Application of DCGANs in drawing faces

Mohamed Dwidar

20100361

TA/ Mahmoud ElMorshedy

Network architecture:

I made the network having two models as instructed, one generator the other discriminator.

I also used mixed precision meaning the network would use 16-byte floats instead of 32-byte floats when it could because without it TensorFlow would throw an Out of VRam (OOM) error.

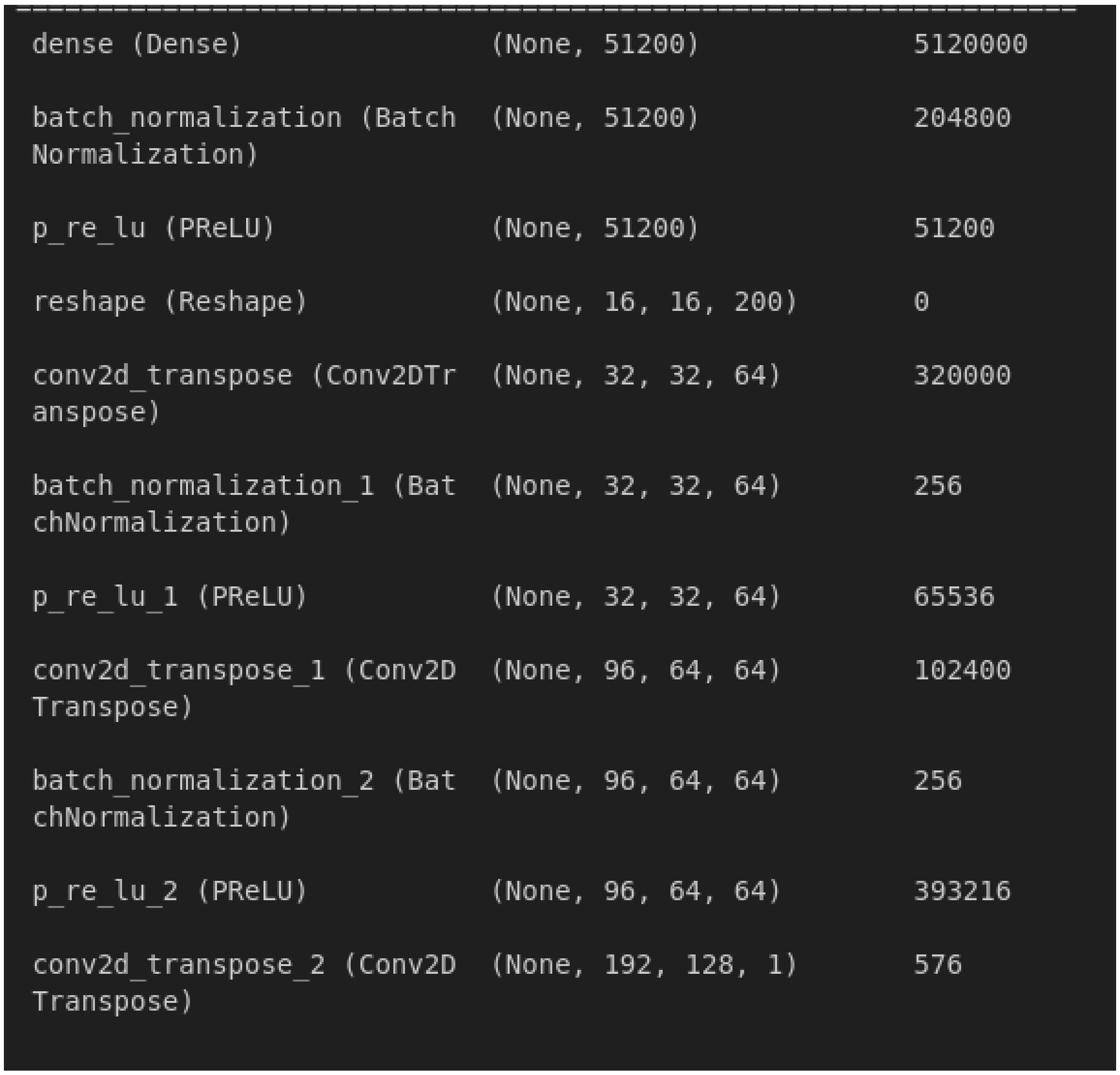
The GPU used for training: RTX 3060 laptop (6GB vram)

The generator:

Takes input shape of (100,)

Makes its way to the shape of (192,128,1) for a grayscale image

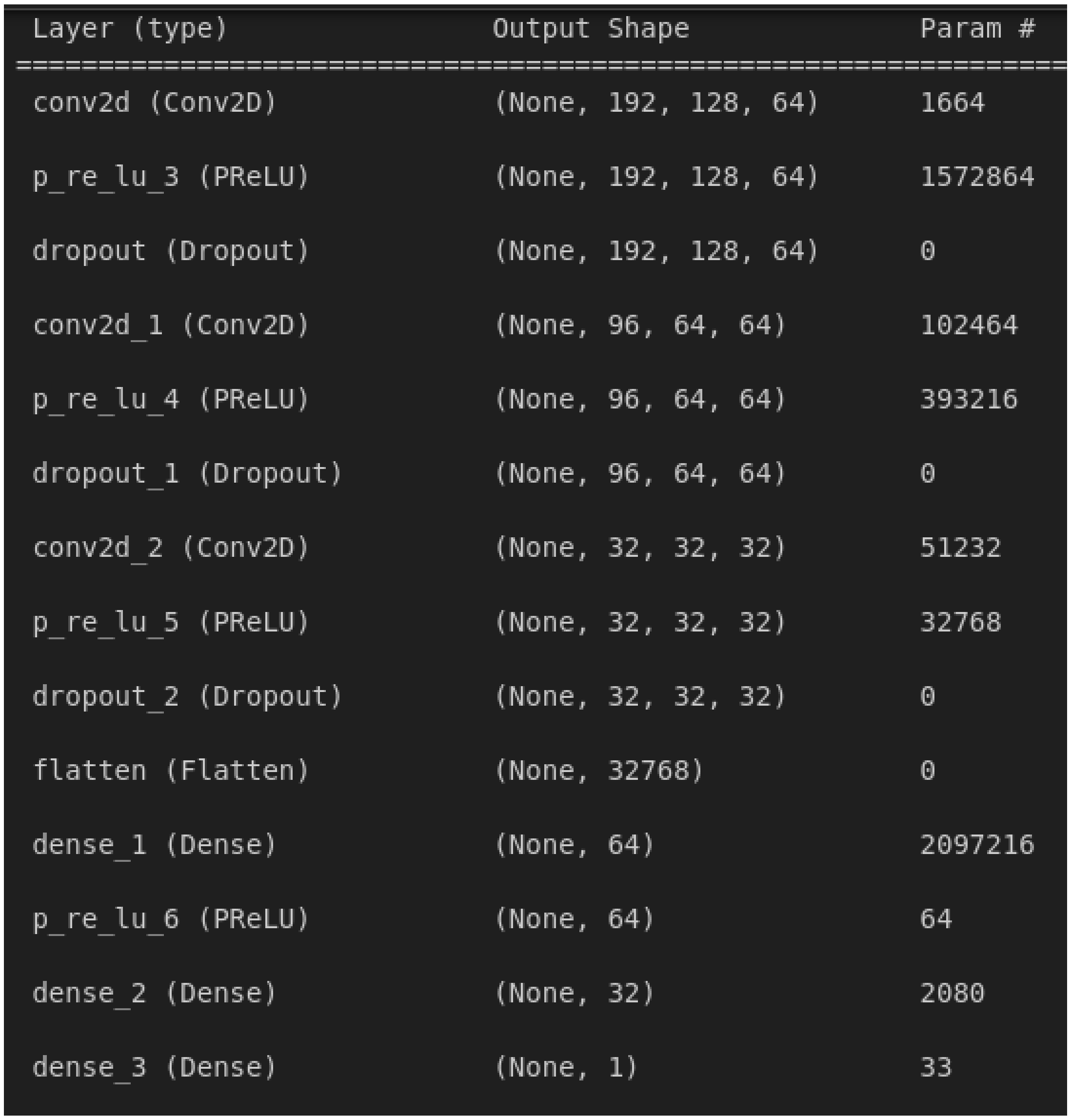
The data set was shaped to be compatible with the output of the network not the opposite



```
=================================================================
 dense (Dense)               (None, 51200)             5120000

 batch_normalization (Batch  (None, 51200)             204800
 Normalization)

 p_re_lu (PReLU)             (None, 51200)             51200

 reshape (Reshape)           (None, 16, 16, 200)       0

 conv2d_transpose (Conv2DTr  (None, 32, 32, 64)        320000
 anspose)

 batch_normalization_1 (Bat  (None, 32, 32, 64)        256
 chNormalization)

 p_re_lu_1 (PReLU)           (None, 32, 32, 64)        65536

 conv2d_transpose_1 (Conv2D  (None, 96, 64, 64)        102400
 Transpose)

 batch_normalization_2 (Bat  (None, 96, 64, 64)        256
 chNormalization)

 p_re_lu_2 (PReLU)           (None, 96, 64, 64)        393216

 conv2d_transpose_2 (Conv2D  (None, 192, 128, 1)       576
 Transpose)
```

The discriminator:

This one uses a very normal CNN to detect real from fake nothing new here



```
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 192, 128, 64)      1664

 p_re_lu_3 (PReLU)           (None, 192, 128, 64)      1572864

 dropout (Dropout)           (None, 192, 128, 64)      0

 conv2d_1 (Conv2D)           (None, 96, 64, 64)        102464

 p_re_lu_4 (PReLU)           (None, 96, 64, 64)        393216

 dropout_1 (Dropout)         (None, 96, 64, 64)        0

 conv2d_2 (Conv2D)           (None, 32, 32, 32)        51232

 p_re_lu_5 (PReLU)           (None, 32, 32, 32)        32768

 dropout_2 (Dropout)         (None, 32, 32, 32)        0

 flatten (Flatten)           (None, 32768)             0

 dense_1 (Dense)             (None, 64)                2097216

 p_re_lu_6 (PReLU)           (None, 64)                64

 dense_2 (Dense)             (None, 32)                2080

 dense_3 (Dense)             (None, 1)                 33
```

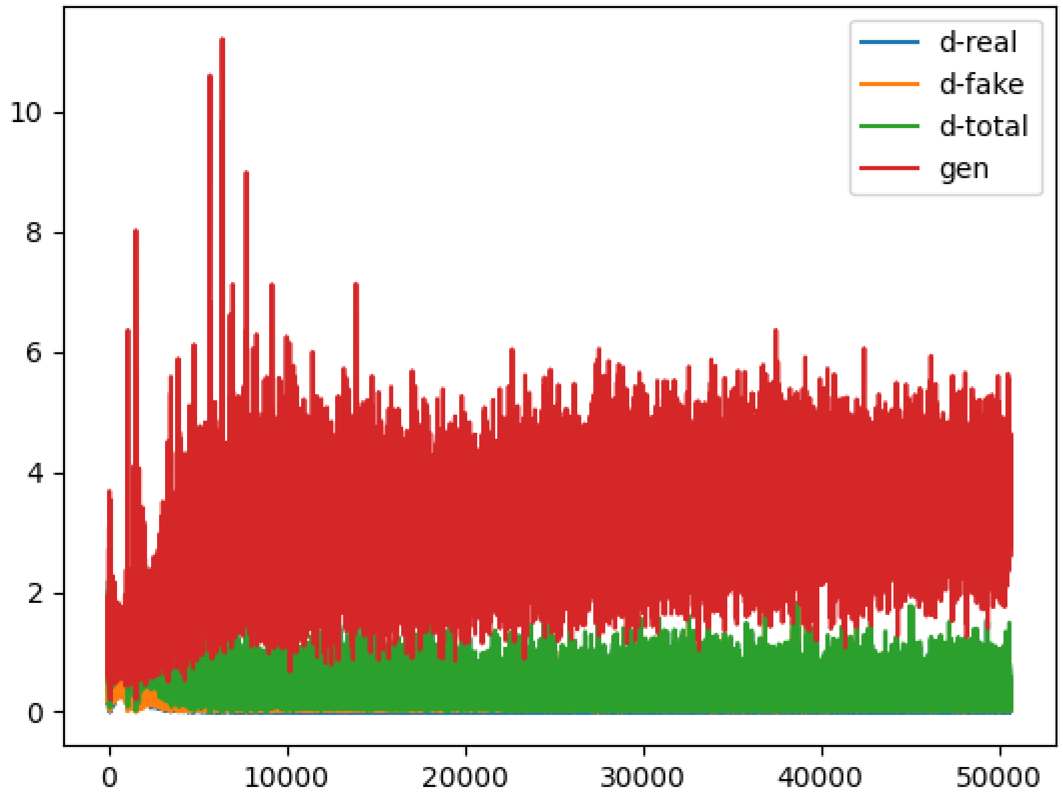
The dataset:

The dataset was resized to 192*128 and gray scaled

The dataset size is 11GB so I loaded it as batches instead of loading the whole dataset to memory as it would cause the kernel to crash

The training results:

The loss chart:



The images: