Conditional Generation with Variational Autoencoders and Generative Adversarial Networks on the MNIST Dataset

Kin Man Lam
15823898

Project for MATH7017 Probabilistic Graphical Models

Lecturer: Dr. Oliver Obst

Tutor: Stuart Fitzpatrick

School of Computer, Data and Mathematical Sciences,
Western Sydney University

Autumn 2023

Executive Summary

This report explores the implementation and comparative analysis of two popular generative models, the Conditional Variational Autoencoders (CVAEs) and the Conditional Generative Adversarial Networks (CGANs). Both models are trained and evaluated using the MNIST dataset with the aim to generate new and unseen data emulating the original data distribution. The report covers key areas such as model architecture, training behaviour, visual results of generated images and classification accuracy on the generated images. Observations made in this study show unique strengths and weaknesses of each model, thus providing insights into the nature and capabilities of these generative models. The study concludes with a discussion on potential improvements and future opportunities of research in this area.

Table of Contents

Executive Summary **i**

Table of Contents **ii**

List of Tables **iii**

List of Figures **iv**

1 Introduction **1**

2 Models and Dataset Description **1**

3 Training Details and Results **2**

3.1 **Hyperparameters Setting for Both Models** 2

3.2 **Training Details and Results on CVAE** 3

3.3 **Training Details and Results on CGAN** 6

4 Models Comparison **8**

4.1 **Models Architecture** 8

4.2 **Training Process** 9

4.3 **Output** 9

4.4 **Classification Accuracy** 9

4.5 **Versatility** 9

4.6 **Computation Time** 9

5 Discussion **10**

6 Potential Improvements **10**

7 Conclusions **11**

References **12**

List of Tables

1 Hyperparameter Choices for MNIST Training 2
2 Chosen Hyperparameters for MNIST Training 3

List of Figures

1 2D Latent Space Visualisation of the CVAE at Final Epoch 4
2 Average Loss per Epoch for CVAE Training 4
3 Generated Images from CVAE 5
4 Visualisation of the Latent Space for the CGAN 6
5 Generator and Discriminator Loss for CGAN Training 7
6 Generated Images from CGAN 7

1 Introduction

Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs) are two popular deep learning based generative models over the recent years. With huge amount of data, well-designed networks architectures and smart training techniques, both models have shown an incredible ability to produce highly realistic pieces of content of various kind, such as images, texts and sounds. Conditional Variational Autoencoders (CVAEs) and Conditional Generative Adversarial Networks (CGANs) are extensions of VAEs and GANs respectively, which allow for the conditioning of the generative process on extra information.

The aim of this report is to study the performances of CVAE and CGAN on the MNIST dataset, a popular choice for various machine learning tasks involving image processing. It will explore key areas including training details, result analysis, model comparisons, challenges faced, insightful observations and potential improvements. Finally, the strengths and weaknesses of both models will also be shown throughout the report.

2 Models and Dataset Description

Given the purpose of this report, background information such as description for both models and the dataset will be briefly discussed in this part.

VAEs are generative models that probabilistically encode high-dimensional data into a lower-dimensional latent space using a Gaussian distribution. There are two parts in VAEs which are an encoder and a decoder. The encoder outputs a mean and a standard deviation defining the distribution. On the other hand, the decoder reconstructs the original data from a random sample drawn from the latent distribution. The training process for VAEs targets to minimise the difference between input and output and ensures the latent distribution closely follows a standard Gaussian distribution [Rocca, 2019]. CVAEs are extensions of VAEs which allow for the conditioning of the encoding and decoding process on extra information. This enables the generation of data that meets certain conditions. Therefore, CVAEs are very useful for tasks that require the generation of specific types of data [Dykeman, 2016].

GANs are another generative models which have gained popularity in recent years. Like VAEs, the network architecture for GANs consists two parts which are a generator and a

discriminator. Both networks are trained together. The generator produces new examples and tries to fool the discriminator while the discriminator classifies whether these examples are real from the domain or fake generated by the generator. Thus, the two networks compete in a zero-sum game [Brownlee, 2019a]. CGANs are extensions of GANs which provide both the generator and discriminator with extra information such as class labels. This information influences the data generation process and allows the model to generate data with specific attributes [Dobilas, 2022].

Lastly, the MNIST dataset plays a crucial role in this report. The dataset consists of 60,000 training and 10,000 testing small square grayscale images of handwritten single digits between 0 and 9 with each image measuring 28 x 28 pixels. The task is to classify a given image of a handwritten digit into one of 10 classes representing integer values from 0 to 9 inclusively [LeCun, 2023]. Although the MNIST dataset is relatively small and simple, it is still considered as a useful platform for studying the complex details of machine learning models such as CVAE and CGAN and for understanding how different parameters affect their performance.

3 Training Details and Results

3.1 Hyperparameters Setting for Both Models

Table 1: Hyperparameter Choices for MNIST Training

Common Hyperparameters	
Input Size	784
Number of Classes	10
Number of Samples	10
Hyperparameter	**Choices**
Latent Dimension	32, 64, 100
Hidden Size	128, 256, 512
Epochs	30, 50, 100
Batch Size	64, 100, 128
Learning Rate	0.01, 0.001, 0.0005

The hyperparameters shown in Table 1 were all tested in the training process for both

models. These values are common choices for the MNIST dataset which is relatively small and simple as stated previously. Nevertheless, the hyperparameters shown in Table 2 were chosen in the final training as they maintain a balance between performance and computation time during training for both models thereby ensuring a fair comparison in later parts of this report.

Table 2: Chosen Hyperparameters for MNIST Training

Hyperparameter	Value
Latent Dimension	64
Hidden Size	512
Epochs	50
Batch Size	100
Learning Rate	0.001

3.2 Training Details and Results on CVAE

The architecture for the CVAE model is built on an Multi-Layer Perceptron (MLP) framework which is a type of Artificial Neural Network (ANN). Both the encoder and the decoder consist of two hidden layers, which is generally viewed as appropriate size for the MNIST dataset, with containing 512 neurons each. ReLU activation functions are used in the hidden layers while a Sigmoid function is used in the output layer. Moreover, a component called reparameterisation follows the encoder stage. This critical step allows for backpropagation through random nodes by generating a latent vector [Dykeman, 2016].

For training the CVAE on the MNIST dataset, the combination of Binary Cross Entropy (BCE) and Kullback-Leibler Divergence (KLD) is used as the loss function with Adam serving as the optimiser. The condition which is the class label of the digit to be generated is integrated into the CVAE by concatenating the one-hot encoded label with the input image data for the encoder and with the latent vector for the decoder [Dykeman, 2016]. It is also worth mentioning that different types of regularisation such as Batch Normalisation, Dropout, L1 and L2 regularisation were tested during the training process. However, these methods did not significantly improve the performance for the model so no regularisation was not used in the final training.

Figure 1 shows a 2D visualisation of the learned latent space from the CVAE at the

final epoch of training [Tiu, 2020]. The result shows latent space for each class is sparse and overlapped with each other. This is completely different from VAE where the latent space for each class is neatly clustered.

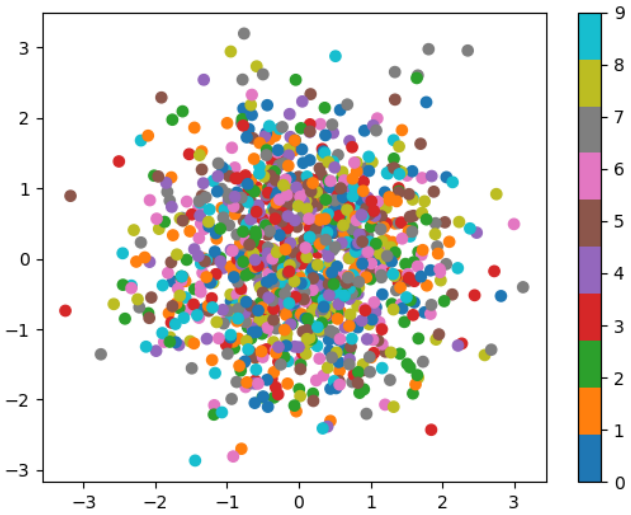

Figure 1: 2D Latent Space Visualisation of the CVAE at Final Epoch

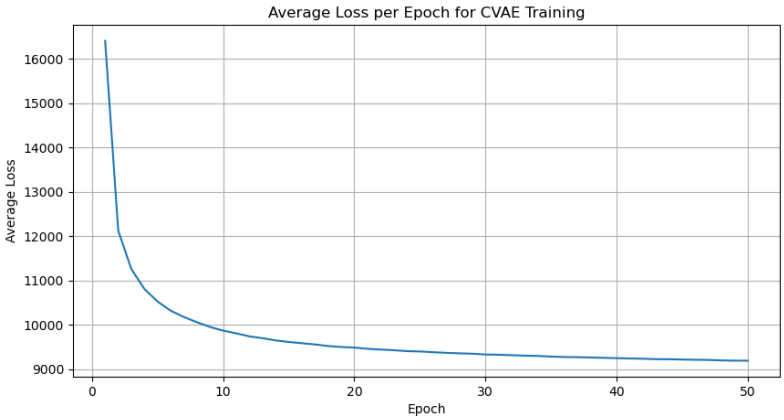

Figure 2: Average Loss per Epoch for CVAE Training

Figure 2 shows the average loss per epoch for the training. There was a sharp decline in the first 10 epochs. After that, the loss continued to decrease gradually until the end of the training.

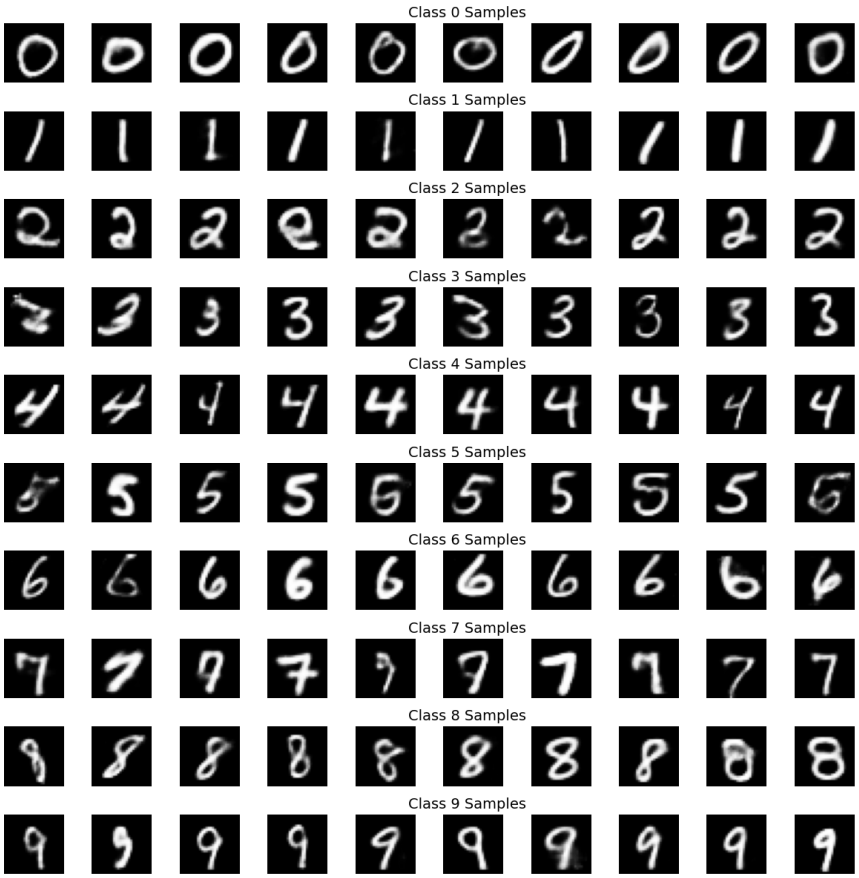

Figure 3: Generated Images from CVAE

The newly generated image samples from the CVAE model are displayed in Figure 3. There are 100 images in total with 10 for each class label. Subjectively, these images are impressive and very close to the original ones in visual assessment.

Lastly, a separate classifier built on an MLP was trained to evaluate the quality of the latent representations learned by the CVAE model. The architecture of this classifier is simple with only one hidden layer containing 512 neurons. A ReLU activation function is applied to the outputs from this hidden layer [Goodfellow et al., 2016]. All other hyperparameters remain the same as in the original model. The classifier achieved an accuracy of 99.29% when predicting the class labels of unseen test data which is very impressive.

3.3 Training Details and Results on CGAN

The architecture of the CGAN model is also built on an MLP which is similar to the CVAE model. Both the generator and the discriminator consist of two hidden layers with each containing 512 neurons. For the generator, a LeakyReLU activation function with a negative slope coefficient of 0.2 is used in the hidden layers followed by batch normalisation. The output layer uses a Tahn activation function which transforms the output to be within the range of -1 to 1. Conversely, a LeakyReLU activation function with a negative slope coefficient of 0.2 is used followed by a dropout operation with a rate of 0.3 after each hidden layer for the discriminator. Its output layer uses a Sigmoid activation function which transforms the output to be within the range of 0 to 1 [Dobilas, 2022].

As with the CVAE, BCE is used as loss function while Adam are used as optimiser for both the generator and the discriminator in the CGAN model. The class label condition which expressed as a one-hot encoded label is concatenated with the input data. This allows the model to generate or discriminate data based on specific classes [Dobilas, 2022].

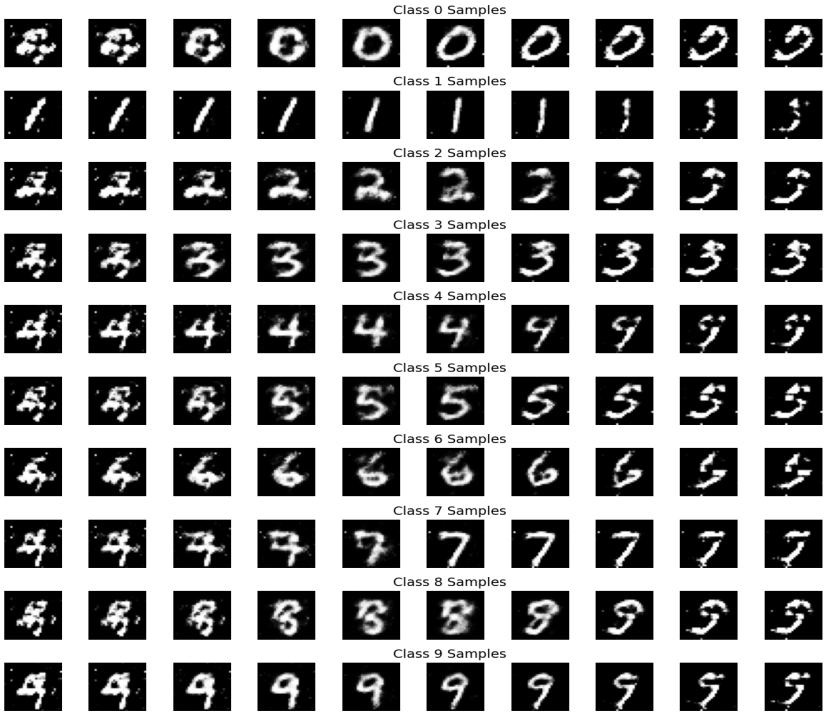

Figure 4: Visualisation of the Latent Space for the CGAN

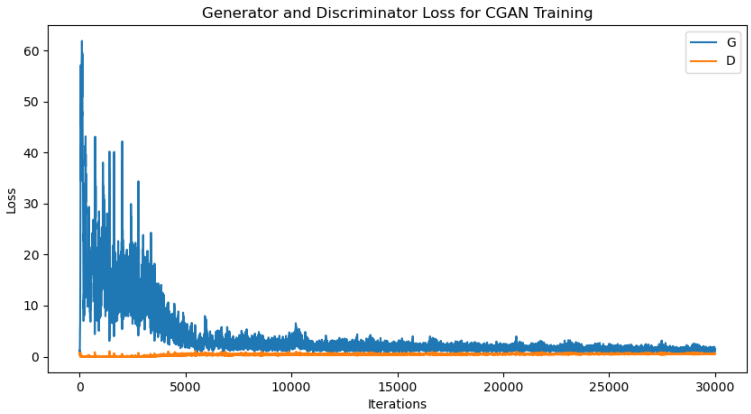

Figure 5: Generator and Discriminator Loss for CGAN Training

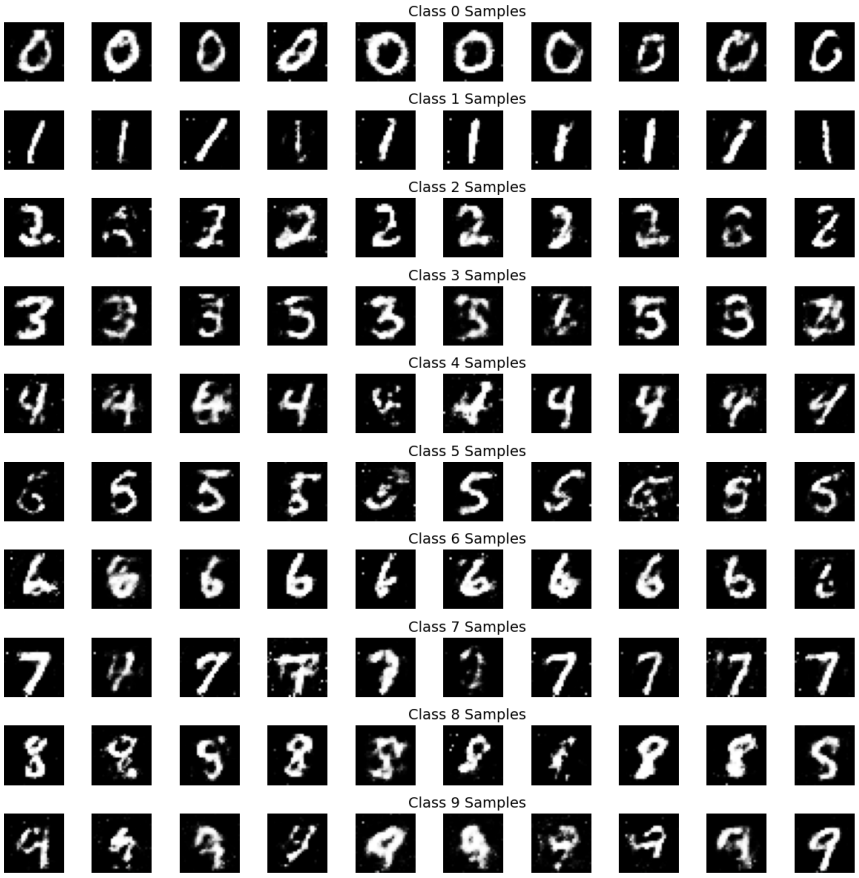

Figure 6: Generated Images from CGAN

Figure 4 shows a concise visualisation of learned latent space for the CGAN model [Tiu, 2020]. It generates a series of noise vectors and corresponding class labels which are used by the generator model to produce images. The produced images represent a smooth transition across the latent space for each class, thus demonstrating the capacity of the model for class-conditional image generation.

Figure 5 displays the generator and discriminator losses during the training for the CGAN model. The generator loss starts high and fluctuates in the early stages, then decreases to a lower level and stabilises after around 5,000 iterations. Meanwhile, the discriminator loss remains consistently low throughout the training process.

Similar to the last part, the newly generated image samples from the CGAN model are displayed in Figure 6. Although the image quality is mediocre, with a small portion of them being unrecognisable, the overall visual assessment considers them acceptable.

In the final analysis, the same classifier trained earlier was also applied to the CGAN model. The results show a accuracy rate of 99.27% in predicting the class labels of unseen test data, which is an excellent outcome.

4 Models Comparison

Although there are similar in structure and approach for the CVAE and CGAN models, they showed different traits during training process and in their final results.

4.1 Models Architecture

Both models are built on an MLP framework and share similar architectures with two hidden layers consisting of 512 neurons each. The design provides a considerable degree of flexibility and enables both models to capture complex structures in the data. However, their operational mechanisms are different. The CVAE uses an encoding-decoding mechanism that maps the input data to a specific latent representation before reconstructing it. Conversely, the CGAN uses an adversarial network consisting of a generator and a discriminator that progressively improve to produce and evaluate synthetic data. These operational mechanisms lead to contrasting training behaviors and outcomes even though their foundational architectures are similar.

4.2 Training Process

During the training process, the CVAE model showed a decrease in loss over the epochs. Such result suggests that the model was effectively learning to generate digits from the MNIST dataset. It produced a sparser and more overlapped latent space compared to the CGAN model as is apparent in the 2D visualisations. On the other hand, the CGAN model experienced fluctuating generator loss with the discriminator loss remaining relatively low. This indicates the adversarial nature of its training.

4.3 Output

In terms of output, the CVAE model performed well in generating new and unseen data with impressive image quality. In contrast, the CGAN model also produced images of acceptable quality but few of them were hard to identify. The result indicates some variability in the ability of the model to accurately generate images.

4.4 Classification Accuracy

The classification accuracy of both models are impressive. The CVAE model showed a very good 99.29% accuracy while it is 99.27% accuracy for the CGAN model. The result suggests that despite their different training behaviors and output variations, both models were successful in learning the underlying patterns of the MNIST dataset.

4.5 Versatility

Both models are well known for their versatility and can perform well even when handling more complex datasets such as colour images, 3D data or time series data. However, the network architecture may need additional layers or the integration of Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) to adapt to these new datasets. Moreover, it is important to ensure the dataset is not too small as this can lead to overfitting problems in generative models.

4.6 Computation Time

Lastly, the training time for both models differed significantly. Training the CVAE model took seven minutes while the CGAN model required twelve minutes. Therefore,

the training time for the CVAE model was particularly less time-consuming than the CGAN model.

In conclusion, both models have their strengths and weaknesses but they both show potential as effective tools for generative tasks using the MNIST dataset.

5 Discussion

The tasks shown in this report posed considerable challenges. There were several difficulties and interesting observations encountered throughout the report. Foremost among these was the implementation of CNNs into both models. As mentioned previously, the MNIST dataset is relatively small and simple so MLPs are generally viewed as sufficient for its training. However, since the dataset is related to image recognition and it would be interesting to assess the performance of the models with the integration of CNNs particularly for the CGAN[Brownlee, 2019b]. Regrettably, multiple attempts at these implementations were unsuccessful due to the complexity of CNNs and a lack of expertise.

In addition, the training of the CGAN is notoriously tricky due to its adversarial nature. The discriminator in the model outperformed the generator in the early stage of training which resulted in the generator struggling to improve. This situation is common for GAN training and is known as 'mode collapse' [Brownlee, 2019a].

Moreover, the outputs of the generator evolved over the progress of training is also worth to watch. In the early stages, the generated images were quite noisy and the MNIST digits can hardly be recognised. However, the outputs started to resemble genuine digits which underlined the capacity of the model to learn the underlying distribution of the data as training progressed.

6 Potential Improvements

There are several areas could improve the performance and efficiency for both models in future work. Firstly, a more sophisticated model architecture could be used. Complex model architectures could potentially improve the ability of the model to capture the complex underlying distributions of the data. As previously mentioned, the application

of CNNs could potentially improve performance especially considering that the input data are images [Brownlee, 2019b].

Additionally, extensive hyperparameter tuning could potentially improve the performance of the models. It is worth to test various values for all hyperparameters to achieve the best performance if time allows.

Furthermore, the CVAE and CGAN models could be adapted for other tasks such as semi-supervised learning or anomaly detection. These adaptations could open up new avenues for using these models beyond their traditional roles in generative tasks.

Finally, the application of more complex dataset such as Fashion-MNIST, CIFAR-10 or ImageNet could lead to better and more sophisticated models. Moreover, certain performance evaluation methods such as Inception Score (IS) and Frechet Inception Distance (FID) are generally recommended for use with colour images. These methods, while considered overkill for the MNIST dataset, can be effectively applied to more complex datasets providing a more comprehensive evaluation.

The potential improvements discussed above could significantly improve the performance of the CVAE and CGAN models and broaden the understanding of their capabilities.

7 Conclusions

This report has provided a comprehensive comparison between the CVAE and the CGAN model. Both models were implemented using the same base architecture while their performance was evaluated using the MNIST dataset. Key areas such as differences in training behaviour, quality of generated images and computation time were observed closely. The study found that the CVAE model generated impressive image quality but also displayed a sparser latent space. In contrast, the CGAN model was slower in training and produced mediocre image quality, but it provided a more structured latent space. Both models display great potential for generative tasks using the MNIST dataset, each with their own strengths and weaknesses. Future work could explore modifications to the model architectures or training strategies to further optimise their performance. Additionally, testing the adaptability of the models using different datasets and tasks would be valuable for expanding the potential application scope of these techniques.

References

Jason Brownlee. A gentle introduction to generative adversarial networks (gans), 2019a. URL https://machinelearningmastery.com/what-are-generative-adversarial-networks-gans/. Last accessed 07 June 2023.

Jason Brownlee. How to develop a cnn for mnist handwritten digit classification, 2019b. URL https://machinelearningmastery.com/how-to-develop-a-convolutional-neural-network-from-scratch-for-mnist-handwritten-digit-classification/. Last accessed 10 June 2023.

Saul Dobilas. cgan: Conditional generative adversarial network — how to gain control over gan outputs, 2022. URL https://towardsdatascience.com/cgan-conditional-generative-adversarial-network-how-to-gain-control-over-gan-outputs-b30620bd0cc8. Last accessed 07 June 2023.

Isaac Dykeman. Conditional variational autoencoders, 2016. URL https://ijdykeman.github.io/ml/2016/12/21/cvae.html. Last accessed 06 June 2023.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

Yann LeCun. Mnist handwritten digit database, 2023. URL http://yann.lecun.com/exdb/mnist/. Last accessed 06 June 2023.

Joseph Rocca. Understanding variational autoencoders (vaes), 2019. URL https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73. Last accessed 06 June 2023.

Ekin Tiu. Understanding latent space in machine learning, 2020. URL https://towardsdatascience.com/understanding-latent-space-in-machine-learning-de5a7c687d8d. Last accessed 09 June 2023.