Machine Learning Project

MLathon

NAMAL INSTITUTE

Group Members:

Name	**Roll#**
Ali Raza Khan	2018-UET-NML-CS-23
Muhammad Musawar Baig	2018-UET-NML-CS-03
Saifullah Khan	2018-UET-NML-CS-12

Dated: July 17th, 2022

Submitted to: Dr. Malik Jahan Khan

1st Classifier (AlexNet model):

AlexNet is a convolution neural network (CNN) architecture that was designed by Alex Krizhevsky in collaboration with Ilya Sutskever and Geoffrey Hinton. Geoffrey was Krizhevsky's Ph.D. advisor.

AlexNet was first utilized in the public setting when it won the ImageNet Large Scale Visual Recognition Challenge(ILSSVRC 2012 contest). It was at this contest that AlexNet showed that a deep convolutional neural network can be used for solving image classification (Alake 1). In our project, We are using AlexNet for the classification of healthy and infected pepper.

2nd Classifier (ArkNet model):

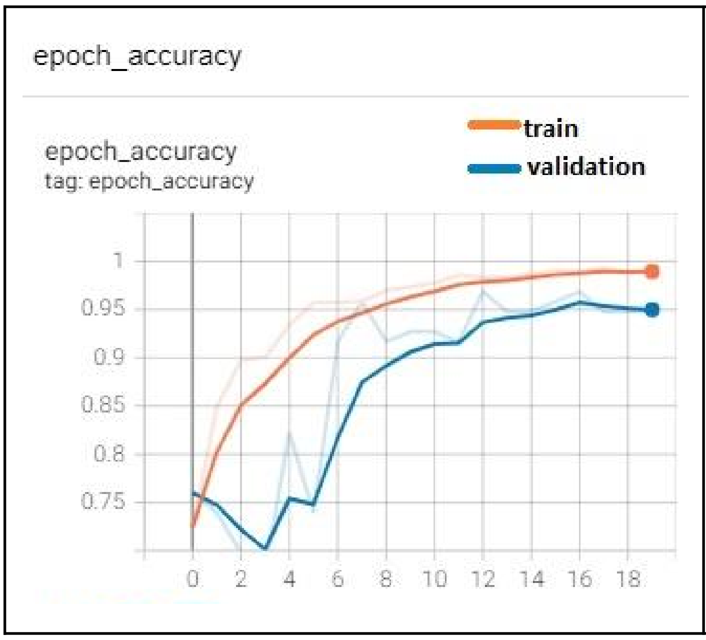
ArkNet is designed/written by us. Some of the layers and activations are inspired by the VGG16 neural network designed and proposed by K. Simonyan and A. Zisserman. VGG16 is an exhaustive CNN model used for the classification. So, instead of using the same model, we have customized the VGG16 model to our needs. ArkNet consists of 3-convolutional, 2 dense, 3-MaxPooling, and a flatten-layer followed by the output layer. We have done 15 epochs for training the model.

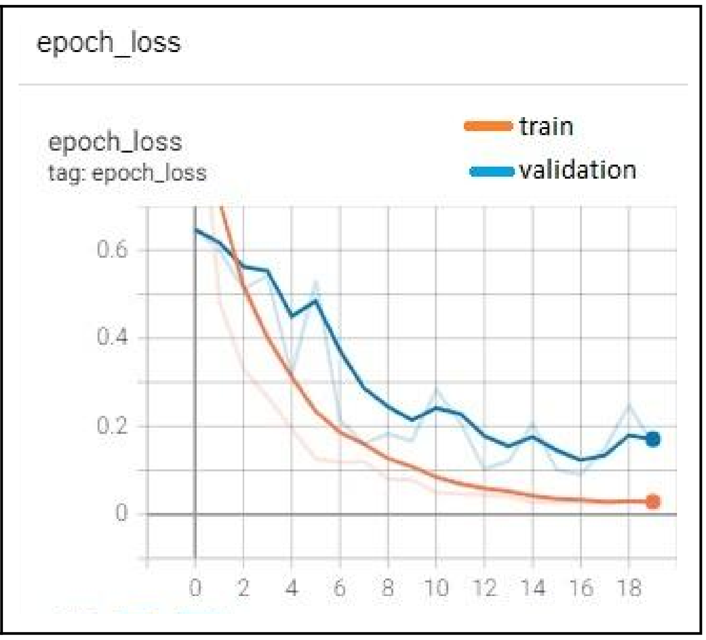
Comparison between AlexNet model and ArkNet (w.r.t evaluation measures):

We have used 50% positive and 50% negative training examples for the following experimentations. Following graphs and charts are used for exhaustive comparison models of AlexNet and ArkNet by analyzing accuracy, recall, and precision:

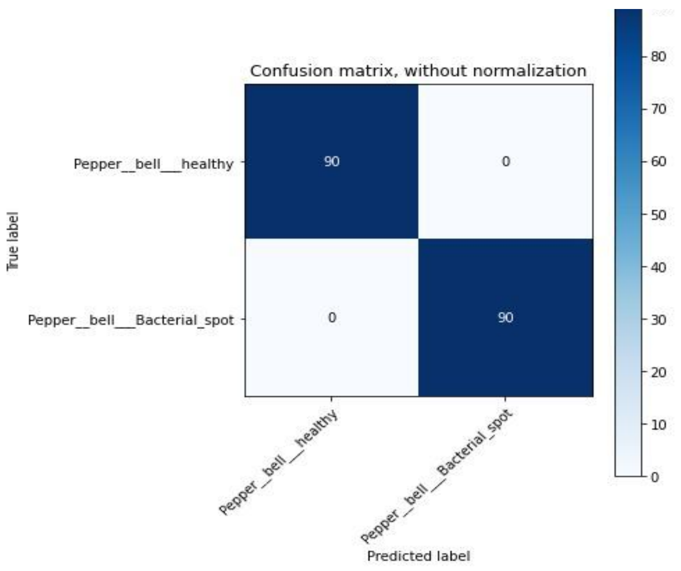
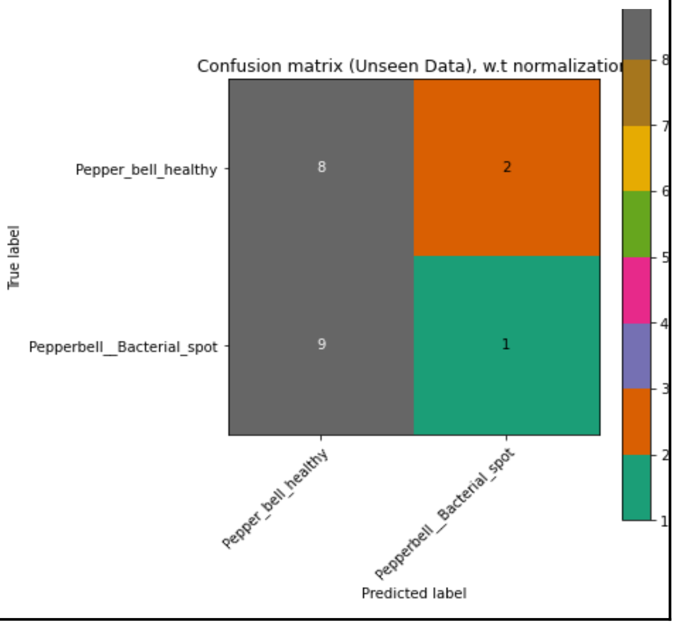
1st Classifier(AlexNet model):

Accuracy and Loss w.r.t. each Epoch while training:

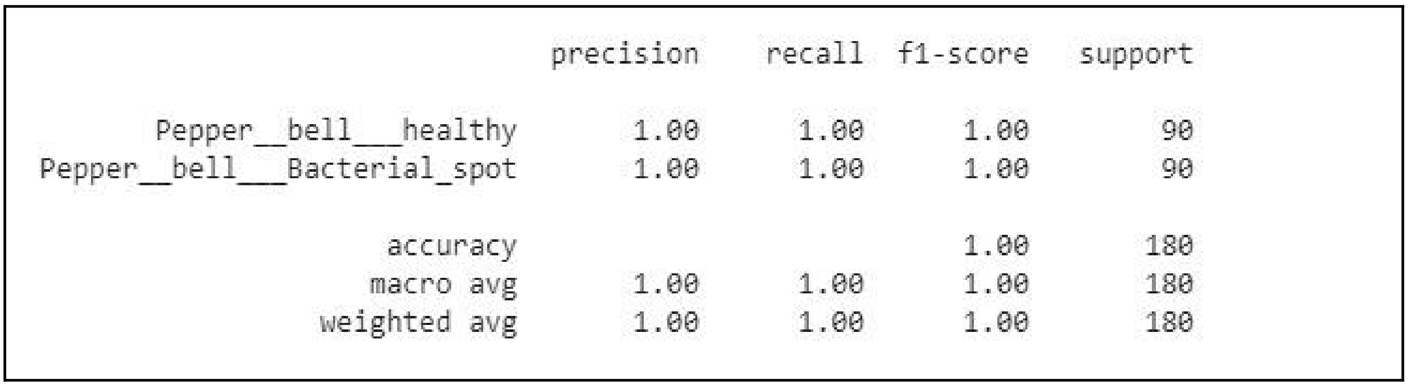




Confusion Matrices on Kaggle and Raw testing Data:

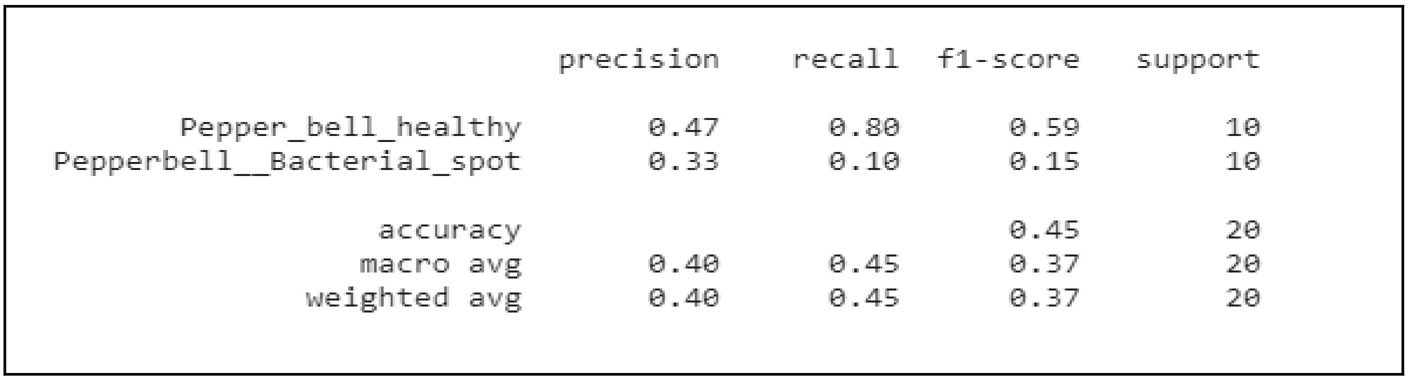
Confusion Matrix on Test Data from Kaggle Dataset	Confusion Matrix on Unseen Raw Data from web
	

Evaluation Measures(On Testing Data):



```
                              precision    recall  f1-score   support

      Pepper__bell___healthy       1.00      1.00      1.00        90
Pepper__bell___Bacterial_spot       1.00      1.00      1.00        90

                    accuracy                           1.00       180
                   macro avg       1.00      1.00      1.00       180
                weighted avg       1.00      1.00      1.00       180
```

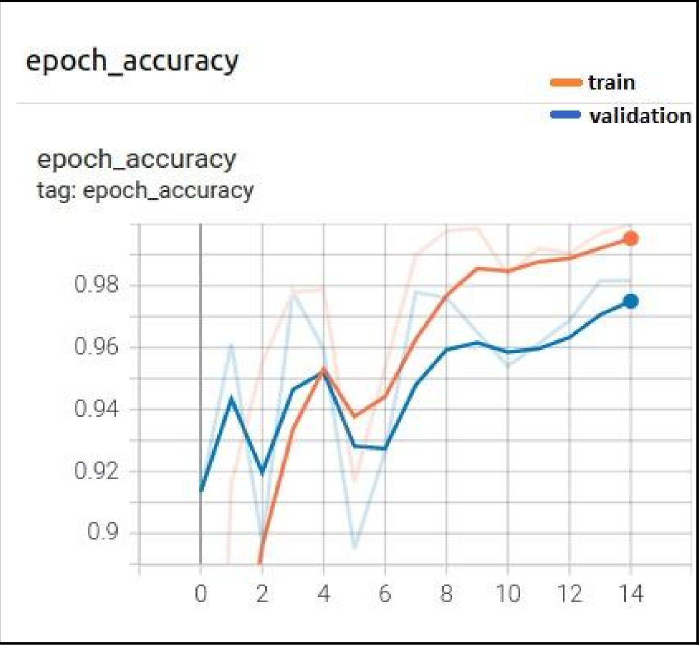
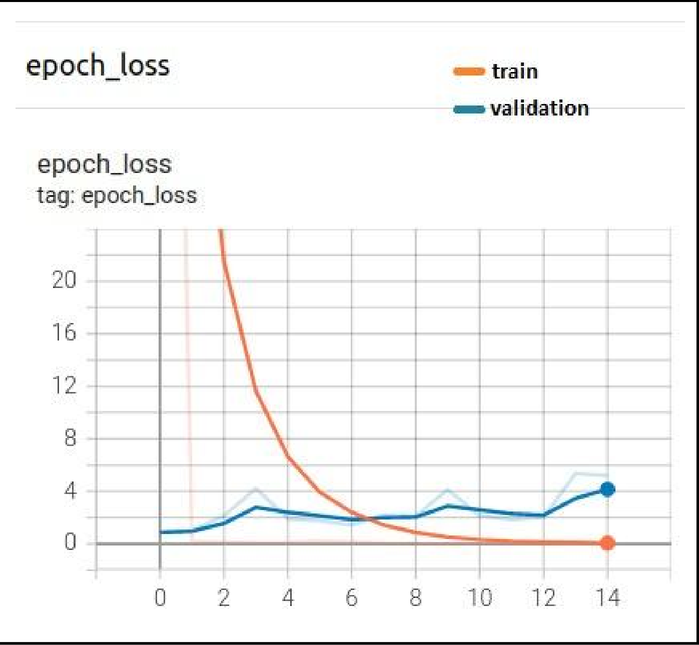
Evaluation Measures(On Unseen/Raw Data):



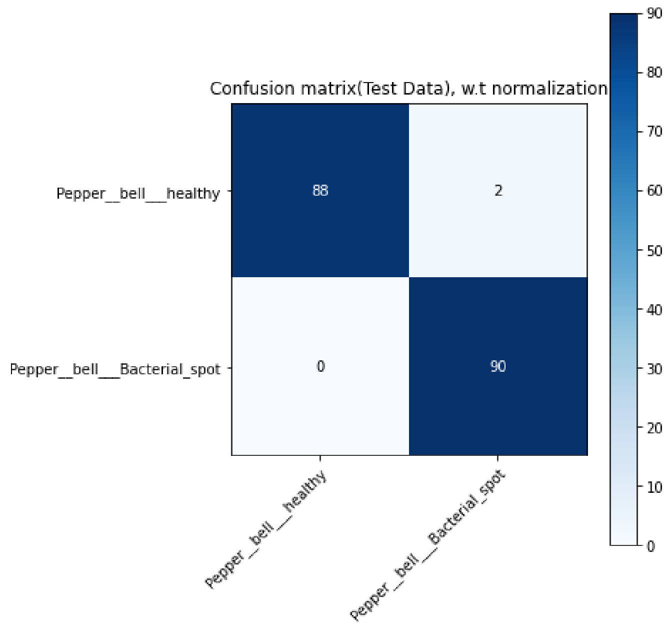
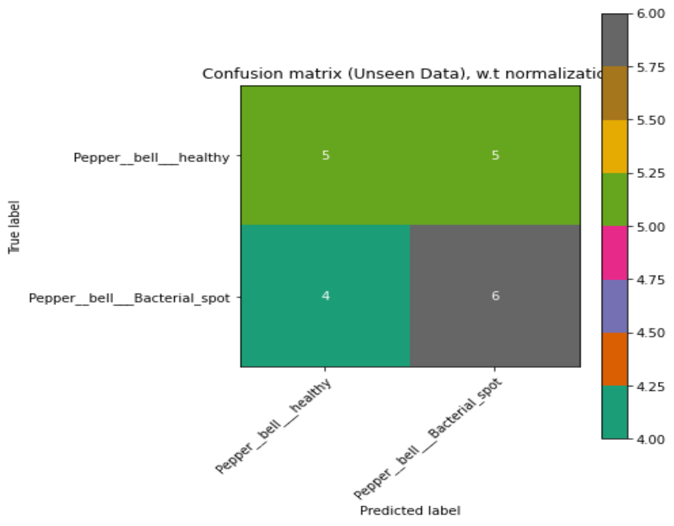
```
                            precision    recall  f1-score   support

       Pepper_bell_healthy       0.47      0.80      0.59        10
Pepperbell__Bacterial_spot       0.33      0.10      0.15        10

                  accuracy                           0.45        20
                 macro avg       0.40      0.45      0.37        20
              weighted avg       0.40      0.45      0.37        20
```

2nd Classifier(ArkNet model):

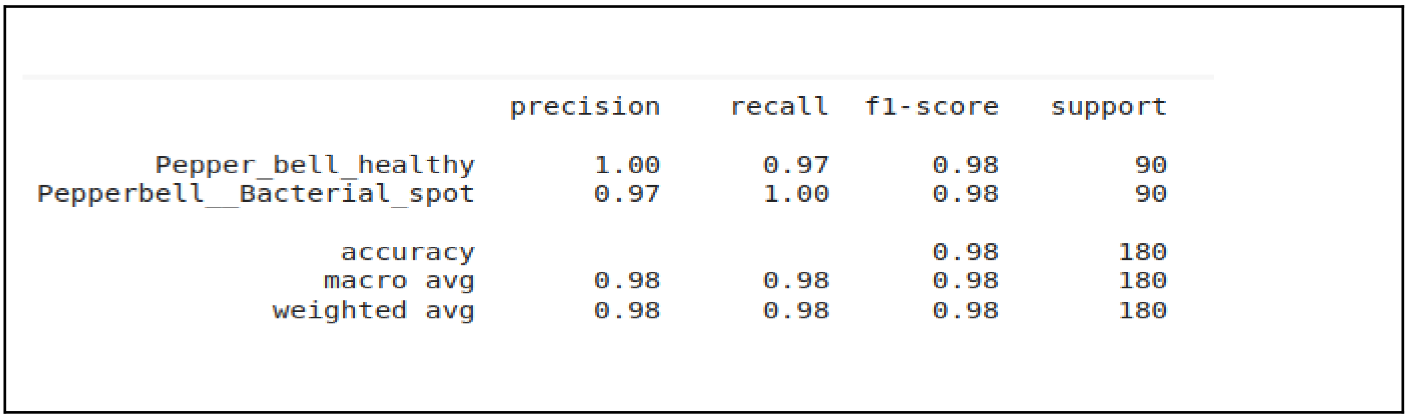
Accuracy and Loss w.r.t. each Epoch while training:

Confusion Matrices on Kaggle and Raw testing Data:

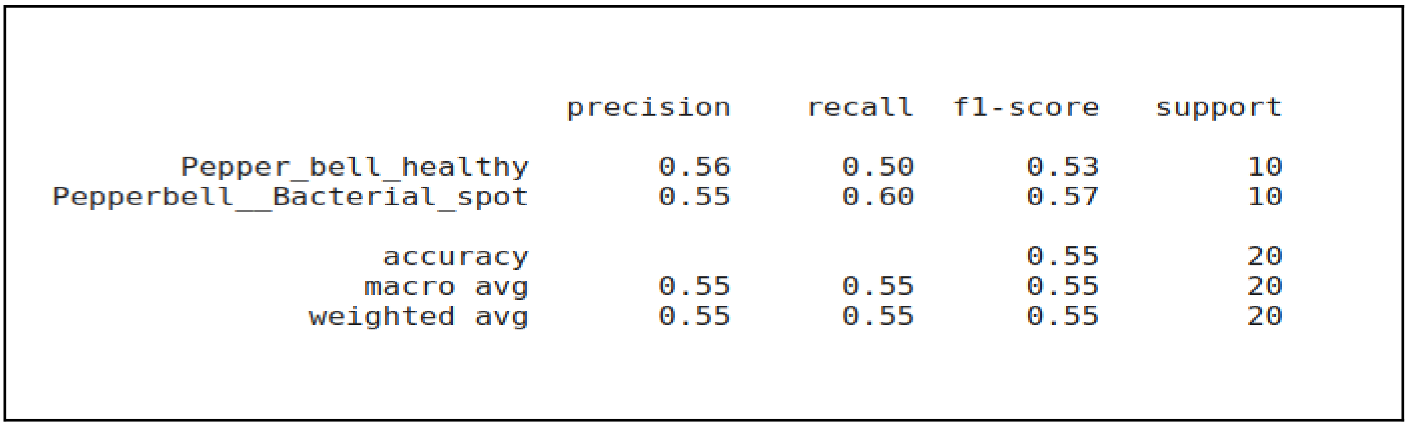
Confusion Matrix on Test Data from Kaggle Dataset	Confusion Matrix on Unseen Raw Data from web
	

Evaluation Measures(On Testing Data):



```
                            precision    recall  f1-score   support

       Pepper_bell_healthy       1.00      0.97      0.98        90
Pepperbell__Bacterial_spot       0.97      1.00      0.98        90

                  accuracy                           0.98       180
                 macro avg       0.98      0.98      0.98       180
              weighted avg       0.98      0.98      0.98       180
```

Evaluation Measures(On Unseen/Raw Data):



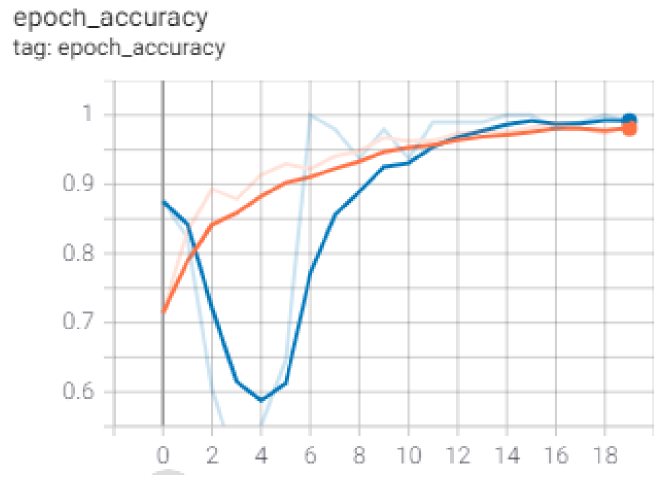
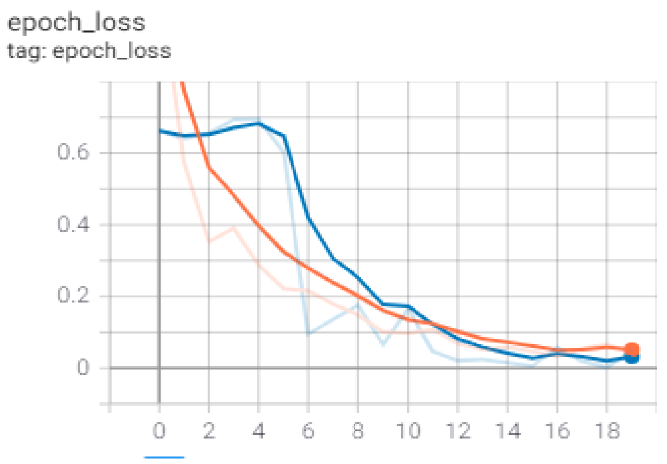
```
                            precision    recall  f1-score   support

       Pepper_bell_healthy       0.56      0.50      0.53        10
Pepperbell__Bacterial_spot       0.55      0.60      0.57        10

                  accuracy                           0.55        20
                 macro avg       0.55      0.55      0.55        20
              weighted avg       0.55      0.55      0.55        20
```

Comparison between AlexNet model and ArkNet model (w.r.t data splits):

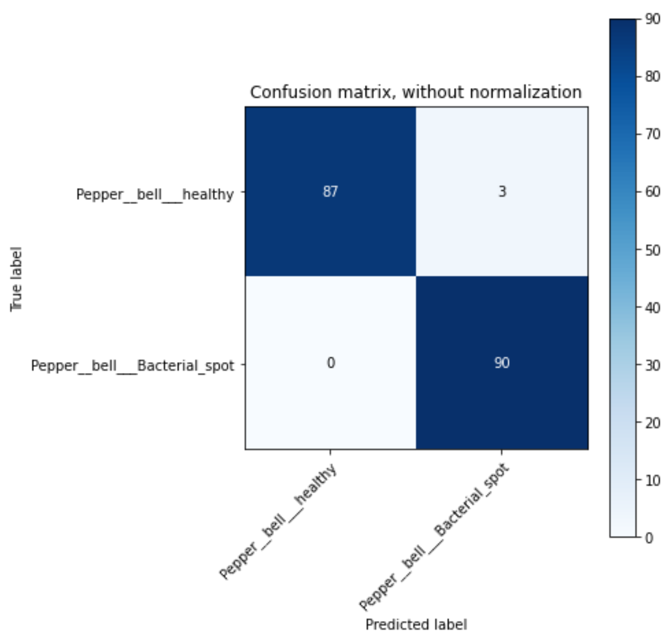
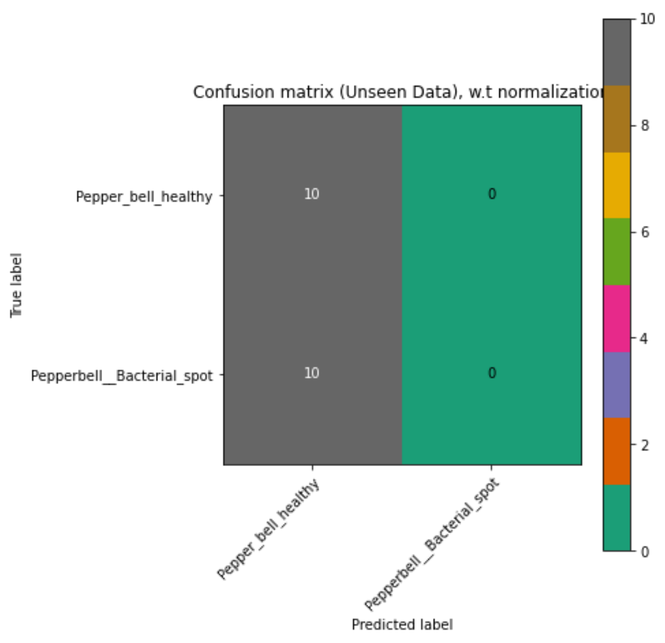
AlexNet:

75% positive and 25% negative training examples:

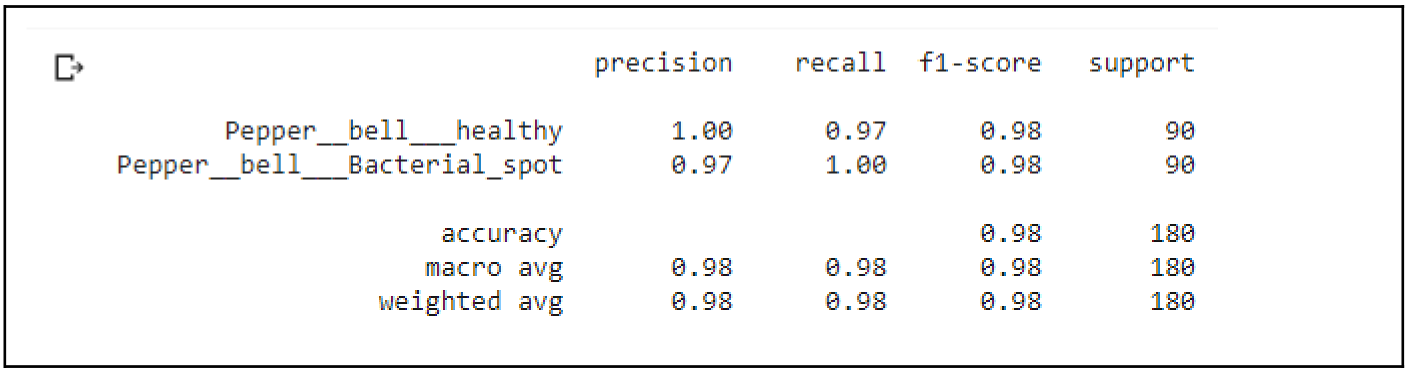
Accuracy and Loss w.r.t. each Epoch while training:

 Orange ->train Blue -> Validation	 Orange ->train Blue -> Validation

Confusion Matrices:

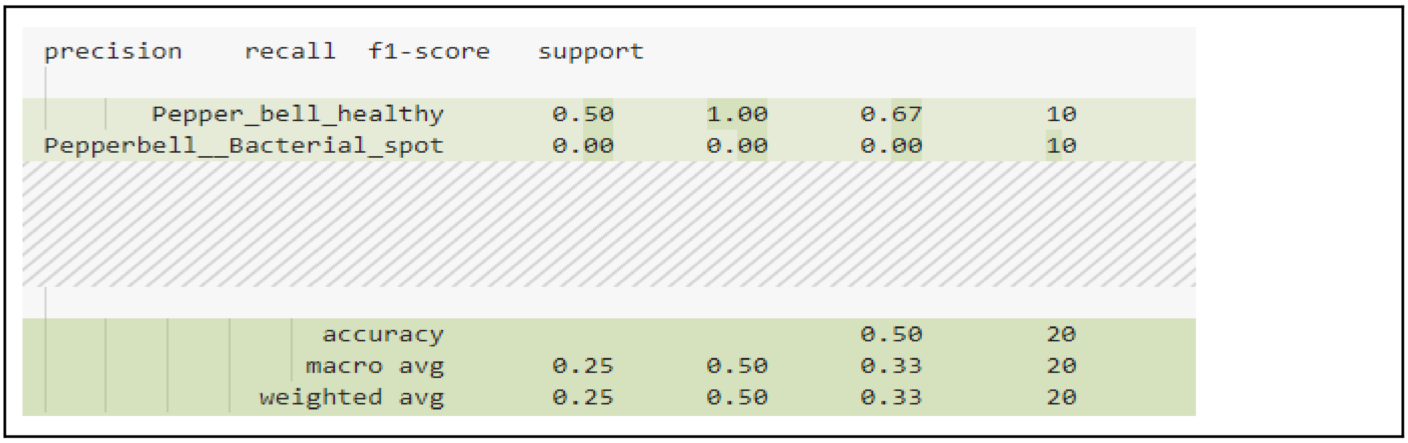
Confusion Matrix on Test Data from Kaggle Dataset	Confusion Matrix on Unseen Raw Data from web
	

Evaluation Measures(On Testing Data):



```
                                  precision    recall  f1-score   support

          Pepper__bell___healthy       1.00      0.97      0.98        90
   Pepper__bell___Bacterial_spot       0.97      1.00      0.98        90

                        accuracy                           0.98       180
                       macro avg       0.98      0.98      0.98       180
                    weighted avg       0.98      0.98      0.98       180
```

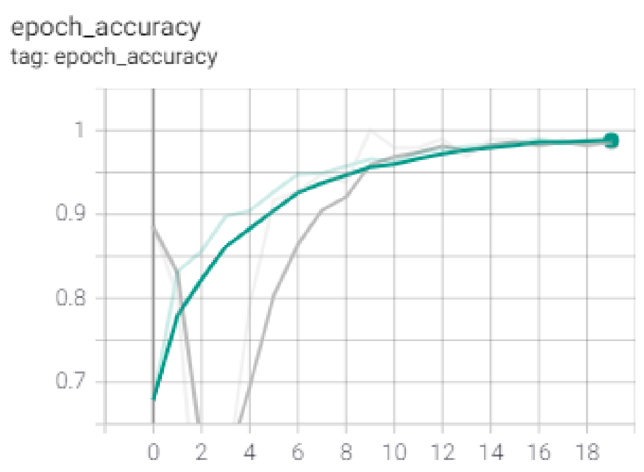
Evaluation Measures(On Unseen/Raw Data):



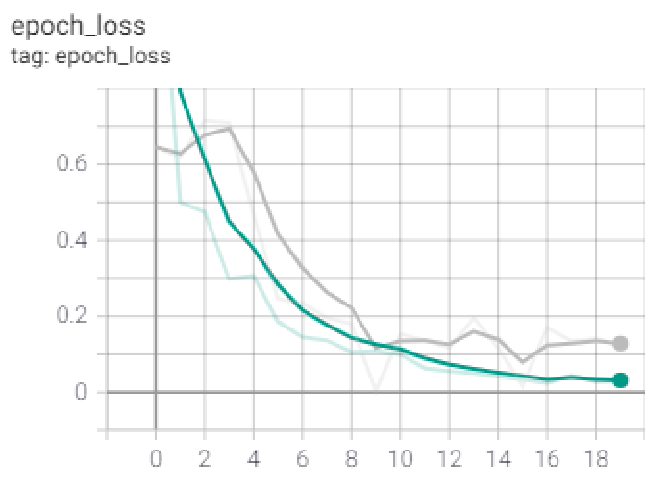
```
precision    recall  f1-score   support

       Pepper_bell_healthy      0.50      1.00      0.67        10
Pepperbell__Bacterial_spot      0.00      0.00      0.00        10

                  accuracy                          0.50        20
                 macro avg      0.25      0.50      0.33        20
              weighted avg      0.25      0.50      0.33        20
```

60% positive and 40% negative training examples:

Accuracy and Loss w.r.t. each Epoch while training:

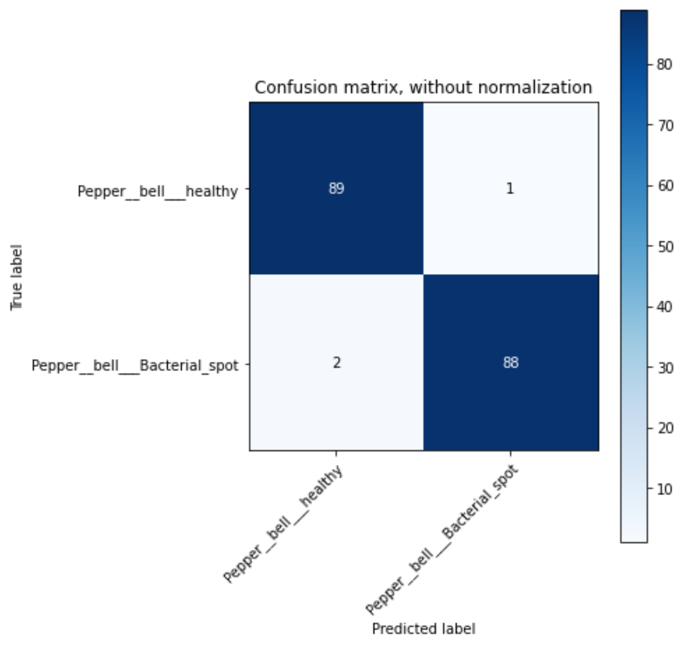
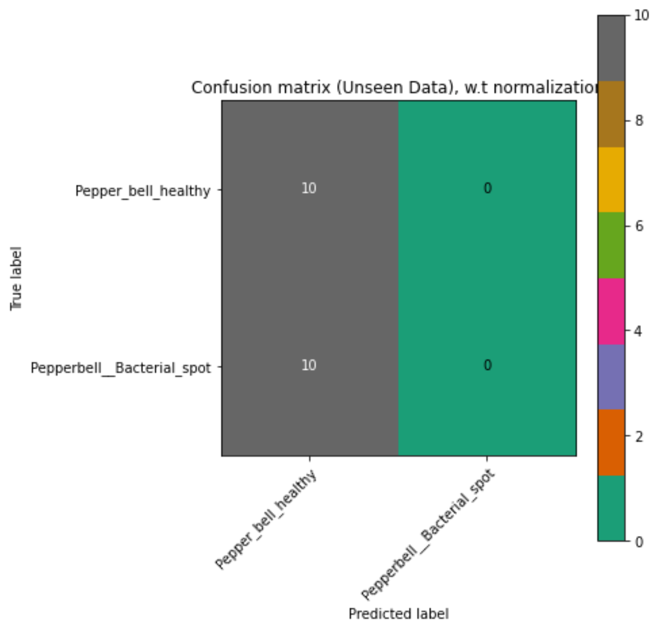


Blue -> train
Gray -> validation

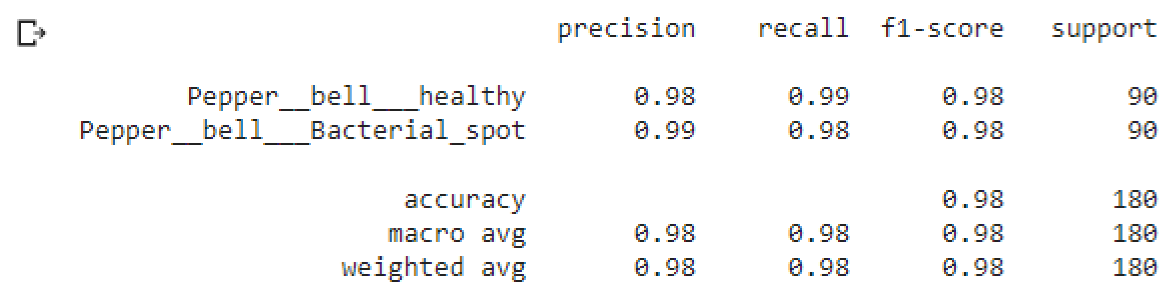


Blue -> train
Gray -> validation

Confusion Matrix:

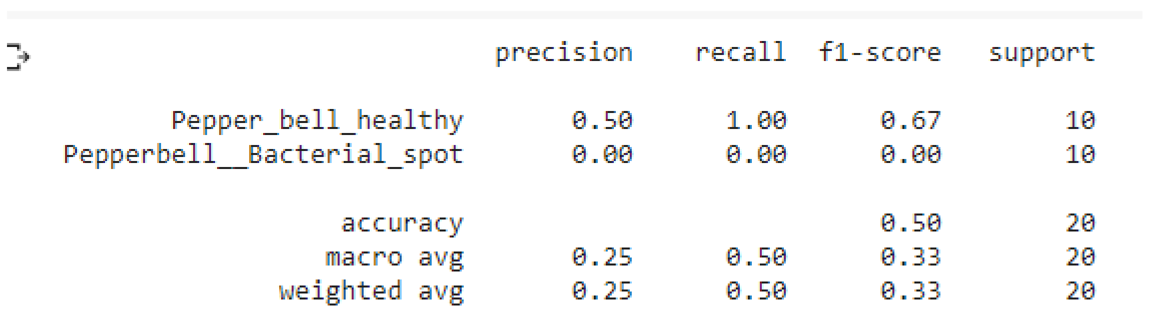
Confusion Matrix on Test Data from Kaggle Dataset	Confusion Matrix on Unseen Raw Data from web
	

Evaluation Measures(On Testing Data Kaggle):



```
                              precision    recall  f1-score   support

       Pepper__bell___healthy      0.98      0.99      0.98        90
  Pepper__bell___Bacterial_spot      0.99      0.98      0.98        90

                    accuracy                           0.98       180
                   macro avg       0.98      0.98      0.98       180
                weighted avg       0.98      0.98      0.98       180
```

Evaluation Measures(On Unseen/Raw Data):

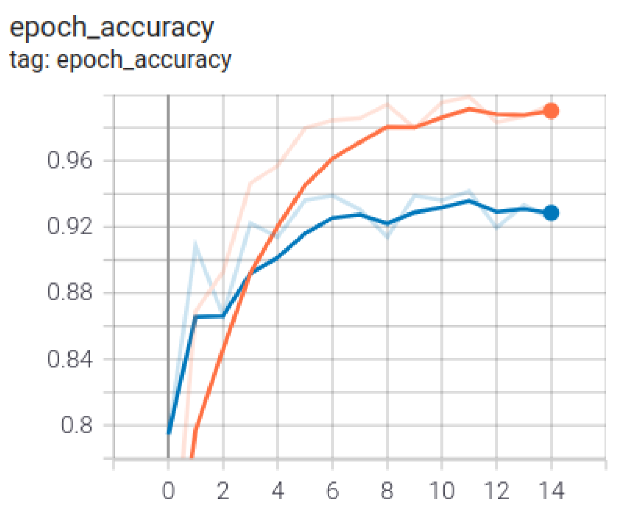
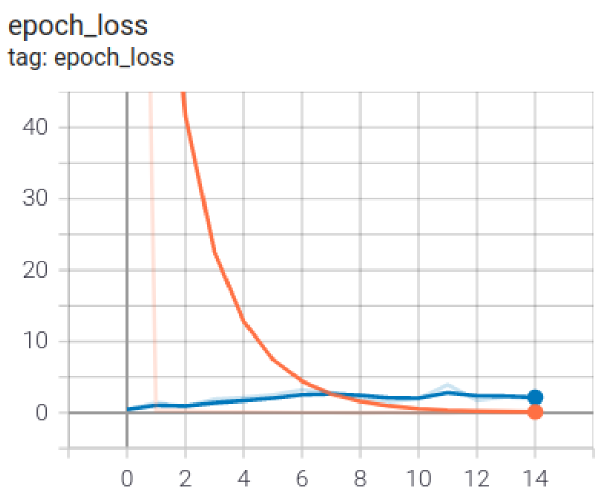


```
                              precision    recall  f1-score   support

         Pepper_bell_healthy       0.50      1.00      0.67        10
  Pepperbell__Bacterial_spot       0.00      0.00      0.00        10

                    accuracy                           0.50        20
                   macro avg       0.25      0.50      0.33        20
                weighted avg       0.25      0.50      0.33        20
```

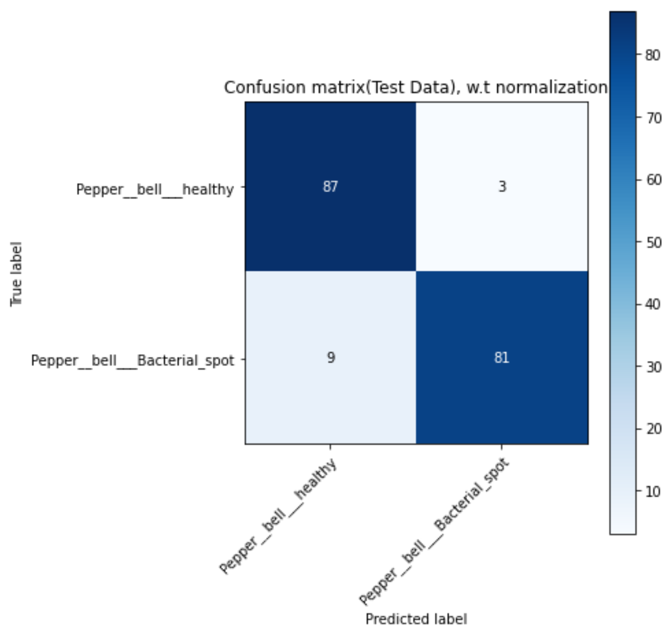
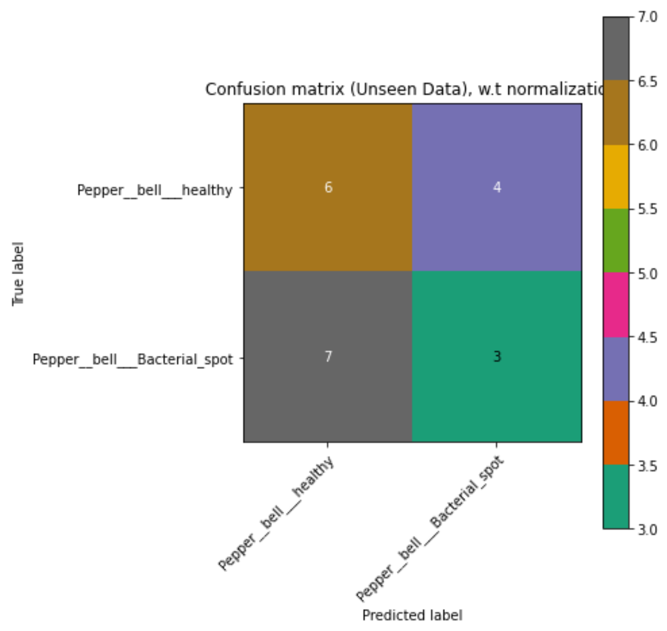
ArkNet:

75% positive and 25% negative training examples:

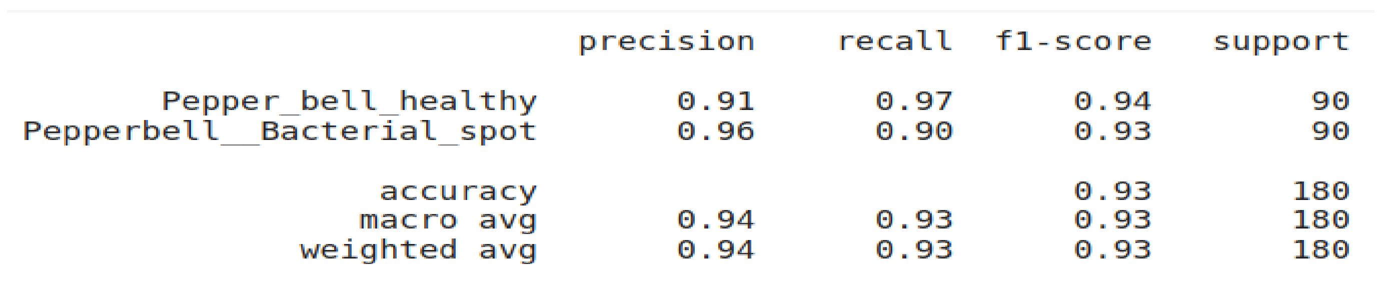
Accuracy and Loss w.r.t. each Epoch while training:

 Orange ->train Blue -> Validation	 Orange ->train Blue -> Validation

Confusion Matrices::

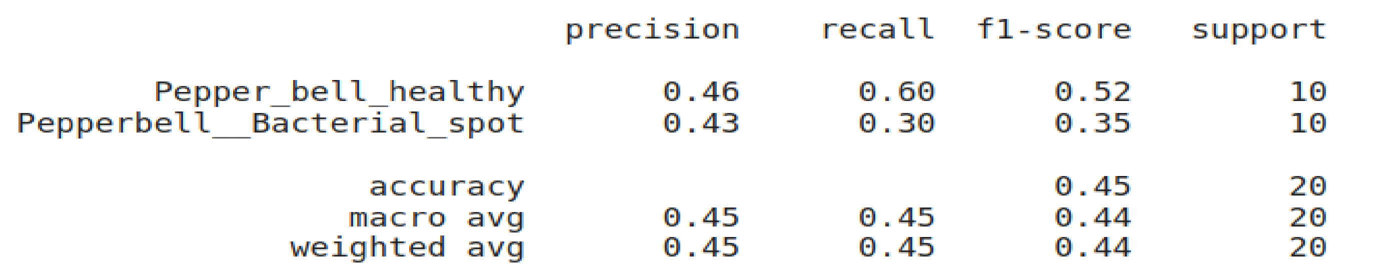
Confusion Matrix on Test Data from Kaggle Dataset	Confusion Matrix on Unseen Raw Data from web
	

Evaluation Measures(On Testing Data):



```
                          precision    recall  f1-score   support

      Pepper_bell_healthy      0.91      0.97      0.94        90
Pepperbell__Bacterial_spot     0.96      0.90      0.93        90

                 accuracy                          0.93       180
                macro avg      0.94      0.93      0.93       180
             weighted avg      0.94      0.93      0.93       180
```

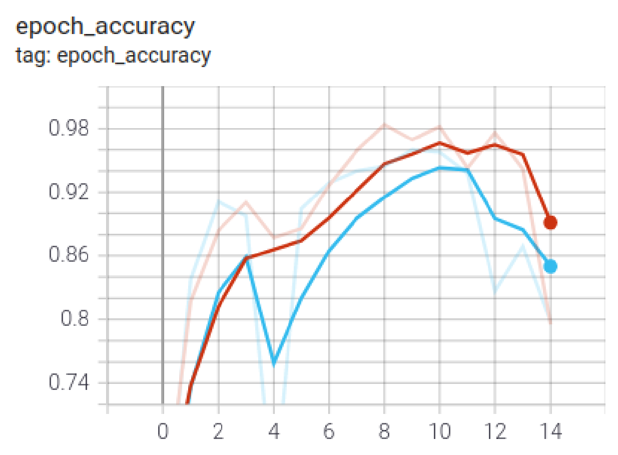
Evaluation Measures(On Unseen/Raw Data):



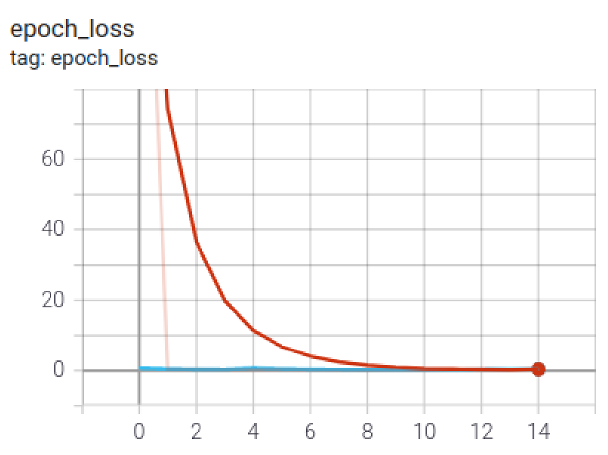
```
                          precision    recall  f1-score   support

      Pepper_bell_healthy      0.46      0.60      0.52        10
Pepperbell__Bacterial_spot     0.43      0.30      0.35        10

                 accuracy                          0.45        20
                macro avg      0.45      0.45      0.44        20
             weighted avg      0.45      0.45      0.44        20
```

60% positive and 40% negative training examples:

Accuracy and Loss w.r.t. each Epoch while training:

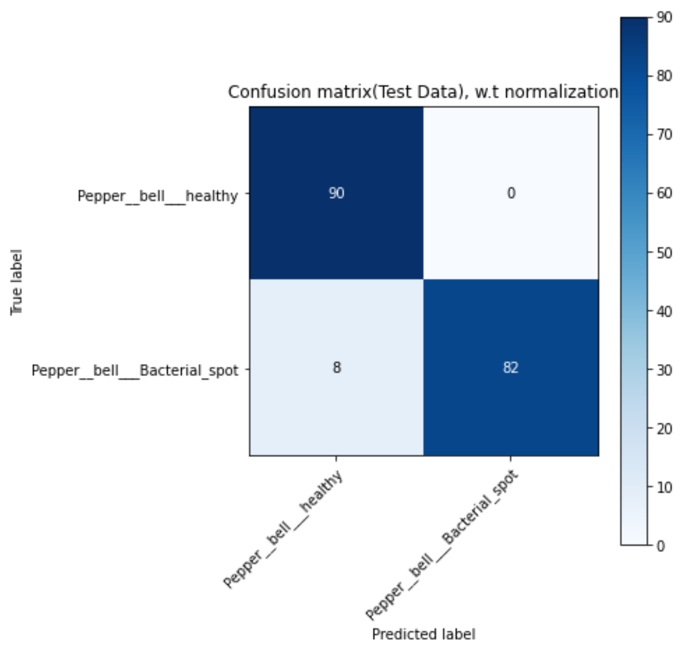
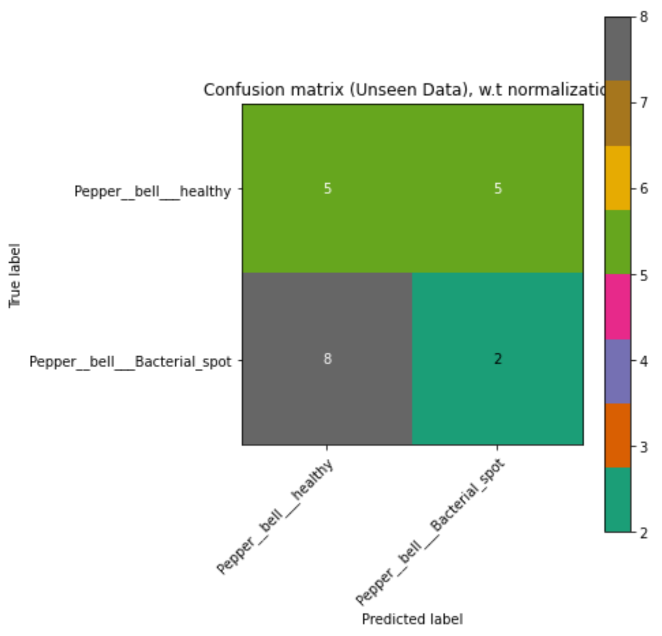


Red --> train
Blue --> validation

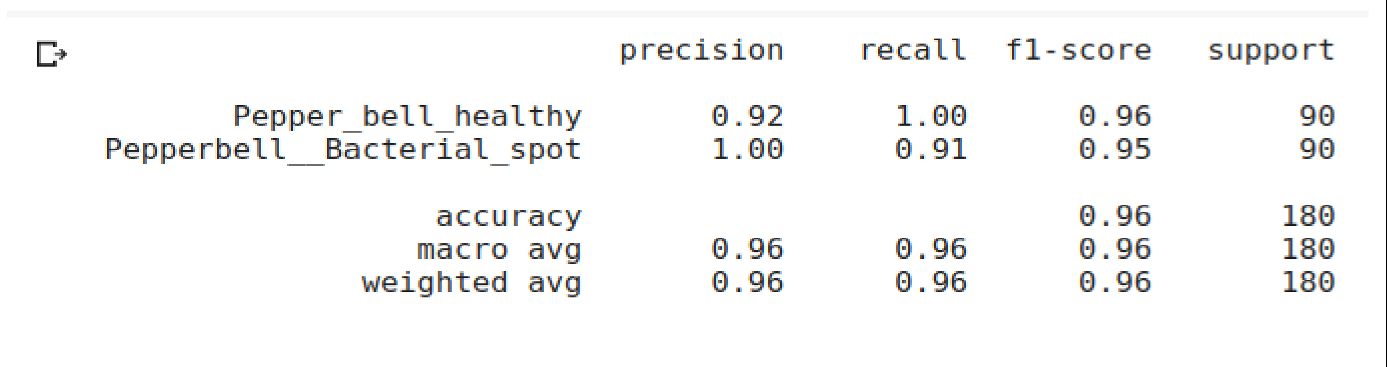


Red --> train
Blue --> validation

Confusion Matrices:

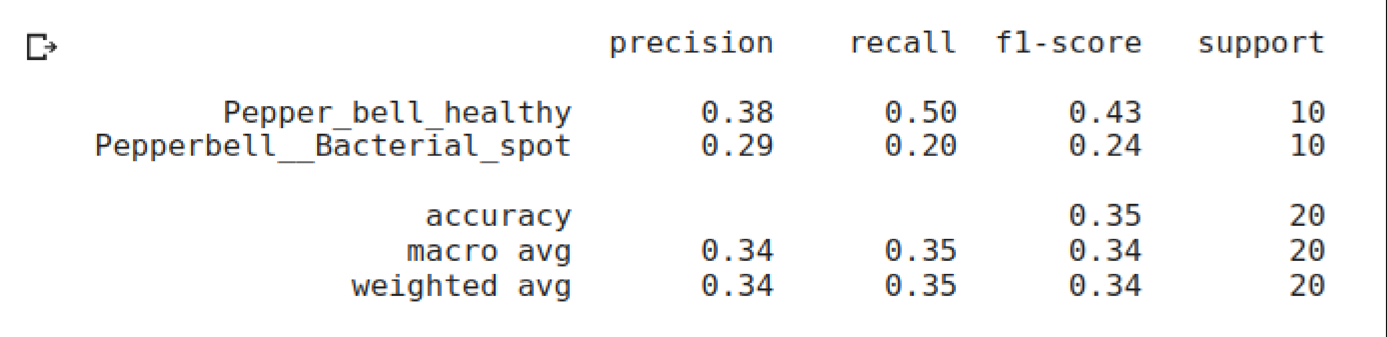
Confusion Matrix on Test Data from Kaggle Dataset	Confusion Matrix on Unseen Raw Data from web
	

Evaluation Measures(On Testing Data Kaggle):



```
                           precision    recall  f1-score   support

      Pepper_bell_healthy       0.92      1.00      0.96        90
 Pepperbell__Bacterial_spot       1.00      0.91      0.95        90

                 accuracy                           0.96       180
                macro avg       0.96      0.96      0.96       180
             weighted avg       0.96      0.96      0.96       180
```

Evaluation Measures(On Unseen/Raw Data):



```
                           precision    recall  f1-score   support

      Pepper_bell_healthy       0.38      0.50      0.43        10
 Pepperbell__Bacterial_spot       0.29      0.20      0.24        10

                 accuracy                           0.35        20
                macro avg       0.34      0.35      0.34        20
             weighted avg       0.34      0.35      0.34        20
```

Key Observations:

Following are our key observations from above experimentations:

1. Both models are performing poorly on unseen data. Accuracy drops to half and in some cases even less then half on the unseen data as compared to testing data from kaggle. We believe this is due to the poor quality and poor format of our unseen data.
2. After experimentation with various (50-50, 75-25, 60-40) data splits between positive and negative examples, We observed that the 50-50 data split is performing best.
3. After running one of our models for a high (50) number of epochs, We observed that the accuracy of the model is dropping on the testing data. Our model was overfitting.
4. When we flipped 75-25 (75% positive examples and 25% negative examples) to 25-75 (25% positive examples and 75% negative examples), We observed mirrored results.

Note: We have uploaded the unseen/raw images that we have used in our experimentations on LMS with our code files of both models.

Bibliography:

1. Alake, Richmound. “Implementing AlexNet CNN Architecture Using TensorFlow 2.0+ and Keras.” -, vol. -, no. -, 2020, -. *towards data science*, https://towardsdatascience.com/. Accessed 17th July 2021.
2. Kumar, Vaibhav. “Hands-on Guide To Implementing AlexNet With Keras For Multi-Class Image Classification.” -, vol. -, no. -, 2020, -. *Analytics India Magazine*, https://analyticsindiamag.com/hands-on-guide-to-implementing-alexnet-with-keras-for-multi-class-image-classification/. Accessed 18th July 2021.