

https://dspace.org/

Improving Semi-Supervised Learning Generative Adversarial Networks

Moolla, Faheem

2023-08
University of the Witwatersrand, Johannesburg
https://hdl.handle.net/10539/41694
Moolla, Faheem. (2023). Improving Semi-Supervised Learning Generative Adversarial Networks. [Master's dissertation, University of the Witwatersrand, Johannesburg]. https://hdl.handle.net/10539/41694

Downloaded from a DSpace repository. https://dspace.org/

Improving Semi-Supervised Learning Generative Adversarial Networks

---

Faheem Moolla

Supervisors:

Terence van Zyl

Hairong Bau



A research report submitted in partial fulfillment of the requirements for the degree of MSc (CW/RR) in Artificial Intelligence

in the

School of Computer Science and Applied Mathematics

University of the Witwatersrand, Johannesburg

4 August 2023

Declaration

I, Faheem Moolla, declare that this proposal is my own, unaided work. It is being submitted for the degree of MSc (CW/RR) in Artificial Intelligence at the University of the Witwatersrand, Johannesburg. It has not been submitted for any degree or examination at any other university.



Faheem Moolla

4 August 2023

Abstract

Generative Adversarial Networks (GANs) have shown remarkable potential in generating high-quality images, with semi-supervised GANs providing a high classification accuracy. In this study, an enhanced semi-supervised GAN model is proposed wherein the generator of the GAN is replaced by a pre-trained decoder from a Variational Autoencoder. The model presented outperforms regular GAN and semi-supervised GAN models during the early stages of training, as it produces higher-quality images. Our model demonstrated significant improvements in image quality across three datasets - namely the MNIST, Fashion MNIST, and CIFAR-10 datasets - as evidenced by higher accuracies obtained from a Convolutional Neural Network (CNN) trained on generated images, as well as superior inception scores. Additionally, our model prevented mode collapse and exhibited smaller oscillations in the discriminator and generator loss graphs compared to baseline models. The presented model also provided remarkably high levels of classification accuracy, by obtaining 99.32% on the MNIST dataset, 92.78% on the Fashion MNIST dataset, and 83.22% on the CIFAR-10 dataset. These scores are notably robust as they improved some of the classification accuracies obtained by two state-of-the-art models, indicating that the presented model is a significantly improved semi-supervised GAN model. However, despite the high classification accuracy for the CIFAR-10 dataset, a considerable drop in accuracy was observed when comparing generated images to real images for this dataset. This suggests that the quality of those generated images can be bettered and the presented model performs better with less complex datasets. Future work could explore techniques to enhance our model's performance with more intricate datasets, ultimately expanding its applicability across various domains.

Acknowledgements

The author would like to thank Professor Terence van Zyl and Dr. Hairong Bau for undertaking the supervision of this project, with a special mention of their time, insight, ideas, and contributions towards this research. This research could also not have been completed without the help of Dr. Helen Robertson, whose insight and assistance in undertaking a thesis made it possible.

Contents

Declaration i

Abstract ii

Acknowledgements iii

List of Figures vi

List of Tables viii

1 Introduction 1
- 1.1 Literature review . . . 2
- 1.2 Research Question . . . 5
- 1.3 Research Aim and Objectives . . . 6
  - 1.3.1 Research Aim . . . 6
  - 1.3.2 Objectives . . . 6
- 1.4 Limitations . . . 7
- 1.5 Overview . . . 7

2 Research Methodology 8
- 2.1 Research Design Overview . . . 8
  - 2.1.1 Process Overview . . . 8
  - 2.1.2 Model Overview . . . 9
- 2.2 Instruments and Technologies Used . . . 9
- 2.3 Data Access and Processing . . . 10
  - 2.3.1 MNIST . . . 10
  - 2.3.2 Fashion-MNIST . . . 11
  - 2.3.3 CIFAR-10 . . . 12

2.3.4 Latent Space Representations . . . 12
2.4 Selected Machine Learning Models . . . 16
2.4.1 VAE and Pre-trained Generator . . . 16
2.4.2 GAN . . . 18
2.4.3 Image Classification . . . 23
2.5 Hyperparameter Tuning . . . 25
2.5.1 Overview . . . 25
2.5.2 Chosen Method . . . 25
2.6 GAN Model and Comparison . . . 27

3 Results and Discussion **30**
3.1 Hyperparameter Optimisation Results . . . 30
3.1.1 GAN . . . 30
3.1.2 VAE . . . 31
3.2 GAN results . . . 31
3.2.1 Image quality . . . 31
3.2.2 Classification Accuracy . . . 35
3.2.3 Loss graphs . . . 36
3.3 CNN Results . . . 37
3.4 Statistical Analysis . . . 38
3.4.1 Inception Score . . . 38
3.4.2 Confusion Matrices . . . 39
3.4.3 Statistical Significance . . . 41
3.5 Classification Accuracy and Comparison . . . 43
3.6 Overview of Baseline Comparisons . . . 43
3.7 Overall Results . . . 45

4 Conclusion **46**
4.1 Summary . . . 46
4.2 Conclusion . . . 47
4.3 Future Work . . . 47

Bibliography **49**

List of Figures

2.1 Overview of design methodology process. 8
2.2 Correlation Plots of the MNIST and FMNIST datasets. 11
2.3 Correlation plot of CIFAR-10 dataset. 12
2.4 Scatter Plot indicating the latent space of the data for the MNIST and FMNIST datasets. 13
2.5 CIFAR-10 Scatter Plot indicating the latent space of the data. 14
2.6 VAE Validation loss indicating the values obtained for different latent space dimensions for the MNIST dataset. 15
2.7 VAE Validation loss indicating the values obtained for different latent space dimensions for the FMNIST dataset. 15
2.8 VAE Validation loss indicating the values obtained for different latent space dimensions for the CIFAR-10 dataset. 16
2.9 Architecture used for the VAE model and decoder for MNIST. 19
2.10 VAE used as the generator in the GAN. 21
2.11 Semi-supervised GAN network. 22
2.12 Architecture used in the stacked discriminator model. 23
2.13 CNN network. 25

3.1 Generated MNIST images after different number of epochs. 32
3.2 Generated FMNIST images after different number of epochs. 33
3.3 Generated CIFAR-10 images after different number of epochs. 34
3.4 MNIST GAN loss functions. 36
3.5 FMNIST GAN loss functions. 37
3.6 CIFAR-10 GAN loss functions. 37
3.7 Inception scores for the MNIST and FMNIST datasets. 39
3.8 Inception score for the CIFAR-10 dataset. 39
3.9 MNIST Confusion Matrix 40

3.10 FMNIST Confusion Matrix 41
3.11 Confusion Matrix for the CIFAR-10 dataset. 42

List of Tables

2.1 VAE Latent dimensions for each dataset 14
2.2 Search range for hyperparameters 27

3.1 Hyperparameter values used in GAN 30
3.2 Epoch values used in VAE 31
3.3 Classification results of the classifier from the stacked discriminator in the proposed GAN model (*n* represents the number of labelled samples used during training of the GAN) 35
3.4 Average classifier scores and standard deviations for each model . . 38
3.5 Classifier scores for each dataset 43

Chapter 1

Introduction

The Fourth Industrial Revolution has brought about a paradigm shift in the technological landscape, where Artificial Intelligence (AI) has emerged as a crucial component. One of the major components of AI is Machine Learning, which has revolutionised how computers learn and optimise. This discipline leverages data and algorithms to imitate human learning, fostering numerous advancements in the broader field of AI. Several learning methods exist within the domain of Machine Learning, including supervised learning, unsupervised learning, as well as semi-supervised learning [19].

Generative modeling is a subset of unsupervised machine learning that involves discovering and learning patterns from data so that the model can create new outputs similar to the probabilistic distribution of the original dataset [5]. Generative Adversarial Networks (GANs) are one example of a generative model. GANs have been widely used in the fields of Natural Language Processing and Computer Vision. They improve the Restricted Boltzmann Machine, vulnerable to modified data [6].

GANs have been applied to various tasks, including image generation, image translation, 3D object generation, video prediction, and object detection [3, 6]. Given the versatility of GANs, they have become increasingly popular in AI, with numerous use cases in the 21st century. Despite the extensive research that has been conducted on GANs, there is still ample room for improvement in this emerging and constantly evolving field.

This study aims to improve the architecture of GANs by incorporating semi-supervised learning into the GAN training process. The objective is to enhance the performance of GAN outputs, as well as the accuracy of the classification of GAN outputs by utilising semi-supervised learning to train part of the GAN architecture. In addition, the study aims to improve classifying the outputs of GANs into different categories to make them more useful.

In recent years, semi-supervised learning has gained attention due to its ability to learn from labelled and unlabelled data. It is a useful technique for improving the performance of machine learning models [18]. By incorporating semi-supervised learning into the GAN training process, the proposed method aims to leverage the benefits of both supervised and unsupervised learning, resulting in a more efficient and effective model [10].

Furthermore, clustering the outputs of GANs into different categories can enable the creation of additional labelled datasets, which can be used to train other machine learning models. Data clustering can be especially useful in cases where labelled data is scarce or expensive to obtain [26]. The proposed method aims to enhance the performance of the GAN and the classification of the GAN outputs. This study has aimed to contribute to the ongoing research and development of GANs, and it has the potential to be applied to a range of tasks in various fields.

The rest of this report is structured as follows: the rest of Chapter 1 introduces the reader to similar projects undertaken in this field, as well as the project's goals, aims, and objectives. Chapter 2 provides the methodology used for the research and the methods and design principles used. Chapter 3 provides the results obtained from the implementation of the study and an in-depth analysis of those results. Conclusions are drawn from all chapters in Chapter 4.

1.1 Literature review

Due to considerable progress in the digital landscape, with the introduction of computers, cell phones, and social media, there is a deep need for evolving technologies to satisfy the requirements of many fields [22]. With the introduction of Generative Adversarial Networks and their versatility, many of these requirements have been

satisfied such as image generation, image manipulation, and natural language processing [6]. Yann LeCun, a renowned professor known as one of the 'Godfathers of Artificial Intelligence', termed GANs 'the coolest idea in machine learning in the last 20 years' [23]. To evolve these GANs further, much research has been conducted on GANs since their inception in 2014 [4].

GANs were introduced in 2014 through a research paper by Ian Goodfellow. They are based on zero-sum games [4]. A zero-sum game is a game in which players are strictly opposed to each other, where one player's gain is equivalent to the other player's loss. The GAN is designed to have a two-part architecture comprising a generator and a discriminator. These components are trained in parallel. The generator generates synthetic data, and the discriminator's function distinguishes between the real data used in the training process and the generated data. The objective of this training process is to optimise the performance of the generator so that it can produce high-quality synthetic data. In the case of the GAN, the distribution of generated data should be as close as possible to the distribution of real-world data. This is crucial in ensuring that the generated data has practical applications and can be used for various tasks [6].

Many different types of GANs have been developed since 2014. All of them are based on the same base architecture, with modifications and variations to improve them [17]. Mehdi Mirza proposed the Conditional GAN (CGAN), which improved regular GANs by refining the quality of generated samples [13]. Mirza proposed Deep Convolution Generative Adversarial Networks (DCGAN) to overcome a few of the issues of general GANs. These issues include the training of a GAN being too slow, the large number of parameters, including weights, and the generalisation effect. DCGAN uses a Convolutional Neural Network, which uses backpropagation to achieve this [3].

Other GAN models were created for specific use cases, such as CycleGAN, which was established for image translation without a one-to-one mapping between domains [29]. WGAN was introduced by Arjovksy et al. as a variant of GANs that uses the Wasserstein distance to measure the difference between the generated and real data distributions. This approach leads to more stable training and better results [2]. Other GANs were also created for specific use cases, such as StyleGAN. The StyleGAN architecture uses a novel mapping network to transform a

low-dimensional latent space into a high-dimensional feature space. This mapping allows precise control over the generated image's style and appearance [7].

With the different GAN models being developed, an interest was naturally sparked in using contrasting supervision techniques during training. Semi-Supervised GAN (SGAN) was one of the first works on using semi-supervised learning in GANs and was published by Augustus Odena [15]. Unlike traditional GANs, where the discriminator makes a binary classification between 'real' or 'fake', SGAN introduces a novel feature in its architecture. Odena's discriminator is designed to perform as a multiclass classifier, predicting which of N classes an input belongs to. The number of classes is then extended to N+1 during training, with the additional class corresponding to the outputs generated by the generative model G. This methodology, Odena demonstrates, results in a more data-efficient classifier and enhances the quality of samples generated compared to a regular GAN." Thus, the aim shifted from focusing on bettering the generator to bettering the discriminator. This then improved the regular GAN network as labels were attached to the generated samples, and the network became more data-efficient. Training times for the generator part of the network also decreased. SGAN provided better accuracy for limited labelled data than Convolutional Neural Networks (CNNs) [15].

In 2016, researchers from OpenAI proposed an improved technique for using GANs through semi-supervised learning. Tim Salimans et al., [20], proposed a model that added to the work done by Odena, [15], by improving and simplifying the architecture of the discriminator in the GAN network. The model was tested on multiple datasets, and the results illustrated that the visual quality of the images produced had increased [20]. Kumar, Sattigeri, and Fletcher [10] proposed further improvements to semi-supervised learning GANs by observing fake samples in the training process. The research helped to improve the architecture as even fewer samples of labeled data were needed [10].

Though there was much success for the GANs proposed by Goodfellow, [6], and the subsequent improved models, there are still some problems that GANs face, such as convergence. According to the ideologies of game theory, Nash Equilibrium in a GAN can be reached in theory. However, in reality, too many parameters must be added to the loss functions of the generator and discriminator to achieve the required balance. One factor that can prevent convergence is mode collapse, which

results in a lack of diversity in the samples produced by the generator. When gradient descent is optimised, the generator generates similar data samples repeatedly, making it easier to deceive the discriminator. [3].

GANs may also experience difficulties in training due to the vanishing gradient problem, which can result in poor convergence. Several modifications have been proposed, including using residual connections [12]. GANs may also face challenges when dealing with imbalanced datasets, where some classes have fewer examples than others. Imbalanced datasets can result in poor performance and biased sampling. Several techniques have been proposed to address this issue, such as Class-Conditional GANs [14].

While much work has been done, this study aims to improve upon the already-existing GAN architecture by overcoming some of the problems faced in GANs and enhancing image quality and classification. Xianwen Yu et al. proposed VAEGAN, which uses a Variational AutoEncoder (VAE) acting as a generator in conjunction with the discriminator from the traditional GAN model [25]. This model does produce higher-quality images but makes it difficult for the model to learn the latent distribution and for the discriminator to be well-trained. This study differs in that it disconnects the generator of the GAN network and uses a VAE to pre-train the generator before connecting it to provide the discriminator with better samples and higher-quality images at earlier training times while using semi-supervised learning. The image quality is verified with a Convolutional Neural Network (CNN).

1.2 Research Question

With the ever-increasing use of GANs in AI, more research is constantly undertaken to improve the architecture to obtain better and more useful outputs. The architecture and output can be improved by refining the machine learning techniques or designing new ones. More specifically, the following research questions can be asked:

1. Could the samples at early epochs of a semi-supervised GAN model be improved by pre-training the generator using a Variational Auto Encoder? Can this architecture lower the risk of mode-collapse and fast convergence of the discriminator?

2. Can the output data of a GAN be labelled and classified into different categories with a high level of accuracy?

1.3 Research Aim and Objectives

The research aims and objectives are essential in understanding the scope of the work. The aim of the research encapsulates what the research study sets out to do, while the goals described are the tasks that assist with leading up to the project's aim.

1.3.1 Research Aim

This research aims to improve upon GAN architecture that utilises semi-supervised learning to train the discriminator of the GAN, through slower convergence of the discriminator during training, avoiding mode collapse, and improving the quality and labelling of generated samples. The research further aims to classify the outputs of the GAN better to be repurposed and to understand the performance better.

1.3.2 Objectives

The following objectives are to be achieved in the research to achieve the aims:

1. Implement a VAE and use the decoder as a pre-trained generator for a GAN
2. Implement a GAN network using semi-supervised learning as well as the pre-trained generator.
3. Make comparisons between the proposed semi-supervised GAN model results and existing semi-supervised learning GAN results using many methods, including statistical analysis.
4. Use machine learning methods to classify the outputs of the GANs to improve the labelling of the GAN output.

1.4 Limitations

As with any project, there are always constraints or limitations that apply. The following limitations apply to this research:

- Timing - a good balance of timing was needed. The training of machine learning models takes a long time, and with more data making it more accurate, sufficient time is needed to be allocated for training the models. A Gantt chart was created with the ideal timelines for each task of the project to overcome this
- The best performance is achieved using Google Colab for coding due to the high number of GPUs and a large amount of memory, as there is no access to a supercomputer, and thus a stable internet connection was required for training and testing purposes
- Many datasets had to be used for training, and therefore the implementation of the code had needed to be completed well in advance to allow the machine learning models time to learn and analysis to be conducted

1.5 Overview

The experimental research involves creating and training Generative Adversarial Networks using semi-supervised learning. The generator of the GAN was pre-trained using a VAE. This network, a typical semi-supervised GAN network, and a regular GAN network were trained. The samples were compared to understand the accuracy and results and whether the pre-training of the generator had aided in the results. The GANs were trained on three datasets: MNIST, Fashion MNIST (FMNIST), and CIFAR-10, each containing a substantial number of images ranging between 60 000 to 70 000. In the semi-supervised learning GAN, a subset of the samples was labelled to facilitate the training process. The study also refined the output of GANs by creating an image classifier in the form of a Convolutional Neural Network (CNN) to work in conjunction with the output of the GAN models, so that the image quality and images can be analysed and better understood. Statistical analysis was then performed to understand the accuracy of the results.

Chapter 2

Research Methodology

2.1 Research Design Overview

2.1.1 Process Overview

In order to undertake the research, a design methodology had to be created. The first step was to identify which technologies would be needed for the undertaking of the project. Once technologies were obtained, the data had to be accessed and processed. Due to the amount of datasets used, this was a lengthy process and the datasets needed to be analysed. Once the data was in a state that could be used, the model selection and building had to take place. This was carefully thought out as there are various models that could be used for the aforementioned research. In order to obtain the best results out of the models, once they were built the hyper-parameters were trained and fine-tuned. The entire process is illustrated in Figure 2.1.

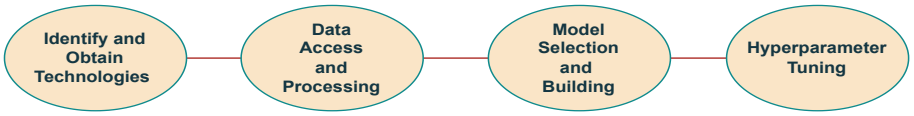


FIGURE 2.1: Overview of design methodology process.

2.1.2 Model Overview

The first part of the model-building process was the creation of the Variational Autoencoder (VAE) and, specifically, its decoder, which was used as the generator of the proposed semi-supervised Generative Adversarial Network (GAN). The GAN model was then built with a stacked architecture for the discriminator, and trained using mainly unlabelled data (unsupervised learning) and a small number of labelled data (supervised learning). Hence, the GAN was trained in a semi-supervised fashion overall.

Upon constructing the GAN, it was trained to utilise the decoder of the VAE as the generator, while employing multiple distinct datasets. Subsequently, the generated outcomes of the aforementioned GAN (termed SSGAN for this study) were juxtaposed with those of other GAN models, including a normal Semi-Supervised GAN model, which was also trained with the same datasets. An extensive evaluation of each model was carried out to ascertain which one performed better.

The output of the GANs was further classified into different categories, and the quality of the images was tested with a classifier. Many classifiers can be used to achieve this. A common method used is the K-Nearest Neighbour (KNN) method. This method is based on a feature space and classifies images to the closest samples of that image. The method is simple. However, it uses all features equally for similarity computing, which leads to classification errors. Another method is using a Support Vector Machine (SVM). SVMs represent features in space using a hyperplane and contain major limitations concerning the size and speed of images during training [27]. The Convolutional Neural Network (CNN) is a more popular method, which performs better than SVMs and KNNs for image recognition tasks [9]. The SSGAN outputs were thus run through a Convolutional Neural Network to obtain an indication of the quality of the generated images.

2.2 Instruments and Technologies Used

This study uses various technologies to run the required expirements:

- Laptop: Dell 11th Generation Intel Core i7 with 16GB RAM and 64-bit operating system.

- Google Colab: Google Colab is an online cloud-computing platform for developers that allows users to use the GPU power provided by them. There exists a version with limited usage which is free to use and was thus used to run the models. Google Colab is integrated with Google Drive. This allows for the output data sets to be easily stored. As Google Colab allows users to gain access to GPUs, the speed with which Google Colab trained the models was faster than if it were to run locally on a regular laptop.
- Python 3: The code for the study was written in the Python 3 programming language.
- Pycharm: The Pycharm IDE and the Anaconda environment were used to conduct testing locally.

2.3 Data Access and Processing

Three datasets were used to train and evaluate the models; MNIST, Fashion-MNIST, and CIFAR-10. The choice of evaluating three different datasets was made to provide a fairer comparison to other models and evaluate the performance of the models more accurately. All three datasets were obtained using *Keras* from the *Tensorflow* library in Python.

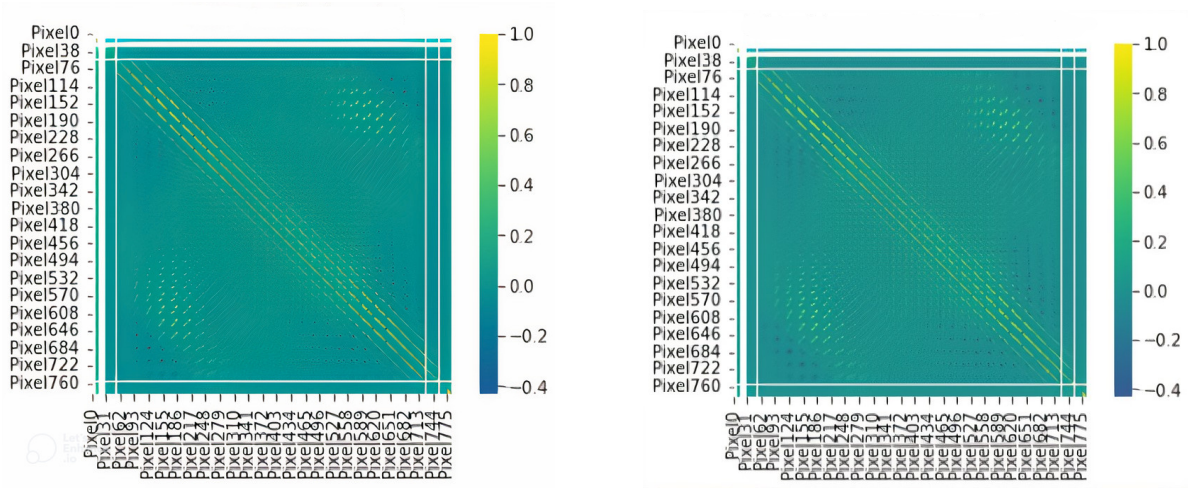
2.3.1 MNIST

The MNIST dataset is a commonly used benchmark dataset in computer vision. It consists of 70 000 images of handwritten digits (0 to 9) that are 28x28 pixels. The images were split according to the optimal splitting values built into *Keras*, with 60 000 images used for training and 10 000 for testing.

As each pixel in the dataset is in the same range [0,255], the MNIST dataset was normalised by dividing each pixel value by 255, which scales the pixel values to be in the range of [0,1]. This normalisation step helps improve the convergence of machine learning models, ensuring that all input features are on a similar scale. Figure 2.2a shows a correlation plot of the data set. The correlation plot represents the pairwise correlation of the dataset's features, which helps to more intuitively identify relationships in the data. A correlation coefficient close to 1 means that

there is a very strong positive correlation between variables, while -1 means there is a very strong negative correlation between variables. This correlation plot was generated by obtaining a correlation matrix of the input datasets' dataframe, and then utilising Seaborn's *heatmap* function.

The plot indicates that the data is well-correlated, as shown by the diagonal and the data surrounding it. In machine learning, the presence of well-correlated data holds great significance as it facilitates the identification of meaningful patterns and relationships between the features of a given dataset. This, in turn, contributes to enhancing the accuracy of predictions made by machine learning models, thereby resulting in improved overall model performance.



(A) MNIST Correlation Plot

(B) FMNIST Correlation Plot

FIGURE 2.2: Correlation Plots of the MNIST and FMNIST datasets.

2.3.2 Fashion-MNIST

The Fashion-MNIST dataset is another popular benchmark data set for computer vision tasks. It comprises 70 000 images of clothing items, including ten different classes, such as T-shirts, dresses, and shoes. The images are also 28x28 pixels in size. The images were split with the same ratios as MNIST.

Like MNIST, the Fashion-MNIST dataset was also normalised by dividing each pixel value by 255, which scales the pixel values to the range [0,1]. Figure 2.2b

shows the data correlation plot of the data set. As with MNIST, the data is well-correlated.

2.3.3 CIFAR-10

The CIFAR-10 dataset is more complex than MNIST and Fashion-MNIST, consisting of 60 000 RGB images of ten different classes such as aeroplanes, birds, and cats, and are 32x32 pixels in size. The images were split with 50 000 images used for training and 10 000 for testing, according to the optimal splitting values built into *Keras*.

As with MNIST and Fashion-MNIST, the CIFAR-10 dataset was normalised by dividing each pixel value by 255, scaling the pixel values to the range [0,1]. Figure 2.3 shows a correlation plot of the data set. Similar to MNIST and Fashion-MNIST, there is a good correlation between the data.

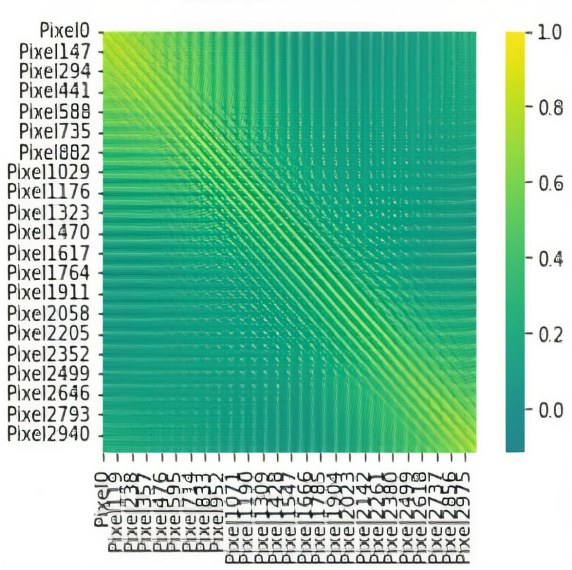


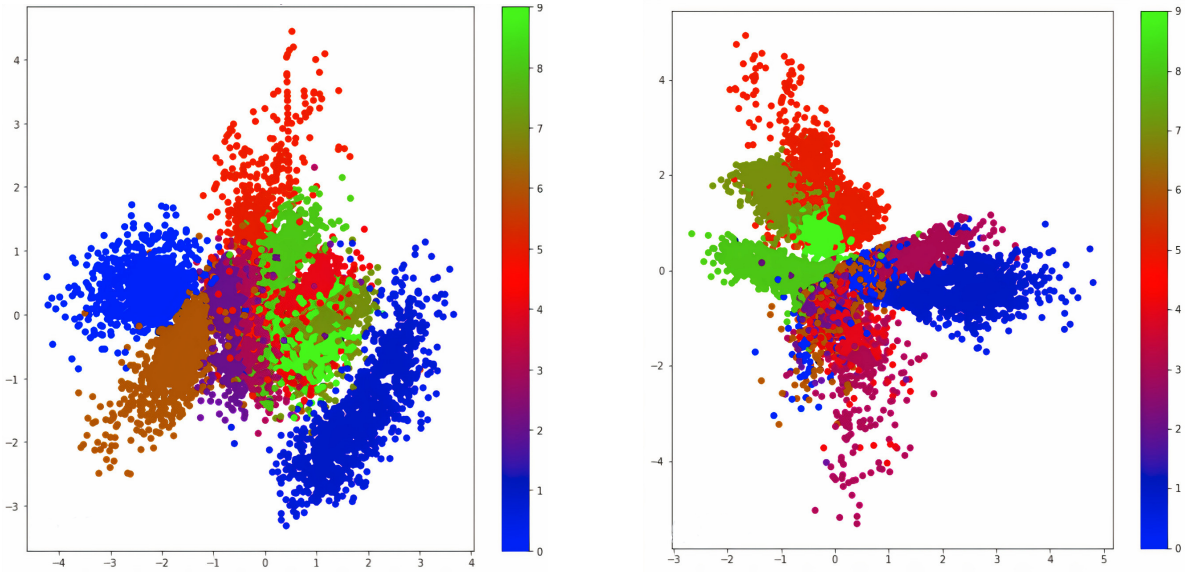
FIGURE 2.3: Correlation plot of CIFAR-10 dataset.

2.3.4 Latent Space Representations

The latent space representation of the validation datasets, identified by the bottleneck of the VAE, are illustrated for the MNIST and FMNIST datasets in Figures 2.4a and 2.4b respectively, and for the CIFAR-10 dataset in Figure 2.5. These plots help to understand better how the data is clustered together, and in the case of the VAE it

helps to show how the VAE has learned to represent images in a lower-dimensional space, where each colour on the chart represents a different class. This was done through the following steps:

1. The encoder is fed the input dimensions of the images and the mean vector of the latent space representation z_{mu}. The encoder then maps the data to the latent space.
2. The encoder predicts the latent space representation for the validation dataset.



(A) MNIST Scatter Plot (B) FMNIST Scatter Plot

FIGURE 2.4: Scatter Plot indicating the latent space of the data for the MNIST and FMNIST datasets.

In a well-trained VAE, the latent space is expected to capture meaningful representations of the input data. In the case of MNIST, this means that points corresponding to the same digit class should cluster together, exhibiting some level of separation from points of other classes. The encoder maps the input images to the latent space, and the decoder reconstructs the images from the latent space back to the original image space.

The scatter plots can provide insights into the quality of the VAE, and do not affect the GAN. If the points form distinct and well-separated clusters for each digit class, it suggests that the VAE is successfully learning a meaningful representation of the

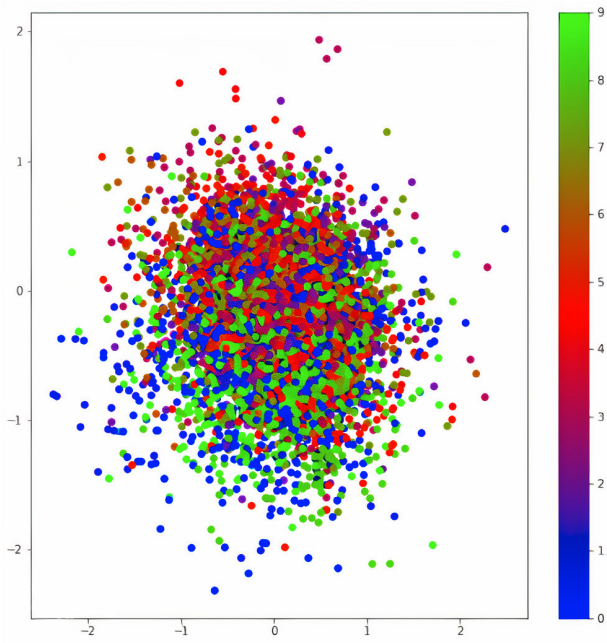


FIGURE 2.5: CIFAR-10 Scatter Plot indicating the latent space of the data.

data. On the other hand, if the points are scattered without clear separation, it indicates that the VAE is struggling to capture the underlying structure of the data.

The plots show that the data is well-clustered for MNIST and FMNIST, while there clustering for CIFAR-10 is more sporadic and the VAE is struggling more to learn the latent space of this data.

For each of the datasets, the latent dimension that resulted in the lowest average validation loss was selected and incorporated into the architecture of the models. The specific values for the chosen latent dimensions are presented in Table 2.1.

TABLE 2.1: VAE Latent dimensions for each dataset

	Dataset		
	MNIST	**FMNIST**	**CIFAR-10**
Latent Dimensions	2	5	10

The latent space dimensions for these datasets were chosen through running experiments for each, where latent dimensions of 2, 5, 10, 20 and 100 were tested. Figures 2.6, 2.7 and 2.8 indicate the experiments run in the VAE for the MNIST, FMNIST and

CIFAR-10 datasets respectively, where the validation loss was measured for each of the runs.

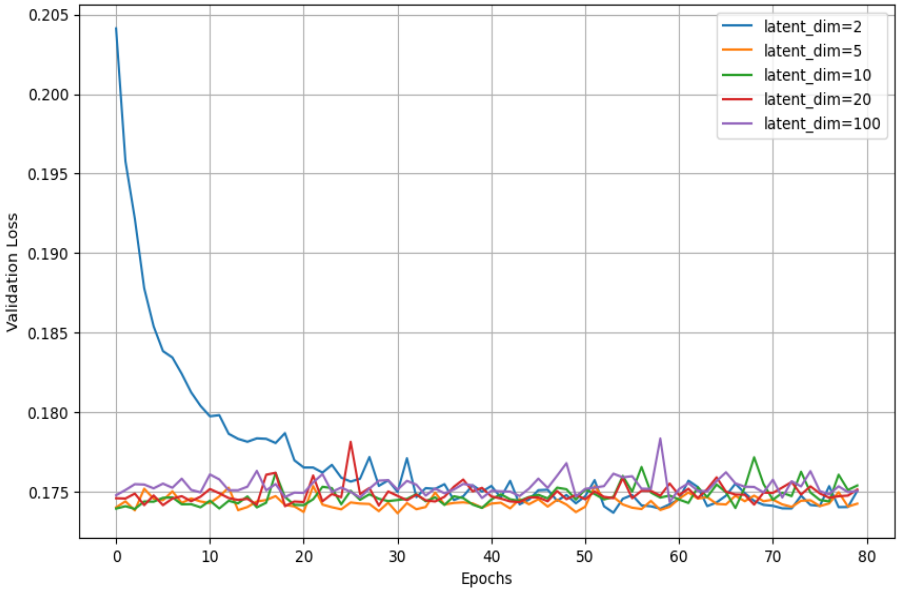


FIGURE 2.6: VAE Validation loss indicating the values obtained for different latent space dimensions for the MNIST dataset.

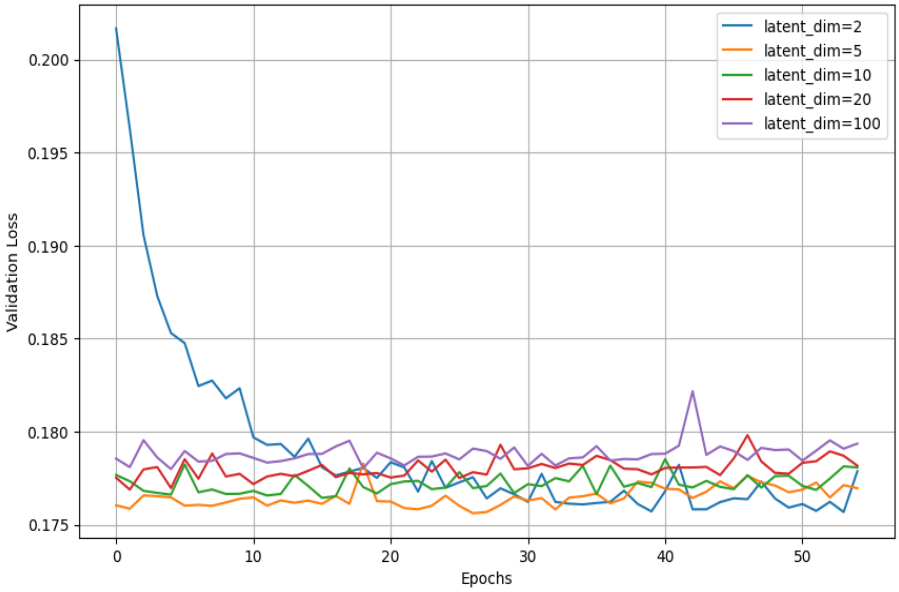


FIGURE 2.7: VAE Validation loss indicating the values obtained for different latent space dimensions for the FMNIST dataset.

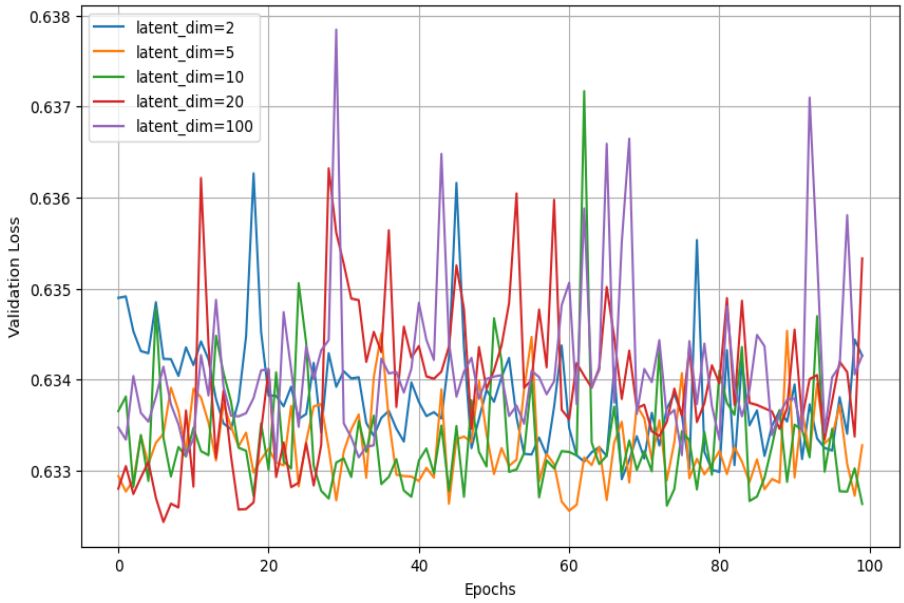


FIGURE 2.8: VAE Validation loss indicating the values obtained for different latent space dimensions for the CIFAR-10 dataset.

2.4 Selected Machine Learning Models

2.4.1 VAE and Pre-trained Generator

To attempt to better the architecture of SGAN, the generator of the GAN was pre-trained using the decoder from a Variational Autoencoder (VAE). VAEs are a type of artificial neural network (ANN), having been derived from an autoencoder. They are used extensively in object detection, dimensionality reduction, and image generation [8].

VAEs are based on two key ideas: encoding the input data into a latent representation and decoding the latent representation back into data space. The encoding and decoding processes are modeled by neural networks, and the VAE objective function is optimised using variational inference [8].

The VAE objective function consists of the reconstruction loss and the KL-divergence regularisation terms. The reconstruction loss measures the difference between the input and reconstructed data from the latent representation. The KL-divergence regularisation term ensures that the learned latent representation follows a standard normal distribution [8].

The VAE objective function is provided in Equation 2.1, where $\mathcal{L}VAE$ is the VAE loss function, $q_\phi(z|x)$ is the encoder network that maps the input data x to the latent representation z, $p_\theta(x|z)$ is the decoder network that maps the latent representation z to the reconstructed data $\hat{x}$, and $p(z)$ is the standard normal distribution.

$$\mathcal{L}VAE = \mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}(q_\phi(z|x)||p(z)) \tag{2.1}$$

The first term in the VAE loss function is the reconstruction loss, which measures the difference between the input sample x and the reconstructed one $\hat{x}$. This term is typically modeled as the negative log-likelihood of the data given the latent representation z. The reconstruction loss can be written as in Equation 2.2, where D is the number of samples, x_i and $\hat{x}_i$ are the i-th elements of the input and reconstructed data, respectively, and σ^2 is the variance of the reconstruction distribution.

$$\log p_\theta(x|z) = -\frac{1}{2}\sum_{i=1}^{D}(x_i - \hat{x}_i)^2 - \frac{D}{2}\log(2\pi\sigma^2) \tag{2.2}$$

The second term in the VAE loss function is the KL divergence regularisation term, which ensures that the learned latent representation learns a distribution. The KL-Divergence is represented in Equation 2.3, where K is the dimensionality of the latent space, and μ_i and σ_i are the mean and standard deviation of the i-th dimension of the latent representation z, respectively.

$$D_{KL}(q_\phi(z|x)||p(z)) = -\frac{1}{2}\sum_{i=1}^{K}(1 + \log\sigma_i^2 - \mu_i^2 - \sigma_i^2) \tag{2.3}$$

To train the VAE, the VAE loss function needs to be optimised for the model parameters θ and ϕ. Stochastic Gradient Descent (SGD) or other optimisation algorithms can be utilised to achieve this optimisation. For this study, the Adam optimiser was used. The learning rate used in the Adam optimiser for the MNIST, FMNIST and CIFAR-10 datasets was 0.0002, 0.0002 and 0.0001, respectively, while the *beta 1* values were all set at 0.5. During training, a random z is sampled from the latent space to reconstruct a new data point. To reconstruct a new data point, this sampled z is fed into the decoder part of the VAE. The decoder then outputs a reconstruction of a data point. Since the model has been trained to reconstruct similar original inputs for similar points in the latent space, the sampled point z is expected to be decoded

into a plausible new data point.

For this study, the VAE for the MNIST dataset was created using the architecture seen in Figure 2.9. The same model architecture was used and trained on each of the data sets mentioned in Section 2.3, with an exception to the input and output layers, as well as the latent dimensions, to cater to the different image sizes of the datasets. The blue block in the image depicts where the latent dimensions are specified, and change according to Table 2.1. The image also includes the decoder architecture in the yellow block. As with the VAE, the only difference between this architecture and that of the FMNIST and CIFAR-10 datasets are the changes in latent dimensions.

2.4.2 GAN

GANs are a class of deep learning models used to generate new data instances that resemble an existing dataset. The architecture of a GAN consists of two neural networks, namely a generator network and a discriminator network, that play an adversarial game against each other [6]. The generator network transforms a random input vector into a data instance that resembles the training data. The discriminator network takes a data instance from the training data or the generator and classifies it as real or fake. The generator is trained to create data instances that fool the discriminator, and the discriminator is trained to classify real and fake data instances correctly [6].

Formally, the generator G and discriminator D networks are trained using the objective function in Equation 2.4, where $p_{data}(x)$ is the distribution of real data instances, $p_z(z)$ is the distribution of random input vectors, $D(x)$ is the discriminator's output when given a data instance x, and $G(z)$ is the generator's output when given a random input vector z. The objective of the discriminator is to maximise this function, while the generator's objective is to minimise it.

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim pdata(x)}[\log D(x)] + \mathbb{E}_{z\sim p_(z)}[\log(1 - D(G(z)))] \tag{2.4}$$

The equations for the generator and discriminator are provided in Equations 2.5 and 2.6, respectively, where z is the input vector to the generator, X is the output of the generator, θ_g are the parameters of the generator, $p_{data}(X)$ is the probability of

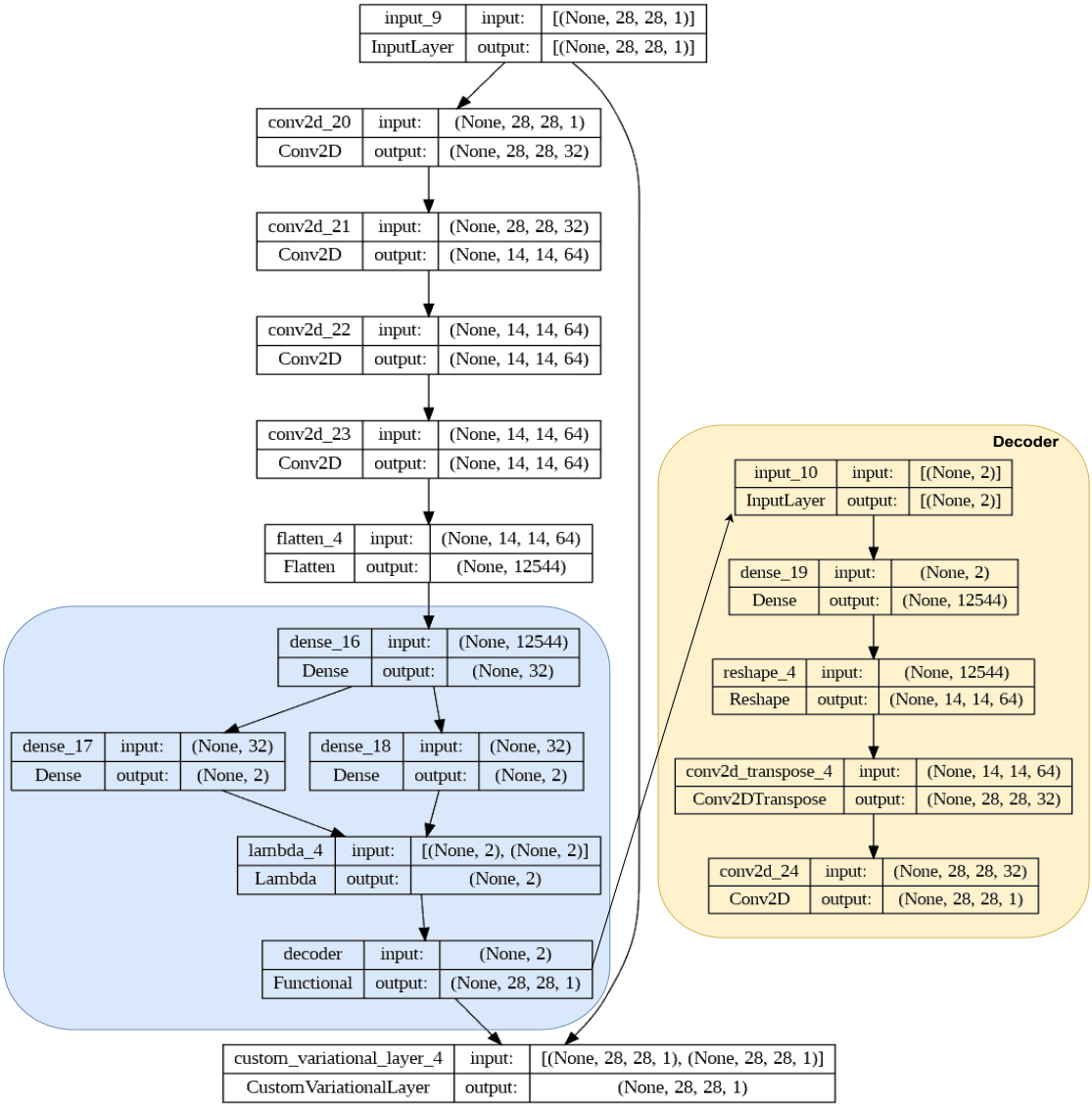


FIGURE 2.9: Architecture used for the VAE model and decoder for MNIST.

X being a real data instance, and θ_d are the parameters of the discriminator.

$$G(z;\theta_g) = X \tag{2.5}$$

$$D(X;\theta_d) = p_{data}(X) \tag{2.6}$$

After training, the generator can generate new data instances by sampling from the distribution of random input vectors and passing them through the generator

network. The generated data instances can be used for various applications, such as image synthesis, data augmentation, and anomaly detection [3].

The optimisation of GANs is a challenging task due to the min-max nature of the objective function in Equation 2.4. The generator and discriminator are trained alternately, which can lead to instability and oscillations in the training process. One common problem is mode collapse, where the generator produces limited variations of the same data instances. Various techniques have been proposed to address this issue, such as adding noise to the input of the discriminator, using different architectures for the generator and discriminator, and using regularisation techniques to prevent overfitting [1, 2, 6].

In a GAN, mode collapse and overfitting are closely related because the generator and discriminator are constantly adversarial. Suppose the discriminator becomes too good at distinguishing real and fake samples. In that case, the generator may struggle to generate new and diverse samples that can deceive the discriminator, leading to mode collapse. Conversely, if the generator becomes too good at generating samples that can deceive the discriminator, the discriminator may overfit the generated samples, leading to a poor generalisation of new data. Therefore, finding the right balance between the generator and discriminator is critical to avoiding mode collapse and overfitting in GANs [1, 20].

Another challenge in GAN optimisation is the vanishing gradient problem, where the gradients of the loss function concerning the parameters of the generator network become very small or even zero. The vanishing gradient problem can occur when the discriminator is too effective in distinguishing real and fake data instances. The generator cannot produce data instances that fool the discriminator. Various techniques have been proposed to address this issue, such as alternative loss functions less susceptible to the vanishing gradient problem or gradient penalty methods that penalise the discriminator for large gradients [2].

To try and address some of these issues, for the architecture of the proposed model, the VAE decoder was used as a pre-trained generator for the GAN. The generator was also replaced with a decoder to improve the quality of images at earlier epochs of the GAN training process. Figure 2.10 illustrates the VAE being used as the generator of the GAN network.

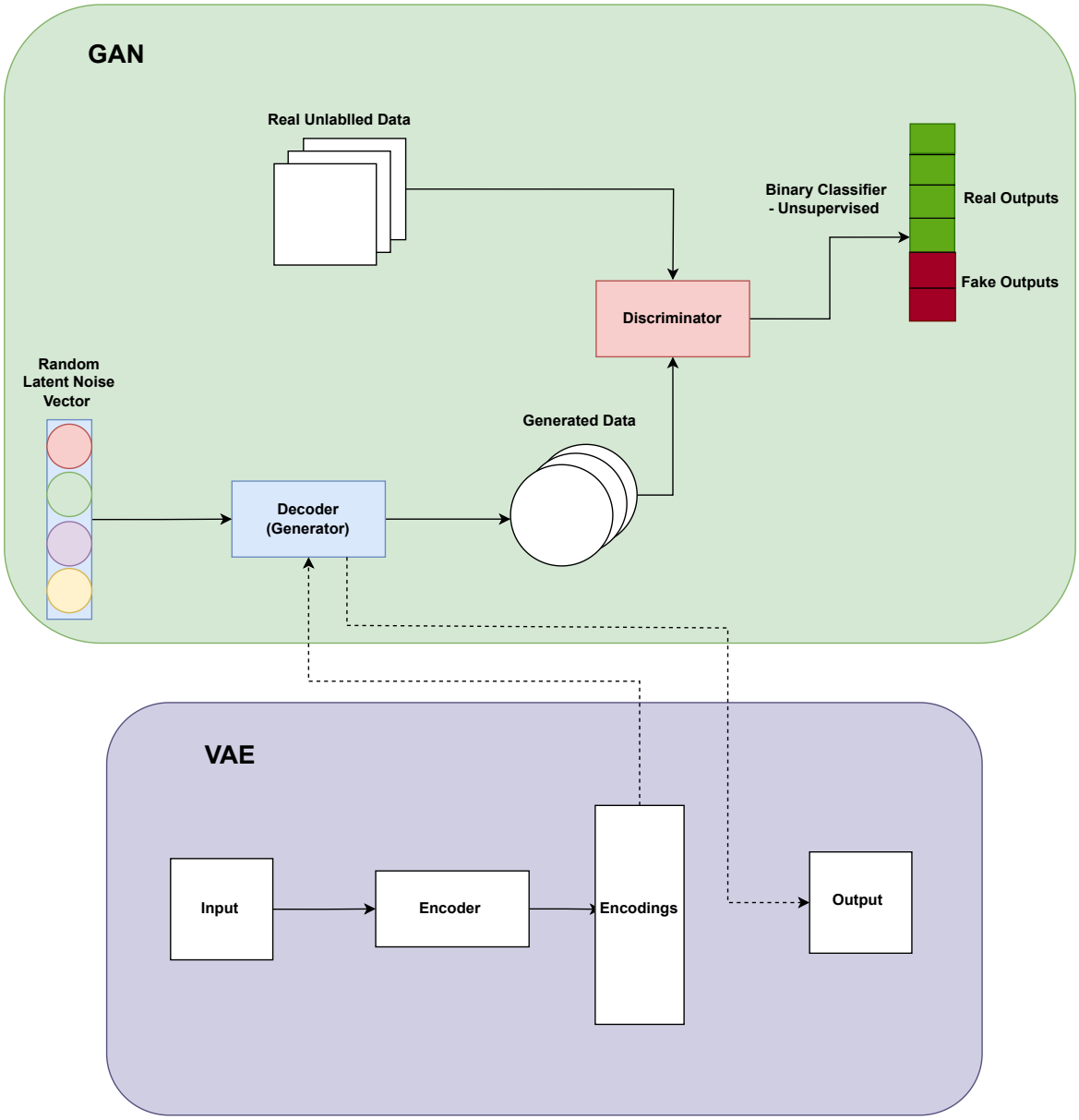


FIGURE 2.10: VAE used as the generator in the GAN.

Figure 2.11 illustrates the architecture for a GAN. The yellow blocks in the figure indicate the additional parts of the architecture for a semi-supervised GAN compared to a regular GAN. As seen from Figure 2.10, the architecture remains mostly the same, except for the labelled data input to the discriminator to help train it for semi-supervised learning.

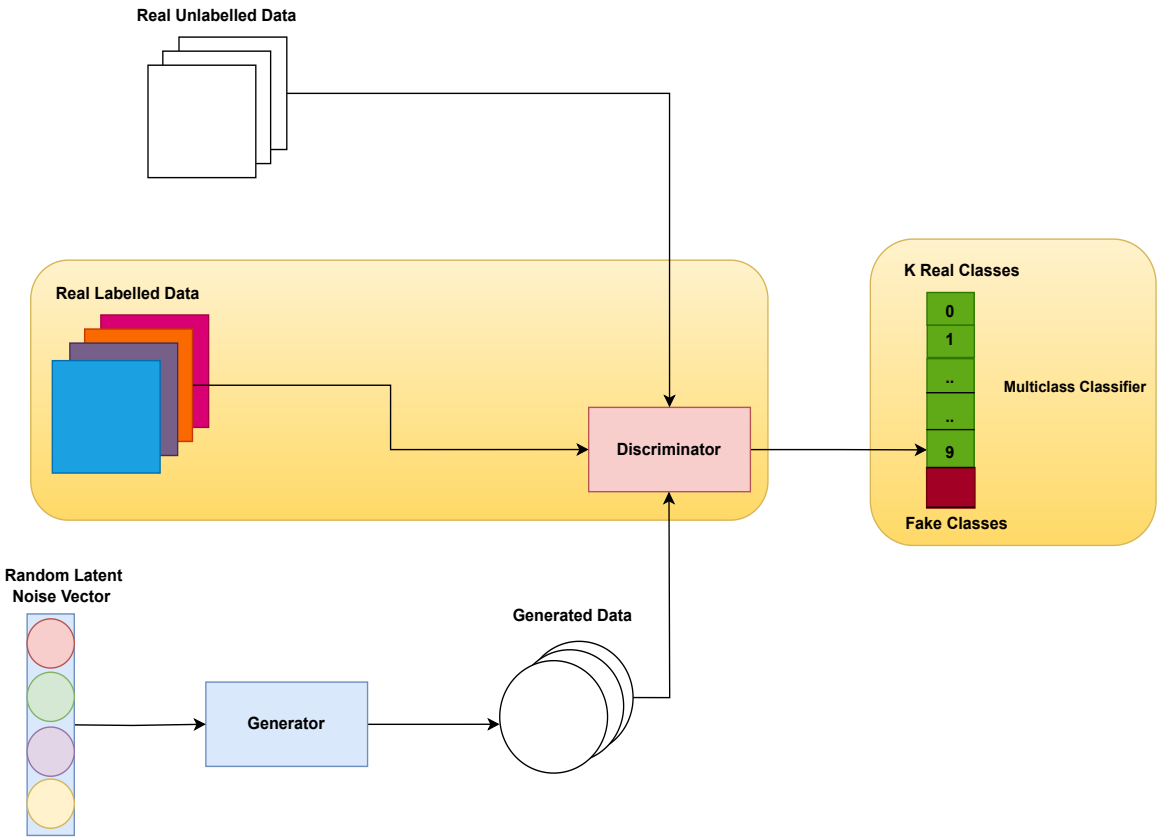


FIGURE 2.11: Semi-supervised GAN network.

The architecture of the generator built for this study is illustrated in Figure 2.9 as the VAE decoder architecture. Figure 2.12 illustrates the discriminator built for this study. This illustrates a stacked discriminator model as mentioned in the beginning of Chapter 2. The stacked discriminator model is a model with multiple layers in the discriminator where the supervised output layer is stacked on top of the unsupervised output layer. The reason for these layers is for the model to be semi-supervised and learn from labelled as well as unlabelled images.

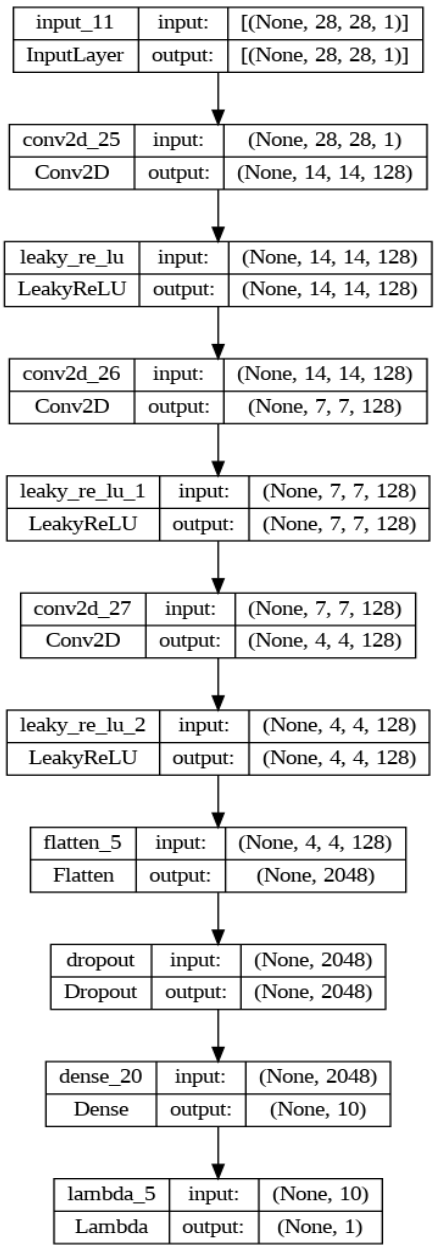


FIGURE 2.12: Architecture used in the stacked discriminator model.

2.4.3 Image Classification

As stated in Section 2.1, various algorithms can be used to classify data. It was found that CNNs generally perform best when classifying images; thus, this research uses CNNs.

A CNN is a deep neural network primarily used for image and video recognition. It has become one of computer vision's most widely used deep learning models.

CNNs are based on a mathematical operation called convolution, which is applied to the input image to extract features. The convolutional operation is defined in Equation 2.7, where f and g are functions, and $*$ denotes the convolution operation. In the context of CNNs, f is typically the input image, and g is the filter or kernel used to extract features.

$$(f * g)(x) = \int_{-\infty}^{\infty} f(\tau)g(x - \tau)d\tau \tag{2.7}$$

The output of the convolution operation is a feature map, which represents the filter's response at different locations in the input image. Applying filters to the input image is repeated to extract additional features, which are then used to train the classifier and make predictions.

A CNN typically consists of multiple layers, each performing convolution with a specified set of filters. The basic layers of a CNN are:

1. Convolutional layer - This layer applies multiple filters to the input image and generates multiple feature maps.
2. Activation layer - This layer applies a non-linear activation function to the output of the convolutional layer. The most commonly used activation function is the Rectified Linear Unit (ReLU) function, which is defined in Equation 2.8.
3. Pooling layer - This layer reduces the spatial dimensionality of the feature maps by downsampling them. The most commonly used pooling operation is max-pooling, which takes the maximum value in a rectangular region of the feature map.

$$f(x) = max(0, x) \tag{2.8}$$

The output of a CNN is obtained by passing the input image through all the layers and then applying a softmax function to the output of the last layer. The general architecture of a CNN is illustrated in Figure 2.13.

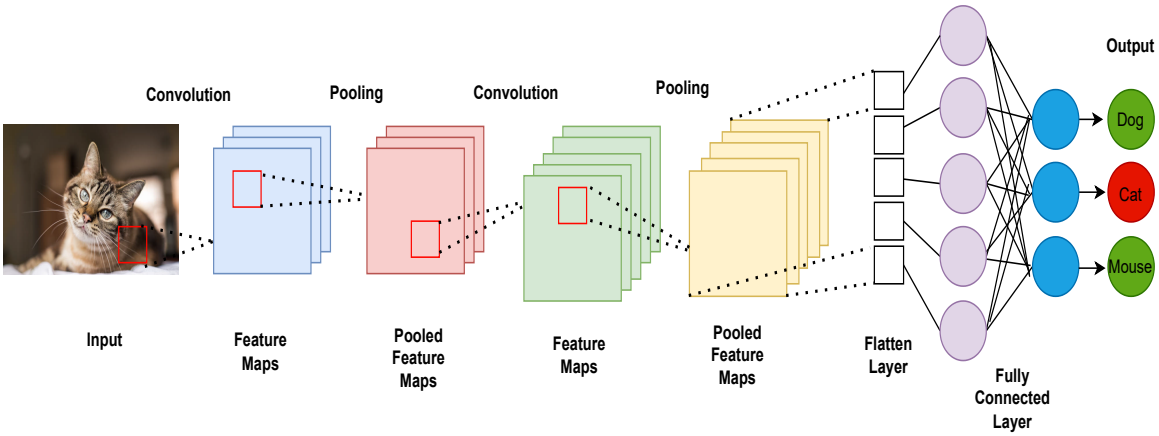


FIGURE 2.13: CNN network.

2.5 Hyperparameter Tuning

2.5.1 Overview

To create the most effective and accurate results, hyperparameter tuning is essential in any machine learning and deep learning model. Hence, it is a crucial step in optimising the performance of a deep learning model. A GAN consists of multiple components, including the generator and discriminator, each with its own set of hyperparameters. These hyperparameters may significantly affect the quality of the generated output. Therefore, finding the optimal values for these hyperparameters is crucial in achieving the best performance for the GAN. Similarly, the fine-tuning of hyperparameters of the CNN is crucial for optimal classification performance.

2.5.2 Chosen Method

To tune the hyperparameters of the GAN and CNN, the Keras Tuner was used [16]. The Keras Tuner library is compatible with Python, allowing one to quickly find the best hyperparameter of a model. The library also has built-in Bayesian Optimisation, Hyperband, and Random Search algorithms. For these models, Bayesian Optimisation was used to tune the hyperparameters. Bayesian Optimisation was chosen over other optimisation techniques for several reasons.

- Efficient Exploration: Bayesian Optimisation uses a probabilistic model to estimate the performance of untested hyperparameters. This allows it to explore

the parameter space more efficiently than traditional grid or random searches. The algorithm can quickly identify which regions of the search space are likely to have high-performing hyperparameters and focus the search on those regions.

- Better Exploration-Exploitation Balance: Bayesian Optimisation balances the parameter space exploration by exploiting promising regions. The algorithm selects new hyperparameters based on the estimated mean and variance of the model's performance, thus directing the search to the promising areas while allowing for exploring other parts of the search space. It does this through the use of an acquisition function seen in Equation 2.9, where $\mu(\mathbf{x})$ is the mean of the posterior distribution, $\sigma(\mathbf{x})$ is the standard deviation of the posterior distribution, and κ is a trade-off parameter that balances exploration and exploitation.
- Ability to Handle Noise: Bayesian Optimisation can handle noise in the objective function, which is common in GANs. The probabilistic model allows the algorithm to model the uncertainty of the objective function and make more informed decisions about which hyperparameters to test next.
- Fewer Evaluations Required: Because Bayesian Optimisation uses a probabilistic model to guide the search, it requires fewer evaluations of the objective function than other optimisation techniques, such as grid or random search. This helps to speed up selection.

$$\text{UCB}(\mathbf{x}) = \mu(\mathbf{x}) + \kappa\sigma(\mathbf{x}) \tag{2.9}$$

For this study, the SSGAN used the Bayesian Optimiser to tune the learning rate, batch size and dropout rate. The loss functions, noise dimensions and the number of layers remained constant, as seen in the architecture of the GAN components in Figures 2.9 and 2.12. The following parameters were passed to the Bayesian Optimiser for each of the datasets:

- *num_initial_points*: the number of random initialisations to perform before the Bayesian Optimisation is carried out - 10
- *max_trials*: the maximum number of trials to run - 100

- *seed*: the random seed used during the search - 42

The hyperparameter ranges specified for each of the hyperparameters being optimised are observed in Table 2.2:

TABLE 2.2: Search range for hyperparameters

Hyperparameter	Start Point	End Point	Step
Batch Size	32	256	16
Dropout Rate	0.1	0.5	0.1
Learning Rate	0.00001	0.001	0.00001

The optimiser evaluates the GANs loss function on a validation set using different combinations of hyperparameters and recommends the set of hyperparameters that achieves the best performance.

Early Stopping was used for the various datasets to obtain the number of epochs. Early Stopping is a technique used to track a statistic, such as validation loss, and when a stopping criterion is reached, the training process terminates. In the case of this study, the stopping criterion used was a change in the classifier accuracy of the SSGAN model. If there was a decrease in the accuracy by over 3% from the highest obtained accuracy, then the learning process was terminated.

2.6 GAN Model and Comparison

The SSGAN was trained and tested against the accuracy measures of two baseline GAN-based models: a regular GAN and a semi-supervised GAN. Specifically, a regular GAN and a semi-supervised GAN trained on the same datasets were compared. To compare accuracy, statistical tests were run. Statistical tests are crucial in Artificial Intelligence, particularly in GANs. Because GANs are based on complex mathematical models, it is essential to use statistical tests to evaluate the quality and reliability of the generated data.

One commonly used statistical test in GANs is the Inception Score (IS). The Inception Score is a measure of the quality of generated images that considers both the

realism and diversity of the images. The score is based on the idea that a good generator should generate diverse, high-quality images. This can be achieved by training a classifier on the generated and authentic images and measuring their predictions' differences. Due to the IS emphasising images' diversification and sharpness, it is a good evaluation index for GANs [21]. The IS is computed as the exponential of the class probabilities' entropy minus the entropy's expected value, which measures the diversity of the generated images. A higher IS score indicates that the generated images are of higher quality and diversity.

Specifically, the Inception Score is defined as in Equation 2.10, where x is a generated sample, $p(y|x)$ is the conditional probability distribution, $p(y)$ is the marginal probability distribution, and D_{KL} denotes the Kullback-Leibler divergence.

$$IS(x) = \exp\left(\mathbb{E}_{x\sim p_g} D_{KL}(p(y|x))||p(y)\right) \tag{2.10}$$

The Fréchet Inception Distance (FID) score is another commonly used metric for evaluating the quality of generated images in GANs. Unlike the Inception Score, the FID score considers the similarity between the real and generated data distribution by calculating the Fréchet distance between the two distributions in a high-dimensional feature space. In terms of generated images, the FID score is a more robust and accurate measure than the Inception Score, as it considers both the quality and the diversity of the generated images.

The FID score is represented in Equation 2.11, where x_{real} is an actual data sample, x_{gen} is a generated data sample, μ is the mean of the features, Σ is the covariance matrix of the features, and Tr is the trace of the matrix. A lower FID score indicates that the generated samples are more similar to the actual data samples, and thus the GAN model is of higher quality.

$$FID(x_{real}, x_{gen}) = \left\|\mu_{real} - \mu_{gen}\right\|^2 + Tr(\Sigma_{real} + \Sigma_{gen} - 2\sqrt{\Sigma_{real}\Sigma_{gen}}) \tag{2.11}$$

While the FID score is a more accurate measure of the generated images' quality than the Inception Score, it requires a large number of actual data samples to be accurate. For smaller datasets, the Inception Score is a better choice as it is less sensitive to the size of the dataset and can provide a reasonable estimate of the

quality of the generated images [24]. However, the FID score is a better choice for larger datasets as it provides a more accurate and robust measure of the quality of the generated images [24]. For this reason, as this study uses small datasets, the Inception Score is measured rather than the FID.

Chapter 3

Results and Discussion

3.1 Hyperparameter Optimisation Results

3.1.1 GAN

After using Bayesian Optimisation and Early Stopping, the hyperparameters used for the Generative Adversarial Network (GAN) were obtained for each of the three different datasets. The obtained hyperparameter values are provided in Table 3.1.

TABLE 3.1: Hyperparameter values used in GAN

Hyper-parameter	Dataset		
	MNIST	**FMNIST**	**CIFAR-10**
Epochs	50	80	145
Batch Size	256	160	128
Dropout Rate	0.1	0.2	0.4
Learning Rate	0.0002	0.0002	0.0001

The results presented in the table show that the batch size and the number of epochs differed for each dataset. This suggests that these hyperparameters are dataset-specific and require careful tuning to achieve optimal performance. The more complex datasets required more epochs for the GAN to learn the semantic features of the images.

Furthermore, it can be observed that the batch size and dropout rate were inversely proportional to the complexity of the dataset. Smaller batch sizes and a higher dropout rate can help prevent over-fitting, which is more likely to occur with complex datasets such as CIFAR-10.

The learning rate for each dataset remained similar, with 0.0001 being the lowest value from the range provided, as higher values above 0.0002 were seen to provide a sub-optimal performance.

To further understand why the Bayesian Optimisation algorithm selected the hyperparameter values, the architecture of the models can be observed. Figure 2.9 provides the architecture of the entire Variational Autoencoder (VAE) model along with the decoder, and hence generator for the GAN.

3.1.2 VAE

The VAE architecture that was used is illustrated in Figure 2.9. Early Stopping was also used for the VAE to determine the number of epochs used for each dataset. In the case of this model, the stopping criterion used was a change in the validation loss of the learned data. If the loss did not improve after five epochs, the model would stop training. Table 3.2 illustrates the epoch values obtained.

TABLE 3.2: Epoch values used in VAE

Hyper-parameter	**Dataset**		
	MNIST	**FMNIST**	**CIFAR-10**
# of Epochs	55	80	135

Similiar to the hyperparameter optimisations for the GAN, the VAE demonstrates that as the complexity of the dataset increases, the training process requires a greater number of epochs.

3.2 GAN results

With the complicated nature of the GAN and the research questions used, various results were obtained for each dataset.

3.2.1 Image quality

For qualitative analysis, the images generated by the GAN part of the presented model are included in this study. Figures 3.1, 3.2 and 3.3 show the quality of generated images at different epochs for the presented model compared to a regular

GAN model for the MNIST, FMNIST, and CIFAR-10 datasets respectively. As illustrated in the figures, the images shown for each dataset are better with the proposed model than a regular GAN model. After one epoch, the difference in image quality is clear due to the generator being pre-trained as the decoder from the VAE. The model presented in this study thus helps improve image quality at earlier epochs.

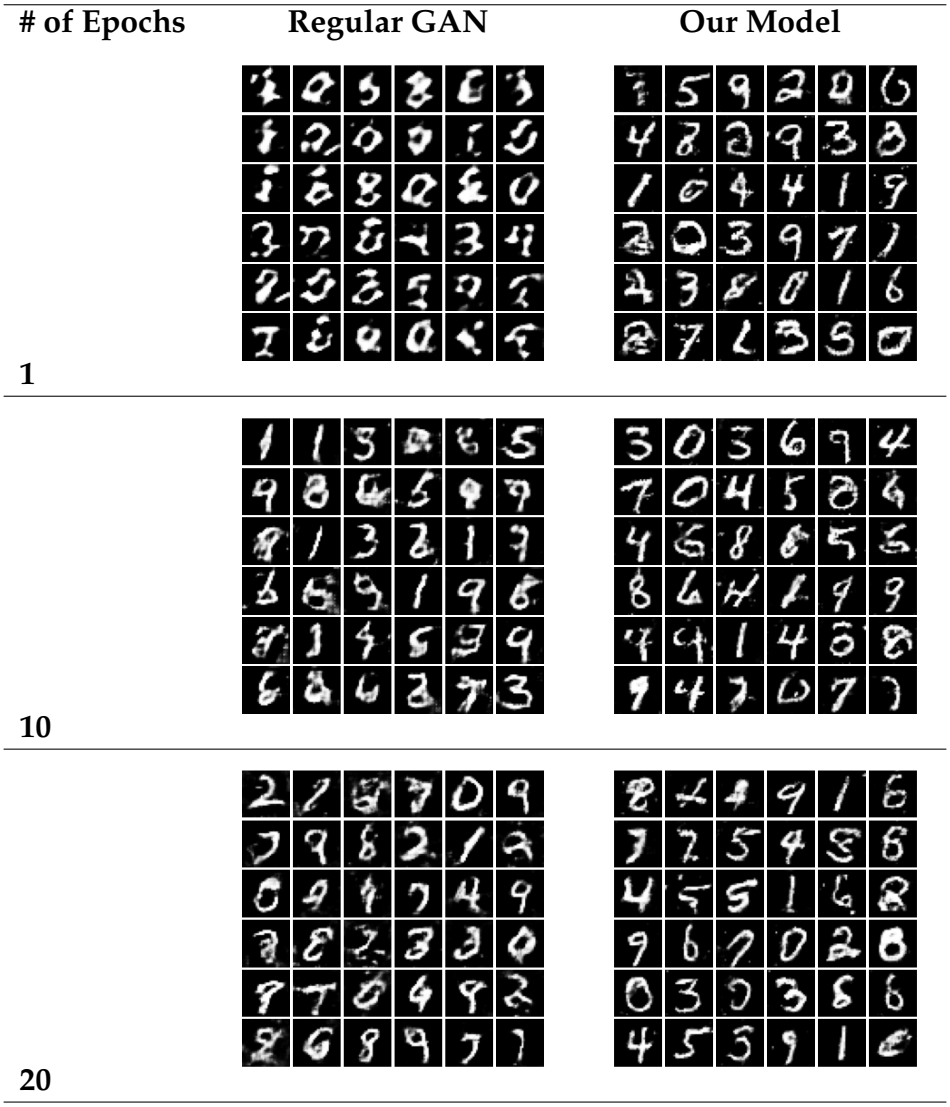


FIGURE 3.1: Generated MNIST images after different number of epochs.

Figures 3.1 and 3.2 show that the presented model produces distinguishable numbers and clothing shapes for the MNIST and FMNIST datasets respectively in early epochs. In contrast, a regular GAN model generates blurry shapes. The presented

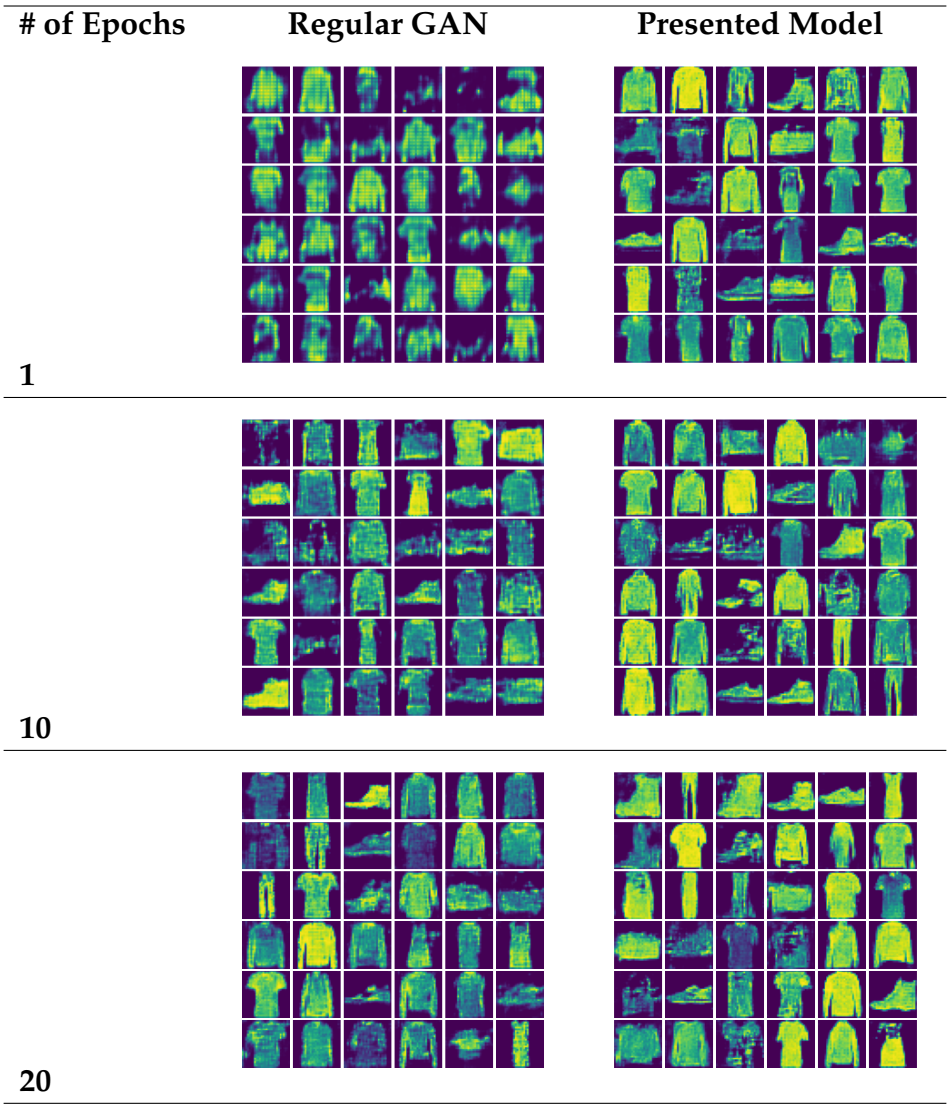


FIGURE 3.2: Generated FMNIST images after different number of epochs.

model also generates clearer numbers and clothing shapes for the later epochs compared to the regular GAN model.

Figure 3.3 presents the images from CIFAR-10. At early epochs, the images presented by the model in this study are clear but do not explicitly indicate animals. In contrast, the regular GAN model only presents blurry images. At later epochs, the CIFAR-10 images slightly increase in quality, but it is still difficult to distinguish which type of animal is presented. This suggests that the model presented in this

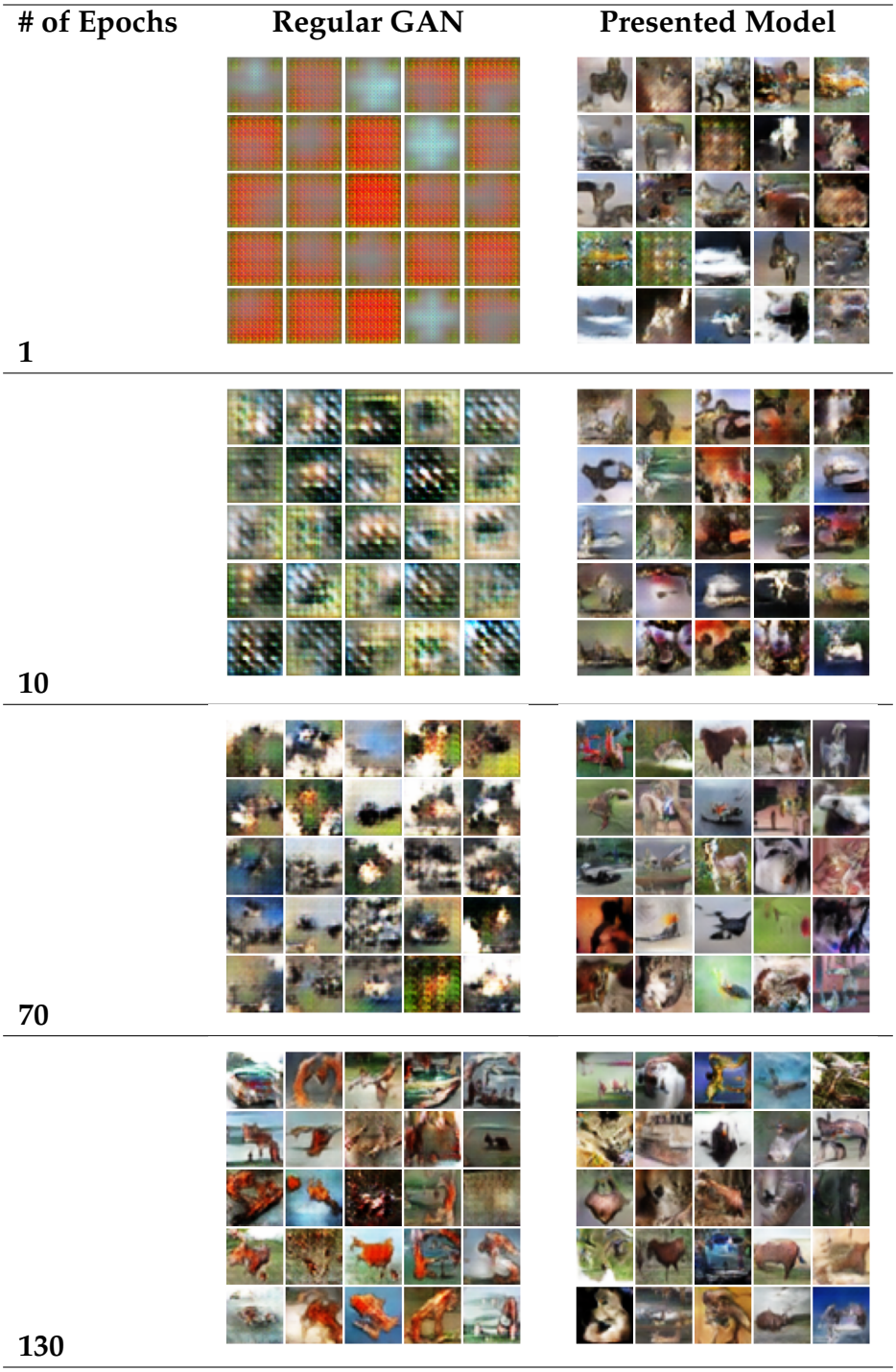


FIGURE 3.3: Generated CIFAR-10 images after different number of epochs.

study and its architecture work well for simpler datasets such as greyscale images but not for more complicated datasets such as RGB images. However, the model produces better-quality images than a regular GAN model. This figure contains images from the later epochs in order to indicate whether mode-collapse did indeed occur.

3.2.2 Classification Accuracy

The accuracy scores, along with their standard deviations from the classifier of the GAN are illustrated in Table 3.3. This table indicates the scores for each of the different datasets using the classification from the stacked discriminator model.

TABLE 3.3: Classification results of the classifier from the stacked discriminator in the proposed GAN model (n represents the number of labelled samples used during training of the GAN)

	MNIST (n=100)	**FMNIST (n=1000)**	**CIFAR-10 (n=4000)**
Accuracy	99.32% $\pm$ 0.21	92.78% $\pm$ 0.56	83.22% $\pm$ 1.08

The semi-supervised part of the GAN model was trained on 100 samples for the MNIST dataset, 1000 for the FMNIST dataset, and 4000 for the CIFAR-10 dataset. The choice of these values is due to state-of-the-art models using the same amount of labelled data for their models, so these values were chosen for comparison purposes. The proportionality also indicates that as the datasets become more complicated, more labelled samples are required.

The results in Table 3.3 demonstrate that the model's classification component successfully learned the images' latent features and accurately predicted their corresponding outputs. The model's performance on MNIST, which only had 100 labelled images, was exceptional, achieving a classification accuracy of 99.32%. This outcome indicates that the model can effectively handle small sample sizes and yield highly accurate results. Moreover, the model loss graphs provide additional support for the robustness of the model.

When applied to FMNIST, which had 1000 labelled images, the model's classification accuracy was slightly lower at 92.78%, but still impressive given the complexity of the dataset. In comparison, the CIFAR-10 dataset, with its intricate features and a greater variety of shapes within images, achieved an accuracy of 83.22%, which

is also commendable since the dataset contains many more complexities than the other datasets.

3.2.3 Loss graphs

Due to the research question over whether fast convergence of the discriminator and mode-collapse can be avoided, the loss graphs for the discriminator model need to be analysed. Figures 3.4, 3.5, and 3.6 illustrate the discriminator loss graphs for the MNIST, FMNIST, and CIFAR-10 datasets, respectively, as well as a comparison to a regular GAN model to determine the difference.

The figures also display the loss graphs of the generator, or in the case of this model, the decoder, as it is being updated through each epoch, as well as a comparison to a regular GAN model. It is important to note that the x-axis measures steps taken, not epochs.

As depicted in Figures 3.4, 3.5 and 3.6, the losses of normal GANs exhibit oscillatory behavior during the initial stages of the training process, gradually decreasing as the training progresses. The losses of the presented model show smaller oscillations from the start. In addition, the losses at the early and subsequent epochs have smaller differences and remain almost constant. The presented model also shows smaller standard deviations compared to the normal GANs.

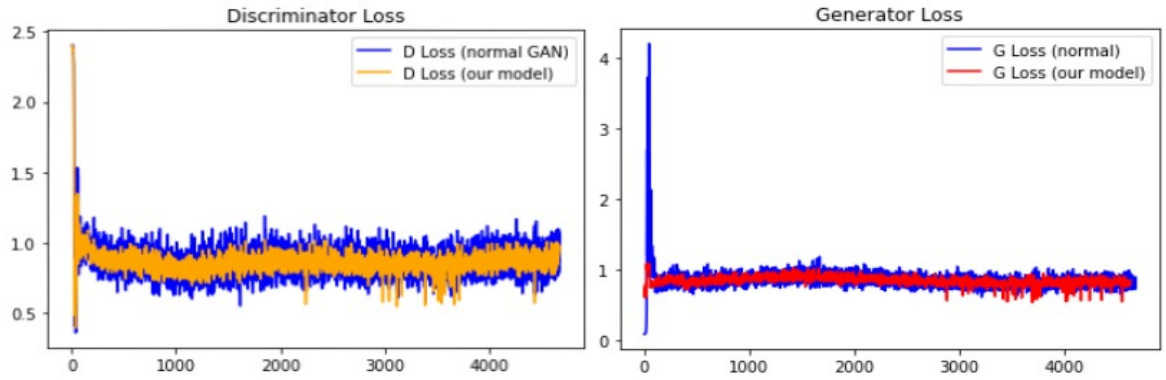


FIGURE 3.4: MNIST GAN loss functions.

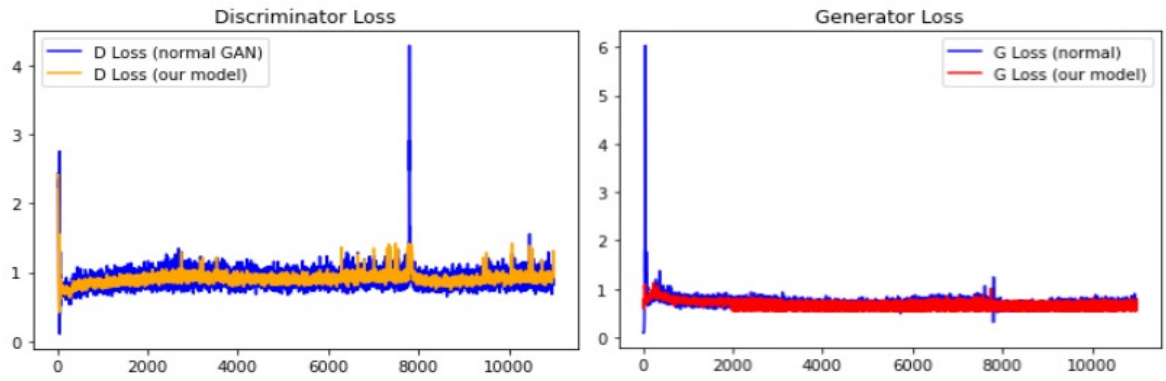


FIGURE 3.5: FMNIST GAN loss functions.

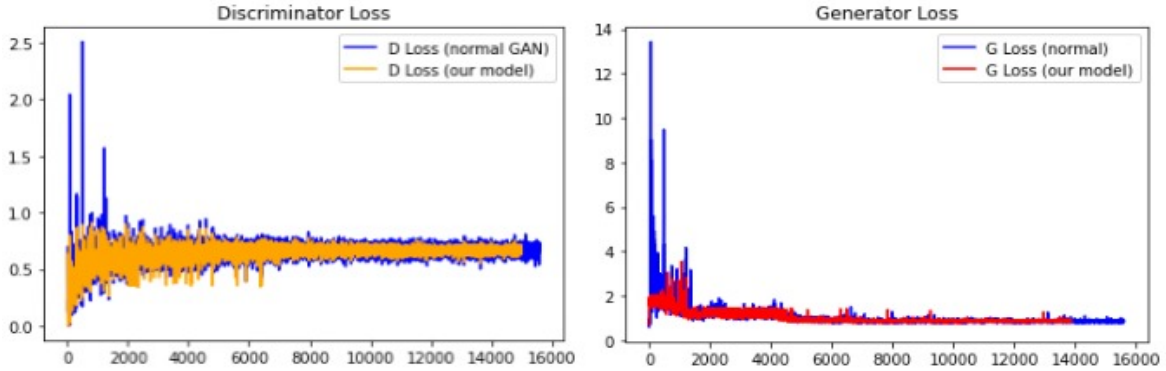


FIGURE 3.6: CIFAR-10 GAN loss functions.

3.3 CNN Results

The Convolutional Neural Network (CNN) was used to obtain an accuracy score for the quality of the images. This was achieved by training three CNNs: one with a real dataset, one with a dataset of generated images from just a regular semi-supervised GAN (RSGAN), and one with a dataset of generated images from the entire model presented in this study. All three CNNs were tested on the test set from the real dataset described in Chapter 2. The CNNs contained the same architecture.

The models were run five times for each dataset to ensure completeness, and the average score along with the standard deviation was recorded. The scores obtained for each of the CNNs on the various datasets are illustrated in Table 3.4.

For the MNIST dataset, the accuracy of the presented model in comparison to the RSGAN model was 5.17% higher, while for the FMNIST dataset, the accuracy was

TABLE 3.4: Average classifier scores and standard deviations for each model

Model	Dataset	Average Score
CNN (Real dataset)	MNIST	99.31% ± 0.15%
	FMNIST	93.17% ± 0.21%
	CIFAR-10	89.14% ± 0.37%
CNN (RSGAN images)	MNIST	92.87% ± 0.29%
	FMNIST	86.41% ± 0.44%
	CIFAR-10	60.11% ± 0.81%
CNN (Presented model images)	MNIST	98.04% ± 0.21%
	FMNIST	91.13% ± 0.33%
	CIFAR-10	69.67% ± 0.63%

4.72% higher. For the CIFAR-10 dataset, the accuracy is 9.56% higher. This table illustrates that the presented model outperformed the semi-supervised GAN baseline model for all datasets and thus indicates that pre-training the generator as the decoder of a VAE assists with semi-supervised learning as well, due to the discriminator having to keep up with the decoder and thus bettering itself greatly.

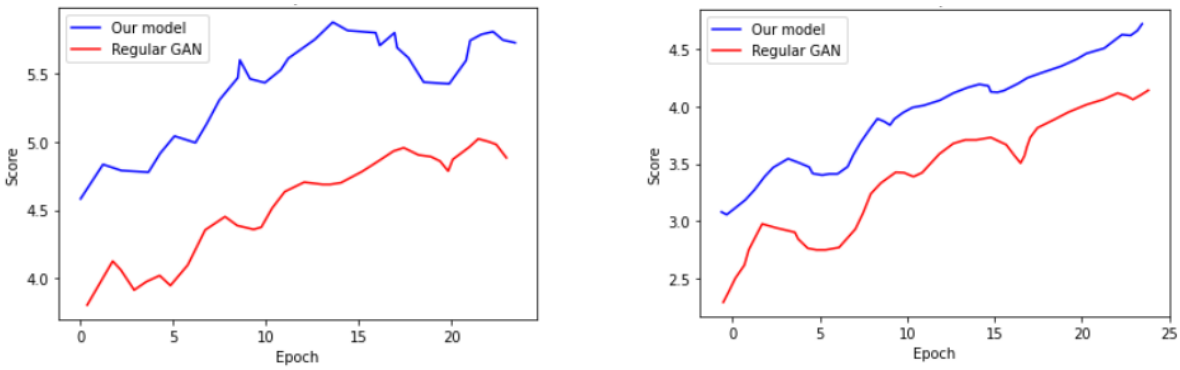
3.4 Statistical Analysis

As observed from the quality of the images in Figures 3.1, 3.2 and 3.3, the presented model outperforms the baseline models, which proves that the image quality is better with the presented model. The classification accuracies shown in Table 3.3 have also proven to be higher than the baseline models. However, for completeness and to advocate for this, statistical tests and statistical classifications were conducted.

3.4.1 Inception Score

The statistical test conducted was the Inception Score (IS). Figures 3.7a and 3.7b illustrate the Inception Scores for the MNIST and FMNIST datasets respectively, while Figure 3.8 indicates the IS for the CIFAR-10 dataset. Based on the figures, it can be concluded that the presented model shows a promising trend of improving Inception Scores over time and performs better than the baseline model, which suggests that it generates higher quality and more diverse images as training progresses. With the CIFAR-10 dataset, the Inception Score for the presented model

initially starts higher but eventually aligns with that of the baseline model. This suggests that the presented model performance is initially better at earlier epochs but reaches the same quality as the epochs progress.



(A) MNIST Inception Score (B) FMNIST Inception Score

FIGURE 3.7: Inception scores for the MNIST and FMNIST datasets.

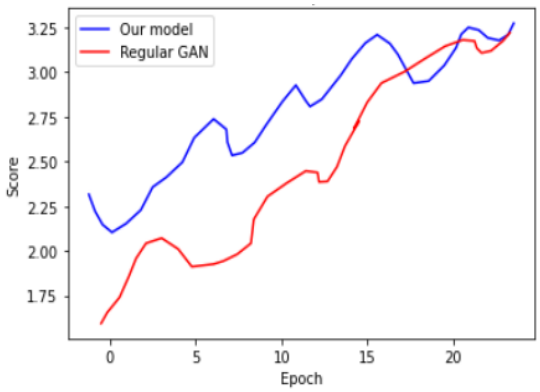


FIGURE 3.8: Inception score for the CIFAR-10 dataset.

Overall, the IS graphs indicate that the presented model outperforms the baseline and that the quality of generated images is generally high for the presented model.

3.4.2 Confusion Matrices

For statistical classification, confusion matrices were created. The benefits of using a confusion matrix are numerous. First, it provides a clear and concise way to measure the accuracy of a model. By comparing the predicted output to the actual output, a confusion matrix can help determine the sensitivity and specificity of the model. This information can then refine the model and improve its performance.

A confusion matrix can further help identify areas where the model is underperforming. Examining the false positive and false negative rates makes it possible to identify patterns or trends in the data causing the model to misclassify certain inputs. This information can then be used to adjust the model's parameters or improve the training data to address these issues.

Three confusion matrices were created, one for each of the datasets, to further strengthen the results obtained from the classifier. Figures 3.9 and 3.10 indicate the confusion matrices for the MNIST and FMNIST datasets respectively, while Figure 3.11 illustrates the confusion matrix for the CIFAR-10 dataset.

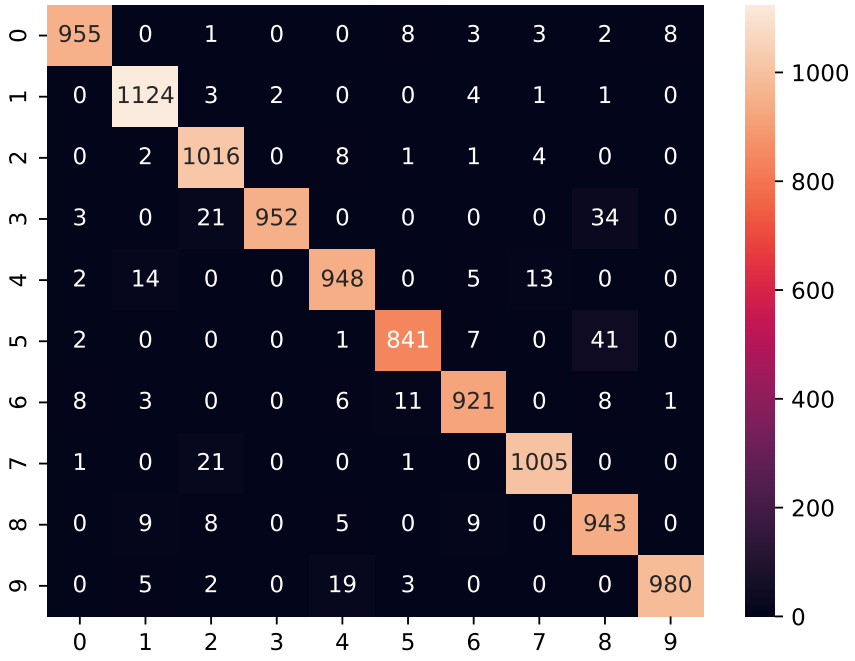


FIGURE 3.9: MNIST Confusion Matrix

The confusion matrices are read from left to right, where the y-axis indicates the true label and the x-axis indicates the predicted label. An ideal confusion matrix with 100% accuracy would have a full diagonal with 0's all around the diagonal. Figures 3.9, 3.10 and 3.11 show that most of the classification is on the diagonal close to the ideal matrix for all three datasets. This is observed more plainly for the MNIST dataset, as with an average classification accuracy above 99%, fewer than

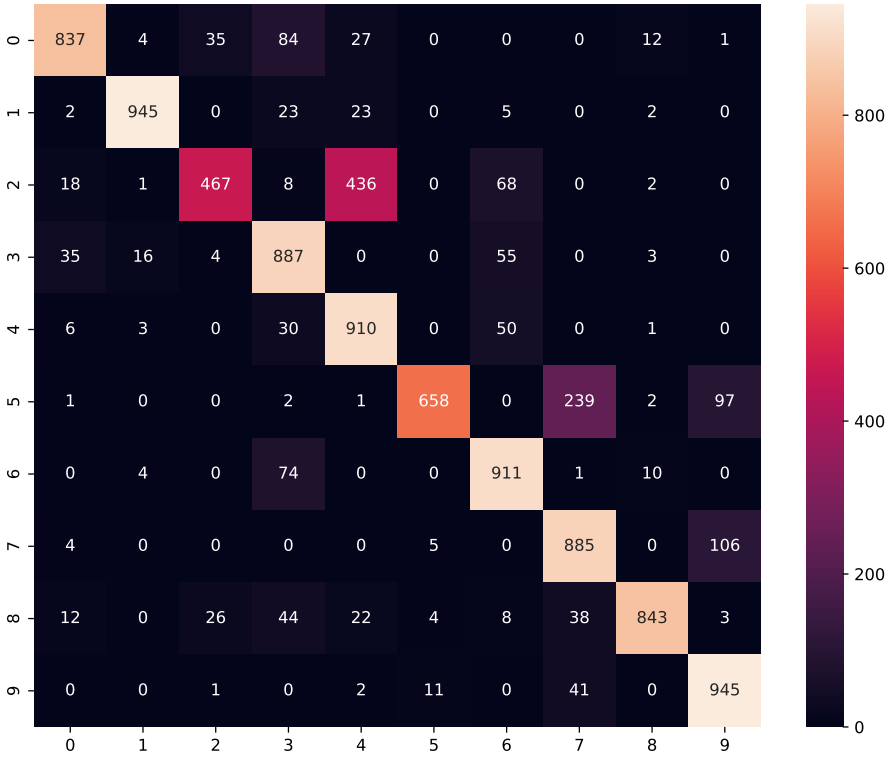


FIGURE 3.10: FMNIST Confusion Matrix

100 labels were incorrectly predicted. The blocks around the diagonals for the FMNIST and CIFAR-10 datasets indicate that more labels were incorrectly predicted, as their classification accuracies were lower. This figure provides a visualisation further supporting the classification accuracies as seen in Table 3.3.

3.4.3 Statistical Significance

To evaluate the statistical significance of the observed differences in accuracy scores among the different datasets (MNIST, FMNIST, and CIFAR-10), an analysis of variance (ANOVA) followed by Tukey's Honestly Significant Difference (HSD) post hoc test was performed.

The Analysis of Variance (ANOVA) test was conducted to assess whether there were statistically significant differences in the means of accuracy scores among the datasets. The null hypothesis (H_0) assumed that the means were equal across all datasets, while the alternative hypothesis (H_a) suggested that at least one dataset's

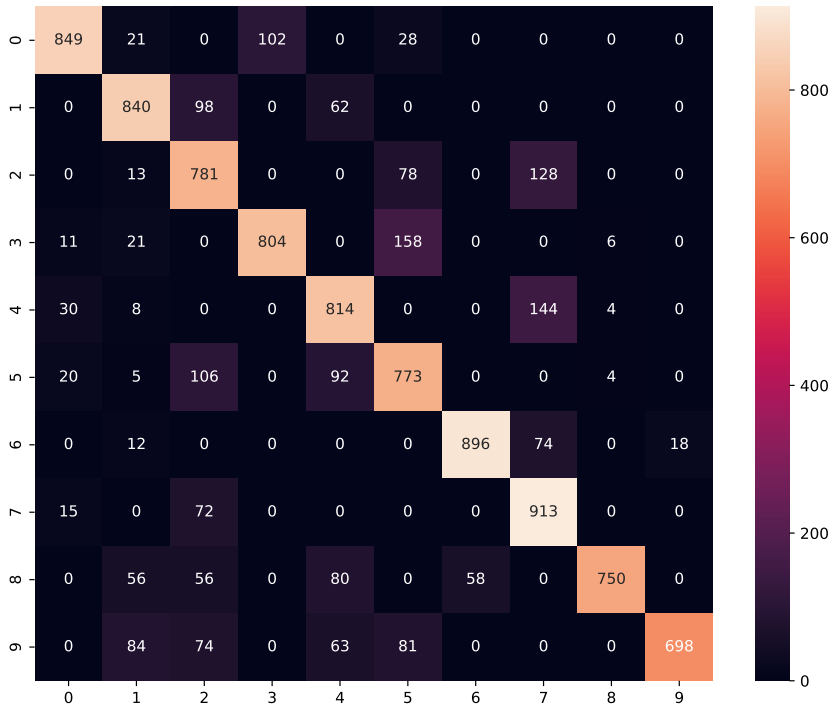


FIGURE 3.11: Confusion Matrix for the CIFAR-10 dataset.

mean accuracy score was significantly different. The obtained F-statistic was 47.85 with a corresponding p-value of 0.0001. Since the p-value is below the chosen significance level of 0.05, the null hypothesis was rejected, indicating that there are significant differences in accuracy scores among the datasets.

Following the significant result from the ANOVA, Tukey's HSD post hoc test was conducted to identify pairwise differences in accuracy scores. The results of the post hoc test indicated that the accuracy scores for all three datasets were significantly different from each other ($p < 0.05$). Specifically, the accuracy scores for MNIST were significantly higher compared to both FMNIST and CIFAR-10 ($p < 0.05$). Additionally, FMNIST's accuracy score was significantly higher compared to CIFAR-10 ($p < 0.05$).

These statistical analyses provide robust evidence that the observed differences in accuracy scores among the datasets are statistically significant, supporting the conclusion that the dataset characteristics have an impact on the performance of the proposed GAN-based classifier.

3.5 Classification Accuracy and Comparison

To ensure the validity of the obtained results and provide a comprehensive analysis, a comparison was performed with two state-of-the-art models, namely the SGAN and Triple GAN models [10, 11]. Table 3.5 displays the outcomes of this comparison. These results were taken directly from the original papers, and thus FMNIST is omitted as no results were provided for the SGAN and Triple GAN models on FMNIST. The results obtained in Table 3.5 for the presented model were an average of results taken over 5 separate runs of the model.

TABLE 3.5: Classifier scores for each dataset

	MNIST (n=100)	**CIFAR-10 (n=4000)**
Triple GAN	90.90%	91.52%
SGAN	99.11%	82.74%
Presented Model	99.32%	83.22%

As illustrated, the presented model beats the accuracy scores obtained for the MNIST and CIFAR-10 datasets in the SGAN model. The presented model's accuracy is 0.21% better than SGAN for MNIST and 0.48% better for CIFAR-10. The presented model's accuracy also triumphs over that of the MNIST dataset for Triple GAN with an accuracy of 8.42% higher. However, the Triple GAN accuracy for CIFAR-10 is significantly higher than the presented model by 5.3%.

The presented model performed better than these two, suggesting that utilising the decoder as the VAE enhances image generation and classification accuracy. The higher image quality produced during the early training epochs likely facilitated the discriminator's learning process, resulting in a better-performing classification component.

3.6 Overview of Baseline Comparisons

Two comparison baseline models were used to identify whether the model performed well. The baseline models are:

1. Regular GAN - a standard GAN model without the semi-supervised component that uses a generator. The architecture of the unsupervised discriminator

component remained the same as the presented model. The reason for this baseline model was to compare the quality of images at early epochs, as well as the loss functions.

2. Semi-supervised GAN model - this was the same as the model presented in this study except for a generator being used rather than a decoder from a VAE. The supervised and unsupervised discriminator architecture remained the same as the proposed model. This model was required for comparison with the classification of the CNN outputs.

Using the two baseline models, many comparisons were made. The first of these comparisons was for the quality of the generated images at early epochs. Figures 3.1, 3.2, and 3.3 indicate these differences. As seen in the images, the model presented in this study outperforms the baseline model by far, and the image quality is greatly improved for all three datasets. The images are more precise, and the shapes within the images can be determined.

A second comparison was made concerning the losses of the GAN. This was done to measure discriminator loss and to observe whether mode collapse can be avoided. As seen in Figures 3.4, 3.5 and 3.6, which show the losses of the discriminator and generator for each of the datasets, the presented model converges faster compared to regular GANs. The chance of mode collapse occurring, however, is decreased substantially as this convergence is reached, and the oscillations are much smaller for both the generator and the discriminator. This is further justified by the images in Figure 3.3. As illustrated in the image, for the regular GAN model, the images generated tend to be very similiar for each of the different epoch values, indicating that mode-collapse did indeed occur for this model. The difference is observed in the presented model, where generated samples differ by a large margin.

The baseline semi-supervised GAN was used to compare the output accuracies of generated images in the CNN. Images were generated using the baseline model, and then those images were used to train the CNN, and the accuracy was determined when testing on the original dataset. Table 3.4 illustrates the accuracies obtained by the CNN models and shows that the presented model outperformed this baseline model.

Table 3.4 also shows that the accuracy of the presented model is not too far off from the accuracy of the actual dataset, indicating that the generated images are of good quality as the CNN was able to learn from them.

3.7 Overall Results

The presented model produced high-quality images early in the GAN training process. The proposed model beat the regular GAN model for early epochs with significant improvements for all three datasets. This superiority of the model in this regard is confirmed by the high accuracies obtained from the CNN when trained on the generated images from the presented model compared to the regular GAN and semi-supervised GAN models. It was further confirmed by the high inception scores obtained for each dataset.

Compared to the baseline models, this model also proved that it has a lower chance of mode collapse occurring, with more minor oscillations in the discriminator and generator loss graphs. However, the presented model's discriminator converges faster than the regular GAN model.

The classification accuracy that the presented model produces is resoundingly high, with the comparison to state-of-the-art models emphasising this. This is further proven with confusion matrices.

The inception score and difficulty in observing the CIFAR-10 images suggest that the presented model works better for less complicated datasets or greyscale images. While the classification accuracy is still high for CIFAR-10, there is a significant drop in the classification accuracy of the generated images compared to those from the actual dataset, suggesting that the image quality could be improved.

Chapter 4

Conclusion

4.1 Summary

The use of Generative Adversarial Networks (GANs) has increased exponentially since its inception due to their many use-cases, with many variations being established to overcome some of the challenges that GANs face. This study aimed to improve upon semi-supervised GANs by generating higher-quality samples at early training epochs and avoiding mode collapse, i.e., fast convergence of the discriminator. Another aim was to provide an evaluation of the GAN outputs in an image classification task. This is to show that the improved semi-supervised GAN could be deployed in an application scenario where labelled data are scarce.

Although studies exist where these problems have been addressed, this study differed in that it aimed to improve the GAN for semi-supervised learning, specifically, using the decoder of a Variational Autoencoder (VAE) as the generator of the GAN and a stacked discriminator model. Three different datasets were tested; the MNIST, Fashion MNIST, and CIFAR-10.

The presented model generated high-quality images at the early epochs, with the images being better than those of a normal GAN. When the generated images were used to train a Convolutional Neural Network (CNN), the model presented higher accuracy scores for classification on the test dataset of the real datasets, beating a normal semi-supervised GAN model by 5.17% for MNIST, 4.72% for FMNIST and 9.56% for CIFAR-10. The presented model also achieved classification accuracy scores of 99.32% for MNIST, 92.78% for FMNIST and 83.22% for CIFAR-10.

Additionally, smaller oscillations in the discriminator and generator losses were achieved, leading to a lower chance of mode collapse.

4.2 Conclusion

Some of the aims of the study were achieved; namely, the quality of the images at early epochs had greatly improved, and the chance of mode collapse occurring during the training of the GAN was decreased. Significant progress was achieved with regard to the first aim, as the quality of images generated by the GAN at early epochs exhibited a marked improvement. This is a significant achievement, as early epochs of GAN training are known to be particularly challenging and often result in poor image quality. However, through the use of novel techniques and modifications to the GAN architecture, the study was able to demonstrate a notable improvement in image quality during these early epochs. The GAN's outputs were also classified well, with the classification accuracy for the MNIST dataset bettering some state-of-the-art models. However, the model presented in this study had its discriminator converging faster than that of regular GANs.

Despite the discriminator's faster convergence, the system was successful in responding to the research questions presented in Section 1.3.1.

4.3 Future Work

The presented model could still be improved further in terms of generating higher-quality images and better classification performance. While this study uses the decoder from a vanilla VAE model, more complex VAE models, such as a Conditional Variational Autoencoder (CVAE) or an InfoVAE can be considered in the proposed framework. CVAE enables the use of generated samples based on the attributes of the input data, while InfoVAE incorporates an information bottleneck into the model. InfoVAE encourages the model to learn more meaningful representations of the data [28] by limiting the amount of information that can flow through the bottleneck. It would be an interesting extension of this study to investigate whether using decoders from different VAE models improves the quality of generated images or helps in avoiding mode collapse.

Different GAN architectures could also be used, such as the architecture from a DCGAN or CGAN model. Using their architecture and adapting it to the stacked discriminator could allow for higher classification accuracies and possible slower convergence of the discriminator.

In addition, if higher capacity computing resources are available, the model's performance could be further tested and improved by utilising larger datasets. This would lead to a significant enhancement in the quality of generated samples and enable a more equitable comparison with newer models that employ more complex datasets.

Optimising more hyperparameters within the GAN and VAE models could also lead to substantial improvements in the model's output quality. By identifying the best possible hyperparameters, the model's performance can be optimised to generate the highest quality outputs. Overall, these potential improvements indicate the promising future of this proposed model and the potential for further advancements in the field.

Bibliography

[1] Martin Arjovsky and Léon Bottou. *Towards Principled Methods for Training Generative Adversarial Networks*. 2017. arXiv: 1701.04862 [stat.ML].

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. "Wasserstein generative adversarial networks". In: *International conference on machine learning*. PMLR. 2017, pp. 214–223.

[3] Antonia Creswell et al. "Generative Adversarial Networks: An Overview". In: *IEEE Signal Processing Magazine* 35.1 (2018), pp. 53–65. DOI: 10.1109/MSP.2017.2765202.

[4] Farzan Farnia and Asuman Ozdaglar. "Gans may have no nash equilibria". In: *arXiv preprint arXiv:2002.09124* (2020).

[5] Harshvardhan GM et al. "A comprehensive survey and analysis of generative models in machine learning". In: *Computer Science Review* 38 (July 2020), p. 100285. DOI: 10.1016/j.cosrev.2020.100285.

[6] Ian Goodfellow et al. "Generative adversarial nets". In: *Advances in neural information processing systems* 27 (2014). DOI: 10.1145/3422622.

[7] Tero Karras, Samuli Laine, and Timo Aila. "A style-based generator architecture for generative adversarial networks". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 4401–4410.

[8] Diederik P. Kingma and Max Welling. "An Introduction to Variational Autoencoders". In: *Foundations and Trends® in Machine Learning* 12.4 (2019), pp. 307–392. ISSN: 1935-8237. DOI: 10.1561/2200000056. URL: http://dx.doi.org/10.1561/2200000056.

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira et al. Vol. 25. Curran Associates, Inc., 2012. URL: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[10] Abhishek Kumar, Prasanna Sattigeri, and Tom Fletcher. "Semi-supervised learning with GANS: Manifold invariance with improved inference". In: *Advances in neural information processing systems* 30 (2017).

[11] Chongxuan Li et al. "Triple generative adversarial nets". In: *Advances in neural information processing systems* 30 (2017).

[12] Xudong Mao et al. "Multi-class Generative Adversarial Networks with the L2 Loss Function". In: (Nov. 2016).

[13] Mehdi Mirza and Simon Osindero. "Conditional generative adversarial nets". In: *arXiv preprint arXiv:1411.1784* (2014).

[14] Takeru Miyato et al. *Spectral Normalization for Generative Adversarial Networks*. 2018. DOI: 10.48550/ARXIV.1802.05957. URL: https://arxiv.org/abs/1802.05957.

[15] Augustus Odena. "Semi-supervised learning with generative adversarial networks". In: *arXiv preprint arXiv:1606.01583* (2016).

[16] Tom O'Malley et al. *KerasTuner*. https://github.com/keras-team/keras-tuner. 2019.

[17] Alec Radford, Luke Metz, and Soumith Chintala. "karras representation learning with deep convolutional generative adversarial networks". In: *arXiv preprint arXiv:1511.06434* (2015).

[18] Y Reddy, Viswanath Pulabaigari, and Eswara B. "Semi-supervised learning: a brief review". In: *International Journal of Engineering & Technology* 7 (Feb. 2018), p. 81. DOI: 10.14419/ijet.v7i1.8.9977.

[19] Shagan Sah. *Machine Learning: A Review of Learning Types*. July 2020. DOI: 10.20944/preprints202007.0230.v1.

[20] Tim Salimans et al. "Improved techniques for training GANS". In: *Advances in neural information processing systems* 29 (2016).

[21] Christian Szegedy et al. "Going deeper with convolutions". In: June 2015, pp. 1–9. DOI: 10.1109/CVPR.2015.7298594.

[22] Sherry Turkle. *Evocative Objects: Things We Think With*. Jan. 2007.

[23] Kunfeng Wang et al. "Generative adversarial networks: introduction and outlook". In: *IEEE/CAA Journal of Automatica Sinica* 4.4 (2017), pp. 588–598.

[24] Yuhuai Wu et al. *On the Quantitative Analysis of Decoder-Based Generative Models*. 2016. DOI: 10.48550/ARXIV.1611.04273. URL: https://arxiv.org/abs/1611.04273.

[25] Xianwen Yu et al. "VAEGAN: A Collaborative Filtering Framework based on Adversarial Variational Autoencoders." In: *IJCAI*. 2019, pp. 4206–4212. DOI: 10.24963/ijcai.2019/584.

[26] Yang Yu et al. "karras representation learning with deep convolutional neural network for remote sensing images". In: *Image and Graphics: 9th International Conference, ICIG 2017, Shanghai, China, September 13-15, 2017, Revised Selected Papers, Part II 9*. Springer. 2017, pp. 97–108.

[27] Wencai Zeng et al. "A comparison study: Support vector machines for binary classification in machine learning". In: *2011 4th International Conference on Biomedical Engineering and Informatics (BMEI)*. Vol. 3. 2011, pp. 1621–1625. DOI: 10.1109/BMEI.2011.6098517.

[28] Shengjia Zhao, Jiaming Song, and Stefano Ermon. *InfoVAE: Information Maximizing Variational Autoencoders*. 2017. DOI: 10.48550/ARXIV.1706.02262. URL: https://arxiv.org/abs/1706.02262.

[29] Jun-Yan Zhu et al. "Unpaired image-to-image translation using cycle-consistent adversarial networks". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2223–2232. DOI: 10.1109/ICCV.2017.244.