Grape Disease Detection and Classification

Revanth Krishna Maddula | Laxmi Prasanna Palle | Bhavana Obellaneni | Jagan Gadamsetty

CMPE - 257 Machine Learning

Project Group 4

Computer Software Engineering Department

San Jose State University

San Jose, California, USA.

Abstract – Grapes are the essential crop in today's world. In order to maintain a good crop yield of grapes, farmers need to protect them from many diseases and some of the common diseases are black measles, black rot and leaf blight. As grapes are vulnerable to these diseases, they should be taken care in early stages. Our project is based on developing machine learning models to detect and classify grape diseases, so that farmers can easily identify the type of disease and take appropriate measures. For this grape disease and classification, we designed four different machine learning models which are powerful enough to accurately predict the type of disease. The four models are, CNN, VGG16 CNN(Advanced CNN), Random Forest Model, Support Vector Machine. For the purpose of training the models and data preprocessing, a dataset is collected from different sources and fed to the models. We have also developed a web application, where the user can give an image of a grape leaf to detect if it is affected by disease and additionally classifies the disease with the accuracy score.

Keywords – CNN, VGG16 CNN, Random Forest Model, Data Preprocessing, Support Vector Machine

INTRODUCTION

Fruits are the most important produce in any country, without which humans can't survive. Additionally, the produce should be fresh with enough nutrients to give out the most benefits to the human when consumed. Therefore, in order to get the most benefit from the produce its quality is also taken into consideration. Out of all the available economical crops, grapes are the essential fruit crops in the world, they are widely consumed and used in producing wine. But sadly, they are more prone to diseases as they are very sensitive to the variation in weather conditions and if the crop is attacked with the disease, extra care should be taken with precautionary measures so that the according diagnosis is performed, before the whole crop gets destroyed with the spread of the disease. The faster the disease is predicted, the less time and effort it takes to eradicate the disease and it is more affordable to observe the early signs of disease. There will be heavy economic loss due to the grape diseases, the survey shows that Georgia, 2015 has huge economic loss due to the grape disease and incurred heavy costs in controlling the disease.

To maintain a good yield of the crop and incur maximum benefits, it is important to distinguish the quality of the produce, if it is healthy or not. However, manually identifying the disease and

classifying it will take a lot of time and effort and the results also can vary with many factors taken into consideration. Fortunately, use of machine learning in image classification has been very advantageous to farmers which resulted in a good yield of a healthy grape crop, with more profit for the yield. Our Project aims to identify and classify the Grape leaf diseases using powerful machine learning models like CNN, VGG16 CNN, Random Forest Classifier, Support Vector Machine.

The Project report on image detection and classification of grape diseases presents the overview of the need for machine learning models to detect grape diseases. Then further sections elaborate on the description of the selected machine learning models with the parameters used and accuracy obtained in our project. Along with the preparation of training and testing data set used in the model including the case study and statistics

I. PROJECT TEAM WITH ROLES

We, a group of four members, contributed in developing each machine learning model and collected 9200 images from various data sources.

Revanth Krishna Maddula : Revanth designed the CNN model and developed the backend logic for the web application. He collected the data sets for black measles and labelled them using label me.

Jagan Gadamsetty : Jagan developed the VGG 16 CNN model and collected images for black rot. He also contributed to the web application by providing the architecture diagram.

Palle Laxmi Prasanna : Prasanna worked on Random Forest Model and collected datasets for black rot followed by data preprocessing. She developed the front end UI for the web application.

Bhavana Obellaneni : Bhavana developed the support vector machine for our project and collected healthy images for training our models. She integrated both logic and the frontend for our web application.

II. TRAINING AND TEST DATA PREPARATION

A. Training and Test data preparation

A very basic strategy to make a machine learning model to give accurate predictions is to train the model using a proper dataset so that it predicts with better accuracy. Hence, to provide a healthy dataset , we worked a lot on producing a good number of quality images and labelled them using the labelme tool. The images are collected from various sources like, kaggle, and from different internet sources. To increase our dataset, we used a data augmentation tool called imgaug and finally we were able to collect 15422 images.

Accuracy of the ML model is directly proportional to quality and quantity of the dataset size. Keeping this approach in mind, the healthy leaves are also collected as a negative dataset. Hence, our model is capable enough to even predict whether the given grape leaf image has a disease or not.

Despite minor differences among all the 3 diseases, the model should be able to correctly classify the disease, when an image is given. This dataset is used to train the models and is categorized into three parts, training dataset, testing dataset, validation dataset respectively.

Each team member worked on the dataset and labelled the images using label me tool and the disease on the leaf is highlighted along with the edges of the leaf in label me tool which converts the image to json file and the resulted files are used to fit the model perfectly and make the model provide the highest probability of accuracy for the given classification. We made the standard split of the dataset to 80%-20%, training and testing dataset respectively. Moreover, we also used 20% of the dataset as a validation dataset.

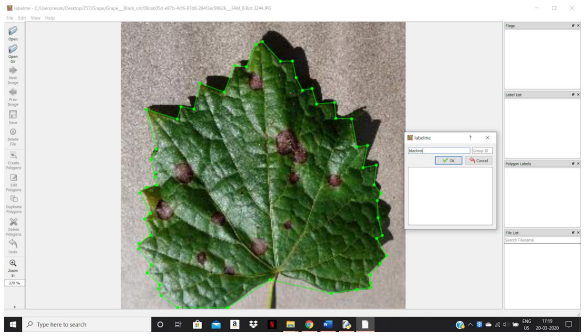


Figure 2.1: Raw image before labelling in Label Me

Figure:2.2:Black rot training data image labelled using Labelme

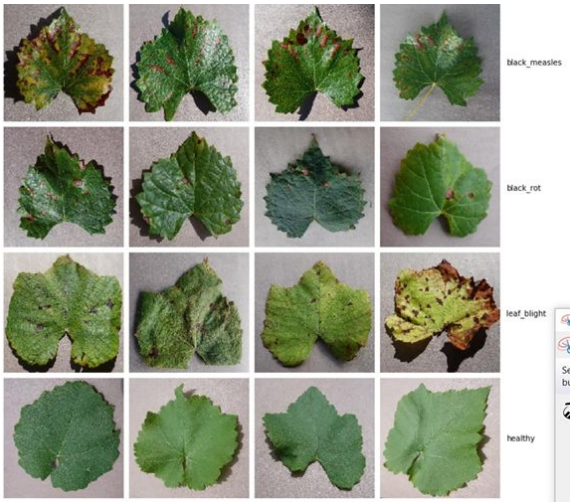


Figure:2.3: training and test data images labelled using Labelme

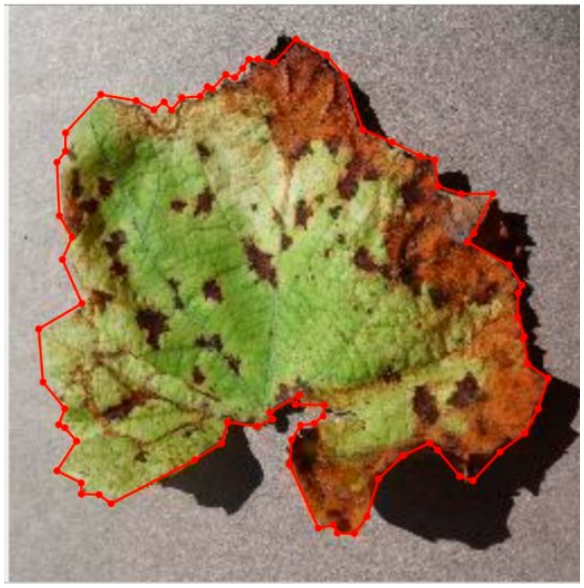


Figure:2.4:leaf blight labelled using Labelme

Steps followed for the preparation of image data:

- Collect data from the data source.
- Check the correctness of the data.
- Pre-process and cleanse the data.
- Remove the least correlated values.
- Label the data.

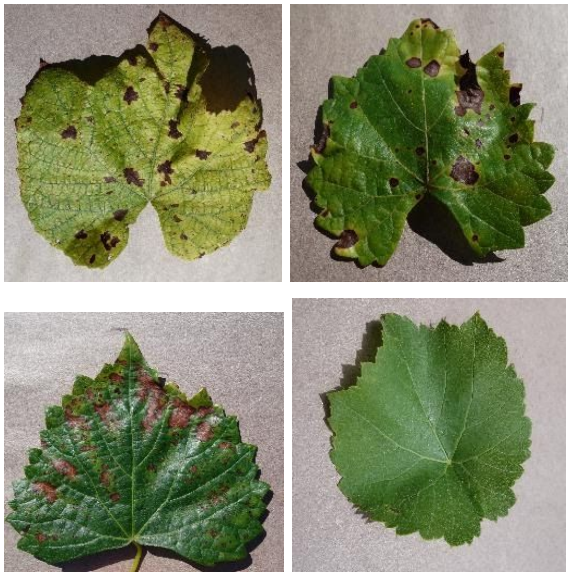


Figure:2.5: Samples of test dataset which are given to the model to predict the classification of the disease.

B. *Data Augmentation*

Data Augmentation is used for our dataset to improve the accuracy of our model and to reduce the disadvantages of over fitting, different approaches are used by numpy library , to produce images with different transformations

Figure: 2.6. - Original Image transformed to vertical flip

Figure: 2.7 - Gamma Correction of the Original Image

Figure: 2.8. -Sharpening of the Original Image

Disease Classification	Total Number of Samples(before Augmentation)	Total Samples (Including Augmented Images)
Black Measles	1433	5621
Black Rot	1530	2910
Leaf Blight	577	4670
Healthy	1342	2221
Total Images	4890	15422

Table: 2.1. Table Representing the total data set for training and testing of the model

C. *Data Validation:*

Data validation is necessary to make sure that there are no anomalies in our dataset and to do such a process, we need to compare the statistics of our dataset with our actual schema.

```
other_stats = tfdv.generate_statistics_from_tfrecord(data_location=other_path)
anomalies = tfdv.validate_statistics(statistics=other_stats, schema=schema)
```

D. Data Preprocessing:

I. Grape Leaf Images:

Data Labelling: We used the “Label Me” tool to label each image.

Dimensionality Reduction: Images are resized into one common resolution of 221x221 and feature detection techniques were used for better performance.

II. Hold Out:

Our data set consists of 15422 images, from which 80% of the dataset is used for the training data and is labelled with one of the 3 diseases among Leaf Blight, Black Measles, and Black Rot including the healthy leaves. 20% of the remaining, is used as testing dataset, and similar to the training dataset modules, testing dataset is also classified into 4 modules, Black Measles, Leaf Blight, Black rot, Healthy respectively. 20% of the training dataset is taken as a validation data set, to make the model more efficient in making predictions. To make the data split in this fashion, we used Scikit_learn’s train_test_split method.

III. Fine-tuning:

The crucial step in any data preprocessing is fine tuning, we did the fine-tuning by performing the principal component analysis.

E. Model Training:

I .CNN:

```
In [29]: # Train vanilla CNN model
         model.train(data)
```

```
Epoch 24/30
81/81 [==============================] - 192s 2s/step - loss: 0.2782 - acc: 0.8989 - val_loss: 0.2857 - val_acc: 0.8906
Epoch 25/30
81/81 [==============================] - 217s 3s/step - loss: 0.2615 - acc: 0.9082 - val_loss: 0.2684 - val_acc: 0.8938
Epoch 26/30
81/81 [==============================] - 187s 2s/step - loss: 0.2572 - acc: 0.9098 - val_loss: 0.2551 - val_acc: 0.9047
Epoch 27/30
81/81 [==============================] - 189s 2s/step - loss: 0.2484 - acc: 0.9164 - val_loss: 0.2482 - val_acc: 0.9094
Epoch 28/30
81/81 [==============================] - 186s 2s/step - loss: 0.2482 - acc: 0.9129 - val_loss: 0.2353 - val_acc: 0.9172
Epoch 29/30
81/81 [==============================] - 192s 2s/step - loss: 0.2390 - acc: 0.9117 - val_loss: 0.2277 - val_acc: 0.9234
Epoch 30/30
81/81 [==============================] - 197s 2s/step - loss: 0.2261 - acc: 0.9218 - val_loss: 0.2416 - val_acc: 0.9109
WARNING:tensorflow:From C:\Users\revan\AppData\Roaming\Python\Python37\site-packages\tensorflow_core\python\ops\resource_variable_ops.py:1786: calling BaseResourceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops) with constraint is deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
```

Figure: 2.9. - CNN Model training

II. VGG 16 CNN:

```
In [11]: model.train(data)

         # Save model
         name = 'vgg16_model_epochs30'
         model.save('vgg16.h5')
```

```
WARNING:tensorflow:From C:\Users\revan\FINAL PROJECT\models\vgg16.py:56: Model.fit_generator (from tensorflow.python.keras.engine.training) is deprecated and will be removed in a future version.
Instructions for updating:
Please use Model.fit, which supports generators.
WARNING:tensorflow:sample_weight modes were coerced from
  ...
    to
  ['...']
WARNING:tensorflow:sample_weight modes were coerced from
  ...
    to
  ['...']
Train for 81 steps, validate for 20 steps
Epoch 1/30
81/81 [==============================] - 1118s 14s/step - loss: 0.8322 - acc: 0.6604 - val_loss: 0.4046 - val_acc: 0.8828
Epoch 2/30
81/81 [==============================] - 1148s 14s/step - loss: 0.3657 - acc: 0.8775 - val_loss: 0.2740 - val_acc: 0.9250
Epoch 3/30
81/81 [==============================] - 1061s 13s/step - loss: 0.2651 - acc: 0.9152 - val_loss: 0.2234 - val_acc: 0.9422
Epoch 4/30
```

Figure: 2.10 - VGG 16 CNN model training

III.Random Forest Classifier:

```
from sklearn import svm
from sklearn.ensemble import RandomForestClassifier

clf=RandomForestClassifier(max_depth=9, random_state=0,n_estimators=100)
clf.fit(fruit_images, labels)
```

```
C:\Users\revan\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The default value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
```

```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=9, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=None,
            oob_score=False, random_state=0, verbose=0, warm_start=False)
```

Figure: 2.11 – Random Forest Classifier model training

IV. Support Vector Machine:

```
In [4]: from sklearn import svm

        clf = svm.SVC(decision_function_shape='ovo')
        #clf = svm.SVC()

        clf.fit(fruit_images, labels)

C:\Users\revan\Anaconda3\lib\site-packages\sklearn\svm\base.py:196: FutureWarning: The default value of gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to avoid this warning.
  "avoid this warning.", FutureWarning)

Out[4]: SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
          decision_function_shape='ovo', degree=3, gamma='auto_deprecated',
          kernel='rbf', max_iter=-1, probability=False, random_state=None,
          shrinking=True, tol=0.001, verbose=False)
```

Figure: 2.12 - Support Vector Machine model training

F. Web Application Model

Finally, after all the models are trained, we have selected the model which gave the highest accuracy, which is the VGG16 CNN model. This model takes its input in the form of an image and predicts the image into a particular class, along with the predicted probability of how much sure the model is.

G. Project Workflow

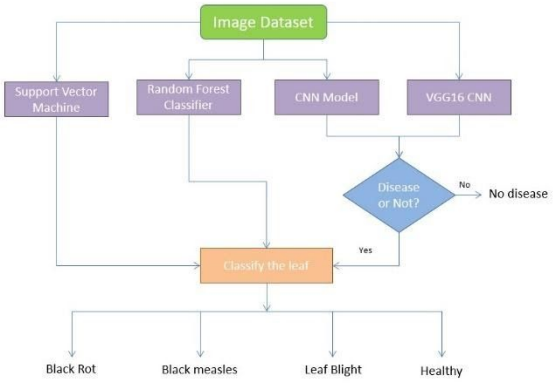


Figure:2.13 - Project Design Diagram

III. OUR MODELS

We used four machine learning models namely, CNN, VGG 16 CNN, random forest classifier, Support Vector Machine to detect the grape disease and classify them. The represented models are explained in detail and their justifications and comparisons are mentioned:

A. Convolutional Neural Network (CNN)

One of the best neural networks models, CNN is used in image classification of grape leaf diseases.

The best part of CNN model is, when an image is given, it learns all the distinct features from each class of images and gets trained accordingly without any human involvement. CNN model has a series of convolution layers and pooling layers operated on the given image, which is in the form of matrix input and then followed by fully connected layers to output a SoftMax. CNN models can be generally categorized into two components: Feature Extraction and classification. The convolution process comes under the feature extraction and finally the fully connected layers classifies the image

The input matrix is supplied to a convolution function, whereas other matrix called convolution filter is multiplied with the given input matrix and feature map is derived, in our model, we applied 4 convolution operations, resulting in 4 feature maps which are also followed by 4 pooling layers, the stride and padding features are applied to match the feature map dimensions with the input.

Pooling is performed to reduce the dimensions of the resulting feature map after each convolution, but the depth is not reduced, generally a stride of 2 is with max pool operation. Then the result is vectorized and passed through the fully connected layers, which classifies the image with the values of 0 and 1. As the image passes through each convolution and pooling operation it fits the data and learns each distinct feature of the image in detail.

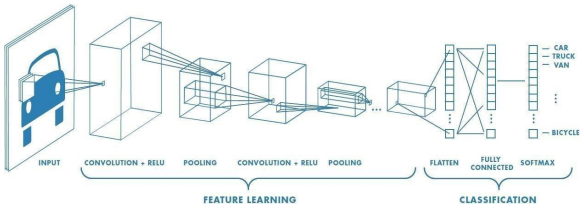


Figure 3.1: CNN model with multiple convolution layers

The summarized steps involved in a CNN model are as below:

- An input image is provided
- Multiple feature Maps are derived using convolution and pooling operations of four layers.
- Stride parameters are applied to resize the feature maps along with the filter size.
- Relu, activation function is applied to the matrix
- This output is fed as a 1D vector to a fully connected layer.
- The final output for the image classification is derived after applying the softmax function.

Layers applied in the CNN model :

Layer (type)	Output Shape	Param #
input_2 (InputLayer)	[(None, 224, 224, 3)]	0
conv2d_4 (Conv2D)	(None, 222, 222, 32)	896
max_pooling2d_4 (MaxPooling2	(None, 111, 111, 32)	0
conv2d_5 (Conv2D)	(None, 109, 109, 32)	9248
max_pooling2d_5 (MaxPooling2	(None, 54, 54, 32)	0
conv2d_6 (Conv2D)	(None, 52, 52, 64)	18496
max_pooling2d_6 (MaxPooling2	(None, 26, 26, 64)	0
conv2d_7 (Conv2D)	(None, 24, 24, 64)	36928
max_pooling2d_7 (MaxPooling2	(None, 12, 12, 64)	0
flatten_1 (Flatten)	(None, 9216)	0
dense_2 (Dense)	(None, 128)	1179776
dropout_1 (Dropout)	(None, 128)	0
dense_3 (Dense)	(None, 4)	516

Figure 3.2: convolution _ pooling applied in CNN Model

Experiment Results:

Classification Report

	precision	recall	f1-score	support
black_measles	0.96	0.83	0.89	276
black_rot	0.82	0.93	0.87	236
leaf_blight	0.92	0.98	0.95	84
healthy	0.97	0.98	0.98	215
micro avg	0.91	0.91	0.91	811
macro avg	0.92	0.93	0.92	811
weighted avg	0.92	0.91	0.91	811

Figure 3.3: Classification Report for CNN model

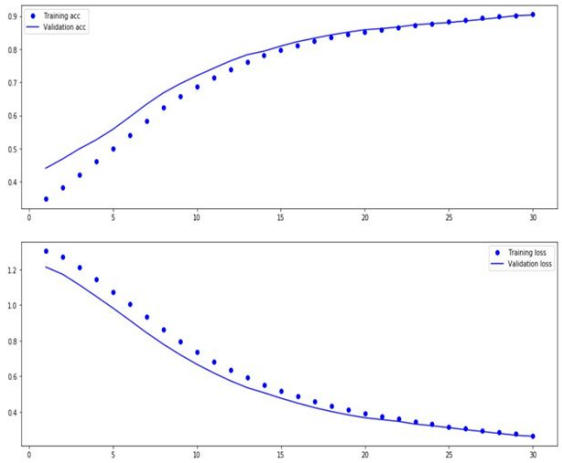


Figure.3.4: Training and Validation datasets

Performance Metrics:

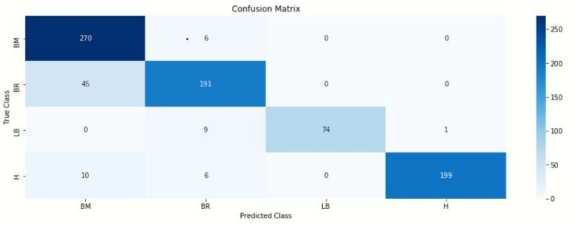


Figure:3.5: Confusion Matrix for CNN Model

Advantages of CNN:

- The prediction time for the CNN model is less compared to other models, it also has an accurate prediction compared to other models
- The requirement of preprocessing is much less compared to other models
- It automatically learns the features of a given image without human supervision
- Interpreting or visualizing various components is difficult in most of the deep learning models, but CNN is quite opposite
- The main advantage of using neural networks is it is very powerful enough and classifies the image accurately.

Disadvantages of CNN:

- Humongous amount of data is needed to train a CNN model, to get an accurate prediction.
- A fast processor is necessary to train CNN with complex data.
- High computation cost is needed to invest while training the CNN models.

Justification for using this model:

CNN model, despite the disadvantages which are minor, gives an accurate prediction for the image classification. Hence, this model is opted for image classification on grape diseases.

B. VGG 16 CNN model

The VGG model is very simple yet very accurate in image classification and prediction.

It consists of 16 layers including the convolution layers with pooling and then passed on to the fully connected neural networks, only 3x3 convolution layers are used in the entire network. Instead of one convolution layer, 2 convolution layers are used so that relu operation is performed two times and thereby increasing the non-linearity for the given model which helps in accurate prediction by making the model more powerful.

The VGG16 CNN model consists of 3 components, they are:

The Feature Maps: They are also called with the name of intermediate activations as the final output is passed through an activation layer

Each convolution layer formed is named as blockX_convY, the output of each block learns a distinct feature from the image and activates the feature in the image, it means when a new image having that feature is given to the model, it can easily classify the feature as it encoded that feature using the convolution layers.

CovNet Filters: In the beginning, the starting layers has the fundamental features, clearly distinguished features are detected hence these layers have maximum information regarding the given image input, they distinguish simple features which are easily observable like colors and edges

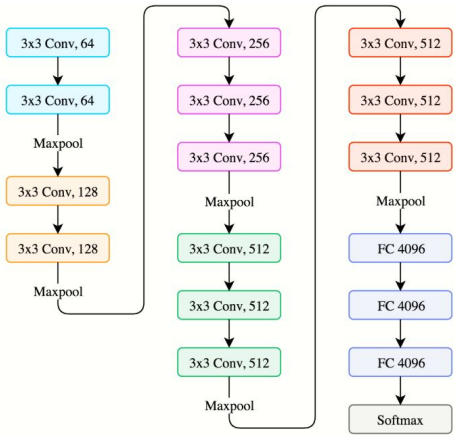


Figure:3.6: VGG 16 CNN model with multiple convolution layers

As the layers get deeper inside the network, each feature is extracted in detail, like the object specific parts, hence, the images which are taken in the deeper layers are not easily identifiable like the given image, but the feature extraction is actively done even inside the deeper layers.

To further boost the accuracy, we use the dropout technique which is the famous one in deep neural networks, some of the neurons are temporarily dropped from the training, and again they can become active in the next step, dropping out makes the neurons to act independent and avoid depending on a small number of neurons. This also reduces the concept of overfitting.

Visualization: Given a classification, the CNN model finds the image regarding the classification.

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 224, 224, 3)]	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0

Figure 3.7: VGG 16 CNN model with multiple convolution layers in our model

Experimental Results:

Classification Report

	precision	recall	f1-score	support
black_measles	0.94	0.97	0.95	276
black_rot	0.96	0.93	0.94	236
leaf_blight	0.95	1.00	0.98	84
healthy	1.00	0.98	0.99	215
accuracy			0.96	811
macro avg	0.96	0.97	0.97	811
weighted avg	0.96	0.96	0.96	811

Figure 3.8: Classification Report obtained for VGG 16 CNN Model

Performance Metrics:

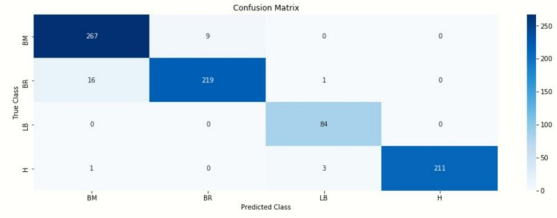


Figure 3.9: Confusion Matrix for VGG 16 CNN Model

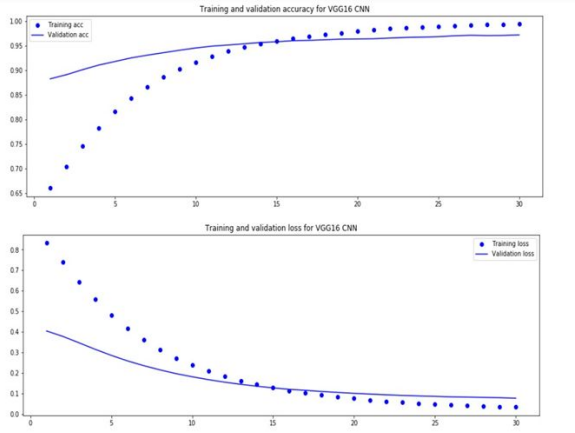


Figure 3.10: VGG16 CNN performance on training and validation dataset

Advantages of VGG 16 CNN:

- The 2 convolutions are stacked together before sending to pooling, hence increasing the non linearity by adding two relu functions, thereby increasing accuracy in prediction
- It is a simple model and can be easily interpreted

Selection Justification:

The VGG16 CNN is also one of the neural networks models, but using this model, we can detect the image even faster and with more accuracy.

C. Random Forest Classifier

Our team used random forest classifier to predict and classify the disease on grape leaves.

Random forest is a collection of multiple uncorrelated decision trees, where the prediction of the model comes out to be the mean of all the decision trees in the case of regression or the maximum votes for the case of classification.

By implementing the bagging and feature randomness, the accuracy of the models prediction is improved and making these decision trees more uncorrelated will make the model to increase its accuracy in prediction.

While all the decision trees may not be correct in their classification but the overall classification which is determined by majority of trees is most probably the correct prediction.

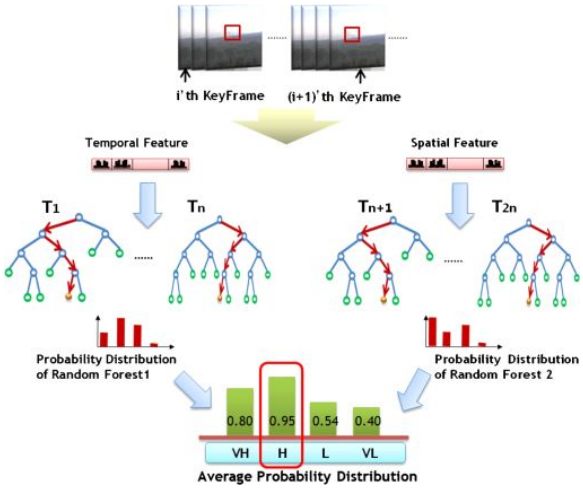


Figure: 3.11 – Representation of a Random Forest Model

Experimental Results:

Confusion matrix
Classification report

	precision	recall	f1-score	support
black_measles	0.82	0.84	0.83	269
black_rot	0.77	0.74	0.76	243
healthy	0.89	0.97	0.93	77
leaf_blight	0.87	0.84	0.85	222
accuracy			0.82	811
macro avg	0.84	0.85	0.84	811
weighted avg	0.82	0.82	0.82	811

Figure: 3.12 – Classification Report for Random Forest Classifier

Performance Metrics:

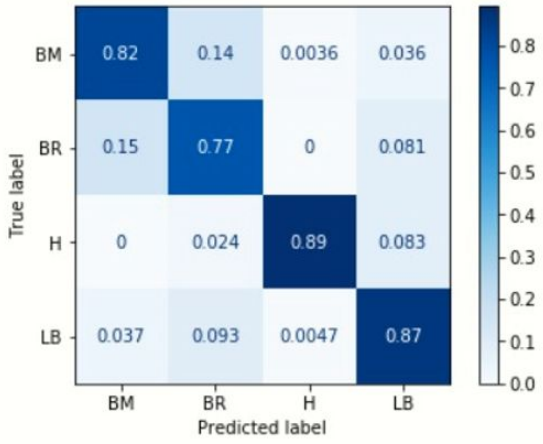


Figure: 3.13 – Confusion Matrix for Random Forest Classifier

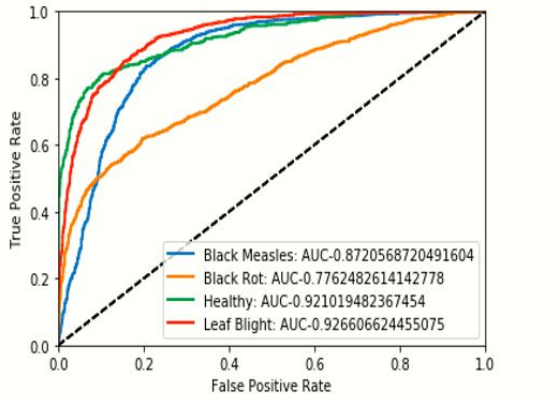


Figure: 3.14 – ROC - AUC Curve for Random Forest Classifier

Advantages of using Random Forest:

- The model gets trained within less time compared to other models.
- There need not be much pre processing of data, the model can deal with missing values and outlier data.
- Unlike decision trees, which check for best split on deciding with perfect attributes, decision trees in random forest take random features and classify the data, which in turn makes the model predict more accurately.

Justification for using the Model:

It is the best one to demonstrate the feature importance, the feature selection is implicit and happens on its own. In Spite of the missing data, it can provide accurate results

D. Support Vector Machine

Support vector machines are used as one of our models to detect and classify grape disease.
SVM is a classification algorithm, it finds hyperplane when N number of features are given, which classify all the data points.

The dimension of the hyperplane varies according to the number of input features given to distinguish the data points, if the number input features are 2, then the decision boundary is just a line, if 3, then it becomes a 2D plane, when it's more than 3, it is difficult to imagine the hyperplane.We have to use kernel functions for the non linear data to view in high dimensional space.here we have four categories to classify the leaves.so we use radial kernel function to represent data in high dimensional space.

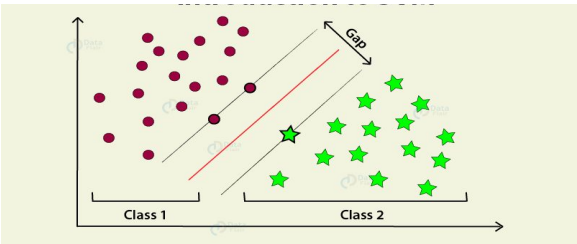


Figure 3.15: Demonstration of SVM with two features and hyperplane is a line

To maximize the margin, the loss function called hinge loss is used.

$$c(x, y, f(x)) = \begin{cases} 0, & \text{if } y * f(x) \geq 1 \\ 1 - y * f(x), & \text{else} \end{cases}$$

Figure 3.16: loss function to maximize the margin

After finding the loss function, the gradients obtained from partial derivatives are used to update the weights of the data points, if there is an incorrect classification, then the gradient needs to be updated with the loss function, if there classification is correct then no need to update the gradient with the loss function.

Experimental Results:

Classification report

	precision	recall	f1-score	support
black_measles	0.79	0.83	0.81	260
black_rot	0.74	0.67	0.70	261
healthy	0.70	0.98	0.82	60
leaf_blight	0.84	0.78	0.81	230
accuracy			0.78	811
macro avg	0.77	0.82	0.78	811
weighted avg	0.78	0.78	0.77	811

Figure.3.17: Classification Report for Support Vector Machine

Performance Metrics:

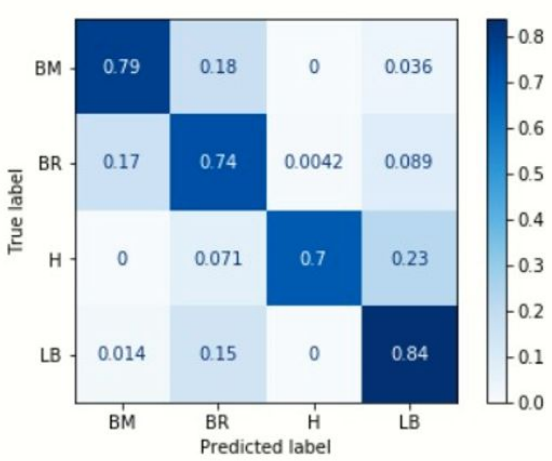


Figure: 3.18: Confusion Matrix for Support Vector Machine

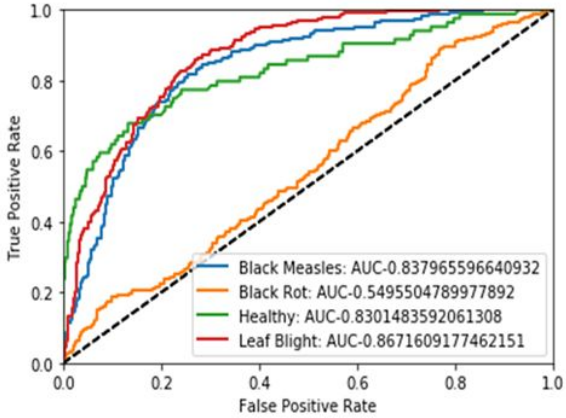


Figure 3.19: ROC-AUC Curve for Support Vector Machine

Advantages of using Support Vector Machine:

- Feature mapping has much computational load, but in the case of SVM ,Kernel trick can be used for obtaining the feature mapping using a simple dot product.
- The model finds out the best hyperplane which categorizes the data points accurately and minimizes the risk of data error

Disadvantages of Support Vector Machine:

- If the data is missing or if it contains outliers, it can drastically affect the performance.

- Choosing the right kernel is probably the difficult one as there are multiple numbers of kernels, it is hard to find which kernel is the correct one for the given data.

Selection Justification:

- For the image classification, this model can be used even if the data is labelled or unlabeled, given the complex neural networks, the support vector machine relatively consumes less memory.

E. Model Justification & Statistics

Model	Justification
1] CNN	The main reason for using CNN is that performing feature engineering is not required in this model. The local understanding of an image in CNN is good. It has very high accuracy when it comes to image classification.
2] VGG 16 CNN	This model divides images into smaller parts to classify it, hence, the accuracy achieved is much higher than regular CNN model which makes it more reliable in detecting diseases. It can provide good accuracy even when there is limited data.
3] Random Forest Classifier	Random Forest reduces the variance part of error rather than the bias part. Better accuracy on unexpected data set validation and it is great at avoiding overfitting. Also, it is a good indicator of importance assigned to the features.
4] Support Vector Machine	Capturing complex relationships between the data points does not require any transformations on your own by using a support vector machine using a non-linear kernel.

IV. PERFORMANCE AND RESULTS

A. Statistics based on current accuracy of the models:

Model	Input Data Type	Accuracy
CNN	Grape Images(Healthy + Disease)	Val_accuracy = 91.09%
VGG 16 CNN	Grape Images(Healthy + Disease)	Val_accuracy = 93.12%
Random Forest Classifier	Grape Images(Healthy + Disease)	Val_accuracy = 82%
Support Vector Machine	Grape Images(Healthy + Disease)	Val_accuracy = 78%

B. Complexity and Limitations of the models:

Algorithm	Decision Boundary	Model Complexity Reduction	Limitations
CNN	Non-Linear	Reduce number of hidden layers, regularization, early stopping	High computational cost
VGG 16 CNN	Non-Linear	Reduce the network complexity.	The network size of this model is over 550mb
Random Forest Classifier	Wiggly contours with complex shapes.	Low bias and low variance	Accuracy is less as compared to CNN
Support Vector Machine	Non-Linear	Reduce the dataset for large datasets.	Several parameters should be taken care of for best results.

C. Demonstration of final result

For demo purposes, we have created a web application which will take an input image from the user with the upload button and after uploading a file, then the user can click on the user friendly interface to predict the image and along with its predicted accuracy.

Screenshot of the web application:

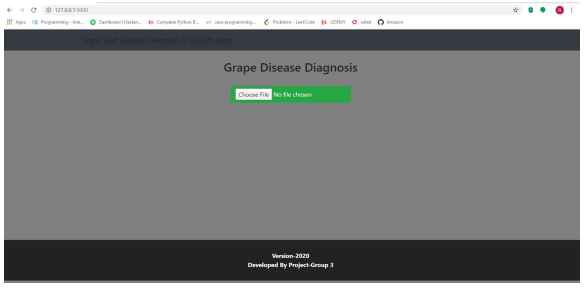
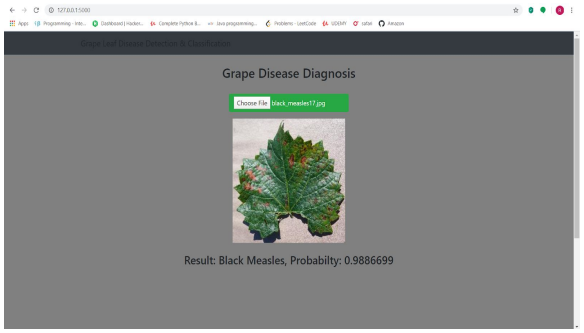
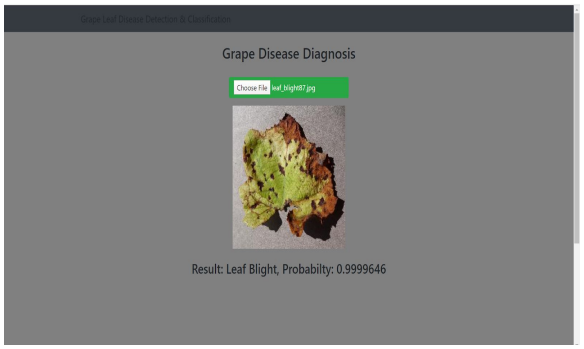


Image Upload and Predicted Result:





V. CHALLENGES AND LESSONS LEARNED

A. Challenges:

1. Gathering images from multiple sources has happened to be the major challenge during the project.

I. Collection of good number of leaves which are affected with the diseases, which we want to train our model are not many in number. Hence, we got to search those images from multiple sources.

2. Improving the accuracy of the grape detection and classification models

To make the models perform better and give accurate predictions, just fitting the model with raw data is not a valid approach. Hence, proper data preprocessing, labelling, finding the missing values in the dataset is to be done and this has taken a lot of time and effort.

In order to reduce the disadvantage of overfitting, we tried using a validation dataset and made sure that the hyper parameters are tuned properly within the defined constraints.

B. Lessons learned:

1. Enhancing the traditional CNN image classification to increase its accuracy.

Using the CNN model, we initially achieved an accuracy of 82%. Thus, to increase the accuracy of this model, we have introduced a new feature which divides images into smaller parts to classify it, hence, the accuracy achieved is much higher than regular CNN model which makes it more reliable in detecting fire/smoke. It can detect correctly even when there is very little fire.

2. Increasing the accuracy of the Random Forest model.

Random Forest basically builds an ensemble of decision trees, by applying the bagging method. Using the output of Decision Tree regressor, we can understand which parameter needs to be tweaked to get higher accuracy. The Decision Tree model is easy to interpret, and we can easily identify which parameters most affect the output.

VI. CONCLUSION

In this project, we evaluated 4 different machine learning models for the datasets and analyzed their accuracy individually. According to the classification and the accuracies achieved by the individual models, VGG16 CNN did a good job by achieving an accuracy around 96%.

A. Result Analysis

Based on the prediction accuracy, we think that the result of our project has a significant impact on users' ability to correctly detect grape leaf diseases. Still we believe that the algorithm can be improved.

B. Future Work

In our demonstrated models, at present we only have detection of 3 diseases along with healthy leaf prediction. We can make the algorithm more predictable with other grape leaf diseases as well by training it using other diseases and by finetuning it with hyperparameters to get more accuracy. Our models also have a scope of making improvements in predicting whether the grape leaf disease is in early or matured stages, so that appropriate measures can be taken before the crop yield gets worsened.

VII. REFERENCES

[1] *Li, X., Xie, H., Wang, R., Cai, Y., Cao, J., Wang, F., Min, H. and Deng, X. (2014). Empirical analysis: stock market prediction via extreme learning machine. Neural Computing and Applications, 27(1), pp.67-78.*

[2] Harshal Waghmare, Radha Kokare, and Yogesh Dandawate, "Detection and Classification of Diseases of Grape Plant Using Opposite Colour Local Binary Pattern Feature and Machine Learning for Automated Decision Support System.", 2016 3rd International Conference on Signal Processing and Integrated Networks (SPIN)

[3] Miaomiao Ji, Lei Zhang, QiufengWu, College of Engineering, NorthEast Agricultural University grape leaf diseases identification via UnitedModel based on multiple convolutional neural networks.

[4] Tanmay, Snehal. B. Gaikwad, International Journal of Computer Applications, Grape Leaf Disease Detection using Convolutional neural networks.

[5] Grape leaf disease identification using machine learning techniques,2019 International Conference on Computational Intelligence in Data Science (ICCIDS)