DCGAN for Enhancing Eye Diseases Classification

Mahmoud Smaida[a], Serhii Yaroshchak[a], Youness El Barg[c]

[a] *The national university of water and Environmental Engineering, Rivne, Ukraine*
[b] *Multi-disciplinary Faculty Laraxche, Larache, Morocco*

Abstract

One of the most important factors in image classification is the amount of data, especially in medical images. However, In the medical field, obtaining these datasets remains a challenge. In this work, we present Deep Convolutional Generative Adversarial Network (DCGAN) method that generate synthetic medical images. In addition, using GMD (Glaucoma, Myopia and Diabetic retinopathy) model to improve eye diseases classification with and without synthetic medical images. Four types of eye diseases, Glaucoma, Myopia, Diabetic retinopathy and Normal are our dataset. Our method is demonstrated on a limited dataset of eye disease (300 Glaucoma, 300 Myopia, 300 Diabetic retinopathy and 300 Normal). Firstly, we exploit DCGAN to generate synthetic medical images. then we utilize GMD method for eye diseases classification. finally, training our method using original data and synthetic medical images, then compare performance.

The accuracy of the model had improved significantly from 76.58% in training set and 76.42% in validation set, to 80.45% in training set and 83.74% in validation set. We suggest that this work can be applied to other image classification models such as Vgg16, Inception v3, ResNet to enhance the accuracy.

Keywords 1

Eye diseases, Deep learning, Myopia, DCGAN, Glaucoma, Diabetic retonapthy

1. Introduction

One of the main challenges in the field of medical imaging is how to deal with small data sets, especially when we use supervised learning to classify images. In medical imaging assignments, explanatory reports are made by specialists with experience in his or her specialty. Although, some of medical data sets are able to be used online, and major challenges have been declared, most data sets remain limited and only apply to specific medical issues. Collecting medical data is a complex and costly procedure that requires collaboration between researchers and clinics [1].

Many researchers try to solve this problem with augmentation diagrams, including making some adjustments to the images of the data set such as rotation, cropping and size. Using such redundant data to improve network training has become a standard procedure for computer vision tasks, recently one of the most important method used is GANs.

Generative Adversarial Networks (GANs), are a type of machine learning network that Ian Goodfellow and colleagues invented in 2014. Two neural networks compete with each other in a game (meaning game theory, often but not always in the form of a zero-sum game). The aim of it is to creating fabricated data similar to real data, which are difficult for a human or mechanical observer to distinguish between them. This technique learns to generate new data with the same statistical properties of a training set. For example, a GAN trained in photographs can create new images that

CMIS-2021: The Fourth International Workshop on Computer Modeling and Intelligent Systems, April 27, 2021, Zaporizhzhia, Ukraine
EMAIL: m.e.smaida@nuwm.edu.ua (M. Smaida); s.v.yaroshchak@nuwm.edu.ua (S. Yaroshchak); youness.elbarg@etu.uae.ac.ma (Y. El Barg)
ORCID: 0000-0002-5552-2768 (M. Smaida); 0000-0001-9576-2929 (S. Yaroshchak); 0000-0002-6386-5402 (Y. El Barg)
© 2020 Copyright for this paper by its authors.
Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).
CEUR Workshop Proceedings (CEUR-WS.org)

CEUR-WS.org/Vol-2864/paper3.pdf

look real to human observers, and have many photorealistic characteristics. Although it was originally proposed as a form of the generative model of unsupervised learning, GANs have also proven useful for semi-supervised learning, fully supervised learning, and reinforcement learning. At the 2016 symposium, AI expert Yann LeCun described GANs as "the coolest idea in the field of machine learning in the past 20 years" [2].

The Generative Adversarial Network consists of two separate neural networks, Generator (G) and Discriminator (D) constantly competing against each other, the generator, which tries to generate examples that seem the real data, it works to deceive the discriminator, while the role of discriminative network is to distinguish between the real and fake data. a vector of random number will be feeds to the generator, and the outputs are unreal synthetic data denoted by G (z), the discriminator feeds real data and the output of generator. The output of the discriminator is the possibility whether the entered data is real or fake, the output denoted by D (x). The output D (x) represents the probability that x will be a real image if it is 1, which means 100% is a real image, and the output is 0 which means that the image cannot be real.

During training, the goal of creating the Generator is to try to create real images to deceive the Discriminator. The goal of Discriminator is to separate the Generated images from the real images as closely as possible.

What is the outcome of the final match? In the most perfect case, Generator can create a real and fake G (z) images. For D it is difficult to determine whether the image generated by G is real so D (G (z)) = 0.5.

In this paper, we investigate the problem of lack of medical images such as eye diseases for image classification. We propose Generative Adversarial networks (GANs) to increase our dataset in order to improve the accuracy of classification using our model (Glaucoma, Myopia and Diabetic retinopathy model). The model based on CNN for image classification.

In this way, our goal is achieved and we obtain the generative model G, which can be used to create images in order to increase our dataset [4].

1.1. How GAN is work:

The above is just a rough overview of the basic principles of GAN, how can it be described in mathematical language? Here is a straightforward excerpt from the formula from Ian Goodfellow's paper, which called objective function of GAN [3]:

$$\min_G \max_D V(D,G) = E_{x \approx pdata(x)}\left[\log(D(x)\right] + E_{z \approx pz(z)}[\log(1 - D(G(z)))] \quad (1)$$

You only need to analyze this formula:

- The complete formula consists of two periods. X represents the real image; Z represents the noise input to the G network.
- G (z) is a fake image generated by the generator G.
- D (x) is the probability of that X came from the real data (0~1), the discriminator should classify a real image as real. And this value should be close to 1.
- Purpose of G as mentioned above. D (G (z)) is the probability that a D network governs whether the image generated by G is real. In other words, G wants D (G (z)) to be as large as possible (close to 1), and V (D, G) will be smaller at this time. So, we see that the forward tag of the formula is min_G.
- Purpose of D: is simply a classifier. It tries to distinguish real data from the data created by the generator. train D to classify real images as real, D(x) should be close to 1. Train D to classify fake images as fake, D (G (z)) should be close to 0.

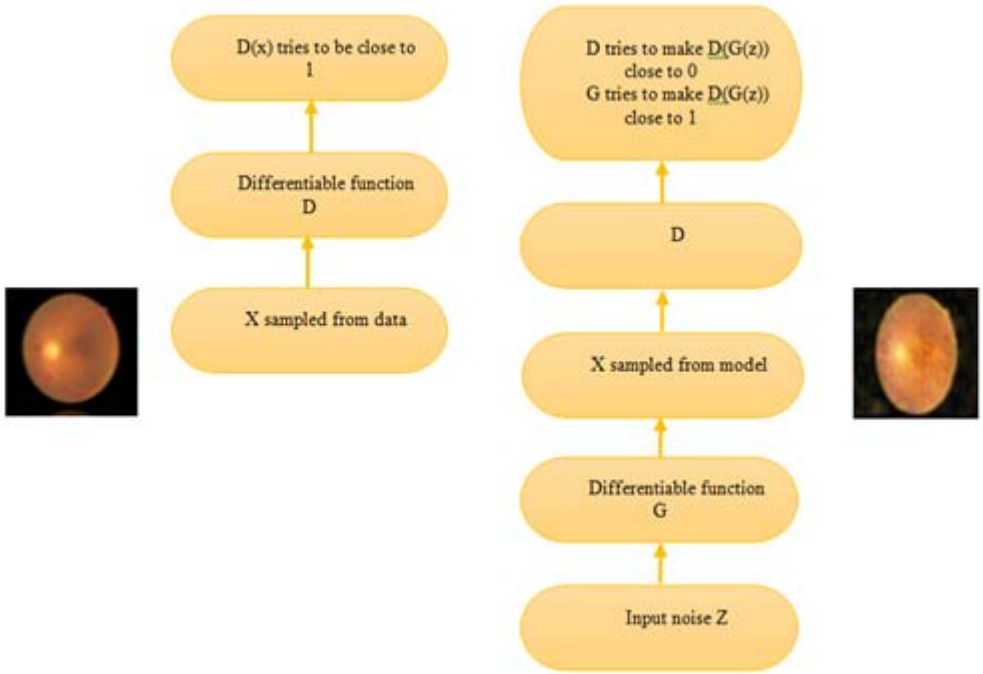

Figure 1: Adversarial Nets Framework

1.2. Training process in GAN:

Since the generative model and the discriminant model in the generative adversarial network are completely independent models and cannot be trained at the same time, separate repetitive training is used. For the discriminant, the label for the true sample is marked with the number 1, and the label for the generated sample is marked as 0, regardless of the quality of the sample generated.

Updating the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta d} \frac{1}{m} \sum_{i=1}^{m} \log D(x^{(i)}) + \log(1 - D(G(z^{(i)})))] \quad (2)$$

Updating the generator by descending its stochastic gradient:

$$\nabla_{\theta g} \frac{1}{m} \sum_{i=1}^{m} \log(1 - D(G(z^{(i)}))) \quad (3)$$

We train D with the hope that V (G, D) is close to 1, so we add a gradient (ascending). When training G in the second step, V (G, D) is as small as possible close to 0, so the gradient is subtracted. And then the entire training process rotates.

1.3. Different types of GANs

There have been many types of GAN since its first appearance in 2014, including Convolutional GAN (CGAN), Deep Convolutional GANs (DCGANs), Least Squares GANs(LSGAN), Boundary Equilibrium GANs(BEGAN), StackGAN, InfoGANs, Wasserstein GANs (WGAN) and DiscoGANs. The dcgan that has been used in this paper will be explained briefly [4], [5], [6], [7], [8].

1.3.1. Concept of DCGAN

DCGAN is Deep Convolutional Generative Adversarial Network, the same as GAN, so we won't repeat it here. It only replaces the G and D above with two convolutional neuronal networks (CNN). But the change is not directly enough, DCGAN made some changes to the structure of the

convolutional neural network to improve sample quality and convergence velocity. These changes include:

- Instead of using pooling layers use stride convolutions (discriminator) and fractional-stride convolutions (generator).
- Use batch normalization in both D and G.
- Remove the FC layer and make the network a fully convolutional network.
- The G ReLU network is used as the activation function, and the last layer uses tanh.
- Use LeakyReLU as the activation function in the D network.

Figure 2 shows G-Network in DCGAN [3].

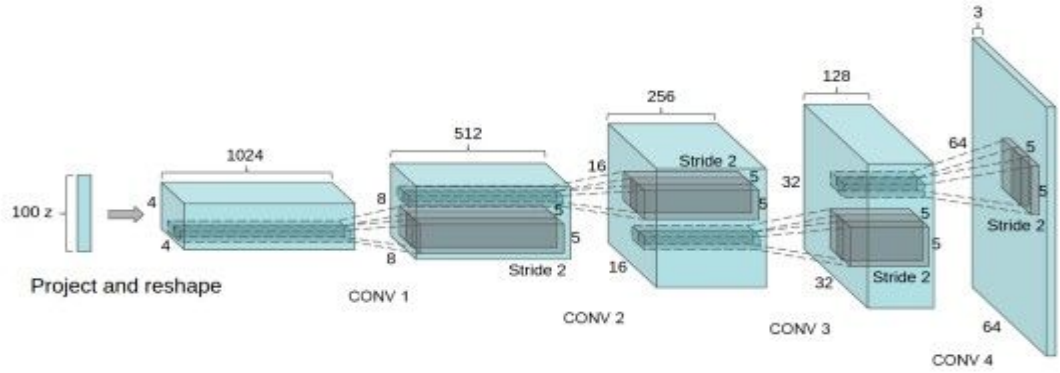

Figure 2: G-Network in DCGAN

2. Related work

several researches have been published in the study of GAN. Most of these studies done recently. Emphasis will be placed on increasing the sample size of the database, a few reviews are as follows:

Frid-Adar, Maayan, et al. [9] In this article, the authors proposed an enlargement scheme based on a combination of standard imaging and artificial liver lesions using GANs to improve the classification of liver lesions. Using the GAN, they pooled high-quality focal liver lesions from CT images and developed CNN for the liver lesion classification task and augmented the CNN training set using synthetic data created to improve classification outcomes.

The classification efficiency gave a sensitivity of 78.6% and a specificity of 88.4%. By adding the synthetic data, they obtained a sensitivity of 85.7% and a specificity of 92.4%.

Iqbal, Talha, and Hazrat Ali. [10] in this work authors propose a new GAN for Medical Images (MIGAN). The MIGAN generates synthetic retinal image with their segmented masks, which will be used to apply a controlled analysis of medical images. One of the features of the proposed model is the ability to learn features from a few dataset.

Shmelkov, Konstantin, Cordelia Schmid, and Karteek Alahari. [11] The authors suggest two measures based on the classification of images, GANTrain and GANTest, which come close to recall and precision. They introduce new assessment metrics to compare class conditional GAN structure with GANTrain and GANTest scores. They used NN architecture for the classification of images for these two measurements. Authors train their networks with generated images to calculate the GANTrain. Then they obtained performance on a test set which collected from real images.

Bowles, Christopher, et al. [12] this article explains the feasibility of injecting synthetic data obtained from the GAN into the original datasets in two brain segmentation problems, resulting in an improvement in the dice similarity rate by 1 to 5 percentage points under various conditions. They used a Progressive Growing of GANs (PGGAN) network to generate synthetic data. PGGAN was chosen, because the ability of training at large image sizes. They added the synthetic data with the real data to increase their dataset. The best result obtained when the authors used GAN + Rotation augmentation.

Zeid Baker, Mousa. [13] the researcher in this paper used Generative Adversarial Networks to generate synthetic images similar to the real dataset in order to expand a dataset. they used two types

of experiments were achieved, the first is using fine-tune a Deep Convolutional Generative Adversarial Network for a specific dataset, while the second experiment was used to analyze how synthetic data affect the accuracy in a classification task. authors used three types of datasets MNIST, Fashion-MNIST and Flower photos. The authors conclusion that DCGAN leads to an increase the model accuracy depends on the type of the dataset and the data preprocessing plays a big roll on a DCGANs performance, as for almost any ML algorithm.

Wu, Qiufeng, Yiping Chen, and Jun Meng. [14] in this work, authors used Generated images augmented by deep convolutional generative adversarial networks (DCGAN) and original images to Identify Tomato Leaf Disease. They used GoogLeNet classifier to training and testing 5 classes of tomato leaf images, this model achieved accuracy of 94.33%. the authors result that images generated by DCGAN not only enlarge the size of the data set, but also have the characteristics of diversity, which makes the model have a good generalization effect.

3. Material and method

3.1. Datasets

The dataset was analyzed and preprocessed of four different classes which contain 1692 of Glaucoma, Myopia, Diabetic retinopathy and Normal images, were selected for this study to aim for the highest variance among classes. where 1200 images used for training, 246 images for testing and 246 images used for validation purpose, which mean 15% of the total images were used for testing and 15% for validation purpose. All the images were collected in total from Kaggle dataset and iChallenge-GON Comprehension, in high resolution images.

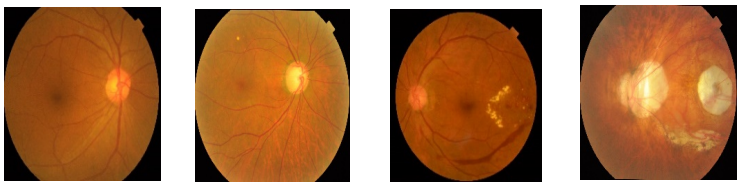

Figure 3: Images of Normal Fundus, Glaucoma, Diabetic retinopathy and Myopia

The size of all images should be the same, colored and png format. In machine learning and deep learning, accuracy of model will be affected if the samples are not equally distributed [15].

Table 1
Eye diseases samples in original dataset

Diseases	Training	Validation
Glaucoma	300	48
Diabetic retinopathy	300	66
Myopia	300	66
Normal	300	66
Total	1200	246

In addition, 300 images have been used of each class to train our model. There are two issue should be considered: The number of parameters when the number of network layers increases, and the small number of eye diseases which we have collected. That will lead to the overfit under the influence of many parameters and few datasets. Increasing of the dataset is an effective way to solve this issue. In this paper, authors proposed DCGAN network to increase eye diseases dataset by generate synthetic dataset. Therefore, the training data set, and validation set will be increased (data enhancement).

3.2. Model building

The main goal in this work is to generate images that look like realistic images in each class in order to solve the problem of insufficient medical images. in addition, using GMD model to classify eye diseases with and without DCGAN.

3.2.1. GMD model

As shown in Figure 4, the size of input image is set to 64×64 pixels with 3 RGB channels. To extract the features from the image, two convolution layers were used: the first is 256 filters of size 3×3 pixels and the second is 128 filters of size 3×3 pixels. For the pooling layer, a window of size 2×2 pixels were used, which compresses the original image size for further processing. After that, another three convolution layers were used of 128, 64, and 32 filters with size 3×3 pixels with a maximum pooling size 2×2 pixels. Then, fully connection is used (Dense 100 units and 60 units) and output layer (4 units) to predict the eye cases. CNNs adjust their filter weights through backpropagation, which means that after the forward pass, the network is able to look at the loss function and make a backward pass to update the weights.

The GMD model performance was evaluated against training, testing, and validation datasets using the accuracy measure, which reflects the ability of the model to classify the whole database into four categories with almost no error. The following represents the accuracy measure equation [16]:

$$Accuracy = TP / TP + FP + FN + TN \quad (4)$$

Where, TP is True Positive: Your predicted positive and it is true, TN is True Negative: Your predicted negative and it is true, FP is False Positive: your predicted positive but it is false and False Negative FN: Your predicted negative but it is false.

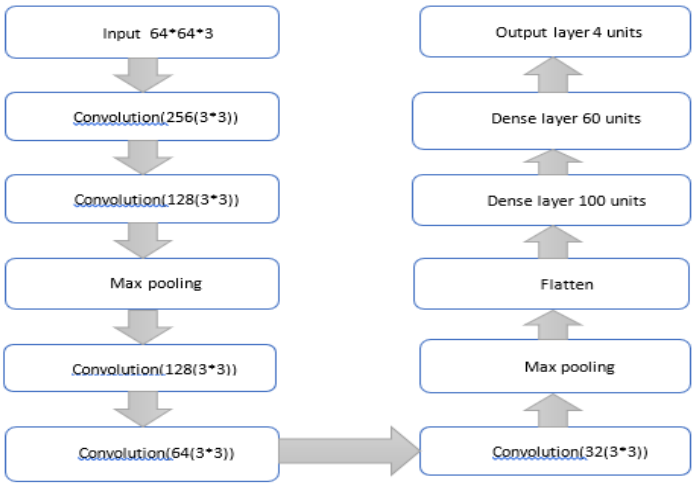

Figure 4: Block diagram of GMD model

3.2.2. DCGAN architecture

GAN consists of two separate neural networks, Generator (G) and Discriminator (D) constantly competing against each other, the generator, which tries to generate examples that seem the real data, it works to deceive the discriminator, while the role of discriminative network is to distinguish between the real and fake data. a vector of random number will be feeds to the generator, and the outputs are unreal synthetic data denoted by G (z), the discriminator feeds real data and the output of generator. The output of the discriminator is the possibility whether the entered data is real or fake, the output denoted by D (x). The output D (x) represents the probability that x will be a real image if it is 1, which means 100% is a real image, and the output is 0 which means that the image cannot be real [17].

Objective function and how GAN is work, we mentioned above in introduction in section 1. DCGAN is Deep Convolutional GAN, it only replaces the G and D with two convolutional neuronal networks (CNN). Compared with GAN, DCGAN made some changes to the CNN architecture to improve sample quality and convergence speed. These changes are mentioned above in section 1.3.1. In this paper, the block diagram of eye diseases identification is shown in Figure **5.** We also will interduce the proposed generator and discriminator network used by referring to the DCGAN architectures as shown in Figure **6**:

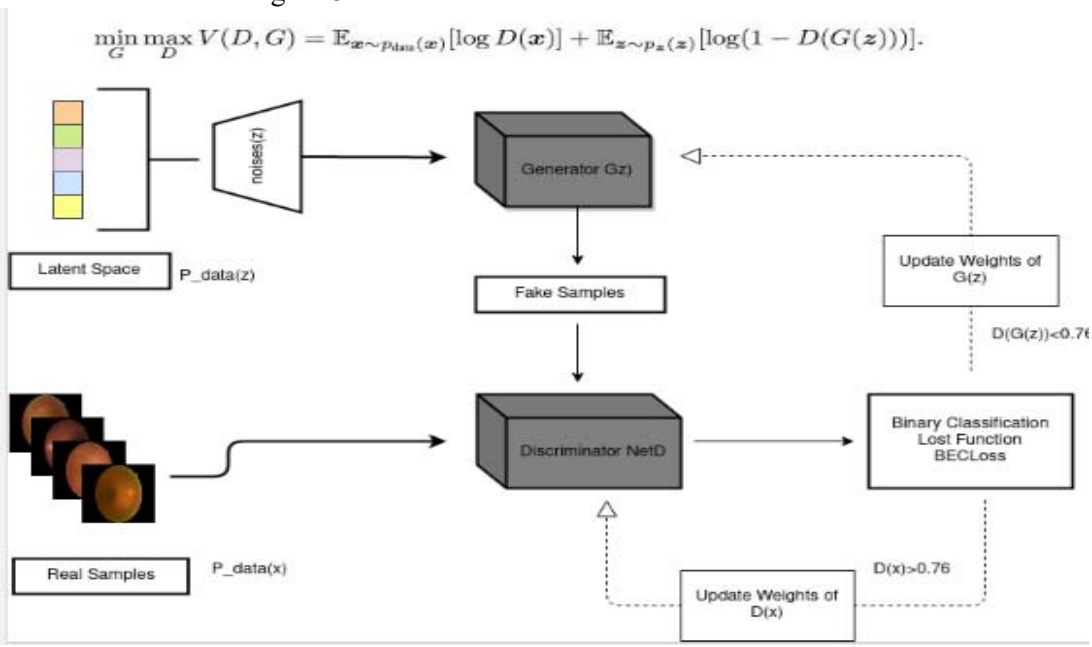

Figure 5: Block diagram of eye diseases identification

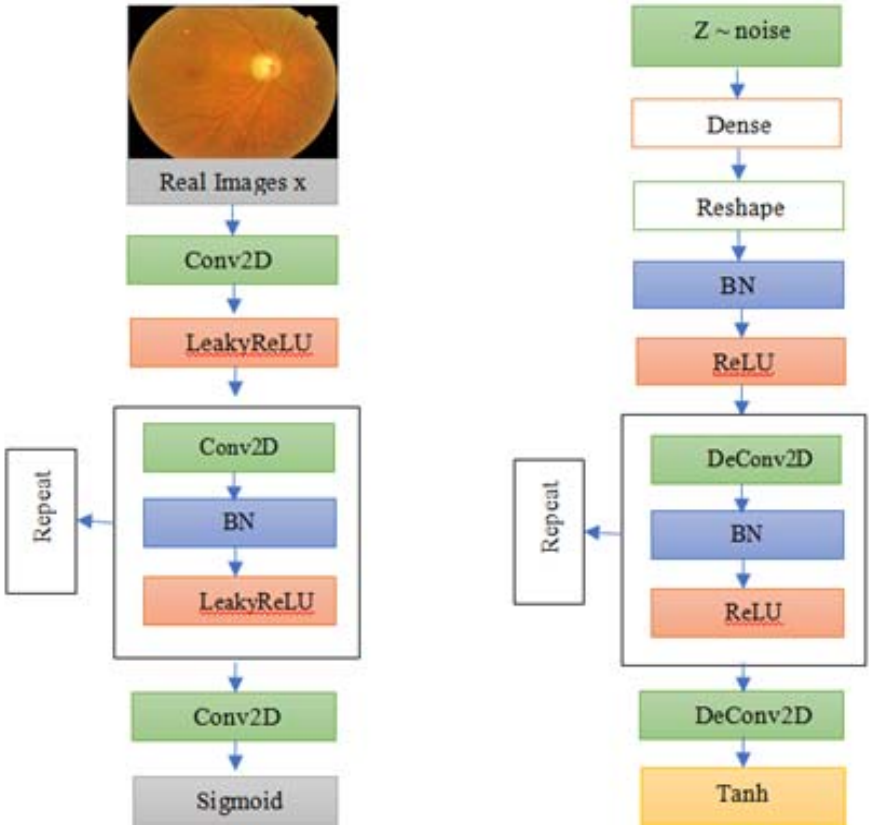

Figure 6: discriminator (left) and generator architecture (right)

4. Experiments and Results

According to the models explained above, all these models were implemented by Python language (Google Colab) using dual Graphics Processing Unit (GPU). The main objective of this paper is to increase our dataset using DCGAN. These data will be feds to GMD model in order to improve eye diseases classification. Accuracy will be measured before and after adding synthetic dataset.
Dataset has been explained in section 3.1. it consists of four types of eye diseases, Glaucoma, Diabetic retinopathy, Myopia and Normal eyes. Table 1 shows the number of samples of each class in training set and validation set.

4.1. DCGAN results

DCGAN is to increase our dataset in order to improve the diagnosis of eye diseases. The schema of our experiment as shown in Figure 5, has been applied. We utilized our dataset which consists of 1200 training samples, 246 validation samples of eye diseases size 64x64 coloured images for four general classes. We used the same network structure for the generator and discriminator as shown in figure 6, and training our medical images, the parameters of the discriminator and generator are updated in the same number of iterations. During this process BCElose will be calculated on all real and fake batch, and calculate gradients for D and G in backward propagation.

```
# (1) Update D network: maximize log(D(x)) + log(1 - D(G(z)))
## Train with all-real batch
netD.zero_grad()
# Format batch
 real_cpu = data[0].to(device)[:1]
b_size = real_cpu.size(0)
label = torch.full((b_size,), real_label, dtype=torch.float, de
vice=device)
# Forward pass real batch through D
output = netD(real_cpu).view(-1)
# Calculate loss on all-real batch
errD_real = criterion(output, label)
# Calculate gradients for D in backward pass
errD_real.backward()
D_x = output.mean().item()
## Train with all-fake batch
# Generate batch of latent vectors
noise = torch.randn(b_size, nz, 1, 1, device=device)
# Generate fake image batch with G
fake = netG(noise)
label.fill_(fake_label)
# Classify all fake batch with D
output = netD(fake.detach().to(device)[:1]).view(-1)
# Calculate D's loss on the all-fake batch
 errD_fake = criterion(output, label)
 # Calculate the gradients for this batch
errD_fake.backward()
D_G_z1 = output.mean().item()
# Add the gradients from the all-real and all-fake batches
errD = errD_real + errD_fake
# Update D
optimizerD.step()
# (2) Update G network: maximize log(D(G(z)))
netG.zero_grad()
label.fill_(real_label)  # fake labels are real for generator c
      ost
# Since we just updated D, perform another forward pass of all-
fake batch through D
output = netD(fake.to(device)[:1]).view(-1)
# Calculate G's loss based on this output
errG = criterion(output, label)
# Calculate gradients for G
errG.backward()
D_G_z2 = output.mean().item()
# Update G
optimizerG.step()
```

For the discriminant, the label for the true sample is marked with the number 1, and the label for the generated sample is marked as 0, regardless of the quality of the sample generated. So, D(x)

should be close to 1 and D(G(z)) should be close to 0. In our experiment we save just the images that D(G(z))>=0.76 which close to 1.

```
x=0.76
if errD_fake >= float(x) or errD_fake == errD_real :
vutils.save_image(vutils.make_grid(fake[:1].to(device)[:1], padding=2, normalize=True), "%s/fake_samples_epoch_%03d.png" % ("/content/drive/MyDrive/GAN2/FakeImages", epoch), normalize = False)
```

Adam was used as the gradient method for learning parameters of model. Its initial learning rate is 0002.

```
optimizerD = optim.Adam(netD.parameters(), lr=lr, betas=(beta1, 0.999))
optimizerG = optim.Adam(netG.parameters(), lr=lr, betas=(beta1, 0.999))
```

The number of epochs we have used is 4000, that is not mean 4000 images will be generated, because we save the images that D(G(z))>=0.76. Table 2 shows the synthetic images and total images which we obtained. Figure 7 shows samples of the fake and real images.

Table 2
Eye diseases samples after using DCGAN

Diseases	synthetic data using DCGAN	Original data + synthetic data	
		Training	Validation
Glaucoma	318	618	48
Diabetic retinopathy	294	594	66
Myopia	428	728	66
Normal	306	606	66
Total	1346	2546	246

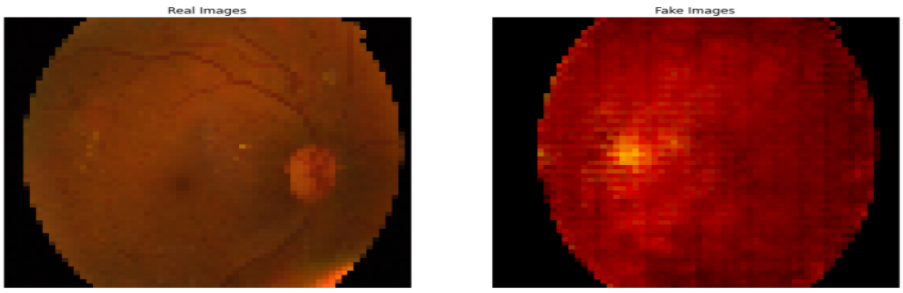

Figure 7: sample of real and fake image

The figures below show the loss during training in both Generator and Discriminator for all classes:

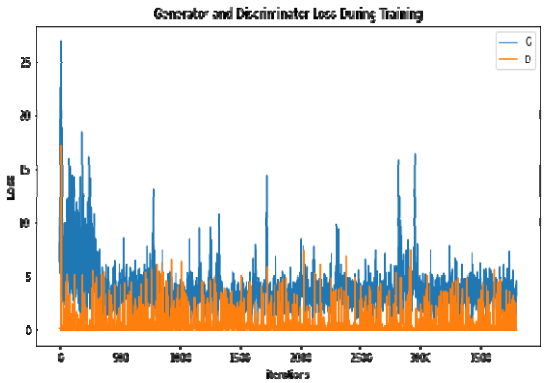

Figure 8: Loss in Glaucoma

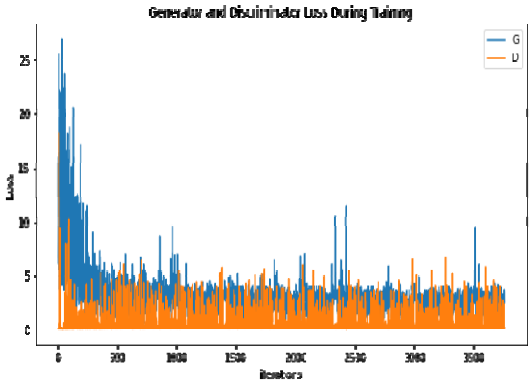

Figure 9: Loss in Diabetic Retonapthy

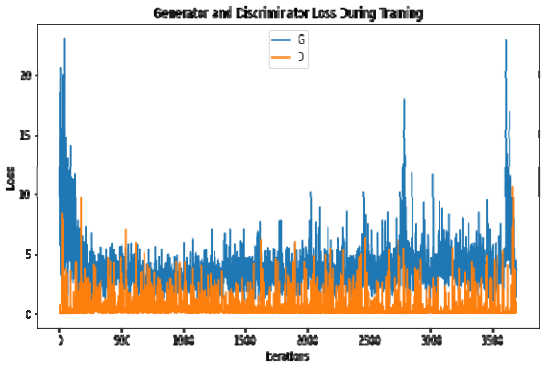

Figure 10: Loss in Myopia

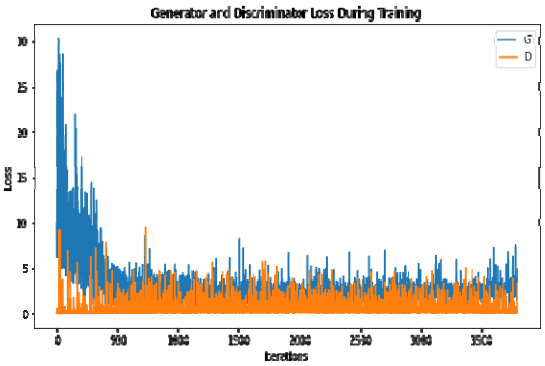

Figure 11: Loss in Normal

4.2. GMD model Results with Original data

GMD model has been applied, Constant learning rate is the default learning rate schedule in SGD. Both momentum and decay rate are set to zero. Learning rate was set to 0.001 which showed a good performance to start. In constant learning rate schedule, the model was defined and the learning rate value was set to constant as follows.

 # define CNN_ model
model1 = CNN_model()
epochs = 100
 # define SGD optimizer and set to default except lr
learning_rate = 0.001
sgd = SGD(lr=learning_rate, momentum=0.0, decay=0.0, nesterov=False).

Then, compiling, fitting GMD model and plotting the model accuracy as shown in Figure.12. We obtained Accuracy on training set 76.58% and Accuracy on validation set 76.42%.

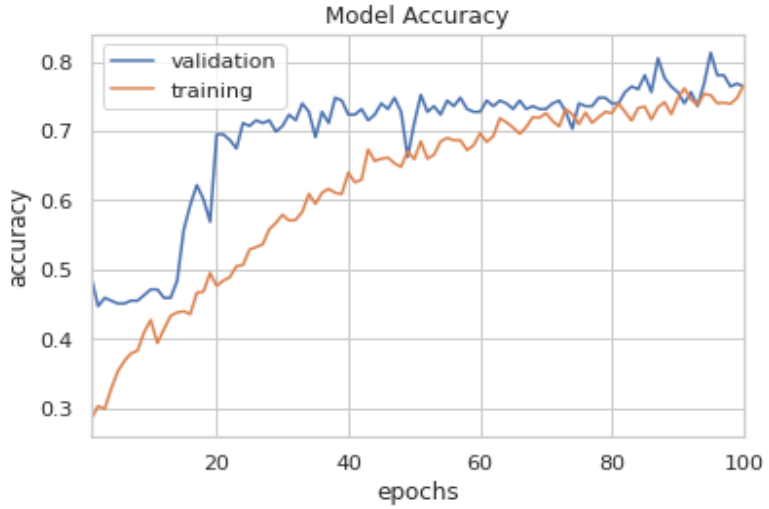

Figure 12: model accuracy using constant learning rate without data generated

4.3. GMD model Results with synthetic data

The model has been applied and constant learning rate is the default learning rate schedule in SGD as mentioned in paragraph 4.1. Then, compiling, fitting our model and plotting the model accuracy as shown in Figure.13. The result obtained, accuracy on training set 80.45% and accuracy on validation set 83.74%.

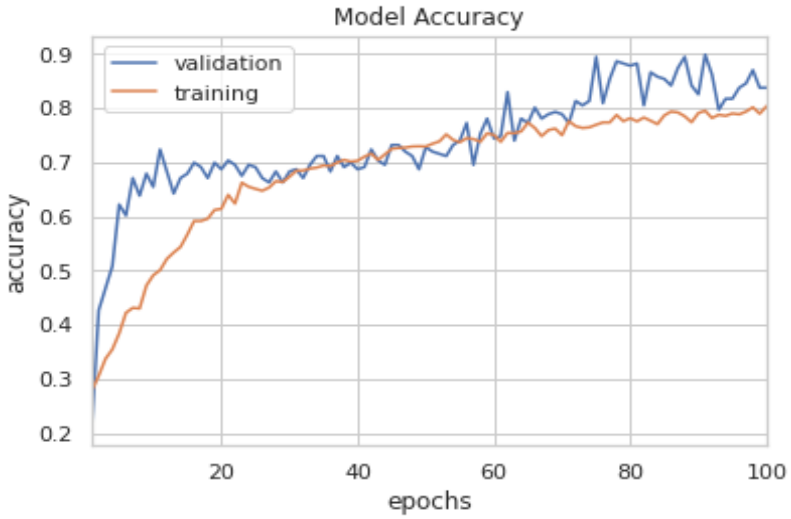

Figure 13: model accuracy using constant learning rate with data generated

5. Conclusion

In this research, a Deep Convolutional Generative Adversarial Network, and GMD model based on Keras, TensorFlow and pycharm has been deployed using Python on a database of four types of ophthalmic cases: Glaucoma, Myopia, Diabetic retinopathy, and Normal eyes.
The implemented DCGAN and GMD model have been trained using GPU in Google Colab, in order to increase our dataset to improve eye diseases classification.

In conclusion, DCGAN strongly can be used for generating high quality of medical images and increasing training data. Furthermore, the performance of GMD model with original data and with synthetic images data were compared. We confirmed that with the help of DCGAN, GMD model (CNN model) can yield a better *accuracy* compared to traditional methods. Moreover, the *accuracy* of the model had improved significantly from 76.58% in training set and 76.42% in validation set, to 80.45% in training set and 83.74% in validation set. Based on this work, we suggest that DCGAN can be optimized for multi-class generator. We also suggest that this work can be applied to other image classification models such as Vgg16, Inception v3, ResNet to enhance the accuracy.

We noticed that the images we have obtained from DCGAN had some noise. Therefore, in the next paper we will use denoising Autoencoder to enhance these images to improve the model accuracy.

6. References

[1] H. Greenspan, B. van Ginneken, and R. M. Summers, “Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique,” IEEE Transactions on Medical Imaging, vol. 35, no. 5, pp. 1153–1159, May 2016.

[2] Mustaffa Hussain, GANs-What and Where?, 2020. URL: https://medium.com/thecyphy/gans-what-and-where-b377672283c5.

[3] Goodfellow, Ian J., et al. "Generative adversarial networks." *arXiv preprint arXiv:1406.2661* (2014).

[4] Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." *arXiv preprint arXiv:1511.06434* (2015).

[5] Mirza, Mehdi, and Simon Osindero. "Conditional generative adversarial nets." arXiv preprint arXiv:1411.1784 (2014).
[6] Mao, Xudong, et al. "Least squares generative adversarial networks." Proceedings of the IEEE international conference on computer vision. 2017.
[7] Berthelot, David, Thomas Schumm, and Luke Metz. "Began: Boundary equilibrium generative adversarial networks." arXiv preprint arXiv:1703.10717 (2017).
[8] Martín Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein Generative Adversarial Networks. International Conference on Machine Learning, ICML, 2017.
[9] Frid-Adar, Maayan, et al. "Synthetic data augmentation using GAN for improved liver lesion classification." *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*. IEEE, 2018.
[10] Iqbal, Talha, and Hazrat Ali. "Generative adversarial network for medical images (MI-GAN)." *Journal of medical systems* 42.11 (2018): 1-11.
[11] Shmelkov, Konstantin, Cordelia Schmid, and Karteek Alahari. "How good is my GAN?." Proceedings of the European Conference on Computer Vision (ECCV). 2018.
[12] Bowles, Christopher, et al. "Gan augmentation: Augmenting training data using generative adversarial networks." arXiv preprint arXiv:1810.10863 (2018).
[13] Zeid Baker, Mousa. "Generation of Synthetic Images with Generative Adversarial Networks." (2018).
[14] Wu, Qiufeng, Yiping Chen, and Jun Meng. "DCGAN-based data augmentation for tomato leaf disease identification." IEEE Access 8 (2020): 98716-98728.
[15] F. J. Pulgar, A. J. Rivera, F. Charte, and M. J. del Jesus, ‘‘On the impact of imbalanced data in convolutional neural networks performance,’’ in Proc. Int. Conf. Hybrid Artif. Intell. Syst., vol. 10334. Cham, Switzerland: Springer, 2017, pp. 220–232.
[16] Visa, Sofia, et al. "Confusion Matrix-based Feature Selection." MAICS 710 (2011): 120-127.
[17] M. Mirza and S. Osindero, ‘‘Conditional generative adversarial nets,’’ 2014, arXiv:1411.1784. [Online]. Available: http://arxiv.org/abs/ 1411.1784
[18] Kong, Quan, et al. "Active generative adversarial network for image classification." Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 33. No. 01. 2019.
[19] Khalifa, Nour Eldeen M., et al. "Detection of coronavirus (covid-19) associated pneumonia based on generative adversarial networks and a fine-tuned deep transfer learning model using chest x-ray dataset." arXiv preprint arXiv:2004.01184 (2020).