*A Progress Report*

*on*

# CELEBRITY FACE GENERATION USING DCGAN

*carried out as part of the course CS1634 Submitted by*


***Yashu Singla***

***189301132***

***VI-CSE***

***Gopal***

***189302160***

***VI-CSE***


*in partial fulfillment for the award of the degree*

*of*

**BACHELOR OF TECHNOLOGY**

In

**Computer Science & Engineering**

MANIPAL UNIVERSITY JAIPUR


**Department of Computer Science & Engineering,**
**School of Computing and IT,**
**Manipal University Jaipur,**


***April 2021***

## ABSTRACT

Face synthesis has achieved advanced development by using generative adversarial networks (GANs). Existing methods typically formulate GAN as a two-player game, where a discriminator distinguishes face images from the real and synthesized domains, while a generator reduces its discriminativeness by synthesizing a face of photorealistic quality. Their competition converges when the discriminator is unable to differentiate these two domains. We will use Deep Convolutional Generative Adversarial Network (DCGAN) which has proven to be a great success in generating images.

## Table of Contents

# 1. INTRODUCTION

Face generation is the task of generating (or interpolating) new faces from an existing dataset. Here we present a machine learning model which generates images based on the feature provided by the training images. For our objective adversarial networks can learn good representations of images for supervised learning and generative modeling (Radford, Metz & Chintala, 2016).

Generative Adversarial Networks, or GANs for short, are an approach to generative modeling using deep learning methods, such as convolutional neural networks.

Generative modeling is an unsupervised learning task in machine learning that involves automatically discovering and learning the regularities or patterns in input data in such a way that the model can be used to generate or output new examples that plausibly could have been drawn from the original dataset.

GANs are a clever way of training a generative model by framing the problem as a supervised learning problem with two sub-models: the generator model that we train to generate new examples, and the discriminator model that tries to classify examples as either real (from the domain) or fake (generated). The two models are trained together in a zero-sum 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 generate realistic examples across a range of problem domains, most notably in image-to-image translation tasks such as translating photos of summer to winter or day to night, and in generating photorealistic photos of objects, scenes, and people that even humans cannot tell are fake.

## 1.1 MOTIVATION

These are the following Motivations:

- Acquiring labeled data is a manual process that takes a lot of time. Our project model does not require labeled data; it can be trained using unlabeled data as it learns the internal representations of the data.

• One of the best things about our model is that it generates data that is similar to real data. Because of this, it has many different uses in the real world.

• Our model learns the internal representations of data. And it can learn messy and complicated distributions of data.

## 2. REQUIREMENT ANALYSIS

### 2.1 FUNCTIONAL REQUIREMENTS

Our project is designed to generate different types of Celebrity Faces. This project will help the cartoon industry to use different types of Celebrity Faces for their Celebrity Movies. Our model is implemented by DCGAN in which there are two models Generator and Discriminator.

These are the following Training Steps:-

- Celebrity Image Dataset is used for training our model.
- First, Discriminator is trained by using Real images which are having ground truth of 1 and Fake images with ground truth of 0 generated by Generator.
- Then Generator is trained with ground truth 1.
- After several epochs, our Generator will be trained.
- After every 5 Epoch, model weights are saved in a folder so that we can access results of any intermediate Epochs.
- When Noise is provided to Generator and Celebrity Faces are generated by the Generator.

### 2.2 NON-FUNCTIONAL REQUIREMENTS

#### 2.2.1 PERFORMANCE

Our model is trained in such a way that it can generate Real like Celebrity images with a training of 70 Epochs which will take approximately 3 hours. Earlier, we have also trained our model on 'ADAM' optimizer, but it was generating Real like Celebrity images with a training of 150 Epochs.

#### 2.2.2 EXTENSIBILITY

Our model is built in such a way that it allows easy modifications and extensions. By using Conditional GAN our model can generate images according to user's preferences.

#### 2.2.3 MAINTAINABILITY

The Project source code is divided into two separate repositories in which both Generator and discriminator are trained separately. Any programmer can do changes according to his choice.

### 2.2.4 USABILITY

Our project is designed to generate different types of Celebrity Faces. This project will help the cartoon industry to use different types of Celebrity Faces for their Celebrity Movies. Our model is implemented by DCGAN in which there are two models Generator and Discriminator.

## 2.3 USE CASES

Some of the newly discovered uses cases of GANs are:

### Security:

Artificial intelligence has proved to be a boon to many industries, but it is also surrounded by the problem of Cyber threats. GANs are proved to be a great help to handle the adversarial attacks. The adversarial attacks use a variety of techniques to fool deep learning architectures. By creating fake examples and training the model to identify them we counter these attacks.

### Generating Data using GANs:

Data is the most important key for any deep learning algorithm. In general, more is the data, better is the performance of any deep learning algorithm. But in many cases, such as health diagnostics, the amount of data is restricted, in such cases, there is a need to generate good quality data. For which GANs are being used.

# 3 SYSTEM DESIGN

## 3.1 Datasets

The Celebrity dataset consists of over 10k identities and 200k total images. All images are originally of size 160X160 pixels. They are rescaled to 64X64 pixels.

## 3.2 Model Implementation

### 3.2.1 Discriminative Model Implementation

For feature extraction 64 filters of size 4X4 were applied on the original image. Batch normalization was performed on the layers to reduce noise and to generalize the features.

Again, the resulting layers were stacked on top of each other and 128 filters and 256 filters of size 4X4 each were applied. Each convolution layer was followed by average pooling and batch normalization.

The layer was flattened and dropout with probability 0.5 was applied. A Dense network was stacked on top of the convolutional network with an output of 1 which determined whether the image fed into discriminator was real or fake. After several epochs, our Generator will be trained. After every 5 Epoch, model weights are saved in a folder so that we can access results of any intermediate Epochs. The discriminator model was the classification model which classified the images as real or fake.

### 3.2.2 Generative Model Implementation

A generator network maps single vectors to images of shape (64, 64, 3). The features of generative models are same as the discriminator except that it applies convolution with a fractional stride (convolution transpose) (Chollet, n.d.).

## 3.3 Optimizing the Model

Weights are updated as to maximize the probability that any real data input x is classified as belonging to the real dataset, while minimizing the probability that any fake image is classified as belonging to the real dataset. In more technical terms, the loss/error function used maximizes the function D(x), and it also minimizes D(G(z)).

Furthermore, the generator function maximizes D(G(z)).

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

Since during training both the Discriminator and Generator are trying to optimize opposite loss functions, they can be thought of two agents playing a minimax game with value function V(G,D).

Models were trained for 70 epochs with a batch size of 128 for Celebrity dataset.

## 3.4 System Specification

Programming Language : Python3
Framework Used : TensorFlow
Development Platform : Jupyter Notebook
Training Time : 4 hours for Celebrity dataset.

# 4. Work Done

## 4.1 Details

As of now we have completed the following modules:

- We have completed Data pre-processing part successfully.
- We have completed Generator Model successfully.
- We have completed Discriminator Model successfully.
- Also, we have successfully combined both Generator and Discriminator Model.
- Also, we have successfully trained both the models.

## 4.2 Results

### Fake samples generated after training

### Images on 1st Epoch

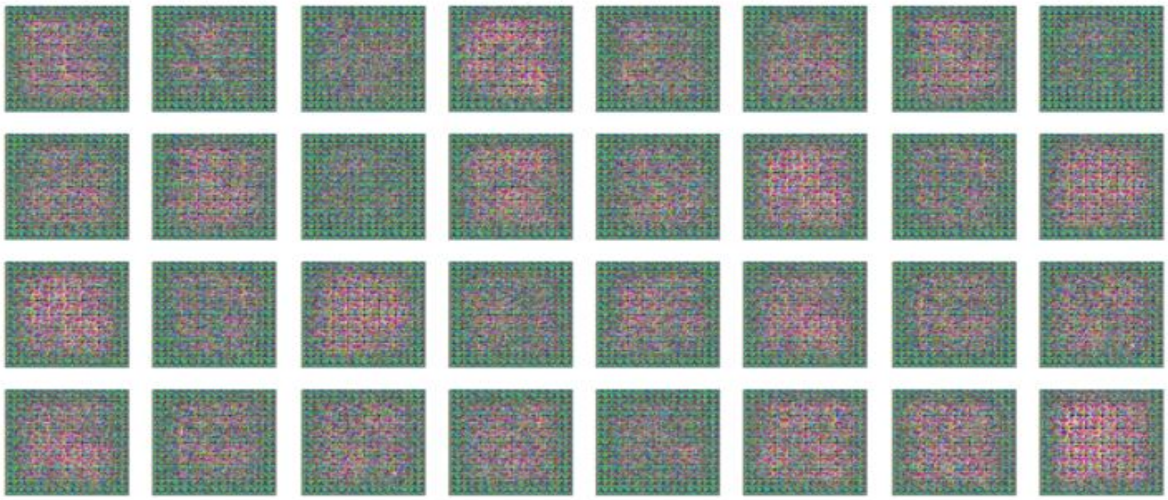

**Figure 1**

**Images after 45 Epochs**

**Figure 2**

### Images after 70 Epochs

**Figure 3**

### 4.3 Individual Contribution

### Gopal:

- Data pre-processing
- Discriminator
- Generator

### Yashu Singla:

- Combination of models
- Training of both the models

## 5. Conclusion and Future Work

We propose a more stable set of architectures for training generative adversarial networks, and we give evidence that adversarial networks learn good representations of images for supervised learning and generative modelling. There are still some forms of model instability remaining - we noticed as models are trained longer, they sometimes collapse a subset of filters to a single oscillating mode.

Further work is needed to tackle this from of instability. We think that extending this framework to other domains such as video (for frame prediction) and audio (pre-trained features for speech synthesis) should be very interesting. Further investigations into the properties of the learnt latent space would be interesting as well.

## 6. Reference

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in sAdvances in Neural Information Processing Systems, 2014, pp. 2672–2680.

[2] M.-Y. Liu and O. Tuzel, "Coupled generative adversarial networks," in Advances in neural information processing systems, 2016, pp. 469–477.

[3] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," in Advances in Neural Information Processing Systems, 2016, pp. 2234–2242.

[4] M. O. Turkoglu, L. Spreeuwers, W. Thong, and B. Kicanaoglu, "A layer-based sequential framework for scene generation with GANs," in Thirty-Third AAAI Conference on Artificial Intelligence, Honolulu, Hawaii, United States, 2019.

[5] H. Wu, S. Zheng, J. Zhang, and K. Huang, "GP-GAN: Towards realistic high-resolution image blending," arXiv preprint arXiv:1703.07195, 2017.