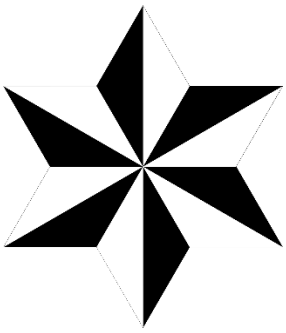
Generative art using DCGANs



Eloise Derham

Student number: 19809305

University of Brighton

This report is submitted for the degree of

Computer Science with Artificial Intelligence

Supervisor: Karina Rodriguez

Computing division 2022

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other University. This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration, except where specifically indicated in the text. This dissertation contains less than 65,000 words including appendices, bibliography, footnotes, tables and equations and has less than 150 figures.

Eloise Derham

2022

Abstract

This report studies the ability of DCGANS to create unique pieces of artwork relating to the training data fed into them. It investigates the functionality of DCGANS and exploring multiple ways used to optimize their performance using qualitative and quantitative evaluation metrics including the Fretchet inception distance, to evaluate and improve the output.

Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Project motivation

Generative art is a growing field of study that has implications for machine learning. This new creative territory has driven my expansion, learning and innovative thinking in this specific research. Raising questions around the creative nature of machine learning and how this can be translated into art. By researching the use of GANs (Generative Adversarial Networks) within a creative field of study we can also begin to explore other possibilities for GANs in machine learning.

I use the goal of creating unique pieces of generative art to allow an in-depth exploration of the DCGAN (Deep convolutional Generative Adversarial Network) architecture. Using in depth research I explore the multiple ways of optimizing GANs, experimenting with different parameters and testing the effects of one-sided label smoothing. In order to evaluate and improve my model I use recommended evaluation metrics based on in depth studies surrounding GANs.

In addition, I explore the social, ethical, and legal issues surrounding the use of my algorithm and image generation. Although there are few issues for my project specifically, I discuss my thought process on the possibility of future issues should my algorithm or images be used for non-educational purposes.

In conclusion, I reflect on the agile methodology and its strengths and weaknesses. I continue to consider the successful use of GANs within my project and the possibilities for future improvements. As a result, creating a broader conversation about the wider field of generative art.

1.2 Project objectives

The objectives of my project are to understand the structure of GANs and how to use them to build a successful machine learning model. Using a DCGAN I should create code that can be trained using my own set of data and learn to recreate the images it is fed.

Once I have learnt to build the DCGAN architecture the next objective is to optimize the output. I will evaluate each output based on reliable metrics and by the end of the project I should have created an optimized algorithm with better results for the final images both with improved definition and contrast than were originally produced.

1.2.1 Deliverables

The deliverables for this project are as follows:

1. DCGAN algorithm code
2. Trained model with optimizations
3. Project report
4. Video presentation

1.3 Report outline

The report below takes the following structure.

- An introduction to the project and deliverables.
- Information on my project methodology and planning.
- An in depth look into the literature and research used to inform the project.
- A description of the code and experimental results gathered from the project.
- A brief description of legal and ethical issues surrounding the project.
- Evaluation and conclusion of the finished project.

Chapter 2 Methodology

2.1 Methodology

Using an efficient planning methodology is important for a successful project, it helps manage time and workload as well as setting clear goals and objectives during a project's lifespan.

My chosen approach for planning and project development was the agile methodology. Agile is the best option for my project due to its incremental nature. It leaves a lot of room for requirements changes as well as having a high focus on user feedback, in my case user feedback was feedback received from my supervisors. The reason agile is so flexible is because it uses sprints, these are blocks of time in which allocated tasks must be finished, after each sprint new tasks are set, and changes are made to the plan for the next sprint. The average time for sprints is 2 weeks.

In practice I found using 2-week sprints quite difficult due to the nature of my project it was hard to have concrete plans for 2 weeks, my project timeline ebbed and flowed depending on the research I had done and changed drastically from week to week. For this reason, I decided to use weekly sprints to allow for the many changes to be added to the project plan as I went along.

2.2 Development tools and techniques

Using development tools aids the use of a planning methodology by helping you to keep up to date visually with plans and project changes.

I used Trello over other project management services such as Monday.com or Click Up because I had used it in other university projects and therefore was very familiar with the layout and how to interact with. Furthermore, it offers many important features that help with implementing agile, for example the ability to easily add lists and cards that can be edited to your needs with attachments notes and checklists. It also supports integration with third-party apps making it a very flexible software.

I used my Trello board to keep track of my work following the layout for a classic agile project, with sprint tasks, tasks in action and tasks that have been completed. I also added a questions list to keep track of questions I wanted to ask my supervisor. Using Trello really helped me to keep track of my project and visualise the timeline of tasks. Below is a screenshot of my Trello board and its layout. Further screenshots can be found in appendix A.

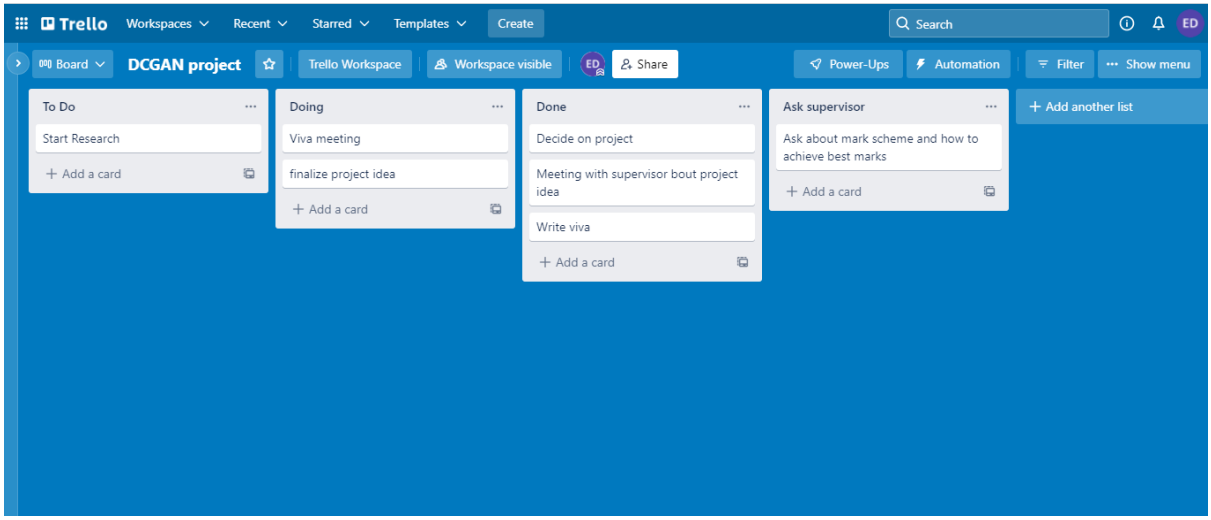


Figure 1 Trello board screenshot

For development of my project, I used Google Colab. Google Colab is cloud based Jupyter notebooks, allowing the user all the perks of a Jupyter notebook plus the added bonus of

developing on the cloud. Google Colab is perfect for AI projects that includes computationally expensive training functions as Colab uses its own GPUs (Graphics processing unit) and TPUs (Tensor processing unit) making training a lot faster. A free user is allowed access to a GPU for 12 hours and then runtime disconnects, this is satisfactory for most small AI projects. I opted for the Google Colab pro which allowed me access to faster GPUs as well as longer runtimes.

What's more Colab saves all projects to your google drive making them easily accessible anywhere as well as safely backed up, this also meant I was able to access my dataset from my drive meaning less memory taken up on my personal computer.

2.3 Risk analysis

Table 1 Project risk analysis

Risk	**Severity 1-3**	**Likelihood 1-5**	**Impact**	**Mitigation**
Not being able to complete work due to illness.	2	1	There is a risk of getting ill or getting covid but the impact of this is quite low as it shouldn't last longer than a week.	Factor in enough time for issues such as illness in my work plan. Also make sure to look after myself to prevent illness.

Not being able to complete work due to pressure from other modules	2	4	This is a very possible risk that could have a large impact on the project as if I am behind on other work it could lead to me not having the time to complete this work and stick to schedule.	Plan workload efficiently and ensure I do enough work for each module every week. Planning is very important to mitigate this risk.
Computer breaks	3	1	If my computer broke this would have a massive impact as I would have no way to complete the coursework and I cannot afford a new computer, I would probably also lose my work.	Make sure to maintain my computer and look after it to decrease the risk of it breaking. Also back up all project folders. If my computer breaks, I can use university computers if my work is backed up
No access to internet	3	2	As I am using google Colab I will need internet to work on my code and train my model, having good reliable internet is therefore very important.	This is not a big issue as if my internet goes down there are many other places to access internet such as University or a caffe.

Loss of work due to hardware/software issues	3	1	This would be bad for my progress as I would have to start my project from the beginning.	Make sure to regularly back up my work.
Running out of time due to unforeseen circumstances	2	3	If I ran out of time and my deliverables weren't finished for the deadline this would affect my overall mark and be very bad for the project.	Ask for an extension if circumstances are serious and keep track of app progress and edit work plan where necessary as the project progresses.

Chapter 3 Literature research

3.1 Research

Throughout my research 4 main studies made an appearance within most articles and papers I read, they lay the foundations for research within GANs and helped me to have an in depth understanding of the subject area.

The first paper by Ian Goodfellow “NIPS 2016 Tutorial: Generative Adversarial Networks” (2016) provides an overview of the inner workings of GANs and the need for further research within the topic of study. It focuses on the use of GANs within other machine learning models such semi-supervised learning where many labels within the data are missing as “Generative models can be trained with missing data and can provide predictions on inputs that are missing data”. In addition, the paper describes the architecture of GANs in depth.

This paper was followed by “Improved Techniques for Training GANs” which outlines a new improved GAN architecture for DCGANS which I have followed in my code. Within the paper they propose “a set of constraints on the architectural topology of Convolutional GANs that make them stable to train in most settings.” (Radford, A., Metz, L. and Chintala, S. (2016)).

The third paper “Improved Techniques for Training GANs” (Salimans, 2016) investigates the main ways used to optimize the outcome of GANs, and the fourth paper “Pros and cons of GAN evaluation measures. Computer Vision and Image Understanding” (2019) discusses the evaluation options for GANs.

3.1.1 GANs

GANs (Generative adversarial networks) use deep learning methods such as convolutional networks to achieve generative modelling. “GANs are an exciting and rapidly changing field, delivering on the promise of generative models in their ability to generate realistic examples across a range of problem domains” (https://www.facebook.com/MachineLearningMastery, 2019)

GANs use two models that are trained side by side, a generator and a discriminator. The generator aims to fool the discriminator into classifying “fake” images as real. The generator takes a fixed length random vector as input and uses it to generate new data. Once the generator is trained it is hoped that “points in this multidimensional vector space will correspond to points in the problem domain.” (https://www.facebook.com/MachineLearningMastery, 2019)

Once trained, the generator model is used to generate new “fake” samples to be fed into the discriminator.

The discriminator takes input from both the real and generated images sets and works to assign them with the correct “using traditional supervised learning techniques, dividing inputs into two classes (real or fake).” (Openai, n.d.)

Below figure 1 shows the GAN framework in a diagram from Ian Goodfellows paper.

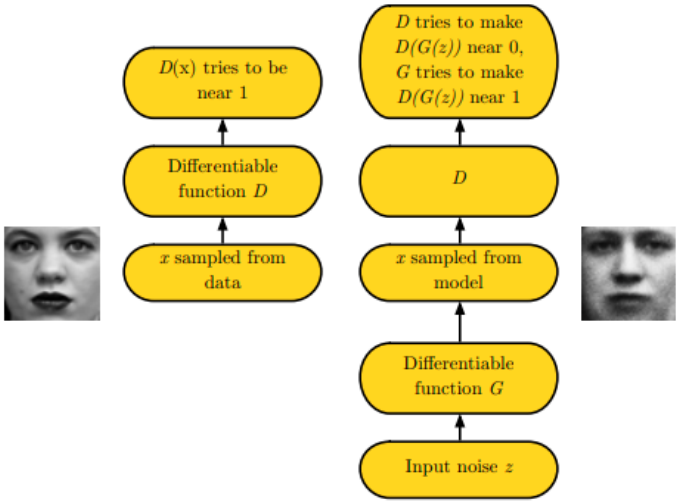


Figure 2 GAN framework (Ian Goodfellow, 2016)

3.1.2 DCGANs

DCGAN refers to type of GAN architecture which uses convolutional and convolutional transpose layers in the generator and discriminator as first proposed by Radford in 2016.

The guidelines below allow for an architecture "that result(s) in stable training across a range of datasets and allow(s) for training higher resolution and deeper generative models"

Architecture guidelines for stable Deep Convolutional GANs

- Replace any pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).
- Use batchnorm in both the generator and the discriminator.
- Remove fully connected hidden layers for deeper architectures.
- Use ReLU activation in generator for all layers except for the output, which uses Tanh.
- Use LeakyReLU activation in the discriminator for all layers.

Figure 3 DCGAN architecture guidelines

Included below is the general architecture created by Radford to be used as the generator when using DCGANs.

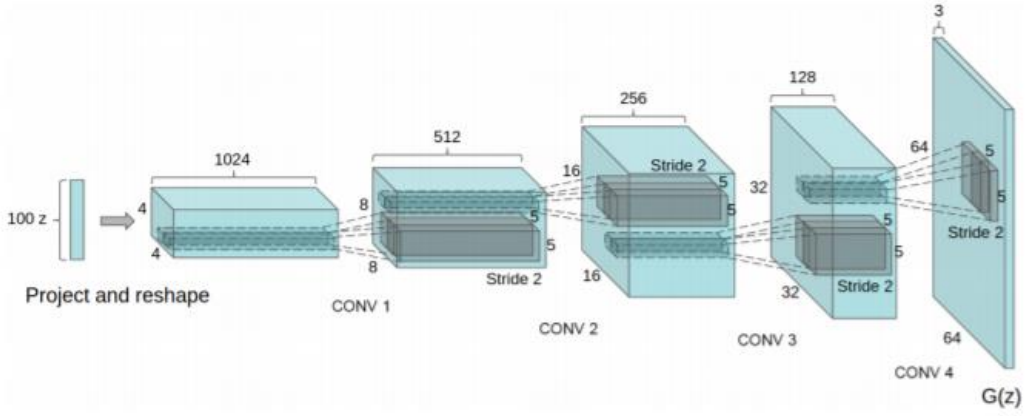


Figure 4 DCGAN generator architecture

3.1.3 Optimization

"Improved Techniques for Training GANs" (Salimans, 2016) describes multiple ways to optimize the output of a GAN. Within the paper they introduce "techniques that are heuristically motivated to encourage convergence". Some of these include feature matching, minibatch discrimination and virtual batch normalization. Later in this report I go into more detail on the techniques I tried within my DCGAN implementation.

But what does it mean to optimize a GAN?

"Training GANs consists in finding a Nash equilibrium to a two-player non-cooperative game. Each player wishes to minimize its own cost function, J (D) (θ (D) , θ (G)) for the discriminator and J (G) (θ (D) , θ (G)) for the generator. A Nash equilibrium is a point (θ (D) , θ (G)) such that J (D) is at a minimum with respect to θ (D) and J (G) is at a minimum with respect to θ (G) ."

However, comparing the outcome of different models is difficult with GANs because "Generative adversarial networks lack an objective function", below I talk about evaluation techniques that can be used.

3.1.4 GAN evaluation

Evaluation of GANs is an ongoing topic of study, due to the nature of the game like training, it is hard to find evaluation metrics which accurately evaluate the output. Many times, human judgement is used to assess the quality of generated samples.

However, for obvious reasons this is not a secure evaluation metric as people's opinions can differ making it a highly subjective form of evaluation. Furthermore, the reviewers need knowledge of what is realistic and what is not for the target domain, and this can also be a subjective measure. Only small batches can be reviewed for each model by manual inspection making it an expensive process.

For this reason, I also researched quantitative measures for GAN evaluation, the two main techniques used are the Inception Score (IS) and the Frechet Inception Distance (FID).

Both techniques involve using the pre-trained neural net model used for image classification called the Inception v3 model. The IS measures the probability that a given image belongs to its predicted class, a higher inception score suggests a better-quality generated image.

FID is like IS as it also uses the Inception v3 model however it is said to mimic human judgement much better. FID compares images from the real and generated batch to assess their distance within the gaussian distribution. The lower the FID the closer in similarity the generated images are to the real ones.

In Appendix 1 of "Pros and cons of GAN evaluation measures. Computer Vision and Image Understanding" (Borji, A , 2019) an in-depth study into FID is shown, including a comparison of both IS and FID. In the images shown below (just a snapshot of the study) you can clearly see why FID is a better evaluation metric than IS and how its score is directly impacted by the quality of the image.

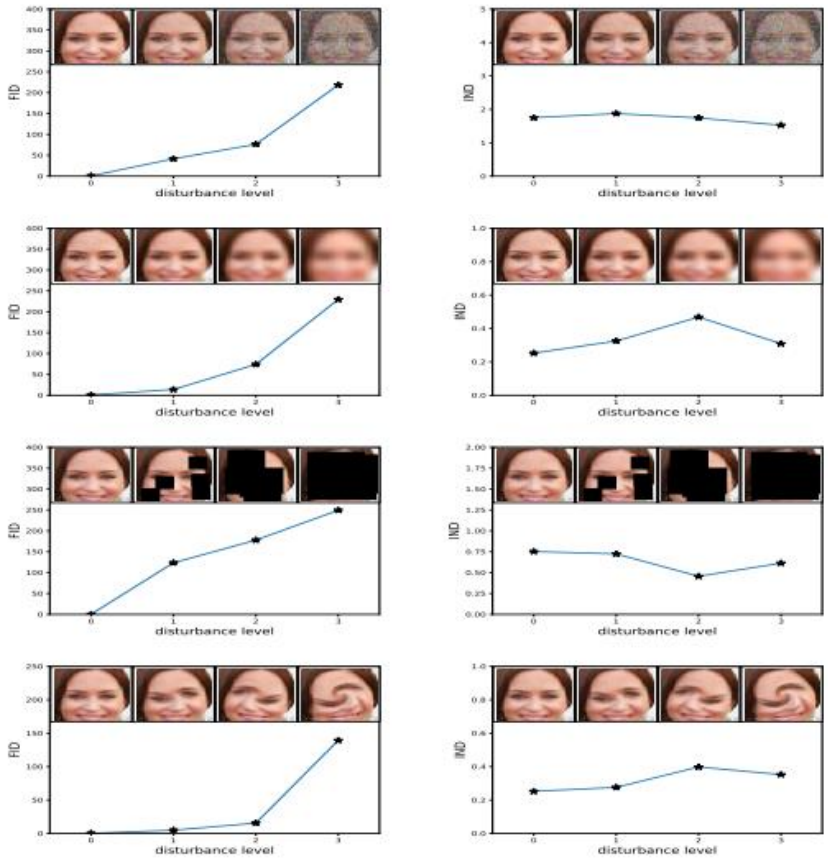


Figure 5: On the right we see the FID and on the left we see the IS calculated with increasing level of disturbance (Heusel et al., n.d.)

3.1.1 GANs and generative art

The inspiration for this project came from the fast-growing area of research and artistic creation that is generative art. In broad terms generative art is “work that has been produced by the activation of a set of rules and where the artist lets a computer system take over at least some of the decision-making” (Digital Creativity, 2022). There are so many ways to create generative art with the earliest examples not including a computer at all, for example Mozart’s dice roll game “Musikalisches Würfelspiel” where players randomly generated music using pre composed pieces and dice. However, computer generative art uses code as a

medium to generate unique creative outputs with the first ever computer art exhibition being in 1965 titled “Generative Computergraphik” showcasing the work of Georg Nees.

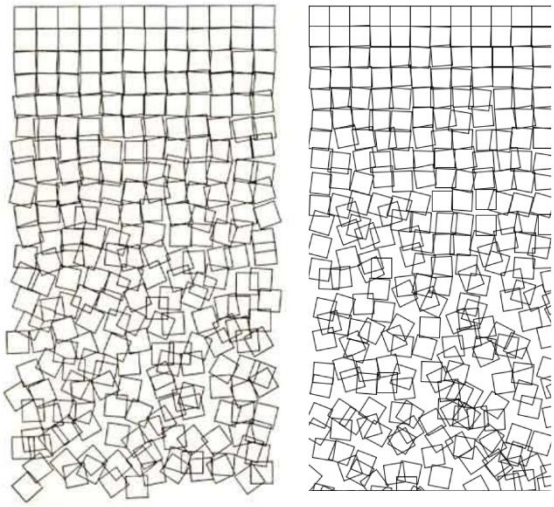


Figure 6 Generative artwork by Georg Nees

From then on, the field has grown immensely and generative art is now used for everything from music to poetry. One of the ways that generative art is being put into practice is by using GANs, multiple “GAN artists” already exist. The photo below for example shows a piece titled “AI Generated Nude Portrait #1” by an artist named Robbie Barrat. Robbie barat used a dataset of 19th century portraits to train his GAN and although the results are described as “grotesque” (Droitcour, 2021) it is clear that they imitate paintings.

Figure 7 “AI Generated Nude Portrait #1” by Robbie Barrat

It might not be the future of art, but it is definitely a growing medium continuing to be explored. This is the future of an artistic movement with massive possibilities and huge areas yet to be discovered.

In addition to emerging GAN artists there are also many new softwares being released that allow users to generate their own unique images with the help of GANs. Some examples include deep dream generator, a website that allows users to choose an image and a style to create a new generated image. Another example is a mobile app called dream which lets users input words or a sentence which is then used with a GAN to generate a piece of art.

Figure 8 (left) image generated by deep dream style transfer website (right) Image generated by Dream app with the prompt "pretty flower and bird" and psychedelic style

Chapter 4 Experimental results

4.1 Datasets

I tested my algorithm with many datasets but decided to use the cityscape dataset from Wiki-Art for final testing and optimization. I began by using a wiki art dataset on abstract art however this made it very hard to evaluate the output from the GAN as abstract art is very subjective, with cityscapes however I knew what I was looking for and there were a more finite set of evaluative features. Furthermore, cityscapes contained enough images to produce good results after training, I experimented with a dataset that only had 400 images and quickly realised this would not be enough to gather good results.

The cityscape dataset contained 6600 files and was 2.98GB. I downloaded it from https://www.kaggle.com/datasets/ipythonx/wikiart-gangogh-creating-art-gan?select=cityscape

Below is a small batch of images from the cityscape dataset.

Figure 9 collection of cityscape paintings from database

4.2 Code structure

The structure and code for my GAN was from a tutorial by PyTorch on coding DCGANs (Pytorch.org, 2014) the architecture follows that set out by Ian Goodfellow (Goodfellow, 2016) and Radford. I have changed the dataset and added optimization techniques to the code to try and improve it.

4.2.1 Data and initialization

The code starts by installing and transforming the dataset. The batch size is defined along with important parameters such as number of epochs, the data is then transformed into 64 by 64 images. In order to use the data loader, I had to create an image folder class within the datasets folder with information about the data that allowed for the transformations to be made.

Following this the weights for the generator are initialized using batch normalization as suggested in Ian goodfellows paper.

4.2.2 Generator and Discriminator

The generator is defined by “a series of strided two-dimensional convolutional transpose layers, each paired with a 2d batch norm layer and a Relu activation. The output of the generator is fed through a Tanh function to return it to the input data range of [-1,1]” (Pytorch.org, 2014) This follows the structure of DCGANS from Radfords paper.

Similar to the Generator, the Discriminator contains convolutional 2d layers, although it uses LeakyRelu layers instead of Relu and its final layer uses a Sigmoid function. This also follows the architecture set out by Radford.

Both the Discriminator and Generator use an Adam optimizer as this is what has been suggested in both Ian goodfellow and Radfords papers.

```
Discriminator(
  (main): Sequential(
    (0): Conv2d(3, 64, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (1): LeakyReLU(negative_slope=0.2, inplace=True)
    (2): Conv2d(64, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (3): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (4): LeakyReLU(negative_slope=0.2, inplace=True)
    (5): Conv2d(128, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (6): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (7): LeakyReLU(negative_slope=0.2, inplace=True)
    (8): Conv2d(256, 512, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (9): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (10): LeakyReLU(negative_slope=0.2, inplace=True)
    (11): Conv2d(512, 1, kernel_size=(4, 4), stride=(1, 1), bias=False)
    (12): Sigmoid()
  )
)
```

```
Generator(
  (main): Sequential(
    (0): ConvTranspose2d(100, 512, kernel_size=(4, 4), stride=(1, 1), bias=False)
    (1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): ReLU(inplace=True)
    (3): ConvTranspose2d(512, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (4): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (5): ReLU(inplace=True)
    (6): ConvTranspose2d(256, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (7): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (8): ReLU(inplace=True)
    (9): ConvTranspose2d(128, 64, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (10): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (11): ReLU(inplace=True)
    (12): ConvTranspose2d(64, 3, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1), bias=False)
    (13): Tanh()
  )
)
```

Figure 10 Discriminator and Generator architecture (screenshots from code)

4.2.3 Training

Before defining the training method, the function for calculating the Fretchet distance is defined, I got this code from "Compute FID scores with PyTorch*"* (mseitzer, 2021) an online GitHub repository. The official Fretchet distance code was done in TensorFlow, the code I used is made so that the Fretchet distance can be ustilised with PyTorch.

The training functions works in two parts, one trains the discriminator and other trains the generator. Essentially the discriminator is trained on the real batch and fake batches and its overall loss is the loss from both batches. The generator then generates images and calculates its loss based on the output from passing the fake batch through the discriminator. Both discriminator and generator are then updated using their optimizers.

At the beginning and end of each epoch training stats are printed onto the screen, this helps to keep track of training as well as help discover any issues whilst training is underway. Below is an explanation of the stats from the PyTorch tutorial.

Table 2 Displays information about training statistics from PyTorch tutorial (Pytorch.org, 2014)

Loss_D - discriminator loss calculated as the sum of losses for the all real and all fake batches (log(D(x)) + log(1 - D(G(z)))log(D(x))+log(1−D(G(z)))).
Loss_G - generator loss calculated as log(D(G(z)))log(D(G(z)))
D(x) - the average output (across the batch) of the discriminator for the all real batch. This should start close to 1 then theoretically converge to 0.5 when G gets better. Think about why this is.
D(G(z)) - average discriminator outputs for the all fake batch. The first number is before D is updated and the second number is after D is updated. These numbers should start near 0 and converge to 0.5 as G gets better. Think about why this is.

After the model has finished training images from the final generator are shown below the code as well as an animation to show the progress of the generator throughout training. Lastly graphs showing the Fretchet distance, generator loss and discriminator loss are printed for evaluation purposes.

4.3 Optimization Techniques

To optimize my algorithm, I started by analysing the difference in outcome between the number of epochs as well as the batch size.

I found that 100-200 epochs were long enough to reach the best possible outcome of the GANs, if I left it training longer than that the model often went into mode collapse and the images did not get any better in quality. Mode collapse is a common issue with GANs and it is when the generator learns that a certain output fools the discriminator regularly and will therefore begin to generate only that output. As stated by Google Developers

"Each iteration of generator over-optimizes for a particular discriminator, and the discriminator never manages to learn its way out of the trap. As a result, the generators rotate through a small set of output types."

However, I did find some techniques helped limit the likelihood of mode collapse, these included decreasing the batch size from 128 to 64 and using label smoothing.

4.3.1 Label smoothing

Our GAN generator can become overconfident and begin to produce features it knows will fool the discriminator; this is what happens with mode collapse. To avoid this, we can make the discriminator less sensitive by changing the label for real images from 1 to 0.9. You can also experiment with smoothing the label for generated images by make it 0.1 instead of zero.

I attempted one-sided label smoothing where only the label for real images is adjusted as mentioned in Improved Techniques for Training GANs.

"To encourage the discriminator to estimate soft probabilities rather than to extrapolate to extremely confident classification, we can use a technique called one-sided label smoothing" (Salimans et al., 2016).

4.4 Testing

For my project testing consisted of evaluating the output of the GAN to see if optimization techniques used improved the quality of the images. I evaluated the output based on 3 different metrics.

First, I used human evaluation, I judged the output of the images based on the cityscape data I was familiar with as well as seeing whether I could detect mode collapse by checking if the images within the generated batch had similar features or if lots of them looked alike.

In addition, at the beginning and end of all epochs training stats were logged and printed. They showed the Fretchet distance and loss of the generator and discriminator, which I also used to evaluate the performance of my GAN.

At the end of training, I also created graphs showing the evolutions of the Fretchet distance and losses to compare the lifecycle of my GAN with different optimizations.

4.5 Results

The images below show results from training using different parameters.

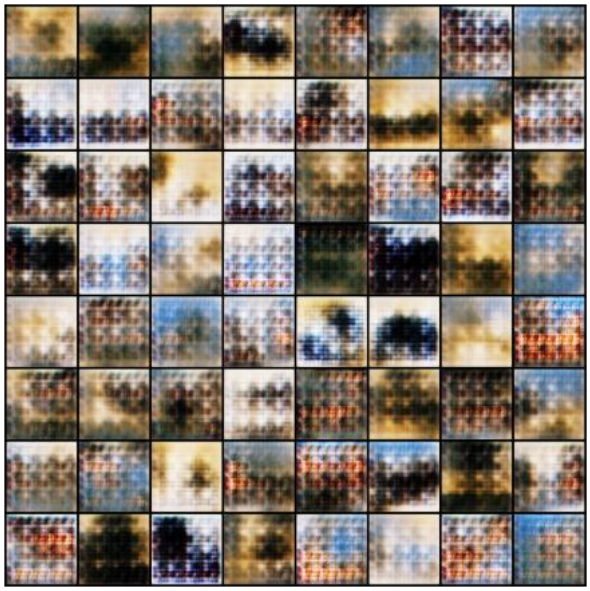


Figure 11 Training with 20 epochs

Figure 12 Training with 300 epochs

Figure 13 Training with 150 epochs and batch size of 64

Figure 14 Training with 150 epochs, batch size 64 and label smoothing

At first, I experimented with the number of epochs used. It is clear that 20 epochs are not enough to produce a quality output. Although it is evident that training for 300 epochs produces higher quality images than 20 epochs, there is little definition within features and there are signs of mode collapse as some features can be identified in more than one image.

Although it could be suggested that training for less epochs would generate worse results, in the images generated after 150 epochs it becomes apparent that this is not the case. The images generated are as good if not better in quality and show a wider variety of features and colours. This is also due to the smaller batch sized used.

Finally, I found that implementing one sided label smoothing very clearly improves the output of the generated images, visually we can see more defined features as well as a larger array of unique images compared to those created using no optimization techniques. I can see structures I would define as buildings as well as clouds and sky like features. I can confidently say that the optimization techniques used improved the overall image generation of my model.

Unlike with the visual evaluation, if we look at the graphs below, we can see that the improvement of the Fretchet distance is relatively uniform between all instances except the training done with 20 epochs. Training with no optimization techniques managed to achieve a lower score overall. As mentioned above it is difficult to properly assess GAN performance and although the Fretchet distance is a method widely used sometimes its scores do not match up to human evaluation.

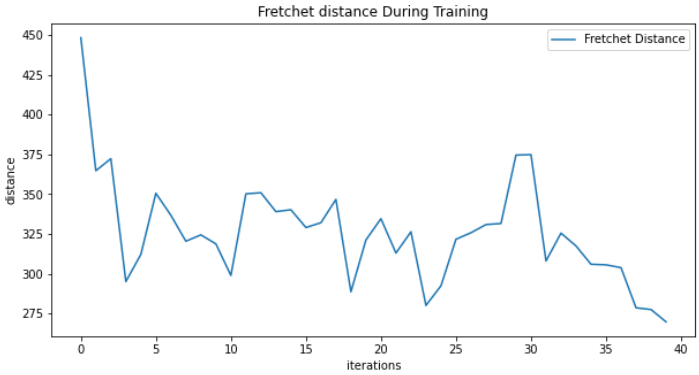


Figure 15 Fretchet distance for training with 20 epochs

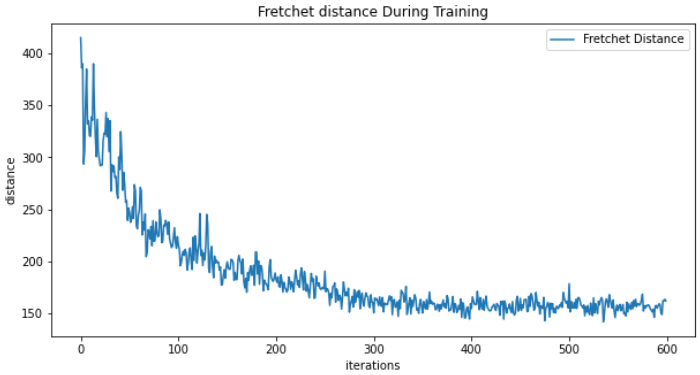


Figure 16 Fretchet distance graph for 150 epochs and batch size 64

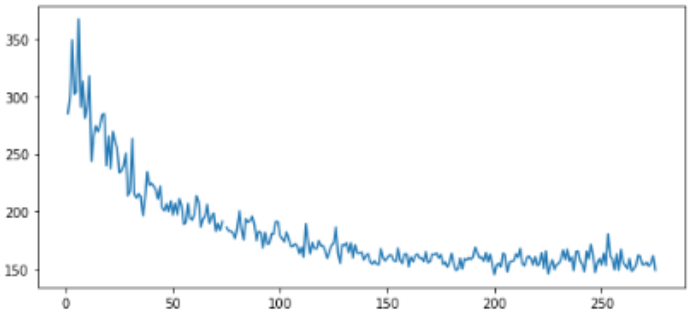


Figure 17 Fretchet distance graph for Training with 150 epochs and batch size of 64

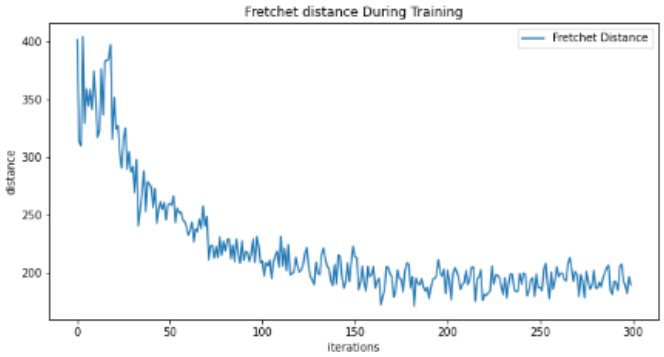


Figure 18 Fretchet distance graph training with 150 epochs, batch size 64 and label smoothing

From looking at the graphs above in relation to the results I garnered through human evaluation, I believe that human judgement is still the best way to evaluate GANs and that it will be difficult to find an evaluation metric that will be able to measure generated images successfully. However, using the Fretchet distance and losses during training was very useful to track training progress although they don't give an in depth evaluation of the generated images they do work to understand if your generator is improving.

Chapter 5 Professional issues

As this software is not built to be used by external users nor will it be tested by other people there is no social or ethical issues raised. The dataset used is from a reliable source and raises no legal issues as it is sourced from Kaggle, a free to use database discovery website. Lastly the output from the algorithm will not be posted to use anywhere and is purely for educational purposes, leading to no further legal issues.

However, that is not to say that there are no issues surrounding this topic. I have thought about possible issues that could arise should the images generated be displayed or sold. The main problem area is ownership of the images generated, is it legal or ethical to say that the images created are that of the coder who made the algorithm? It could be argued that as the AI has been trained on other artists paintings that the generated images are not in fact original and some form of accreditation should be given to the artists' work used in training.

As put by Margaret A. Boden & Ernest A. Edmonds in "what is generative art"

"Does the artist's choice of fitness function suffice to give him/her the authorial credit for whatever artwork emerges after many generations of random (i.e. undirected) change?"

Chapter 6 Conclusion

To complete my report, I will give an in-depth evaluation of the project sections as well as a personal reflection of the project as a whole. I will finally discuss my future plans leading on from this project and how the process and what I have learnt will help me to continue my work within this area of study.

6.1 Project summary

By the end of my project, I have created an algorithm that uses cityscape paintings to create new pieces of unique cityscape artwork. It uses a DCGAN architecture with a generator and discriminator. I have used multiple optimization techniques and evaluated the outcome of each one.

To complete my project, I performed a lot of research around GANs and began by experimenting with the Pytorch DCGAN architecture. Then using my research, I optimized this architecture and added further evaluation metrics.

From completing this project I have discovered that after a certain point, more epochs do not necessarily mean a better outcome, therefore it is important to analyse your model as it is training to see when the model reaches its optimum output. Furthermore, I have found that smaller batch sizing and one-sided label smoothing help to create more unique images with clearer features.

6.2 Evaluation

6.2.1 Methodology evaluation

Using the Agile methodology allowed for weekly changes within my project and gave me the flexibility needed to adapt to the constant research being done throughout my project. However, reflecting on the project planning I do believe I would have had an easier time planning each sprint if I had made a solid set of deliverables before starting development on my code. Without a clear set of deliverables, it made it hard to know what I was working towards and therefore what I had left to do and plan for.

Although the Agile methodology allowed for regular change it didn't set me up with a concrete plan for the whole project. This led to me getting a bit lost in the middle of my project as I didn't have a clear vision of what the final outcome would look like.

In retrospect I would be more clear on the final outcome of my project from the beginning and understand what is needed to be done to fulfil this criteria. I believe if I had been working of a piece of software then it would have been more straight forward to define a list of deliverables from the beginning. I understand that due to the nature of my project it was challenging to have structured plans, but I believe that I now have a better understanding of how to plan for a machine learning based project that revolves around research and testing.

6.2.2 Implementation evaluation

Overall, I was satisfied with my implementation of the DCGAN and pleased with the outcome it provided. I have used multiple optimization techniques including label smoothing but wish I had had time to use more, such as feature matching. During implementation I attempted to use feature matching but did not have the time to figure out how to make it work, for this reason I decided instead to focus on other optimization techniques.

I successfully used the Colab workspace to implement and train my model and believe this worked well for the scale of my project.

6.2.3 Evaluation of the final product

My final product is the code used to train my DCGAN model. My code can successfully be used to train the GAN on any dataset and has been optimized to try and ensure the best results from training. I believe a clear improvement of images generated can be seen from optimization techniques used and therefore I think the final product is a success.

However, I do believe there are ways in which I could improve the final product with more time and research, these mainly include testing different more advanced optimization techniques and potentially using the trained generator outside of my code to generate individual images for personal use.

6.3 Personal reflection

Overall, I am satisfied with my final project outcome. Throughout the project I was worried that I would not be able to finish in time and faced a lot of issues due to not being able to find the information needed through research. As GANs are still a relatively new machine learning technique it was hard to find information pertinent to my project, meaning that required a lot of my time researching, rather than coding. Each stage of coding was followed by research so that I could complete the next stage, it was impossible to research for the whole project before beginning. Furthermore, it was hard to troubleshoot problems as once again there was little information of errors within stack overflow, this led to a lot of time spent debugging code when an issue arose.

However due to all the research I had to undertake I feel like I have gained much from this project. Not only to I feel more proficient in Python and Pytorch, I now also have a strong understanding of GANs and its architecture. I believe this base understanding can now aid me well in future development and experimentation with other GAN types.

Another aspect of the project that took a long time was training, each time I wanted to train my GAN using new parameters it would take at least a couple of hours. Google Colab was helpful as it offered GPU and TPU services to train faster however there were also many

frustrating downfalls from using Colab. Colab only allows for 12-hour training periods and when your web page is not active it ends your runtime environment, losing all progress from training, this was a massive problem for me. My house Wi-Fi is very temperamental and every time it would disconnect my training would be halted. This made training time consuming and frustrating, on reflection it would be better to train your algorithm locally, however with only a laptop to use this wasn't an option for me.

Overall, the project has equipped me with knowledge about GANs that I would not have otherwise learnt. I now understand what to research and where to find important information should I wish to extend my project or begin work on a new GAN. I have become better acquainted with project management and how I could improve and organise things differently when doing a project of this scale again.

6.4 Future work

Completing this project has opened many doors for me to create generative art using GANs. From my research I have not only learnt thoroughly about the architecture of GANs and how they work to achieve their results, but I have also come across other GAN architectures that are recommended for creating images. One architecture I would like to experiment with is the super GAN as this allows the user to create high quality images. I think the logical next step of this project would be to be able to use the images created for pieces of artwork either for NFTs or printed onto canvas, for this reason high quality images are imperative. I now feel as though I have the knowledge to further my research within this field and continue to create computer generated work through a number of different mediums.

As the field of generative art begins to grow so will the uncertainty around its means of generation. I believe there are two roads it could take: that of the creative computer or that off the generative artist. As written in "What is generative art?" (Digital Creativity, 2022) "art involves the expression and communication of human experience, so that if we did decide that it is the computer which is generating the 'artwork', then it cannot be an artwork after all" However within the same essay the question is raised of what can be considered the real artwork the images created or "the program which generates them?". The infinite possibilities

within generative art leads to a large obscurity in definitions of the practice and I believe this is why it is such an interesting field of study that will only expand in the coming years.

By the end of this project, I am left with more questions about AI and creative machine learning than I started with, it is clear to me that there are so many possibilities for GANs and generative art as a new medium of expression.

References

Borji, A. (2019). Pros and cons of GAN evaluation measures. *Computer Vision and Image Understanding*, [online] 179, pp.41–65. Available at: https://www.sciencedirect.com/science/article/pii/S1077314218304272.

Digital Creativity. (2022). *What is generative art?* [online] Available at: https://www.tandfonline.com/doi/full/10.1080/14626260902867915?scroll=top&needAccess=true [Accessed 4 May 2022].

Droitcour, B. (2021). *GANs and NFTs*. [online] ARTnews.com. Available at: https://www.artnews.com/list/art-in-america/features/gans-and-nfts-1234594335/mario-klingemann-planned-obsolescence-hic-et-nunc/ [Accessed 30 Apr. 2022].

Google Developers. (2020). *Common Problems | Generative Adversarial Networks | Google Developers*. [online] Available at: https://developers.google.com/machine-learning/gan/problems [Accessed 26 Apr. 2022].

Hui, J. (2018). *GAN — How to measure GAN performance? - Jonathan Hui - Medium*. [online] Medium. Available at: https://jonathan-hui.medium.com/gan-how-to-measure-gan-performance-64b988c47732

Openai, I. (n.d.). *NIPS 2016 Tutorial: Generative Adversarial Networks*. [online] Available at: https://arxiv.org/pdf/1701.00160.pdf.

M.Innat (2020). *Wiki-Art : Visual Art Encyclopedia*. [online] Kaggle.com. Available at: https://www.kaggle.com/datasets/ipythonx/wikiart-gangogh-creating-art-gan?select=cityscape [Accessed 25 Apr. 2022].

mseitzer (2021). *mseitzer/pytorch-fid: Compute FID scores with PyTorch.* [online] GitHub. Available at: https://github.com/mseitzer/pytorch-fid [Accessed 26 Apr. 2022].

Pytorch.org. (2014). *DCGAN Tutorial — PyTorch Tutorials 1.11.0+cu102 documentation*. [online] Available at: https://pytorch.org/tutorials/beginner/dcgan_faces_tutorial.html [Accessed 26 Apr. 2022]

Radford, A., Metz, L. and Chintala, S. " *UNSUPERVISED REPRESENTATION LEARNING WITH DEEP CONVOLUTIONAL GENERATIVE ADVERSARIAL NETWORKS*" [online] Available at: https://arxiv.org/pdf/1511.06434.pdf.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A. and Chen, X. (2016). *Improved Techniques for Training GANs*. [online] Available at: https://arxiv.org/pdf/1606.03498.pdf.

Bibliography

Google.com. (2014). *Google Colab*. [online] Available at: https://research.google.com/colaboratory/faq.html [Accessed 23 Apr. 2022].

Hui, J. (2018). *GAN — Ways to improve GAN performance - Towards Data Science*. [online] Medium. Available at: https://towardsdatascience.com/gan-ways-to-improve-gan-performance-acf37f9f59b [Accessed 26 Apr. 2022].

Hui, J. (2018). *GAN — How to measure GAN performance? - Jonathan Hui - Medium*. [online] Medium. Available at: https://jonathan-hui.medium.com/gan-how-to-measure-gan-performance-64b988c47732 [Accessed 6 May 2022].

Johnson, D. (2021). *What is Trello? Here's what you need to know*. [online] Business Insider. Available at: https://www.businessinsider.com/what-is-trello?r=US&IR=T [Accessed 23 Apr. 2022].

monday.com Blog. (2018). *Agile Planning: Step-by-Step Guide | monday.com Blog*. [online] Available at: https://monday.com/blog/project-management/agile-planning/ [Accessed 23 Apr. 2022].

Mittal, T. (2019). *Tips On Training Your GANs Faster and Achieve Better Results*. [online] Medium. Available at: https://medium.com/intel-student-ambassadors/tips-on-training-your-gans-faster-and-achieve-better-results-9200354acaa5 [Accessed 6 May 2022].

Pytorch.org. (2022). *Writing Custom Datasets, DataLoaders and Transforms — PyTorch Tutorials 1.11.0+cu102 documentation*. [online] Available at: https://pytorch.org/tutorials/beginner/data_loading_tutorial.html [Accessed 6 May 2022].

soumith (2020). *soumith/ganhacks: starter from 'How to Train a GAN?' at NIPS2016*. [online] GitHub. Available at: https://github.com/soumith/ganhacks [Accessed 6 May 2022].

Appendix A Link to code

GitHub repository link:

https://github.com/EloiseDerham/DCGAN-art

Google Colab link:

https://colab.research.google.com/drive/1Z_qIVY5FBSRaT0iCobLmyK02L0Ha4PGV?usp=sharing

(Recommend following the google colab link instead of viewing code on GitHub)

Link to download dataset:

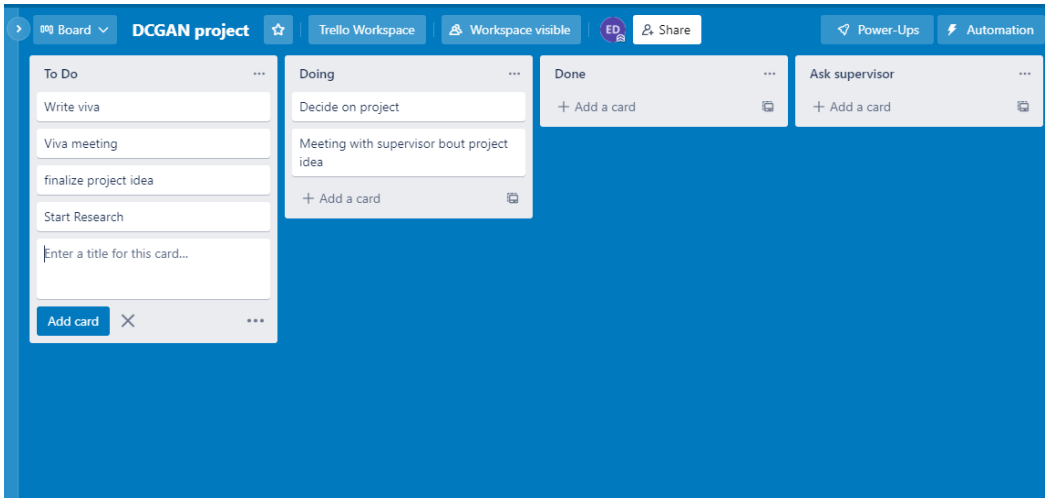
Download from Google Drive:
https://drive.google.com/file/d/1EaShtyBGlZEFleti2hWuW7h1luJkF7Ld/view?usp=sharing

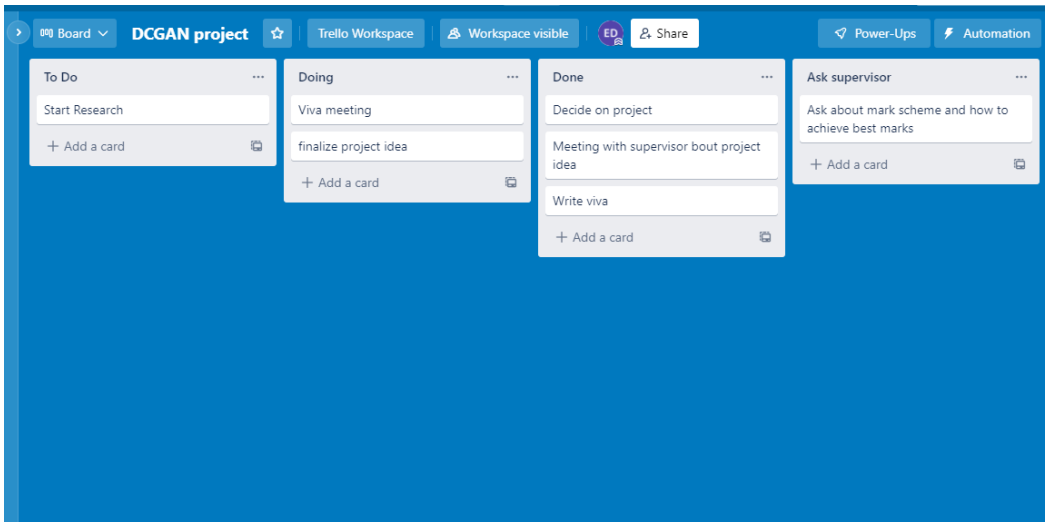
Notes for running code:

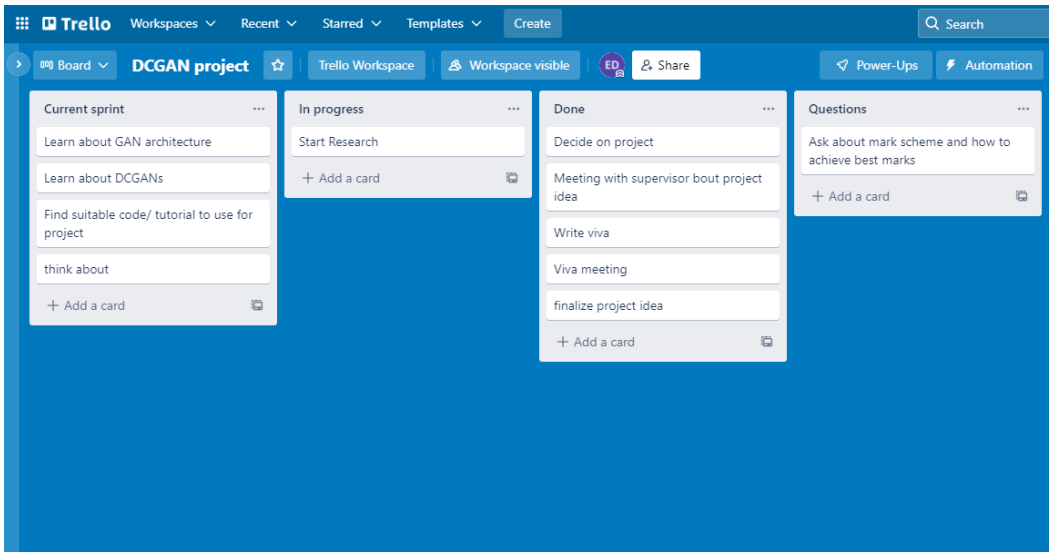
Must download the data to your Google Drive as a Zip folder in order to run the code.

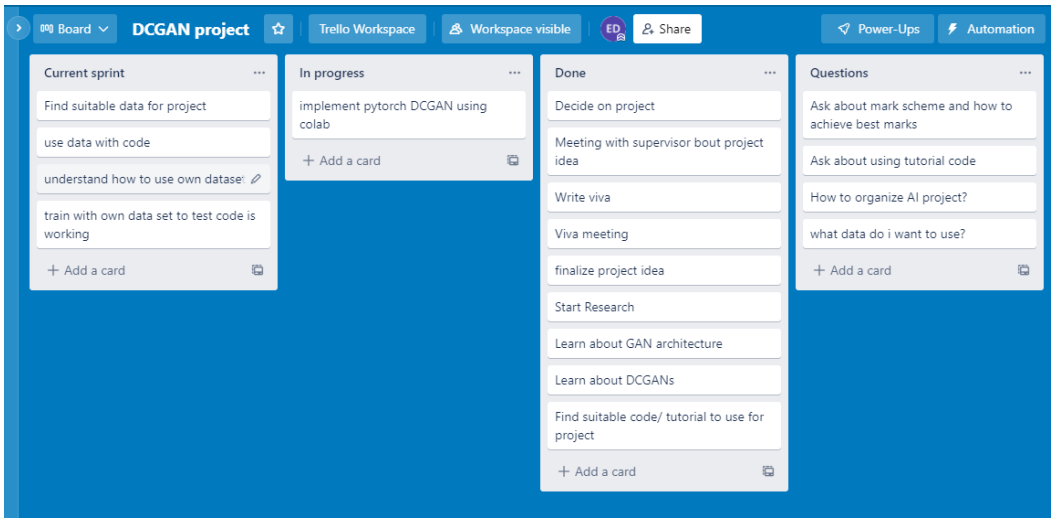
Appendix B Trello screenshots

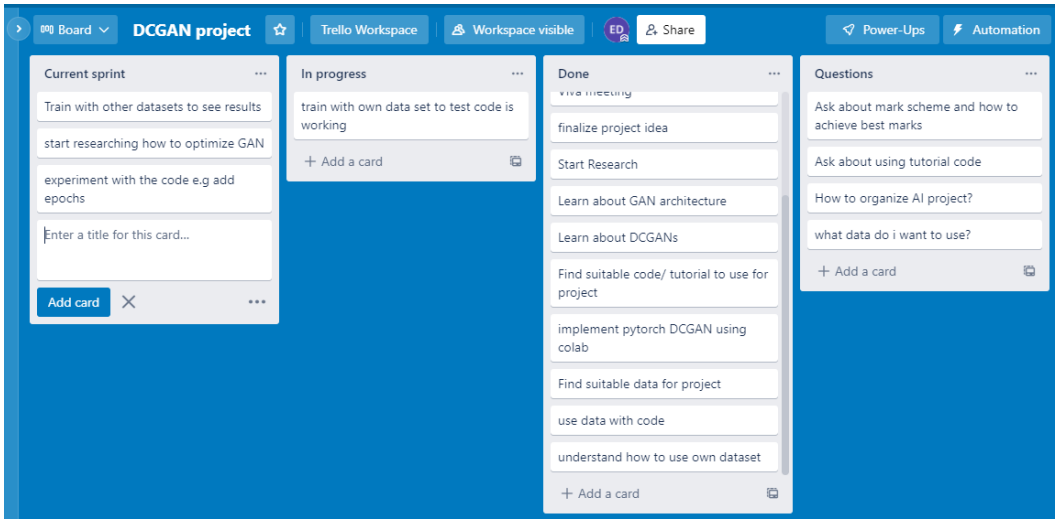
Selection of Trello screenshots from throughout the project to showcase project management.

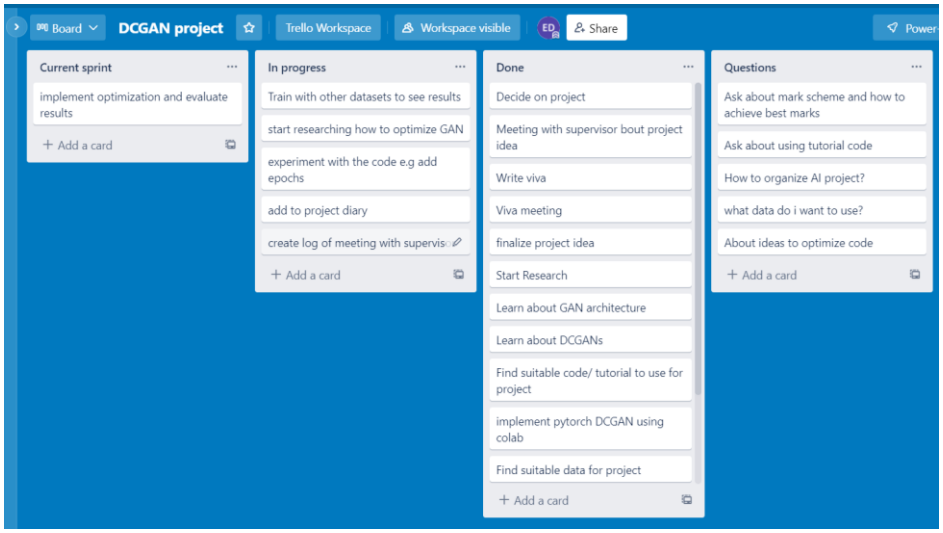


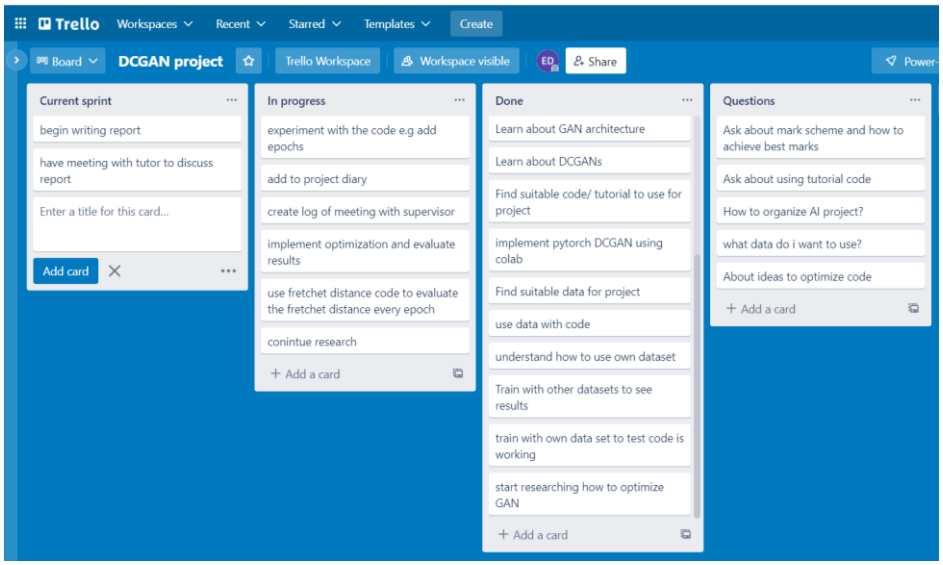



Trello
Workspaces
Recent
Starred
Templates
Create
Search
Board
DCGAN project
Trello Workspace
Workspace visible
ED
Share
Power-Ups
Automation
Current sprint
Learn about GAN architecture
Learn about DCGANs
Find suitable code/ tutorial to use for project
think about
Add a card
In progress
Start Research
Add a card
Done
Decide on project
Meeting with supervisor bout project idea
Write viva
Viva meeting
finalize project idea
Add a card
Questions
Ask about mark scheme and how to achieve best marks
Add a card


Board
DCGAN project
Trello Workspace
Workspace visible
ED
Share
Power-Ups
Automation
Current sprint
Find suitable data for project
use data with code
understand how to use own datase
train with own data set to test code is working
Add a card
In progress
implement pytorch DCGAN using colab
Add a card
Done
Decide on project
Meeting with supervisor bout project idea
Write viva
Viva meeting
finalize project idea
Start Research
Learn about GAN architecture
Learn about DCGANs
Find suitable code/ tutorial to use for project
Add a card
Questions
Ask about mark scheme and how to achieve best marks
Ask about using tutorial code
How to organize AI project?
what data do i want to use?
Add a card


Board
DCGAN project
Trello Workspace
Workspace visible
Share
Power-Ups
Automation
Current sprint
Train with other datasets to see results
start researching how to optimize GAN
experiment with the code e.g add epochs
Enter a title for this card...
Add card
In progress
train with own data set to test code is working
Add a card
Done
finalize project idea
Start Research
Learn about GAN architecture
Learn about DCGANs
Find suitable code/ tutorial to use for project
implement pytorch DCGAN using colab
Find suitable data for project
use data with code
understand how to use own dataset
Add a card
Questions
Ask about mark scheme and how to achieve best marks
Ask about using tutorial code
How to organize AI project?
what data do i want to use?
Add a card


Board
DCGAN project
Trello Workspace
Workspace visible
Share
Current sprint
implement optimization and evaluate results
Add a card
In progress
Train with other datasets to see results
start researching how to optimize GAN
experiment with the code e.g add epochs
add to project diary
create log of meeting with supervis
Add a card
Done
Decide on project
Meeting with supervisor bout project idea
Write viva
Viva meeting
finalize project idea
Start Research
Learn about GAN architecture
Learn about DCGANs
Find suitable code/ tutorial to use for project
implement pytorch DCGAN using colab
Find suitable data for project
Add a card
Questions
Ask about mark scheme and how to achieve best marks
Ask about using tutorial code
How to organize AI project?
what data do i want to use?
About ideas to optimize code
Add a card


Trello
Workspaces
Recent
Starred
Templates
Create
Board
DCGAN project
Trello Workspace
Workspace visible
Share
Current sprint
begin writing report
have meeting with tutor to discuss report
Enter a title for this card...
Add card
In progress
experiment with the code e.g add epochs
add to project diary
create log of meeting with supervisor
implement optimization and evaluate results
use fretchet distance code to evaluate the fretchet distance every epoch
conintue research
Add a card
Done
Learn about GAN architecture
Learn about DCGANs
Find suitable code/ tutorial to use for project
implement pytorch DCGAN using colab
Find suitable data for project
use data with code
understand how to use own dataset
Train with other datasets to see results
train with own data set to test code is working
start researching how to optimize GAN
Add a card
Questions
Ask about mark scheme and how to achieve best marks
Ask about using tutorial code
How to organize AI project?
what data do i want to use?
About ideas to optimize code
Add a card

Appendix C Meeting logs

Meeting Logs

Date	**Reason**
30/11/2021	Viva meeting to discuss initial project idea • Received feedback on the project as a whole. • How to make it into a project with substance. • Look into improved the GAN code • Start with research.
09/03/2022	Meeting about project progress • Presenting code with abstract images. • Deciding to not use abstract image dataset as discussed that it is hard to evaluate the output.
16/03/2022	General project update, showing progress since last meeting.
11/04/2022	Meeting to discuss the report and what is important to include for my project. • Talk about evaluation metrics • talk about methodology and planning
21/04/2022 (email correspondence)	Sent over literature section of my report for feedback. • Need to include more about generative art.
28/04/2022 (email correspondence)	Confirming report contains necessary

	sections. Planning a meeting to organize and discuss presentation.
06/05/2022	Meeting on presentation (yet to happen)