Project Report

on

PLANT DISEASE DETECTION USING CNN

Submitted by

AYUSHMAN SAINI (R164217012)
HIMANSHU TIWARI (R164217021)
SARVAGYA TRIPATHI (R164217050)
YASH SAINI (R110217188)

Under the guidance of

Mr. Vidyanand Mishra
Assistant Professor, Department of Systemics

SCHOOL OF COMPUTER SCIENCE
UNIVERSITY OF PETROLEUM & ENERGY STUDIES
Bidholi Campus, Energy Acres, Dehradun – 248007.
May - 2020

CANDIDATES DECLARATION

I/We hereby certify that the project work entitled Plant Disease Detection using CNN in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science And Engineering and submitted to the Department of Systemics at School of Computer Science, University of Petroleum and Energy Studies, Dehradun, is an authentic record of my/our work carried out during a period from January, 2020 to May, 2020 under the supervision of Mr. Vidyanand Mishra, Assistant Professor, Department of Systemics.

Name	Ayushman Saini	Himanshu Tiwari	Sarvagya Tripathi	Yash Saini
Roll No.	R164217012	R164217021	R164217050	R110217188

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

(Date: 15 May 2020)

Dr. Neelu Jyothi Ahuja	Mr. Vidyanand Mishra
Head, Department of Systemics	Project Mentor
University of Petroleum and Energy Studies	Assistant Professor
Dehradun- 248007 (Uttarakhand)	Department of Systemics

ACKNOWLEDGEMENT

We wish to express our deep gratitude to our guide Mr. Vidyanand Mishra, Assistant Professor for all advice, encouragement and constant support he has given us throughout our project work. This work would not have been possible without his support and valuable suggestions.

We sincerely thank to our Head of the Department, Dr. Neelu Jyothi Ahuja, for her great support in doing our project Plant Disease Detection Using CNN at SCS.

We are also grateful to Dr. Manish Prateek, Professor and Dean SCS, UPES for giving us the necessary facilities to carry out our project work successfully.

We would like to thank all our friends for their help and constructive criticism during our project work. Finally, we have no words to express our sincere gratitude to our parents who have shown us this world and for every support they have given us.

Name	Ayushman Saini	Himanshu Tiwari	Sarvagya Tripathi	Yash Saini
Roll No.	R164217012	R164217021	R164217050	R110217188

ABSTRACT

Crop diseases, if can be detected in earlier stages can be very beneficial to our farmers. Taking wheat crop into consideration, diseases like yellow rust can affect the plant drastically, reducing the seed count. There are a few different ways to recognize plant pathology. CNN (Convolutional Neural Network) can be used to identify those symptoms in plants to detect disease in earlier stages. CNN is utilized for object acknowledgment and picture classification. Dataset of images of wheat plant is taken. This dataset is prepared by us and not from any other source. It characterizes crop diseases from leaf pictures taken under uncontrolled conditions. Then the testing pictures are characterized as ill or healthy.

Keywords- crop, convolutional, classification, disease, detection

TABLE OF CONTENTS

Contents

INTRODUCTION

Agribusiness has gotten considerably more than essentially a way to take care of consistently developing populaces. However, crop diseases are compromising the work of this significant source. Crop diseases cause significant creation and financial misfortunes in horticulture and ranger service. For instance, wheat **yellow rust** and **stem rust**, it is a fungal disease that assaults the leaves of wheat crop by framing yellow stripes and influence their photosynthesis that causes the withering of the additional size. About 31 nations of east and north Africa, the Near East, Central and South Asia, representing over 37 percent of worldwide wheat creation territory and 30% of creation, are in danger of wheat rust diseases (UN-FAO). In this way, early recognition and identification of crop diseases assume the most extreme significant job to take convenient measures. There are a few different ways to recognize plant pathology. A few diseases don't have any noticeable side effects related, or those show up just when it is past the point where it is possible to act. In these cases, it is important to perform the complex examination, for the most part by methods for the ground-breaking magnifying lens. Now and again, the signs must be distinguished in parts of the electromagnetic range that are not unmistakable to people.

Most diseases, however, create a sign in the unmistakable range. The diseases may show side effects on various pieces of the crop, for example, leaves, stem, seeds and so forth. This examination centers around the identification and classification of wheat crop diseases depending on the side effects of the diseases that give indications on the leaves and stem of the crop. As a rule, the determination, or if nothing else a first surmise about the disease, is performed outwardly by people. Prepared specialists might be efficient in perceiving the disease. Sadly, more often than not there are no specialists in the territory to give an information-based examination and encourage the ranchers. Thusly; searching for a quick, programmed, more affordable and precise technique to distinguish plant diseases is critical. Since the late 1970s, PC based picture handling innovation applied in the agrarian designing exploration has become typical practice.

Recently **convolutional neural networks** (CNN) have been utilized for object acknowledgment and picture classification. A convolutional neural network is a kind of deep neural network (DNN) roused by the human visual system, utilized for preparing pictures. Different CNN models were proposed to be utilized for object acknowledgment.

In this project, the attainability of CNN to characterize crop diseases from leaf pictures taken under uncontrolled conditions has been considered. The models are structured dependent on the LeNet design. The dataset utilized for preparing is assembled from various wheat crop fields in various stages. Since CNN requires a huge measure of information, information growth is utilized to expand the preparation information.

MOTIVATION

India is an agrarian country and its economy largely based upon crop production. Agriculture is the backbone of every economy. The agriculture sector needs a huge up-gradation in order to survive the changing conditions of Indian economy. For optimum yield, the crop should be healthy, therefore some highly technical method is needed for periodic monitoring of crop. Crop disease is one of the major factor which indirectly influence the significant reduction of both quality and quantity of agricultural products. The presence of disease on the plant is mainly reflected by symptoms on leaves. So there is a need of an automatic, accurate and less expensive Machine Vision System for detection of diseases from the image.

LITERATURE REVIEW

Literature analysis will provide numerous methods to tackle our problem. Right now, late patterns in utilizing CNN and deep learning structures in the horticultural application are talked about. Before the appearance of deep learning, picture handling and AI methods have been utilized to arrange distinctive crop diseases.

Different CNN structures were proposed to be utilized for object acknowledgment for instance. LeNet, AlexNet, GoogLeNet and so on.

LeNet engineering is the first CNN presented by [1]LeCun et al. to perceive manually written digits (LeCun et al. 1998). It comprises of two convolutional layers and two subsampling layers followed by a completely associated MLP.

Barely any analysts proposed the utilization of CNN for leaf acknowledgment and harvest illness classification. [2]Atabay planned a convolutional neural system design to recognize plants dependent on leaf pictures. The proposed design comprises five layers. After each convolutional layer a Rectified Linear Unit (ReLU) or Exponential Linear Unit (ELU) actuation work is utilized and for each pooling layer, the MaxPooling approach is applied. The proposed framework is applied to Flavia and Swedish leaf datasets containing 32 plant species with 1907 examples and 15 species with 1125 examples separately. The pictures in the dataset are photos of a solitary leaf taken at the uniform foundation. All the information pictures are 160x160 pixel grayscale pictures. The model accomplished a classification exactness of 97.24% and 99.11% precision for each dataset. The outcomes indicated that the proposed engineering for CNN-based leaf classification is intently contending with the most recent broad methodologies on formulating leaf highlights and classifiers.

Angie K. Reyes et al. [3](Reyes, Caicedo, and Camargo 2015), utilized a deep learning approach where the total framework was found out without hand-built segments. The structured framework has 5 Conv layers followed by 2 completely associated layers. CNN is prepared to utilize 1.8 million pictures from ILSVRC 2012 dataset 1 and utilized a finetuning system to move took in acknowledgment abilities from general areas to the specific challenge of crop Identification task. The dataset is blending of pictures of a plant or part of a plant taken both under a controlled situation just as in the regular habitat. They acquired a normal exactness of 0.486.

Mrinal Kumar, Tathagata Hazra, Dr. Sanjaya Shankar Tripathy in their paper, [4] Wheat Leaf Disease Detection Using Image Processing, is done by using important tools like k-mean clustering, GLCM and PNN. Some of the challenges that they faced was to detect plant disease as soon as possible.

PROBLEM STATEMENT

The system should identify the diseases plants from the given test pictures using CNN. The system should be trained on dataset and classify the images with high accuracy. The applied algorithm should result in high percentage. So that the diseases in the crops can be identified in earlier stages. Thus increasing the final production of crops.

OBJECTIVE

To detect the diseases of the wheat crop at an early stage, overcome it and take the best measures to prevent the crop get further damaged. We will be using our dataset, pictures of wheat plant, that will be used along with CNN for the detection of diseases.

PROPOSED METHOD

CNN is proposed method in this project. CNNs are multi-layer directed networks that can gain includes naturally from datasets. Throughout the previous barely any years, CNNs have accomplished best in class execution in practically terrifically significant classification assignments. It can perform both component extraction and classification under a similar design. A CNN [Figure 1] is a unique sort of neural network that has been broadly applied to an assortment of example acknowledgment issues, for example, PC vision, discourse acknowledgment, and so forth. A convolutional neural network (CNN) is a kind of deep neural network (DNN) motivated by the human visual framework, utilized for handling pictures. A CNN has a few phases, each normally made out of two layers: the principal layer does a convolution of the info picture witha channel and the subsequent layer down-examples the consequence of the main layer, utilizing a pooling activity. These stages assemble progressively unique portrayals of the information design: the main may be delicate to edges, the second to corners and convergences, etc. The thought is that these portrayals become both increasingly dynamic and progressively invariant as the example information experiences the CNN. The yield of the last stage is typically a vector (not a picture) that is taken care of to a multi-layer perception (MLP) that delivers the last network yield, generally a class label.

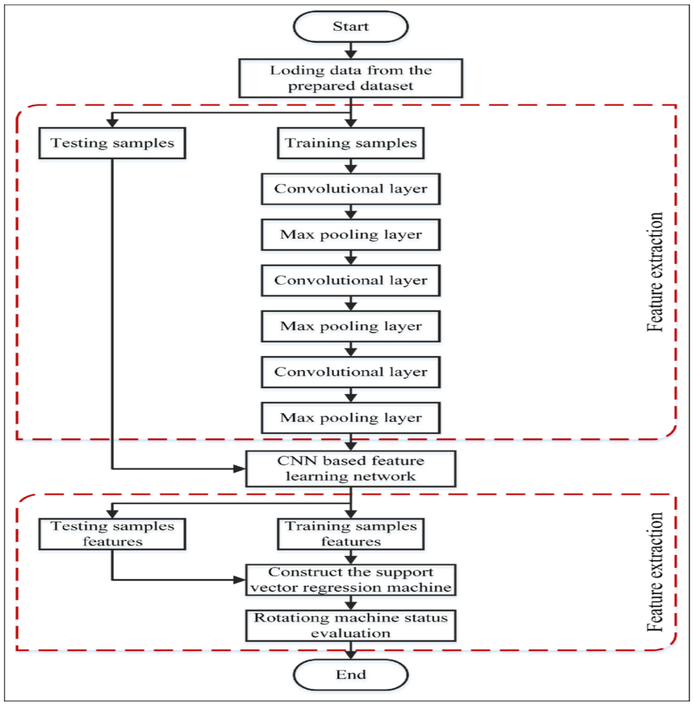


Figure 1 General Flow Chart of CNN

We are using Deeplearning4j library for this project. Eclipse **Deeplearning4j** is a deep learning programming library written for Java and the Java virtual machine (JVM) and a computing framework with wide support for deep learning algorithms. Deeplearning4j includes implementations of the restricted Boltzmann machine, deep belief net, deep auto encoder, stacked denoising autoencoder and recursive neural tensor network, word2vec, doc2vec, and GloVe. These algorithms all include distributed parallel versions that integrate with Apache Hadoop and Spark.

DESIGN APPROACH

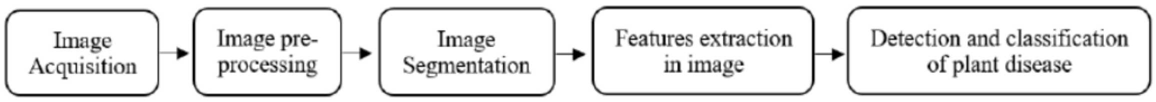


Image Acquisition
Image pre-processing
Image Segmentation
Features extraction in image
Detection and classification of plant disease

Figure 2 Method of Image Processing

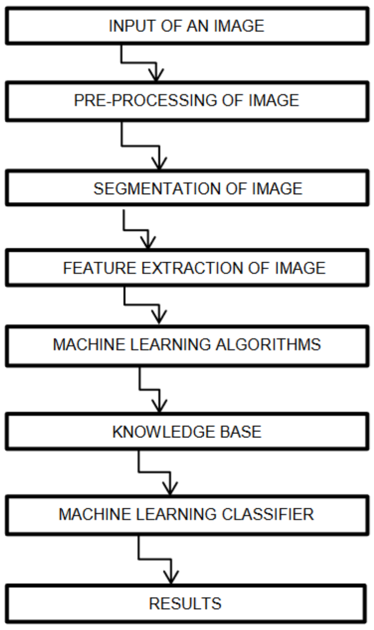


Figure 3 Proposed Method

METHODOLOGY

Initial step for any image processing based project is acquiring proper database which is valid. Most of the time the standard database is preferred but in certain circumstances we do not get proper database .So in such conditions we can collect the images and can form our own database. The database is collected by us using our smart phones. Images were captured of a wheat plant in a field(uncontrolled environment). Data was not labeled .So the first task is to clean and label the database. There is a huge database so basically the images with better resolution and angle are selected. After selection of images we should have knowledge about the infected leaves and the healthy leaves.

Image Acquisition:[Figure 2] First we need to select the plant which is affected by the disease and then collect the leaf of the plant and take a snapshot of leaf and load the leaf image into the system.

Segmentation: It means representation of the image in more meaningful and easy to analyze way. In segmentation a digital image is partitioned into multiple segments can defined as super-pixels. Low Contrast: image pixel values are concentrated near a narrow range.

Converting RGB to HSI: The RGB image is in the size of M-by-N-by-3,where the three dimensions account for three image planes(red, green, blue).if all the three components are equal then conversion is undefined. Generally the pixel range of RGB is [0,255] in his the pixel range is [0, 1].Conversion of pixel range can be done by calculating of the components; Hue, Saturation, Intensity. Then combine the three results into one single value then the HIS image is formed.

Significant highlights are removed from the picture and utilized as a contribution to the classifier. The general classification exactness is along these lines subject to the kind of picture handling and highlight extraction systems utilized.

A dataset consisting of about 500 different images of wheat plant is obtained, out of which any image can be used as a test image for the software. The train dataset is used to train the model (CNN) so that it can identify the test image and the disease it has. CNN has different layers that are Dense, Dropout, Activation, Flatten, Convolution2D, and MaxPooling2D. After the model is trained successfully, the software can identify the disease if the plant species is contained in the database. After successful training and preprocessing, comparison of the test image and trained model takes place to predict the disease.

We made two Codes in this project. One is for training the network using dataset and testing the network using the train dataset. Thereby, calculating the accuracy of the network. Second one is for identifying the single image using the trained network and classify it as ‘Healthy’ or ‘Infected’

SYSTEM REQUIREMENTS (Software/Hardware)

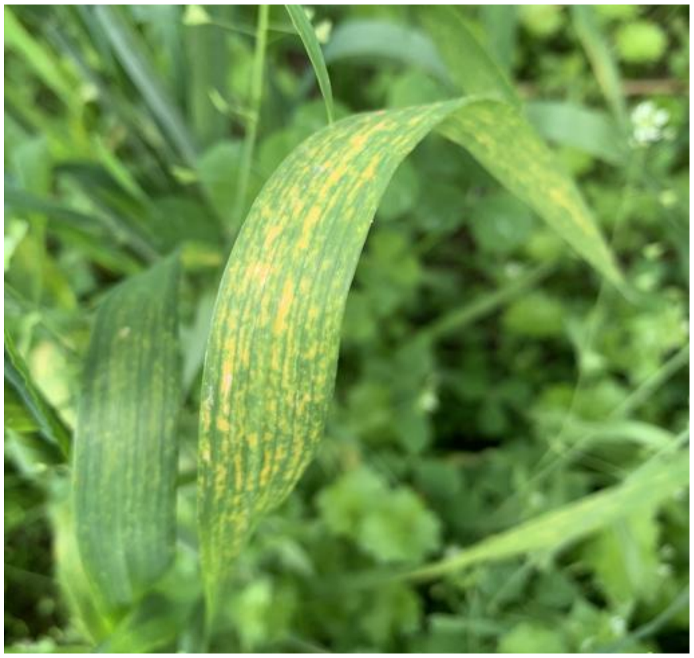
1. Software requirement
    - Language: Java
    - IDE: Eclipse
    - Operating System: Windows10

2. Hardware requirement
    - Random Access Memory(RAM): 8GB and above
    - Processor: Intel Core i5-6100 CPU
    - Processor Speed: 2.60GHz

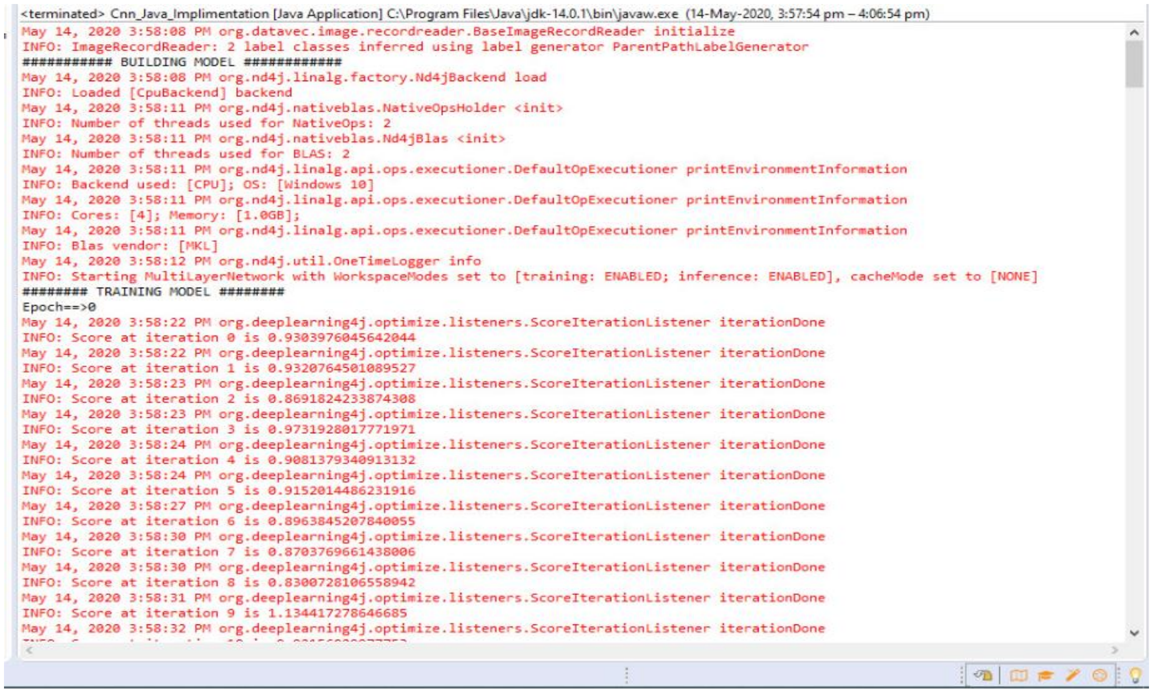
SAMPLE IMAGES OF DATASET

HEALTHY PLANT IMAGE:

INFECTED PLANT IMAGE:

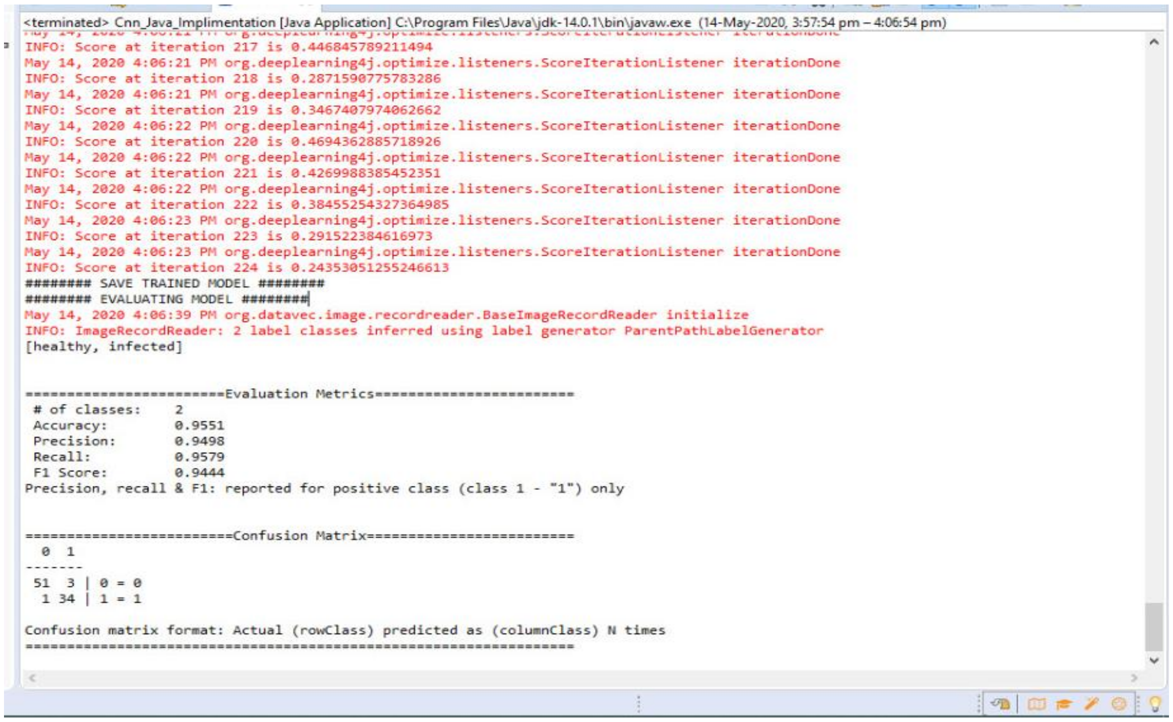


OUTPUT SNAPSHOTS



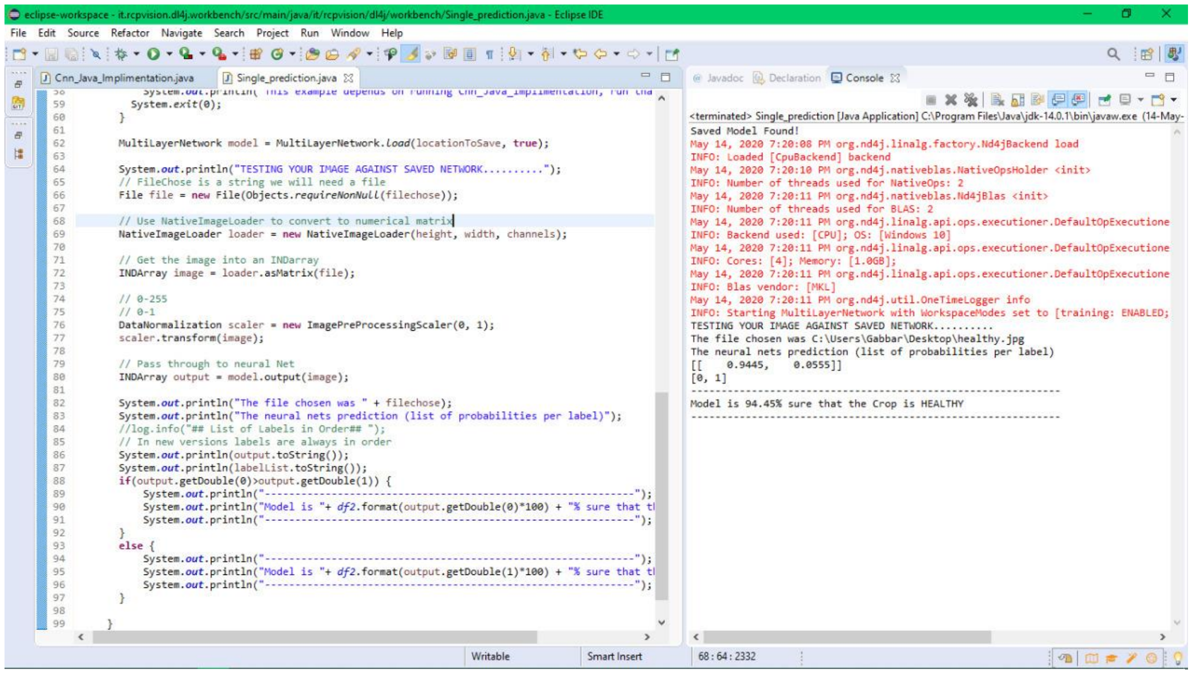
```
<terminated> Cnn_Java_Implimentation [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe  (14-May-2020, 3:57:54 pm – 4:06:54 pm)
May 14, 2020 3:58:08 PM org.datavec.image.recordreader.BaseImageRecordReader initialize
INFO: ImageRecordReader: 2 label classes inferred using label generator ParentPathLabelGenerator
########### BUILDING MODEL ############
May 14, 2020 3:58:08 PM org.nd4j.linalg.factory.Nd4jBackend load
INFO: Loaded [CpuBackend] backend
May 14, 2020 3:58:11 PM org.nd4j.nativeblas.NativeOpsHolder <init>
INFO: Number of threads used for NativeOps: 2
May 14, 2020 3:58:11 PM org.nd4j.nativeblas.Nd4jBlas <init>
INFO: Number of threads used for BLAS: 2
May 14, 2020 3:58:11 PM org.nd4j.linalg.api.ops.executioner.DefaultOpExecutioner printEnvironmentInformation
INFO: Backend used: [CPU]; OS: [Windows 10]
May 14, 2020 3:58:11 PM org.nd4j.linalg.api.ops.executioner.DefaultOpExecutioner printEnvironmentInformation
INFO: Cores: [4]; Memory: [1.0GB];
May 14, 2020 3:58:11 PM org.nd4j.linalg.api.ops.executioner.DefaultOpExecutioner printEnvironmentInformation
INFO: Blas vendor: [MKL]
May 14, 2020 3:58:12 PM org.nd4j.util.OneTimeLogger info
INFO: Starting MultiLayerNetwork with WorkspaceModes set to [training: ENABLED; inference: ENABLED], cacheMode set to [NONE]
######## TRAINING MODEL ########
Epoch==>0
May 14, 2020 3:58:22 PM org.deeplearning4j.optimize.listeners.ScoreIterationListener iterationDone
INFO: Score at iteration 0 is 0.9303976045642044
May 14, 2020 3:58:22 PM org.deeplearning4j.optimize.listeners.ScoreIterationListener iterationDone
INFO: Score at iteration 1 is 0.9320764501089527
May 14, 2020 3:58:23 PM org.deeplearning4j.optimize.listeners.ScoreIterationListener iterationDone
INFO: Score at iteration 2 is 0.8691824233874308
May 14, 2020 3:58:23 PM org.deeplearning4j.optimize.listeners.ScoreIterationListener iterationDone
INFO: Score at iteration 3 is 0.9731928017771971
May 14, 2020 3:58:24 PM org.deeplearning4j.optimize.listeners.ScoreIterationListener iterationDone
INFO: Score at iteration 4 is 0.9081379340913132
May 14, 2020 3:58:24 PM org.deeplearning4j.optimize.listeners.ScoreIterationListener iterationDone
INFO: Score at iteration 5 is 0.9152014486231916
May 14, 2020 3:58:27 PM org.deeplearning4j.optimize.listeners.ScoreIterationListener iterationDone
INFO: Score at iteration 6 is 0.8963845207840055
May 14, 2020 3:58:30 PM org.deeplearning4j.optimize.listeners.ScoreIterationListener iterationDone
INFO: Score at iteration 7 is 0.8703769661438006
May 14, 2020 3:58:30 PM org.deeplearning4j.optimize.listeners.ScoreIterationListener iterationDone
INFO: Score at iteration 8 is 0.8300728106558942
May 14, 2020 3:58:31 PM org.deeplearning4j.optimize.listeners.ScoreIterationListener iterationDone
INFO: Score at iteration 9 is 1.134417278646685
May 14, 2020 3:58:32 PM org.deeplearning4j.optimize.listeners.ScoreIterationListener iterationDone
```

Figure 4 Showing Accuracy at each epoch at runtime



```
<terminated> Cnn_Java_Implimentation [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe  (14-May-2020, 3:57:54 pm – 4:06:54 pm)
INFO: Score at iteration 217 is 0.446845789211494
May 14, 2020 4:06:21 PM org.deeplearning4j.optimize.listeners.ScoreIterationListener iterationDone
INFO: Score at iteration 218 is 0.2871590775783286
May 14, 2020 4:06:21 PM org.deeplearning4j.optimize.listeners.ScoreIterationListener iterationDone
INFO: Score at iteration 219 is 0.3467407974062662
May 14, 2020 4:06:22 PM org.deeplearning4j.optimize.listeners.ScoreIterationListener iterationDone
INFO: Score at iteration 220 is 0.4694362885718926
May 14, 2020 4:06:22 PM org.deeplearning4j.optimize.listeners.ScoreIterationListener iterationDone
INFO: Score at iteration 221 is 0.4269988385452351
May 14, 2020 4:06:22 PM org.deeplearning4j.optimize.listeners.ScoreIterationListener iterationDone
INFO: Score at iteration 222 is 0.38455254327364985
May 14, 2020 4:06:23 PM org.deeplearning4j.optimize.listeners.ScoreIterationListener iterationDone
INFO: Score at iteration 223 is 0.291522384616973
May 14, 2020 4:06:23 PM org.deeplearning4j.optimize.listeners.ScoreIterationListener iterationDone
INFO: Score at iteration 224 is 0.24353051255246613
######## SAVE TRAINED MODEL ########
######## EVALUATING MODEL ########
May 14, 2020 4:06:39 PM org.datavec.image.recordreader.BaseImageRecordReader initialize
INFO: ImageRecordReader: 2 label classes inferred using label generator ParentPathLabelGenerator
[healthy, infected]


========================Evaluation Metrics========================
 # of classes:    2
 Accuracy:        0.9551
 Precision:       0.9498
 Recall:          0.9579
 F1 Score:        0.9444
Precision, recall & F1: reported for positive class (class 1 - "1") only


=========================Confusion Matrix=========================
  0  1
-------
 51  3 | 0 = 0
  1 34 | 1 = 1

Confusion matrix format: Actual (rowClass) predicted as (columnClass) N times
==================================================================
```

Figure 5 Showing the accuracy of test dataset

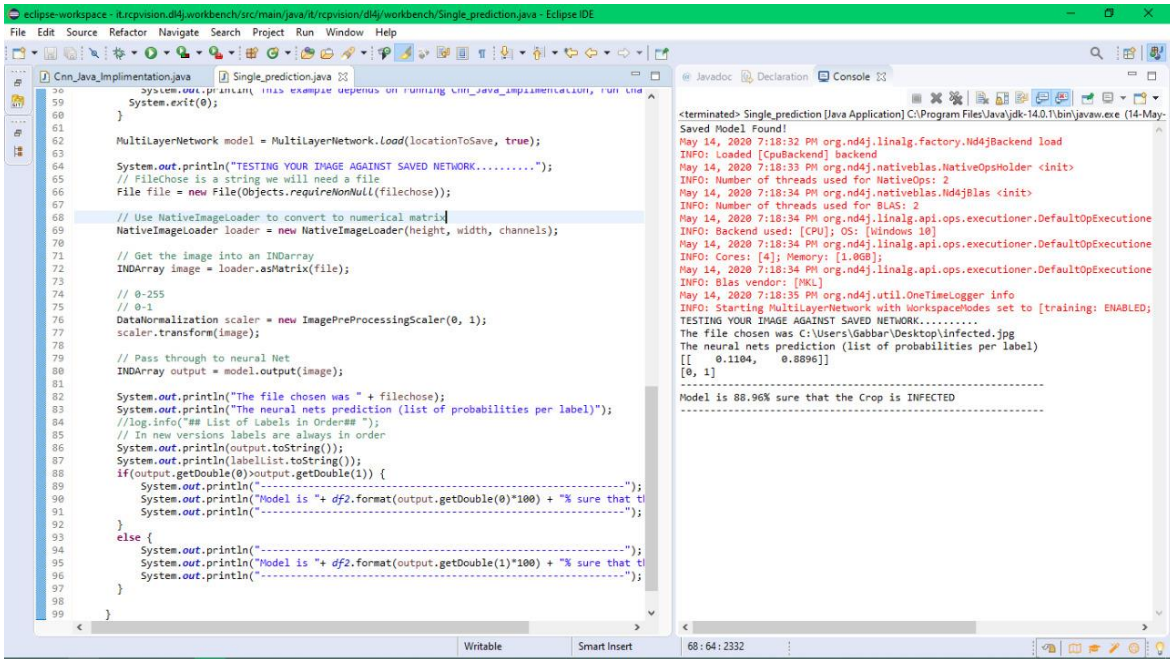

eclipse-workspace - it.rcpvision.dl4j.workbench/src/main/java/it/rcpvision/dl4j/workbench/Single_prediction.java - Eclipse IDE

```java
            System.out.println("This example depends on running Cnn_Java_Implimentation, run tha
          System.exit(0);
        }

        MultiLayerNetwork model = MultiLayerNetwork.load(locationToSave, true);

        System.out.println("TESTING YOUR IMAGE AGAINST SAVED NETWORK..........");
        // FileChose is a string we will need a file
        File file = new File(Objects.requireNonNull(filechose));

        // Use NativeImageLoader to convert to numerical matrix
        NativeImageLoader loader = new NativeImageLoader(height, width, channels);

        // Get the image into an INDarray
        INDArray image = loader.asMatrix(file);

        // 0-255
        // 0-1
        DataNormalization scaler = new ImagePreProcessingScaler(0, 1);
        scaler.transform(image);

        // Pass through to neural Net
        INDArray output = model.output(image);

        System.out.println("The file chosen was " + filechose);
        System.out.println("The neural nets prediction (list of probabilities per label)");
        //log.info("## List of Labels in Order## ");
        // In new versions labels are always in order
        System.out.println(output.toString());
        System.out.println(labelList.toString());
        if(output.getDouble(0)>output.getDouble(1)) {
            System.out.println("-------------------------------------------------------------");
            System.out.println("Model is "+ df2.format(output.getDouble(0)*100) + "% sure that t
            System.out.println("-------------------------------------------------------------");
        }
        else {
            System.out.println("-------------------------------------------------------------");
            System.out.println("Model is "+ df2.format(output.getDouble(1)*100) + "% sure that t
            System.out.println("-------------------------------------------------------------");
        }

    }
```

```
<terminated> Single_prediction [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe (14-May-
Saved Model Found!
May 14, 2020 7:20:08 PM org.nd4j.linalg.factory.Nd4jBackend load
INFO: Loaded [CpuBackend] backend
May 14, 2020 7:20:10 PM org.nd4j.nativeblas.NativeOpsHolder <init>
INFO: Number of threads used for NativeOps: 2
May 14, 2020 7:20:11 PM org.nd4j.nativeblas.Nd4jBlas <init>
INFO: Number of threads used for BLAS: 2
May 14, 2020 7:20:11 PM org.nd4j.linalg.api.ops.executioner.DefaultOpExecutione
INFO: Backend used: [CPU]; OS: [Windows 10]
May 14, 2020 7:20:11 PM org.nd4j.linalg.api.ops.executioner.DefaultOpExecutione
INFO: Cores: [4]; Memory: [1.0GB];
May 14, 2020 7:20:11 PM org.nd4j.linalg.api.ops.executioner.DefaultOpExecutione
INFO: Blas vendor: [MKL]
May 14, 2020 7:20:11 PM org.nd4j.util.OneTimeLogger info
INFO: Starting MultiLayerNetwork with WorkspaceModes set to [training: ENABLED;
TESTING YOUR IMAGE AGAINST SAVED NETWORK..........
The file chosen was C:\Users\Gabbar\Desktop\healthy.jpg
The neural nets prediction (list of probabilities per label)
[[    0.9445,    0.0555]]
[0, 1]
-------------------------------------------------------------
Model is 94.45% sure that the Crop is HEALTHY
-------------------------------------------------------------
```

Writable | Smart Insert | 68 : 64 : 2332

Figure 6 Classifying single image as Healthy


eclipse-workspace - it.rcpvision.dl4j.workbench/src/main/java/it/rcpvision/dl4j/workbench/Single_prediction.java - Eclipse IDE

```java
            System.out.println("This example depends on running Cnn_Java_Implimentation, run tha
          System.exit(0);
        }

        MultiLayerNetwork model = MultiLayerNetwork.load(locationToSave, true);

        System.out.println("TESTING YOUR IMAGE AGAINST SAVED NETWORK..........");
        // FileChose is a string we will need a file
        File file = new File(Objects.requireNonNull(filechose));

        // Use NativeImageLoader to convert to numerical matrix
        NativeImageLoader loader = new NativeImageLoader(height, width, channels);

        // Get the image into an INDarray
        INDArray image = loader.asMatrix(file);

        // 0-255
        // 0-1
        DataNormalization scaler = new ImagePreProcessingScaler(0, 1);
        scaler.transform(image);

        // Pass through to neural Net
        INDArray output = model.output(image);

        System.out.println("The file chosen was " + filechose);
        System.out.println("The neural nets prediction (list of probabilities per label)");
        //log.info("## List of Labels in Order## ");
        // In new versions labels are always in order
        System.out.println(output.toString());
        System.out.println(labelList.toString());
        if(output.getDouble(0)>output.getDouble(1)) {
            System.out.println("-------------------------------------------------------------");
            System.out.println("Model is "+ df2.format(output.getDouble(0)*100) + "% sure that t
            System.out.println("-------------------------------------------------------------");
        }
        else {
            System.out.println("-------------------------------------------------------------");
            System.out.println("Model is "+ df2.format(output.getDouble(1)*100) + "% sure that t
            System.out.println("-------------------------------------------------------------");
        }

    }
```

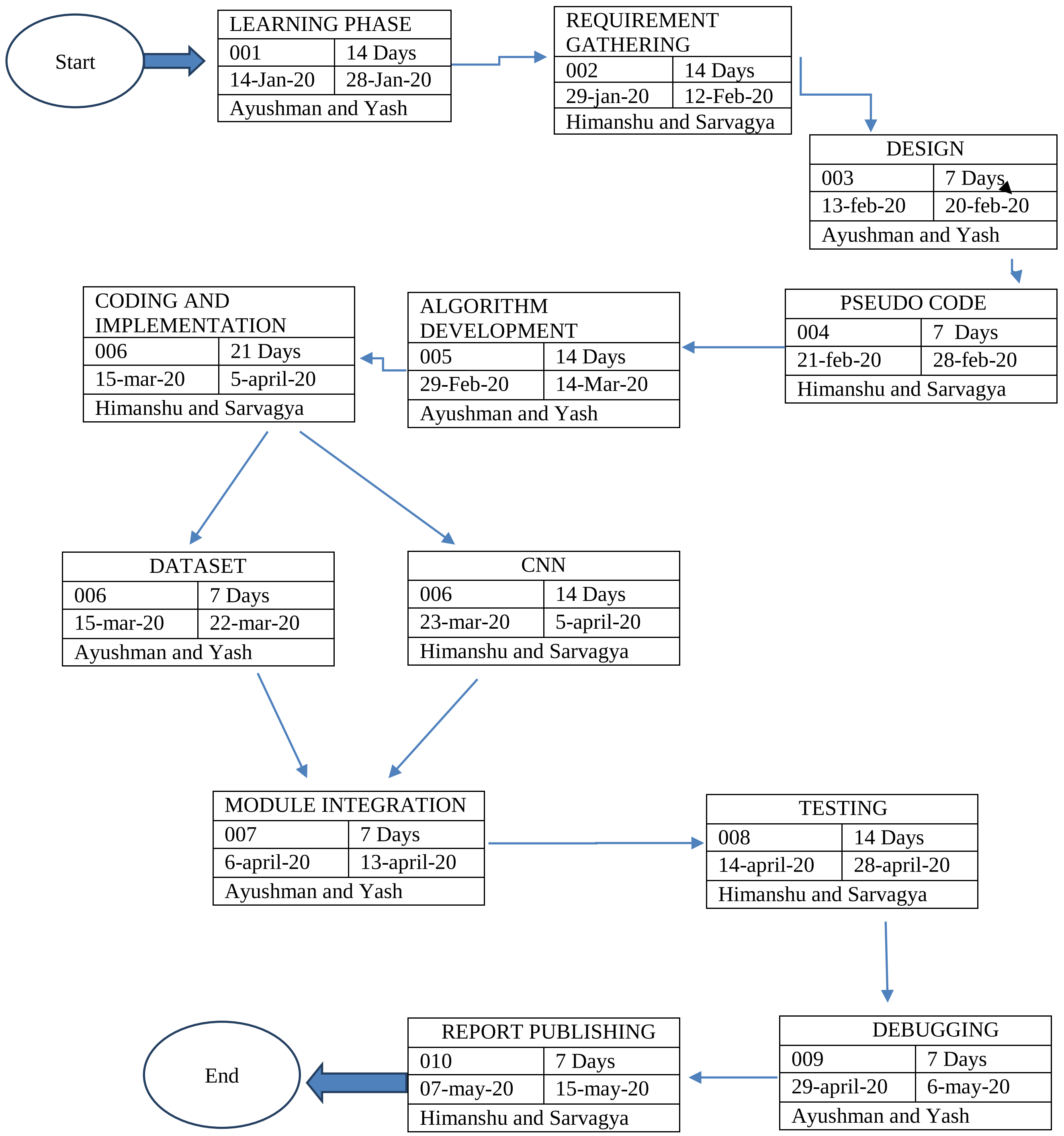
```
<terminated> Single_prediction [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe (14-May-
Saved Model Found!
May 14, 2020 7:18:32 PM org.nd4j.linalg.factory.Nd4jBackend load
INFO: Loaded [CpuBackend] backend
May 14, 2020 7:18:33 PM org.nd4j.nativeblas.NativeOpsHolder <init>
INFO: Number of threads used for NativeOps: 2
May 14, 2020 7:18:34 PM org.nd4j.nativeblas.Nd4jBlas <init>
INFO: Number of threads used for BLAS: 2
May 14, 2020 7:18:34 PM org.nd4j.linalg.api.ops.executioner.DefaultOpExecutione
INFO: Backend used: [CPU]; OS: [Windows 10]
May 14, 2020 7:18:34 PM org.nd4j.linalg.api.ops.executioner.DefaultOpExecutione
INFO: Cores: [4]; Memory: [1.0GB];
May 14, 2020 7:18:34 PM org.nd4j.linalg.api.ops.executioner.DefaultOpExecutione
INFO: Blas vendor: [MKL]
May 14, 2020 7:18:35 PM org.nd4j.util.OneTimeLogger info
INFO: Starting MultiLayerNetwork with WorkspaceModes set to [training: ENABLED;
TESTING YOUR IMAGE AGAINST SAVED NETWORK..........
The file chosen was C:\Users\Gabbar\Desktop\infected.jpg
The neural nets prediction (list of probabilities per label)
[[    0.1104,    0.8896]]
[0, 1]
-------------------------------------------------------------
Model is 88.96% sure that the Crop is INFECTED
-------------------------------------------------------------
```

Writable | Smart Insert | 68 : 64 : 2332

Figure 7 Classifying single image as infected

PERT CHART

Pert Chart for the project will be as follows.



REFERENCES

[1] http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf

[2] https://www.iioab.org/articles/IIOABJ_7.S5_332-336.pdf

[3] https://www.semanticscholar.org/paper/Fine-tuning-Deep-Convolutional-Networks-for-Plant-Reyes-Caicedo/f59d8504d7c6e209e6f8bcb62346140214b244b7

[4] https://www.academia.edu/32856043/Wheat_Leaf_Disease_Detection_Using_Image_Processing

[5] https://towardsdatascience.com/wtf-is-image-classification-8e78a8235acb