**MSDS17037**

**Muhammad Sufian**

**Deep Learning**

**Assignment # 05**

# **Task 1**

Following are the details against task 1.

# VGG16

#### Network Setting:-

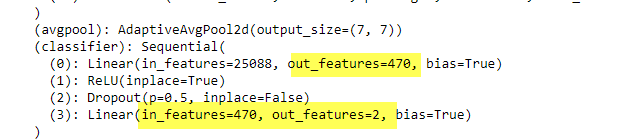
* Random rotation of 30 degree has been applied to augment images
* Images are resized to 256 and later on applied crop of 256, as some of the images causing network failure due to different dimensions
* Batch Normalization is applied with .5 mean and .3 standard deviation

#### Loader Setting:-

* Batch size of 64 is applied on training data, while 6 batch is applied to validation and testing sets. Few memory errors were occurred while changing sizes but this seems good in overall results accuracy

#### Model

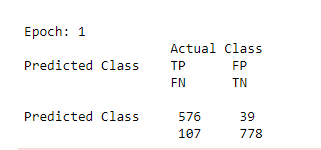
* Pretrained model of VGG16 has been used with following changes
* All the layers except FC layers were freezed
* The last three FC layers are replaced with two new layers according to my roll number last two digits i.e. 37

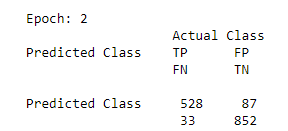


#### Mode Parameters

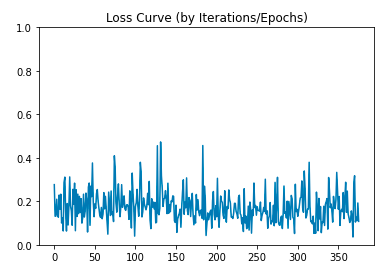
* Stochastic gradient descent as optimization function
* Learning Rate:- 0.005
* Momentum:- 0.9
* Cross Entropy with Softmax for loss function
* # of Epochs: 2 (within two epochs I found quite good accuracy so did not continued with more epochs)

#### Validation Accuracy Matrix

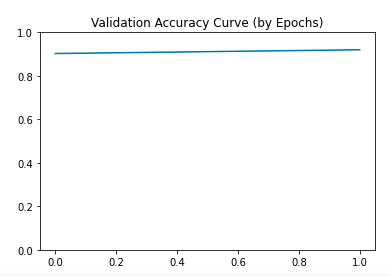




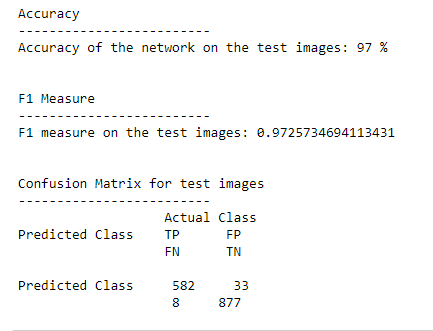
#### Training Loss Curve



#### Validation Accuracy (2 epochs)



#### Testing Error, Confusion Matrix and F1 Measure



# ResNet18

#### Network Setting:-

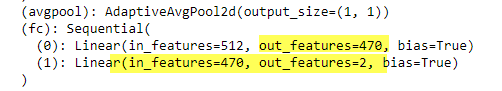
* These are similar to VGG16
  + Random rotation of 30 degree has been applied to augment images
  + Images are resized to 256 and later on applied crop of 256, as some of the images causing network failure due to different dimensions
  + Batch Normalization is applied with .5 mean and .3 standard deviation

#### Loader Setting:-

* Similar to VGG16
  + Batch size of 64 is applied on training data, while 6 batch is applied to validation and testing sets. Few memory errors were occurred while changing sizes but this seems good in overall results accuracy

#### Model

* Pretrained model of ResNet has been used with following changes
* All the layers except FC layers were freezed
* The only FC layer is replaced with two new layers according to my roll number last two digits i.e. 37



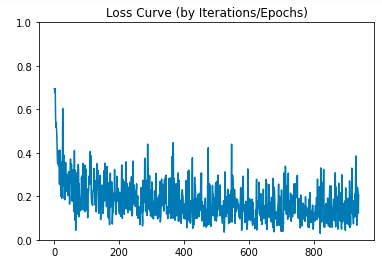
#### Mode Parameters

* Stochastic gradient descent as optimization function
* Learning Rate:- 0.005
* Momentum:- 0.9
* Cross Entropy with Softmax for loss function
* # of Epochs: 5

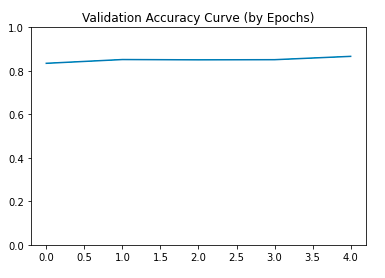
#### Validation Accuracy Matrix (By Epochs)

|  |
| --- |
|  |
|  |
|  |
|  |
|  |

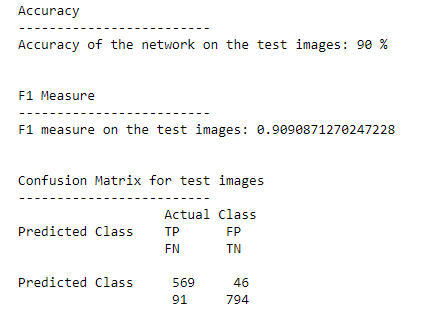
#### Training Loss Curve



#### Validation Accuracy (5 epochs)



#### Testing Error, Confusion Matrix and F1 Measure



# **Task 2**

Following are the details against task 2. Both of the networks mentioned below are trained for 1 conv layer, few convolution layers and all convolution layers.

The below section will explain the entire conv layer learning and later below, comparison is given with 1, or few conv layers learning.

# VGG16

#### Network Setting:-

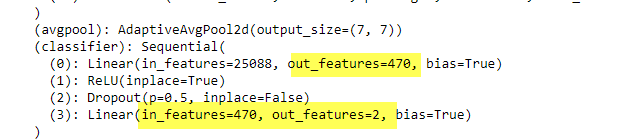
* These are similar to VGG16 mentioned in task 1
  + Random rotation of 30 degree has been applied to augment images
  + Images are resized to 256 and later on applied crop of 256, as some of the images causing network failure due to different dimensions
  + Batch Normalization is applied with .5 mean and .3 standard deviation

#### Loader Setting:-

* Similar to VGG16 mentioned in task 1
  + Batch size of 64 is applied on training data, while 6 batch is applied to validation and testing sets. Few memory errors were occurred while changing sizes but this seems good in overall results accuracy

#### Model

* Pretrained model of VGG16 has been used with following changes
* All the layers were unfreezed
* The only FC layer is replaced with two new layers according to my roll number last two digits i.e. 37



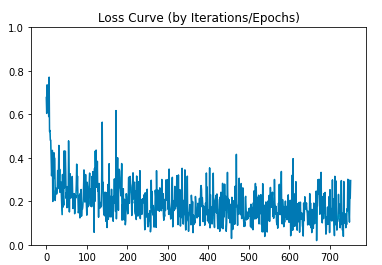
#### Mode Parameters

* Stochastic gradient descent as optimization function
* Learning Rate:- 0.005
* Momentum:- 0.9
* Cross Entropy with Softmax for loss function
* # of Epochs: 4

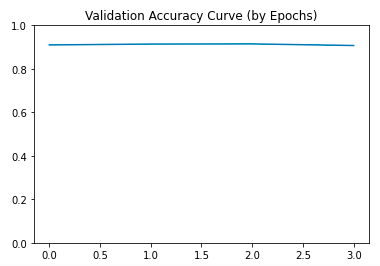
#### Validation Accuracy Matrix (By Epochs)

|  |
| --- |
|  |
|  |
|  |
|  |

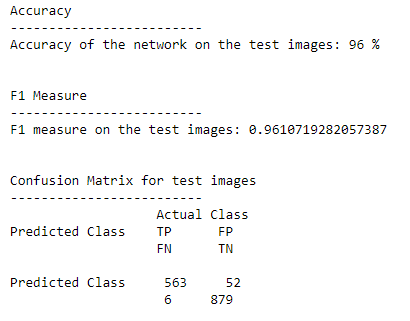
#### Training Loss Curve



#### Validation Accuracy (4 epochs)



#### Testing Error, Confusion Matrix and F1 Measure



# ResNet18

#### Network Setting:-

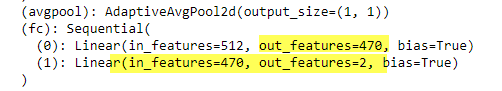
* These are similar to VGG16 mentioned in task 1
  + Random rotation of 30 degree has been applied to augment images
  + Images are resized to 256 and later on applied crop of 256, as some of the images causing network failure due to different dimensions
  + Batch Normalization is applied with .5 mean and .3 standard deviation

#### Loader Setting:-

* Similar to VGG16 mentioned in task 1
  + Batch size of 64 is applied on training data, while 6 batch is applied to validation and testing sets. Few memory errors were occurred while changing sizes but this seems good in overall results accuracy

#### Model

* Pretrained model of ResNet has been used with following changes
* All the layers were unfreezed
* The only FC layer is replaced with two new layers according to my roll number last two digits i.e. 37



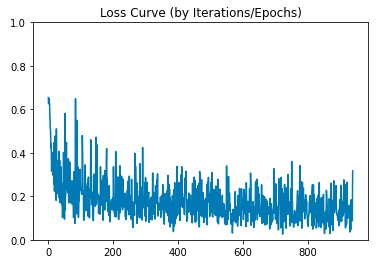
#### Mode Parameters

* Stochastic gradient descent as optimization function
* Learning Rate:- 0.005
* Momentum:- 0.9
* Cross Entropy with Softmax for loss function
* # of Epochs: 5

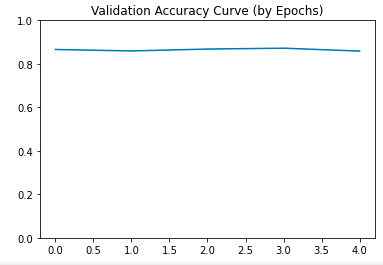
#### Validation Accuracy Matrix (By Epochs)

|  |
| --- |
|  |
|  |
|  |
|  |
|  |

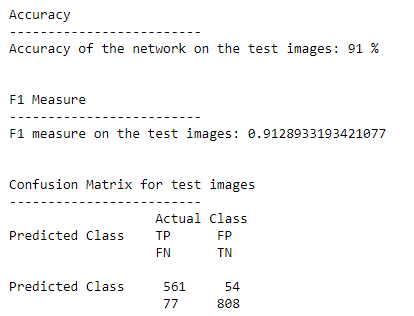
#### Training Loss Curve



#### Validation Accuracy (5 epochs)



#### Testing Error, Confusion Matrix and F1 Measure



# **VGG16 Freezing Layers Comparison**

The noticeable thing in VGG was that either we just retrain FC layers and get the prediction from model, or we train one/few or all convolution layers, the accuracy stays higher (a little bit exception I faced when only last convolution layer was retrained with FC layers).

Below are the results of each of these cases.

|  |  |  |  |
| --- | --- | --- | --- |
| **Only FC Layers Retrained** | **Last Conv Layer Retrained**  **(with FC Layers)** | **21-26 Conv Layers Retrained**  **(with FC Layers)** | **All Conv Layers Retrained**  **(with FC Layers)** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# **ResNet18 Freezing Layers Comparison**

ResNet has one only FC layer, as its model is more deep/complex in convolution layers i.e. having subnetworks. Only replacing FC layer does not produce comparatively better accuracy, as model is more complex in nature for convolution layers i.e. sub networks.

Below are the results of each of these cases.

|  |  |  |  |
| --- | --- | --- | --- |
| **Only FC Layers Retrained** | **Last Conv Layer Retrained**  **(with FC Layers)** | **Last SubNerwork Layers Retrained**  **(with FC Layers)** | **All Conv Layers Retrained**  **(with FC Layers)** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# **VGG16 vs ResNet18 Comparison**

VGG is relatively simpler model and provided good results on given dataset. The one reason of ResNet18 not performing well, as it might need to be trained carefully with parameters. The ResNet is complex model with subnets and not easily retainable as of VGG.

However Task 1 and Task 2 both provides a good understanding of transfer learning, where just replacing FC layer will produce good results if our dataset is similar (or like subset of imageNet) on which these models are trained.

**GitHUB Link:**

<https://github.com/mmsufian1982/deeplearning>

Please note all the above results were obtained from multiple note books running in parallel. The notebook with submission is compiled for FC layers and Entire convolution layers, not for 1 layer and few layers that are already shown in above report.