Plant Disease Classification using Pre-Trained Deep Learning Algorithm

Lakshay Chauhan

2986744

Submitted in partial fulfillment for the degree of

Master of Science in Big Data Management & Analytics

Griffith College Dublin
June 2020

Under the supervision of
Osama Abushama

Disclaimer

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the Degree of Master of Science in Big Data Management & Analytics at Griffith College Dublin, is entirely my own work and has not been submitted for assessment for an academic purpose at this or any other academic institution other than in partial fulfilment of the requirements of that stated above.

Signed: Lakshay Chauhan **Date: 18/06/2020**

Acknowledgement

First and foremost, we thank God for granting us knowledge, health, and the patience to successfully complete this project. We would also like to show our gratitude and appreciation to the following people:

Osama Abushama, our supervisor, for his generous and constructive feedback and continuous encouragement throughout the project. We also wish to take this opportunity to thank Osama Abushama for his advice about where to start with learning for the Project. Special thanks and gratitude go to all our colleagues and friends, who have been there cheering us on and supporting us. Finally, we would like to express my indebtedness to our parents, who have always been supportive and encouraged us morally.

Table of Contents

List of Figures

List of Tables

Abstract

Plant diseases are a significant danger to food security, but their rapid identification remains difficult in numerous parts of the world due to the lack of the necessary infrastructure. The advancement in computer vision due to deep learning has made a way for plant disease detection and its classification. In the past, many Machine Learning (ML) models have been used on plant disease for its detection and classification but after some advancement in Deep learning (DL) a subset of Machine Learning it can increase the accuracy. Using the plant disease image dataset of 38 classes are used to train the different DL models to detect different plant diseases based on the different test images. The performances of the trained algorithm on plant disease dataset and deep models are fined tuned by transfer learning are evaluated in this paper. The best model is the deep CNN model trained with transfer learning, which yields an overall accuracy of 98.85% on the test images. Other models like Alexnet, VGG16, and resnet50 are also evaluated in this paper. The proposed deep learning models may have great potential in controlling plant disease for modern agriculture.

Chapter 1. Introduction

Crop diseases are a main threat to modern agriculture production. The diagnosis of different plant diseases rapidly and accurately will help reduce yield losses [1]. In the past, crop disease severity is marked with the visual aid of plant tissue by trained experts. Human assessment of plant diseases is very costly and less efficient that can affect the modern agriculture. With the advancement in computer vision, the automated plant disease detection system is highly in demand by farmers in the agriculture industry. Plant disease classification using images from the new plant disease dataset will be a significant challenge. The recent trend in the use of various Deep neural network algorithms for plant disease classification has shown positive results [2]. The primary goal is to make use of different deep learning algorithms for training models on the plant disease dataset and then test the models on random leaf images.

Disease in plants or crops is majorly studied in the scientific area, focusing on the biological characteristics of diseases [3]. The problem of crop disease is a worldwide issue. The effects of crop diseases in plants cause significant losses to farmers. This project focuses on the recognition and classification of several plant diseases that affect the crops. Several techniques are applied to identify plant diseases. Plant disease identification by visual aid is a more difficult task and at the same time, less accurate and can be done only in limited areas. Whereas if an automated plant disease prediction system will take less effort, less time and become more accurate.

Transfer learning approach is been used in the project. Some models like Convolutional Neural Network (CNN) models had evolved that has increased its accuracy of classification. Alexnet is also successful deep learning model which uses pre-trained model for classification of plant disease. The Alexnet model had classified 38 classes in the given the dataset using 87K images. Resnet50 is another deep learning model that is used to classify 38 classes of plant diseases. VGG16 (Visual Geometry Group) is also deep learning model used for classification of disease using new plant disease dataset which is based on the stacked architecture of Alexnet with more convolution layers added to the model.

1.1 Transfer Learning

Transfer learning approach is the application of pre-trained deep learning models for classifying new class objects. In transfer learning, we can use knowledge (model history) from previously trained models to train for a new set of data. Transfer learning methods are better than traditional learning approach which are isolated. Deep convolutional neural network models may take days or even weeks to train on a very large dataset but using transfer learning involves the use of pre-trained models on one problem as a starting point.

The Deep Learning models will reuse the weights in one or more layers from the pre-trained models in the modified model. Transfer learning will improve the performances of deep learning models by train faster, reduce overfitting, and make better predictions (high accuracy score). Transfer Learning can be useful when there is insufficient data or a neural network cannot handle the new domain.

1.2 Goals

Plant diseases can affect the quality of fruits, vegetable crops and it can cause heavy losses in the agriculture industry [4]. Deadly plant diseases can result in high mortality in crops. The identification and classification of plant disease are very significant to develop a solution to prevent crop loss due to plant disease. The advancement in computer vision and object recognition has made to possible to diagnosis the plant disease through the images of the different leaves. In the past, plant disease identification has been supported by agriculture institutions such as local plant clinics or it can be done manually by crop experts. Nowadays the plant disease images are available online so we can use machine learning technology to make an application to identify the plant disease. The proposed system in this project will demonstrate which deep learning model will have the best accuracy that enables in the identification of plant disease through image recognition. This project will benefit the farmers for the identification of the different classes of plant disease using technology and they will not be relying on the traditional method.

1.3 Overview of Approach

The project is a python GUI application built using python. This project has divided into five subparts and thus follows a divide and conquer approach development process. The first four-part deal with training and testing of four models (Alexnet, VGG16, Resnet50, CNN). The fifth part of the project is to design the GUI using QT designer in anaconda and implement all four models in the GUI and test different plant images.

The new plant disease dataset has 87K different images of healthy and diseased crop leaves which are categorized into 38 classes and the total dataset is divided into 80:20 ratio and it is augmented and it does need pre-processing (Resizing of Original Image). Deep learning models like Alexnet, VGG16, Resnet50, and CNN are trained on the dataset and are stored in pkl format. The second part is to test the model on some test images and check the accuracy of the four models (it takes the highest probability from probabilities of 38 classes) and identify the plant disease on the test image. After training on the models, accuracy and loss are plotted in the form of line charts.

1.4 Document Structure

The project document is divided into chapters. The rest of this document is as follows. Chapter Two provides a literature review of the area of Transfer Learning and deep learning models, in addition to related work. Chapter Three describes the methodology and deep learning models architectures of my project. In Chapter Four, I discuss the system specifications including three parts of the project development followed by System Design. This section also discusses the internal working of models and use case scenarios. Chapter Five provides the implementation details. In Chapter Six, details of the validation and plotting of my project are provided. I also discuss the validation of deep learning models used and the results including any revisions to the overall design and implementation that were deemed necessary. Last Chapter Seven presents conclusions and future work.

Chapter 2. Background

Machine learning and Deep learning are the terms in computer science that are receiving tremendous attention from the scientific community. Neural network is pushing the computer vision technology. Plant disease classification can be a complex task as it depends on the people who are experts in plant disease knowledge. Traditional methods like a consulting expert for plant disease is costly and time-consuming [5]. The development process for making an automatic plant disease classification system that is applicable for a large number of plant classes is a difficult task but it will be cheaper and less time consuming [6]. A convolutional neural network has shown countless results in image classification tasks that can be used to improve accuracy in cases like plant disease classification. Studies show that the investment in neural networks and deep learning models are increasing rapidly and is being added on open source software like Anaconda. This section discusses various technologies and methods which were referenced for conducting the literature reviews and some of the related works studied.

2.1 Literature Review

Identification of plant diseases manually is very expensive and time consuming and also requires skilled professionals. More researches have taken place in finding an automated identification and classification system which can be cost-effective and less time consuming. Due to advancement in deep learning and computer vision various solution are been put forth to tackle this problem providing solution to them.

Deep learning neural networks model may take days or even weeks to train on large datasets. To reduce this timing to train the models there is a process to re-use the model weights from pre-trained models that were developed for computer vision benchmark dataset using image recognition task. The technique discussed above is called transfer learning that is motivated by the fact people can intelligently apply knowledge learned previously to solve new problem faster with better solution or improving accuracy [7]. The motivation behind the idea of transfer learning in the field of machine learning was discussed in NIPS-95 workshop on “learning to learn” [8] which focused on the need for machine learning methods that can retain and reuse previously learned knowledge.

The Keras documentation [9] provide ample information how to use the API for this project. It is deep learning API that runs on the top of tensorflow that can efficiently run on GPU or CPU of the system. Some of APIs used in the project are model training, Data pre-processing, optimizer and layers API of keras.

The Tensorflow documentation [10] provide multiple information on image classification, transfer learning with pretrained deep learning models and how to use them to train our model for the project. There is also reference in the documentation how to train the model on the GPU of the system rather than using CPU power to reduce time.

Four deep learning models are used in the project like CNN, Alexnet, VGG-16 and Resnet50. The first model is the Alexnet is fast GPU-implementation of CNN for image recognition. The Alexnet model contains eight layers in which first five are convolutional layers, some of the layer are followed by max-pooling layer and last three layer are fully connected layer [11]. It uses ReLU activation function which shows improved training performance and increase accuracy of the model.

The second model in discussion is the CNN (convolutional neural network) which is a linear operation that uses general matrix multiplication in convolutional layer. CNN consist of input and output layer (with some hidden layer) [12]. The hidden layer are series of convolutional layers that do multiplication or dot products. It is ReLU layer as activation function followed by some pooling and normalized layers.

The third model is VGG-16 (Visual geometry group) is convolutional neural net used for image recognition which is preferred for extracting feature from the image. VGG-16 consist [13] of 16 convolutional layers followed by max-pooling layer that use ReLU activation function which are of uniform architecture. VGG can be achieved by transfer learning in which model is pretrained on a dataset and accuracy can be updated.

The last model is the Resnet50 (Residual neural network) is a deep learning model that consist of 50 convolutional layers with some skip connection to jump over some layers. Resnet models are implemented [14] with double or triple layer that contain non-linearities and normalization layer in between.

Kaggle is huge repository containing over 20,000 public datasets and over 300,000 public notebooks to serve for any type of project. Reference from the website, a dataset file is chosen for plant disease classification project [15] which was used for this project.

Disease classification and detection technique can be used for identification of crop disease in the field [16]. Before identification and classification, pre-processing is done on the dataset before feature extraction. RGB images of leaf are converted into white them converted into grey kevel image to extract the vein of the leaf to identify and classify the infected disease. Classification and identification are the important part of the project.

Identification of plant diseases is important part for preventing the losses in crop field. It requires lots of work and time for experts to identify and classify diseases in field. New technology like image processing is used for detection of plant disease [17]. This paper discusses the method of detection of crop plant diseases using their leaves image and various techniques to segment the part of plant disease. Classification and feature extraction are the techniques to extract the features of plant disease and then classify into different classes. The accuracy of this method is very important for successfully cultivation of crop.

The survey on plant diseases detection using image processing techniques are showing positive result. Plant diseases in crops causes reduction in quantity and quality of agricultural byproducts. Identification and classification of plant diseases by naked eye is difficult for farmer. This paper presents a survey of propose system that can detect tomato and corn diseases using their leaves images [18]. Some of the plant disease cannot be identified using a single method.

2.2 Related Work

Classification and identification of plant diseases manually is very expensive, time consuming and require experts professional. As a result, research has taken place in finding an automated detection system for classification of plant disease which are reliable, accurate and less time consuming. Due to recent advancement in deep learning and computer vision there are various method that tackle this problem are mentioned below:

The paper [19] presents the technique of detecting and identification jute plant disease using image processing. In the proposed system, image is captured and it resize the image to store in the database. Then the image is enhanced in quality and noises are removed. Hue based segmentation is applied on the image. The image is converted into HSV from RGB as it helps extract the feature of the plant. This approach has significantly detected and identify stem-oriented diseases for jute plant.

The image edge detection segmentation techniques in which the captured plant diseases image is processed for identification and classification. RGB color feature image segmentation is carried out to get disease spots on the infected leaves. The image feature such as edges, shape and color are extracted for plant disease spot to identify the infection and it can used later for pest control [20]. The proposed system has three part for cotton leaf spot by cotton leaf color segmentation, edge detection based on leaf image and classification of disease.

This proposed system discusses about the cucumber disease detection system [21]. The methods include image acquisition, image preprocessing, feature extraction with Gray-level co-occurrence matrix (GLCM) and finally classify into two types: Unsupervised classification and supervised classification of the cucumber disease.

The image processing techniques are used to detect the citrus leaf disease [22]. This proposed system has method like image preprocessing, segmentation of the citrus leaf using K-means clustering to determine the infected areas, feature extraction and classification of disease. Using GLCM (Gray-level Co-occurrence matrix) for feature extraction and classification using support vector machine (SVM).

The proposed automatic pixel-based classification system for detecting the infected part of the plant leaf region is presented in the system [23]. The algorithm used in this project have been tested extensively. To classify each pixel in leaf image linear SVM is used. The presented algorithm could be extended for other detection task like color information. The project is done in three steps. First, divide the image dataset into foreground and background. For the second step, support vector machine is applied to leaf pixel image belonging to foreground. The final step in the project is to refinement by neighbourhood check to delete false-classified leaf pixel from the second step.

Paddy plant are the important part of the continental region. The proposed system [24] says RGB images are converted into gray scale image. Histogram equalization and contrast adjustment are some enhancement techniques used for the image quality that are acquired from paddy plant. Different types of classification features like SVM, ANN classification is used in this project. Feature extracted are like texture, structure and geometric feature. By using SVM and ANN identify and detection of disease of paddy plants.

The paper [25] discusses tomato disease detection using deep learning and computer vision. The Gray-scale image is converted into binary image depending on threshold value. The image segmentation is done by threshold algorithm. The threshold value is given color like red, green and blue. It is not very reliable method for distinguishes red tomatoes from other color. K-means clustering algorithm is used to overcome the drawback to create a number of non-hierarchal clusters. It separates the infected parts from the leaf the RGB was converted into YcbCr to enhance the feature of image. The last step is to calculate the percentage of infection in the tomato.

The paper [26] have proposed a technique that can be used for detection of paddy plant disease by comparing it with 200 healthy paddy plant images and 150 sample of disease1 and another 200 sample of disease2. It may not be sufficient enough to detect disease in paddy field or classify it using training data is not linearly separable.

Above are some system that were proposed in different research around the world and it inspire for theses project idea and help to create an easy system to identify and classify of plant disease using pre-trained deep learning models. Later in this documentation, it will discuss the implementation and testing part of the project.

Chapter 3. Methodology

The tools that will be used for the development of the python application is important for its completion. The python application was implemented on a Dell laptop with the following configuration: Windows 10 operating system with Intel core i7-8750H processor, 16GB RAM, having network connectivity, Graphic Card is NVIDIA GeForce GTX 1060 with Max-Q Design for training the model on GPU, 20 GB of free hard drive space was used.

The project was initially planned to be built using a top-down approach, but deep research and analysis revealed that the model that will be used in the project should be trained that could cost lots of effort and time which would be out of the scope for this project. So, the project implementation using the bottom-up approach, using the transfer learning technique utilizing the pre-trained models to train on the dataset. TensorFlow GPU and Keras are the libraries are used in the project. The pre-trained network inside of Keras is capable to recognize 1000 objects. Anaconda was software will be used in the project that include a QTdesigner that will be used to design the UI of the application.

The project consists of multiple tasks in different areas, so a divide and conquer approach was used. The project is split into three parts. Part one is for training each model on train images of the dataset using TensorFlow GPU and Keras and storing the results later using for testing. The second part is to test the model on test images and determine the accuracy of each model. The last part of the project to design the UI using QTdesigner and any user can use this application to determine an image of a leaf is diseased or healthy. Each part of the project is tested.

The following sub-sections will discuss the programming language, models and dataset that are used for implementation of the project.

3.1 Python Language

Python is an interpreted, high-level, general-purpose programming language developed by Guido van Rossum is an open-source software [27]. Python is a powerful programming language, used that runs on one platform does not need to be recompiled to run on another platform. Python has many open source libraries that can be used in this project. TensorFlow-GPU is one of the libraries for the project to train the models on the GPU instead of the CPU of the system. Another important library used in the project was Keras that is an open-source neural network library in the python. Other libraries that are used are OpenCV that is used for real-time computer vision.

3.2 Alexnet

Pre-trained Alexnet model (as shown in Fig. 3.2.1) consists of five convolutional layers (followed by maximum pooling layers) and three fully-connected layers and finally a 1000-way SoftMax classifier. The first convolution layer, a filter dimension 11 X 11X 3 which represents height, width, and depth which is applied over the input image of dimension 224 X 224 X 3. A total of 96 filters are applied in the first layer and 96 activation maps are generated from the rectified linear unit (ReLU) layer of the first convolution layer. Similarly, the second convolution layer with 256 feature maps with a filter dimension of 13 X 13 X 256. Third, fourth and fifth layers have the first two use 384 feature maps and third used 256 filters. The three-convolution layer is followed by a maximum pooling layer with filter 3 X 3, a stride of 1. The sixth layer output is flattening through a fully connected layer with 9216 feature maps with each size 1 X 1. Alexnet models, consist of several convolution layers are followed by ReLU, maxpooling and normalized layers. Sixth and seventh layer have 4096 neutrons where all neutrons are connected to each other. The output layer has softmax output layer with 1000 possible value.

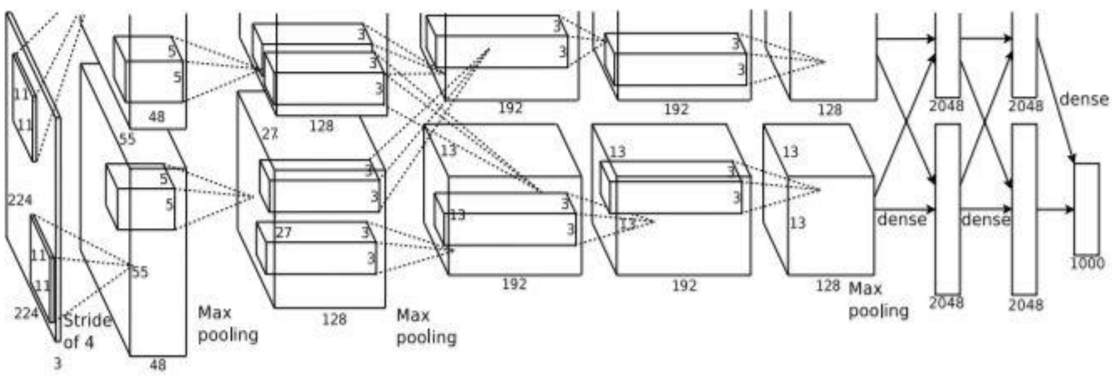

Figure 3.2.1 Alexnet Architecture

3.3 CNN

Convolutional neural network (CNN) is a model (as shown in Fig. 3.3.1) that works on the unstructured image inputs and converts them to classification output labels. CNN belong to the category of multi-layer neural network which can be trained to learn require features for its classification. CNN models consist of convolutional activation, maxpoooling and fully connected layer. The first layer is convolutional layer called a Convolutional2D that use 32 filters with size 3 X 3 each. The max pooling layer with pool size 3 X 3. Another convolutional layer had 64 filters with size 3 X 3 each with the following max pooling layer with pool size 3 X 3. The flatten layer converts 2D matrix data to a 1D vector before building the fully connected layer. Use the fully connected layer with 1024 neurons. Final output layer which 50 neurons for 38 classes and softmax activation function to output the prediction for each class.

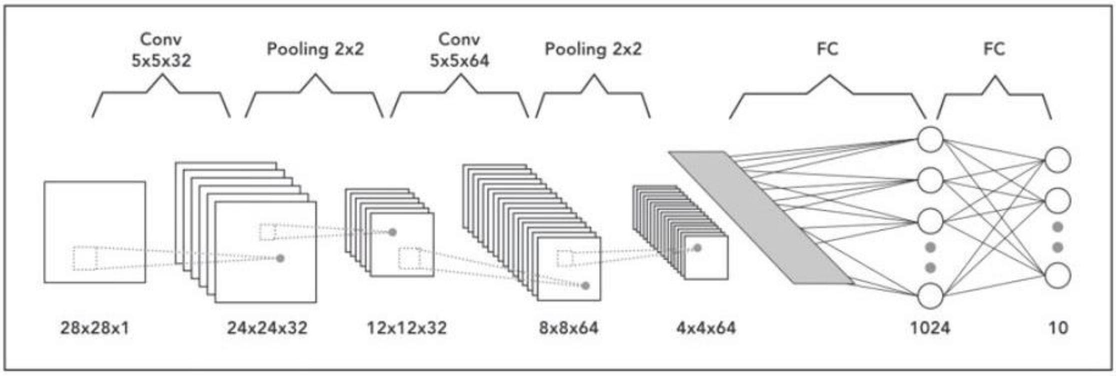

Figure 3.3.1 CNN Architecture

3.4 VGG-16

Pre-trained VGG-16 (Visual Geometry Group) is stacked architecture of Alexnet with more of convolutional layers added to the model (as shown in Fig. 3.4.1). The VGG-16 consist of 12 convolutional layers following by maximum polling layers and four fully-connected layers and finally 1000-way softmax classifier. The input for VGG-6 is 224 X 224 X 3 image which passes through two convolutional layers with 64 feature maps and pooling with a stride of 14. The dimension of image changes to 224 X 224 X 64. Similarly features are for next 10 layers in the VGG model. The thirteen-layer output is flattening through connected layer. The fourteen and fifteen layer connected layers with 4096 units. The output layer has 1000 possible values. The VGG model improves the discrimination of each layer.

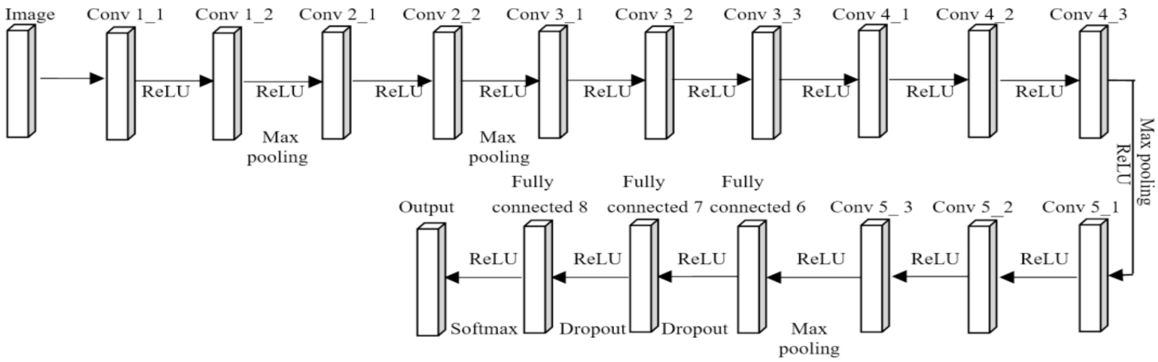

Figure 3.4.1 VGG16 Architecture

3.5 Resnet50

Resnet50 (Residual Networks) has 50 layers (as shown in Fig. 3.5.1) which has network that can take the input image having height, width as multiple of 32 and 3 as channel width. The 2D convolution has 64 filters with maxpooling use 3 X 3 each size. After the initial stage of the residual network and it has 3 residual blocks containing 3 layer each. As the we move from one layer to another, the channel width is doubled and size of the input image will be reduced to half. Resnet model can learn an identity function.

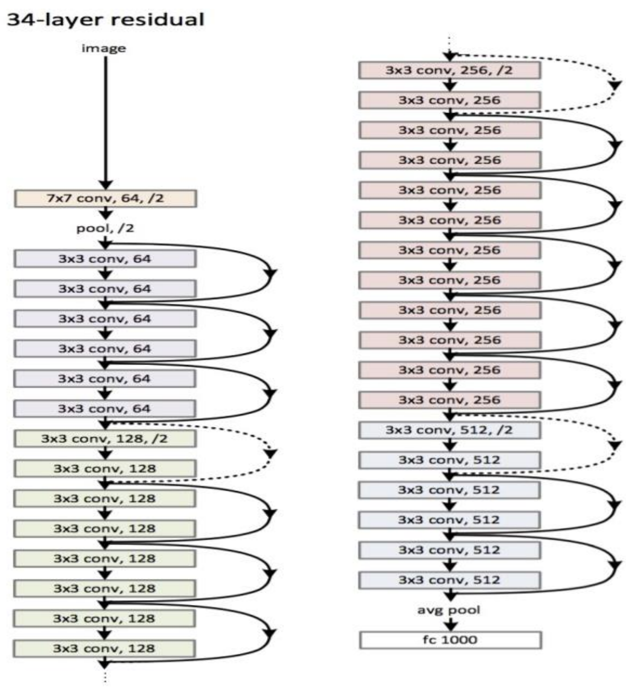

Figure 3.5.1 Resnet50 Architecture

3.6 Dataset

The images of 38 different class of diseases and healthy samples of crop leaves (as shown in Fig. 3.6.1) were obtained from new plant disease dataset. The dataset consists of 87K RGB images of diseases and healthy crop leaves which are sorted into 38 different classes. The dataset is divided into 80:20 ratio of training and validation of the images. The project testing are done using some unlabelled image that are randomly chosen for testing purpose.

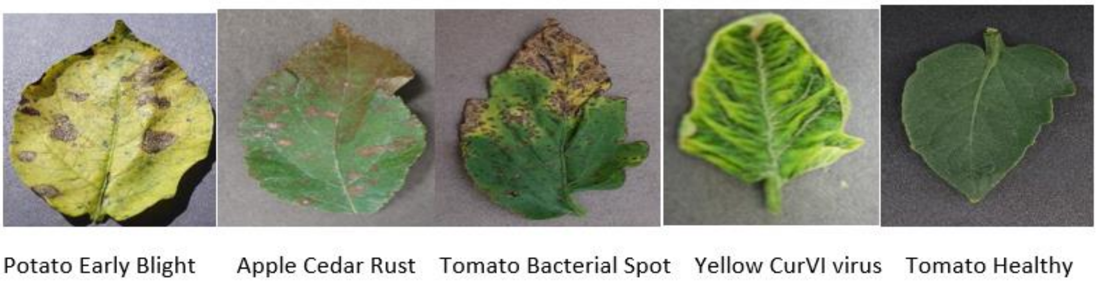

Figure 3.6.1 Dataset Sample Image

3.7 Classification

The convolutional neural network can be used for the creation of a computational model that works on unstructured image inputs and convert them to classification output labels. Deep models with pre-trained models will evaluate the feature of each image it is trained on and feed to the model. The models that are used in the project are pre-trained but it will be trained using transfer learning on the new dataset with several architectures like Alexnet, CNN, Resnet50, or VGG16 and test them on different images to check and compare the accuracy of different models used. The design is created like, if user browses image in GUI it will predict if the leaf is diseased or healthy.

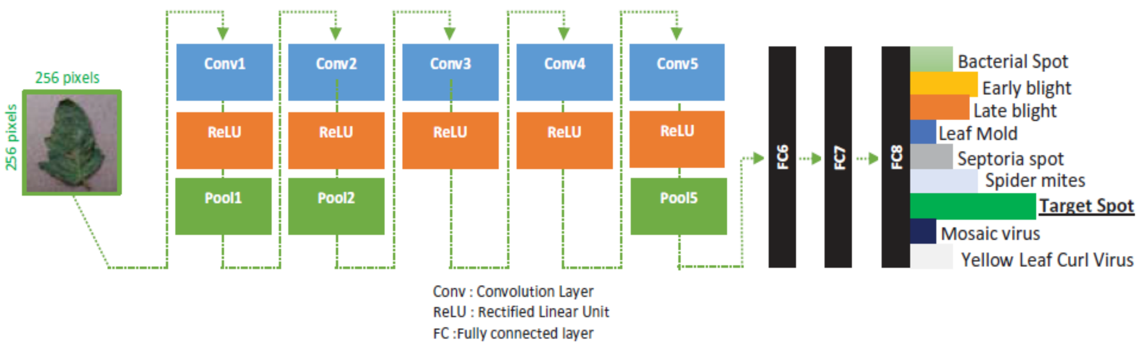

Figure 3.7.1 General Model for Disease Classification

Chapter 4. System Design and Specifications

In this chapter will discuss the system design architecture, deep learning model design, its various tables and their relation and also present a use case system.

4.1 System Design

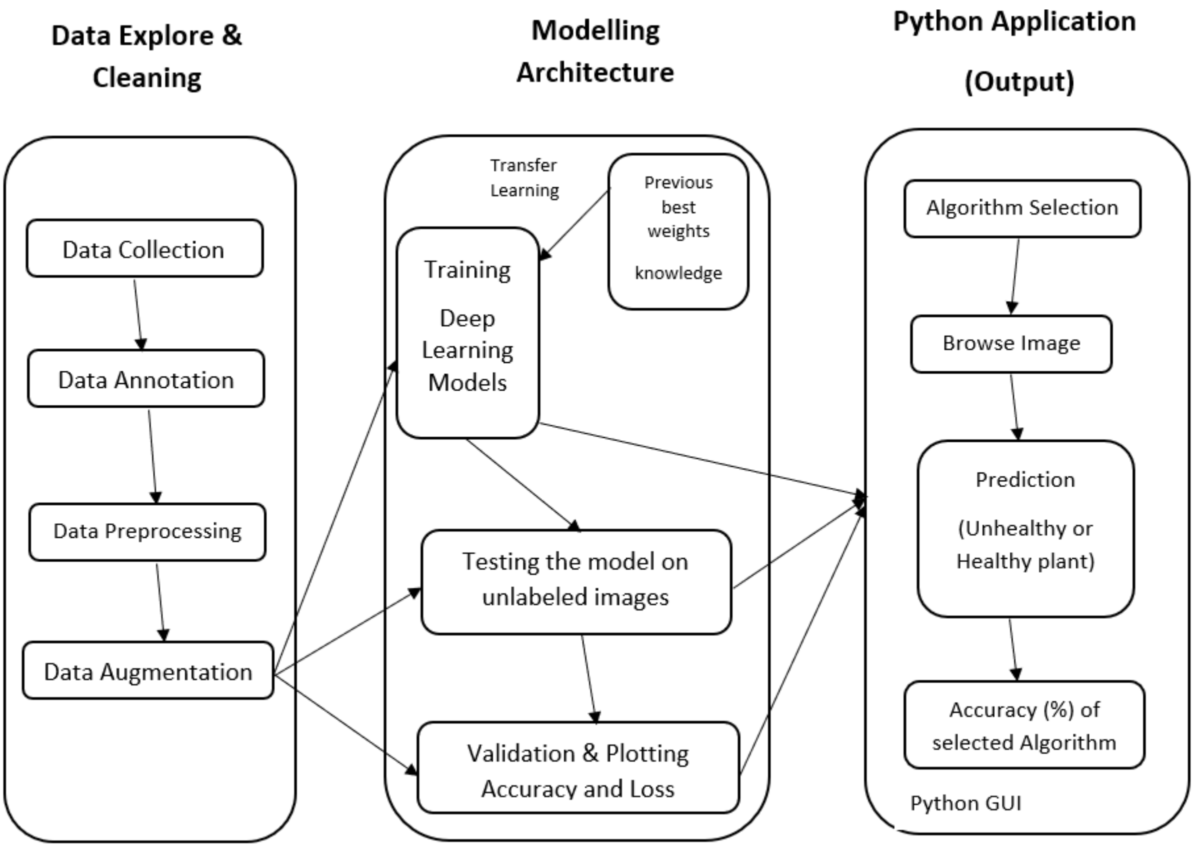

Figure 4.1.1 System Architecture

Figure 4.1.1 shows how the plant disease classification system works from data collection to GUI deployment (output). The system of the project was developed in three steps. First part is Data explore and cleaning of the dataset, second part is Modelling Architecture that includes training, testing and validation of each deep learning model. Last part includes the development of GUI application where a user can easily browse any unlabelled image and predict which type of plant disease that image have using different algorithm and see its accuracy (%).

The first step in the data explore and cleaning part of project development is Data collection process for which Kaggle website was the best to find a good dataset. The chosen dataset “New Plant Diseases Dataset” contain 87,000 RGB images of healthy and diseases crop leaves which are categorized into 38 different classes. In this system, some of commercial/cash crop are considered like tomato crops, corn crops, potato crops, soybean crop and some fruit crops like apple, blueberry, peach and strawberry.

The second, third and fourth part the data explore and cleaning part of project development is data annotation, preprocessing and augmentation. Data annotation is the task of automatically generating description of image which is important for image search and identification. Annotation process is used in the project to label each 38 class and locate the infected area in the leaf crop image. The dataset is recreated using offline augmented technique from the original dataset and total dataset is divided into 80:20 ratio for training and validation. Images in the dataset are of various formats along with different resolution and quality. Images are pre-processed using deep neural-network for feature extraction. Images in the dataset were resized to 224 X 224 in case of VGG16 and Alexnet models and resize to 256 X 256 in case of Resnet50 and CNN to reduce training time using OpenCV framework [28]. Pre-processing technique used in the deep learning models using keras API are like:

```
from keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(rescale=1./255,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   width_shift_range=0.2,
                                   height_shift_range=0.2,
                                   fill_mode='nearest')

valid_datagen = ImageDataGenerator(rescale=1./255)

batch_size = 128
base_dir = ".\\dataset"

training_set = train_datagen.flow_from_directory(base_dir+'/train',
                                                 target_size=(224, 224),
                                                 batch_size=batch_size,
                                                 class_mode='categorical')

valid_set = valid_datagen.flow_from_directory(base_dir+'/valid',
                                            target_size=(224, 224),
                                            batch_size=batch_size,
                                            class_mode='categorical')
```

Keras API provides preprocessing utilities like keras.preprocessing help the object can be used to train the model. ImageDataGenerator class used to generate batches of tensor image data with real time data augmentation. The input image is resized for the deep learning model for training part.

After the completion of Data explore and cleaning part of the project next part of the project is modelling architecture which includes three steps training, testing and validation (plotting accuracy & loss) on the dataset which will generate files, which will be used later in the GUI development.

The first step of modelling architecture is to train the four deep neural models (Alexnet, CNN, VGG16 and Resnet50) using pre-trained weights via transfer learning and base convolutional neural network will use previous knowledge to extract useful and meaningful feature from the new image sample. Using transfer learning for image classification is important because if dataset is large it can take advantage of learned feature maps without having to start from the scratch. During the training of the model ModelCheckpoint class which is callback with training using model.fit() to save weights at some interval to keep the model that has achieved best accuracy after some epochs. Training models (Alexnet) are like:

```
# checkpoint
from keras.callbacks import ModelCheckpoint
weightpath = "best_weights_9.hdf5"
checkpoint = ModelCheckpoint(weightpath, monitor='val_acc', verbose=1, save_best_only=True, save_weights_only=True, mode='max')
callbacks_list = [checkpoint]

#fitting images to CNN
history = classifier.fit_generator(training_set,
                         steps_per_epoch=train_num//batch_size,
                         validation_data=valid_set,
                         epochs=25,
                         validation_steps=valid_num//batch_size,
                         callbacks=callbacks_list)
```

After completion of training of model, the history and model files are saved as pkl format. The second step of modelling architecture is the testing phase in which unlabelled image are tested on different trained deep neural network models using saved models file and test how accurately a model can predict the unlabelled image. The testing phase includes loading of saved models and using them to test random image for identification of plant disease.

The last step of modelling architecture is the validation phase in which plotting of accuracy and loss of each deep learning models used in the project. The history file saved during the training phase are used for plotting each graph and are used for comparison of each four models and determine which deep learning model had the best accuracy rate. The last phase of system design is to make python GUI application from all the above steps that will be discussed in implementation part of the project.

4.2 Deep Learning Model Design

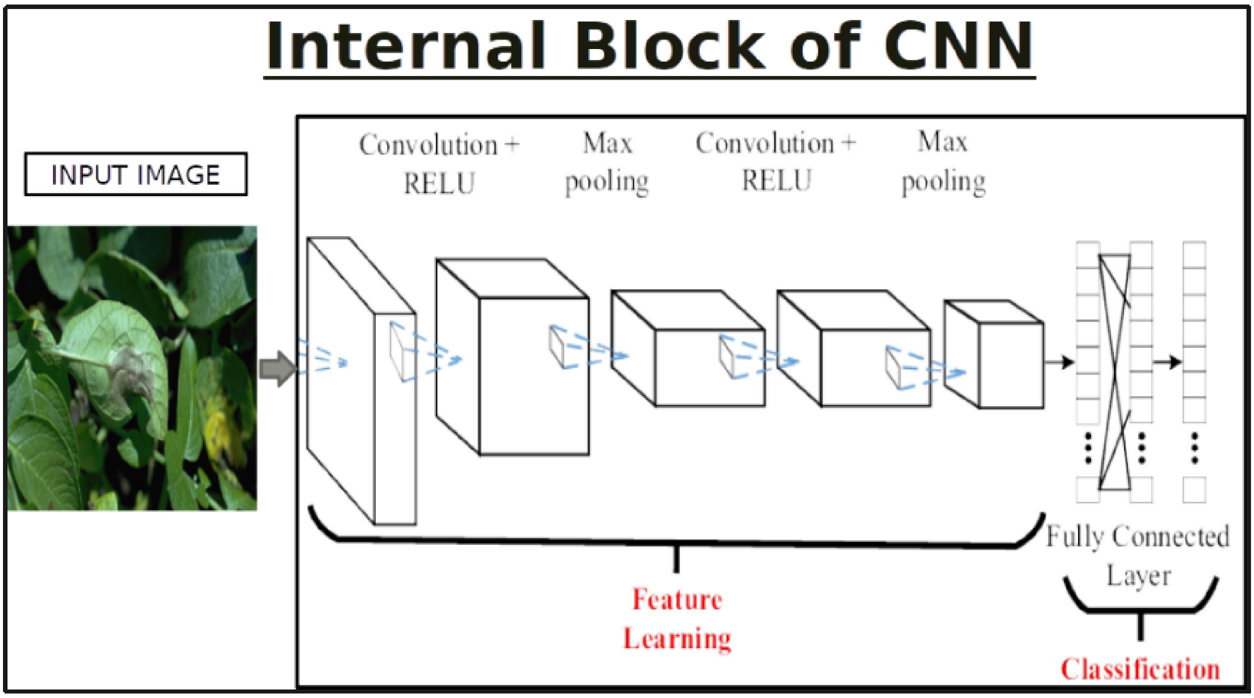

Figure 4.2.1 CNN Architecture

Figure 4.2.1 shows the internal working of Convolutional neural networks (CNN or Conv Nets) as one of the examples to learn the internal working of deep learning models. CNN deep learning model are used in the application of image recognition systems or Natural language Processing (NLP). Convolutionals are simpler and more efficient because it reduces the number of parameters. There are five steps to build a convolutional neural network (CNN) are Convolution operation, ReLU Layer, Pooling Layer (Max pooling), Flattening and Fully Connected Layer.

Convolution is the first layer to extract features from the input image and it learns the important feature from the input images. The ReLU layer (Rectified Linear unit) is used for non-linear operation and is introduced in the CNN model for non-linearity. The third step is Pooling layer that is used to reduce the number of parameters by downsampling and only retain useful information to process. There are two types of pooling layers to choose Max Pooling (Preferred) and Average & Sum pooling. The fourth step in the process is Flattening to flatten matrix into a vector like vertical and then it is passed to the input layer. The last step in the building of CNN is Fully Connected Layer in which the flatten vector is passed into the input layer and then it combines these features to create a model (like CNN in this case). The above example shows the building and internal working of the CNN model that is one of the deep learning models used in this project.

4.3 Use Case Design

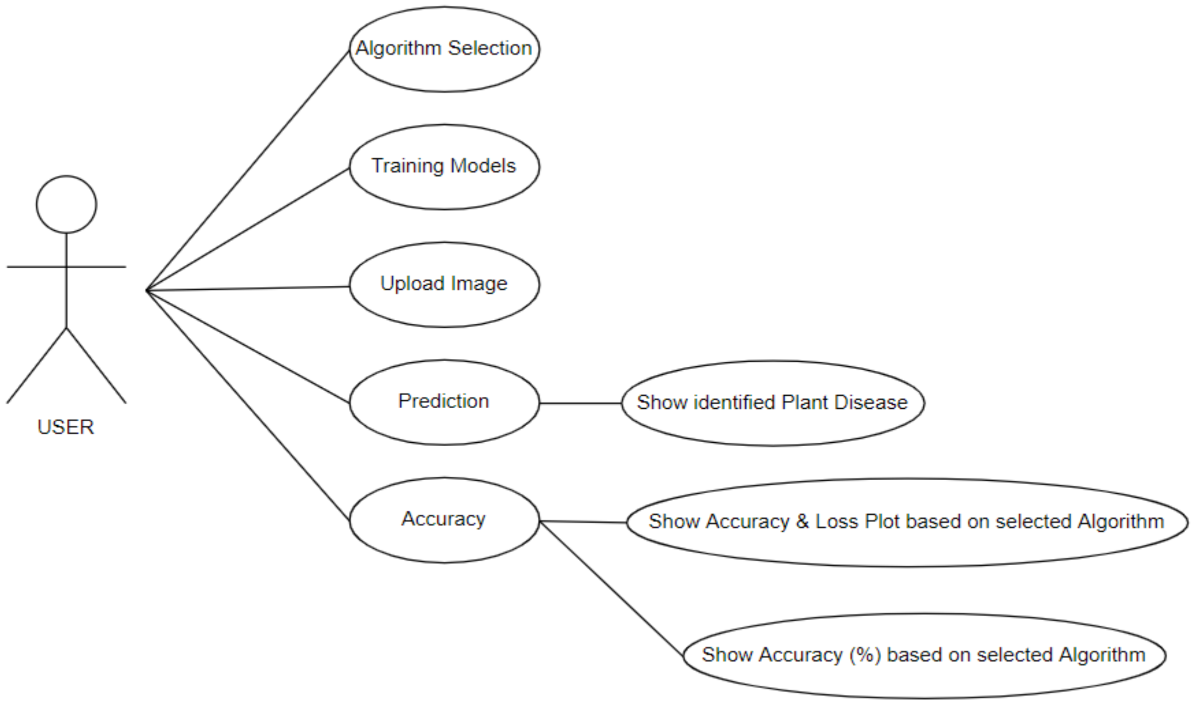

Figure 4.3.1 UML Diagram

Figure 4.3.1 shows the user use-case diagram of the Python GUI application. The application helps the user to browse the image (Upload Image) and identifies the plant disease based on the selected algorithm and shows the name of the plant disease in the dialog box. The user is also given the option to train any deep neural network model again by using the algorithm selection option given to the user. The prediction button in the GUI application lets users show plotted line graph of accuracy and loss of each selected deep learning algorithm. It also lets users show the accuracy (%) percentage in the box. The user interface of the GUI application is very user friendly and any user without prior knowledge can use the application easily.

Chapter 5. Implementation

As stated in Chapter three, this project was divided into three tasks: Training the model, testing unlabelled images and Accuracy (%) of each four model and last part is deployment of above two parts in the python GUI application. The foremost thing to do for this project is to select a good dataset. Before the all the development, Software requirement are Anaconda3 (version 5.2.0) and Nvidia cuda 10.0 (For tensorflow-gpu so model can train on GPU of the system).

The first task in this project to select a dataset for which Kaggle [15] is used for this part. The dataset “New Plant Disease Dataset” has 87,000 RGB images of healthy and diseases crop leaves which are categorized into 38 different classes. The dataset is pre-augmented and some preprocessing techniques are used like resize of original image so the deep neural model can train on the images. Keras API are used for the preprocessing technique [29] in this project.

```
from keras.preprocessing.image import ImageDataGenerator
train_datagen = ImageDataGenerator(rescale=1./255,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   width_shift_range=0.2,
                                   height_shift_range=0.2,
                                   fill_mode='nearest')
valid_datagen = ImageDataGenerator(rescale=1./255)
```

After the preprocessing phase of the project the next part is training of the deep neural models using transfer learning technique. The training phase include to train the deep learning models Alexnet, CNN, VGG16 and Resnet50 using previous weights so it can train lot faster on new dataset. The use of pretrained weights for training deep learning models on large dataset has enhance the accuracy rate in this proposed system. The below figure (5.1) [30] shows the general retraining process of deep learning models using pre trained weights. There is four step procedure in retraining the deep learning model: first is to load the pretrained weights, second step is to add last three to five layers depending on the model to the neural network, third step is to train the model with a new dataset (In case “New Plant Disease Dataset”) and last step to perform the testing and measure accuracy (%).

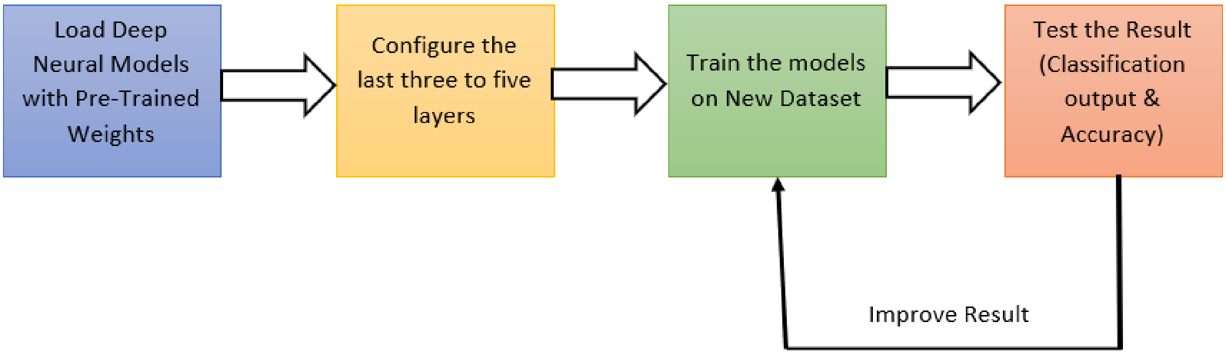

Figure 5.1 Retraining Process of Deep Learning Model

The deep neural models consist of several convolution layers followed by max pooling layers and last three fully connected layers and a softmax classifier (depending on model). The table 5.1 shows the Alexnet architecture (one of the models used in the system) that are reconfigured, modified and adjusted to support retraining process. The above process is similar for another three deep learning models CNN, VGG16 and Resnet50.

Layer	Function	Filter Size	Stride	Activation
Conv 1	Convolution	11 X 11 X 3	4	ReLU
Pool 1	Max Pooling	3 X 3	2	ReLU
Conv 2	Convolution	5 X 5 X 48	1	ReLU
Pool 2	Max Pooling	3 X 3	2	ReLU
Conv 3	Convolution	3 X 3 X 256	1	ReLU
Conv 4	Convolution	3 X 3 X 192	1	ReLU
Conv 5	Convolution	3 X 3 X 192	1	ReLU
Pool 5	Max Pooling	3 X 3	2	ReLU
Output	Fully Connected Layer	1000	-	Softmax

Table 5.1 Alexnet Architecture Table

The models are compiled before the training process start. The project is a multi-class classification (more than two exclusive targets) problem so categorical cross-entropy was used. Categorical Cross-entropy is defined as:

$$\prod_{c=1}^{C} y_c(\mathbf{x}, \mathbf{w}_c)^{t_c}$$

c is the index running over the number of classes

Categorical cross-entropy helps to calculate a score that summarize the average difference the actual and predicted probability distribution for 38 classes in this problem. The score is minimized and perfect cross-entropy value is 0. The softmax activation are used in this case to predict the probability of each class. The below code shows the how models are compiled in this project.

```
classifier. Compile(optimizer=optimizers.SGD(lr=0.001, momentum=0.9, decay=0.005),
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

After training of the models are completed depending of the epochs set in each model and all the files generated after the training are saved and later used for the testing phase of the project and check that how accurately each model can identify the plant disease. Below is code part for training of the model and files generated by its result.

```
history = classifier.fit_generator(training_set,
                         steps_per_epoch=train_num//batch_size,
                         validation_data=valid_set,
                         epochs=25,
                         validation_steps=valid_num//batch_size,
                         callbacks=callbacks_list)

with open("alexnet_train_history.pkl", 'wb') as f:
    pickle.dump(history, f)

#saving model
filepath=".\\saved_models\\AlexNetModel.hdf5"
classifier.save(filepath)
```

After the training phase completion and the second phase of the implementation is testing of each model on unlabeled image for plant disease identification. The testing of each deep learning model has defined a list (array) of each plant disease name [31]. The list is later used when testing of image starts it generates a confusion matrix which has probability of each matching disease and the highest probability will be selected and its corresponding class name is selected from the list. The below code is defining the critical part of testing unlabeled image:

```
def Predict_Test_Image_File(model):
    root = tk.Tk()
    root.withdraw()
    imageFileName = filedialog.askopenfilename()
    image = cv2.imread(imageFileName)

    image = cv2.resize(image, (224, 224))
    image = image.astype("float") / 255.0
    image = img_to_array(image)
    image = np.expand_dims(image, axis=0)
    predictions = model.predict(image)
    print(predictions)
    d = predictions.flatten()
    j = d.max()
    for index,item in enumerate(d):
        if item == j:
            class_name = li[index]
    print('\n\n')
    print(class_name)
```

In the testing phase after the identification of the plant disease or its healthy is determined and then validation part of the project includes plotting of accuracy and loss of each deep learning model (Alexnet, CNN, VGG16 and Resnet50) and can compare accuracy (%) of four models against each other and determine which deep learning model are best. The history files generated during the training part of the project are used to plot accuracy and loss of the four models. The below code shows how to plot the accuracy and loss graph:

```
acc = history.history['accuracy']

val_acc = history.history['val_accuracy']

loss = history.history['loss']
```

```
val_loss = history.history['val_loss']
epochs = range(1, len(loss) + 1)

plt.plot(epochs, acc, color='green', label='Training Accuracy')
plt.plot(epochs, val_acc, color='blue', label='Validation Accuracy')
plt.title('Training and Validation Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')

plt.plot(epochs, loss, color='pink', label='Training Loss')
plt.plot(epochs, val_loss, color='red', label='Validation Loss')
plt.title('Training and Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
```

The last phase in the proposed system is deployment of the before two phases into python GUI application that has a user-friendly interface that any user without prior knowledge can use the application for identification of plant disease. QTdesigner (provided by Anaconda3) has been used to design the UI of the application. It is a drag and drop system where it is easy to develop an interface and, in the backend, it is HTML and CSS working. After the development of the interface and the file is saved in .ui format. The next part is to convert the .ui format to .py format (python format) using pyuic tool provided by QTdesigner and command is pyuic5 -x filename.ui -o filename.py.

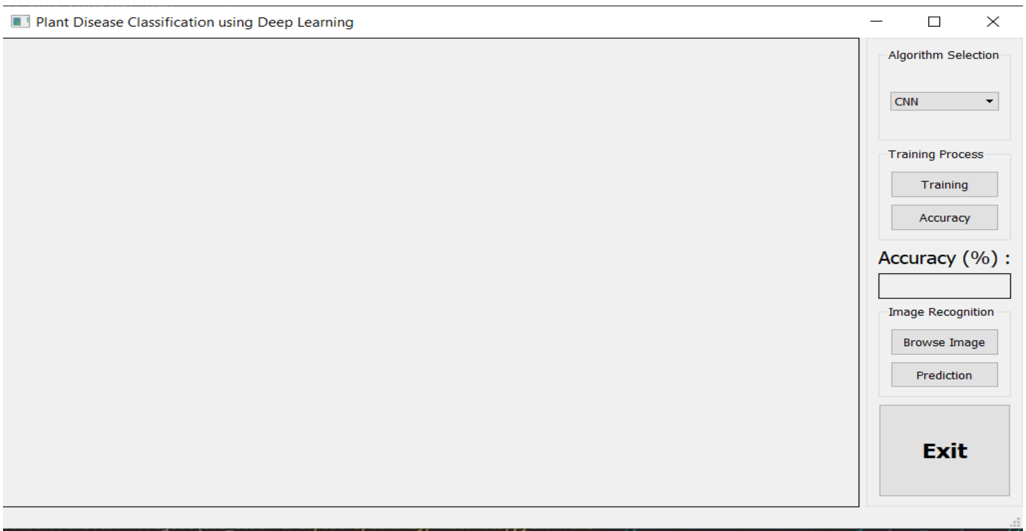

Figure 5.2 Proposed System UI

The figure (5.2) shows the user interface designed for the plant disease classification system. The main python file is created to connect the model files and UI files to make a working system. This was the final step to complete the proposed system. After some reviews from peers the deep learning models are trained by changing the epochs value to see if the accuracy (%) improves and each model can efficiently identify the plant diseases.

Chapter 6. Testing and Evaluation

After the completion of three phase of the project its time to evaluate the performance of the proposed system using some set of quantitative measure comprising of accuracy have been used. The deep learning models were tested on the unlabeled images of crops that are mixture of healthy and disease plant leaves and then check them how accurately it can identify them. Table 6.1 tells the accuracy score of each model after it is trained on the chosen dataset using previous weights (via transfer learning).

Model	**No. of Epochs**	**Accuracy (%)**
Alexnet	25	96.89
VGG16	15	98.05
CNN	25	98.85
Resnet50	5	98.24

Table 6.1 Accuracy Result

The maximum validation accuracy of 96.89% was obtained over 25 epochs of training the Alexnet deep learning model. This is an actual measure of the classification made by this deep learning model. The plots of training and validation accuracy and loss against the epochs (figure 6.1) provides the visualization and indication of the speed of deep learning model.

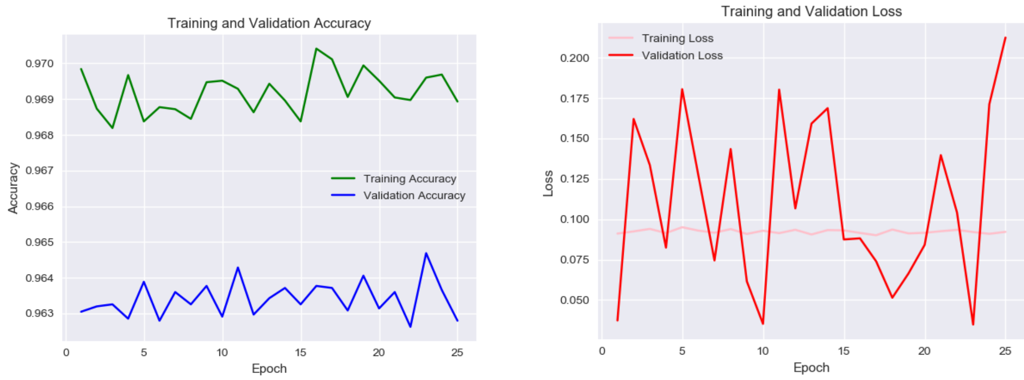

Figure 6.1 Alexnet Model Accuracy & Loss Graph

In case of Convolutional Neural Network (CNN) model maximum validation accuracy achieved was 98.85 after 25 epochs completion while an average accuracy of 98.5% has been obtained. Another important observation was that the model has stabilized after 15 epochs and accuracy metrics do not improve after 12 epochs. The plots of training and validation accuracy and loss against the epochs (figure 6.2) provides the visualization deep learning model.

Figure 6.2 CNN Model Accuracy & Loss Graph

Visual Geometry Graph (VGG16) deep learning model achieves maximum validation accuracy was 98.05 after 15 epochs completion. The model accuracy does not improve after 10 epochs. The plots of training and validation accuracy and loss against the epochs (figure 6.3) provides the visualization deep learning model.

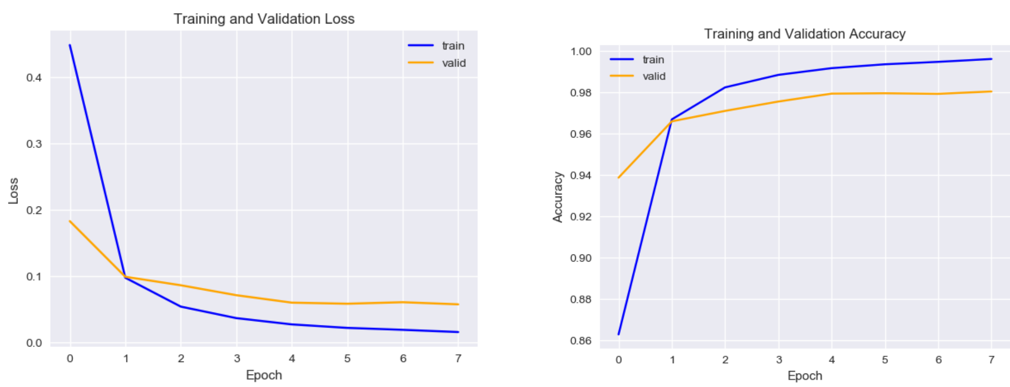

Figure 6.3 VGG16 Model Accuracy & Loss Graph

The last deep learning model is the Resnet50 which had achieved maximum validation accuracy was 98.24 that is better than Alexnet and VGG16 model after 5 epochs (it has 1000 values to train). The average accuracy score was 98% for the Resnet50 model. The plots of training and validation accuracy and loss against the epochs (figure 6.4) provides the visualization deep learning model.

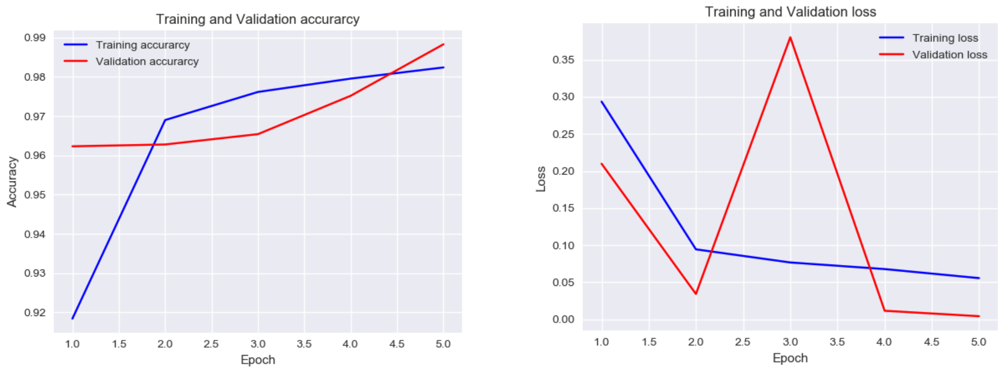

Figure 6.4 Resnet Model Accuracy & Loss Graph

The implementation process requires medium to high hardware requirement for training the four deep learning models which requires high computation resources or use of Graphical processing unit (GPU). The presence of more layers with filters size and train size of image requires high hardware requirement. The four deep learning models (Alexnet, VGG16, Resnet50 and CNN) provide an effective way [32] for solving the problem of plant disease identification system. On the application development side agile methodology was used and so testing was done manually after each button and function was built. It helped in reducing bugs in the python application as they were fixed when found in time. Once this application was ready for prototyping, it was run on friend's laptop to check if its working on their GPU and if it requires higher hardware configuration and it was running fine on other system with lower configuration.

Chapter 7. Conclusions and Future Work

The accurately identification and classification of the plant diseases is very important for the cultivation of crops and this can be achieved using plant disease classification system. This paper discusses how various deep learning models like Alexnet, CNN, VGG16 and Resnet50 can identify the leaves images are healthy or diseased and comparing how each model performed based on accuracy metrics. The training of the models was performed on 87,000 RGB images which were resized according to requirement of the deep learning models. The dataset used in this project comprises of 38 classes of healthy and diseases crop leaves. The most successful model architecture, CNN (Convolution Neural Network) deep learning model achieved a maximum accuracy rate of 98.85% in classification of plant diseases. The second-best model based on accuracy is Resnet50 (with accuracy rate 98.24%) followed by VGG16 (with accuracy rate 98.05%) and Alexnet (with accuracy rate 96.89%). Based on the performance, it has become evident that CNN model is highly suitable for automated plant disease classification system and diagnosis of plant disease through analysis of leaves image. The python GUI application for the project has the user-friendly interface and easy to use. The GUI application requires a good hardware configuration to run smoothly on the system.

The proposed system in this project has good success rate but there are various reasons that make it still quite far from being a generic tool that can be used in real world conditions. The expansion of existing database of plant disease classes would be next future step to process that can be challenging in several aspect and time demanding. Another important issue that can be solved later is the testing dataset used for assessment of the deep learning models that was part of the same dataset that constitute training image. The test images should come from different sources or database that are captured in real situation. The GUI application designed for this project requires good hardware configuration but in the future an application can be developed that can run on low hardware configuration. Later we can develop an android/iOS application for the plant disease identification system.

References

[1] C.H. Bock, G. H. Poole, P. E. Parker, and T. R. Gottwald, “Plant disease severity estimated visually, by digital photography and image analysis, and by hyperspectral imaging,” Critical Reviews in Plant Sciences. [Accessed 02-05-2020]

[2] Barbedo, J.G.A.B. (2016) “A review on the main challenges in automatic plant disease identification based on visible range images” Biosystems Engineering,. [Accessed 02-05-2020]

[3] Nutter, F.W., Jr.; Esker, P.D.; Coelho, R.(2016), Disease assessment concepts and the advancements made in improving the accuracy and precision of plant disease data. Eur. J. Plant Pathol. [Accessed 02-05-2020]

[4] M. Desai, A.K. Jain, N.K. Jain, K. Jethwa (2016), Detection and classification of fruit disease: a review Int Res J Eng Technol. [Accessed 02-05-2020]

[5] Pranjali B. Padol, Prof. AnjilA.Yadav (2016), "SVM Classifier Based Grape Leaf Disease Detection" Conference on Advances in Signal Processing(CAPS) Cummins college of Engineering for Women, Pune. [Accessed 03-05-2020]

[6] Detecting jute plant disease using image processing and machine learning 2016 3rd International Conference on Electrical Engineering and Information Communication Technology (ICEEICT). [Accessed 03-05-2019]

[7] Henry C. Ellis.(1965). The Transfer of Learning. The Macmillan Company, New York. [Accessed 03-05-2020]

[8] Pan, Sinno Jialin, and Qiang Yang (2010). "A Survey on Transfer Learning." IEEE Transactions on Knowledge and Data Engineering. [Accessed 04-05-2020]

[9] Keras, "Keras API," [online]. Available at https://keras.io/api/ [Accessed 04-05-2020]

[10] Tensorflow, “Transfer learning with a pretrained ConvNet,” Tensorflow, [Online]. Available:https://www.tensorflow.org/tutorials/images/transfer_learning [Accessed 04-05-2020]

[11] Krizhevsky, Alex; Sutskever, Ilya; Hinton, Geoffrey E. (2017). "ImageNet classification with deep convolutional neural networks". [Accessed 04-05-2020]

[12] Balas, Valentina E.; Kumar, Raghvendra; Srivastava, Rajshree (2019). Recent Trends and Advances in Artificial Intelligence and Internet of Things. [Accessed 05-05-2020]

[13] Simonyan, K., Zisserman, A. (2015) “Very deep convolutional networks for large-scale image recognition”, International Conference on Learning Representations. [Accessed 05-05-2020]

[14] He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; Sun, Jian (2015-12-10). "Deep Residual Learning for Image Recognition". [Accessed 05-05-2020]

[15] Samir Bhattarai, “New Plant Diseases Dataset” [Online]. Available: https://www.kaggle.com/vipoooool/new-plant-diseases-dataset. [Accessed 05-05-2020]

[16] Dhiman Mondal, Dipak Kumar Kole, Aruna Chakraborty, D. Dutta Majumder (2015)." Detection and Classification Technique of Yellow Vein Mosaic Virus Disease in Okra Leaf Imagesusing Leaf Vein Extraction and Naive Bayesian Classifier., 2015, International Conference on Soft Computing Techniques and Implementations- (ICSCTI) Department of ECE, FET, MRIU, Faridabad, India. [Accessed 05-05-2020]

[17] Sachin D. Khirade, A.B Patil, “Plant Disease Detection Using Image Processing”, International Conference on Computing Communication Control and Automation”,. [Accessed 05-05-2020]

[18] M.Malathi, K.Aruli , S.Mohamed Nizar, A.Sagaya Selvaraj, “A Survey on Plant Leaf Disease Detection Using Image Processing Techniques”,International Research Journal of Engineering and Technology (IRJET). [Accessed 05-05-2020]

[19] Detecting jute plant disease using image processing and machine learning 2016 3rd International Conference on Electrical Engineering and Information Communication Technology (ICEEICT). [Accessed 05-05-2020].

[20] P.Revathi, M.Hemalatha, “Advance Computing Enrichment Evaluation of Cotton Leaf Spot Disease Detection U sing Image Edge detection” [Accessed 05-05-2020].

[21] Ms. Poojapawer, Dr. varshaTukar, prof.parvinpatil "Cucumber Disease detection using artificial neural network". [Accessed 05-05-2020]

[22] Detection of leaf diseases and classification using digital image processing (2017) International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS). [Accessed 05-05-2020]

[23] Heeb Al Bashish, Malik Braik, and Sulieman Bani-Ahmad, “A Framework for Detection and Classification of Plant Leaf and Stem Diseases”. [Accessed 05-05-2020]

[24] Detection and measurement of paddy leaf disease symptoms using image processing. [Accessed 05-05-2020]

[25] Tanvimehera, vinaykumar,pragyagupta (2016). "Maturity and disease detection in tomato using computer vision" Fourth international conference on parallel, distributed and grid computing(PDGC) [Accessed 05-05-2020]

[26] Tejoindhi M.R, Nanjesh B.R, JagadeeshGujanuru Math, AshwinGeetD'sa (2016) "Plant Disease Analysis Using Histogram Matching Based On Bhattacharya's Distance Calculation" International Conference on Electrical, Electroniocs and Optimization Techniques(ICEEOT). [Accessed 05-05-2020]

[27] Python, “Python,” [Online]. Available: https://www.python.org/about/. [Accessed 06-05-2020]

[28] Srdjan Sladojevic, Marko Arsenovic, Andras Anderla, Dubravko Culibrk, and Darko Stefanovic,(2016) Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification, Hindawi Publishing Corporation, Computational Intelligence and Neuroscience. [Accessed 07-05-2020]

[29] Keras, "Keras Data PreProcessing," [online]. Available at https://keras.io/api/preprocessing/ [Accessed 08-05-2020]

[30] Pratical Deep Learning Example with MATLAB (2018), MathWorks. [Accessed 09-05-2020]

[31] Caglayan, A., Guclu, O., & Can, A. B. (2013). “A plant recognition approach using shape and color features in leaf images.” In International Conference on Image Analysis and Processing.[Accessed 09-05-2020]

[32] Sharada P Mohanty, David P Hughes, and Marcel Salath´e (2016). “Using deep learning for image-based plant disease detection”. [Accessed 10-05-2020]