THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

Cotton Plant Disease Prediction

Group 5

Rohan Thekanath

Nigel Martis

Pranay Bhakthula

DATS 6203 ML2: Final Project

December 6th, 2020

Motivation

- Cotton is one of the economically significant agricultural products.
- But it is exposed to different constraints in the leaf area.
- Mostly, these constraints are identified as diseases and pests that are hard to detect with bare eyes.
- This study focused to develop a model to boost the detection of cotton leaf disease and pests using the deep learning technique, CNN.

Introduction

Diseased leaf

Diseased plant

Fresh leaf

Fresh plant

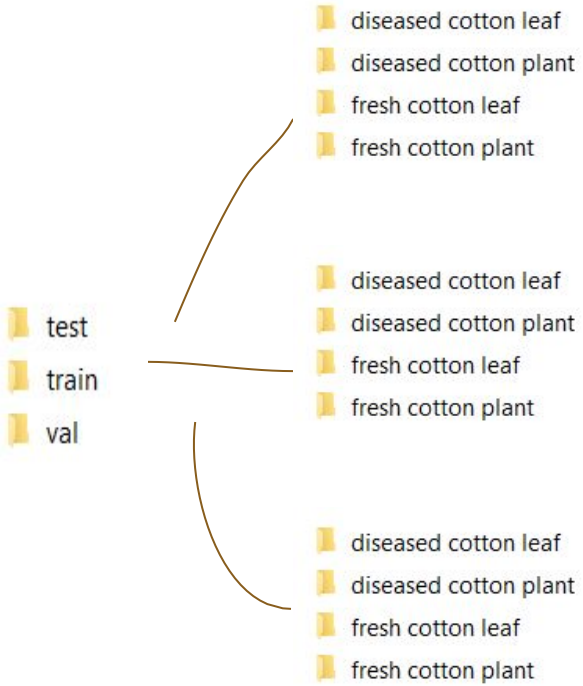
- Our project aims to identify 4 classes of labels in a cotton plant/leaf:
  1. Diseased cotton leaf
  2. Diseased cotton plant
  3. Fresh cotton leaf
  4. Fresh cotton plant

- Thus we can efficiently recognize whether a cotton plant/leaf is infected or not and it could be treated accordingly.

Dataset

- For our project we used Cotton Plant Disease dataset from Kaggle
- This dataset contains hand taken pictures of cotton plants/leaves from a farm.
- Since the dataset contains 4 classes.

Data distribution and Train-Test Split



- 160 MB of image Data
- Of the total ~2000 images used for training Custom CNN model,
  - 288 images of **diseased cotton leaves**
  - 815 images of **diseased cotton plants**
  - 427 images of **fresh cotton leaves**
  - 421 images of **fresh cotton plants**
- For the pretrained models, the image augmentations has given us nearly 12,000 images to train.

Models Trained:

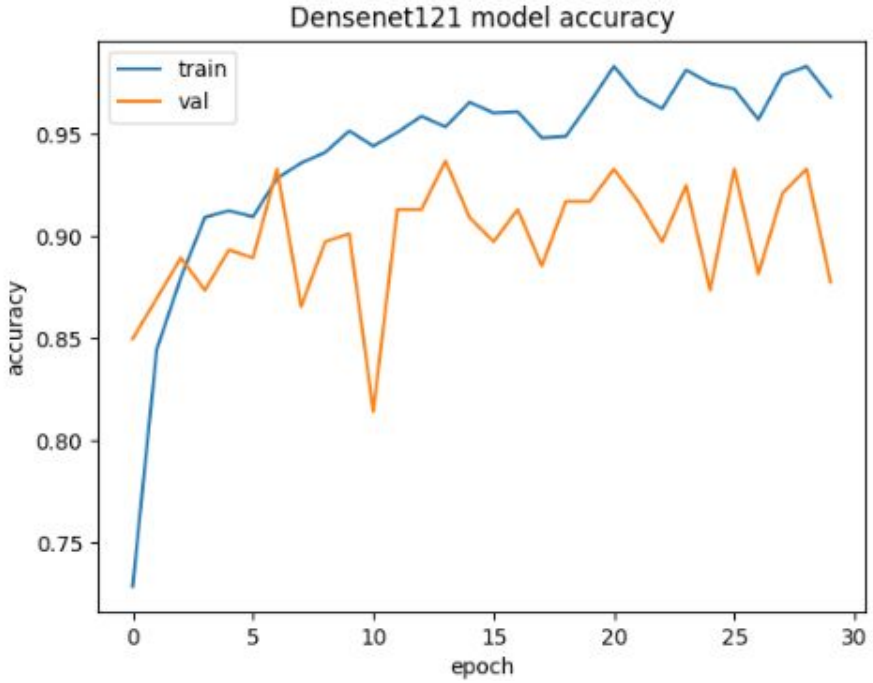
Pre-trained networks:

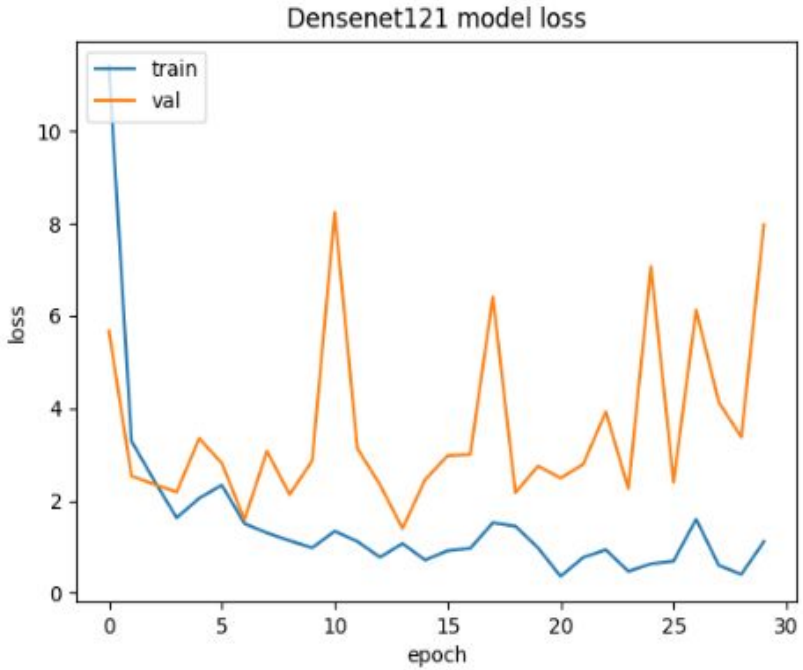
1. Densenet
2. VGG16
3. Resnet16

Manual Built network:

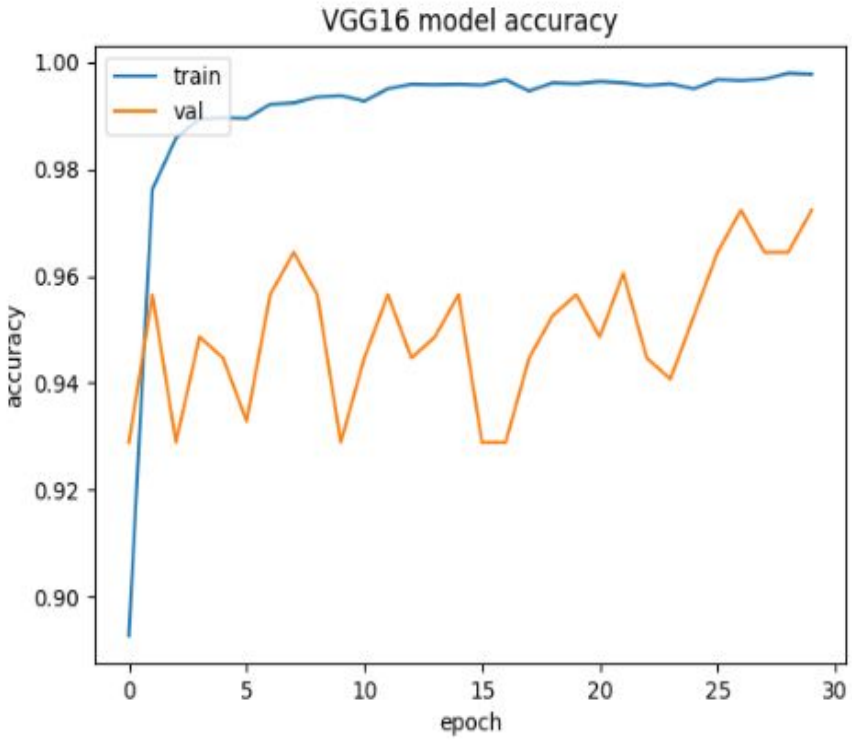
1. CNN model

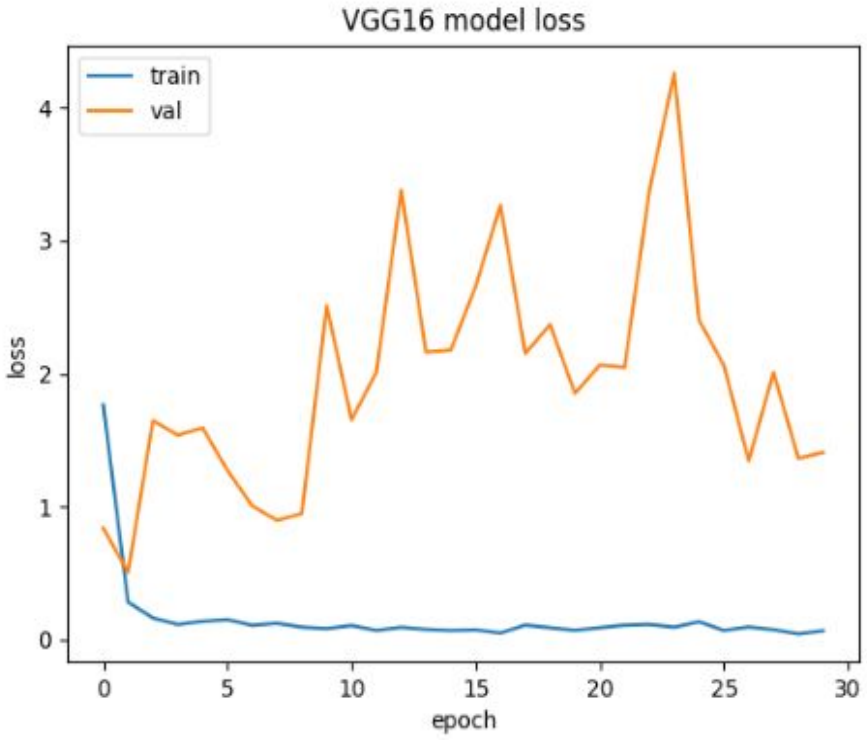
Results: Inference for Classification



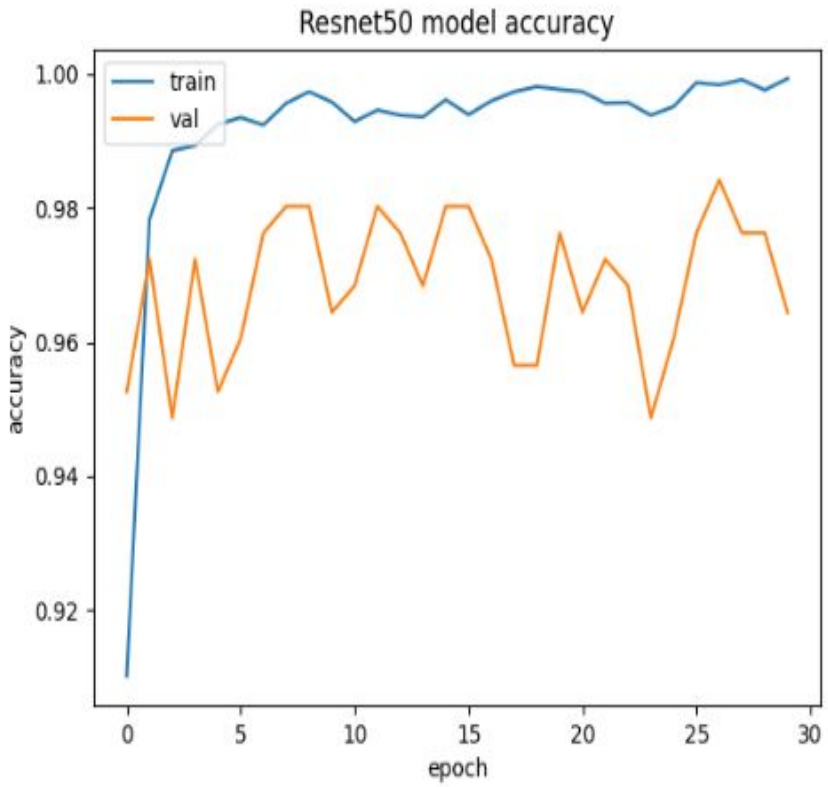


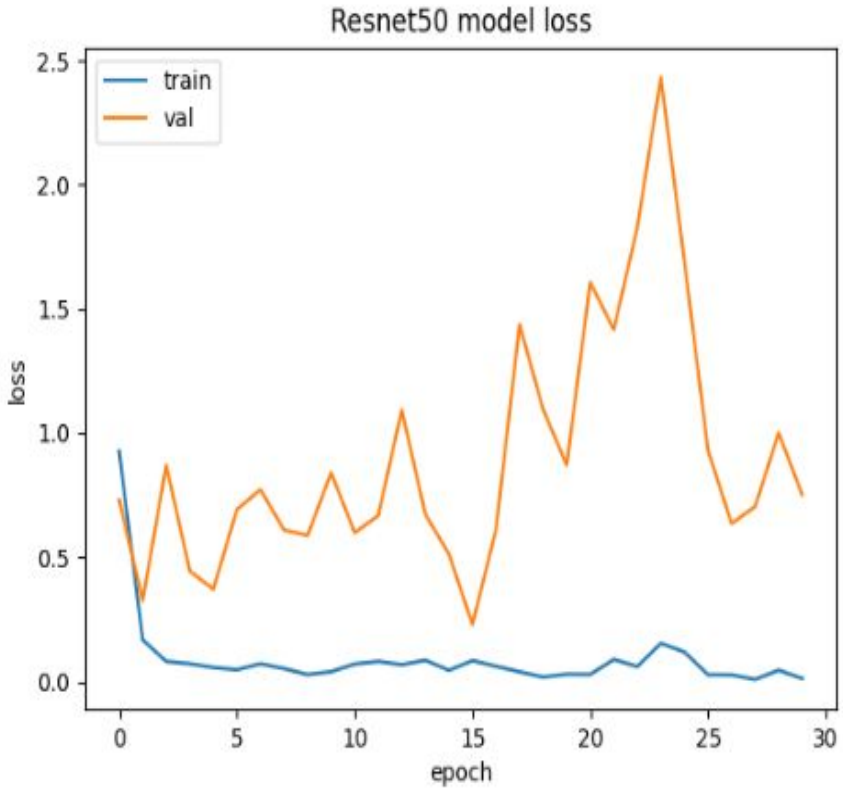
Results: Inference for Classification



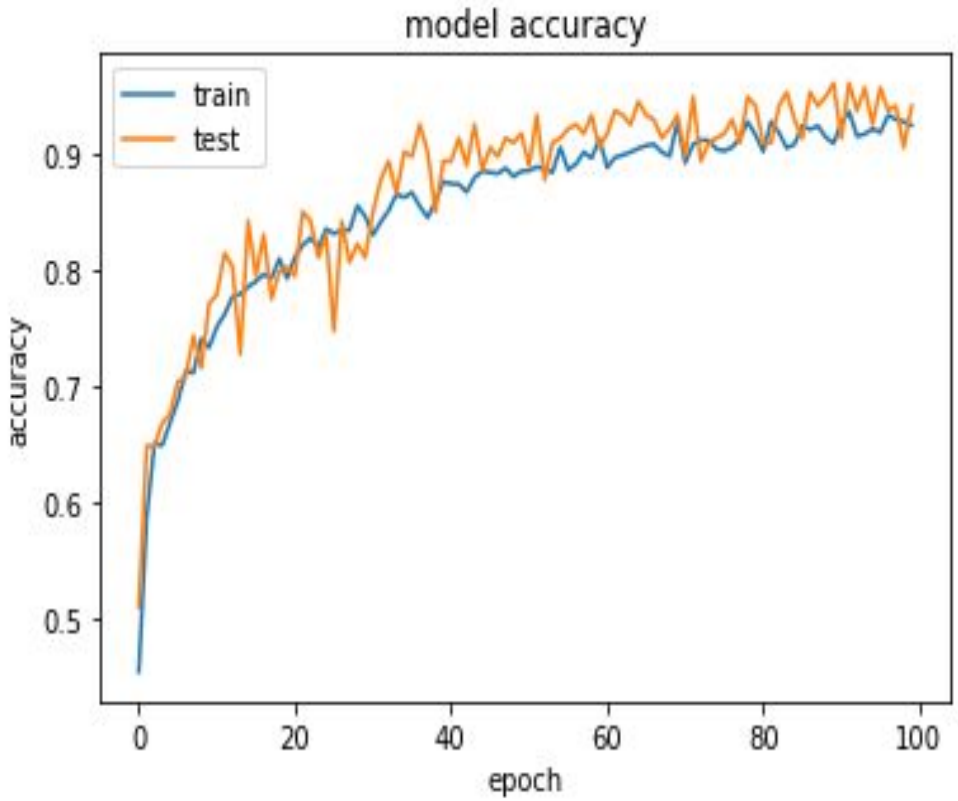


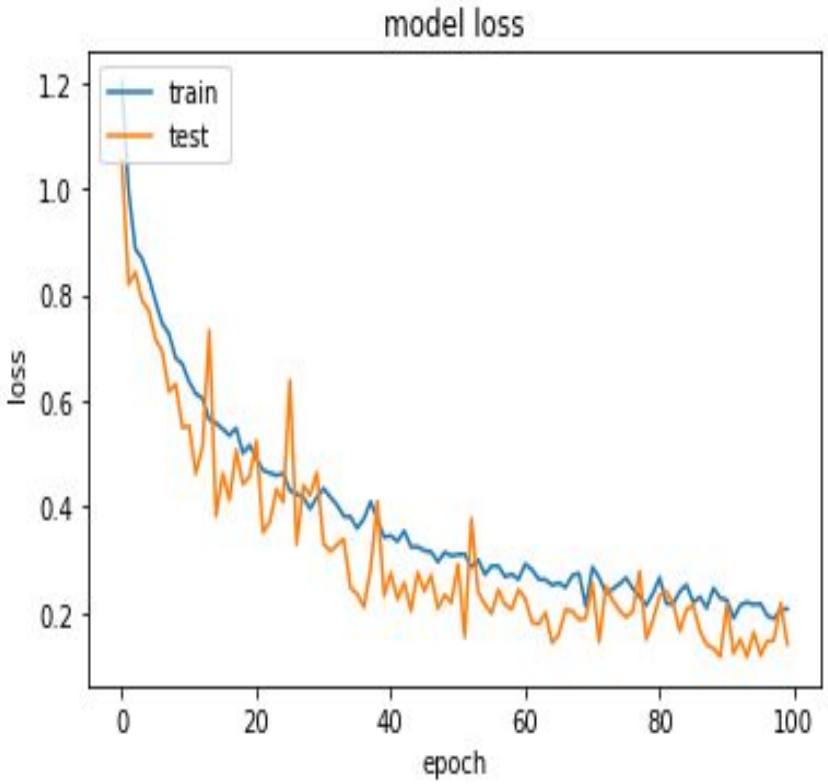
Results: Inference for Classification





Results: Inference for Classification





Deep Learning Network & Fine Tuning

- A manual CNN model worked best for the dataset and accuracy of ~ 97% was achieved. This was compared with pretrained models like Resnet50,VGG16 and Densenet.
- The data consisted of 3 folders: train,test and validation of which train and validation was used to create and test the model and validation was used as unseen data to check how well the model performs.

- **Transformations**
- Resize
- Horizontal Flip
- zoom
- Distortion
- Height and width movement

- **Hyperparameters**
- Batch Size - 32
- Learning Rate - 0.0001
- Epochs - 100
- Optimizer - Adam

- **Regularization**
- Dropout Rate - 0.25,0,5,0.1

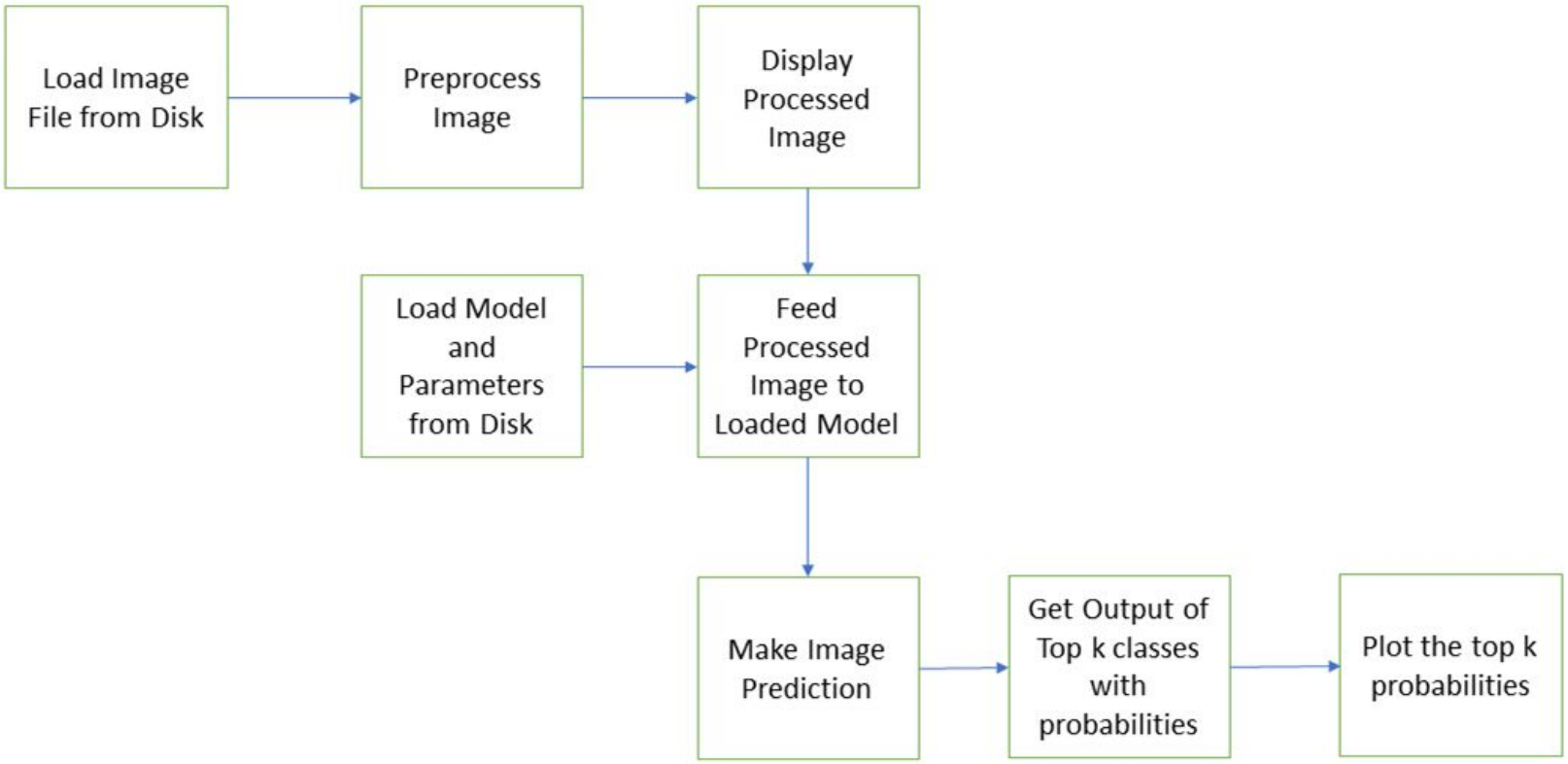
Results: Inference for Classification

Confusion matrix along with other F1 scores of each class of the best model(Manual Cnn)

```
In [29]: print("Accuracy:" + str(accuracy_score(val_actual, val_preds)))
Accuracy:0.9683794466403162

In [30]: print("F1 Score:" + str(f1_score(val_actual, val_preds, average='micro')))
F1 Score:0.9683794466403162

In [31]: print("Confusion Matrix:\n" + str(confusion_matrix(val_actual, val_preds)))
Confusion Matrix:
[[42  0  1  0]
 [ 0 75  1  2]
 [ 2  0 64  0]
 [ 1  1  0 64]]

In [32]: print("Classification Report:\n" + str(classification_report(val_actual, val_preds)))
Classification Report:
              precision    recall  f1-score   support

         0.0       0.93      0.98      0.95        43
         1.0       0.99      0.96      0.97        78
         2.0       0.97      0.97      0.97        66
         3.0       0.97      0.97      0.97        66

    accuracy                           0.97       253
   macro avg       0.96      0.97      0.97       253
weighted avg       0.97      0.97      0.97       253
```

CNN model Architecture

```
#Building cnn model
cnn_model = keras.models.Sequential([
                                    keras.layers.Conv2D(filters=32, kernel_size=3, input_shape=[300, 300, 3]),
                                    keras.layers.MaxPooling2D(pool_size=(2,2)),
                                    keras.layers.Conv2D(filters=64, kernel_size=3),
                                    keras.layers.MaxPooling2D(pool_size=(2,2)),
                                    keras.layers.Conv2D(filters=128, kernel_size=3),
                                    keras.layers.MaxPooling2D(pool_size=(2,2)),
                                    keras.layers.Conv2D(filters=256, kernel_size=3),
                                    keras.layers.MaxPooling2D(pool_size=(2,2)),
 s
                                    keras.layers.Dropout(0.5),
                                    keras.layers.Flatten(), # neural network building
                                    keras.layers.Dense(units=128, activation='relu'), # input layers
                                    keras.layers.Dropout(0.1),
                                    keras.layers.Dense(units=256, activation='relu'),
                                    keras.layers.Dropout(0.25),
                                    keras.layers.Dense(units=4, activation='softmax') # output layer
])
```


Load Image File from Disk
Preprocess Image
Display Processed Image
Load Model and Parameters from Disk
Feed Processed Image to Loaded Model
Make Image Prediction
Get Output of Top k classes with probabilities
Plot the top k probabilities

Model Deployment

Cotton Plant Disease Prediction

Image Classifier

Choose...

Cotton Plant Disease Prediction

Image Classifier

Choose...

Result: The leaf is diseased cotton leaf

Summary & Conclusion

- A cotton disease detector model was developed to detect if a plant/leaf falls under categories like diseased leaf, diseased plant, fresh leaf or fresh cotton plant.

- Pre-trained models like Resnet50,VGG16, Densenet was compared with a manually trained CNN model and surprisingly the manually trained model performed the best.

- Accuracy achieved was about ~97% and the model also performs very well on unseen plant/leaf images.

- This model has also been deployed where the user can choose an image and see its result.

Future Work

Future work would consist of adding model interpretability into the project where one can visually see how the model classifies the images.

A mobile app could also be made where the farmer can take pictures of the plant/leaf to identify if is diseased on not. The app would also display the solution including how and which fertilizers to use.

THANK YOU

github link : https://github.com/Pranay-yanarp/DATS-6203-Final-Project-Group5