Topic: **Detecting Coronavirus Infections through Chest X-Ray images**

Deep Learning

Assignment 5

**REPORT**

Submitted to: Dr. Mohsin Ali

Submitted by: Taha Muzammil

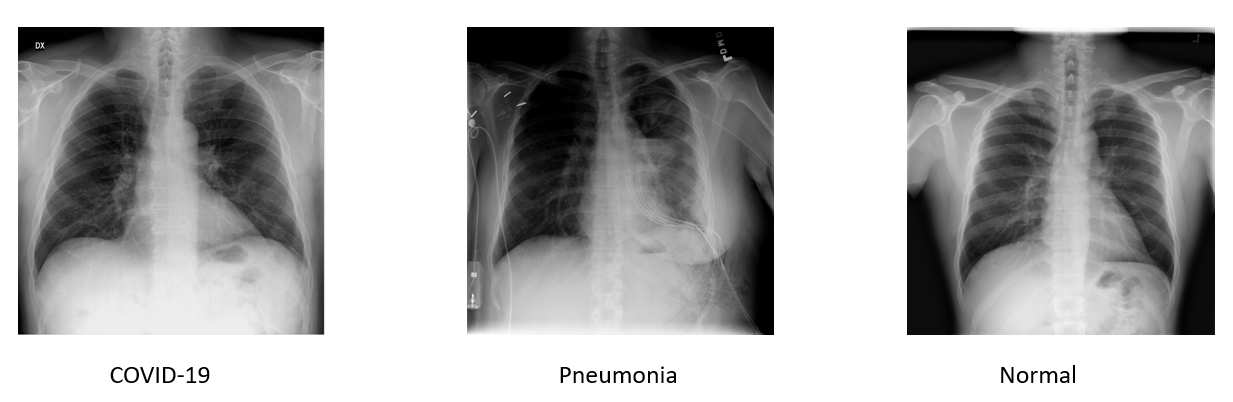
BSCS16063

In this assignment, I used two pre trained models VGG16 and Resnet16, these models are trained on ImageNet dataset of 1000 classes.

# Dataset: For this assignment, Chest X-Ray Images Dataset is used which contains two classes names as ‘Infected’ and ‘normal’. Dataset contain pictures of CONVID-19 and Pneumonia patients. Here is the details of dataset:

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **# of images in training set** | **# of images in validation set** | **# of images in test set** |
| **Infected** | 4,919 | 615 | 615 |
| **Normal** | 7,081 | 885 | 885 |

Chest X-Ray images are taken in different views (AP or PA) depending on which side of the body is facing the X-Ray scanner. Images from different views have slightly different features. For this task, we will be using images without considering their views. A few sample images:



All images having size 1000x1000, in preprocessing normalize all imgaes to size 224x224 with some mean and standard deviation, training images are augmented randomly of angle 45 degree.

**Task 1**

**VGG16**

# Load Model and Fine Tune:

* Firstly load pretrained model of VGG16 usingmodels.vgg16(pretrained=True, progress=True) and freeze all the weights using requires\_grad=False on every layer of model.
* Remove all connected Fully Connected (FC) layers from model and join with 2 FC layer, neurons in the firct FC layer are 730 and 2nd FC layer have 2 neurons of output. This is the final shape of FC layers : (classifier): Sequential(

(0): Linear(in\_features=25088, out\_features=730, bias=True)

(1): ReLU(inplace=True)

(2): Dropout(p=0.5, inplace=False)

(3): Linear(in\_features=730, out\_features=2, bias=False))

# Training Loop:

# Hyperparameters for training loop are:

# 

# Training is done using GPU on google Collab

# Experiments Performed:

# Several experiments performed using pretrained VGG16 model, by changing learning rate and no of neurons in FC layers.

# Experiment 1:

# In this experiment learning rate is 0.001, number of neurons in 1st FC layer are 1630 and 2 in last FC layer.

* Training Accuracy 87.95 %

# Validation Accuracy 89.13333333333334 %

* Accuracy of the network on the test images: 95 %

# F1 testing Score: 0.959251101321586

# 

# Accuracy and Loss curves:

# *(Note: Blue line for training and red line for validation)*

# 

# Confusion matrices:

# (Training, Validation and Testing)

# 

# Experiment 2:

# In this experiment learning rate is 0.0001, number of neurons in 1st FC layer are 1630 and 2 in last FC layer.

* Training Accuracy 86.25 %
* Validation Accuracy 88.06666666666666 %

# Accuracy of the network on the test images: 94 %

# 

# Accuracy and Loss curves:

# *(Note: Blue line for training and red line for validation)*

# 

# Confusion matrices:

# (Training, Validation and Testing)

# 

# Experiment 3:

# In this experiment learning rate is 0.001, no of epochs are 10, number of neurons in 1st FC layer are 730 and 2 in last FC layer.

* Training Accuracy 87.79166666666667 %
* Validation Accuracy 88.26666666666667 %%
* Accuracy of the network on the test images: 94 %

# F1 Score : 0.9559471365638766

# 

# Accuracy and Loss curves:

# *(Note: Blue line for training and red line for validation)*

# 

# Confusion matrices:

# (Training, Validation and Testing)

# 

**Resnet18**

# Load Model and Fine Tune:

* Firstly load pretrained model of Resnet18 usingmodels. Resnet18 (pretrained=True, progress=True) and freeze all the weights using requires\_grad=False on every layer of model.
* Remove all connected Fully Connected (FC) layers from model and join with 2 FC layer, neurons in the firct FC layer are 730 and 2nd FC layer have 2 neurons of output. This is the final shape of FC layers : (classifier): Sequential(

(0): Linear(in\_features=512, out\_features=730, bias=True)

(1): ReLU(inplace=True)

(2): Dropout(p=0.5, inplace=False)

(3): Linear(in\_features=730, out\_features=2, bias=False))

# Training Loop:

# Hyperparameters for training loop are:

# 

# Training is done using GPU on google Collab

# Experiments Performed:

# Several experiments performed using pretrained Resnet18 model, by changing learning rate and no. of neurons in FC layers.

# Experiment 1:

# In this experiment learning rate is 0.001, number of neurons in 1st FC layer are 730 and 2 in last FC layer. (10 Epochs)

* Training Accuracy 84.64166666666667 %
* Validation Accuracy 85.26666666666667 %
* Accuracy of the Resnet-18 on the test images: 92 %

# F1 Score : 0.9345794392523364

# Accuracy and Loss curves:

# *(Note: Blue line for training and red line for validation)*

# 

# Confusion matrices:

# (Training, Validation and Testing)

# 

# Experiment 2:

# In this experiment learning rate is 0.0001, number of neurons in 1st FC layer are 730 and 2 in last FC layer. (10 Epochs)

* Training Accuracy 79.15 %
* Validation Accuracy 80.8 %
* Accuracy of the Resnet-18 on the test images: 87 %

# F1 Score: 0.898550724637681

# Accuracy and Loss curves:

# *(Note: Blue line for training and red line for validation)*

# 

# Confusion matrices:

# (Training, Validation and Testing)

# 

# TASK 2

# In Task 2 we are supposed to unfreeze last convolution layers and FC layers (like Task1), than more convolutions layer and unfreeze complete model.

# All experiments are given below:

**VGG16**

# Load Model and Fine Tune:

* Firstly load pretrained model of VGG16 usingmodels.vgg16(pretrained=True, progress=True) and freeze all the weights using requires\_grad=False on every layer of model.
* Remove all connected Fully Connected (FC) layers from model and join with 2 FC layer, neurons in the firct FC layer are 730 and 2nd FC layer have 2 neurons of output. This is the final shape of FC layers : (classifier): Sequential(

(0): Linear(in\_features=25088, out\_features=730, bias=True)

(1): ReLU(inplace=True)

(2): Dropout(p=0.5, inplace=False)

(3): Linear(in\_features=730, out\_features=2, bias=False))

# Training Loop:

# Hyperparameters for training loop are:

# 

# Training is done using GPU on google Collab

# Experiments Performed:

# Several experiments performed using pretrained VGG16 model, by changing learning rate, no. of unfreeze convolution layer.

# Experiment 1:

# In this experiment learning rate is 0.001, number of neurons in 1st FC layer are 730 and 2 in last FC layer. (10 Epochs)

**Layers Unfreeze:** Unfreeze complete vgg16 model

* Training Accuracy 91.725 %
* Validation Accuracy 91.46666666666667
* Accuracy of the network on the test images: 97 %

# F1 Score : 0.9775784753363229

# Accuracy and Loss curves:

# *(Note: Blue line for training and red line for validation)*

# 

# Confusion matrices:

# (Training, Validation and Testing)

# 

# Experiment 2:

# In this experiment learning rate is 0.001, number of neurons in 1st FC layer are 730 and 2 in last FC layer. (10 Epochs)

**Layers Unfreeze:** Unfreeze last 3 convolution layers and all FC layers

* Training Accuracy 91.13333333333334 %
* Validation Accuracy 89.86666666666666 %
* Accuracy of the network on the test images: 96 %

# F1 Score : 0.9722530521642619

# Accuracy and Loss curves:

# *(Note: Blue line for training and red line for validation)*

# 

# Confusion matrices:

# (Training, Validation and Testing)

# 

# Experiment 3:

# In this experiment learning rate is 0.001, number of neurons in 1st FC layer are 730 and 2 in last FC layer. (10 Epochs)

**Layers Unfreeze:** Unfreeze last 6 convolution layers and all FC layers

* Training Accuracy 91.91666666666667 %
* Validation Accuracy 91.4 %
* Accuracy of the network on the test images: 96 %

# F1 Score : 0.9745575221238938

# Accuracy and Loss curves:

# *(Note: Blue line for training and red line for validation)*

# 

# Confusion matrices:

# (Training, Validation and Testing)

# 

**Resnet18**

# Load Model and Fine Tune:

* Firstly load pretrained model of Resnet18 usingmodels. Resnet18 (pretrained=True, progress=True) and freeze all the weights using requires\_grad=False on every layer of model.
* Remove all connected Fully Connected (FC) layers from model and join with 2 FC layer, neurons in the firct FC layer are 730 and 2nd FC layer have 2 neurons of output. This is the final shape of FC layers : (classifier): Sequential(

(0): Linear(in\_features=512, out\_features=730, bias=True)

(1): ReLU(inplace=True)

(2): Dropout(p=0.5, inplace=False)

(3): Linear(in\_features=730, out\_features=2, bias=False))

# Training Loop:

# Hyperparameters for training loop are:

# 

# Training is done using GPU on google Collab

# Experiments Performed:

# Several experiments performed using pretrained Resnet18 model, by changing learning rate, no. of unfreeze convolution layer.

# Experiment 1:

# In this experiment learning rate is 0.001, number of neurons in 1st FC layer are 730 and 2 in last FC layer. (10 Epochs)

**Layers Unfreeze:** Unfreeze complete resnet18 model

* Training Accuracy 91.71666666666667 %
* Validation Accuracy 89.73333333333333 %
* Accuracy of the Resnet-18 on the test images: 96 %

# F1 Score : 0.9690607734806629

# Accuracy and Loss curves:

# *(Note: Blue line for training and red line for validation)*

# 

# Confusion matrices:

# (Training, Validation and Testing)

# 

# Experiment 2:

# In this experiment learning rate is 0.001, number of neurons in 1st FC layer are 730 and 2 in last FC layer. (10 Epochs)

**Layers Unfreeze:** Unfreeze last layer4 and complete FC layers of resnet18 model

* Training Accuracy 89.68333333333334 %
* Validation Accuracy 89.66666666666667 %
* Accuracy of the Resnet-18 on the test images: 96 %

# F1 Score : 0.9666666666666667

# Accuracy and Loss curves:

# *(Note: Blue line for training and red line for validation)*

# 

# Confusion matrices:

# (Training, Validation and Testing)

# 

# Experiment 3:

# In this experiment learning rate is 0.001, number of neurons in 1st FC layer are 730 and 2 in last FC layer. (10 Epochs)

**Layers Unfreeze:** Unfreeze last layer4 and layer3 and complete FC layers of resnet18 model

* Training Accuracy 91.44166666666666 %
* Validation Accuracy 89.86666666666666 %
* Accuracy of the Resnet-18 on the test images: 96 %

# F1 Score : 0.9690837549184936

# Accuracy and Loss curves:

# *(Note: Blue line for training and red line for validation)*

# 

# Confusion matrices:

# (Training, Validation and Testing)

# 