Assignment 05

Deep Learning – Spring 2020

Muhammad Rauf Tabassam
MSCS18030

25-Apr-20

Information Technology University, Lahore

Detecting Coronavirus Infections through Chest X-Ray images

Detecting Coronavirus infections through chest X-ray images is a computer vision problem by applying deep learning algorithms on it. The details of dataset provided for this assignment is as follows:

Dataset Details:
This dataset contains X-Ray images from 2 classes:

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

I don't know but for some reasons, my code does not collect all 12000 training images, sometimes it gets 10293 or 10582 or 10611 or 11042 etc. But it gets all 1500 images for both validation and testing both. So I complete my assignment with the available data.

GitHub Repository Link:

https://github.com/rauftabassam/MSCS18030_COVID19_DLSpring2020

Task 1: Load pre-trained CNN model and fine-tune FC Layers

1. VGG16

I use the pre-trained VGG16 model on Image Net and change the properties of the FC layers by changing its in_features and out_features. I just remove the last FC layer, dropout and RELU layer and modify the features for remaining two layers as required. Last 2 digits of my roll number are 30 and by provided formula, the out_features of first layer are equal to 400, same for the next output layer, the out_features are 2 and in_features are 400. By applying the same method provided in tutorial for training and testing, the accuracy goes up to **94%**.

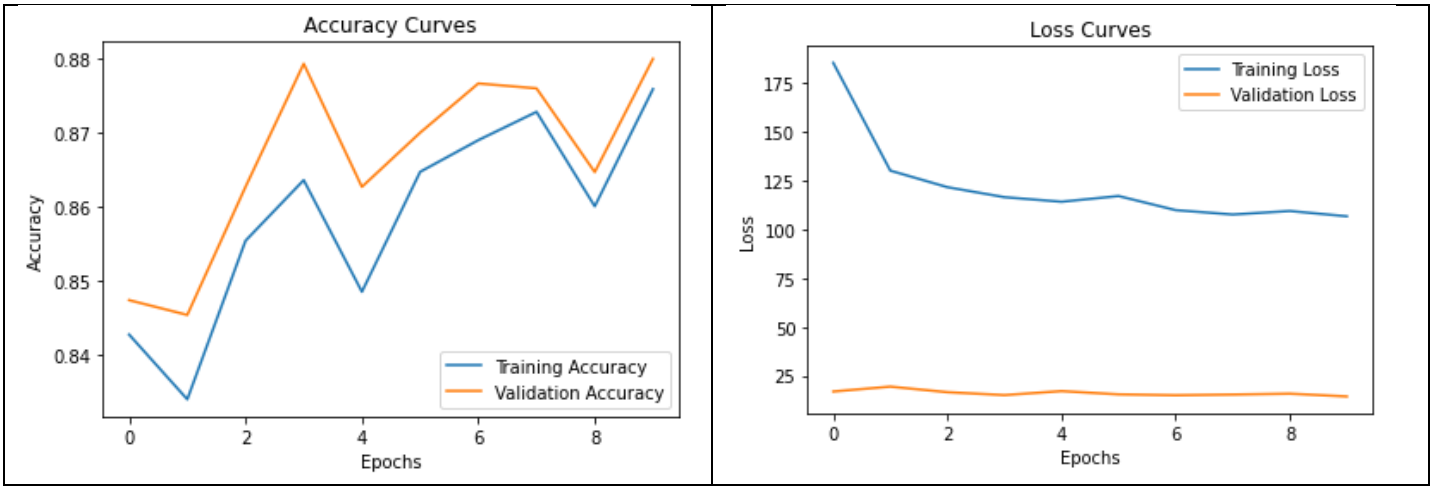
Confusion Matrix

		Predicted Values	
		"infected"	"normal"
Actual Values	"infected"	TP = 552	FN = 63
	"normal"	FP = 27	TN = 858

Experimental Setup

- Learning Rate = 0.001
- Number of fine-tuned fully connected layers = 2
- Neurons in fully connected layers = [400, 2]
- Epochs = 10
- Momentum = 0.9
- Block size = 32

Loss and Accuracy Curves



Final Accuracy and F1 score

- Testing Accuracy = 94%
- F1-Score = 0.94

All the parameters of the models are saved named as "vgg16_FC_Only.pth"

---

2. RES18

Modification of FC layer was quite different than VGG16. I just created two FC layers and update the FC layers of RES18 by direct assigning to the fc of the res18 object. The accuracy on test data is **96%** and the confusion matrix is as follows:

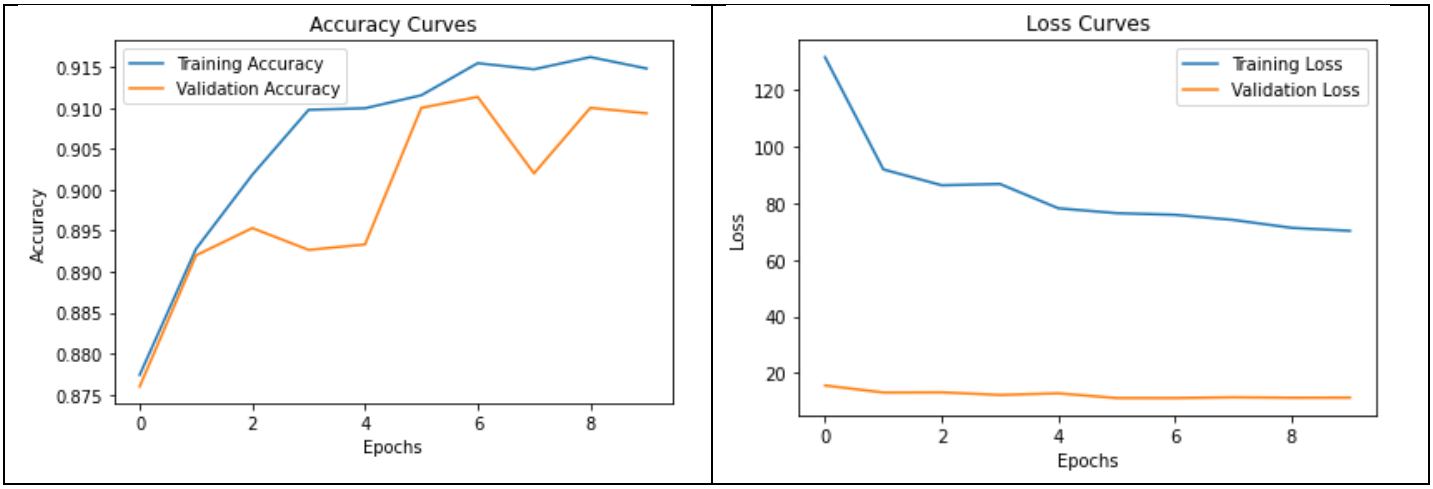
Confusion Matrix

		Predicted Values	
		"infected"	"normal"
Actual Values	"infected"	TP = 587	FN = 28
	"normal"	FP = 21	TN = 864

Experimental Setup

- Learning Rate = 0.001
- Number of fine-tuned fully connected layers = 2
- Neurons in fully connected layers = [400, 2]
- Epochs = 10
- Momentum = 0.9
- Block size = 32

Loss and Accuracy Curves



Final Accuracy and F1 score

- Testing Accuracy = 96%
- F1-Score = 0.97

All the parameters of the models are saved named as “res18_FC_Only.pth”

Task 2: Fine-tune the CNN and FC layers of the network

3. VGG16 with backpropagation of 3 convolutions

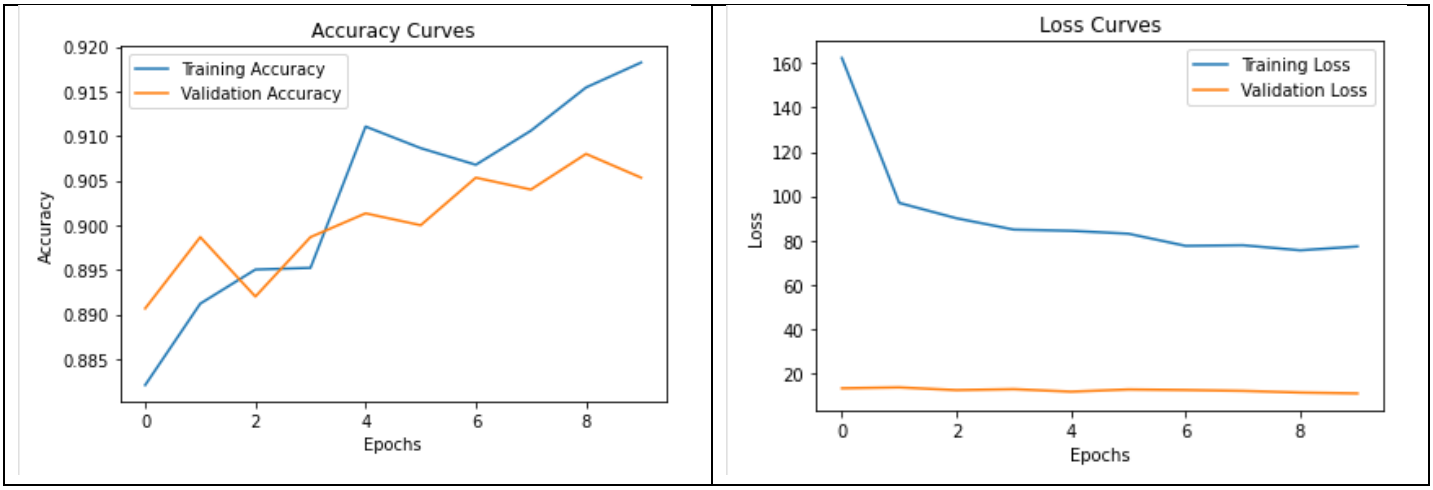
Confusion Matrix

		Predicted Values	
		"infected"	"normal"
Actual Values	"infected"	TP = 584	FN = 31
	"normal"	FP = 14	TN = 871

Experimental Setup

- Learning Rate = 0.001
- Number of fine-tuned fully connected layers = 2
- Neurons in fully connected layers = [400, 2]
- Number of fine-tuned convolution layers = 3
- Epochs = 10
- Momentum = 0.9
- Block size = 32

Loss and Accuracy Curves



Final Accuracy and F1 score

- Testing Accuracy = 97%
- F1-Score = 0.97

All the parameters of the models are saved named as "vgg16_FC_conv3.pth"

4. VGG16 with backpropagation of 7 convolutions

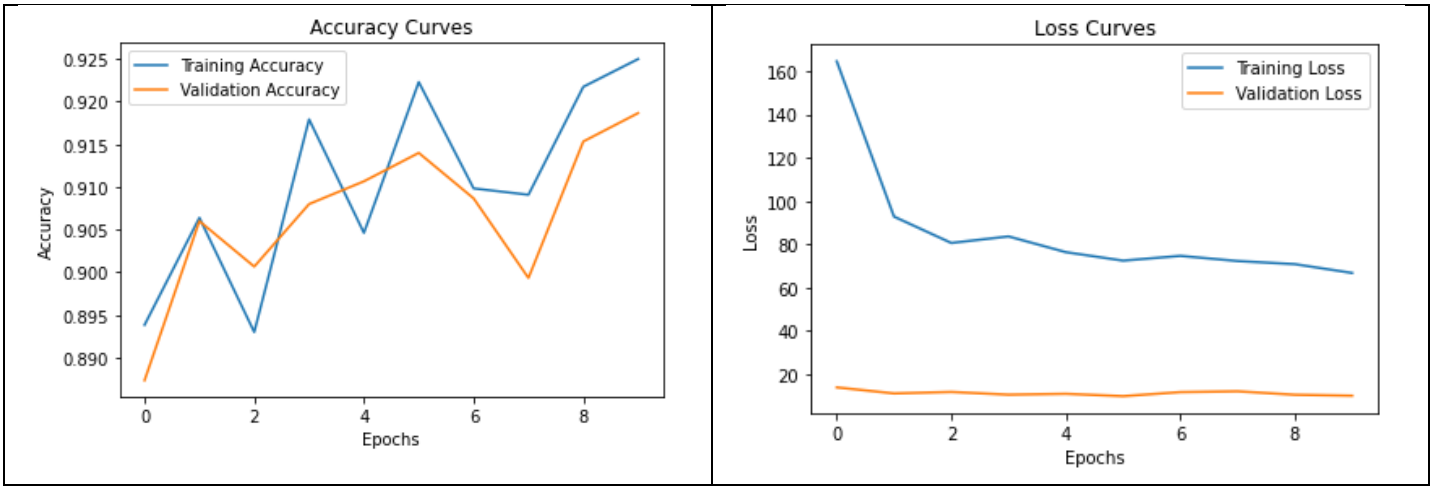
Confusion Matrix

		Predicted Values	
		"infected"	"normal"
Actual Values	"infected"	TP = 590	FN = 25
	"normal"	FP = 20	TN = 865

Experimental Setup

- Learning Rate = 0.001
- Number of fine-tuned fully connected layers = 2
- Neurons in fully connected layers = [400, 2]
- Number of fine-tuned convolution layers = 7
- Epochs = 10
- Momentum = 0.9
- Block size = 32

Loss and Accuracy Curves



Final Accuracy and F1 score

- Testing Accuracy = 97%
- F1-Score = 0.97

All the parameters of the models are saved named as "vgg16_FC_conv7.pth"

5. VGG16 with backpropagation of entire network

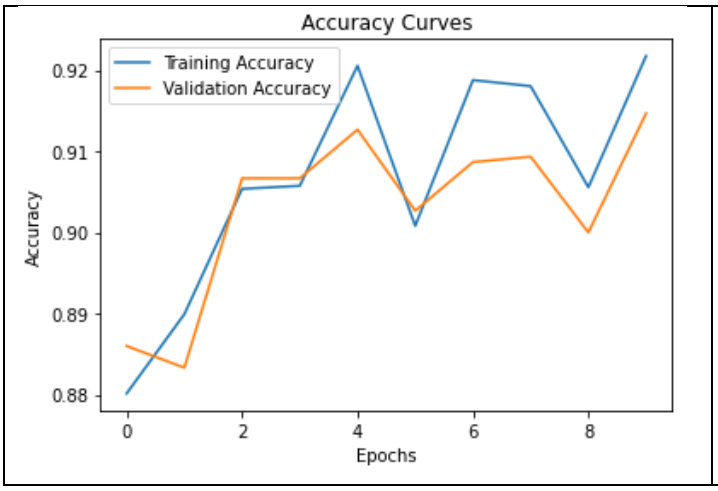
Confusion Matrix

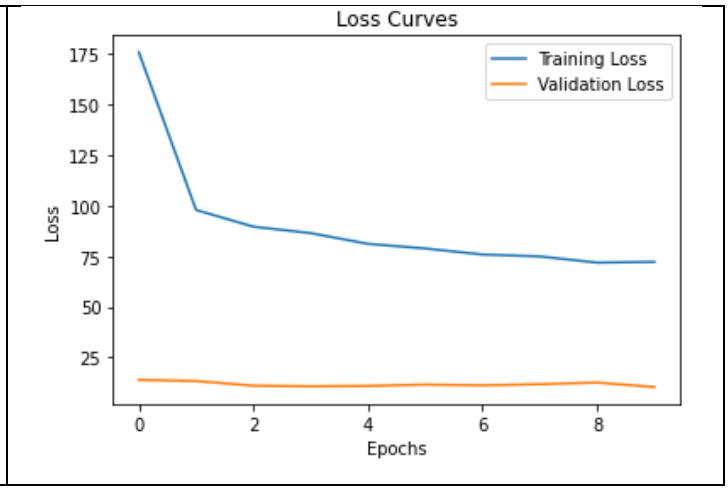
		Predicted Values	
		"infected"	"normal"
Actual Values	"infected"	TP = 594	FN = 21
	"normal"	FP = 14	TN = 871

Experimental Setup

- Learning Rate = 0.001
- Number of fine-tuned fully connected layers = 2
- Neurons in fully connected layers = [400, 2]
- Number of fine-tuned convolution layers = All
- Epochs = 10
- Momentum = 0.9
- Block size = 32

Loss and Accuracy Curves





Final Accuracy and F1 score

- Testing Accuracy = 97%
- F1-Score = 0.98

All the parameters of the models are saved named as "vgg16_FC_entire.pth"

6. VGG16 with backpropagation of entire network without Dropout

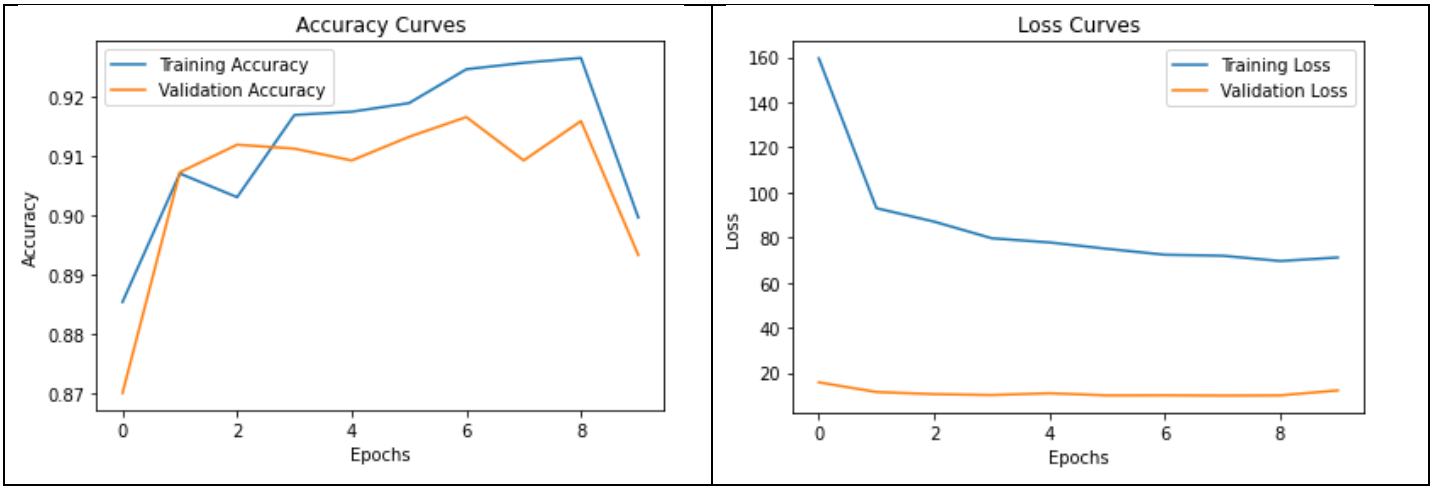
Confusion Matrix

		Predicted Values	
		"infected"	"normal"
Actual Values	"infected"	TP = 545	FN = 70
	"normal"	FP = 2	TN = 883

Experimental Setup

- Learning Rate = 0.001
- Number of fine-tuned fully connected layers = 2
- Neurons in fully connected layers = [400, 2]
- Number of fine-tuned convolution layers = All
- Epochs = 10
- Momentum = 0.9
- Block size = 32

Loss and Accuracy Curves



Final Accuracy and F1 score

- Testing Accuracy = 95%
- F1-Score = 0.96

All the parameters of the models are saved named as "vgg16_FC_entire_wd.pth"

7. RES18 with backpropagation of 1 complete layer of convolutions

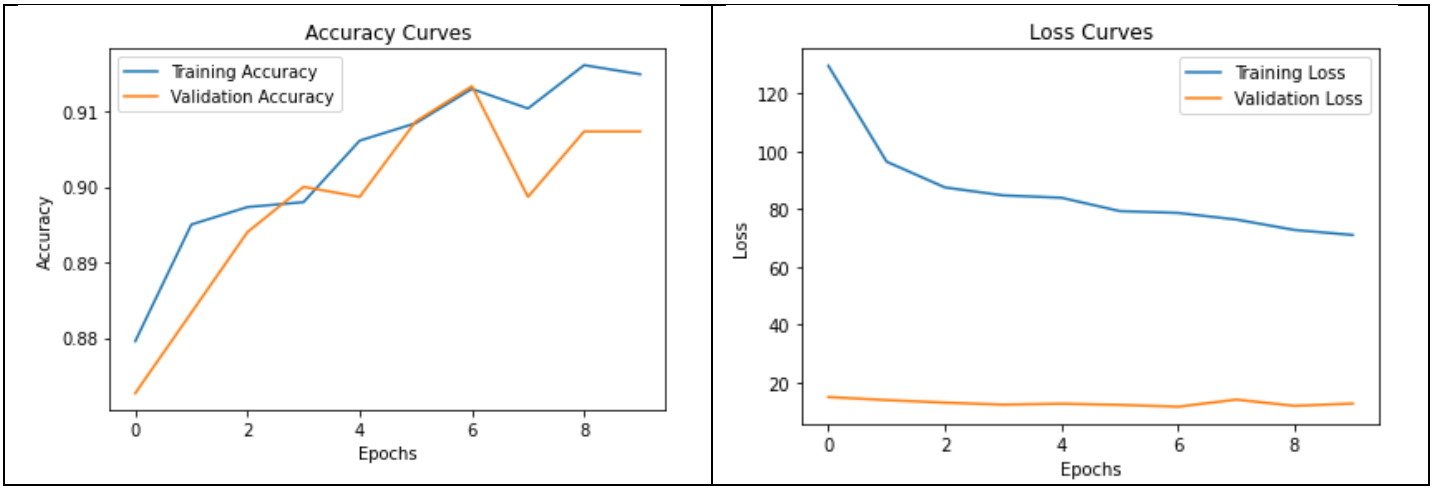
Confusion Matrix

		Predicted Values	
		"infected"	"normal"
Actual Values	"infected"	TP = 585	FN = 30
	"normal"	FP = 37	TN = 848

Experimental Setup

- Learning Rate = 0.001
- Number of fine-tuned fully connected layers = 2
- Neurons in fully connected layers = [400, 2]
- Number of fine-tuned convolution layers = Last complete block of convolution
- Epochs = 10
- Momentum = 0.9
- Block size = 32

Loss and Accuracy Curves



Final Accuracy and F1 score

- Testing Accuracy = 95%
- F1-Score = 0.96

~~All the parameters of the models are saved named as "vgg16_FC_conv1.pth"~~

---

8. RES18 with backpropagation of 2 complete layers of convolutions

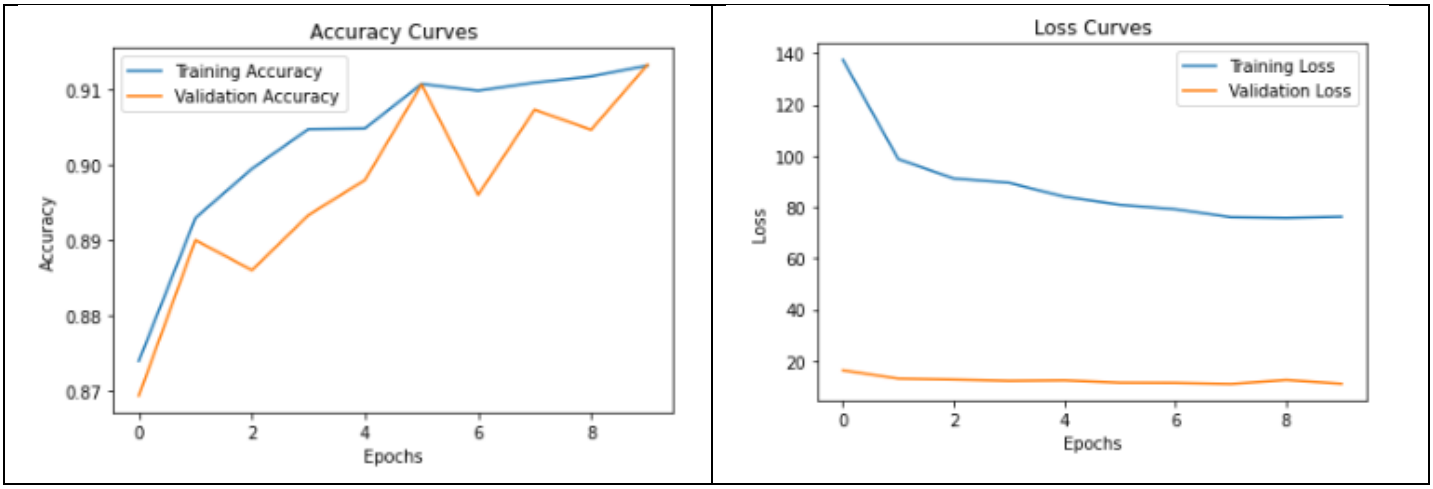
Confusion Matrix

		Predicted Values	
		"infected"	"normal"
Actual Values	"infected"	TP = 583	FN = 32
	"normal"	FP = 12	TN = 873

Experimental Setup

- Learning Rate = 0.001
- Number of fine-tuned fully connected layers = 2
- Neurons in fully connected layers = [400, 2]
- Number of fine-tuned convolution layers = Last 2 blocks of convolution
- Epochs = 10
- Momentum = 0.9
- Block size = 32

Loss and Accuracy Curves



Final Accuracy and F1 score

- Testing Accuracy = 97%
- F1-Score = 0.97

~~All the parameters of the models are saved named as "vgg16_FC_conv2.pth"~~

9. RES18 with backpropagation of 3 complete layers of convolutions

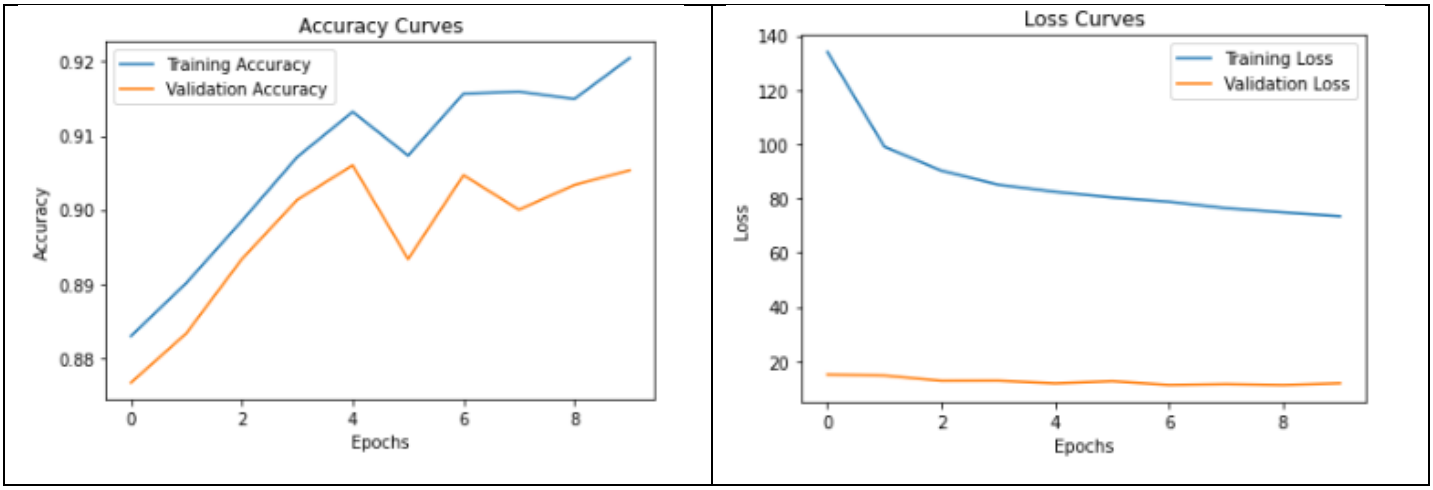
Confusion Matrix

		Predicted Values	
		"infected"	"normal"
Actual Values	"infected"	TP = 587	FN = 28
	"normal"	FP = 21	TN = 864

Experimental Setup

- Learning Rate = 0.001
- Number of fine-tuned fully connected layers = 2
- Neurons in fully connected layers = [400, 2]
- Number of fine-tuned convolution layers = Last 3 blocks of convolution
- Epochs = 10
- Momentum = 0.9
- Block size = 32

Loss and Accuracy Curves



Final Accuracy and F1 score

- Testing Accuracy = 96%
- F1-Score = 0.97

~~All the parameters of the models are saved named as "vgg16_FC_conv3.pth"~~

10. RES18 with backpropagation of entire network

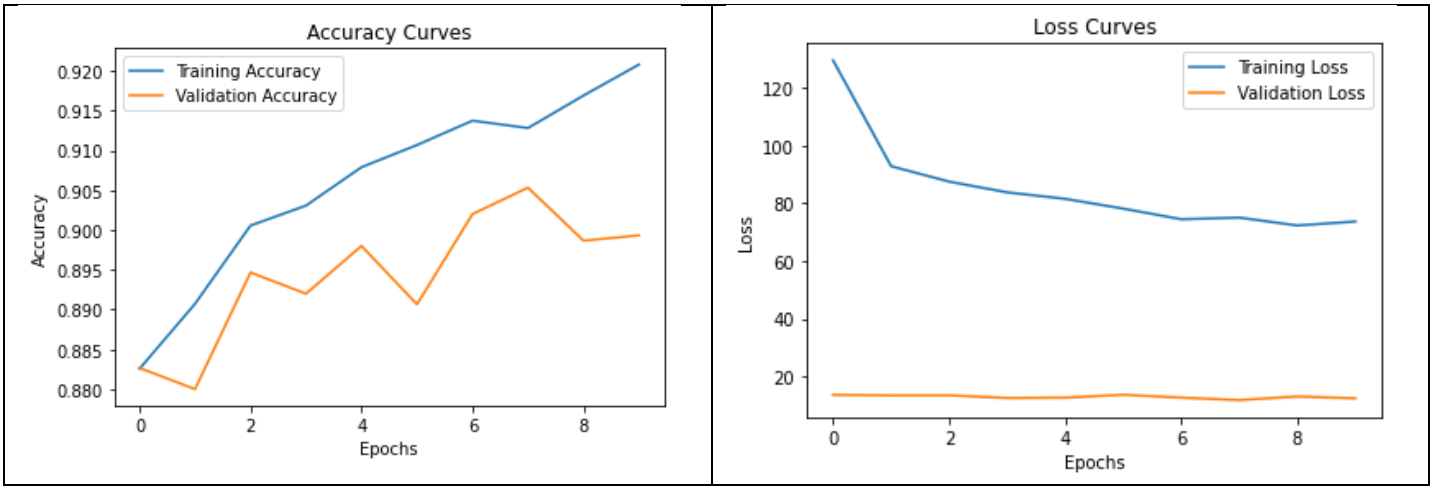
Confusion Matrix

		Predicted Values	
		"infected"	"normal"
Actual Values	"infected"	TP = 586	FN = 29
	"normal"	FP = 23	TN = 862

Experimental Setup

- Learning Rate = 0.001
- Number of fine-tuned fully connected layers = 2
- Neurons in fully connected layers = [400, 2]
- Number of fine-tuned convolution layers = All blocks of convolution
- Epochs = 10
- Momentum = 0.9
- Block size = 32

Loss and Accuracy Curves



Final Accuracy and F1 score

- Testing Accuracy = 96%
- F1-Score = 0.97

~~All the parameters of the models are saved named as "vgg16_FC_entire.pth"~~

11. RES18 with backpropagation of entire network without Dropout

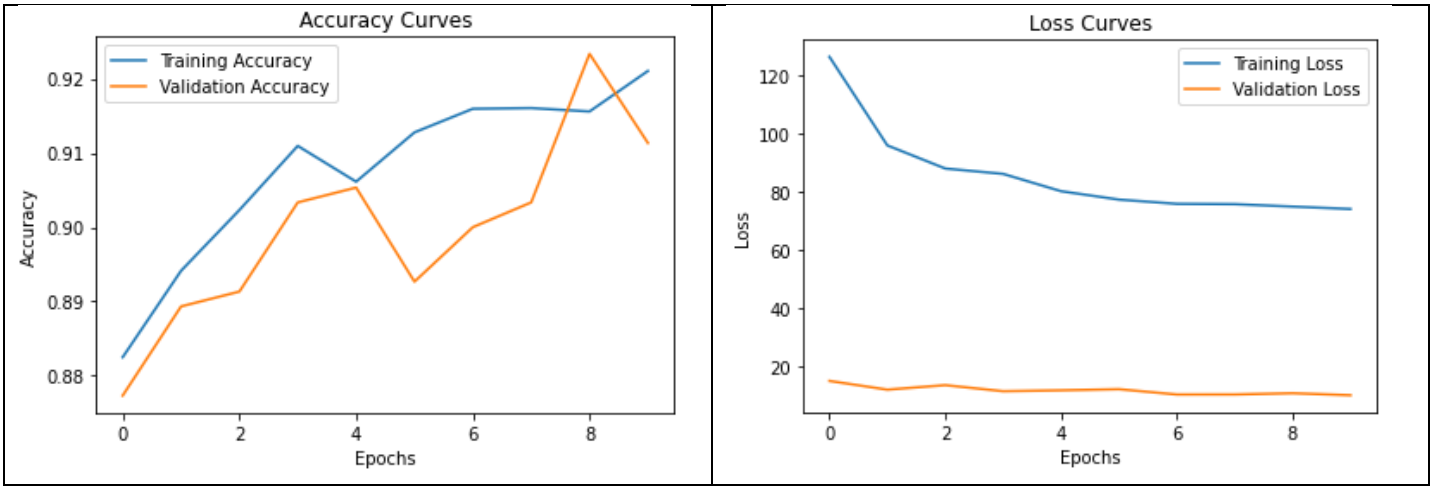
Confusion Matrix

		Predicted Values	
		"infected"	"normal"
Actual Values	"infected"	TP = 585	FN = 30
	"normal"	FP = 25	TN = 860

Experimental Setup

- Learning Rate = 0.001
- Number of fine-tuned fully connected layers = 2
- Neurons in fully connected layers = [400, 2]
- Number of fine-tuned convolution layers = All blocks of convolution
- Epochs = 10
- Momentum = 0.9
- Block size = 32

Loss and Accuracy Curves



Final Accuracy and F1 score

- Testing Accuracy = 96%
- F1-Score = 0.96

All the parameters of the models are saved named as "vgg16_FC_entire_wd.pth"

Comparison

Sr.	Architecture	No of training of convolution layers	Dropout in FC layers	Testing Accuracy	F1 Score
1	VGG16	0	0.5	94%	.94
2	RES18	0	0.5	96%	.97
3	VGG16	3	0.5	97%	.97
4	VGG16	7	0.5	97%	.97
5	VGG16	all	0.5	97%	.98
6	VGG16	all	0	95%	.96
7	RES18	1 block	0.5	95%	.96
8	RES18	2 block	0.5	97%	.97
9	RES18	3 block	0.5	96%	.97
10	RES18	all	0.5	96%	.97
11	RES18	all	0	96%	.96

Conclusion

VGG and ResNet are very good architecture and have high accuracy for images of ImageNet. In this assignment, we were asked to use pre-trained models of VGG16 and RES18.The FC layers was modified to two FC layers having 400 and 2 neurons respectively. The backpropagation of only FC layers also provide the accuracy of 96% but it can be improved by back-propagating the error to the convolution layers also. I tried to do 11 different experiments. For task 1, I did not change any other parameter provided in tutorial but the batch size. I change the batch size to 32 and that's work pretty good. The accuracy for VGG network was 94% and for ResNet was 96%.

For task 2, I've done 4 experiments for VGG network. In the first experiment, I train last 3 convolution layers and FC layers. All remaining parameters were same as in task1. The accuracy improved to 97%. For other 2 experiments where I trained last 7 and all convolution layers respectively with the FC layers and the accuracy was 97% but the F1 score for last experiment where all network was pre-trained was 0.98. I also tried to remove Dropout layer to overfit the model on training images but it reduce the accuracy to 95%. ResNet architecture have convolution layers in blocks, hence I did different experiments by training convolution layers block by block. The accuracy was between 95-97% for these experiments.

Training convolution layers does not have much impact on the accuracy in this assignment. Even I achieve 96% accuracy without backpropagating the convolution layers. Training convolution layers might help where we can't achieve much higher accuracy by simply applying the FC network on the features extracted by the convolution layers.

As this assignment was having so short deadline, I did not get any change to do more experiments by changing the learning rate, momentum value, no of epochs, or by applying different layers. As far for these experiments, I noted that these both networks VGG and ResNet both convolution layers have very powerful features extraction. Even we can increase accuracy and overfit out model by extracting the features and applying reasonable FC layers on that features.