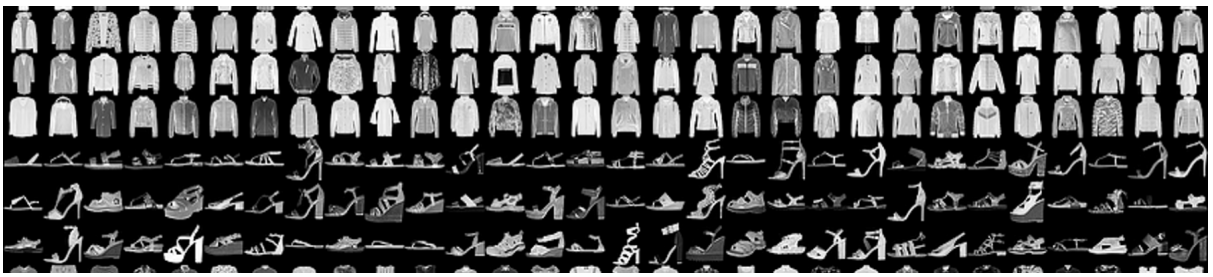
COMP4423 Computer Vision Project
Fashion Image Generation
Inidivual Report

YIM MAN TSUN 19064851d



Group Members:
LAM Chun Pang 19080189d
SHEK Yuan Sang 19037587d
YIM Man Tsun 19064851d

Table of content

I. Introduction

Overview of the project goals

The goal of this project is to develop a back-end algorithm and model that can automatically generate a diverse range of fashion images with control labels. The application is intended to help fashion designers like Sheila to reduce the stress and time associated with creating a plethora of fashion images manually. The objectives of the project are to generate quality images, with a variety of styles for each control label, using a provided fashion image dataset. This is a group project with a maximum of three group members per group, and each member must submit their own report emphasizing their contribution to the project. The project will be assessed through various tasks, including preprocessing the dataset, implementing a base model, identifying limitations and proposing enhancements, and developing a model capable of generating fashion images with controllable labels.

How do we achieve teamwork?

To promote teamwork in this project, we adopted a collaborative approach to develop the GAN model. We started by discussing the project requirements and objectives as a group and identifying the different roles that each member could take on based on their skills and interests. We then allocated specific tasks and responsibilities to each member, taking into account their strengths and preferences, while also ensuring that everyone had exposure to the different processes such as data preprocessing, designing the generator and discriminator, optimization, and model evaluation.

During the development of the GAN model, we held team meetings to discuss our progress, share ideas, and brainstorm solutions to any challenges we faced. We also made sure to provide feedback to one another. To ensure that each member had a clear understanding of their role, we maintained a shared document outlining the responsibilities and timelines for each task.

Overall, by adopting a collaborative approach, allocating specific roles and responsibilities, and maintaining open communication, we were able to promote teamwork and achieve our project objectives effectively.

II. Background

Overview of the problem

Fashion designers are required to create a plethora of clothing and accessory designs every day, which can be a time-consuming and stressful process. Traditional methods of manually creating these designs can limit the variety and quality of the output, leading to a suboptimal result. This problem is particularly relevant in the modern world, where consumers demand a diverse range of styles and designs to choose from.

Issue the project addressed

The issue that this project addresses is the need for a more efficient and automated way to generate a plethora of fashion images with control labels. By developing a back-end algorithm and model capable of generating a diverse range of fashion images, fashion designers like Sheila can reduce the time and stress associated with manual creation, while also producing a higher quality and more varied output. This project seeks to provide a solution to this problem by leveraging the power of deep learning and GANs to create a model that can generate fashion images with control labels, allowing designers to have more creative freedom and flexibility in their work.

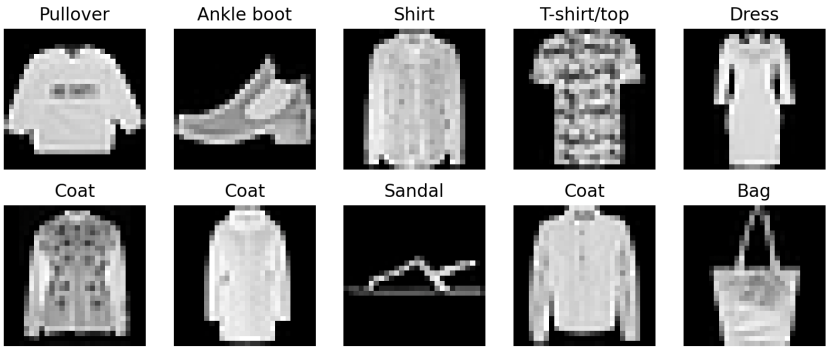
III. Methodology

Data Preprocessing

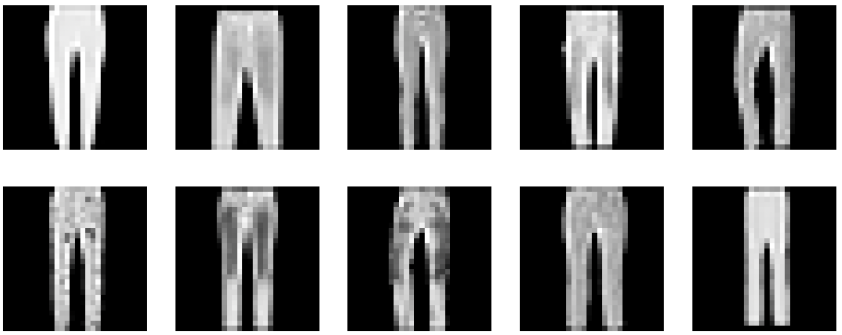
In order to train models for image generation, it is necessary to first obtain a dataset for the training process. The provided dataset includes 60000 images from 10 different categories, each with 785 columns that represent the label and pixel values in grayscale. However, before incorporating the data into the training process, it is important to normalize the pixel values from [0, 255] to [-1, 1] to enhance stability and generalization of the GAN models. This normalization of pixel values can effectively prevent mode collapse in the generator and converge real and generated images effectively for both the discriminator and generator.

Visualizing example images from the dataset

Next, we visualized some example images from the fashion dataset。
One of each:



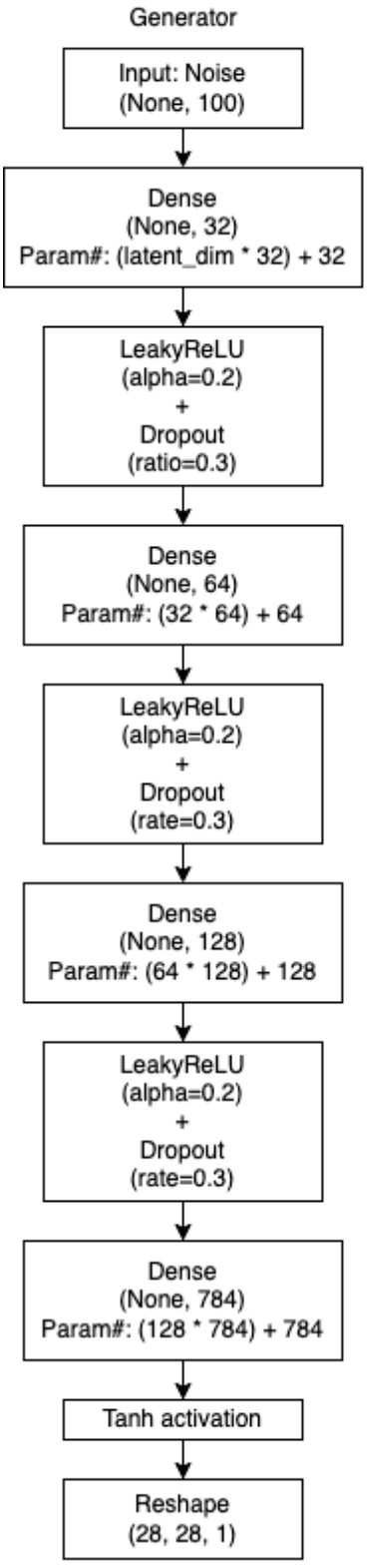
We could also display some with specific labels.
Trousers:

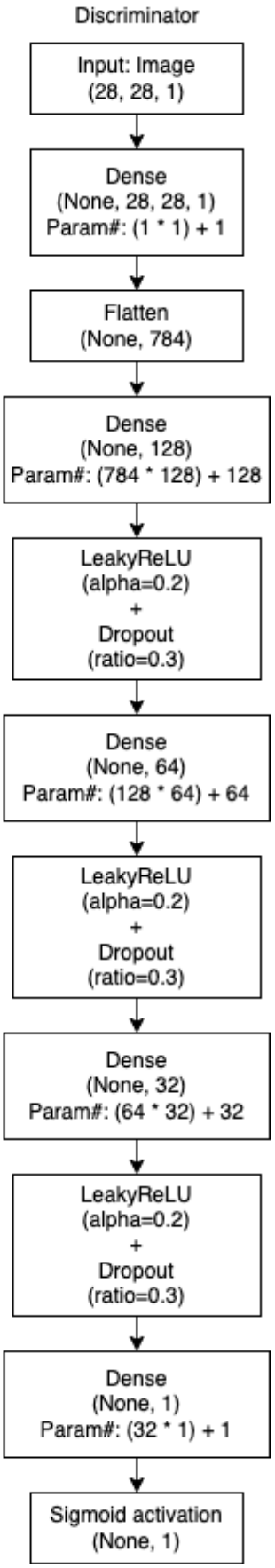


Coats:

Sandals:

Base Model (Without control label)





The generator architecture in this GAN is a simple feedforward neural network consisting of fully connected layers. It takes a noise vector as input and maps it to a higher-dimensional space through a series of dense layers with increasing numbers of units (32, 64, and 128). Leaky ReLU activation functions with an alpha value of 0.2 are applied after each dense layer to introduce non-linearity and help mitigate the vanishing gradient problem. Dropout layers with a rate of 0.3 are also included to prevent overfitting by randomly dropping out some neurons during training. The final dense layer with 784 units and a tanh activation function outputs a flattened 28x28 image. A reshape layer then converts the output into a 28x28 grayscale image with a single channel.

The discriminator architecture is also a feedforward neural network, but it is designed to classify input images as real or generated. The input to the discriminator is a 28x28 grayscale image, which is first passed through a dense layer with a single unit, then flattened into a 784-element vector. Following this, the input is passed through a series of dense layers with decreasing numbers of units (128, 64, and 32). Each dense layer is followed by a Leaky ReLU activation function with an alpha value of 0.2 and a dropout layer with a rate of 0.3 to introduce non-linearity and prevent overfitting. The final dense layer with a single unit and sigmoid activation function outputs the probability that the input image is real.

Limitation of our base model

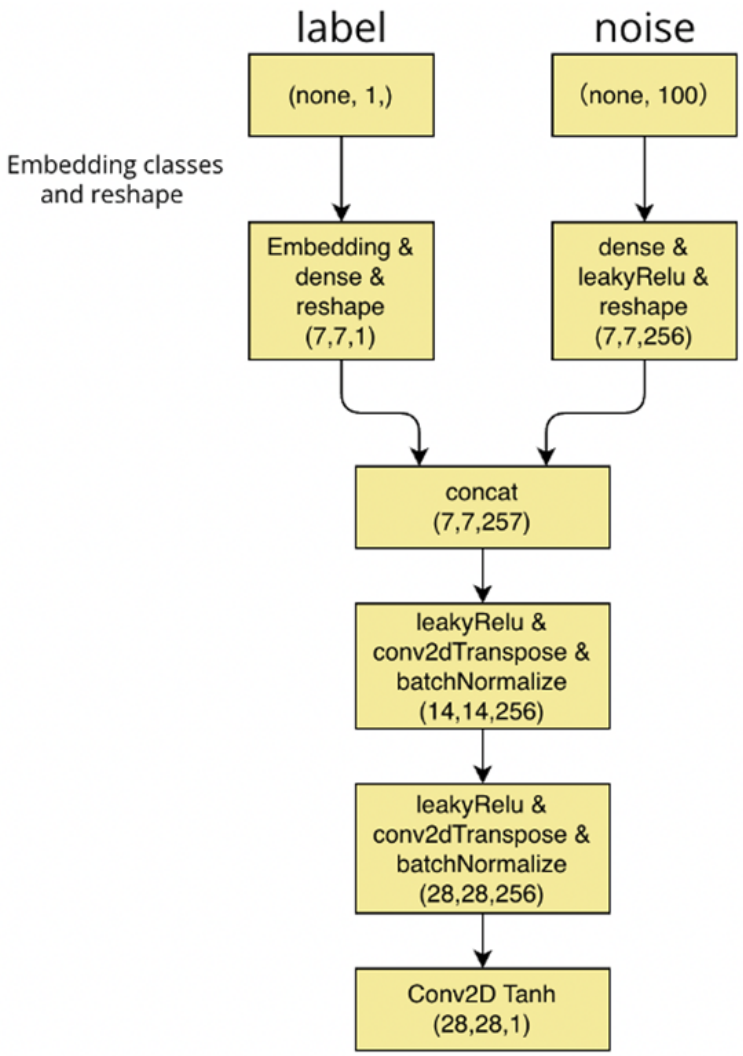
The major limitation of the baseline model is its reliance on fully connected layers, which are less effective at capturing complex features and patterns present in fashion images. Since fully connected layers do not consider the spatial information in images, the model may be less robust to noise and more prone to overfitting.

Advance Models

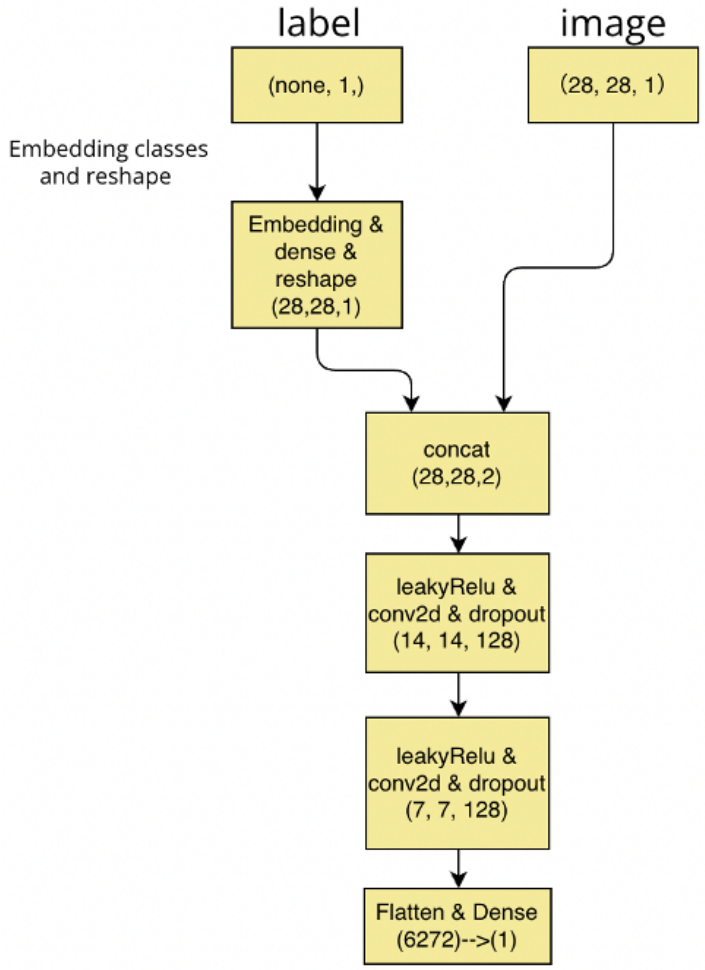
Conditional GAN (CGAN)

We considered using the conditional GAN to be an advanced approach, which we also provide the label to achieve supervised learning. The architecture of the given conditional GAN (CGAN) consists of a generator and a discriminator, both of which are designed to condition on class labels.

Generator:
The generator takes two inputs: a latent vector of size `latent_dim` and a class label. The label input is first passed through an embedding layer, which converts the integer class label into a continuous vector of size `latent_dim`. Then, it is passed through a dense layer to produce a tensor of shape (7, 7, 1), which is reshaped accordingly. The latent vector input is processed through a dense layer and then reshaped into a 7x7x256 tensor. The processed label input and latent vector input are merged using a concatenate layer.



Discriminator:
The discriminator also takes two inputs: an image of shape (28, 28, 1) and a class label. Similar to the generator, the label input is processed through an embedding layer followed by a dense layer, and then reshaped to match the input image shape. The processed label input and the image input are merged using a concatenate layer.
The merged tensor is then passed through two downsampling blocks, each consisting of a convolutional layer with a Leaky ReLU activation function and a dropout layer with a rate of 0.4. After the downsampling blocks, the tensor is flattened and passed through a dense layer with a single output unit and a sigmoid activation function to produce the output probability. The discriminator model is defined using the input and output tensors and is compiled with the Adam optimizer and binary cross-entropy loss.

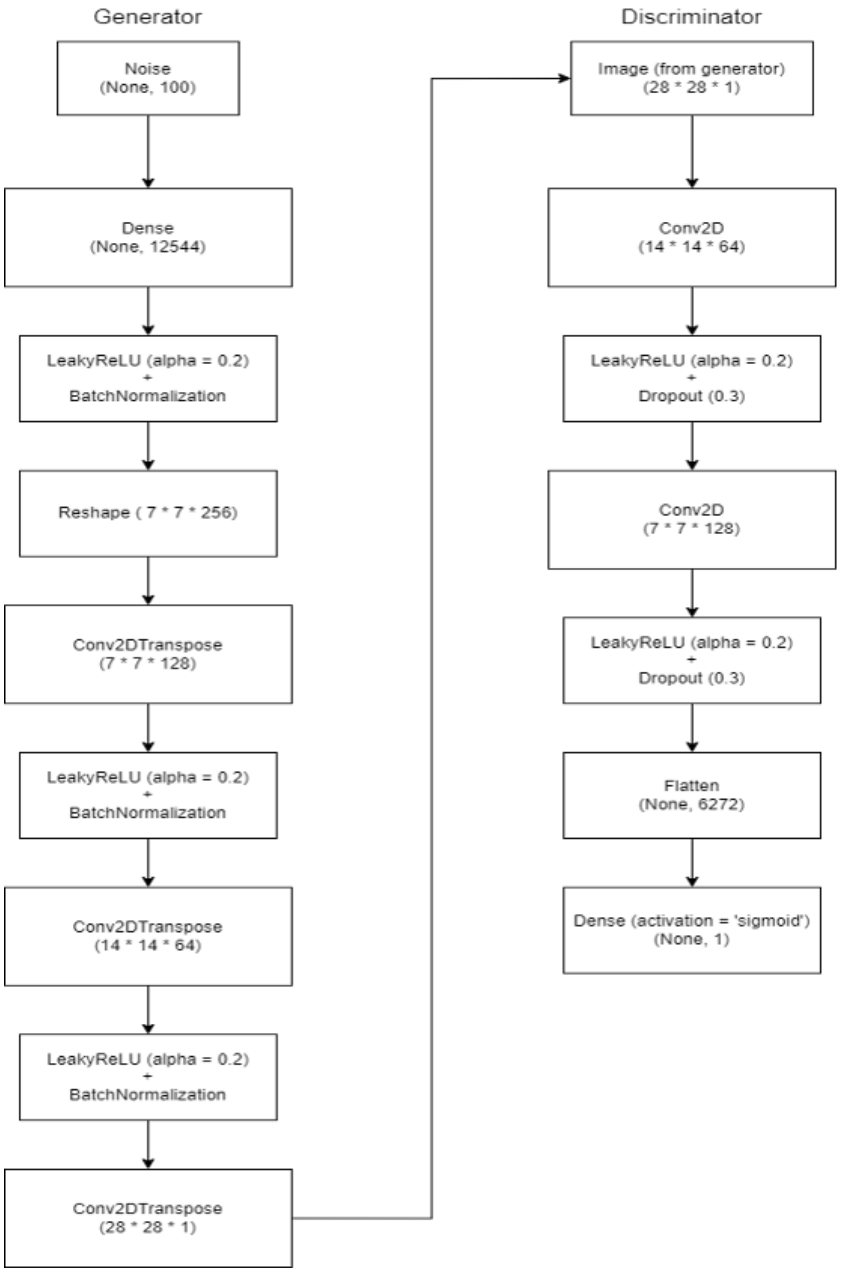


The CGAN performed well in both generating and discriminator(See the results in the RESULTS section). Yet we still propose some limitations of the conditional GAN:

- The architecture is relatively simple and might not be powerful enough to capture complex patterns in the data.
- The model might be prone to mode collapse, where the generator generates a limited variety of samples that are not diverse enough to represent the entire dataset.
- The model might be sensitive to hyperparameter settings such as the learning rate, the latent dimension size, and the number of layers in the generator and discriminator networks.

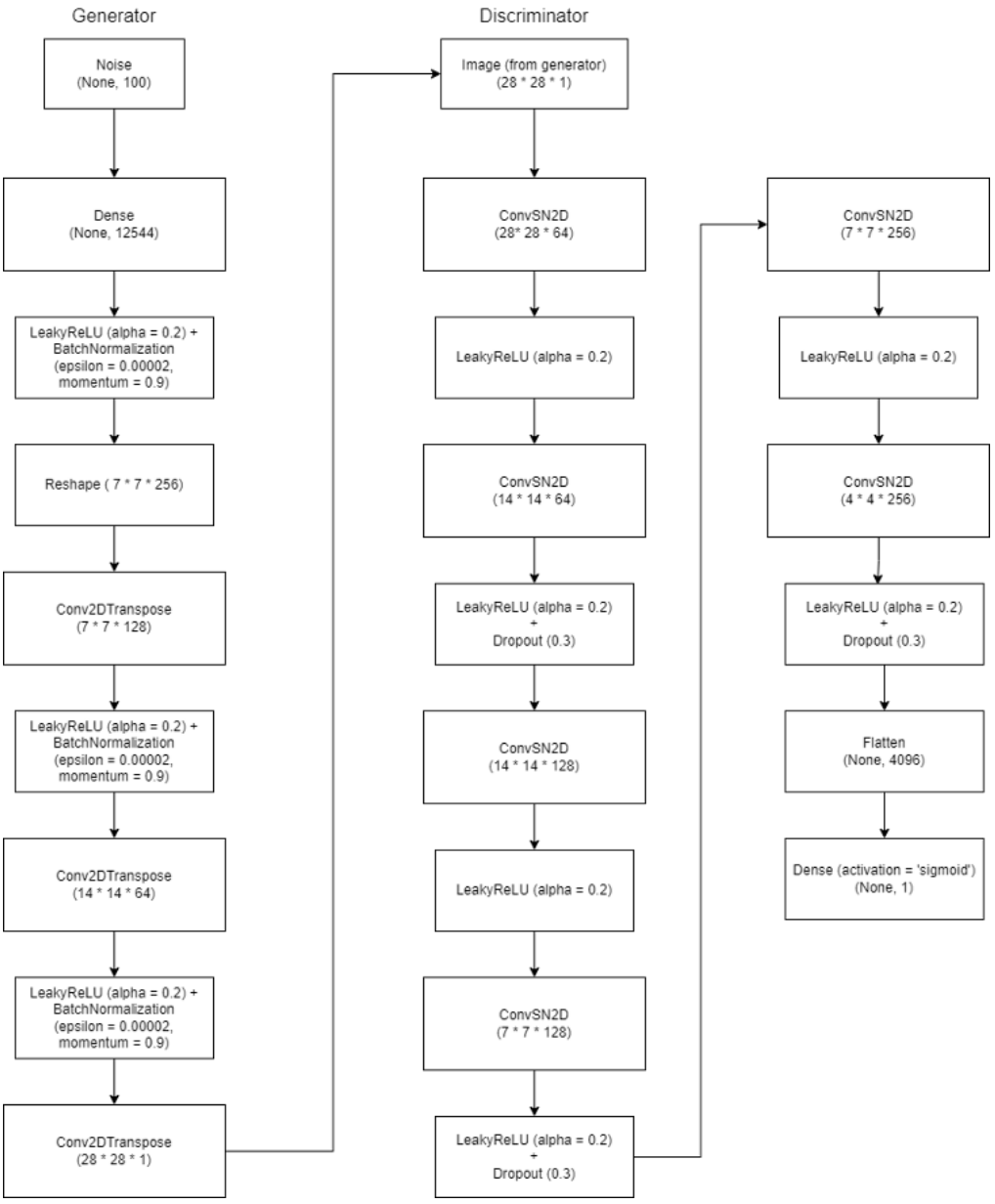
Deep Convolutional GAN (DCGAN)

This implementation of DCGAN is based on a TensorFlow GAN model tutorial. The generator comprises a dense layer with LeakyReLU activation and batch normalization, followed by two sets of Conv2DTranspose layers, LeakyReLU activations, and batch normalization. The final Conv2DTranspose layer generates a 28x28 fake image. The discriminator consists of two sets of a convolutional layer, LeakyReLU activation, and dropout layer, with the output being flattened and fed into a dense layer to produce the probability. This model replaces fully connected layers with convolutional layers, which are more suitable for image data and enable the generation of new images by up-sampling spatial dimensions. Batch normalization is employed to stabilize the training process and improve the stability of the model. However, the model may still suffer from mode collapse, vanishing gradients, and instability, which are sensitive to hyperparameter selection.



Spectral Normalization GAN (SNGAN)

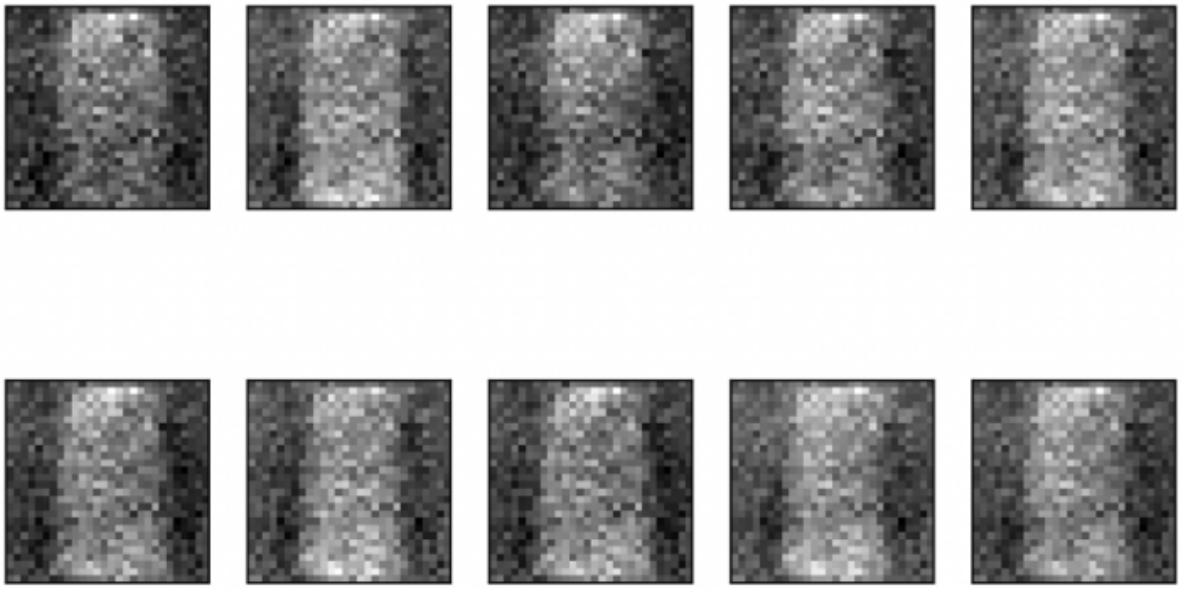
In this GAN, the generator undergoes only a minor alteration, which involves implementing batch normalization with specific parameters (epsilon=0.00002 and momentum=0.9) that can be advantageous for a dataset with significant variations. On the other hand, the discriminator undergoes more substantial changes, with the Conv2D layer being replaced by ConvSN2D and the network being adjusted to incorporate three sets of two ConvSN2D layers, each of which is followed by a LeakyReLU activation and dropout layer. ConvSN2D is a technique that introduces spectral normalization to normalize the weight matrices' spectral norm. This approach helps control the Lipschitz constant of the discriminator, which can otherwise dominate the generator during training. Additionally, utilizing two ConvSN2D layers with varying stride and filter values enables the discriminator to capture features from a range of input image scales.



IV. Results

Results from the base model

Images generated by the base model’s generator:



Based on the above screenshot, it is our belief that the model's ability to produce a diverse range of high-quality images with controlled labels is limited. This could be due to the fact that this particular category of images is relatively easy for the discriminator to classify, which makes it easier for the generator to produce images within this category. Since the unsupervised approach employed in this model does not provide labels, the discriminator must determine whether the image is authentic or not, regardless of its specific fashion category.

Results from CGAN

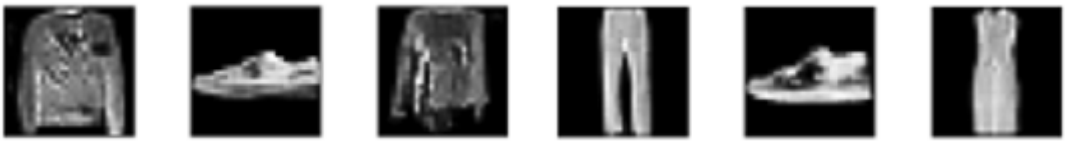
The following are some images generated by the GAN with control labels.
Trousers:

Shirts:

Ankle boots:

We can see that the model performed well and can generate diverse versions of fashion items.

Results from DCGAN



Results from SNGAN

VI. Conclusion

In this project we have developed several approaches to perform discrimination and generation tasks with the fashion dataset. Approaches included the base model, CGAN, DCGAN and SNGAN.

What have I learnt from this project

Throughout this GAN project, I have gained a comprehensive understanding of both supervised and unsupervised GAN models. Specifically, I have explored the functionality of supervised models such as CGAN, DCGAN, and SNGAN, as well as the base model of unsupervised GAN. Through this project, I have learned how to train models for image generation, normalize pixel values to enhance stability, and prevent mode collapse in the generator. Additionally, I have gained experience in working with large datasets, manipulating images, and evaluating model performance. Overall, this project has provided me with a valuable opportunity to develop my skills in deep learning and computer vision, as well as a greater understanding of the potential applications of GANs in various industries.

VII. References

Conditional GAN:
A Beginner's Guide to Building a Conditional GAN.
https://pub.towardsai.net/a-beginners-guide-to-building-a-conditional-gan-d261e4d94882

Deep Convolutional GAN:
Deep Convolutional Generative Adversarial Network
https://www.tensorflow.org/tutorials/generative/dcgan

Spectral Normalization GAN:
SpectralNormalizationKeras
https://github.com/WongKinYiu/yolov7/