COMP 5212 HA6 - GAN

Leung Pak Hei 20690382

April 2023

1 Introduction

In this computer science report, we investigate the performance of a Generative Adversarial Network (GAN) model trained on the Fashion-MNIST dataset. Our goal is to explore the impact of various factors on the performance of the GAN model by conducting a series of experiments on a baseline model.
We begin by introducing the baseline GAN model and its architecture, which is used as the starting point for all our experiments. We then describe the Fashion-MNIST dataset, which consists of 60,000 training images of 10 different fashion categories. We also provide an overview of GANs and their training process, including the roles of the generator and discriminator models and the objective function used to train them.
Our experiments focus on three main factors that can influence the performance of the GAN model: the dimensions of the latent variable z, the architecture of the model, and the number of steps (k) used to train the discriminator. For each factor, we explore several variations and evaluate their impact on the quality of the generated images. Our ultimate goal is to identify the optimal configurations for each factor and determine how they can be combined to achieve the best performance for the GAN model on the Fashion-MNIST dataset.
For easy implementation, the risk metrics we use in this task will be primarily based on the testing loss. The training epoch is 5 and the batch size is 100 for quick evaluation.

2 Comparison using different dimensions of the latent variable z:

The latent variable in a GAN represents the input to the generator network and is typically a vector of random noise. The dimensions of the latent variable determine the size and complexity of the input space for the generator network, and can have a significant impact on the performance of the GAN model.
This section will explore the performance of the prediction using different dimensions of the latent variable z. The result is as follows

Comparison using different z	
latent variable	Evaluation
4	d loss: 0.2327, g loss: 3.2408 D(x): 0.91, D(G(z)): 0.08
8	d loss: 0.5392, g loss: 3.2785 D(x): 0.84, D(G(z)): 0.12
16	d loss: 0.2601, g loss: 3.9402, D(x): 0.82, D(G(z)): 0.14
32	d loss 0.2201, g loss: 2.9402, D(x): 0.85, D(G(z)): 0.15

From the results, it is clear that increasing the latent variable will decrease the testing loss . The reason for this may be due to the fact that increasing the dimensions of the latent variable can potentially increase the diversity and complexity of the generated images, as the generator network has more flexibility to generate different variations of the same image. However, if the dimensionality is too high, it can also lead to overfitting and generate unrealistic or noisy images. On the other hand, decreasing the dimensions of the latent variable can result in a more limited input space for the generator network, which may lead to less diverse and repetitive generated images. Also, the training seems to be of little fluctuation, so the evaluation is difficult.

3 Comparison using different architecture

The architecture of a GAN model refers to the specific design and configuration of its generator and discriminator networks. Varying the architecture of a GAN model can have a significant impact on its performance and the quality of the generated images.
Changing the architecture of the generator network can affect the complexity and capacity of the model to generate diverse and high-quality images. A more complex generator network with more layers or hidden units can potentially generate more detailed and realistic images. However, a very complex generator network may also lead to overfitting or instability during training.
Similarly, varying the architecture of the discriminator network can also impact the performance of the GAN model. A more complex discriminator network can potentially better distinguish between real and fake images, resulting in higher-quality generated images. However, a very complex discriminator network may also lead to overfitting or instability during training.
In addition, changing the architecture of both the generator and discriminator networks can impact the balance of power between the two networks, which can also affect the performance of the GAN model.
The result is as follows

Comparison using different architecture in the discriminator	
discriminator's architecture	Evaluation
Linear(imagesize, hidden size) + Leaky ReLU(0.2)	d loss: 0.7005, g loss: 1.3945, D(x): 0.73, D(G(z)): 0.29
(Linear(imagesize, hiddensize) + Leaky ReLU(0.2))*2	d loss: 0.2327, g loss: 3.2408, D(x): 0.91
(Linear(imagesize, hiddensize) + Leaky ReLU(0.2))*4:	d loss: 0.0002, g loss: 9.1831, D(x): 1.00, D(G(z)): 0.00
(Linear(imagesize, hiddensize) + Leaky ReLU(0.2))*6	d loss: 0.0002, g loss: 9.1831, D(x): 1.00, D(G(z)): 0.00

From the above experiments, increasing the depth of the architecture refers to adding more layers to the generator or discriminator networks. This can increase the capacity of the model to capture more complex and intricate patterns in the input data, which can result in higher-quality and more diverse generated images. However, increasing the depth too much can also lead to overfitting or instability during training, especially if the model does not have enough training data to learn from. For example, (Linear(imagesize, hiddensize) + LeakyReLU(0.2))*4 and (Linear(imagesize, hiddensize) + LeakyReLU(0.2))*6, the model seems cannot learn .

4 Comparison using a different number of steps (k) to train the discriminator

The number of steps (k) to train the discriminator in a GAN model refers to the number of times the discriminator is trained for each iteration of training the generator. During each iteration of training, the generator produces a batch of fake images, which are fed into the discriminator along with a batch of real images. The discriminator then tries to distinguish between the real and fake images and produces a score for each image.
After the discriminator has been trained for k steps, the generator is trained for one step to produce a new batch of fake images. The objective of the generator is to produce fake images that can fool the discriminator into believing that they are real. This process of alternating training between the generator and discriminator continues for multiple epochs until the generator produces realistic and high-quality images.
The number of steps (k) to train the discriminator is an important hyperparameter that can affect the stability and performance of the GAN model. A larger value of k can potentially improve the quality of the discriminator by providing it with more opportunities to learn from both real and fake images. However, this can also slow down the training process and may result in the generator failing to learn from the feedback provided by the discriminator.
On the other hand, a smaller value of k can speed up the training process but may result in a less accurate discriminator that fails to provide effective feedback to the generator.
Finding the optimal value of k requires experimentation and tuning of the hyperparameters of the GAN model, including the learning rate, batch size, and number of epochs.
The results are as the following:

Comparison using different number of steps (k) to train the discriminator	
number of steps (k)	testing loss
default	d loss: 0.2327, g loss: 3.2408, D(x): 0.91 D(G(z)): 0.08
default*2	d loss: 0.2121, g loss: 3.1401, D(x): 0.87 D(G(z)): 0.13
default*3	d loss: 0.1802, g loss: 2.906, D(x): 0.85 D(G(z)): 0.15
default*4	d loss: 0.2327, g loss: 3.324, D(x): 0.89 D(G(z)): 0.11

From the above experiments, given the same archetecture, the value of k, i.e., the number of steps to train the discriminator before updating the generator, has a significant impact on the training performance of GANs. k value tends to have effect in learning rate and the performance A small value of k result in the generator not being able to learn much from the current state of the discriminator, leading to instability and mode collapse. On the other hand, a large value of k may slow down the training process and make it difficult for the generator to catch up with the discriminator.

5 Comparison of VAE and GAN

Training time: In general, VAEs are faster to train than GANs. This is because VAEs use a simpler loss function that can be optimized using gradient descent. GANs, on the other hand, require training of both the generator and discriminator networks, which can be computationally expensive.
Image quality: GANs are known for producing high-quality images that are almost indistinguishable from real images. It has higher quality.