Generative Adversarial Networks (GAN)
A Group Project for CS419, IITB

Piyush Raj, Mahavir Gandhi, Vansh Maheshwari, Hari Krishna Sahoo , Ashwin S

Abstract

Generative Adversarial Networks (GANs) have emerged as a powerful framework for generating realistic data through a competitive process between two neural networks. This report presents an overview of GANs, their architecture, and the specific configurations and results obtained from a GAN-based model trained for image generation using the Fashion MNIST dataset. The project was conducted as part of the course CS419 at IIT Bombay.

1 Introduction

Generative Adversarial Networks (GANs) were introduced by Ian Goodfellow et al. in 2014 as a novel approach for generating new data that resembles a given dataset. A GAN consists of two components: a Generator (**G**) and a Discriminator (**D**). The Generator creates fake data from random noise, while the Discriminator attempts to distinguish between real data and the fake data produced by the Generator. The two networks are trained simultaneously in a process referred to as adversarial training, where the Generator learns to fool the Discriminator, and the Discriminator learns to better detect fake data.

In this project, we used the "Fashion MNIST" dataset, which contains grayscale images of 28x28 pixel clothing items, as our target data for training the GAN. This report describes our approach and results for generating similar grayscale images from random noise.

2 GAN Architecture and Training

Our GAN implementation follows the typical architecture with a Generator and a Discriminator, both utilizing deep learning architectures to perform their respective tasks.

2.1 Generator

The Generator network is designed to take random noise as input and generate a grayscale image as output. The noise vector is reshaped into a 7x7x128 tensor and passed through two upsampling layers followed by two convolutional blocks. The first two convolution layers refine the features, and the final convolution block produces the output image. The Generator's goal is to create realistic images that are indistinguishable from the real dataset.

2.2 Discriminator

The Discriminator is tasked with classifying images as real or fake. It contains four convolution layers followed by a final dense layer to output a binary classification result. The Discriminator quickly converges as it is trained to distinguish between real and fake images. We employ a binary cross-entropy loss function and the Adam optimizer for both the Generator and Discriminator.

2.3 Training Parameters

The training of the GAN follows specific configurations:

- Both the Generator and Discriminator use the Adam optimizer.
- The learning rate for the Generator is set to 10^{-4}, while for the Discriminator it is 10^{-5}. This

setup ensures that the Discriminator converges quickly, while the Generator has more flexibility to improve its output.

- We ran the training for a total of 20 epochs.
- The loss function used is binary cross-entropy.

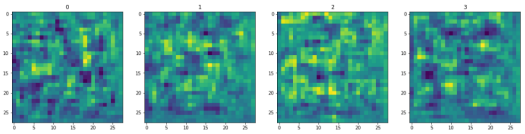

Figure 1: Gaussian noise generated at the start of training, which serves as input to the Generator.

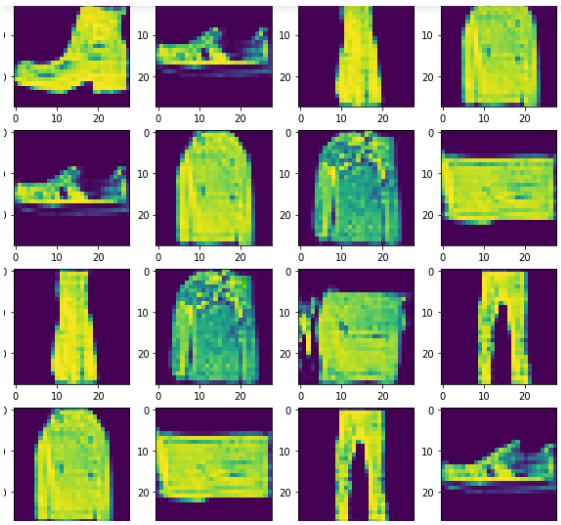

Figure 2: Examples of generated grayscale images after 20 epochs of training. These images resemble items from the Fashion MNIST dataset.

3 Results and Evaluation

The GAN model was trained on the Fashion MNIST dataset for 20 epochs. Throughout the training, we observed the evolution of the generated images and the performance of both the Generator and Discriminator. The following sections highlight the key results.

3.1 Initial Output: Gaussian Noise

At the start of the training process, the Generator's output is nothing more than random noise. This can be seen in Figure 1, where the initial images are completely unstructured and bear no resemblance to the target fashion images.

3.2 Final Output: Generated Images

After several epochs of adversarial training, the Generator improves its ability to generate images that closely resemble the Fashion MNIST dataset. Figure 2 presents some of the generated images after the model has trained for 20 epochs.

3.3 Training Progress: Loss Graph

During training, both the Generator and Discriminator undergo error minimization. As the Discriminator quickly learns to classify images, its loss rate drops faster compared to the Generator. Figure 3 shows the loss curves for both the Discriminator and Generator over the 20 epochs. This graph highlights the adversarial nature of the training process, where both networks improve as they "compete" against each other.

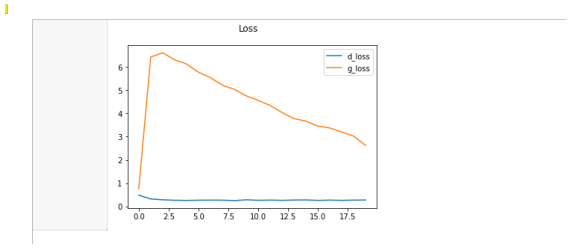

Figure 3: Loss graph of the Discriminator and Generator during training. The Discriminator converges faster than the Generator, which requires more epochs to generate realistic images.

4 Conclusion

This project presented an implementation of a Generative Adversarial Network (GAN) using the Fashion MNIST dataset. The architecture of the model involved a Generator that used noise vectors to cre-

ate images and a Discriminator to classify real versus fake images. The results show that, after 20 epochs of training, the Generator successfully created realistic images resembling those in the Fashion MNIST dataset. This work was conducted as part of the group project for the CS419 course at IIT Bombay.

References

Generative Adversarial Nets
Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio
Département d'informatique et de recherche opérationnelle
Université de Montréal
Montréal, QC H3C 3J7