Minor Project

Conversational AI: Data Science (UCS663)

MNIST Data Generation with GANs

Submitted by:

Alwinder Singh

101917042

BE Third Year- CSE

Under the Mentorship of

Dr. Sahil Sharma

Assistant Professor

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

Computer Science and Engineering Department

Thapar Institute of Engineering and Technology, Patiala

May 2022

Abstract

In this project, I have worked on a Deep Learning technique known by the name Generative Adversarial Networks (GANs) which has gained immense popularity in the recent times owing to its high-quality results. The GANs are being used to generate handwritten digits by starting from scratch. The specific technique of Deep Convolutional Generative Adversarial Networks (DCGANs) has been used for the above-mentioned task. It uses a coupling of generator and discriminator to generate the desired output. The generator is a Deep Neural Network that generates the images of the handwritten digits. It repeatedly generates the images based on the inputs given. The discriminator is also a Deep Neural Network which is trained as a classifier between the MNIST handwritten dataset, and the images generated by the generator. The aim of this study is to provide a comprehensive information about GAN in the field of image synthesis. Finally, the results are saved in the form of a GIF which shows the progress made by the generator over time.

TABLE OF CONTENTS

Sr. No.	Title	Page No.
1	Introduction	1
2	Related Work	4
3	Methodology	6
4	Proposed Solution Architecture	8
5	Experiment and Result	10
6	Screenshots of Model Training	13
7	Conclusion and Future Scope	16
8	References	17

1. **Introduction**

1.1 Convolutional Neural Networks

Convolutional Neural Network (CNN) is a branch of artificial neural network that is primarily used in various computer vision tasks like image processing, image synthesis etc. Its basis is formed by the mathematical linear operation of convolution. It includes multiple layers, each performing a different operation namely, convolutional layer, non-linearity layer, pooling layer and fully connected layer. It uses backpropagation of these different building blocks to learn spatial hierarchies of features.

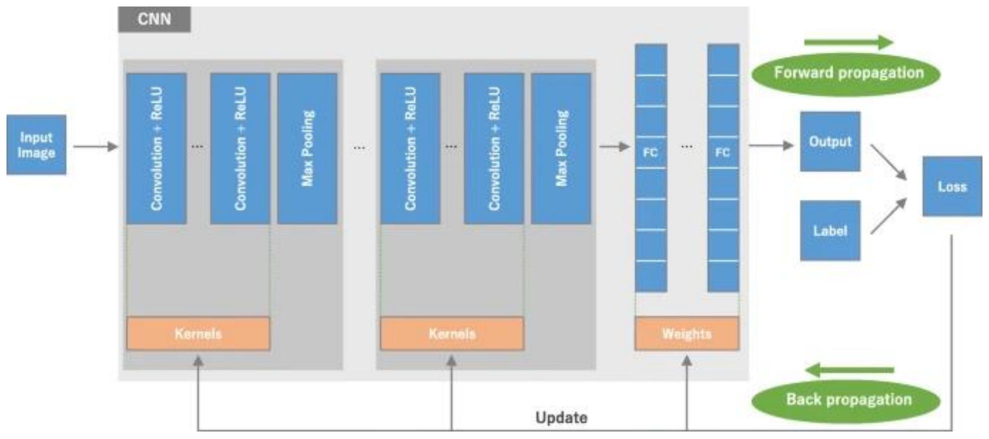

Fig.1 Overall Structure of a CNN

1.2 Generative Adversarial Networks

Generative Adversarial Networks, or GANs is a method to learn deep representations using generative learning. Generative learning, or more precisely Generative Modelling is an unsupervised machine learning task. It involves learning patterns in data to generate new examples that could have been plausibly drawn from the original dataset. GANs convert this task to a supervised learning task wherein the model comprises of two sub models, namely the generator, and the discriminator. Nowadays, GANs is a field that is increasingly attracting the attention of the Artificial Intelligence world as the representations learned by GANs have numerous applications, majorly in the image processing field. Some of the applications include image synthesis, semantic image editing, image super resolution, classification and image reconstruction.

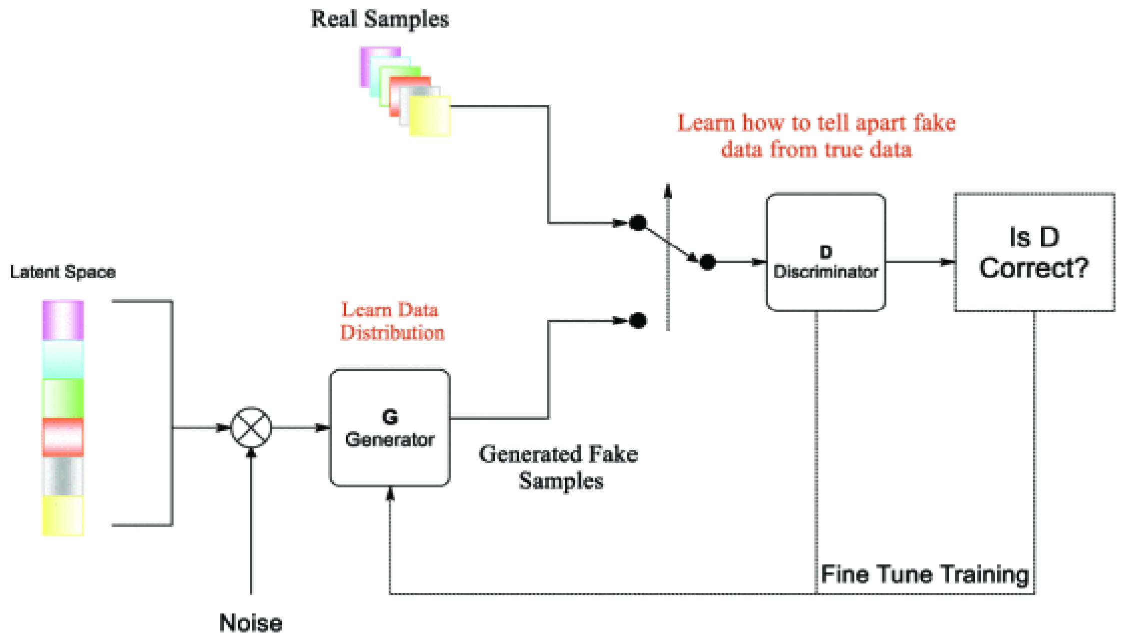

Fig.2 General Structure of a GAN

1.2.1 Generator

Generator of a GAN is a model that starts from random noise and slowly with learning generates the required output. It learns from the results of its counterpart i.e. the discriminator in order to mimic the actual dataset. Its ultimate objective is to make the discriminator classify its output as real.

1.2.2 Discriminator

Discriminator of a GAN is basically a model that performs the task of classification. It is a supervised model that takes the results of the generator and classifies them as real/fake on the basis of the dataset provided to it. Its ultimate goal is to detect the fake images generated by the generator. The GAN is said to be trained if the discriminator fails to label its input as fake for at least half of the images generated by the generator, i.e. it is in an equilibrium.

Following are the details of the model used in the project:

- GANs come in different variants. The model being used here is Deep Convolutional Generative Adversarial Networks (DCGAN).
- Both the Generator and Discriminator are deep Convolutional Neural Networks that harbor the principle of backpropagation to learn patterns in the input dataset and hence generate new images.
- While training, the Generator and Discriminator use the adam optimizer in order to modify the attributes of the neural network so that the overall loss is reduced.
- The loss function used in the Generator is the binary cross entropy between the fake output and the real images similar to the fake outputs.
- The loss function used in the Discriminator is the binary cross entropy between the fake output and the real images similar to the fake outputs and the binary cross entropy between the real output and the real images similar to real outputs.

2. Related Work

2.1 Learning Patterns from Dataset

Unsupervised Learning is a technique of Machine Learning which doesn’t require labelling of the training dataset. It makes use of some techniques/ algorithms in order to learn similarities and patterns in the dataset so as to perform specified tasks. One of the biggest example of unsupervised tasks is Clustering. It involves formation of clusters/ aggregates in the dataset based upon their similarity according to some fixed similarity measures. This holds a great importance in the field of image processing where images can be clustered into disjoint groups. Moreover, image patches can be hierarchically clustered to learn powerful image representation.

Another example is Auto Encoders. They are a special type of unsupervised deep learning algorithm which try to provide an output which is extremely similar to the given input. It consists of two parts namely, encoder and decoder. The task of the encoder is to convert the input image into a lower dimensional representation by remembering only the important part of the image. The task of the decoder is to recreate the image from this lower dimensional representation.

2.2 Generative Modelling

Generative modelling is the study of input dataset in order to determine how it is generated. It basically learns the patterns in the dataset in order to determine its origins as its major task is to generate an output that is similar to the dataset. It is a probabilistic modelling technique rather that a deterministic modelling technique as it cannot depend on fixed calculations due to the stochastic nature of its output. Its goal is to build a model that is able to generate new data that is similar to the original data which basically means it follows the trends of the original dataset.

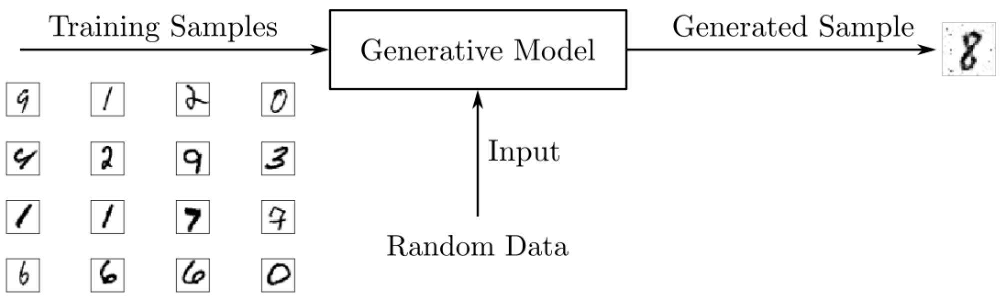

Fig.3 Block Diagram of Generative Modelling

2.3 Types of GANs

- Conditional GAN and Unconditional GAN(CGAN)
  It includes additional parameters in the discriminator so as to ensure that it is not fooled by the generator easily and improve the performance of GANs.
- Information Maximizing Generative Adversarial Network (InfoGAN)
  Its goal is increase some particular semantic meaning to the variable in the generator. This results in better performance of the GAN.
- Auxiliary Classifier Generative Adversarial Network (AC-GAN)
  It makes the generator class conditional and trains an additional model to reconstruct class label in the discriminator.
- Cycle Generative Adversarial Network
  It is primarily used to learn transformation between images of different styles. FaceApp is considered one of the best example of CycleGAN that transforms human faces into different ages.
- Style Generative Adversarial Network
  It is used to create images of super resolution. Its main task is to generate images of very high resolution starting form low resolution inputs.

3. Methodology

The following flowchart explains the overall methodology followed in this project:

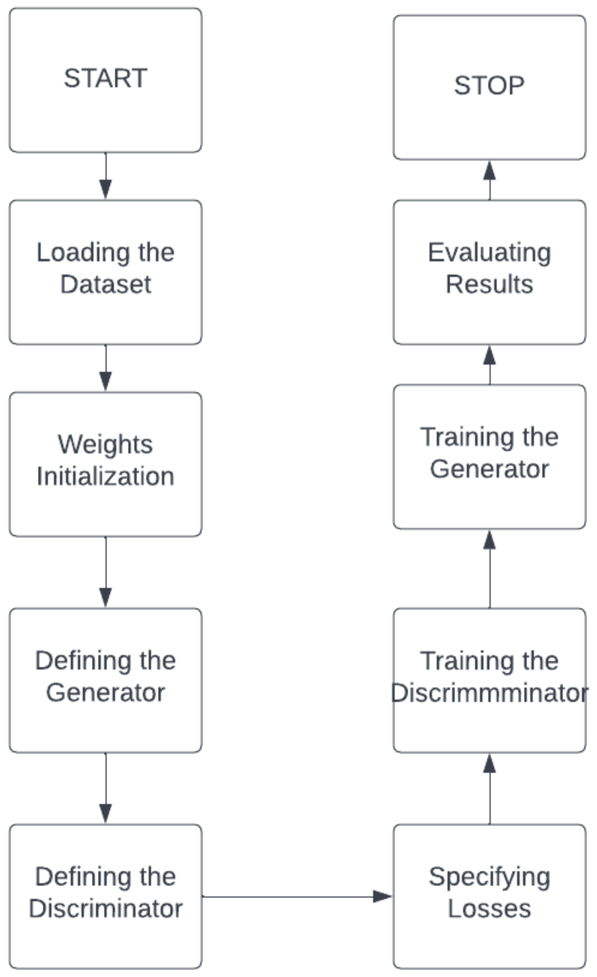

Fig.4 Flowchart of the project

- Loading the Dataset

  The dataset being used here is MNIST dataset for handwritten digits. The dataset is being imported from pytorch. The dataset is loaded in such a way that it is shuffled and grouped into batches of size 256.
- Weights Initialization

  After extensive research it has already been formulated by prevuous researchers that all model weights must be randomly initialized from normal distribution with mean 0 and standard deviation 0.2. And the same is done for the all convolutional, convolutional transpose and batch normalization layers.

- Defining the Generator

  The Generator is defined as a neural network consisting of series of 2- dimensional convolutional transpose layers and each paired with 2- dimensional batch normalization layer and ReLU activation layer.

- Defining the Discriminator

  The Discriminator is a neural network that performs the binary classification task of identifying the real and fake images. It uses a neural network consisting of series of 2-dimension convolutional layers, batch normalization layers and LeakyReLU layers with the final layer being a Sigmoid activation layer to output probability of each case.

- Specifying Losses

  The loss functions being used are primarily binary cross entropy functions of different type. The loss is basically defined as $\log(D(x))+\log(1-D(G(z)))$, where $D(x)$ is average output of discriminator for all real batches and $D(G(z))$ is average discriminator batch for all fake batches.

- Training the Discriminator

  The discriminator is trained to maximize the probability of correctly classifying the input as real or fake, basically we want to maximize the value of $\log(D(x))+ \log(1-D(G(z)))$.

- Training the Generator

  The Generator is trained to generate images very close to the real images that is minimize the value of $\log(1-D(G(z))$.

- Evaluating Results

  The results are evaluated by plotting the various losses calculated and also saving the training process as an animation to visualize the journey of the training process.

4. Proposed solution architecture

4.1 Generator Model

The Generator model takes the input of size 1*28*28. It consists of 11 layers in total which include three sets of convolutional transpose 2D layer, batch normalization 2D layer and a ReLU layer with a final convolutional transpose 2D layer and a dense tanh layer. The convolutional transpose layers provide the feature of strided convolution. The batch normalization layer help with flow of gradient during training.

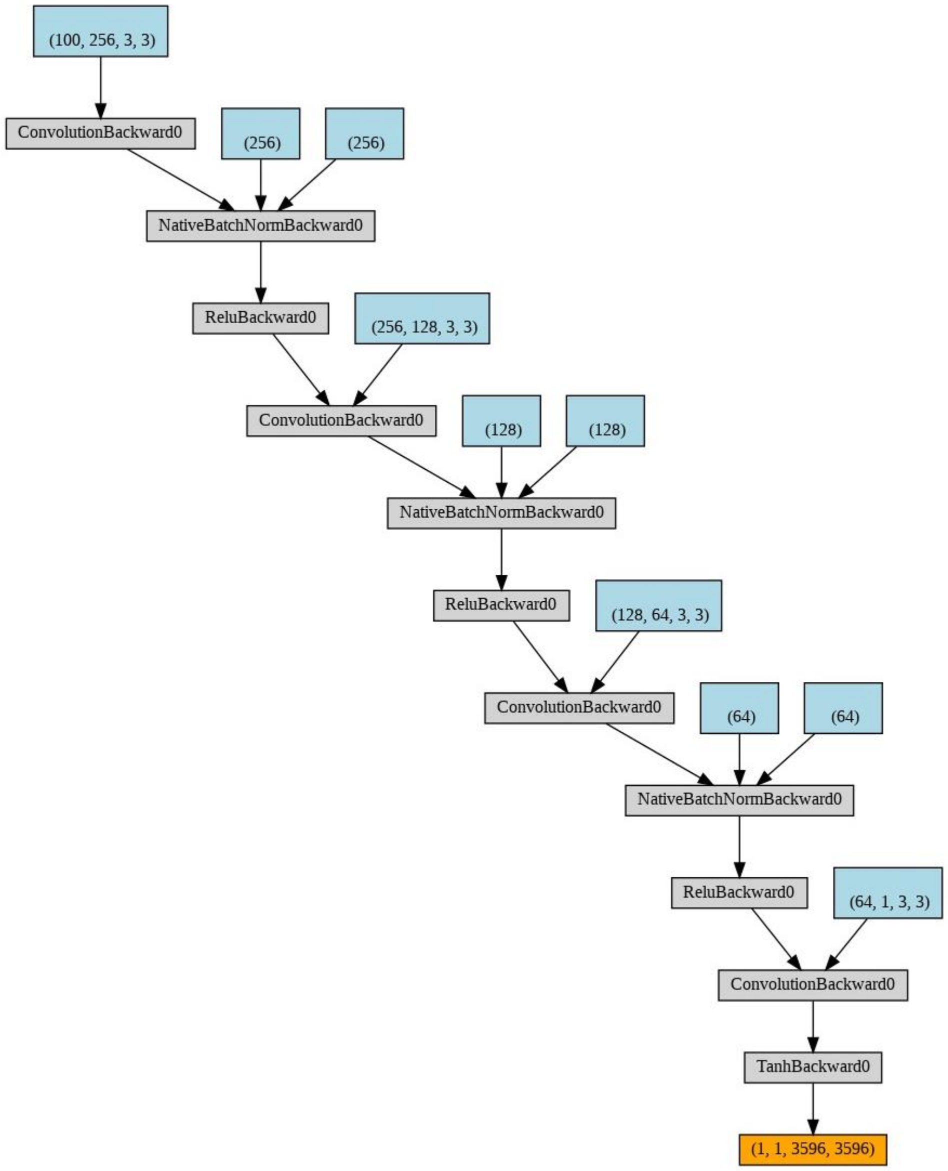

Fig.5 Architecture of Generator

4.2 Discriminator Model

The Discriminator model takes the input of size 1*28*28. It consists of 9 layers in total which includes a set of convolutional 2D layer, batch normalization 2D layer, two sets of convolutional 2D layer, batch normalization 2D layer, and a ReLU layer with a final convolutional 2D layer. The convolutional 2D layer provide for strided convolution which is better than pooling to down sample as the network learns its own pooling function. Batch Normalization and LeakyReLU allow healthy gradient flow.

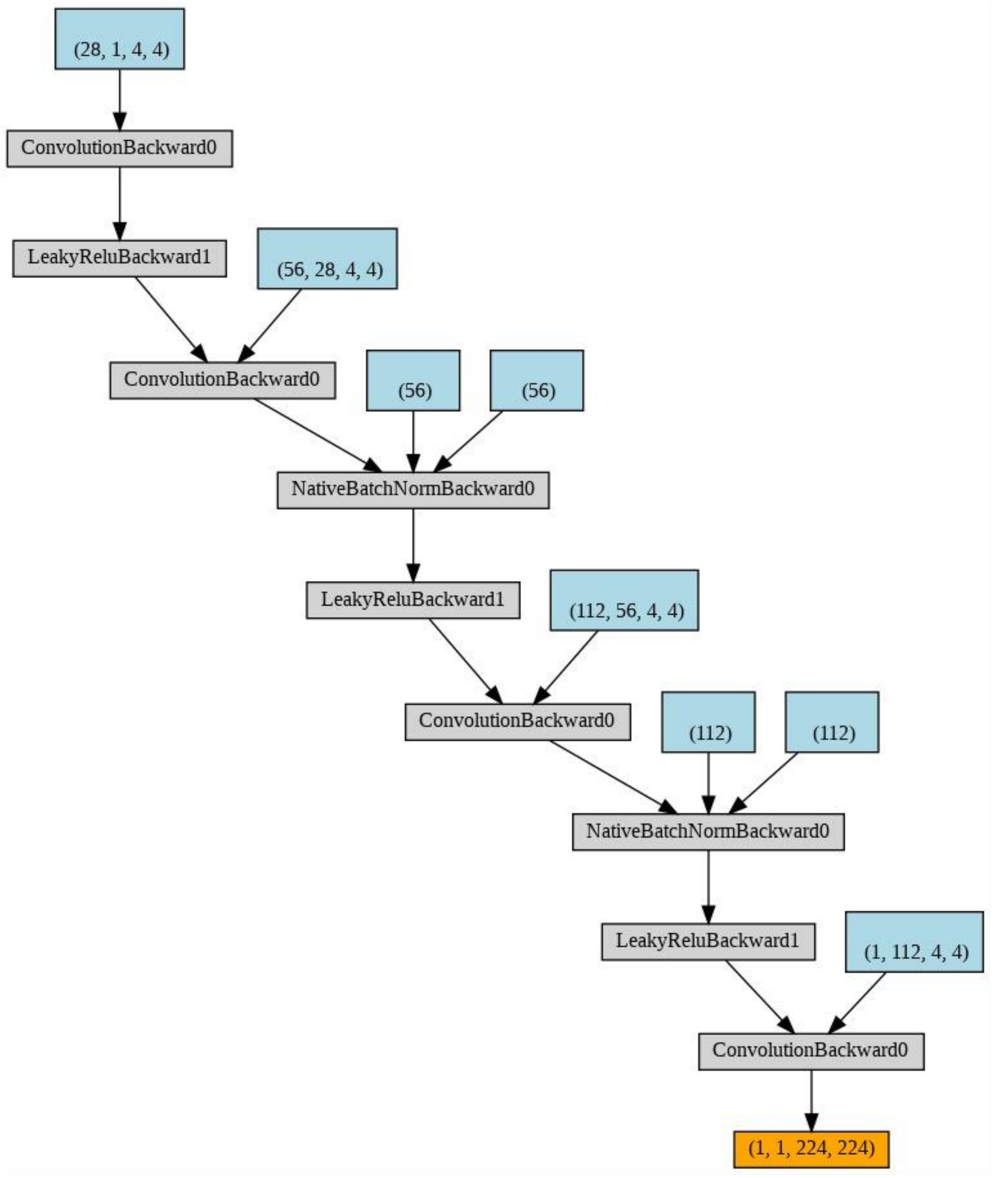

Fig. 6 Architecture of Discriminator

5. Experimentation and Results

5.1 Dataset Details

The MNIST database (Modified National Institute of Standards and Technology database) is a large database of handwritten digits that is commonly used for training various image processing systems. The database is also widely used for training and testing in the field of machine learning. It was created by "re-mixing" the samples from NIST's original datasets.

The MNIST database contains 60,000 training images and 10,000 testing images. Half of the training set and half of the test set were taken from NIST's training dataset, while the other half of the training set and the other half of the test set were taken from NIST's testing dataset.

5.2 Hyperparameter Tuning

Although the Hyperparameter Tuning for various parameters of the neural network is being done using the principle of backward propagation, the initialization of weights being done on the basis of research by many previous scientists who concluded that the initial parameters must be from a normal distribution with mean 0 and standard deviation 0.2. Hence, the parameters are randomly initialized from this distribution.

5.3 Modelling

The loss function which drives the whole GAN is Binary Cross Entropy with logistic loss. It combines the Binary Cross Entropy of each class with a Sigmoid layer. This allows us to use the log-sum-exp trick which ensures greater numerical stability than simply placing a Sigmoid layer followed by Binary Cross Entropy Loss.

For the discriminator, we define our real label as 1 and fake as 0. We use two separate optimizers, one for the generator and the other for the discriminator. Both are Adam optimizers with learning rate 0.0002 and Beta1 = 0.5.

The model is trained twice, once with saturation and other without saturation. This results in two types of losses namely, Saturating loss and Non-saturating loss.

Saturating loss: $\log(1-D(G(z)))$, derivation: $-1/(1-D(G(z)))$

Non-saturating loss: −log(D(G(z))) , derivation: −1/D(G(z)

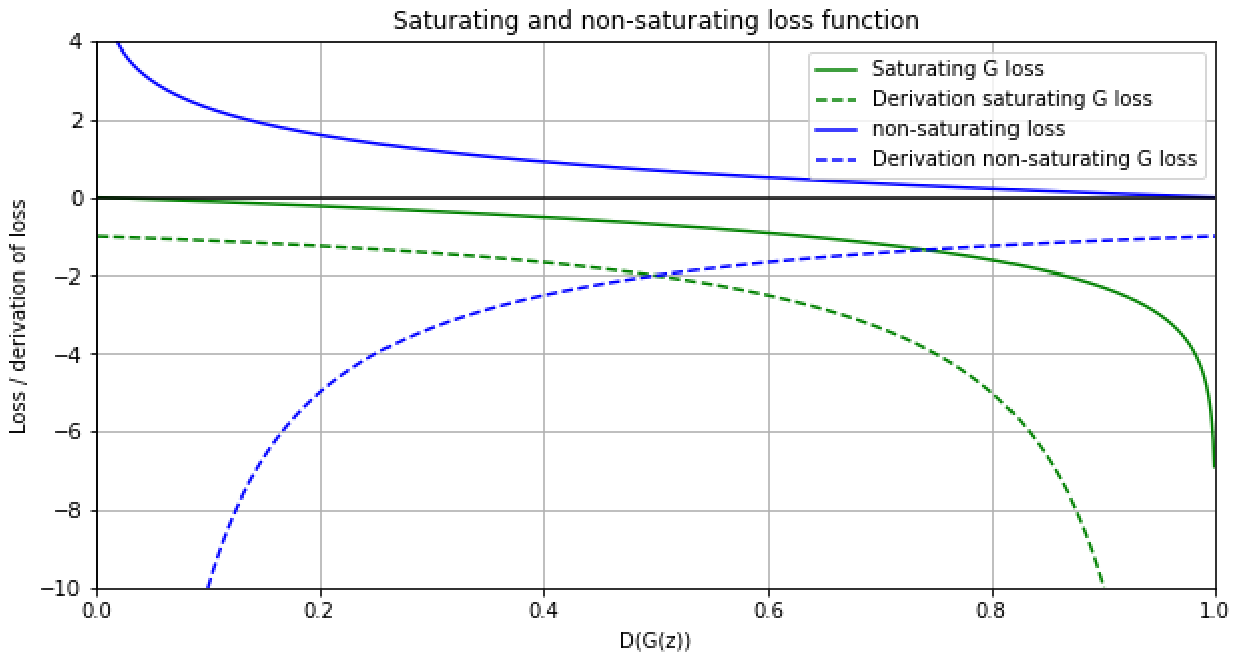

Fig.7 Plot Showing the curves of Saturation loss and Non-Saturation loss

5.4 Evaluation Results

Following are the results obtained while training the generator and discriminator in both saturation and non-saturation mode:

5.4.1 Loss VS Training Iteration

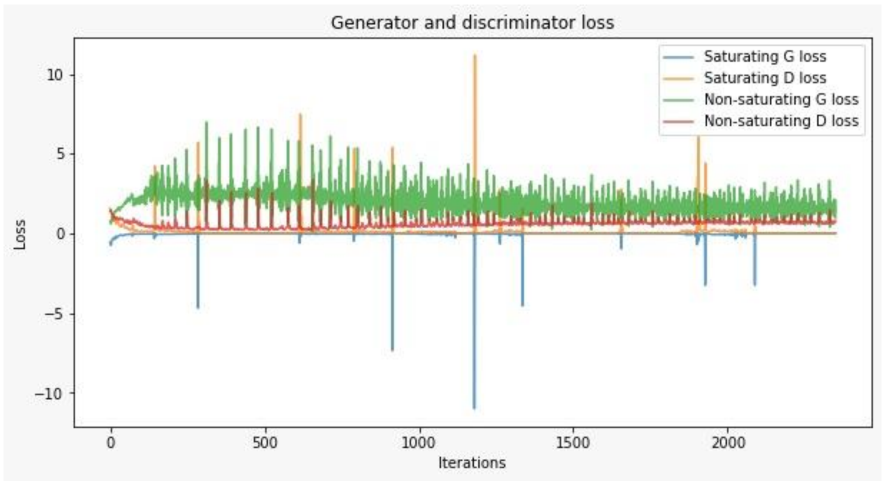

Fig.8 Loss VS Training Iteration

5.4.2 Generator Gradients Mean VS Training Iteration

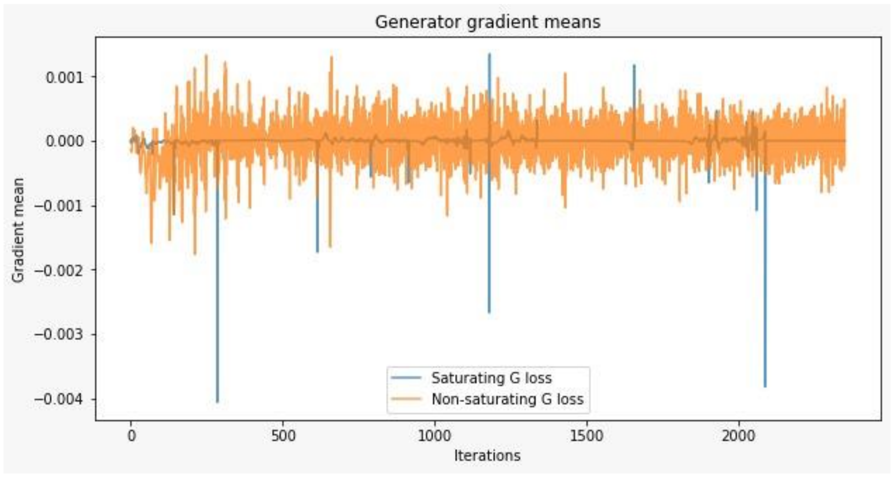

Fig. 9 Generator Gradients Mean VS Training Iteration

5.4.3 Generator Gradients Standard Deviation VS Training Iteration

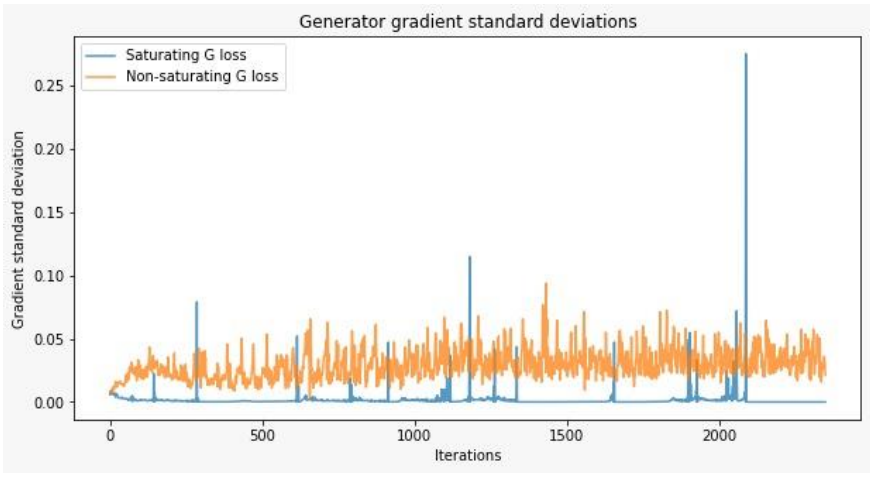

Fig.10 Generator Gradients Mean VS Training Iteration

6. Screenshots of Model Training

6.1 Model Training with Saturation

6.1.1 Initial Output

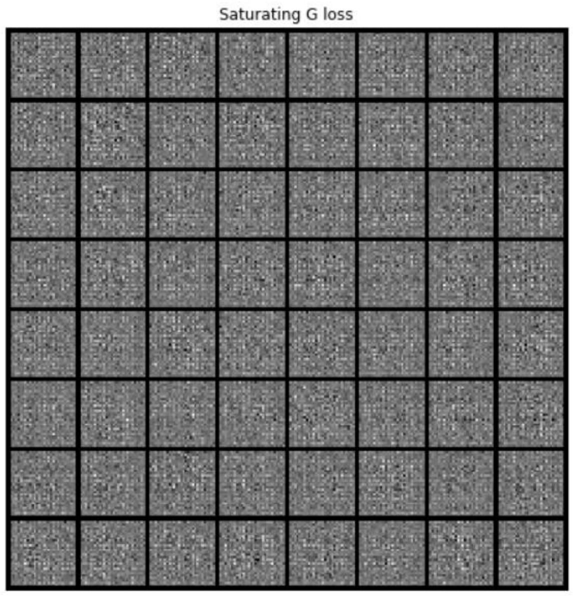

Fig.11 Initial Output

6.1.2 Output after 5 epochs

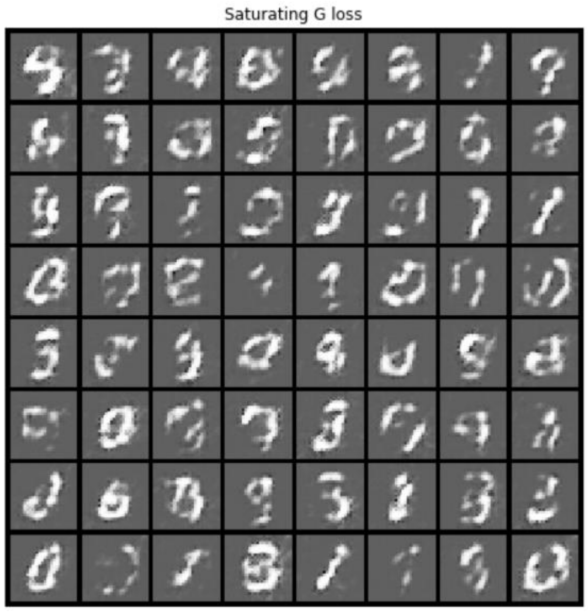

Fig.12 Output after 5 epochs

6.1.3 Output after 10 epochs

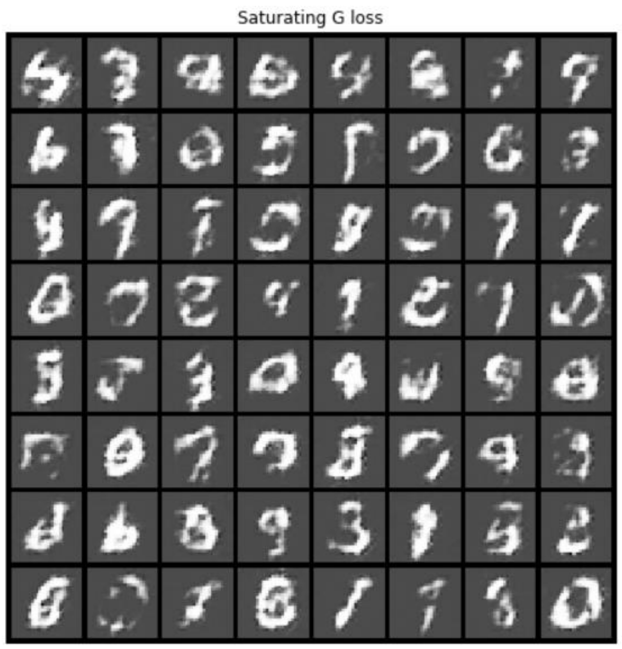

Fig. 13 Output after 10 epochs

6.2 Model Training without Saturation

6.2.1 Initial Output

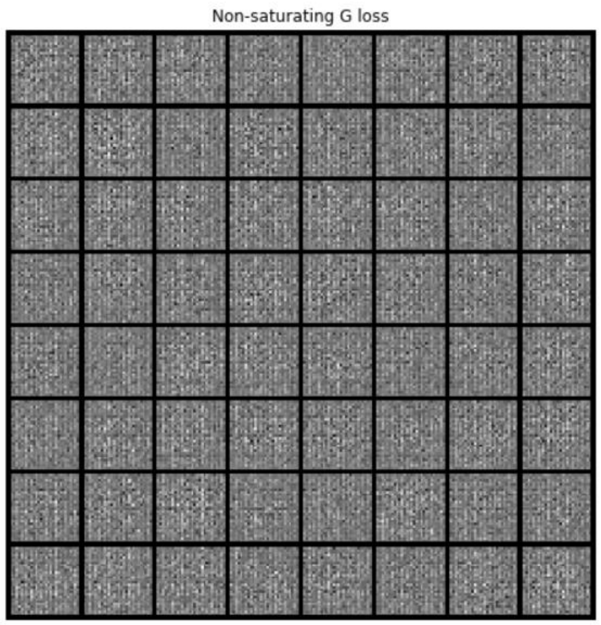

Fig.14 Initial Output

6.2.2 Output after 5 epochs

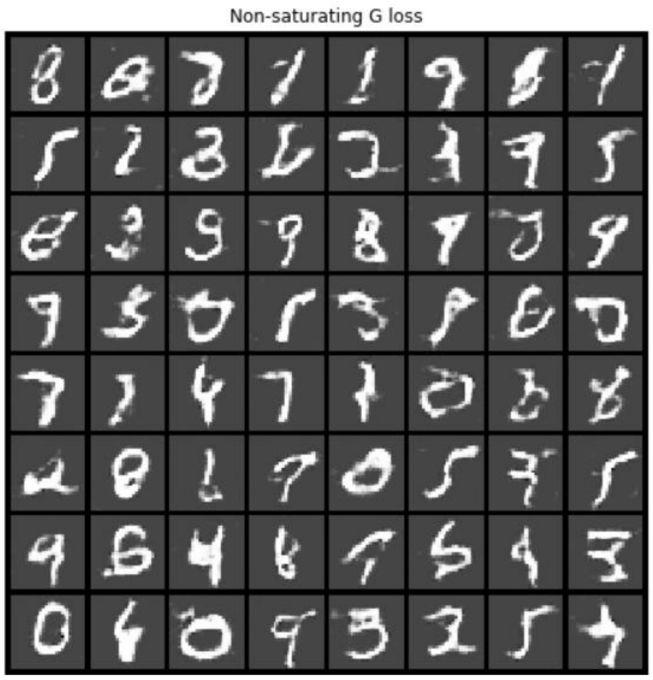

Fig.15 Output after 5 epochs

6.2.3 Output after 10 epochs

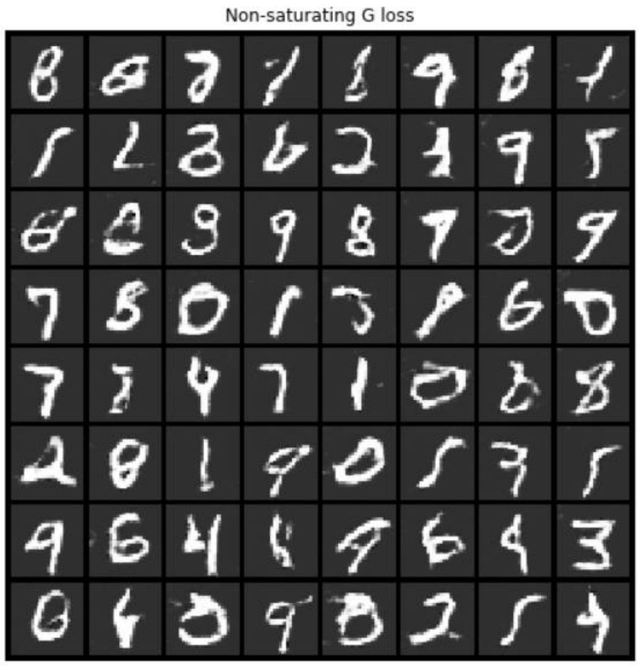

Fig. 16 Output after 10 epochs

7. Conclusion and Future Scope

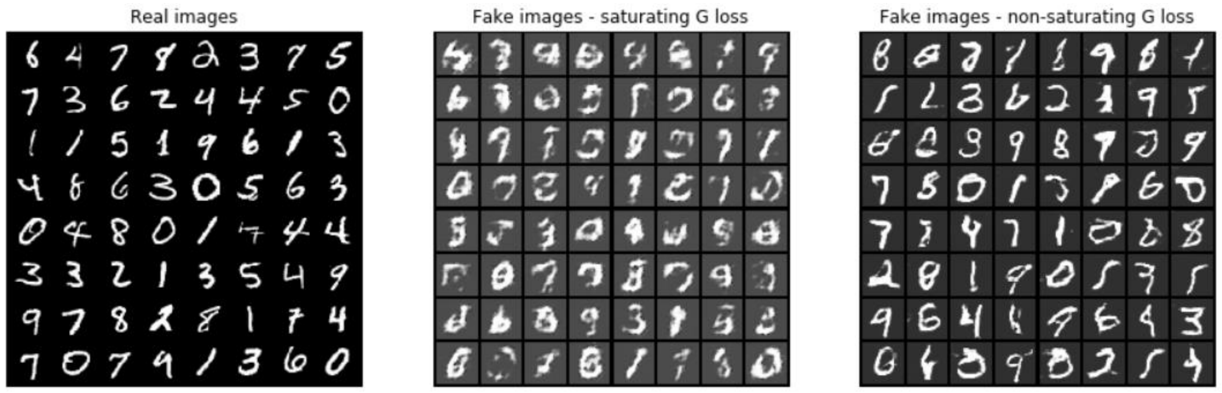
Real images
Fake images - saturating G loss
Fake images - non-saturating G loss

Fig. 17 Comparison of Real Image with Fake Image

The model has produced good results as shown in the image above. The fake images being produced by the non-saturating model are almost readable and identical to the real image whereas the images produced with saturation do not perform upto the mark. Still there is a scope for improvement as many of the digits are still illegible and some of them are even wrong. This can be improved by using better GANs like InfoGAN (which increase some particular semantic meaning to the variable in the generator), CGAN (which includes additional parameters in the discriminator so as to ensure that it is not fooled by the generator easily) or AC-GAN (which makes the generator class conditional and trains an additional model to reconstruct class label in the discriminator).

References

- Yamashita, R., Nishio, M., Do, R.K.G. et al. Convolutional neural networks: an overview and application in radiology. Insights Imaging 9, 611–629 (2018).
- P. Shende, M. Pawar and S. Kakde, "A Brief Review on: MRI Images Reconstruction using GAN," 2019 International Conference on Communication and Signal Processing (ICCSP), 2019, pp. 0139-0142, doi: 10.1109/ICCSP.2019.8698083.
- Dewi, C., Chen, RC., Liu, YT. et al. Synthetic Data generation using DCGAN for improved traffic sign recognition. Neural Comput & Applic (2021)
- Wikipedia contributors. (2022, May 13). Generative adversarial network. In Wikipedia, The Free Encyclopedia. Retrieved 17:22, May 18, 2022
- Wikipedia contributors. (2022, April 17). MNIST database. In Wikipedia, The Free Encyclopedia
- K. Wang, C. Gou, Y. Duan, Y. Lin, X. Zheng and F. -Y. Wang, "Generative adversarial networks: introduction and outlook," in IEEE/CAA Journal of Automatica Sinica, vol. 4, no. 4, pp. 588-598, 2017, doi: 10.1109/JAS.2017.7510583.
- L. Deng, "The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]," in IEEE Signal Processing Magazine, vol. 29, no. 6, pp. 141-142, Nov. 2012, doi: 10.1109/MSP.2012.2211477.