Feature Learning & Image Generation using Deep Convolutional GANs

ABSTRACT

With the advancement in the Machine Learning domain, supervised learning with the convolutional network has always been in Limelight. Unsupervised learning has been less highlighted in this course of action. Deep Convolutional Generative Adversarial networks (DCGANs) are CNN with an architectural limitation that makes them an ideal candidate for unsupervised learning. GANs are part of the Generative model family that can create/generate new content. We have discussed the operation of both the generator and the discriminator in order to achieve Feature Learning and Image Generation. Using these models, we are attempting to produce new designs and art pieces for a fashion set based on the current MNIST fashion dataset.

Keywords - CNN, DCGANs, Supervised learning, Unsupervised learning

INTRODUCTION

One of the most intriguing applications of GANs is image-to-image conversion. It is a thriving and rapidly evolving technology that protects the theory underlying the Generative model by creating certain real-world examples across a wide variety of issue domains. Changing pictures from winter to summer, and from day to night mode. Furthermore, GANs aid in the generation of more than actual images of fictitious things, situations, and people. DCGANs for clothes may be extremely useful for generating, developing, and manufacturing new trends and designs based on existing designs, as well as wholly unique new artworks. Machines can create Natural Images, thanks to the strong collaboration of GANs and Adversarial networks. These visuals are more linked and resemble real-world data. Along with that, one advantage of GAN is that it can produce more data, which means that we can have entirely fresh and unique data/images based on existing datasets. GANs have two main networks:

- The generator model generates new data objects.
- The discriminator model functions as a classifier, identifying phony friends in the group.

Working of GANs:

- The generator produces an image from random numbers.
- The discriminator is given photos that are both dummy and genuine.
- Discriminator functions as a Cop from the Thief-cop anomaly, distinguishing between counterfeit and genuine entities. It returns a prospect with a value ranging from 0 to 1.
- 0(zero) denotes fake data, which is bogus data, while 1(one) validates authentic photographs.

Double feedback Loop:

The GAN model has a significant structure in which double-feedback plays a key role. The discriminator is linked to veracious pictures, while the generator is linked to the discriminator, delivering it look-alike images that appear real but are actually fraudulent. To differentiate between counterfeit and genuine photographs, the discriminator must always have access to the authentic dataset and real images. The finished product is nearly identical to the original dataset, but not a full clone.

LITERATURE SURVEY

A. Image Creation Using K-means Clustering [2] provides a colour-based picture generating approach based on the K-means clustering technique, which is an iterative process used to divide an image into k clusters. The clustering procedure is completed by extracting pixel values from the original picture and grouping them into K-partitions based on their colours. The clustered colours are then blended to produce the final output, which is a target image. Although this approach works well for bigger values of K, it takes a long time to run, and it also performs poorly when the number of colours in the original image is huge.

B. Variational GANs

VAE, or Variational AutoEncoders, are largely based on the same principle of mapping the original picture to latent space, which is done via the encoder, and re-constructing values in latent space into their original dimension, which is done through the decoder. The photos are a little washed out/blurry, according to [3]. This is a well-known issue with vanilla VAEs.

C. StyleGANs

This article describes a Style Transfer Literature-based Generator architecture. By enabling unsupervised separation of high-level characteristics and dependent variation in spawning photographs, instinctive knowledge is provided. This novel structure results in extremely intrinsic, scale-specific synthesis control. In terms of traditional distribution efficiency norms, the most recent generators exceed the most recent technologies. This significantly increases interpolation quality and helps to pinpoint probable causes of dispersion. The author provided two novel automated approaches for assessing interpolation efficiency and disentanglement that are applicable to any generator design. Finally, they present a new collection of varied and high-quality human faces. Some Image Generated are approaching the boundaries of the training data, since the most unpleasant flaws in many photos are the severe compression artefacts inherited from the low-quality training data.

D. CycleGAN

The primary principle of image-to-image conversion is to master the chart between input and output images by using previously trained photos. Despite the fact that paired training data cannot be utilised for undeniable purposes. The author proposes a strategy for converting a source domain X to a target zone Y without providing an example pair. The purpose of employing adversarial loss is to investigate the mapping G: X > Y such that the image distribution in G(X) is indistinguishable from the Y distribution. They also invented reverse mapping, which turns out to be under-constrained. Quality results are offered for several operations that do not need paired training data, such as transferring collection styles, altering items, improving images, and so on. Quantitative comparisons to some previous approaches show these methodologies are superior. CycleGANs ‘Generator model couldn‘t handle more complex and extreme transformations, especially geometric changes. The distribution characteristics of training datasets are responsible for certain failures. The findings obtained with paired training data and those obtained with our unpaired approach are too apart.

E. Conditional GANs

Generative Adversarial Nets are a new model generation technique that was just developed. The conditional character of GANs is described in this study. Conditions on both the generator and the discriminator can be applied using a single set of data. Based on class labels, this model is proved to be capable of producing MNIST digits. It also explains how to upgrade a multi-modal model. Additionally, there are preliminary examples of photo tagging that offer evidence of descriptive tags not included in the training label. For the work, more advanced models, as well as a more extensive investigation of model performance and attributes, are necessary. The author has only utilised each tag individually; but, in the future, they want to widen their study by tagging many tags at once. A collaborative training system to learn the language model may also be developed for future work.

F. Progressive Growing GANs

This study expresses a strong desire to improve or up-skill the generator and discriminator resolutions. GANs begin with poor resolution, but when further layers and the rainfall model are added, more accurate information becomes available. This speeds up and stabilises the process, resulting in photographs of unrivalled quality, such as the celebrity shot taken at epoch 10242. They also developed a simple strategy for increasing variance in produced pictures and obtained an unsupervised CIFAR10 inception score of 8.80. They've also gone through a few technical details that are critical for preventing unhealthy competition between the generator and the discriminator. In addition, a new metric for quantifying GAN performance, incorporating picture quality variance, has been presented. As an added bonus, the study describes the creation of a higher-quality version of the CELEBA dataset. In certain ways, this paper embraces realism.

Table I : Previous paper learning & Conclusions

S.No	Paper Title	Learning	Conclusion
1	Image Generation by using K-Means Clustering	Color-based Image Generation used to partition an image into K-Clusters.	Time Consuming, Slow
2	Variational GANs	Mapping Original Images to Latent Image using Encoders & Decoders.	Washed Out/ Blurry images
3	StyleGANs	Generate new Human faces with both diversity & High-Quality.	Severe compression artefacts
4	CycleGANs	Demonstrate the connective connection between Input and Output when considering comparable picture pairings, such as vision class and graphics.	1. Can't handle extreme transformation with Geometric shapes. 2. Paired & Unpaired approach findings were way apart.
5	Conditional GANs	These GANs work on certain conditions that trigger for Generator & discriminator.	Collaborative training to learn language model needs to be considered.
6	Progressive Growing GANs	The Maine idea behind this, is to increase the resolution of both the generator and discriminator.	Lack semantic sensibility, dataset-dependent constraint understanding. The microstructure of images can be enhanced.
7	Evolutionary GANs	Evolutionary Generative Adversarial networks (E-GAN), a new GAN paradigm for stable GAN training and improved generative efficiency.	Future research will concentrate more on the understanding the relationship between Discriminator & Generator, as well as improving generative efficiency.

In terms of semantic awareness and comprehending dataset-dependent restrictions, such as some things being straight rather than curved, much remains to be desired. Furthermore, the microstructure of the photos can be improved.

G. Evolutionary GANs

GANs have played an important role in explaining Generative models. While dealing with these machines right now, the inability and mode collapse of GANs has been a major source of concern. The development of GANs is highlighted in this work in order to make them more stiff and efficient. For evolution, Generator employs unique adversarial training as an operation. Generators react to their surroundings, and some characteristics are taken into account while determining divergence and quality. E-GANs function entirely by picking the best offspring. E-GANs give higher efficiency and performance satisfaction with datasets, according to a variety of evidence.

The relationship between Generator and Discriminator will be examined in future work in order to improve performance.

DESIGN/MODELS

The design structure demonstrates how code works and how the entire prototype works together. We'll start by importing all of the necessary libraries for reading, writing, and creating networks, as well as training and testing. Following that, we load the MNIST fashion dataset, which has 70,000 samples of various types of clothes such as T-shirts, Tops, Trousers, Pullovers, Dresses, Coats, Sandals, Shirts, and so on. Some preprocessing procedures are necessary to transform the dataset into a more acceptable format that our model can interpret. Creates a generator and discriminator model with several interconnected layers to get the most out of the two, i.e. feature learning and picture creation for producing and verifying new fashion sets.

Both models have some sort of loss function attached to them. The loss function for the generator is lower if the generator model can fool the discriminator model. If the discriminator is properly able to discriminate between authentic and fraudulent data, the generator is penalized, and the discriminator suffers a significant or no loss. Finally, the photographs that are created are saved. Various fashion businesses can utilize these new photos, or fashion sets, to create new designs and artworks.

METHODOLOGIES

The Deep Convolutional Generative Adversarial Network proposed in this study is trained using Fashion MNIST pictures. T-shirts/tops, Trousers, Pullovers, Dresses, Coats, Sandals, Shirts, Sneakers, Bags, and Ankle Boots are among the photos. Using this current collection, a fresh new fashion set based on existing data is developed, as well as innovative distinctive new designs. During training, the generator model creates bogus pictures in an attempt to trick the discriminator. If the discriminator model properly classifies false and real photos, it will have a lower loss value. Both the generator and the discriminator are linked. The goal is to maximize discriminator accuracy while also generating fresh new designs with Generator.

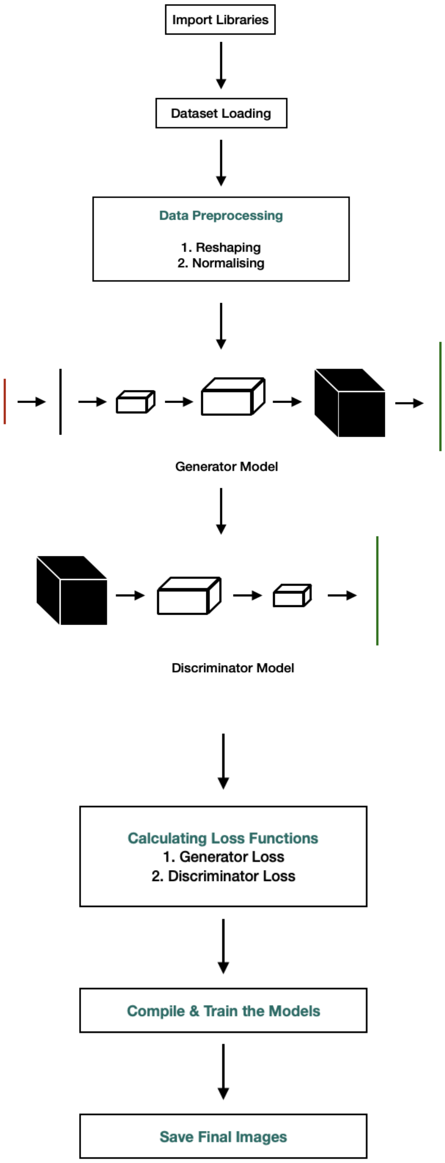


Fig 1: Flow diagram of the code

1. Dataset

The dataset has 70,000 samples in total, 60,000 of which will be utilized as training samples and the remaining samples will be used throughout the testing process. Each specimen is a 28 X 28 grayscale image that is numerically labeled from 0 to 9 based on the fact that 0 stands for T-shirt/top, 1 stand for Trouser, 2 stands for Pullover, 3 stands for Dress, 4 stands for Coat, 5 stands for Sandal, 6 stands for Shirt, 7 stands for Sneaker, 8 stands for Bag, and 9 stands for an Ankle boot. Zalando is the creator of the Fashion MNSIT dataset, which was created specifically for use in the Machine Learning domain and is based on the original MNIST dataset. Every structure and size requirement remains the same for both splits.

Every image has a height to width ratio of 28 X 28 pixels and takes up a total of 784 pixels. Every pixel is assigned a unique number that confirms the darker or lighter nature of that pixel. A higher score indicates a preference for the darker side. The training and test datasets include a total of 785 columns. The first column is used to identify the kind of clothing, while the remaining columns carry the pixel value of the image.

- Decompose x into x = i * 28 + j to determine the pixels in the picture. i and j are numbers ranging from 0 to 27. Pixels in row i and column j make up the 28 x 28 matrix.
- Pixel 31, represents the pixels in the fourth column from the left and the second row from the top.

2. Discriminator Model

Discriminator is a simple binary classifier that helps determine if a picture is authentic or fraudulent. The most acceptable discriminator aim is to categorize the images, as in the Cop-Thief Analogy. As a cop attempts to detect fake and real cash in order to apprehend the thief, a discriminator will classify fraudulent photos from real ones based on its learning.

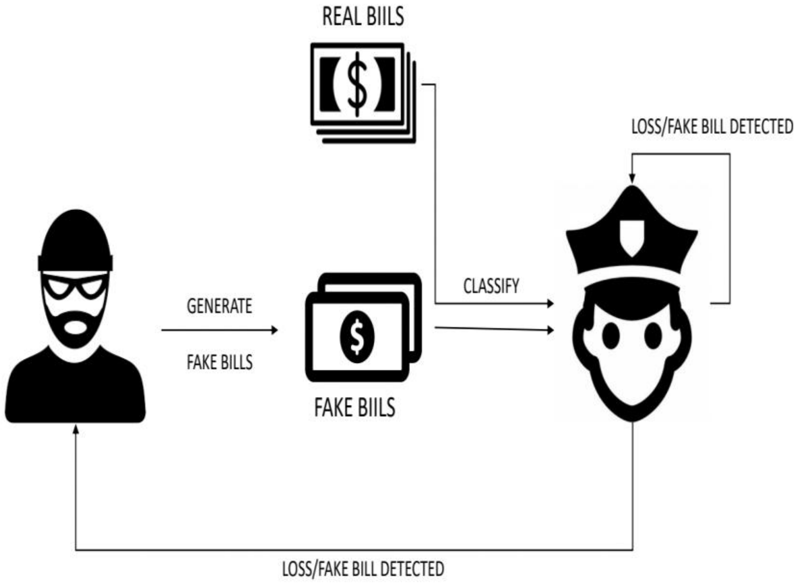


Fig 2: Thief-Cop Anomaly

The Discriminator model is a CNN that includes a Leaky ReLU Layer, a Dropout Layer, a Flatten Layer, and an Input and Output Layer. The leaky ReLU layer aids in dealing with negative column values; any improper value is converted to an acceptable value by multiplying it by tiny alpha values such as 0.001, etc.

This alpha value can be changed based on the results of the analysis. The dropout layer aids in the prevention of overfitting by rendering a few layers fully inoperable by disconnecting them. Flattening the layer, in the end, aids in achieving the desired output dimension.

At this layer, multidimensional intermediate outputs from several layers are eventually converted to single dimensional output.

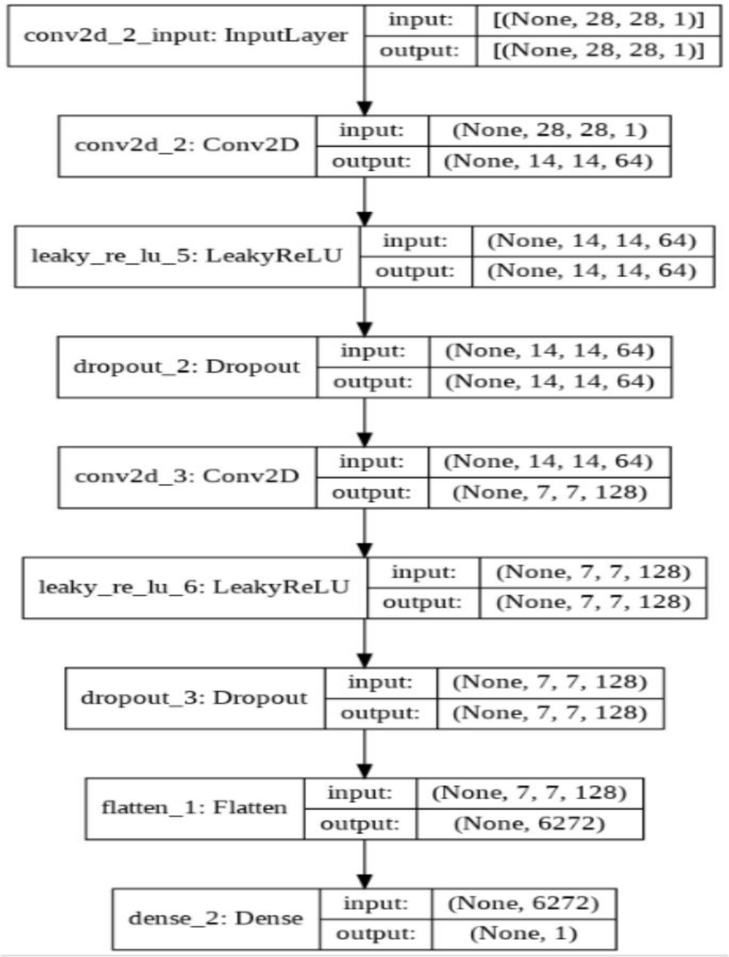
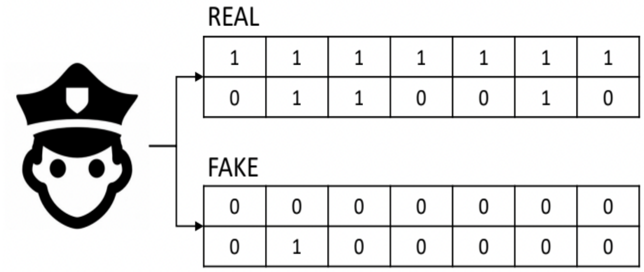

conv2d_2_input: InputLayer
input: [(None, 28, 28, 1)]
output: [(None, 28, 28, 1)]
conv2d_2: Conv2D
input: (None, 28, 28, 1)
output: (None, 14, 14, 64)
leaky_re_lu_5: LeakyReLU
input: (None, 14, 14, 64)
output: (None, 14, 14, 64)
dropout_2: Dropout
input: (None, 14, 14, 64)
output: (None, 14, 14, 64)
conv2d_3: Conv2D
input: (None, 14, 14, 64)
output: (None, 7, 7, 128)
leaky_re_lu_6: LeakyReLU
input: (None, 7, 7, 128)
output: (None, 7, 7, 128)
dropout_3: Dropout
input: (None, 7, 7, 128)
output: (None, 7, 7, 128)
flatten_1: Flatten
input: (None, 7, 7, 128)
output: (None, 6272)
dense_2: Dense
input: (None, 6272)
output: (None, 1)

FIG 3 : Discriminator Model

3. Loss Function

Because our model is made up of two critical components, the losses from both models are quite important.

The discriminator loss will be estimated based on the exact and accurate categorization of actual and fraudulent photos. If the pictures created by the discriminator are correctly labelled from 0 to 9, the loss number will be minimal. If the label is not accurately detected, the discriminator model suffers a higher loss value.



REAL

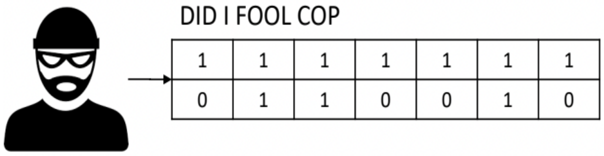
1	1	1	1	1	1	1
0	1	1	0	0	1	0

FAKE

0	0	0	0	0	0	0
0	1	0	0	0	0	0

Fig 4: Discriminator Model Loss

Generator, on the other hand, is the one who steals. The generator makes artificial pictures depending on its learning; it may be trained on random noise as well as the Gaussian technique. Generator loss is reduced if the generator can fool the discriminator, but it is increased if the generator can properly recognise the false inputs.



1	1	1	1	1	1	1
0	1	1	0	0	1	0

Fig 5: Generator Model Loss

4. Generator Model

GAN's generator model serves as a thief in the Thief-Cop Anomaly. The thief constantly attempts to imitate genuine money and manufactures duplicate copies in such a way that the Cop is fooled. Similarly, the generator tries to deceive the discriminator in order to improve its efficiency.

In the GAN model, efficiency comes at the cost of tricking the discriminator.
The relationship between the two, i.e. Generator and Discriminator, is more strongly reliant on Generator training. If the discriminator clearly distinguishes between authentic and counterfeit photos, Generator loss is quite close to the loss amount.
In order to obtain accuracy, we change the weights in order to train the network.

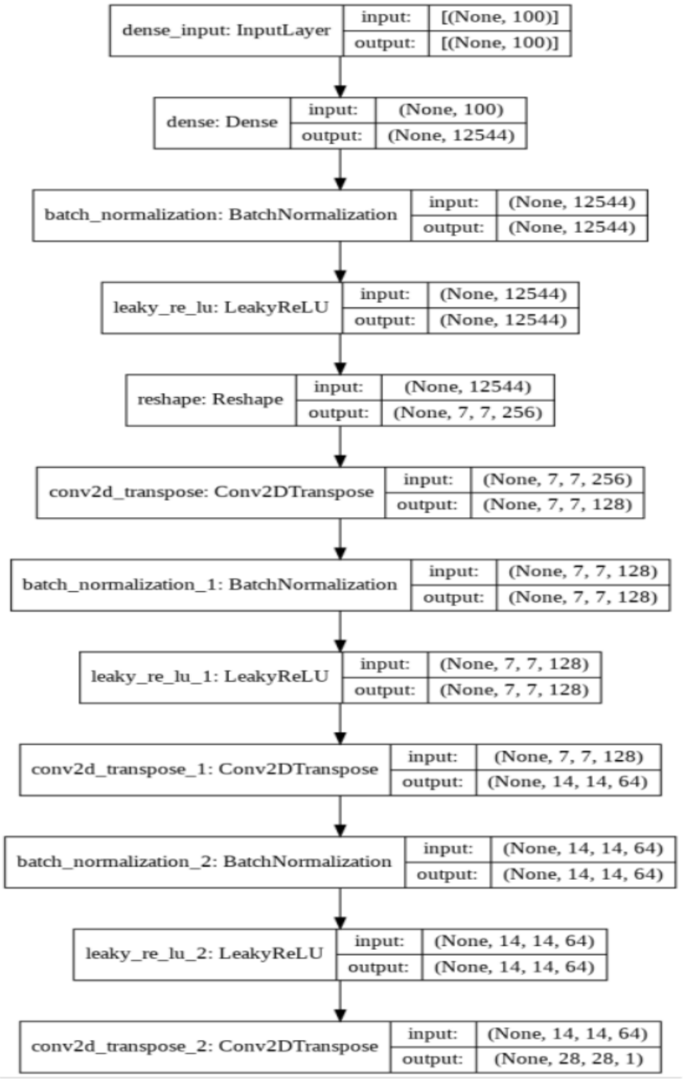

dense_input: InputLayer | input: | [(None, 100)] | output: | [(None, 100)]
dense: Dense | input: | (None, 100) | output: | (None, 12544)
batch_normalization: BatchNormalization | input: | (None, 12544) | output: | (None, 12544)
leaky_re_lu: LeakyReLU | input: | (None, 12544) | output: | (None, 12544)
reshape: Reshape | input: | (None, 12544) | output: | (None, 7, 7, 256)
conv2d_transpose: Conv2DTranspose | input: | (None, 7, 7, 256) | output: | (None, 7, 7, 128)
batch_normalization_1: BatchNormalization | input: | (None, 7, 7, 128) | output: | (None, 7, 7, 128)
leaky_re_lu_1: LeakyReLU | input: | (None, 7, 7, 128) | output: | (None, 7, 7, 128)
conv2d_transpose_1: Conv2DTranspose | input: | (None, 7, 7, 128) | output: | (None, 14, 14, 64)
batch_normalization_2: BatchNormalization | input: | (None, 14, 14, 64) | output: | (None, 14, 14, 64)
leaky_re_lu_2: LeakyReLU | input: | (None, 14, 14, 64) | output: | (None, 14, 14, 64)
conv2d_transpose_2: Conv2DTranspose | input: | (None, 14, 14, 64) | output: | (None, 28, 28, 1)

Fig 6: Generator Model

However, in GAN, the generator is not directly related to the loss you are trying to change. The generator network feeds the discriminator network, which provides the output you want to affect. This additional network segment should be included in the backpropagation. Backpropagation modifies the weights by estimating the effect of the weights on the output (how the output changes when the weights change). On the other hand, the effect of generator weights is determined by the effect of the discriminator weights that the generator feeds.

5. Training

The training step will be exclusively responsible for the desired outcome, which is to create new pictures and categorise them with a class label ranging from 0 to 9. This categorization will also help us assess the accuracy of our models. Discriminator and Generator losses show the performance of the model both alone and in combination. The images were scaled to the [-1,-1] range of the tanH activation function; no additional initialization was performed on the dataset. All models were trained using 256-batch mini-batch stochastic gradient descent (SGD). In all models of the LeakyReLU, the slope of the leak was adjusted at 0.2. To properly train our model, we employ the Adam optimizer with optimized hyperparameters and a learning rate of 0.0001.

SIMULATION EXPERIMENT

- Import the libraries - Tensorflow, matplotlib, numpy
- Load MNIST Fashion Dataset
- Reshape the Images, Normalize the Images
- Make a generator Model
    - Initialize a sequential Model
    - Add a dense layer
    - Add batch Normalization Layer
    - LeakyReLU Layer
    - Reshape layer
    - 2 Convolutional Layer followed by Batch Normalization & LeakyReLU Layer
    - Final Output layer
- Make a discriminator Model
    - Initialize a sequential model
    - Add 2 convolutional layer followed by LeakyReLU and Dropout layer
    - Flatten layer to finally convert the output to one single dimension
    - Output Layer
- Define the loss functions
    - Discriminator Loss
    - Generator Loss
- Defining Gradient Descent for both Discriminator and Generator, in order to minimize a cost function as far as possible. Gradient Descent is an optimization algorithm for finding a local minimum of a differentiable function.
- Training the model
- Testing the model

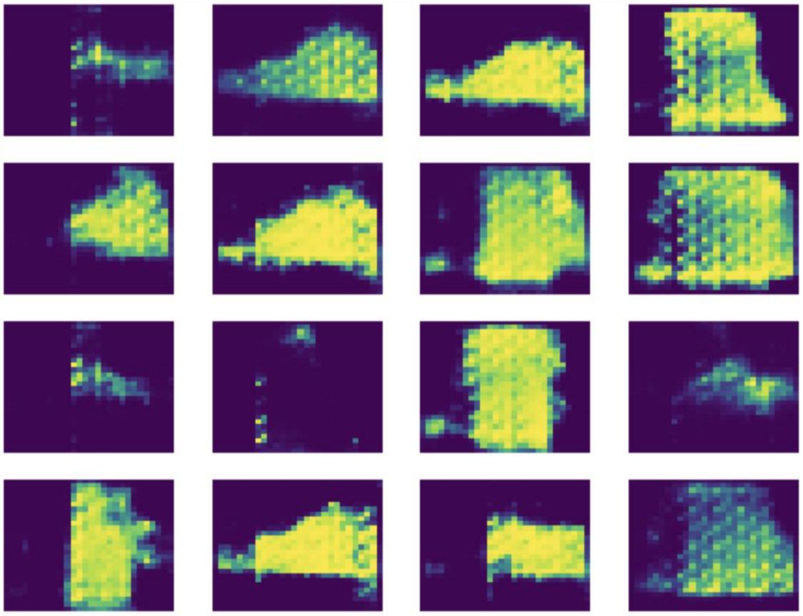
RESULTS

Using MNIST fashion clothing picture datasets, we demonstrate that our Deep Convolutional Adversarial pair learns a chain of instructions for expressing objects, components, and scenes. Both the Generator and the Discriminator have their own separate functioning. The Generator will produce a fresh new collection of designs for fashion firms. The new designs will be based on current datasets, and applying this knowledge generator will result in some highly imaginative new ideas. The discriminator will be in charge of feature learning and categorizing photos into appropriate class labels ranging from 0 to 9.

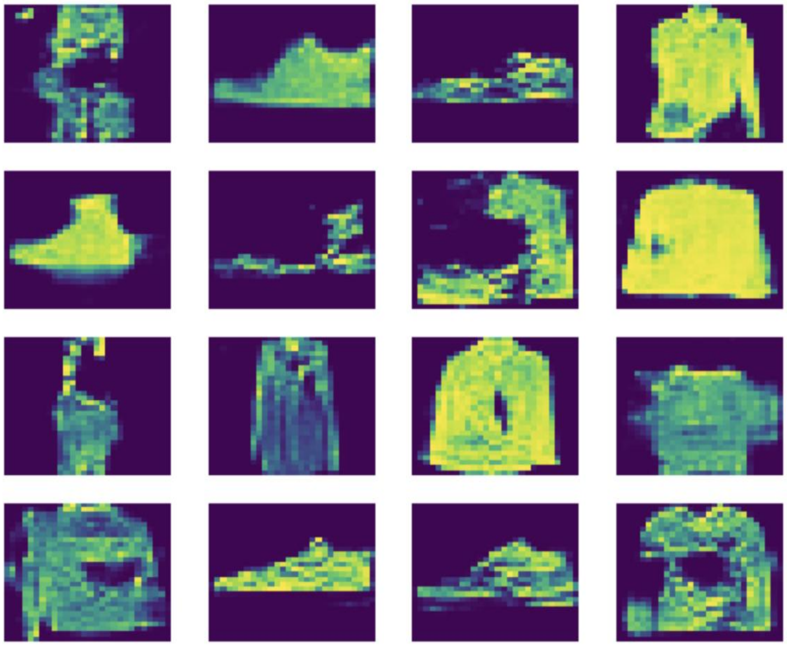
We noticed that when our model was being trained, it collapsed a subset of filters into a single oscillation mode. At epoch 25, the model is attempting to build a 3 X 3 shirt picture, however as it proceeds to period 56, it generates a Pant. By increasing the number of epochs and the variety of our dataset, we may obtain more accuracy and clarity in our designs.

The following are the key accomplishments:
1. Create fresh stylish garment designs.
2. Labeling of the new design in the appropriate class
3. Analyzing and fixing loss in the generator and discriminator

With future advancements in GAN performance, numerous sectors will be able to leverage the knowledge to create totally new art pieces, as well as certain and explicit categorization of diverse items utilizing DCGANs.



```
Time for epoch 25 is 27.00157380104065 sec
```



```
Time for epoch 173 is 27.06604552268982 sec
```

REFERENCES

[1]. Introduction: Guide to Generative Adversarial Networks
[2]. Mariam A. AliImage generation by using K-Means clustering technique, compusoft, 2019
[3]. Carl Doersch Tutorial on Variational Autoencoders, arxiv, 2016
[4]. Andrew Brock, Jeff Donahue, Karen Simonyan, Large Scale GANs, arvix, 2019
[5]. Tero Karras, Timo Aila, Samuli Laine, Jaakko Lehtinen Progressive Growing GANs, arvix, 2017
[6]. Simon Osindero Conditional GANs, arvix, 2014
[7]. Jun-Yan Zhu, Taesung Park, Phillip Isola, Alexei, A.Efros, StyleGANs, ervix, 2014
[8]. Tero Karras, Samuli Laine, Timo Aila, StyleGANs, arxiv, 2019