Neural Network Report

Adriano Puglisi 1743285
Vincenzo Colella 1748193

April 2022

Contents

1 Introduction

The goal of this project is to implement an attention based generative adversarial network (GAN) that is able to generate Fundus Angiography with a fast and non-invasive technique.
The dataset used is composed of 60 pairs of images between the fundus and angiogram pairs but since not all of them are aligned, only a small number of those pairs were selected.
The original image size is 720 x 576 but we took 20 crops of 256 x 256 due to hardware limitations.

2 Data Preprocessing

The chosen dataset is taken from the research paper *S. Hajeb Mohammad Alipour, H. Rabbani, and M. R. Akhlaghi, "Diabetic retinopathy grading by digital curvelet transform," Computational and mathematical methods in medicine, vol. 2012, 2012.* The dataset contains fundus fluorescein angiogram photographs and corresponding color fundus images of 30 healthy persons and 30 patients with diabetic retinopathy.
From these, we selected a total of 17 pairs, 10 from normal patients and 7 from abnormal patients, because they were the only ones with the fundus images and angiography images lined up.

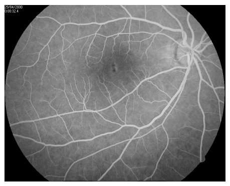


Figure 1: Real Image 720x576

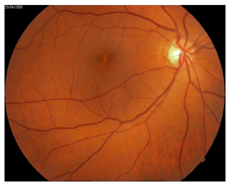


Figure 2: Real Image 720x576

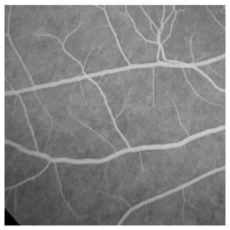


Figure 3: Random Crop

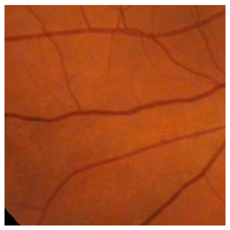


Figure 4: Random Crop

3 Models

The proposed GAN architecture is composed of two generators and four discriminators (two fine and two coarse):

- Fine Generator - synthesizes FA from fundus images by learning local information
- Coarse Generator - aims to extract and preserve global information
- Fine Discriminator - dictate the fine generator to produce more detailed local features. It takes as input the sample size
- Coarse Discriminator - tries to convince the coarse generator to retain more global features. It takes as input half of the sample size

3.1 Generators Structure

Fine Generator structure

- Convolution block
- Feature map
- Encoder block
- Three residual block
- Decoder block
- Feature map
- Convolution block

There is a skip connection between the two feature map.

Coarse Generator structure

- Convolution block
- Feature map
- Encoder block
- Feature map
- Encoder block
- Nine residual block
- Decoder block
- Feature map
- Decoder block
- Feature map
- Convolution block

It has two skip connection between first and third feature map and second and fourth feature map.

The last Convolutional layer of the coarse generator is added to the first residual block of the fine generator.

3.2 Discriminators Structure

Fine/Coarse Discriminator 1

- Convolution block
- Encoder block
- Residual block
- Encoder block
- Residual block
- Encoder block
- Residual block
- Convolution block

Fine/Coarse Discriminator 2

- Encoder block
- Convolution block
- Encoder block
- Residual block
- Encoder block
- Residual block
- Encoder block
- Residual block
- Convolution block

4 Modifications

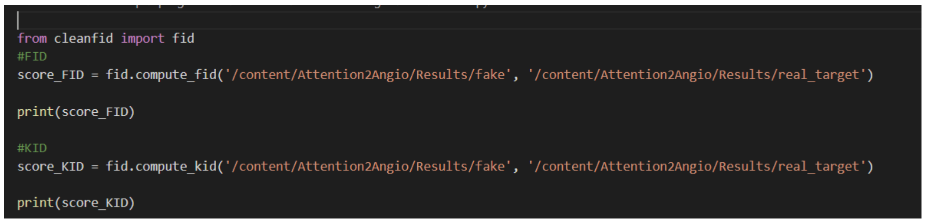
4.1 First fixes

First we had to update the code provided by the original author to make the Neural Network actually work. Our first step was to add few necessary lines in our code:

- The function g_model_fine in g_fine_model wasn't defined correctly, and we had to correct it.
- The function g_coarse_fine in g_fine_coarse wasn't defined correctly, and we had to correct it.
- There were some missing imports, as 'import os' and 'from functools import partial'.
- We implemented the function performance_visualize in order to be able to check the performance during the training.

4.2 Performance Visualizers

The only way to analyse the performance of the network, since we needed more precise and dynamic performance visualizers to understand the progression of our generators in creating the final image, was to implement two GAN evaluation measures:



```
from cleanfid import fid
#FID
score_FID = fid.compute_fid('/content/Attention2Angio/Results/fake', '/content/Attention2Angio/Results/real_target')

print(score_FID)

#KID
score_KID = fid.compute_kid('/content/Attention2Angio/Results/fake', '/content/Attention2Angio/Results/real_target')

print(score_KID)
```

Figure 5: FID and KID code

- **Fréchet Inception Distance (FID)** - is a metric that calculates the distance between feature vectors calculated for real and generated images. The score summarizes how similar the two groups are in terms of statistics.
- **Kernel Inception Distance (KID)** - measures the dissimilarity between two probability distributions using samples drawn independently from each distribution.
- **Structural Similarity Index Measure (SSIM)** - measures degradation that is caused by processing images or by losses in data transmission. We implemented it but later realized that it required two graphically identical images in which one has been filtered or compressed and SSIM measured that loss. Since it isn't our goal, we aren't considering the SSIM function as a metric in our project.

Using the FID and KID functions in every step of the training, we are able to see how is performing our model without having to wait the end of the training. Since the implementation found takes images from three folders and compare them, we had to implement a code that saves our original crops, generated crops, and ground truth crops at every step in three different folders. The code erases the old pictures in the folders (from the previous step) and finally computes the kid and fid values.

4.3 Loss Functions

Then we decided to modify the loss function of the original model as it was a custom *perceptual_ loss_ coarse* and *perceptual_ loss_ fine* that the author implemented to enhance the results of his model. By studying these loss functions, we realized that they obviously have their pros and cons. Perceptual Loss is useful to avoid distorted local structure due to not learning contrast and abnormal color of the optic disc.
On the other hand, many GANs for synthesizing Fluorescein Angiography don't use this loss function and work well, and in some circumstances, even better that the this model. For this reasons we decided to implement a new argument in the training function that allows to chose to train the model using the mean squared error (MSE) function or the original Perceptual Loss (PL).
By setting mod=0 we let the model train with the original settings as written

in the paper, while by setting mod=1 we replace both perceptual loss functions to mean squared error.

5 Results

In order to train the network we first randomly cropped 20 instances of each picture with a size of 256 x 256 for a total of 340 images for each size. After this we saved the cropped images inside an *.npz* file to take up less memory. At this point we were able to train the network, we decided to make different train on the 256 image size by changing the batch_size, number of epochs in order to check which network gets the best results.

the combinations that we used are:

- –input_dim=256 –batch=8 –epochs=100 –n_crops=20 –mod=0
- –input_dim=256 –batch=4 –epochs=100 –n_crops=20 –mod=0
- –input_dim=256 –batch=4 –epochs=100 –n_crops=20 –mod=1
- –input_dim=256 –batch=2 –epochs=200 –n_crops=20 –mod=0
- –input_dim=256 –batch=2 –epochs=200 –n_crops=20 –mod=1

The last two combinations were the most successful ones. The time necessary to complete the training process with Perceptual Loss (mod = 0) in Google Colab was approximately 8 hours for 100 epochs and 12 hours for 200 epochs. Instead, for the training with Mean Squared Error (mod = 1) the time reduced to 6 hours for 100 epochs and 10 hours for 200 epochs.

The results for the various training are the following:

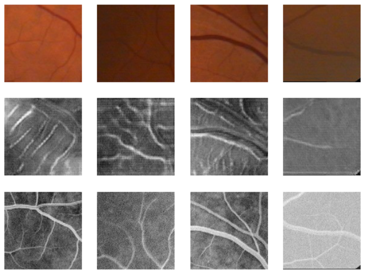


Figure 6: global_8_100_mod0

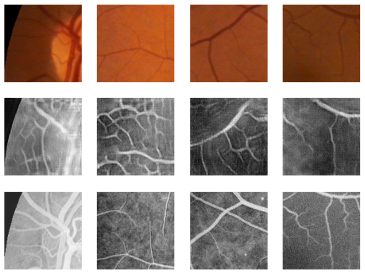


Figure 7: global_4_100_mod0

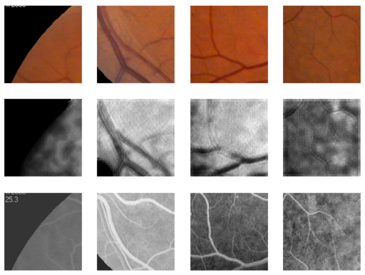


Figure 8: global_4_100_mod1

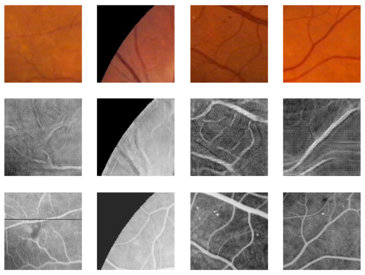


Figure 9: global_2_200_mod0

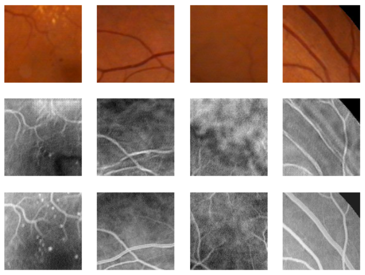
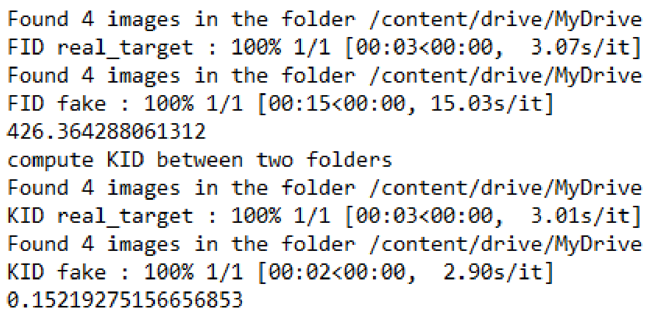


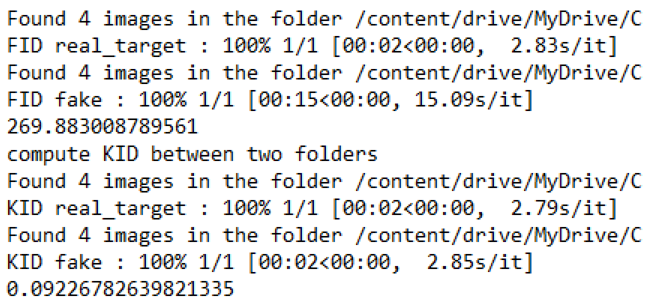
Figure 10: global_2_200_mod1

As we can see from the pictures the most similar to the ground truth and therefore realistic are the one produced with 200 epochs and batch 2 as we could have imagined. This is because we train the network with few batches and for a great number of epochs, that allows it to better retain the important information of the optic disk. Between those two, the best method is the one with mod = 0 and this is confirmed by the test done with FID and KID:



```
Found 4 images in the folder /content/drive/MyDrive
FID real_target : 100% 1/1 [00:03<00:00,  3.07s/it]
Found 4 images in the folder /content/drive/MyDrive
FID fake : 100% 1/1 [00:15<00:00, 15.03s/it]
426.364288061312
compute KID between two folders
Found 4 images in the folder /content/drive/MyDrive
KID real_target : 100% 1/1 [00:03<00:00,  3.01s/it]
Found 4 images in the folder /content/drive/MyDrive
KID fake : 100% 1/1 [00:02<00:00,  2.90s/it]
0.15219275156656853
```

Figure 11: FID and KID with mod 1



```
Found 4 images in the folder /content/drive/MyDrive/C
FID real_target : 100% 1/1 [00:02<00:00,  2.83s/it]
Found 4 images in the folder /content/drive/MyDrive/C
FID fake : 100% 1/1 [00:15<00:00, 15.09s/it]
269.883008789561
compute KID between two folders
Found 4 images in the folder /content/drive/MyDrive/C
KID real_target : 100% 1/1 [00:02<00:00,  2.79s/it]
Found 4 images in the folder /content/drive/MyDrive/C
KID fake : 100% 1/1 [00:02<00:00,  2.85s/it]
0.09226782639821335
```

Figure 12: FID and KID with mod 0

In figure 11 the FID (lower is better) is 426 and KID (lower is better) is 0.15. Meanwhile in the figure 12 the FID and KID are respectively 269 and 0.09.

6 Conclusions

As expected, the Perceptual Loss model has a lower FID and KID in all cases. It is able to generate less distorted pictures since it detects better the contrast and abnormal color of the optical disk in the picture. This means that Perceptual Loss is the recommended loss function for better final results. On the other hand, by using mean squared error as loss function will drastically reduce training time and still produce appreciable results, in fact we still get a fluorescein angiography but with less details. Moreover, we observed that for this model the best approach was to lower as much as possible the batch size, and maximize the number of epochs.