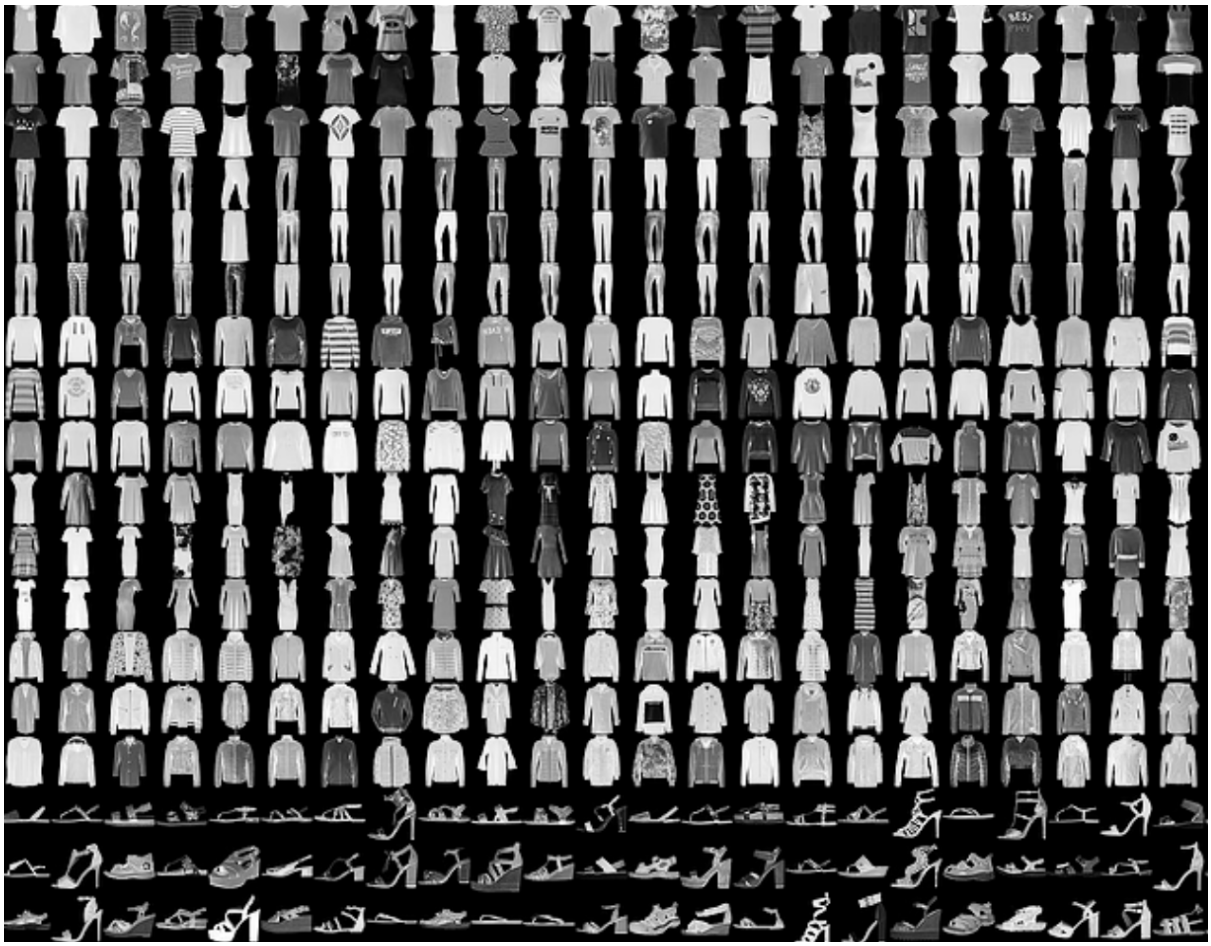
COMP4423 – Fashion Image Generation

Group Report



Group 7

Group Members:

LAM Chun Pang 19080189d

SHEK Yun Sang 19037587d

YIM Man Tsun 19064851d

Introduction

The Scenario is to develop an algorithm or model which can be used by a fashion designer to generate fashion images.

Our task is to develop a model which generate a **diverse range** of Quality plethora, with a **control label**, and **generate with variety**.

Data Preprocessing

As our aim is to train models for the image generation, before building the models for it, we need a dataset for the training process.

In the provided dataset, it contains 60000 images from 10 different categories. For each data, it contains 785 columns, the 1st column represents the label and the others are pixel value in greyscale, which is in range [0, 255]

However, before putting into our models for training, we normalize the values from [0, 255] to [-1, 1].

0	T-shirt/top
1	Trouser
2	Pullover
3	Dress
4	Coat
5	Sandal
6	Shirt
7	Sneaker
8	Bag
9	Ankle boot

GAN

For the GAN approach, we follow the structure of both generator and discriminator with the provided code and implement it in **Keras**.

However, we are implementing it with differences, by changing the **output shape of the generator** and the **input shape of the discriminator**.

It is important to make sure they are **sharing the same shape**, as the generator is providing images to trick our discriminator.

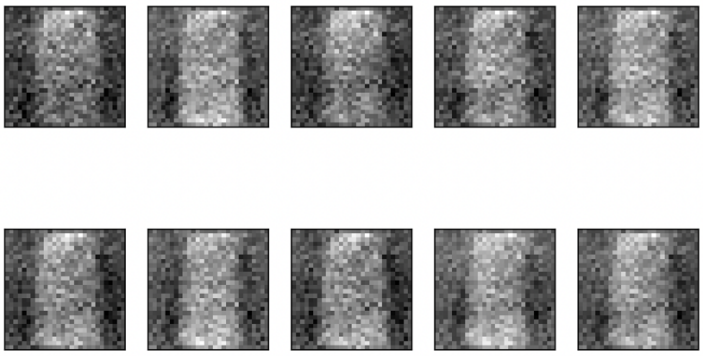
```python
def build_generator(latent_dim):
        #100->32->64->128->784
        model = Sequential(name='generator')
        model.add(Dense(32, input_dim=latent_dim))
        model.add(LeakyReLU(alpha=0.2))
        model.add(Dropout(0.3))

        model.add(Dense(64))
        model.add(LeakyReLU(alpha=0.2))
        model.add(Dropout(0.3))

        model.add(Dense(128))
        model.add(LeakyReLU(alpha=0.2))
        model.add(Dropout(0.3))

        model.add(Dense(784, activation='tanh'))
        model.add(Reshape((28, 28, 1)))
        return model
```

```python
def build_discriminator():
        #784->128->64->32->1
        model = Sequential(name='discriminator')
        model.add(Dense(1,input_shape=(28,28,1)))
        model.add(Flatten())

        model.add(Dense(128))
        model.add(LeakyReLU(alpha=0.2))
        model.add(Dropout(0.3))

        model.add(Dense(64))
        model.add(LeakyReLU(alpha=0.2))
        model.add(Dropout(0.3))

        model.add(Dense(32))
        model.add(LeakyReLU(alpha=0.2))
        model.add(Dropout(0.3))

        model.add(Dense(1, activation='sigmoid'))
        return model
```

Result Reviewing

However, there are some weaknesses for this approach. One of the main concerns is that, the generated image are more likely from a category.

Figure 1 10 generated samples from the GAN's generator



As shown as above capture, we believe that this model is **not good enough** to provide a **diverse range** of Quality plethora, with a **control label**, and **generate with variety**. The possible reason is this kind of images are easy to trick our discriminator. Therefore, the generator is more likely to generate images from this category.

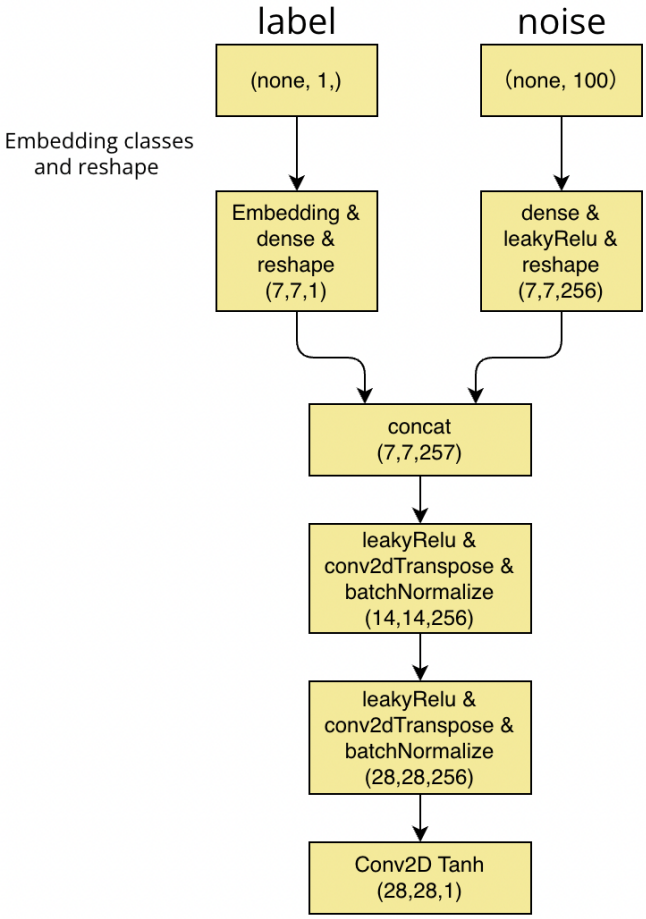
From the view of the discriminator, as this is an unsupervised approach, **labels are not provided** to the model. Therefore, the discriminator only needs to decide if this is a truth image **from any categories of the fashion images.** In here, as this category for generator is easy to learn and trick the discriminator, the generator is more likely to generate images from this category.

Conditional GAN

After that, we consider the conditional GAN. As we are providing extra information of the data(label), Conditional-GAN is the supervised version of GAN.

For the generator, the network structure is as below:

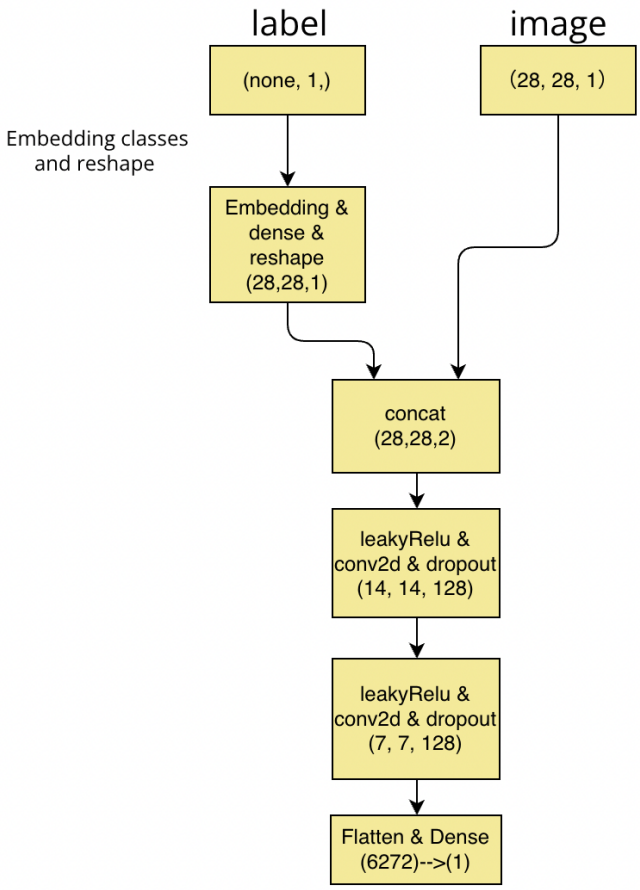
Generator network description



This graph shows the network structure of the generator, from the original source provided, the latent noise is 50 and the channels is 128. However, in our implementation, the latent noise is 100 and the channels is 256. the reason we made these changes is trying to build a more different result and have a better performance on the generator.

This network is building a sample image of size 7 * 7 and perform up sampling to a 28 * 28 * 1 image, by receiving a label and a noise from the beginning.

Discriminator Description



This graph shows the network structure of the discriminator. It takes the label and the generated image as input and see if the image matches its label or not.

Result Reviewing

The training hyperparams and settings are: (Adam optimizer: lr 0.0002, loss: binary cross entropy)

We generate 20 samples from each category and review the samples.

The results are improved, and the model can generate images for each different category.

For better display purpose, we reverse the value, making a white background and black item instead.

T-shirt/top

Trouser

Pullover

Dress

Coat

Sandal

Shirt

Sneakers

Bag

Ankle boot

The training result are good, and it can generate a **diverse range** of Quality plethora, with a **control label**, and **generate with variety**.

References

Conditional-Gan reference (https://pub.towardsai.net/a-beginners-guide-to-building-a-conditional-gan-d261e4d94882)