Digital Egypt Youth

Internet of things (IoT) and artificial Intelligent (AI)

شباب مصر الرقمية

Classification of Retina Diseases using Convolution neural network (CNN)

(NORMAL – DRUSEN – CNV - DME)

By

Ahmed Mohamed Abd El-ghany Ahmed

Ahmed Adel Abd El-hakeem Abd El-mohsen

Mohammed Ahmed El-desoukey El-abasiry

Sami Abdelaziz Abdelhamid Ali Selim

Ahmed Abdullah Kamal

Submitted to

Dr. Mohammed Al-Zaraqani

Dr. Ghazal A. Fahmy

Acknowledgement

We cannot express enough thanks to **Dr. Ghazal A. Fahmy** and **Dr. Mohamed Al-Zaraqani** They understood the nature of our thoughts and guided us step by step until this work brought to light. Their professional touches are sensed within every phase of this course.

Abstract

Artificial intelligence is one of the most important sciences that serve humanity in many fields, and among these many fields is the medical field, where artificial intelligence is used in the diagnosis of some medical diseases in order to reduce some errors and relieve some of the burdens that fall on the shoulders of doctors, so we did this project with the aim of helping in diagnosing retinal diseases using an artificial intelligence application.

In this report we fitted a different practical model of common types of Convolutional neural network (CNN) such like:

- LeNet
- VGGNet 16
- VGGNet 19
- ResNet
- DenesNet

Table of Contents:

List of figures:

List of Tables:

Introduction

1.1 What is OCT?

Optical Coherence Tomography (OCT) is a non-invasive diagnostic technique that renders an in vivo cross-sectional view of the retina. OCT utilizes a concept known as inferometry to create a cross-sectional map of the retina that is accurate to within at least 10-15 microns. OCT was first introduced in 1991 and has found many uses outside of ophthalmology, where it has been used to image certain non-transparent tissues. Due to the transparency of the eye (i.e., the retina can be viewed through the pupil), OCT has gained wide popularity as an ophthalmic diagnostic tool.

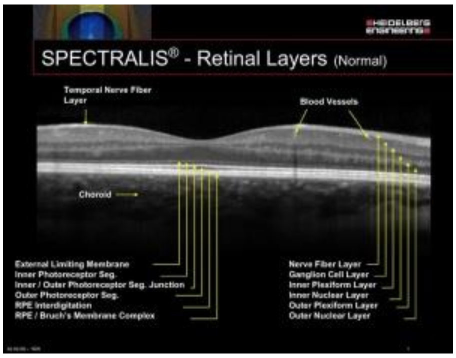


Figure 1 OCT Image of Retinal Layers (Courtesy of Heidelberg Engineering)

OCT has attracted interest among the medical community because it provides tissue morphology imagery at much higher resolution (less than 10 µm axially and less than 20 µm laterally) than other imaging modalities such as MRI or ultrasound.

OCT can be particularly helpful in diagnosing:

- Macular hole
- Macular pucker/epiretinal membrane
- Vitreomacular traction
- Macular edema and exudates
- Detachments of the neurosensory retina
- Detachments of the retinal pigment epithelium (e.g. central serous retinopathy or age-related macular degeneration)
- Retinoschisis
- Pachychoroid
- Choroidal tumors

In some cases, OCT alone may yield the diagnosis (e.g. macular hole). Yet, in other disorders, especially retinal vascular disorders, it may be helpful to order additional tests (e.g. fluorescein angiography or indocyanine green angiography).

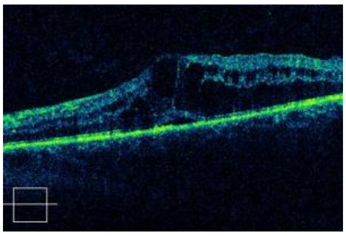


Figure 2 OCT showing both macular edema and subretinal fluid in a diabetic patient

1.2 Eye Diseases:

Optometrists use Optical Coherence Tomography (OCT) of the human eye to analyze and detect various age-related eye abnormalities such as Choroidal Neovascularization (CNV), Diabetic Macular Edema (DME), Drusen and the anatomy of normal eye as shown in the figures 3 ,4,5

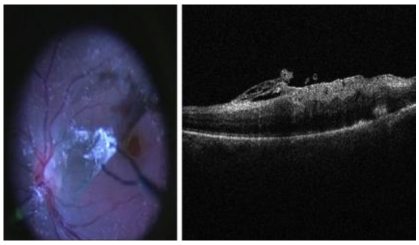


Figure 3 output of OCT figure of the anatomy of eye

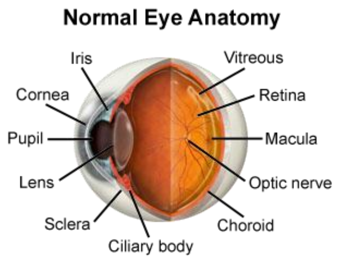


Figure 4 anatomy of normal eye

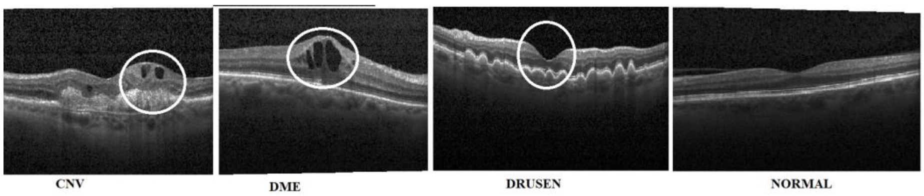


Figure 5 Difference Eye Diseases

1.3 Choroidal Neovascularization

Choroidal neovascularization (CNV) is considered as a part of the spectrum of exudative age-related macular degeneration (AMD) that contains of an abnormal growth of blood-vessel from the choroidal to the neurosensory retina through the Bruch's membrane. Hemorrhage from CNV in AMD menaces visual intensity.

1.4 Diabetic Macular Edema

Diabetic Macular Edema (DME) is a gathering of fluid in the macula part in the retina that controls vision capabilities. Will appear DME, after patients have had diabetic retinopathy. Diabetic retinopathy (DR) is a disease that damages the blood vessels in the retina, causing of vision frailty. Without treatment these blood vessels, the leak will fluid in the eye and, causing of DME. DME usually consists of two types:
Focal DME, which happens because of abnormalities in the blood vessels in the eye. Diffuse DME, which happens because of widening or swelling retinal capillaries.

1.5 DRUSEN

Drusen are yellow sediments under the retina. Drusen are consisted of grease which is a fatty protein. Drusen probable do not cause age-related macular degeneration (AMD). But Druse raises a person's risk of causing of AMD.There are different types of drusen. "Hard" drusen may not occasion vision problems for a long time. They are small, independent, and far away from one to another. "Soft" drusen are large and grouped closer together. Their borders are not as clearly defined as hard drusen. This type of drusen increases the development of AMD.

1.6 Convolutional neural network (CNN):

Convolutional neural network is a class of deep learning methods which has become dominant in various computer vision tasks and is attracting interest across a variety of domains, including radiology. CNN is composed of multiple building blocks, such as convolution layers, pooling layers, and fully connected layers, and is designed to automatically and adaptively learn spatial hierarchies of features through a backpropagation algorithm.

1.7 Common types of CNN

The various types of CNN, designed and implemented successfully in various fields of image processing and object recognition. The common types of CNN are:

- LeNet
- AlexNet
- VGGNet 16
- VGGNet 19
- ResNet
- DenesNet

1.8 Dataset Information:

We used standard dataset of 8,000 image classified into 3 categories:

	Normal	**DRUSEN**	**CNV**	**DME**
Training	1500	1500	1500	1500
Testing	250	250	250	250
Validation	250	250	250	250

Table 1 Dataset details

- Dataset link: https://www.dropbox.com/s/1b4l87j3vcc02ti/OCT2019.zip

Different practical models of CNN algorithms

2.1 LeNet:

We fitted two standard models with batch size=64, 100 Epoch and different image sizes and optimizers:

➢ 2.1.1 LeNet Model with image size (150*150) and optimizer ('RMSprop')

```
Epoch 92/100
93/93 [==============================] - 48s 509ms/step - loss: 0.8225 - accuracy: 0.6538 - val_loss: 1.2825 - val_accuracy: 0.4930
Epoch 93/100
93/93 [==============================] - 48s 508ms/step - loss: 0.8140 - accuracy: 0.6613 - val_loss: 1.1086 - val_accuracy: 0.5240
Epoch 94/100
93/93 [==============================] - 47s 505ms/step - loss: 0.8331 - accuracy: 0.6498 - val_loss: 1.0832 - val_accuracy: 0.5530
Epoch 95/100
93/93 [==============================] - 48s 507ms/step - loss: 0.8072 - accuracy: 0.6611 - val_loss: 1.1018 - val_accuracy: 0.5440
Epoch 96/100
93/93 [==============================] - 47s 507ms/step - loss: 0.8106 - accuracy: 0.6779 - val_loss: 1.2703 - val_accuracy: 0.4940
Epoch 97/100
93/93 [==============================] - 47s 506ms/step - loss: 0.8021 - accuracy: 0.6544 - val_loss: 1.2422 - val_accuracy: 0.4980
Epoch 98/100
93/93 [==============================] - 48s 509ms/step - loss: 0.8047 - accuracy: 0.6608 - val_loss: 1.2438 - val_accuracy: 0.4970
Epoch 99/100
93/93 [==============================] - 47s 507ms/step - loss: 0.7973 - accuracy: 0.6593 - val_loss: 1.0907 - val_accuracy: 0.5260
Epoch 100/100
93/93 [==============================] - 48s 507ms/step - loss: 0.8046 - accuracy: 0.6590 - val_loss: 1.0961 - val_accuracy: 0.5400
```

```
test acc: 0.746999979019165
test loss: 0.6334831714630127
```

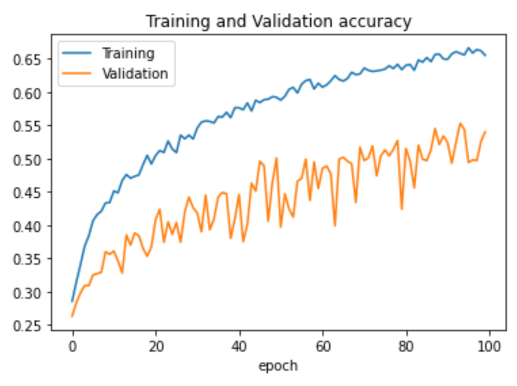
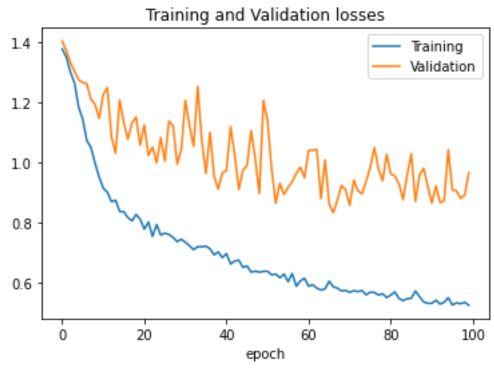


*Table 2 LeNet Model with image size (150*150) and optimizer ('RMSprop')*

- 2.1.2 LeNet Model with image size (224*224) and optimizer ('Adam')

```
Epoch 92/100
93/93 [==============================] - 82s 873ms/step - loss: 0.5573 - accuracy: 0.7750 - val_loss: 0.9233 - val_accuracy: 0.6390
Epoch 93/100
93/93 [==============================] - 82s 876ms/step - loss: 0.5106 - accuracy: 0.7898 - val_loss: 0.8673 - val_accuracy: 0.6570
Epoch 94/100
93/93 [==============================] - 82s 874ms/step - loss: 0.5282 - accuracy: 0.7799 - val_loss: 0.8740 - val_accuracy: 0.6570
Epoch 95/100
93/93 [==============================] - 82s 872ms/step - loss: 0.5470 - accuracy: 0.7847 - val_loss: 1.0430 - val_accuracy: 0.5930
Epoch 96/100
93/93 [==============================] - 82s 880ms/step - loss: 0.5372 - accuracy: 0.7906 - val_loss: 0.9092 - val_accuracy: 0.6630
Epoch 97/100
93/93 [==============================] - 82s 875ms/step - loss: 0.5307 - accuracy: 0.7910 - val_loss: 0.9062 - val_accuracy: 0.6520
Epoch 98/100
93/93 [==============================] - 82s 878ms/step - loss: 0.5249 - accuracy: 0.7914 - val_loss: 0.8813 - val_accuracy: 0.6700
Epoch 99/100
93/93 [==============================] - 82s 876ms/step - loss: 0.5455 - accuracy: 0.7820 - val_loss: 0.8923 - val_accuracy: 0.6560
Epoch 100/100
93/93 [==============================] - 82s 876ms/step - loss: 0.5308 - accuracy: 0.7920 - val_loss: 0.9666 - val_accuracy: 0.6270
```

```
test acc: 0.8130000233650208
test loss: 0.6019443273544312
```



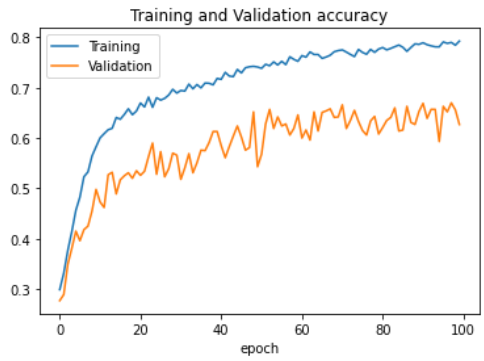


*Table 3LeNet Model with image size (224*224) and optimizer ('Adam')*

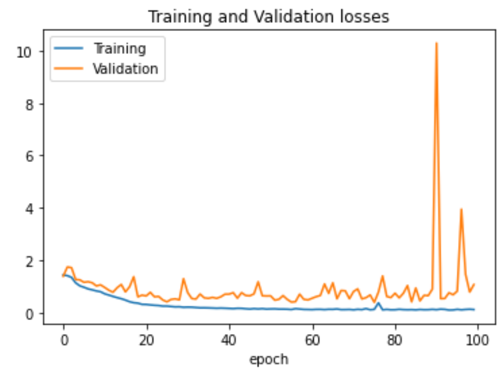
2.2 VGGNet 16

We fitted two standard models with batch size=64, image sizes (224*224) and different optimizers and Epochs:

- 2.2.1 VGGNet 16 Model with optimizer ('RMSprop') and (100) Epochs

```
Epoch 92/100
93/93 [==============================] - 117s 1s/step - loss: 0.1253 - accuracy: 0.9687 - val_loss: 0.5285 - val_accuracy: 0.8720
Epoch 93/100
93/93 [==============================] - 117s 1s/step - loss: 0.1083 - accuracy: 0.9725 - val_loss: 0.5344 - val_accuracy: 0.8770
Epoch 94/100
93/93 [==============================] - 117s 1s/step - loss: 0.0902 - accuracy: 0.9758 - val_loss: 0.7539 - val_accuracy: 0.8750
Epoch 95/100
93/93 [==============================] - 117s 1s/step - loss: 0.0883 - accuracy: 0.9718 - val_loss: 0.6759 - val_accuracy: 0.8810
Epoch 96/100
93/93 [==============================] - 116s 1s/step - loss: 0.1319 - accuracy: 0.9620 - val_loss: 0.8147 - val_accuracy: 0.8890
Epoch 97/100
93/93 [==============================] - 116s 1s/step - loss: 0.0944 - accuracy: 0.9759 - val_loss: 3.9481 - val_accuracy: 0.8570
Epoch 98/100
93/93 [==============================] - 116s 1s/step - loss: 0.1279 - accuracy: 0.9684 - val_loss: 1.4593 - val_accuracy: 0.8740
Epoch 99/100
93/93 [==============================] - 116s 1s/step - loss: 0.1120 - accuracy: 0.9667 - val_loss: 0.7799 - val_accuracy: 0.8530
Epoch 100/100
93/93 [==============================] - 117s 1s/step - loss: 0.0986 - accuracy: 0.9685 - val_loss: 1.0652 - val_accuracy: 0.8640
```

test acc: 0.9829999804496765
test loss: 0.06923840939998627



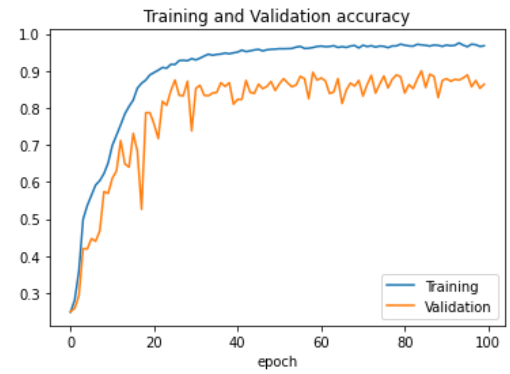
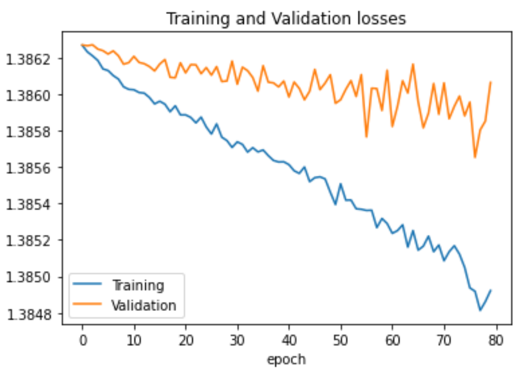


Table 4 VGGNet 16 Model with optimizer ('RMSprop') and (100) Epochs

➢ 2.2.2 VGGNet 16 Model with optimizer ('SGD')

```
Epoch 72/80
93/93 [==============================] - 139s 1s/step - loss: 1.3851 - accuracy: 0.3097 - val_loss: 1.3859 - val_accuracy: 0.2970
Epoch 73/80
93/93 [==============================] - 138s 1s/step - loss: 1.3851 - accuracy: 0.3112 - val_loss: 1.3859 - val_accuracy: 0.2740
Epoch 74/80
93/93 [==============================] - 139s 1s/step - loss: 1.3851 - accuracy: 0.3095 - val_loss: 1.3860 - val_accuracy: 0.2890
Epoch 75/80
93/93 [==============================] - 138s 1s/step - loss: 1.3850 - accuracy: 0.3016 - val_loss: 1.3859 - val_accuracy: 0.3060
Epoch 76/80
93/93 [==============================] - 138s 1s/step - loss: 1.3850 - accuracy: 0.3195 - val_loss: 1.3860 - val_accuracy: 0.2870
Epoch 77/80
93/93 [==============================] - 138s 1s/step - loss: 1.3849 - accuracy: 0.3185 - val_loss: 1.3857 - val_accuracy: 0.2990
Epoch 78/80
93/93 [==============================] - 138s 1s/step - loss: 1.3849 - accuracy: 0.3229 - val_loss: 1.3858 - val_accuracy: 0.2740
Epoch 79/80
93/93 [==============================] - 138s 1s/step - loss: 1.3850 - accuracy: 0.3041 - val_loss: 1.3859 - val_accuracy: 0.2940
Epoch 80/80
93/93 [==============================] - 138s 1s/step - loss: 1.3850 - accuracy: 0.3134 - val_loss: 1.3861 - val_accuracy: 0.2850
```

test acc: 0.3799999952316284
test loss: 1.384906291961670



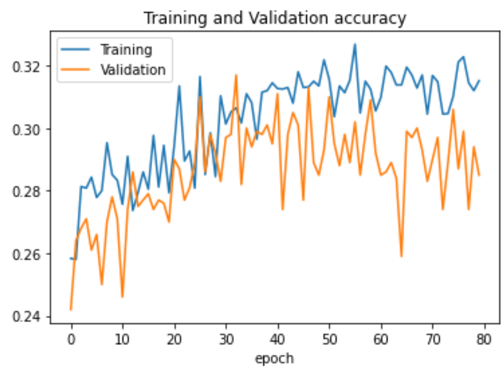


Table 5 VGGNet 16 Model with optimizer ('SGD')

2.3 VGGNet 19

We fitted four models with batch size=64, image sizes (224*224) and different optimizers:

➢ 2.3.1 VGGNet 19 Model with optimizer ('RMSprop')

```
Epoch 92/100
93/93 [==============================] - 113s 1s/step - loss: 0.0921 - accuracy: 0.9721 - val_loss: 0.3912 - val_accuracy: 0.8970
Epoch 93/100
93/93 [==============================] - 114s 1s/step - loss: 0.1111 - accuracy: 0.9687 - val_loss: 8.6112 - val_accuracy: 0.6650
Epoch 94/100
93/93 [==============================] - 114s 1s/step - loss: 0.1769 - accuracy: 0.9596 - val_loss: 0.5018 - val_accuracy: 0.8930
Epoch 95/100
93/93 [==============================] - 114s 1s/step - loss: 0.4306 - accuracy: 0.9694 - val_loss: 0.7405 - val_accuracy: 0.8720
Epoch 96/100
93/93 [==============================] - 114s 1s/step - loss: 0.0899 - accuracy: 0.9717 - val_loss: 0.4639 - val_accuracy: 0.8590
Epoch 97/100
93/93 [==============================] - 114s 1s/step - loss: 0.0902 - accuracy: 0.9724 - val_loss: 0.8594 - val_accuracy: 0.8820
Epoch 98/100
93/93 [==============================] - 114s 1s/step - loss: 0.1144 - accuracy: 0.9668 - val_loss: 0.7321 - val_accuracy: 0.8870
Epoch 99/100
93/93 [==============================] - 114s 1s/step - loss: 0.1136 - accuracy: 0.9701 - val_loss: 0.7258 - val_accuracy: 0.8880
Epoch 100/100
93/93 [==============================] - 114s 1s/step - loss: 0.2657 - accuracy: 0.9528 - val_loss: 0.5345 - val_accuracy: 0.8760
```

```
test acc:  0.9850000143051147
test loss:  0.054889362305402756
```

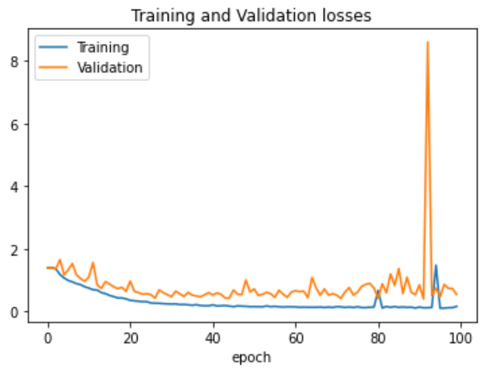
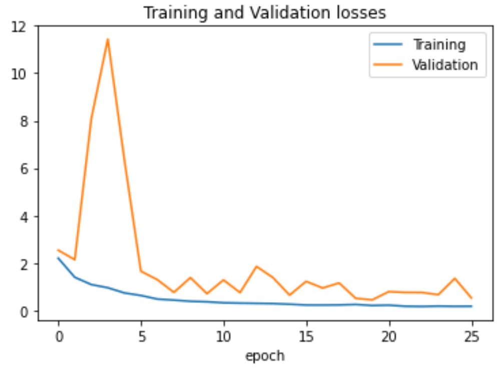


Table 6 VGGNet 19 Model with optimizer ('RMSprop')

➢ 2.3.2 VGGNet 19 Model with optimizer ('RMSprop') , dropout (0.5) and using early stop

```
Epoch 19/100
93/93 [==============================] - 123s 1s/step - loss: 0.3027 - accuracy: 0.9248 - val_loss: 0.5104 - val_accuracy: 0.8500
Epoch 20/100
93/93 [==============================] - 123s 1s/step - loss: 0.1811 - accuracy: 0.9407 - val_loss: 0.4449 - val_accuracy: 0.8530
Epoch 21/100
93/93 [==============================] - 123s 1s/step - loss: 0.2198 - accuracy: 0.9402 - val_loss: 0.7906 - val_accuracy: 0.7900
Epoch 22/100
93/93 [==============================] - 123s 1s/step - loss: 0.1626 - accuracy: 0.9498 - val_loss: 0.7599 - val_accuracy: 0.8160
Epoch 23/100
93/93 [==============================] - 123s 1s/step - loss: 0.1664 - accuracy: 0.9488 - val_loss: 0.7572 - val_accuracy: 0.7960
Epoch 24/100
93/93 [==============================] - 123s 1s/step - loss: 0.1990 - accuracy: 0.9460 - val_loss: 0.6638 - val_accuracy: 0.8030
Epoch 25/100
93/93 [==============================] - 124s 1s/step - loss: 0.1733 - accuracy: 0.9505 - val_loss: 1.3497 - val_accuracy: 0.6690
Epoch 26/100
93/93 [==============================] - 126s 1s/step - loss: 0.1833 - accuracy: 0.9473 - val_loss: 0.5260 - val_accuracy: 0.8520
```

```
test acc: 0.972000002861023
test loss: 0.09611707925796509
```



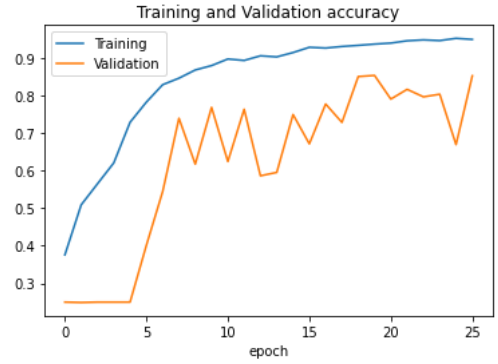


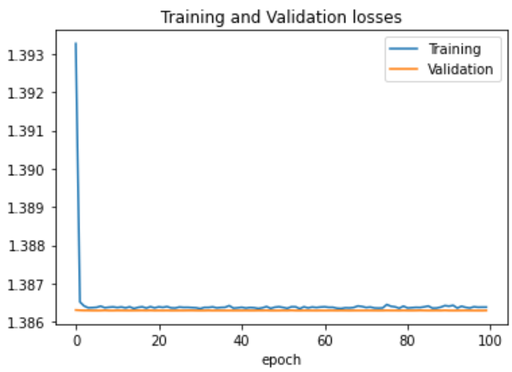
Table 7 VGGNet 19 Model with optimizer ('RMSprop') , dropout (0.5) and using early stop

---

➢ 2.3.3 VGGNet 19 Model with optimizer ('Adam')

```
Epoch 92/100
93/93 [==============================] - 114s 1s/step - loss: 1.3864 - accuracy: 0.2465 - val_loss: 1.3863 - val_accuracy: 0.2500
Epoch 93/100
93/93 [==============================] - 113s 1s/step - loss: 1.3863 - accuracy: 0.2422 - val_loss: 1.3863 - val_accuracy: 0.2500
Epoch 94/100
93/93 [==============================] - 114s 1s/step - loss: 1.3863 - accuracy: 0.2587 - val_loss: 1.3863 - val_accuracy: 0.2500
Epoch 95/100
93/93 [==============================] - 113s 1s/step - loss: 1.3863 - accuracy: 0.2508 - val_loss: 1.3863 - val_accuracy: 0.2500
Epoch 96/100
93/93 [==============================] - 113s 1s/step - loss: 1.3863 - accuracy: 0.2439 - val_loss: 1.3863 - val_accuracy: 0.2500
Epoch 97/100
93/93 [==============================] - 113s 1s/step - loss: 1.3863 - accuracy: 0.2388 - val_loss: 1.3863 - val_accuracy: 0.2500
Epoch 98/100
93/93 [==============================] - 113s 1s/step - loss: 1.3864 - accuracy: 0.2491 - val_loss: 1.3863 - val_accuracy: 0.2500
Epoch 99/100
93/93 [==============================] - 113s 1s/step - loss: 1.3863 - accuracy: 0.2573 - val_loss: 1.3863 - val_accuracy: 0.2500
Epoch 100/100
93/93 [==============================] - 113s 1s/step - loss: 1.3863 - accuracy: 0.2428 - val_loss: 1.3863 - val_accuracy: 0.2500
```

```
test acc: 0.25
test loss: 1.3862971067428589
```



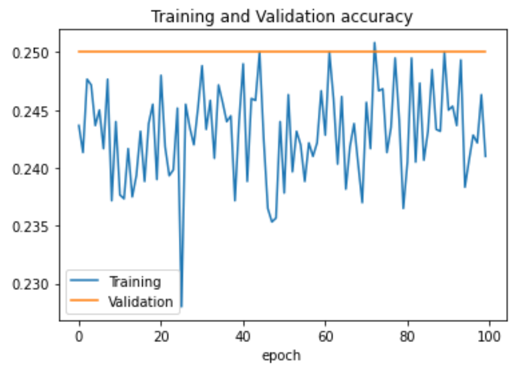


Table 8 VGGNet 19 Model with optimizer ('Adam')

- 2.3.4 VGGNet 19 Model with optimizer ('SGD') and dropout (0.5)

```
Epoch 92/100
93/93 [==============================] - 96s 1s/step - loss: 1.3851 - accuracy: 0.3120 - val_loss: 1.3861 - val_accuracy: 0.2860
Epoch 93/100
93/93 [==============================] - 96s 1s/step - loss: 1.3851 - accuracy: 0.3067 - val_loss: 1.3861 - val_accuracy: 0.2760
Epoch 94/100
93/93 [==============================] - 96s 1s/step - loss: 1.3851 - accuracy: 0.3078 - val_loss: 1.3860 - val_accuracy: 0.2820
Epoch 95/100
93/93 [==============================] - 96s 1s/step - loss: 1.3850 - accuracy: 0.3197 - val_loss: 1.3861 - val_accuracy: 0.2820
Epoch 96/100
93/93 [==============================] - 95s 1s/step - loss: 1.3852 - accuracy: 0.3115 - val_loss: 1.3861 - val_accuracy: 0.2870
Epoch 97/100
93/93 [==============================] - 95s 1s/step - loss: 1.3850 - accuracy: 0.3189 - val_loss: 1.3860 - val_accuracy: 0.2700
Epoch 98/100
93/93 [==============================] - 95s 1s/step - loss: 1.3849 - accuracy: 0.3131 - val_loss: 1.3860 - val_accuracy: 0.2930
Epoch 99/100
93/93 [==============================] - 95s 1s/step - loss: 1.3851 - accuracy: 0.3078 - val_loss: 1.3860 - val_accuracy: 0.2840
Epoch 100/100
93/93 [==============================] - 95s 1s/step - loss: 1.3850 - accuracy: 0.3122 - val_loss: 1.3861 - val_accuracy: 0.2760
```

```
test acc: 0.39500001072883606
test loss: 1.385149359703064
```

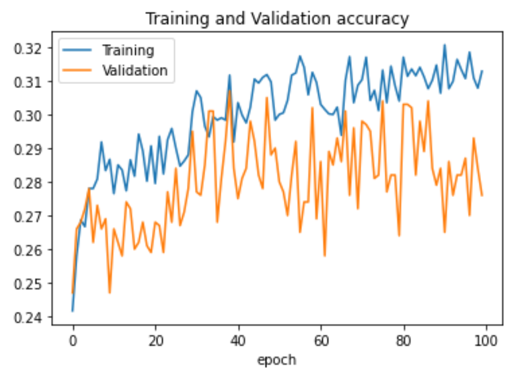


Table 9 VGGNet 19 Model with optimizer ('SGD') and dropout (0.5)

2.4 ResNet 50

We fitted two models with batch size=64, 100 Epoch, image sizes (224*224) and different optimizers and multilayer perceptron:

- 2.4.1 ResNet 50 Model with optimizer ('Adam'), dropout (0.5) and Dense(256, activation="relu")

```
Epoch 92/100
93/93 [==============================] - 106s 1s/step - loss: 0.0513 - accuracy: 0.9845 - val_loss: 0.4417 - val_accuracy: 0.9040
Epoch 93/100
93/93 [==============================] - 106s 1s/step - loss: 0.0429 - accuracy: 0.9862 - val_loss: 1.0532 - val_accuracy: 0.8400
Epoch 94/100
93/93 [==============================] - 106s 1s/step - loss: 0.0607 - accuracy: 0.9834 - val_loss: 0.8179 - val_accuracy: 0.8650
Epoch 95/100
93/93 [==============================] - 106s 1s/step - loss: 0.0477 - accuracy: 0.9832 - val_loss: 0.7092 - val_accuracy: 0.8310
Epoch 96/100
93/93 [==============================] - 106s 1s/step - loss: 0.0504 - accuracy: 0.9825 - val_loss: 1.1148 - val_accuracy: 0.8050
Epoch 97/100
93/93 [==============================] - 106s 1s/step - loss: 0.0510 - accuracy: 0.9864 - val_loss: 2.9248 - val_accuracy: 0.7530
Epoch 98/100
93/93 [==============================] - 106s 1s/step - loss: 0.0627 - accuracy: 0.9791 - val_loss: 0.8703 - val_accuracy: 0.7440
Epoch 99/100
93/93 [==============================] - 106s 1s/step - loss: 0.0461 - accuracy: 0.9850 - val_loss: 0.7213 - val_accuracy: 0.8440
Epoch 100/100
93/93 [==============================] - 106s 1s/step - loss: 0.0567 - accuracy: 0.9830 - val_loss: 3.3957 - val_accuracy: 0.5020
```

```
test acc: 0.7609999775886536
test loss: 0.9364423751831055
```

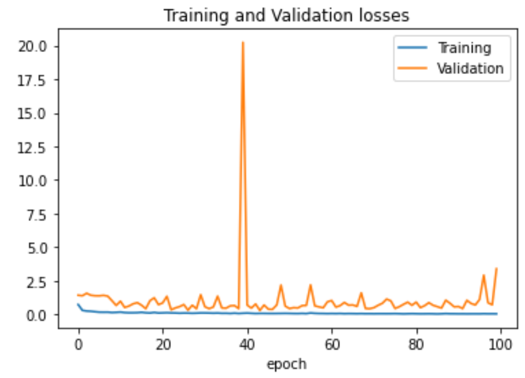


Table 10 ResNet 50 Model with optimizer ('Adam'), dropout (0.5) and Dense(256, activation="relu")

- ### 2.4.2 ResNet 50 Model with optimizer (' SGD'), dropout (0.5) and two MLP layers Dense(512, activation="relu")

```
Epoch 92/100
93/93 [==============================] - 108s 1s/step - loss: 0.0571 - accuracy: 0.9839 - val_loss: 0.4563 - val_accuracy: 0.8890
Epoch 93/100
93/93 [==============================] - 108s 1s/step - loss: 0.0714 - accuracy: 0.9746 - val_loss: 0.4668 - val_accuracy: 0.8890
Epoch 94/100
93/93 [==============================] - 108s 1s/step - loss: 0.0694 - accuracy: 0.9783 - val_loss: 0.5128 - val_accuracy: 0.8710
Epoch 95/100
93/93 [==============================] - 108s 1s/step - loss: 0.0594 - accuracy: 0.9789 - val_loss: 0.4797 - val_accuracy: 0.8890
Epoch 96/100
93/93 [==============================] - 108s 1s/step - loss: 0.0702 - accuracy: 0.9761 - val_loss: 0.4473 - val_accuracy: 0.8930
Epoch 97/100
93/93 [==============================] - 108s 1s/step - loss: 0.0802 - accuracy: 0.9735 - val_loss: 0.4467 - val_accuracy: 0.8890
Epoch 98/100
93/93 [==============================] - 108s 1s/step - loss: 0.0606 - accuracy: 0.9789 - val_loss: 0.5000 - val_accuracy: 0.8810
Epoch 99/100
93/93 [==============================] - 108s 1s/step - loss: 0.0694 - accuracy: 0.9773 - val_loss: 0.4688 - val_accuracy: 0.8830
Epoch 100/100
93/93 [==============================] - 108s 1s/step - loss: 0.0599 - accuracy: 0.9801 - val_loss: 0.4875 - val_accuracy: 0.8800
```

```
test acc: 0.9909999966621399
test loss: 0.04197156801819801
```

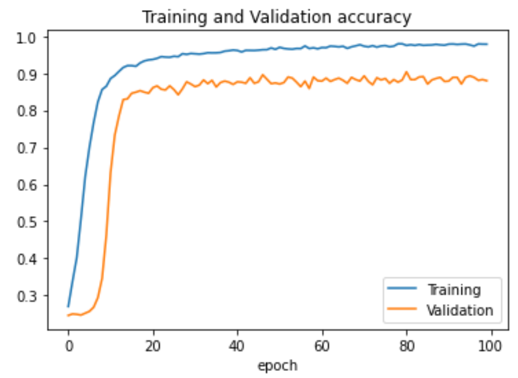


Table 11 ResNet 50 Model with optimizer (' SGD'), dropout (0.5) and two MLP layers Dense(512, activation="relu")

2.4 ResNet 101

We fitted one model with batch size=64, 100 Epoch, image sizes (224*224) , optimizer (' Adam'), dropout (0.5) and one MLP layers Dense(256, activation="relu")

```
Epoch 92/100
93/93 [==============================] - 220s 2s/step - loss: 0.0619 - accuracy: 0.9774 - val_loss: 0.5571 - val_accuracy: 0.8610
Epoch 93/100
93/93 [==============================] - 220s 2s/step - loss: 0.0658 - accuracy: 0.9793 - val_loss: 0.9868 - val_accuracy: 0.8580
Epoch 94/100
93/93 [==============================] - 220s 2s/step - loss: 0.0441 - accuracy: 0.9832 - val_loss: 0.8398 - val_accuracy: 0.8580
Epoch 95/100
93/93 [==============================] - 220s 2s/step - loss: 0.0411 - accuracy: 0.9837 - val_loss: 0.4327 - val_accuracy: 0.9000
Epoch 96/100
93/93 [==============================] - 220s 2s/step - loss: 0.0448 - accuracy: 0.9841 - val_loss: 0.9682 - val_accuracy: 0.8280
Epoch 97/100
93/93 [==============================] - 220s 2s/step - loss: 0.0669 - accuracy: 0.9775 - val_loss: 0.8753 - val_accuracy: 0.8210
Epoch 98/100
93/93 [==============================] - 220s 2s/step - loss: 0.0453 - accuracy: 0.9848 - val_loss: 0.4073 - val_accuracy: 0.8880
Epoch 99/100
93/93 [==============================] - 220s 2s/step - loss: 0.0562 - accuracy: 0.9796 - val_loss: 0.6261 - val_accuracy: 0.8450
Epoch 100/100
93/93 [==============================] - 221s 2s/step - loss: 0.0421 - accuracy: 0.9845 - val_loss: 0.4648 - val_accuracy: 0.8640
```

```
test acc: 0.9800000190734863
test loss: 0.09789120405912399
```

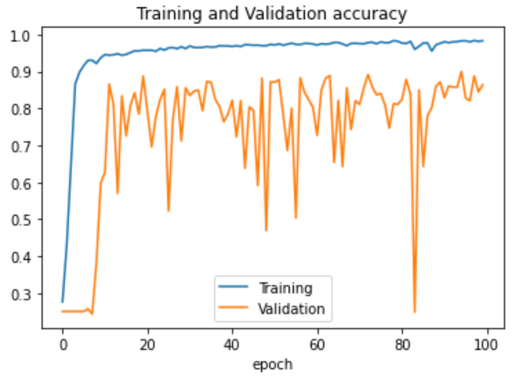


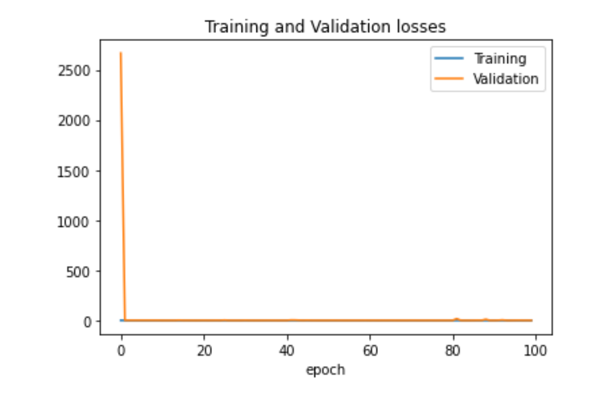
Table 12 ResNet 101 Model with batch size(64), image size (224) , optimizer (' Adam'), dropout (0.5) and one MLP layers Dense(512, activation="relu")

2.5 ResNet 152

We fitted one model with batch size=64, 100 Epoch, image sizes (224*224) , optimizer (' SGD'), dropout (0.5) and one MLP layers Dense(512, activation="relu")

```
Epoch 92/100
93/93 [==============================] - 164s 2s/step - loss: 0.0709 - accuracy: 0.9753 - val_loss: 1.6833 - val_accuracy: 0.6460
Epoch 93/100
93/93 [==============================] - 164s 2s/step - loss: 0.0781 - accuracy: 0.9734 - val_loss: 4.6877 - val_accuracy: 0.6580
Epoch 94/100
93/93 [==============================] - 165s 2s/step - loss: 0.0517 - accuracy: 0.9839 - val_loss: 0.9489 - val_accuracy: 0.8220
Epoch 95/100
93/93 [==============================] - 165s 2s/step - loss: 0.0510 - accuracy: 0.9835 - val_loss: 0.4858 - val_accuracy: 0.8750
Epoch 96/100
93/93 [==============================] - 165s 2s/step - loss: 0.0529 - accuracy: 0.9839 - val_loss: 1.0967 - val_accuracy: 0.8310
Epoch 97/100
93/93 [==============================] - 165s 2s/step - loss: 0.0525 - accuracy: 0.9831 - val_loss: 1.6046 - val_accuracy: 0.7580
Epoch 98/100
93/93 [==============================] - 164s 2s/step - loss: 0.0479 - accuracy: 0.9824 - val_loss: 0.7097 - val_accuracy: 0.8270
Epoch 99/100
93/93 [==============================] - 165s 2s/step - loss: 0.0646 - accuracy: 0.9773 - val_loss: 0.9955 - val_accuracy: 0.8410
Epoch 100/100
93/93 [==============================] - 165s 2s/step - loss: 0.0570 - accuracy: 0.9830 - val_loss: 0.4723 - val_accuracy: 0.8890
```

```
test acc: 0.9679999947547913
test loss: 0.16527485847473145
```



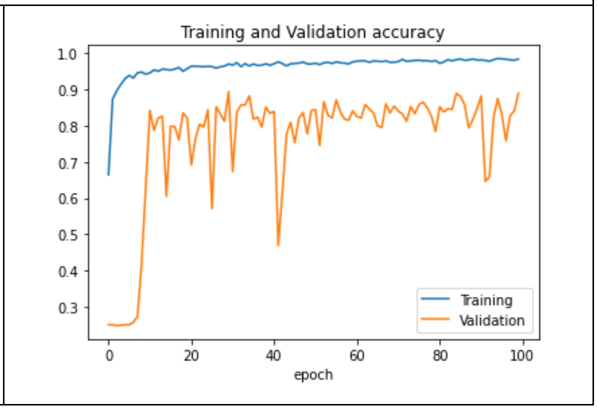


Table 13 ResNet 152 Model with batch size(64), image size (224) , optimizer (' SGD'), dropout (0.5) and one MLP layers Dense(512, activation="relu")

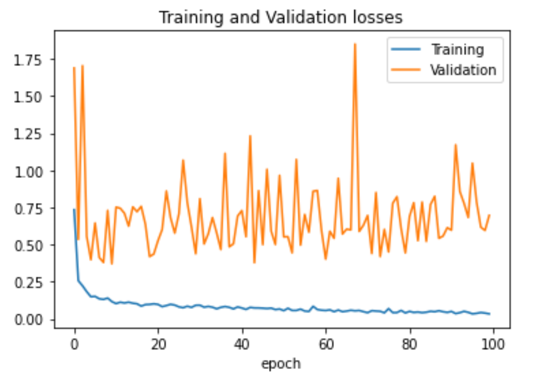
2.6 Densenet121

We fitted two standard models with 100 Epoch and different optimizers, batch size and image sizes:

➢ 2.6.1 Densenet121 Model with image size (224*224) ,batch size (64) and optimizer (' Adam')

```
Epoch 92/100
93/93 [==============================] - 114s 1s/step - loss: 0.0351 - accuracy: 0.9878 - val_loss: 1.1720 - val_accuracy: 0.7610
Epoch 93/100
93/93 [==============================] - 114s 1s/step - loss: 0.0302 - accuracy: 0.9902 - val_loss: 0.8573 - val_accuracy: 0.8470
Epoch 94/100
93/93 [==============================] - 114s 1s/step - loss: 0.0417 - accuracy: 0.9870 - val_loss: 0.7781 - val_accuracy: 0.8440
Epoch 95/100
93/93 [==============================] - 114s 1s/step - loss: 0.0439 - accuracy: 0.9853 - val_loss: 0.6829 - val_accuracy: 0.8520
Epoch 96/100
93/93 [==============================] - 116s 1s/step - loss: 0.0318 - accuracy: 0.9901 - val_loss: 1.0484 - val_accuracy: 0.8490
Epoch 97/100
93/93 [==============================] - 116s 1s/step - loss: 0.0418 - accuracy: 0.9870 - val_loss: 0.7856 - val_accuracy: 0.8380
Epoch 98/100
93/93 [==============================] - 115s 1s/step - loss: 0.0434 - accuracy: 0.9865 - val_loss: 0.6174 - val_accuracy: 0.8460
Epoch 99/100
93/93 [==============================] - 115s 1s/step - loss: 0.0369 - accuracy: 0.9863 - val_loss: 0.5954 - val_accuracy: 0.8800
Epoch 100/100
93/93 [==============================] - 115s 1s/step - loss: 0.0294 - accuracy: 0.9925 - val_loss: 0.6965 - val_accuracy: 0.8520
```

```
test acc: 0.984000027179718
test loss: 0.04853156581521034
```



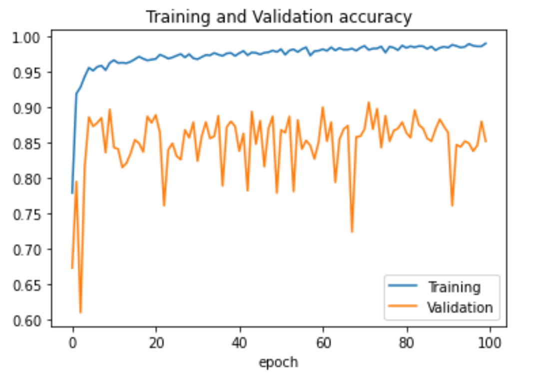


*Table 14 Densenet121 Model with image size (224*224) ,batch size (64) and optimizer (' Adam')*

2.6.2 Densenet121 Model with image size (256*256), batch size (32) and optimizer (' SGD')

```
Epoch 92/100
187/187 [==============================] - 135s 718ms/step - loss: 0.0316 - accuracy: 0.9888 - val_loss: 0.5735 - val_accuracy: 0.8850
Epoch 93/100
187/187 [==============================] - 135s 718ms/step - loss: 0.0385 - accuracy: 0.9867 - val_loss: 0.5492 - val_accuracy: 0.8910
Epoch 94/100
187/187 [==============================] - 135s 719ms/step - loss: 0.0371 - accuracy: 0.9871 - val_loss: 0.4928 - val_accuracy: 0.9020
Epoch 95/100
187/187 [==============================] - 135s 718ms/step - loss: 0.0340 - accuracy: 0.9887 - val_loss: 0.4724 - val_accuracy: 0.8950
Epoch 96/100
187/187 [==============================] - 135s 718ms/step - loss: 0.0281 - accuracy: 0.9911 - val_loss: 0.4986 - val_accuracy: 0.9030
Epoch 97/100
187/187 [==============================] - 135s 718ms/step - loss: 0.0398 - accuracy: 0.9874 - val_loss: 0.6414 - val_accuracy: 0.8730
Epoch 98/100
187/187 [==============================] - 135s 719ms/step - loss: 0.0258 - accuracy: 0.9931 - val_loss: 0.5485 - val_accuracy: 0.8920
Epoch 99/100
187/187 [==============================] - 135s 718ms/step - loss: 0.0381 - accuracy: 0.9885 - val_loss: 0.5362 - val_accuracy: 0.8910
Epoch 100/100
187/187 [==============================] - 135s 720ms/step - loss: 0.0386 - accuracy: 0.9879 - val_loss: 0.6284 - val_accuracy: 0.8870
```

```
test acc: 0.9900000095367432
test loss: 0.04768458753824234
```

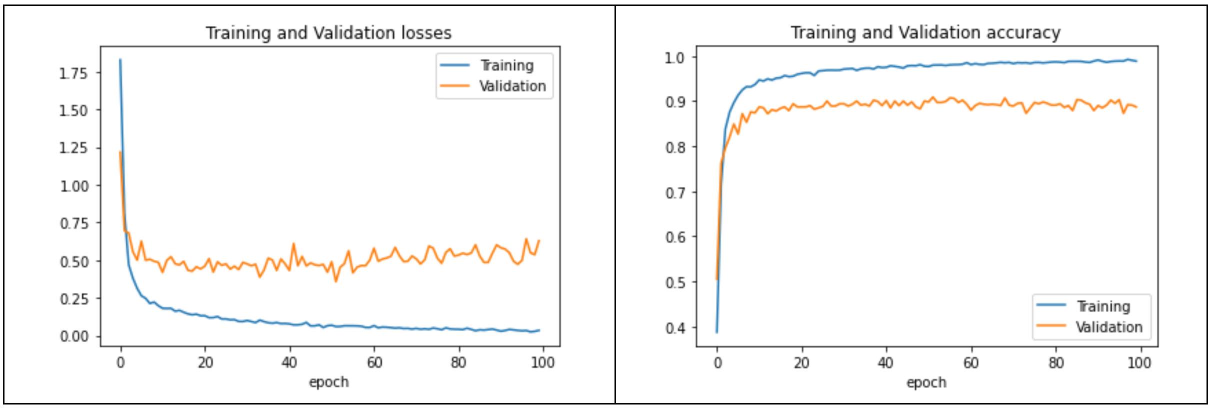


*Table 15 Densenet121 Model with image size (256*256), batch size (32) and optimizer (' SGD')*

Summary

After flitted a lot of models we found that there are four CNN algorithms gave us a good results such as VGGNet16 ,VGGNet19 ,ResNet 50 and DenesNet 121

3.1 ResNet 50 model code

```
import cv2
import tensorflow as tf
from tensorflow.keras.preprocessing import image
from tensorflow.keras.preprocessing.image import ImageDataGenerator ,
 load_img ,img_to_array
from tensorflow.keras.models import Sequential,Model
from tensorflow.keras.layers import Activation, Dropout, Flatten, Den
se, Conv2D, MaxPooling2D,GlobalAveragePooling2D
from tensorflow.keras.applications.resnet import ResNet50
from tensorflow.keras.callbacks import EarlyStopping
from tensorflow.keras.optimizers import SGD
from sklearn.metrics import classification_report,confusion_matrix

model = ResNet50(weights='imagenet',include_top=False)
x = model.output
x = GlobalAveragePooling2D()(x)
x = Dense(512, activation='relu')(x)
x = Dropout(0.5)(x)
x = Dense(512, activation='relu')(x)
x = Dropout(0.5)(x)
out = Dense(4,activation='softmax')(x)
model_final = Model(inputs = model.input,outputs=out)

model_final.compile(optimizer=SGD(lr=0.0001, momentum=0.9), loss='cat
egorical_crossentropy', metrics = ['accuracy'])
history=model_final.fit_generator(train_generator,
                    steps_per_epoch=train_generator.samples/train_gen
erator.batch_size,
                    epochs=100,
                    validation_data=valid_generator,
                    validation_steps=valid_generator.samples/train_ge
nerator.batch_size,
                    verbose=1)
```

3.2 Models Summary

Model	Parameters	Num of Epochs	Total accuracy and Losses
LeNet	batch size=64 Image size (150) optimizer ('RMSprop')	100	test acc: 0.746999979019165 test loss: 0.6334831714630127
LeNet	batch size=64 Image size (224) optimizer ('Adam')	100	test acc: 0.8130000233650208 test loss: 0.6019443273544312
VGGNet 16	batch size=64 Image size (224) optimizer ('RMSprop')	100	test acc: 0.9829999804496765 test loss: 0.06923840939998627
VGGNet 16	batch size=64 Image size (224) optimizer ('SGD')	100	test acc: 0.3799999952316284 test loss: 1.384906291961670
VGGNet 19	batch size=64 Image size (224) optimizer ('RMSprop')	100	test acc: 0.9850000143051147 test loss: 0.054889362305402
VGGNet 19	batch size=64 Image size (224) dropout (0.5) optimizer ('RMSprop')	100	test acc: 0.972000002861023 test loss: 0.09611707925796509
VGGNet 19	batch size=64 Image size (224) optimizer ('Adam')	100	test acc: 0.25 test loss: 1.3862971067428589
VGGNet 19	batch size=64 Image size (224) dropout (0.5) optimizer ('SGD')	100	test acc: 0.39500001072883606 test loss: 1.385149359703064
ResNet 50	batch size=64 Image size (224) dropout (0.5) optimizer ('Adam') Dense(256, activation="relu")	100	test acc: 0.7609999775886536 test loss: 0.9364423751831055
ResNet 50	batch size=64 Image size (224) dropout (0.5) optimizer ('SGD') Dense(512, activation="relu")	100	test acc: 0.9909999966621399 test loss: 0.04197156801819801

ResNet 101	batch size=64 Image size (224) dropout (0.5) optimizer ('Adam') Dense(256, activation="relu")	100	test acc: 0.9800000190734863 test loss: 0.09789120405912399
ResNet 152	batch size=64 Image size (224) dropout (0.5) optimizer ('SGD') Dense(512, activation="relu")	100	test acc: 0.9679999947547913 test loss: 0.16527485847473145
DenseNet121	batch size=64 Image size (224) optimizer ('Adam')	100	test acc: 0.984000027179718 test loss: 0.04853156581521034
DenseNet121	batch size=32 Image size (224) optimizer ('SGD')	100	test acc: 0.9900000095367432 test loss: 0.04768458753824234

Table 16 Models summary