fake images as real. During this way, the discriminator was trained for few epochs and also the whole process was repeated for sure times so both the generator and discriminator is trained okay and obtain better at

doing their jobs. In discriminator, strides of two are wont to progressively reduce the scale of the output feature map and within the generator, leaky ReLU is employed which is different from the regular ReLU. It allows the pass of a tiny low gradient signal for negative values. As a result, it makes the gradients from the discriminator flows stronger into the generator. Rather than passing a gradient (slope) of 0 within the back-prop pass, it passes a tiny low negative gradient. The output of the discriminator may be a single number between 0 and 1, which might be interpreted because the probability of the input image is real (1) or fake (0). It absolutely was very difficult to coach the GAN since we were working with images and generating new images which don't exist. The result wasn't satisfactory initially, but after increasing epochs and decreasing the training rate (lr), it slowly began to provide a good result. The output got better from time to time and at last at 500 epochs, we got a quite impressive accuracy for both real and faux images.

6. Conclusion

Developing a GAN for generating images requires both a discriminator convolutional neural network model for classifying whether a given image is real or generated and a generator model that uses inverse convolutional layers to remodel an input to a full two-dimensional image of pixel values. Things we might try within the future: Training the GAN on an outsized training set while it won't be balanced the identical as our validation set. More tuning to balance the various components of the loss function.

References

1. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu,D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In Advances in neural information processing systems, pages 2672–2680, 2014.
2. L. Denton, S. Chintala, a. szlam, and R. Fergus. Deep generative image models using a laplacian pyramid of adversarial networks. In Advances in neural information processing systems, pages 1486–1494, 2015.
3. A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434, 2015.
4. A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther. Autoencoding beyond pixels using a learned similarity metric. arXiv preprint arXiv:1512.09300, 2015.
5. D. P. Kingma and M. Welling. Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114, 2013.