COLLEGE OF SCIENCE AND TECHNOLOGY

Website: www.aceiot.ur.ac.rw
Mail: aceiot@ur.ac.rw

COLLEGE OF SCIENCE AND TECHNOLOGY
MASTERS OF SCIENCE IN INTERNET OF THINGS
EMBEDDED COMPUTING SYSTEMS

Research Thesis Title:

AI-enabled IoT mobile application for early maize plant disease detection

Submitted By

MITSINDO Rene (REF.NO: 220014161)

Under supervision of

Dr. Emmanuel Masabo

Dr. Damien Hanyurwimfura

A dissertation submitted in partial fulfillment of the requirements for the degree Master of Science in the Internet of Things-Embedded Computing System (ACEIoT)

March, 2022

Bonafide Certificate

Certified that this thesis titled **AI-enabled IoT mobile application for early maize plant disease detection** is the bonafide work of **MITSINDO Rene (REF NO: 220014161)**, MSc. IoT-ECS student at University of Rwanda/ College of Science and Technology/ African center of Excellence in the Internet of Things. Certified further, that according to the best of my knowledge; the work reported here doesn't form a part of any other research work academic year 2019-2021.

SUPERVISOR:

Dr. EMMANUEL MASABO

Signature:

Date:

CO-SUPERVISOR:

Dr. DAMIEN HANYURWIMFURA

Signature:

Date:

Head of Masters and Trainings ACEIoT:

Dr. James RWIGEMA

Signature:

Date:

Declaration

I, **Mr. MITSINDO Rene**, hereby declare that this research thesis report entitled **" AI-enabled IoT mobile application for early maize plant disease detection"** has not been previously submitted and approved for the award of degree by this or any other academic institution. To the best of my knowledge and belief, the thesis contains no material previously published or written by another researcher except where due reference is made in the thesis itself.

Name: **MITSINDO Rene**

REG.NO: **220014161**

Signature:

Date: **15/03/2022**

Acknowledgment

First and foremost, I would like to thank almighty GOD for keeping me alive and for showering His blessings on me throughout my study work. I would like to express my heartfelt gratitude to my research supervisors, Dr. Damien HANYURWINFURA and Dr. Emmanuel MASABO for their tireless support and important guidance throughout this research. Their enthusiasm, ideas, sincerity, and determination have left a lasting impression on me. They showed me how to conduct research and present my findings most clearly and concisely possible. I'm appreciative of everything they've done for me.

I am extending my heartfelt thanks to all office staff, lectures of the African Center of Excellence in the Internet of Things, and my beloved classmates for their helpful guidance and unlimited support for accomplishing this undertaking.

Finally, I deeply express my thanks to my parents, my sisters, and my caring and loving wife for their support, prayers, and encouragement during my studies.

May the Almighty God bless you all abundantly!

Abstract

Maize crop has become significant food security and income-generating crop for small-scale farmers in Rwanda. Unfortunately, maize farmers are still experiencing significantly lower yields due to several diseases. Diseases affect the quality of maize crops and reduce the efficiency of agriculture production resulting in a significant loss to the farmers.

Maize plant health conditions play a vital role in earning good profit to farmers; moreover, plant health conditions should be monitored at different stages of plants growth for early treatment of plant diseases. Currently, the techniques used by Rwandan farmers to diagnose maize diseases depend on naked eyes observation requires being well trained and experienced, as some plant diseases are very hard to be recognized. Another technique is sending samples to the lab for testing; this is expensive and time-consuming. To overcome limitations presented by these techniques IoT and AI technologies are great imperative technologies for making farming more efficient; these technologies can mitigate measures to help farmers avoid losses and ensure good food security in the different sides of the country.

In this thesis, for early maize plant disease detection, an AI-enabled IoT mobile application was prosed to help farmers automatically detect maize plant diseases at an early stage of plant growth. For detecting plant disease, plant image is captured through the camera and uploaded to the local server using an android application, the plant image undergoes various image processing algorithms at the server for determining the disease, and detected disease is sent back to the farmer's mobile application with remedies.

Various performance indicators, such as classification accuracy and processing time, were used to evaluate our system. The model has an overall classification accuracy of 80% when it comes to distinguishing the three most common disease groups that damage maize leaves.

Keywords: IoT, AI, mobile application, maize plant diseases.

List of Symbols and Abbreviations

ACEIoT: African Center of Excellence in the Internet of Things

AI: Artificial Intelligence

ANN: Artificial Neural Network

CNN: Convolutional Neural Network

IoT: Internet of Things

KNN: K-Nearest-Neighbor algorithm

MINICOM: Ministry of Trade and Industry

Rpi: Raspberry pi

SVR: Support Vector Regression

SVM: Support Vector Machine

TCP: Transmission Control Protocol

List of Figures

List of tables

Table of Contents

CHAPTER 1: GENERAL INTRODUCTION

1.1 Background and Motivation

The agricultural sector is one of the most important sectors of the Rwandan economy[1]. The sector is considered to be the engine of Rwanda's economic growth and the majority of jobs in the country depend on agriculture[2]. About 75% of the population is directly dependent on this sector[3]. Among the most products cultivated in Rwanda, For small-scale farmers in the country, maize has become major food security and income-generating crop[4]. Approximately, half of the maize in Rwanda is produced in the East Province and about 60 percent of the maize is sold through manufacturing industries in Rwanda[5]. MINICOM report (2014) indicated that informal trade accounted for at least 80% of the maize traded in the country.

Recently, the number of maize diseases increased, mainly due to the degradation of agricultural land, climate changes, which in turn affect crop productivity[6]. Plants are affected not just by a lack of nutrients, but also by microbes such as fungi, bacteria, viruses, and mites. [7], [8]. These kinds of borne diseases are quite harmful as they affect huge farms. And so it is very important to take precautions to ensure that crop yields are maintained. Among the various diseases that affect maize plantations, leaf diseases are one of the most critical and cause scaling down the crop yield and food nutrition value[9] so detection of plant diseases can easily be achieved through leaves as they are a prominent and sensitive part of the plant.

At present, the identification of plant diseases mostly depends on naked eyes for small scale farmers and This can lead to issues, such as farmers misidentifying a disease by judging it from their past experience resulting in the use of improper pesticides so experts and pathologists are needed to help farmers for plant diseases identification and their remedies, however, due to the shortage of experts they can't reach all farmers in real-time [10]. If the diseases are detected promptly, appropriate remedies can be applied and crop loss can be prevented[11]. For maize diseases detection, regular inspection of cornfields is required.

In Comparison to human vision, computer image processing is a step forward since it can identify features like speediness large amount of data and distinguish even small diversity which human vision cannot[12], [13]. As a result, image processing can assist farmers in precisely identifying the condition and recommending the appropriate treatment.

Around the world, different AI algorithms were used by previous researchers for plant diseases detection such as a K-Nearest Neighbors (KNN), Logistic Regression, Random Forest, Gradient

Boosting, Support Vector Regression (SVR), and Adaptive Neuro-Fuzzy inference and Image classification using deep learning has proven to be helpful[14]–[18]. However, through the literature, the researchers did not address the way for controlling plant diseases remotely their system requires the farmers to be on the field to take the picture of a plant and this is time-consuming for farmers having a large plot of arable land.

In this research, the researcher has implemented and tested the TensorFlow Lite python running on Raspberry pi and controlled remotely through a mobile application to perform real-time image classification using image streaming from pi-camera. By using deep learning technology, the researcher has been able to classify the types of maize leaf diseases and monitor the farm remotely by using IoT technology.

1.2. Problem statement

Maize farmers are suffering from low productivity and low quality of maize. The root causes of this problem are lack of knowledge in farming, natural disasters, disease, etc. That low productivity leads to lowering the farmer's income and losing the international maize market. Plant diseases are the most common cause of decreased maize yields. Therefore, in the field of agriculture, plant diseases detection is critical.

Maize plants are particularly vulnerable to diseases that damage the plant's growth, which has an impact on the farmer's sustainability. The symptoms of maize plants can be seen in various parts of the plant, such as the leaves. However, identifying maize plant disease by visual observation is a time-consuming and inaccurate task as most of the medium and small-scale farmers of maize plants don’t have the skills of proper identification of maize plant diseases that they are cultivating resulting in low crop yield. Only agronomists or very experienced farmers can know the types of diseases on the cornfield. Farmers have to invite the agronomists to their farms or farmers take the sample from the farm and go to the laboratory of the agriculture research center. This is expensive and time-consuming. Hence with the use of emerging theologies such as the combination of IoT and AI a visual inspection can be created and helps maize farmers to detect on real-time basis diseases that are facing their plant.

1.3. Study Objectives

1.3.1. General Objective

The research aims to develop an AI-enabled IoT mobile application for early maize plant disease detection.

1.3.2. Specific objectives

i. Identify the disease infecting maize plants across the country.

ii. Create an IoT system for collecting the health condition of maize plants from the field.

iii. Use an AI algorithm for analyzing data corrected from the field.

iv. Create a mobile application that will help farmers to monitor the maize plant from the field remotely.

1.4. Hypotheses

With help of Artificial intelligence and internet of things-based technologies, it is possible to build a Convolutional Neural Network Vision-based model imbedded in raspberry pi for early maize plant diseases detection and give farmers real-time remedies through the mobile application based on the disease.

1.5. Study Scope

This research study was carried out to design and implement a system that should detect the maize disease. The research lay it's focused on three diseases that greatly affect maize yield which in turn result in serious food insecurity. The diseases in focus were Blight-Disease, common-Rust Disease, Gray-Leaf-Spot Disease. Backpropagation Neural Network algorithm was used to identify the diseases affecting the maize leaves based on the pixel features obtained from the image of the leaf. To increase the scope of diseases detection large datasets of different diseases should be used.

1.6. Limitations

The use of drones would have been more appropriate for image capture over large-scale maize farms. However, due to the complexity of their implementation, it was not discussed as part of the research. Another limitation of the research was the availability of datasets of maize plant diseases taken in farms with the board itself and used for training and testing the neural network.

1.7. Contribution of research

Due to similarities of maize leaves diseases, analysis of maize leaves diseases is done by specialists. The task is expensive and takes time to be executed. Small maize growers lack access to these experts, resulting in inefficient production and increased costs. By developing a system for identifying and classifying diseases, maize farmers can increase productivity while lowering costs. There is also a possibility of further studies in the laboratories to make a diagnosis for new diseases on a real-time basis.

1.8. Organization of the Study

This thesis is organized into 6 chapters as follow: The first chapter is the introduction; it describes the background of the study, the problem statement, and the objectives of the study, the scope of the study, the significance of the study, the contribution of research, and the organization of the study. The second chapter is mainly about basic theories and related literature reviews. It clarifies the works done by other researchers on maize plant diseases detection and the rationale of this research after analyzing the existing works. Chapter three is mainly about research methodology, it focuses on different methodological approaches used by the research to carry out the research. Chapter four gives details about the system design and simulation models used. Chapter five explains the machine learning evaluation metrics used and the results found throughout the research. Chapter six gives a conclusion about the research and recommendation for future researchers.

CHAPTER 2: LITERATURE REVIEW

This chapter will lay its focus on a brief analysis of other research that is similar or related to the current research. This includes the problem investigated by previous researchers, proposed techniques, solutions, methodology approach used, and results found. Mobile applications that are associated with the identification of diseases will be discussed. This research will also focus on AI vision systems and the different machine learning techniques that aid in the identification of diseases. Eventually, the gaps in the previous similar and related research were identified through this research study and these provide the justification and motivation for undertaking this research.

In sub-Saharan Africa, Cassava is one of the most important foods for the people. It has the third-largest source of carbohydrates for food. In [19], Godliver et al.,2016; created Machine Learning for Plant Disease Incidence and Severity Measurements from Leaf Images. They captured 7,386 images of leaves of cassava plants, the images are further classified into 5 categories. The health class of images (1476 images), four classes of diseased images representing the 4 diseases, cassava massive diseases (3012 images), cassava brown streak disease (1751 images), cassava bacterial blight (425 images), Cassava green mite (722 images). For the classification of diseases incidence, they use a scikit-learn machine learning toolbox to train a suitable classifier. Three classifiers were trained and used (Linear SVC, KNN, and Extra Trees) unfortunately, the research does not provide complete information on the performance of the model on both training and testing data. The research in [20] Selvaraj et al.,(2019); Various images of plant banana diseases and pest symptoms were collected from various places to implement AI-powered banana disease and pest detection. Data set development, feature extraction, model training, and classification were the four stages of the research. ResNet 50, Inception V2, and MobileNetV1 were utilized to train CNN models. In comparison to MobileNetV1 and ResNet 50, Inception V2 had a high accuracy of 90% at the end of the study. In research [21] Mohamed et al.,(2019); researched plant diseases using support vector machines, during research they used 799 images of different training that further divided it by 80% for training and 20% for testing the accuracy was 88.1%. However, the research doesn’t provide the full information from which plants the model was trained, the model performance for health and unhealthy on each category of the plant because the prediction accuracy alone cannot illustrate how the model performs on each class category of plant.

In research[22] Deep learning was used to create a system that could detect and recognize many plant varieties, including apple, corn, grapes, potato, sugarcane, and tomato. The technology can also detect a variety of plant diseases. The researchers were able to train deep learning models to detect and distinguish plant diseases and the absence of diseases using 35,000 images of healthy plant leaves and diseased plant leaves. The trained model had a 96.5 percent accuracy rate, and the system was able to detect and recognize the plant variety and the type of disease the plant was infected with up to 100 percent accuracy.

In research[23] Aakansha et al., (2015) implemented leaf disease detection and grading using Fuzz logic during research they have been taken into account in developed applications namely, Hydrangea and Maple having two types of diseases namely leaf spot and leaf scorch. The proposed system was divided into two phases, in the first phase the plant was recognized based on the features of the leaf, it was included pre-processing of the leaf. In the second phase the disease in the leaf was classified, this process included K-mean-based segmentation of defected area, feature extraction of defected portion, and the ANN-based classification of disease. Then the disease grading was done based on the amount of disease present in the leaf. After analyzing different studies that have been undertaken by different researchers worldwide on plant diseases detection by using different AI algorithms, the researcher found that those different studies generated different values of plant diseases detection accuracy due to the quality of data. AI projects require a huge amount and good quality of data for allowing the created model to make the best detection with high accuracy. Deep learning algorithms are shown to have good accuracy compared to other algorithms, the ability to train deep learning (DL) systems on large amounts increases the speed of analysis and results[24].

The researchers did not also address the way the created model has to be integrated with Internet of Things (IoT) data from field sensors for helping farmers get information on their plant health remotely. Internet of Things allows the physical objects embedded with sensors to be monitored and controlled remotely at any time and any place by using existing network connectivity[25]. Integrating IoT with the AI vision model embedded in raspberry pi connecting the raspberry pi camera as a sensor adding more features to model like detecting disease remotely.

CHAPTER 3: RESEARCH METHODOLOGY

3.1 Overview

For conducting this research activity, different approaches have been used. Thus, this chapter describes in detail the methodology used by the researcher throughout this study. These include Data Collection, proposed system requirements, Detective Model Modeling, Designing and implementation of IoT-based systems, and finally integration of IoT systems with a Machine Learning predictive model for doing prediction on data from sensors.

3.2 Data Collection

For implementing this research study, the researcher has collected different types of data for creating an appropriate dataset for the classification of corn or maize plant leaf diseases. The dataset images used are from corn plantations obtained from the plant village dataset originally hosted at the Kaggle dataset[26]. Interview as a tool of collecting data was used to obtain information from the farmer and on the awareness of the disease, the measures they take, and the challenges they face. The method was appropriate as it enabled the farmer to highlight things that may have been left out during research. Internet sources were used to gather data on related information to the researcher’s area of study. This was useful in identifying the gaps that would be filled in by the research.

Figures 3.1, 3.2, 3.3, and 3.4 illustrate typical images from all four classes of a dataset. The leaf of a maize plant affected with common rust disease is shown in Figure 3. 2. Brown pustules on both leaf surfaces are the most prevalent symptoms of common rust infection on leaves[27]. In severe cases, the infection might extend to plant sheaths and other parts. Figure 3.3 illustrates Gray leaf spot lesions on maize leaves reduce photosynthetic activity, lowering the number of carbohydrates available for grain fill[28]. The amount to which gray leaf spot reduces crop yields can be calculated using the proportion of infected leaves to grain fill[29]. Figure 3.4 illustrates the impact of Northern Leaf Blight on corn. Northern corn leaf blight lesions begin first on the lower sections of the plant, then spread to the entire plant, turning the leaves pale gray as the blight lesions become larger[30]. Northern maize leaf blight is characterized by cigar-shaped lesions that range in size from one to six inches. Figure 3.1 illustrates a healthy corn leaf.

Figure 3. 1 Health Maize plant

Figure 3. 2: Common-Rust infect corn leaf

Figure 3. 3: Gray-Leaf-Spot infect corn leaf

Figure 3. 4: Northern Leaf Blight infect corn leaf

3.3. Proposed system requirement

This part discusses the design process, software, and hardware components used by the researcher for conducting the research. Software and hardware were selected because they are popular in solving IoT and machine learning classification problems and their availability in huge documentation as many researchers have been using them.

3.3.1. Design process

For implementing maize plant diseases detection application different hardware and software systems were integrated for producing an effective system. The figure below shows the design process.

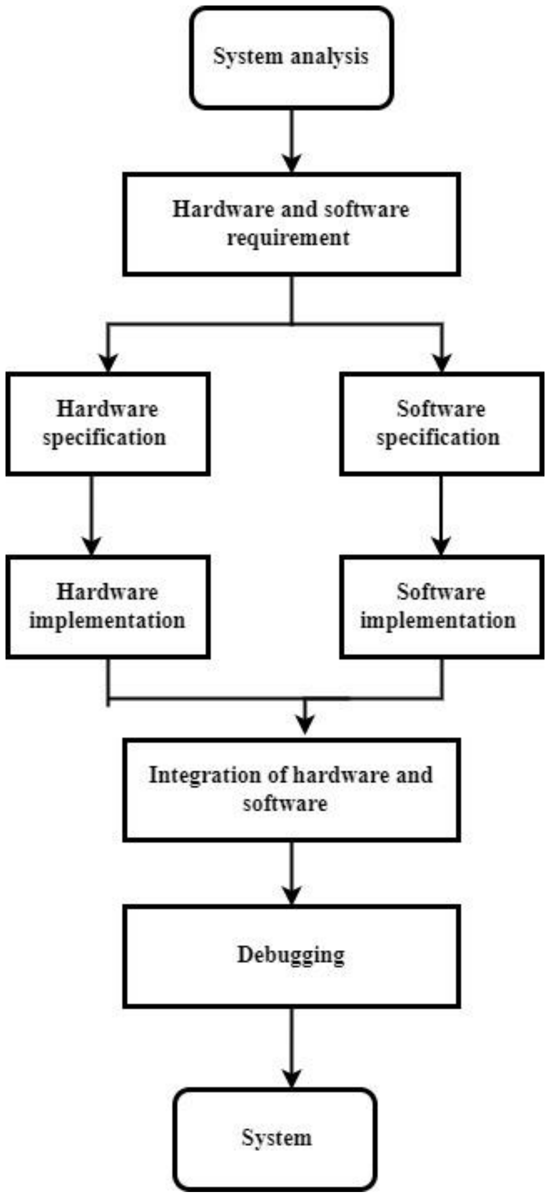


Figure 3. 5: Hardware and Software codesign

3.3.2. Software component

3.3.2.1. Python

Python is a high-level supported language for programming in raspberry pi[31]. For controlling the hardware there are lots of APIs that are nice and easy to use in python. It executes line by line since it is an interpreter. It is available on all platforms like Windows, Linux, etc. and it is also open-source which means anyone can use it. It is frequently used in machine learning and AI programming because it is straightforward to create and understand. After all, the code types are not explicitly declared.

3.3.2.2. TensorFlow

Google Brain Team created TensorFlow, an open-source Python library. It combines several algorithms and models to allow users to build deep neural networks for applications like image recognition and classification. TensorFlow is a sophisticated framework that operates by putting together a series of processing nodes, each representing a mathematical operation. The total set of nodes is referred to as a "graph."[32].

3.3.2.3. Keras

Keras is a Python-based Deep Learning API that runs on top of the TensorFlow machine learning platform. It was developed with the intent of allowing quick experimentation with minimal coding work. Keras implements several optimizers, namely Stochastic Gradient Descent (SGD), RMSProp, AdaGrad, AdataDelta, Adam, Adamax, and Nadam. Keras allows users to experiment with various optimizers with minimal coding effort. Keras implements several activation functions, namely, tanh, sigmoid, and relu (rectified linear unit), etc[33].

3.3.3. Hardware component

3.3.3.1. Raspberry pi

Raspberry pi is a small single-board computer[34]. It has the same component as desktop and personal computer but in a much smaller factor. Depending on how raspberry pi is used it can be an IoT device, reason, why raspberry pi is similar to an IoT device, is that it has network connectivity and computational intelligence which is needed for IoT. It can interface directly with sensors, and actuators via pins. The Raspberry Pi Model B+ has additional GPIO and USB ports than the Model B[35]. Additionally, the power consumption, audio circuit, and SD card are all improved. It's better suited to embedded projects.

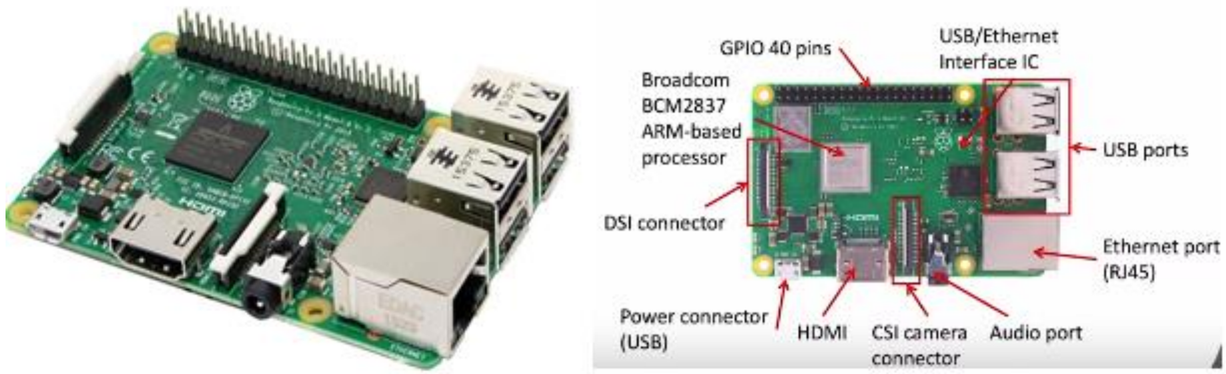


Figure 3. 6: Raspberry pi 3 model

3.3.3.2. Raspberry pi specification

Table 3. 1: Pin description of the Raspberry pi

Chip	BrodcomBCM2835 SoC
Core architecture	ARM 11
CPU	700MHz Low Power ARM1176JZFS core
GPU	Dual Core Video Core IV
	Open GL ES 2.0, Hardware-accelerated Open VG, 1080p30H.264 high-profile decode
	Capable of 1 Gpixel/s, 1.5 Gtexel/s, or 2.4 GFLOPs with texture filtering and DMA infrastructure
Memory	512 MB SDRAM
Operating system	Supports Debian GNU/Linux, Fedora, Arch Linux, RISC OS, and More
Power	Micro USB socket 5V, 2A
Ethernet	10/100 Base T Ethernet socket
Video Output	Composite RCA (PAL and NTSC)
Audio Output	3.5mm jack, HDMI
USB	4xUSB2.0 Ports with up to 1.2A output
GPIO Interface	40-pin 2.54 mm (100mil) expansion header : 2x20 strip providing 27 GPIO pins as well as +3.3V, +5V and GND supply lines
Camera Interface	15-pin MIPI Camera Serial Interface (CSI-2)
JTAG	Not populated

Display Interface	Display Serial Interface (DSI) 15way flat flex cable connector with two data lanes and a clock lane
Memory Card Slot	SDIO

3.3.3.3. Raspberry pi camera module

The Raspberry Pi Camera Module is a Raspberry Pi add-on with a fixed focus 5MP camera capable of 2592x1944 stills, as well as 1080p30, 720p60, and 640x480p60/90 video.[36]. It is connected to Raspberry Pi through small sockets on the raspberry pi’s upper surface boards. This interface uses the dedicated CSI interface, which was designed especially for interfacing with cameras. The CSI bus can handle exceptionally high data rates and is only used to transport pixel data.

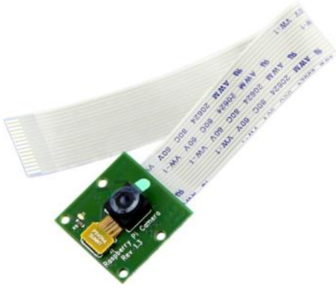


Figure 3. 7: Raspberry pi camera module

3.3.3.4. Android Smart Phone

Smartphones are being utilized for a variety of purposes. They are convenient and have a long battery life, processing at warp speed, a crystal-clear display, and high pixel camera, etc. The research utilized an Android app that employs the phone's camera and takes a picture, uploads a picture to the model, and then the model sends back the plant's identity, such as its disease name and remedies.

3.4. Detective Model Modeling

This part gives a detailed explanation and implementation procedures of Machine Learning Classification algorithm used by the researcher throughout this research work. The Convolutional Neural Network algorithm has been used for data preparation, model creation, training of the model, and model evaluation.

3.4.1. Data pre-processing

Images should be filtered before being sent to CNN for better readability. To achieve better outcomes, images must be processed to an automated deep learning model. Images are augmented during model execution utilizing various parameters like rotation, zoom, scale, batch size, and image size. As a result, it improves deep learning's accuracy and performance. In this research, image preprocessing was used to get data ready for usage, but since a prepackaged dataset is used in this research, the input data normalization should be used.

3.4.1.1. Images Augmentation

Image augmentation is used to make smaller datasets more effective. If the images are fed into the training model with image augmentation such as rotation, zoom, scaling, batch size, and image size.

3.4.2. Convolutional Neural Network

CNN is a set of functions that employ deep learning used for image recognition and natural language processing. CNN consists of multiple layers namely input layers, hidden layers, and output layers with several convolutional and pooling layers [37].

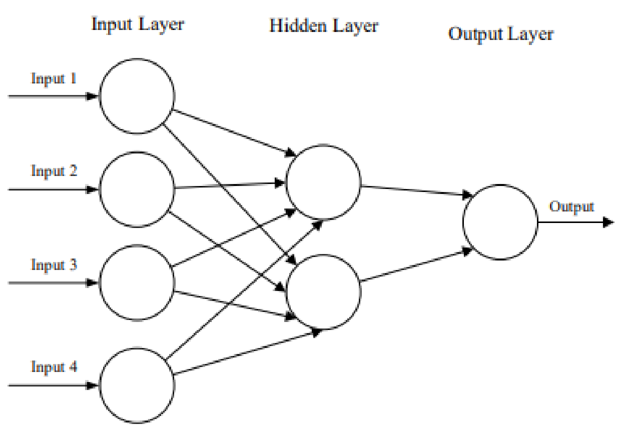


Figure 3. 8: A three-layer Convolutional Neural Network

The operating principle and comprehensive architecture of the constructed Deep Convolutional Neural Network, as shown in figure 3.8, are presented in the next part. Convolutional Neural Networks are made up of several layers, including Convolutional Layers, Max-Pooling Layers, Activation Layers, and Dropout Layers. These steps will be detailed further down.

3.4.2.1. Convolution Layer

Convolution is the most common method for extracting features from input photos. In 2D convolution calculation, a 2-D picture can be projected onto a convolution window that slides continuously to produce the required convolution value. The activation function produces a non-linear mapping between the layers of the network, which allows it to learn and perform more complex tasks. Except for the output layer, where softmax activation is utilized, in this research Rectified Linear Unit (RELU) has been used as an activation function for other layers. It asked Keras to construct 64 filters based on the code, and the filters are 3 by 3.

3.4.2.2. Max-pooling Layer

Max-pooling Layer is placed after each convolution layer so that the network can learn another set of convolutions on top of the existing one, and again pool to reduce the size. So by the time the image gets to flatten layers to go to dense layers, it is already smaller. Its content has been greatly simplified, the goal being that the convolutional will filter it to the features that determine the output. Maximum-pooling has been chosen because maximum value should be taken and in this research, it has been set as a 2 by 2 pool.

3.4.2.3. Drop out Layer

The major goal of employing the dropout layer is to increase the trained model's generalization capability. During the training phase, a drop-out layer with hyper-parameter P ignores activation by probability P at random. Regularization is achieved by lowering correlation between neurons, which successfully reduces over-fitting. During the testing phase, all activations are used, but they are scaled by factor P.

3.4.2.4. Dense Layer

Dense layers are utilized to perform a linear action on input. To produce the output, these entirely linked layers multiply the input by a weight matrix and then add a bias vector[38]. The image is already significantly smaller by the time it flattens to move into the dense layers. It's being quartered, with the intention of convolutional filtering it down to the features that determine the output.

3.4.2.5. Model Summary

The model summary allows inspecting the layers of the model and seeing the journey of the image through the convolution neural network.

3.4.2.6. Compiling Model

The compilation is a time-saving procedure that must always be done after defining a model. It converts the simple layer sequence established into a very efficient series of matrix transforms in a manner that can be run on either the GPU or the CPU, depending on how Keras is set up. This covers loading a set of pre-trained weights from a save file as well as training it using an optimization strategy. The compilation process creates an efficient model of the network, which is also required to make hardware predictions. Several parameters must be given during compilation, all of which are specifically tuned to train your network. The optimization algorithm to use to train the network, as well as the loss function to evaluate the network that is minimized by the optimization process, is shown below. The loss is categorical-crossentropy in this case, and the optimizer is Adm. Categorical-crossentropy is used as a loss function in multi-class classification applications.

3.4.2.6.1. The categorical crossentropy

The categorical crossentropy loss function calculates the loss of an example by computing the following sum:

$$\text{Loss} = -\sum_{i=1}^{\substack{\text{output}\\ \text{size}}} y_i \cdot \log \hat{y}_i$$

The number of scalar values in the model output is the output size, where ŷi is the ith output size is the number of scalar values in the model output, Yi is the corresponding target value, and scalar value is the value in the model output. This loss is a useful indicator of how easy it is to identify two discrete probability distributions from one another. In this case, Yi represents the likelihood that event I will occur, and the sum of all yi equals 1, implying that only one event will occur. When the distributions become closer to one other, the minus sign indicates that the loss decreases.

3.4.2.6.2. Adam Optimizer

Adam is an optimization approach that uses repeated cycles of "adaptive moment estimation" to create more efficient neural network weights[39]. Adam improves on stochastic gradient descent (sdg) to solve non-convex problems more quickly and with fewer resources than many existing optimization tools. It works best in very big data sets since the gradients are kept "tighter" throughout many learning iterations. Adam combines the benefits of Adaptive Gradients and Root Mean Square Propagation, two other stochastic gradient techniques, to provide a novel learning methodology for optimizing a variety of neural networks.

CHAPTER 4: SYSTEM ANALYSIS AND DESIGN

4.1 Introduction

This chapter of the research details the proposed system's design and implementation structure by taking into account the various requirements gathered in the previous chapter through various interactions with the potential users and specialists. The proposed system consists of an embedded system, an Android application, and a connectivity mechanism. Design diagrams and structures explain how farmers will interact with the system and the proposed system's functioning procedure.

4.2 Requirements Analysis

Based on the objectives as well as the user requirements for implementing the proposed system, this section details the numerous research requirements that must be met.

4.2.1 Functional Requirements

i. Picture of maize plant’s leaf should be either taken through mobile application via mobile camera or raspberry pi camera.

ii. Image should be uploaded to the system via internet connectivity.

iii. The system should classify the disease affecting the leaf of the maize using CNN.

iv. The system should return a classification of disease presented on a maize plant leaf.

v. The system should provide a recommendation to be taken based on the type of disease.

4.2.2 Usability Requirements

The application should be basic and straightforward to use because the study site is based in a rural area.

4.2.3 Supportability Requirements

The application should run on a standard smartphone without the need to change any settings of the phone.

4.3 System Architecture

The system architecture shows how to use a Raspberry Pi to detect plant leaf diseases and display the results on an Android app using Wi-Fi technology.

4.3.1 Block diagram design

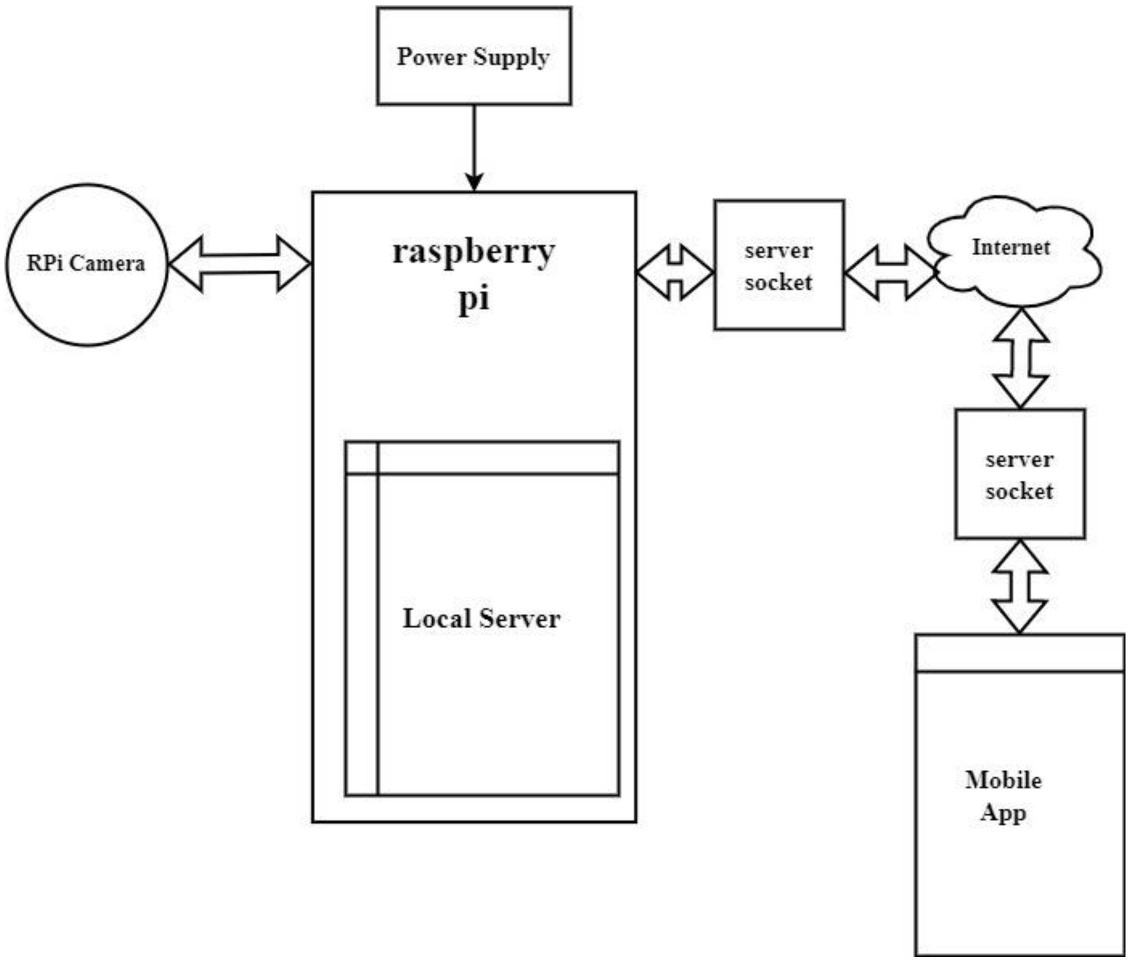


Figure 4. 1: Block diagram

4.3.2. Block Diagram description

It has the Raspberry pi which acts as the main controller. AI model is running on the server implemented in Raspberry pi this is used for processing the picture sent from the mobile app and sending back the result after analysis. A picture can be taken by using a raspberry pi camera module or smartphone camera. The Raspberry pi camera module is connected to Raspberry pi and is controlled by a mobile app. Socket allows communication between raspberry pi and mobile apps through internet connectivity.

4.3.2.1. Raspberry pi installation

The Raspberry Pi is a computer that runs on the Raspberry Pi operating system and is programmed in Python 3.7. Raspberry Pi model B having a system on chip (Soc) BCM2837 has been used in this research. It has 512 MB of RAM and does not have a storage drive, instead of booting off an SD card. The Raspberry Pi Model Bboard is shown in Figure 4.2. It's also linked to a Raspberry Pi camera, which is used to take pictures.

Figure 4. 2: RPi camera connected to raspberry pi

4.3.2.2. Raspberry pi camera configuration

Raspberry pi camera configuration is done by navigating to interfacing options and hitting Enter. And then select enable the raspberry pi camera option, hit the enter key to enable it, and select ok.

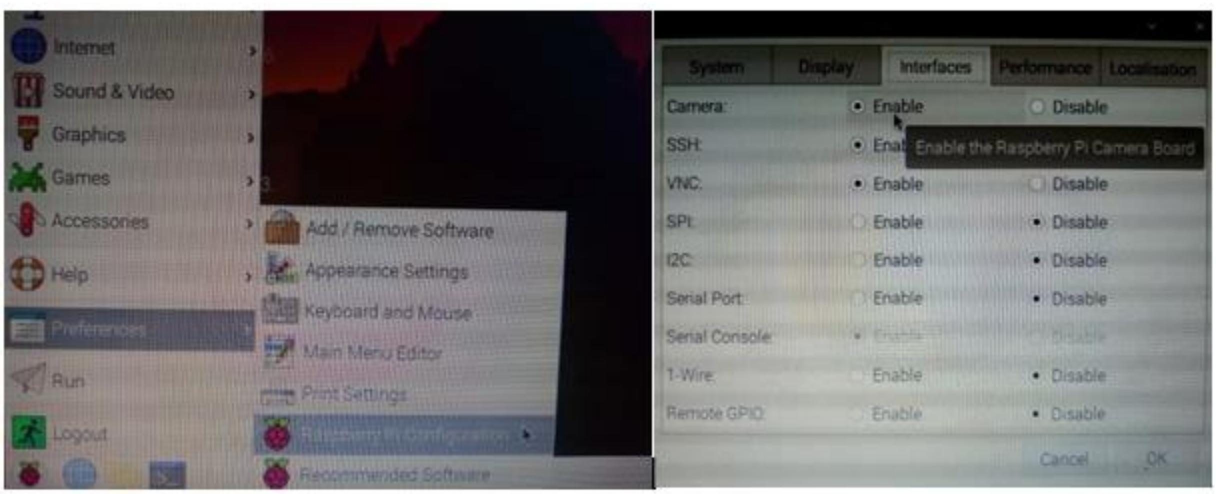


Figure 4. 3: RPi camera configuration

4.3.2.3. Mobile application implementation

A mobile application has been implemented and it is named Doctor Corn. The figure below shows the Doctor Corn app icon on the smartphone.

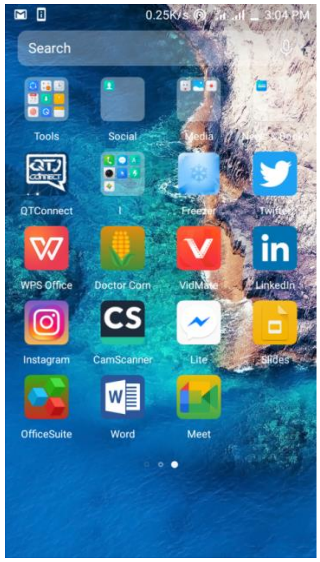


Figure 4. 4: The presence of Doctor Corn in smartphone

The mobile app is used to upload a picture to the local server either by taking a picture from an RPi camera, by using a phone camera, or by selecting a picture from Gallery.

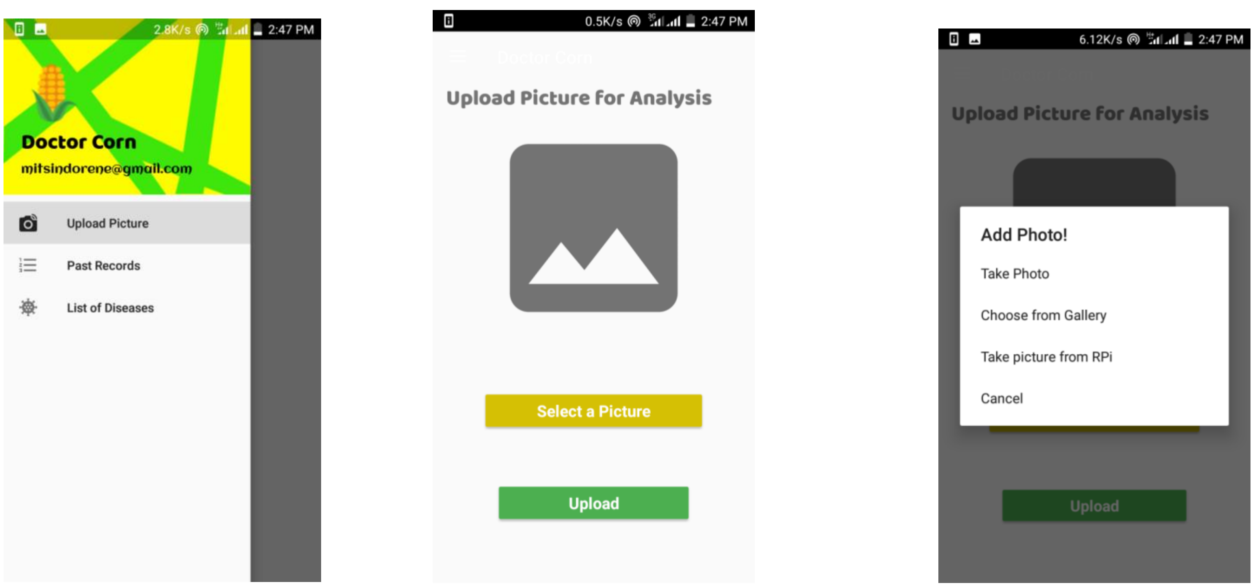


Figure 4. 5: Steps for taking and uploading pictures using a mobile app

4.3.3. Flow chart design

The flow chart diagram indicates the flow of information from sensing data in the farm up to the end-users through the display of information on the mobile application.

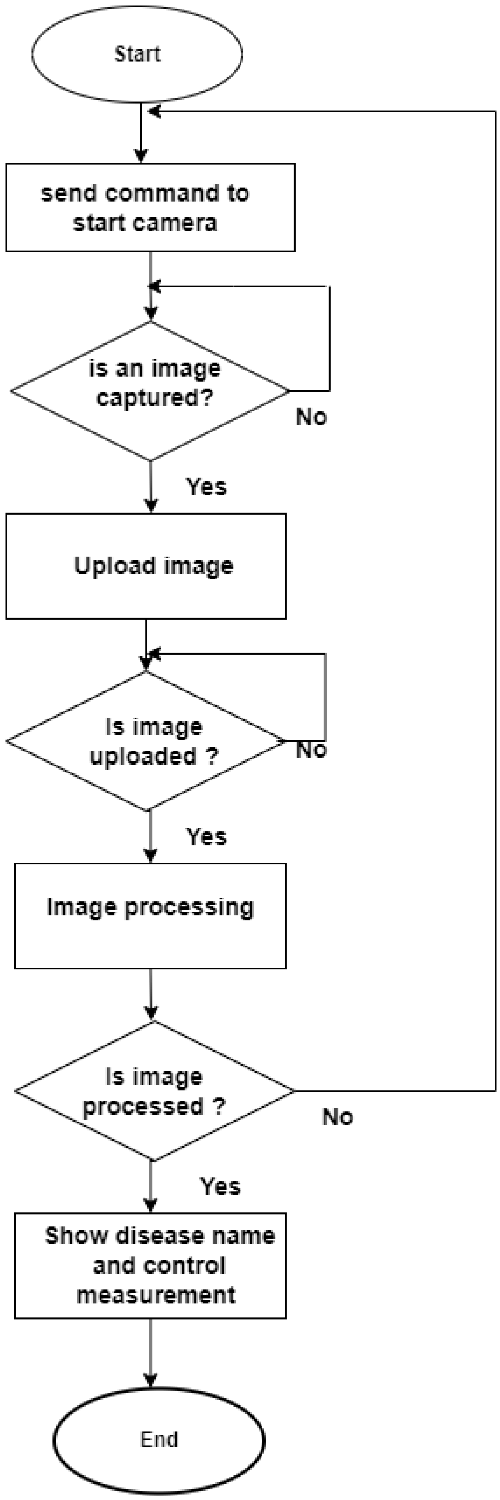


Figure 4. 6: Flow chart design

4.3.4. System overview

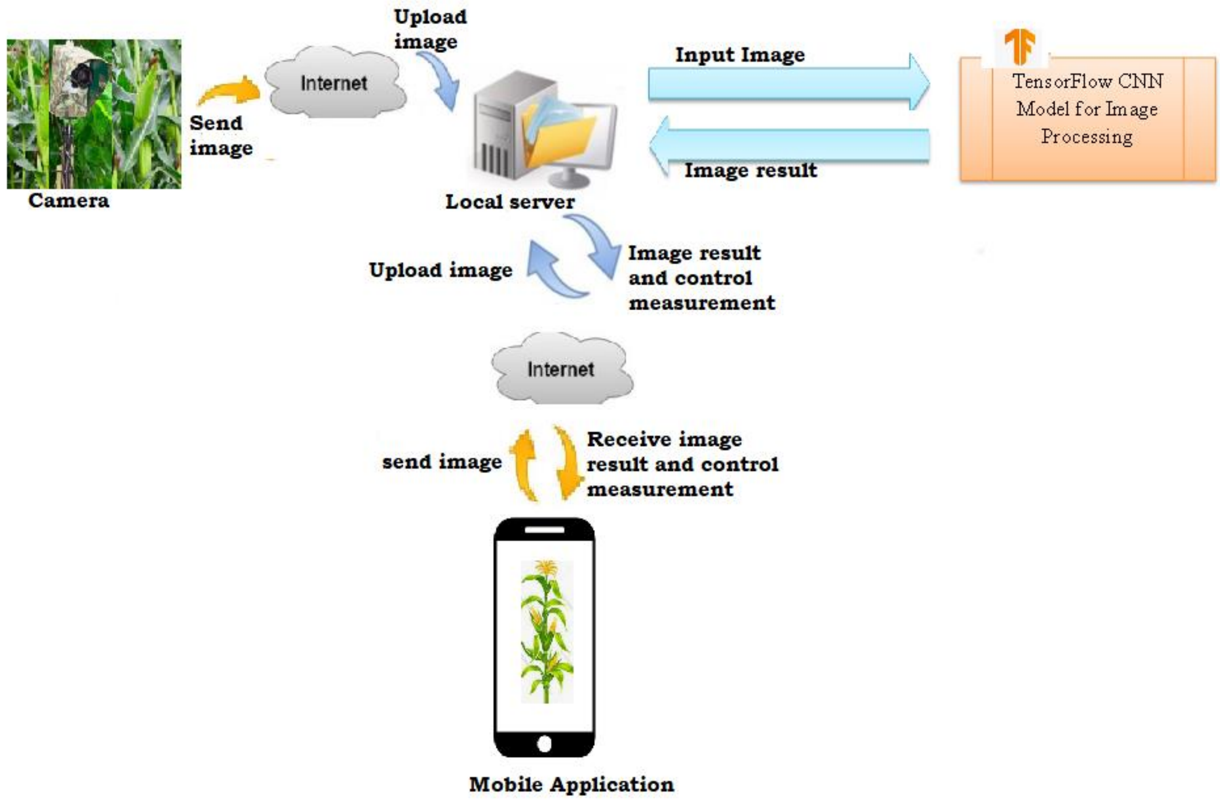


Figure 4. 7: System overview

The model accepts an image of a maize leaf of the user's choice as input, classifies it by comparing it to the pre-saved characteristics of each of the four states (blight, common rust, gray leaf spot, and healthy) that the leaf might be in, and then returns an array of four probability values ranging from zero (0) to one (1), with the highest value among them representing the category that the leaf is most likely to be associated with.

This array will then be compared to a preset array of similar size that contains strings of the names of the classifications present in the model by grabbing the index of the highest probability value and associating it with the index of the string array, and then return the predicted state in the form of a single string. For example, let the resulting probability array be [0.1 0.05 0.09 0.76] and the preset string array be ['Blight','Common_Rust','Gray_Leaf_Spot','Healthy']. Since the fourth element of the probability array is the maximum value, its index value of '3' would then be carried over to the string array, which would result in the string array's fourth element of 'Healthy' to be picked. This would mean that the model has determined to leaf in the photograph to be healthy.

Input Methods

To ensure a high level of flexibility of the system in terms of usability, 3 ways of choosing an image for the model to the process have been provided: taking a picture with the phone's camera, choosing a picture from the phone's gallery, and taking a picture with the Raspberry Pi by using the Pi's camera.

i) Using the phone's camera: Choosing this option in the mobile application will result in the camera process being run. The user will then take a picture which will then be sent to the Raspberry Pi for processing. This is a good method since smartphone cameras take photos that are generally better than the generic Raspberry Pi camera pictures.

ii) Picking a picture from the phone's gallery: Choosing this option in the mobile application will start the phone's gallery where the user can pick any picture of their choice that will then be processed by the Pi. This is very efficient as it will allow for the analysis of samples that the user doesn't have direct access to, such as past samples and samples that are too far for the user to reach. Associates of the user can even take pictures for them and send them to the user for analysis.

iii) Taking a picture from the Raspberry Pi's camera: This method will be helpful in case the Pi system and camera can be securely mounted on the field near the maize. This way, the user can carry out scans from wherever they may be remote.

CHAPTER 5: RESULT AND DISCUSSION

5.1 Machine Learning Model training and evaluation

For training and evaluating the model, a dataset was split into training, validation, and testing sets. 70% of the dataset was used as a training set for training the model, 20% of the dataset was used as a validation set for optimizing model parameters, and 10% of the dataset was used as a testing set for evaluating the final model performance.

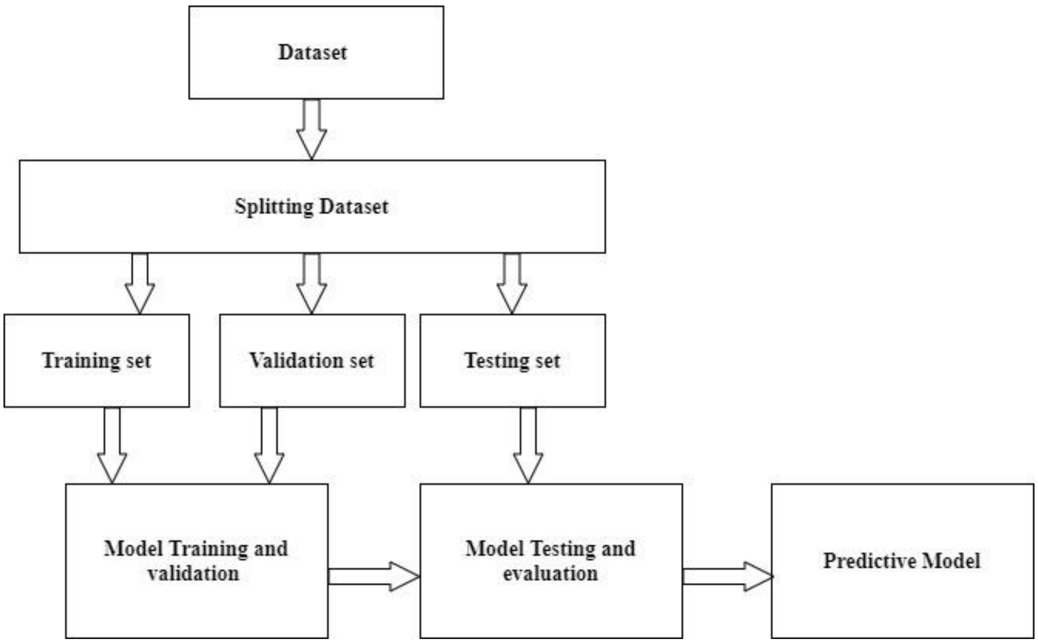


Figure 5. 1: Machine Learning Training and Evaluation Process

A data set of 4691 maize leaves images was divided into 4 different categories namely: healthy, Blight diseases, Common-Rust diseases, and Gray-Leaf-Spot diseases. Table 5.1 shows the dataset description for corn leaf disease classification.

Table 5. 1: Dataset for corn leaf disease classification

Class	health	Common-Rust	Gray-Leaf-Spot	Northern Leaf Blight
Train set	1091	1013	140	1041
Test set	156	145	20	148
Validation set	311	289	40	297
Total images	1558	1447	200	1486

5.1.1 Summary of the model

Convolution and pooling are the two types of layers in the CNN network. Each of these activities is performed by a group of specialized neurons in each layer. Convolution is the process of recognizing the visual properties of objects in an image, such as edges, lines, color drops, and so on. The CNN network can avoid learning unimportant aspects of objects by focusing only on the most important ones due to the pooling process. By summarizing the features into patches, the pooling operation is performed to the output of the convolutional layers to down-sample the resulting feature maps. Average-pooling and maximum-pooling are the two most prevalent pooling procedures. The average-pooling method was utilized in this study, which determines the largest value for each patch of the feature map as the dominating feature. Every Conv2D and MaxPooling2D layer produces a 3D form tensor, as shown in Table 5.2. (Height, width, channels). As we progress deeper into the network, the width and height dimensions begin to reduce. The number of output channels for each Conv2D layer is controlled by the third argument (e.g., 16, 32, or 64). The CNN model generated roughly ten million trainable parameters during the training phase.

Table 5. 2: The Structure of the CNN model

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 222, 222, 64)	1792
max_pooling2d (MaxPooling2D)	(None, 111, 111, 64)	0
conv2d_1 (Conv2D)	(None, 109, 109, 128)	73856
max_pooling2d_1 (MaxPooling2	(None, 54, 54, 128)	0
conv2d_2 (Conv2D)	(None, 52, 52, 256)	295168
max_pooling2d_2 (MaxPooling2	(None, 26, 26, 256)	0
conv2d_3 (Conv2D)	(None, 24, 24, 512)	1180160
max_pooling2d_3 (MaxPooling2	(None, 12, 12, 512)	0
flatten (Flatten)	(None, 73728)	0
dense (Dense)	(None, 128)	9437312
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 4)	516

Total params: 10,988,804
Trainable params: 10,988,804
Non-trainable params: 0

5.1.2 Model evaluation

Finally, the generated Machine Learning model has been evaluated for selecting the best predictive model using training and validation accuracy and with training and validation losses. The prediction accuracy is used to measure how well the generated predictive model is performing. In other words, it compares a predicted value and an observed or known value. The higher prediction accuracy value indicates that the model is performing better. The number of epochs to train over can be used to set the length of training for a network. The longer you train a model, the better it gets, but too many training epochs' increase the risk of overfitting. Our epoch is 20 means that the model will go 20 times. Figures 5.2, 5.3 illustrate the calculated training accuracy and loss graphically. The mean squared error loss decreases over the twelve training epochs, while the accuracy increases unevenly and it is 80% at twenty epochs.

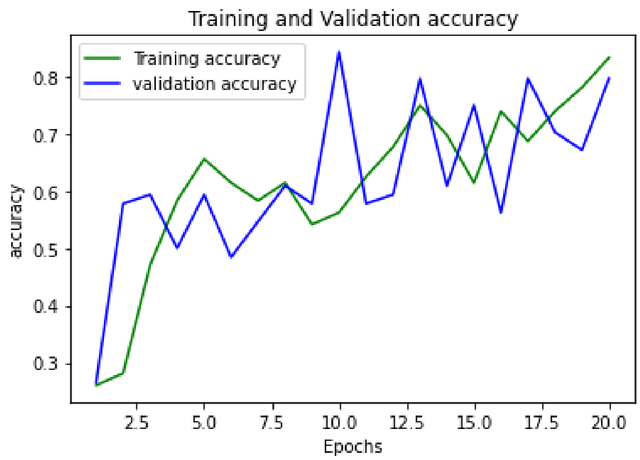


Figure 5. 2: Training and validation accuracy CNN model

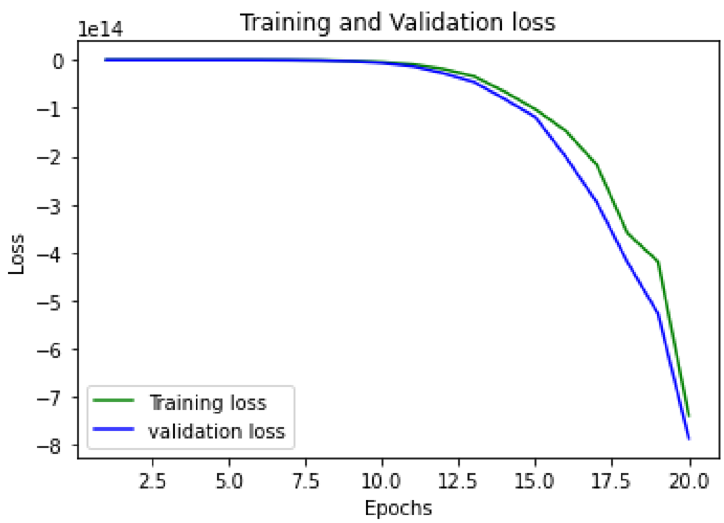


Figure 5. 3: Training and validation loss of CNN model

5.1.3 Execution Time

In addition to accuracy, the model's training time was taken into account; the model's training time was 720 seconds per epoch. This shows that the model is more accurate and takes less time to train. GPU-enabled cores, on the other hand, should be employed for such analyses because they can greatly reduce training time.

5.1.4 Performance evaluation of Convolution Neural Networks

The model has been tested by taking a test image from the test set and printing it with the help of the Matplotlib plotting library. Model process the selected image, then after processing, the model will tell the name of the disease as shown in figure5.4.

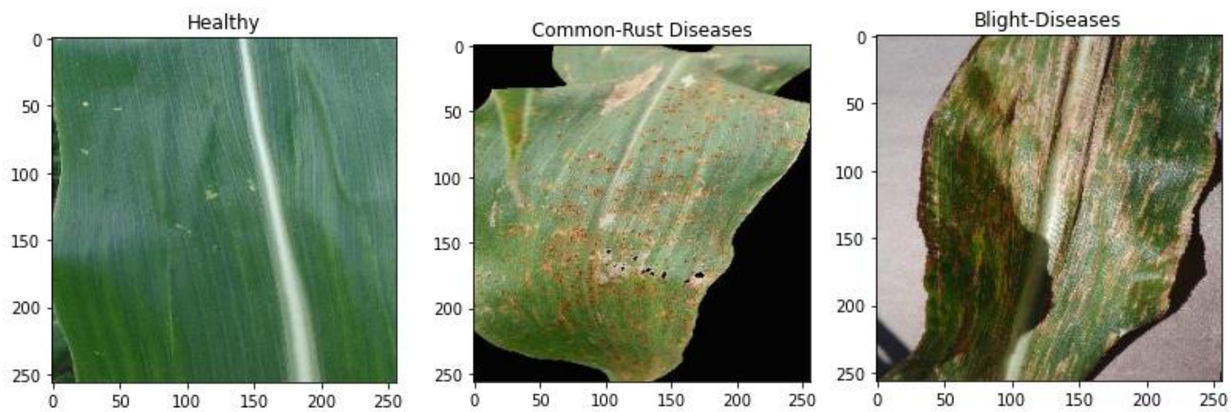
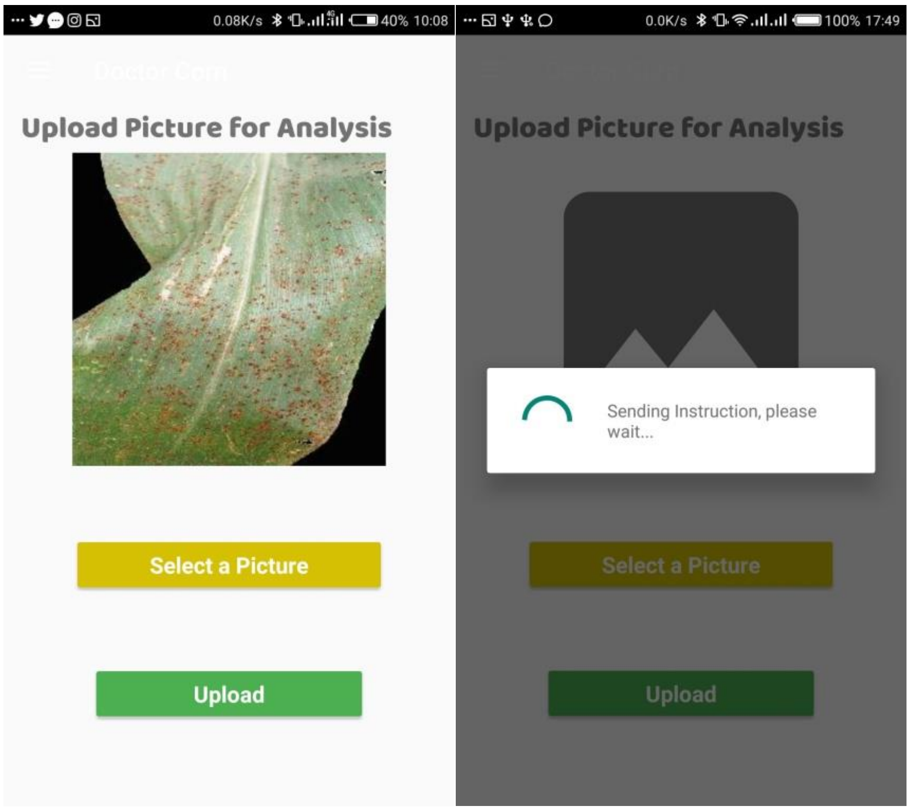


Figure 5. 4: Maize leaf diseases and non-diseases images

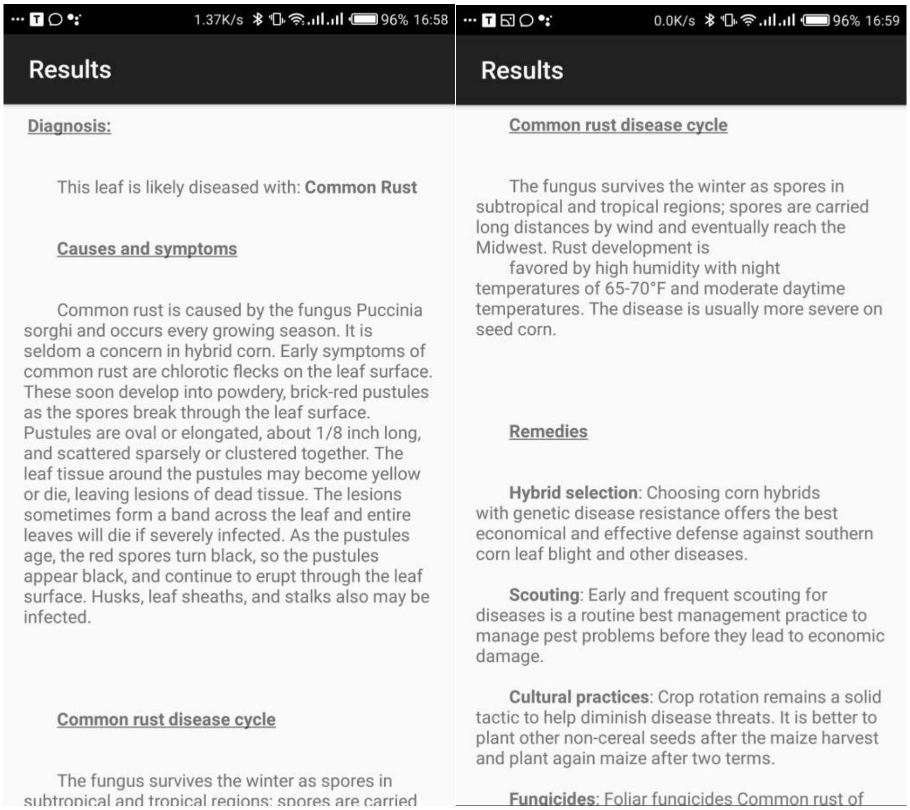
5.2 Integration of IoT system with Machine Learning predictive model evaluation

After training and evaluating the model, it has been deployed in the local server running on raspberry pi for edge computing. A connection between Raspberry pi and smartphones was investigated in this research. There were two approaches to programming communication between a local server operating on a Raspberry Pi and a client. The Raspberry Pi would check to see whether there was any other connected client and if there was, a mutual data exchange would take place. This allows an application to send data through a socket to a hostname called raspberry pi and a port in the Transport Layer using the TCP protocol. The CPU, time required to accomplish various operations, such as photo capture from a Raspberry Pi camera or a phone camera, image preprocessing, and disease recognition procedures has been measured to evaluate the prototype implementation such as classification accuracy and performance. Each experiment in this section was carried out for a total of twenty trials, after which the findings were averaged.

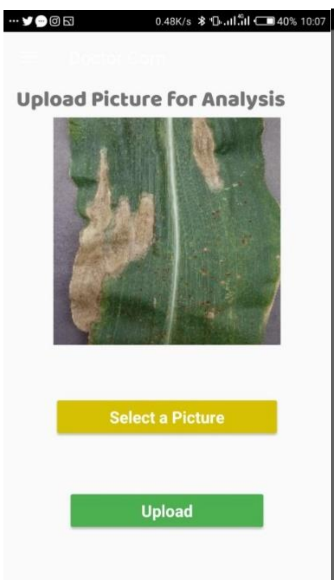
The prediction of disease class and the display of the result took about 10 seconds. This demonstrates that the approach can be utilized as a real-time plant disease detector at the edge. When the plant photos are captured from 10 cm distances from the camera, the method produces good results in terms of classification accuracy. Figure 5.5 illustrates several examples of successful plant leaf disease recognition. This disease detector gets a high classification rate for most of the classes in the testing dataset, as shown in Figures 5.5 a to 5.5 f. However, due to a lack of sufficient training data, the system fails to attain high accuracy for Gray-Leaf-Spot Diseases.

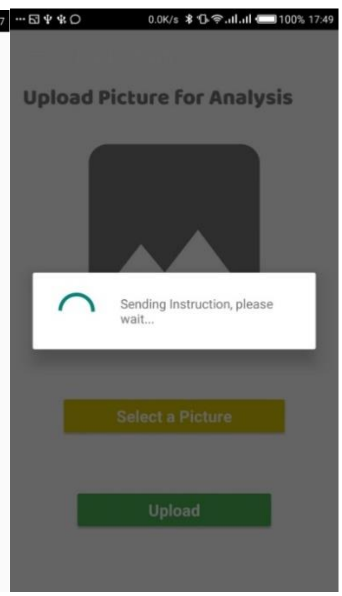


a) Image captured b) Image processing



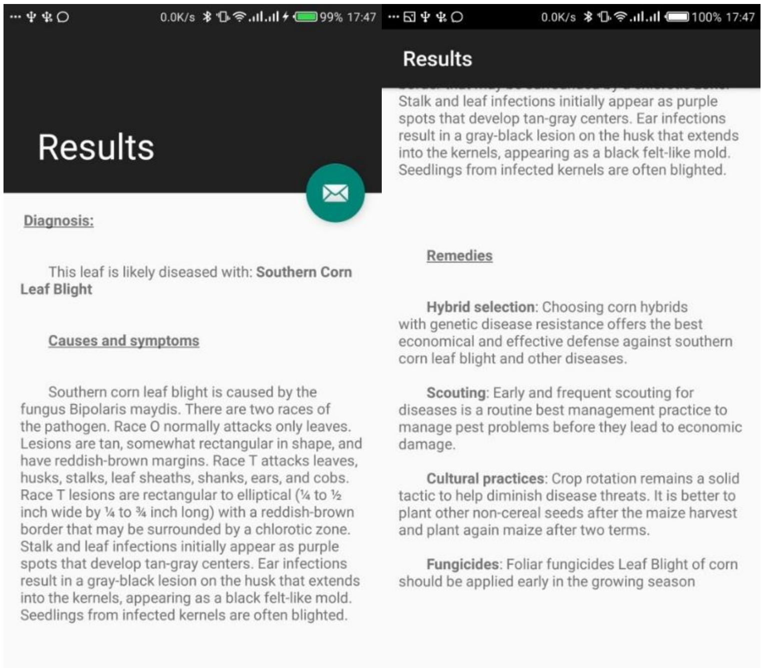
c) Image result





d) Image captured

e) Image processing



f) Image result

Figure 5. 5: Screenshots of the Mobile App for Detecting Plant Leaf Diseases and remedies

5.3 Discussion

The main objective was to implement a detection model that is efficient in the identification and classification of maize leaf disease. To understand the current limits and research gaps, a thorough literature review was conducted. These primarily highlighted the necessity for models that could predict well-unseen data remotely. A deep learning model using convolutional neural networks algorithms is implemented on raspberry for such issues. The model was implemented by using maize leaf images containing healthy leaves and diseases leaves. For building the model necessary pre-processing involving images resizing and converting them to numpy arrays were done. Image Augmentation and validation were done for every epoch to ensure that the model is accurate. After training and testing the model, the results show an average accuracy of 80 % for detecting diseases in maize leaves.

For detecting diseases remotely which is making this research different from other methods used to detect plant diseases a farmer can take a picture of an image plant remotely by triggering a remote camera through a mobile app. When the camera is triggered, it directly takes a picture and loads it to the model deployed on raspberry pi for prediction, the model process the image and gives back the result to the farmer through a mobile app. Bidirectional communication between a mobile app and the model deployed on a raspberry pi camera is achieved by the implementation of a socket.

Based on the results obtained from the discussions held with farmers, the method used by a farmer for plant diseases identification is visual observation and this method is prone to errors and inaccuracies. The approach implemented in this research can eliminate errors committed by farmers during the detection of plant disease based on algorithms used and on the fact that it can be controlled remotely.

Although this study met its objectives, it encountered certain difficulties due to a lack of data sets collected from various locations and backgrounds that could have been utilized to further test and improve the model; as a result, models may underperform. Multiple leaf images or images damaged by multiple diseases might be difficult to classify, and their classifications are not tested.

CHAPTER 6: CONCLUSION, RECOMMENDATIONS, AND FUTURE WORKS

6.1 Conclusion

Early detection and classification of plant diseases are so important for the successful cultivation of crops. Farmers face several challenges in identifying and managing diseases affecting their crops. One of the major problems is misdiagnosis which is based on the prior experience of the farmer. Misdiagnosis results in farmers using the wrong action and thus obtaining a low yield. This research has successfully achieved objectives by detecting the maize leaf diseases with the help of CNN and open CV through a model deployed on raspberry pi and displaying the result on a mobile app. the model has an overall classification accuracy of 80% when it comes to distinguishing the three most common disease groups that damage maize leaves. This proves that the model is capable of detecting and classifying maize plant diseases early with good predictability and response time.

6.2 Recommendations

Based on the results obtained from this research, there are still many improvements and enhancements that can be done in this research:

i. Use of drones would be an added advantage, especially in large fields

ii. Location information should also be added to the application since different zones are prone to different kinds of disease based on the ecological zones

iii. To increase accuracy, more sensors need to be added to collect more data such as soil nutrients, humidity, weather condition, etc.

6.3 Suggestions for Future Research.

i. The researcher recommends that more data from different sensor nodes such as NPK sensor, humidity and temperature sensor, etc. should be considered as inputs to the system to give a more accurate classification.

ii. The application can be built to support local languages since most farmers in the rural areas do not know abroad languages.

iv. The application might be expanded to include forums where farmers could discuss information based on their location. An expert should also be involved at the forum to provide expert advice on the various issues raised by the farmer.

REFERENCES

[1] "https://www.worldbank.org/en/results/2013/01/23/agricultural-development-in-rwanda." .

[2] D. Malunda and S. Musana, "Rwanda Case Study on Economic," *Inst. Policy Anal. Res. Rwanda*, 2012.

[3] "https://rdb.rw/investment-opportunities/agriculture/." .

[4] N. Mulinga, "Economic Analysis of Factors Affecting Technical Efficiency of Smallholders Maize Production on Rwanda," *Rwanda J.*, vol. 1, no. 1, pp. 52–62, 2013, doi: 10.4314/rj.v1i1.4h.

[5] "NATURE OF MAIZE VALUE CHAIN IN RWANDA AND ITS IMPLICATION ON AGRIBUSINESS PERFORMANCE A CASE STUDY OF COOPERATIVE DE MAIZE DU RUKOMO KABAYIZA SOSTHENE Reg N o 219014084 A Thesis Submitted in Partial Fulfillment for the Requirements for the Award of Degree," no. September, 2019.

[6] I. Journal, O. F. Engineering, P. P. Detection, C. Using, and R. Pi, "International journal of engineering sciences & research technology plant pathology detection and control using raspberry pi 1," vol. 7, no. 3, pp. 519–525, 2018.

[7] W. Aragón, J. J. Reina-Pinto, and M. Serrano, "The intimate talk between plants and microorganisms at the leaf surface," *J. Exp. Bot.*, vol. 68, no. 19, pp. 5339–5350, 2017, doi: 10.1093/jxb/erx327.

[8] O. Degani, D. Regev, and S. Dor, "The microflora of maize grains as a biological barrier against the late wilt causal agent, magnaporthiopsis maydis," *Agronomy*, vol. 11, no. 5, pp. 1–15, 2021, doi: 10.3390/agronomy11050965.

[9] E. Lucas, L. F. Rodrigues, and J. F. Mari, "Maize leaf disease classification using convolutional neural networks and hyperparameter optimization."

[10] S. Bansal and A. Kumar, "DEEP LEARNING FRAMEWORKS FOR MAIZE DISEASE DETECTION : A REVIEW," vol. 7, no. 09, pp. 43–49, 2020.

[11] A. Devaraj, K. Rathan, S. Jaahnavi, and K. Indira, "Identification of plant disease using image processing technique," *Proc. 2019 IEEE Int. Conf. Commun. Signal Process. ICCSP 2019*, pp. 749–753, 2019, doi: 10.1109/ICCSP.2019.8698056.

[12] E. Mavridou, E. Vrochidou, G. A. Papakostas, T. Pachidis, and V. G. Kaburlasos, "Machine vision systems in precision agriculture for crop farming," *J. Imaging*, vol. 5, no. 12, 2019, doi: 10.3390/jimaging5120089.

[13] N. Petrellis, "A review of image processing techniques common in human and plant disease diagnosis," *Symmetry (Basel).*, vol. 10, no. 7, 2018, doi: 10.3390/sym10070270.

[14] E. Hossain, M. F. Hossain, and M. A. Rahaman, "A Color and Texture Based Approach for the Detection and Classification of Plant Leaf Disease Using KNN Classifier," *2nd Int. Conf. Electr. Comput. Commun. Eng. ECCE 2019*, pp. 1–6, 2019, doi: 10.1109/ECACE.2019.8679247.

[15] Y. H. Gu *et al.*, "BLITE-SVR: New forecasting model for late blight on potato using support-vector regression," *Comput. Electron. Agric.*, vol. 130, pp. 169–176, 2016, doi: 10.1016/j.compag.2016.10.005.

[16] B. J. Samajpati and S. D. Degadwala, "Hybrid approach for apple fruit diseases detection and classification using random forest classifier," *Int. Conf. Commun. Signal Process. ICCSP 2016*, no. 2013, pp. 1015–1019, 2016, doi: 10.1109/ICCSP.2016.7754302.

[17] K. Sudha Rani and B. Priya Madhuri, "Plant Leaf Disease Detection Using Machine Learning Techniques," *Lect. Notes Data Eng. Commun. Technol.*, vol. 58, pp. 511–518,

2021, doi: 10.1007/978-981-15-9647-6_40.
[18] H. Sabrol and S. Kumar, *Plant Leaf Disease Detection Using Adaptive Neuro-Fuzzy Classification*, vol. 943. Springer International Publishing, 2020.
[19] G. Owomugisha and E. Mwebaze, “Machine learning for plant disease incidence and severity measurements from leaf images,” *Proc. - 2016 15th IEEE Int. Conf. Mach. Learn. Appl. ICMLA 2016*, pp. 158–163, 2017, doi: 10.1109/ICMLA.2016.126.
[20] M. G. Selvaraj *et al.*, “AI-powered banana diseases and pest detection,” *Plant Methods*, vol. 15, no. 1, pp. 1–11, 2019, doi: 10.1186/s13007-019-0475-z.
[21] M. Hussein and A. H. Abbas, “Plant Leaf Disease Detection Using Support Vector Machine,” *Al-Mustansiriyah J. Sci.*, vol. 30, no. 1, p. 105, 2019, doi: 10.23851/mjs.v30i1.487.
[22] T.-H. Meen, Institute of Electrical and Electronics Engineers, National Formosa University, and International Institute of Knowledge Innovation and Invention, “IOT, Communication and Engineering : 2019 IEEE Eurasia Conference on IOT, Communication and Engineering (IEEE ECICE 2019) : Yunlin, Taiwan, October 3-6, 2019,” *2019 IEEE Eurasia Conf. IOT, Commun. Eng.*, pp. 579–582, 2019.
[23] A. Rastogi, R. Arora, and S. Sharma, “Leaf disease detection and grading using computer vision technology & fuzzy logic,” *2nd Int. Conf. Signal Process. Integr. Networks, SPIN 2015*, pp. 500–505, 2015, doi: 10.1109/SPIN.2015.7095350.
[24] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, “Deep learning applications and challenges in big data analytics,” *J. Big Data*, vol. 2, no. 1, pp. 1–21, 2015, doi: 10.1186/s40537-014-0007-7.
[25] F. Wortmann and K. Flüchter, “Internet of Things: Technology and Value Added,” *Bus. Inf. Syst. Eng.*, vol. 57, no. 3, pp. 221–224, 2015, doi: 10.1007/s12599-015-0383-3.
[26] “https://www.kaggle.com/smaranjitghose/corn-or-maize-leaf-disease-dataset.” .
[27] J. Sucher *et al.*, “The durable wheat disease resistance gene Lr34 confers common rust and northern corn leaf blight resistance in maize,” *Plant Biotechnol. J.*, vol. 15, no. 4, pp. 489–496, 2017, doi: 10.1111/pbi.12647.
[28] D. Corn, “What Should You Know about Corn and Soybean Diseases as You Prepare for Harvest ?,” vol. 22, no. 27, 2016.
[29] J. M. J. Ward, E. L. Stromberg, and D. C. Nowell, “’ray ,eaf 3pot !,” vol. 83, no. 10, 1999.
[30] S. Hurni *et al.*, “The maize disease resistance gene Htn1 against northern corn leaf blight encodes a wall-associated receptor-like kinase,” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 112, no. 28, pp. 8780–8785, 2015, doi: 10.1073/pnas.1502522112.
[31] M. Kölling, “Educational programming on the Raspberry Pi,” *Electron.*, vol. 5, no. 3, pp. 1–17, 2016, doi: 10.3390/electronics5030033.
[32] M. Abadi *et al.*, “TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems,” 2016, [Online]. Available: http://arxiv.org/abs/1603.04467.
[33] N. Ketkar, “Deep Learning with Python,” *Deep Learn. with Python*, pp. 95–109, 2017, doi: 10.1007/978-1-4842-2766-4.
[34] M. Saari, A. M. Bin Baharudin, and S. Hyrynsalmi, “Survey of prototyping solutions utilizing Raspberry Pi,” *2017 40th Int. Conv. Inf. Commun. Technol. Electron. Microelectron. MIPRO 2017 - Proc.*, pp. 991–994, 2017, doi: 10.23919/MIPRO.2017.7973568.
[35] I. Beloev, D. Kinaneva, G. Georgiev, G. Hristov, and P. Zahariev, “Artificial intelligence-driven autonomous robot for precision agriculture,” *Acta Technol. Agric.*, vol. 24, no. 1,

pp. 48–54, 2021, doi: 10.2478/ata-2021-0008.
[36] H. Q. Nguyen, T. T. K. Loan, B. D. Mao, and E. N. Huh, "Low cost real-time system monitoring using Raspberry Pi," *Int. Conf. Ubiquitous Futur. Networks, ICUFN*, vol. 2015-Augus, no. February 2020, pp. 857–859, 2015, doi: 10.1109/ICUFN.2015.7182665.
[37] "https://searchenterpriseai.techtarget.com/definition/convolutional-neural-network."
[38] D. M. Pelt and J. A. Sethian, "A mixed-scale dense convolutional neural network for image analysis," *Proc. Natl. Acad. Sci. U. S. A.*, vol. 115, no. 2, pp. 254–259, 2017, doi: 10.1073/pnas.1715832114.
[39] A. Chakrabarty, C. Danielson, S. A. Bortoff, and C. R. Laughman, "Accelerating self-optimization control of refrigerant cycles with Bayesian optimization and adaptive moment estimation," *Appl. Therm. Eng.*, vol. 197, no. July, p. 117335, 2021, doi: 10.1016/j.applthermaleng.2021.117335.

APPENDIX

Python Code for Data Augmentation

```
train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=25,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.15,
    zoom_range=0.15,
    horizontal_flip=True,
    vertical_flip=True)

test_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
        train_dir,
        target_size=(224, 224),
        batch_size=32,
        class_mode='categorical')

validation_generator = test_datagen.flow_from_directory(
        validation_dir,
        target_size=(224, 224),
        batch_size=32,
        class_mode='categorical')
```

Python Code for Convolutional Neural Network

```
model =Sequential()

model.add(Conv2D(64, (3, 3), activation='relu',input_shape=(224, 224, 3)))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(256, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(512, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Flatten())
model.add(Dense(128,activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(4,activation='softmax'))
```

Python Code for Compiling Model

```
learning_rate=0.001
Epochs=25
BS=32
opt=Adam(learning_rate=learning_rate,decay=learning_rate/Epochs)
model.compile(loss='categorical_crossentropy',optimizer=opt,metrics=['accuracy'])
```

Python Code for Training and Validating CNN model

```
history = model.fit_generator(
      train_generator,
      steps_per_epoch=train_images//BS,
      epochs=100,
      validation_data=validation_generator,
      validation_steps=validation_images//BS)
```

Python Code for Training and validating accuracy of CNN model

```
acc_train = history.history['accuracy']
acc_val = history.history['val_accuracy']
epochs = range(1,21)
plt.plot(epochs, acc_train, 'g', label='Training accuracy')
plt.plot(epochs, acc_val, 'b', label='validation accuracy')
plt.title('Training and Validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('accuracy')
plt.legend()
plt.show()
```

Python Code for Training and validating loss of CNN model

```
loss_train = history.history['loss']
loss_val = history.history['val_loss']
epochs = range(1,21)
plt.plot(epochs, loss_train, 'g', label='Training loss')
plt.plot(epochs, loss_val, 'b', label='validation loss')
plt.title('Training and Validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```