A DEEP LEARNING MODEL FOR EARLY DETECTION OF MAIZE DISEASES IN TANZANIA

Flavia Stephen Mayo

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of Master's in Information and Communication Science and Engineering of the Nelson Mandela African Institution of Science and Technology

Arusha, Tanzania

August, 2024

ABSTRACT

Agriculture is considered the backbone of Tanzania's economy, with more than 60% of residents depending on it for their livelihood. Maize is the main and dominating food crop in the country, accounting for 45% of all farmland produce. Inevitably, its production is affected by diseases that, if detected early, could be easily treated. Maize Streak Virus (MSV) and Maize Lethal Necrosis (MLN) are commonly reported diseases that are often detected too late by farmers. This raised the need for a sophisticated method for early detection of these diseases to enable timely treatment. This study investigated the potential of developing a deep-learning model for the early detection of maize diseases in Tanzania. Imagery datasets were used for model training, they were collected through physical observation with the help of a plant pathologist from three regions of the country; Arusha, Kilimanjaro, and Manyara. Two models, a Convolutional Neural Network (CNN) and a Vision Transformer (ViT), were developed from scratch and classified into four classes: Healthy, MLN, MSV, and WRONG. The results revealed that the ViT model outperformed the CNN model, achieving validation accuracies of 0.931 and 0.9096 respectively. Despite the superior performance of the ViT model, the CNN model was selected for deployment in a mobile-based application due to its smaller size, lower memory, and small computational requirements. A quantitative research method using a survey questionnaire was employed to gather requirements for system development and validation. Validation of the model's performance through questionnaires administered to farmers and agricultural experts, yielding highly positive feedback. This enthusiastic reception highlights the potential impact of the developed application on improving early disease detection and enhancing maize productivity in Tanzania. Empowering stakeholders of agriculture to identify more effective methods of managing them before serious harm is done. Consequently, custodians of agriculture, such as the Ministry of Agriculture, organizations, and other companies, will have the potential to detect maize diseases early, henceforth adding up to the increase in the Gross Domestic Product (GDP) and guaranteeing the country's food security.

DECLARATION

I, Flavia Stephen Mayo do hereby declare to the Senate of the Nelson Mandela African Institution of Science and Technology that this dissertation is my original work and that it has neither been submitted nor being concurrently for degree in any other institution.

Flavia Stephen Mayo

Name of Student　　　　Signature　　　　Date

The above declaration is confirmed

Dr. Neema Mduma

The Supervisor's Name　　　　Signature　　　　Date

Prof. Ciira Maina　　　　　　　　12/08/2024

The Supervisor's Name　　　　Signature　　　　Date

Dr. Mvurya Mgala　　　　　　　　12/08/2024

The Supervisor's Name　　　　Signature　　　　Date

CERTIFICATION

The undersigned certify that they have read and hereby recommend for acceptance by The Nelson Mandela African Institution of Science and Technology, a dissertation entitled ***"A Deep Learning Model for Early Detection of Maize Diseases in Tanzania"*** submitted in partial fulfillment of the requirements for award of the degree of "Master's in Information and Communication Science and Engineering" of the Nelson Mandela African Institution of Science and Technology.

Dr. Neema Mduma

The Supervisor's Name   Signature   Date

Prof. Ciira Maina   12/08/2024

The Supervisor's Name   Signature   Date

Dr. Mvurya Mgala   12/08/2024

The Supervisor's Name   Signature   Date

ACKNOWLEDGEMENTS

I thank the Almighty God for the gift of life and the opportunity to study. I appreciate God for leading me and being with me throughout the study period. I have reached this stage because God made a way for me. I thank the Holy Spirit, for always teaching me and enlighten me.

On a more personal level, I thank God for my wonderful family, Father Ingi Mayo Stephen, Mama Celina Mayo, my sister Flora Mayo, and brother Florence Mayo for unending prayers and support, which have propelled me throughout this academic journey.

It has been an honor and a blessing to be supervised by Dr. Neema Mduma, Dr. Mvurya Mgala, and Prof. Ciira Maina, their professional and scientific guidance has enabled me to successfully finish my studies. I am grateful for all their tireless effort.

I would like to thank the International Development Research Centre (IDRC) and Swedish International Development Cooperation Agency (SIDA) through the Lacuna Fund in Agriculture and the African Centre for Technology Studies (ACTS) for funding my research work.

I especially thank my colleagues Christian, Petro, Rebecca, Abuu, Lusungu, Nguse, Edna, Angela, Japheth and all my friends for their support and cooperation.

DEDICATION

To my precious parents Father Stephen Ingi Mayo and Mama Celina Mayo: I am forever grateful for their true love and support, May God bless them abundantly.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

LIST OF ABBREVIATIONS AND SYMBOLS

MSV	Maize Streak Virus
MLN	Maize Lethal Necrosis
AI	Artificial Intelligence
APK	Android Application Package
API	Application Programming Interface
CNN	Convolutional Neural Network
ViT	Vision transformer
DL	Deep learning
GB	Giga Bytes
GDP	Gross Domestic Product
GPU	Graphical Processing Unit
IDE	Integrated Development Environment
ML	Machine Learning
RGB	Red Green Blue
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
SDGs	Sustainable Development Goals
DL	Deep Learning
JPEG	Joint Photographic Group
ODK	Open Data Kit PC
ReLU	Rectified Linear Activation Function

GELU	Gaussian Error Linear Unit
RGB	Red Green Blue

CHAPTER ONE

INTRODUCTION

1.1 Background of the Problem

Tanzania's economy is predominantly centered around agriculture, and the country gains from a wide range of agricultural activities, such as livestock, essential food crops, and several cash crops (Oxfordbusinessgroup, 2018). The agricultural output accounts nearly to 29.1% of the country's Gross Domestic Product (GDP). It is a paramount supplier of food and materials for industry, as well as foreign exchange. Agriculture also employs 67% of the labor force in the country (Trade.gov, 2021). Meanwhile, as agronomy production is far too low, food demand is increasing dramatically (Dewbre *et al.,* 2014). Farmers, scientists, researchers, analysts, specialists, and the government are all working hard to enhance agricultural production to meet the growing need (Panigrahi *et al.,* 2020). However, crop diseases continued to be the challenge affecting major food security crops including maize (Savary & Willocquet, 2020). The most extensively planted food crop in Tanzania is maize, constituting 62.6%, followed by rice which accounts for 21.6% of the total, Pulses at 5.1%, and finally, wheat at 0.7% (Tanzaniainvest, 2021).

Maize was brought to the coast of Africa for the first time in the 17^{th} century. Initially, the Portuguese introduced the cultivation of maize crops to sustain their trading forts, but African farmers immediately embraced it because of its high yield, affordable labor needs, and quick development period. In Tanzania, maize, the primary cereal crop, is extensively cultivated across agricultural zones and regions, with different varieties well-suited for a wide range of agroecological conditions, spanning from nearly sea level to altitudes as high as 2400 meters above sea level. Conversely, the most extensive agricultural regions are situated at elevations ranging from 500 to 1500 meters. The Southern Highlands Zone and the Lake Zone collectively make up approximately 26% and 25% of Tanzania's maize cultivation areas, respectively. These are followed by the Eastern Zone (13%), Northern Zone (12%), Western Zone (10%), Southern Zone (8%), and finally Central Zone (%). Small-scale farmers account for 85% of total production, with medium and large-scale farmers accounting for 10% and 5%, respectively (Tanzaniainvest, 2021). Despite its crucial role in ensuring food security, maize production in Tanzania is still below the expected capacity. According to the study by Robertson *et al.* (2021), Maize is susceptible to plant pathogen infections such as rots in the seeds and blights in the seedlings shortly following the planting. In the mid-season, foliar diseases appear while towards the end of the season stalk and ear rot appear. These diseases are a result of infectious agents that are found in agricultural leftovers around the soil. In recent years, maize growth and production have been threatened by the diseases of Maize Lethal Necrosis (MLN) and Maize Streak Virus (MSV). Maize Lethal Necrosis is a complex viral infection, has been causing significant challenges to the growth of maize plants as well as the well-being of small-scale farmers in East Africa as of 2011, due to the outbreak of the Maize Chlorotic Mottle Virus (Boddupalli *et al.,* 2020). Maize Streak Virus, which belongs to the Mastrevirus genus within the Geminiviridae family, is not only responsible for maize streak disease but also represents the majority of the severe virus threat corn crops in Africa parts of Sub-Saharan. In the majority of areas on the continent, the virus remains the most uncontrolled, contributing to massive yield losses and starvation during epidemic years (Martin *et al.,* 2008).

Recently, technology has been used to improve yield in agriculture. Researchers have devised several solutions. Deep learning (DL), in the past years, has become widely known for its potential and advanced ability to efficiently process large numbers of images, yielding reliable outcomes (Panigrahi *et al.,* 2020). Deep Learning approaches are doing excellent in quantifying, categorizing, and identifying a variety of diseases and pest in cereals (Singh *et al.,* 2016). For instance, a study by Syarief *et al.* (2020) proposed a Convolutional Neural Network (CNN) model for identifying three (3) diseases in maize plants. In their experiment, seven CNN architectures namely: AlextNet, VGG16, VGG19, RessNet50, ResNet101, and GoogleNet, on a dataset of 200 leaf images, Inception-V3 was utilized to train the classifiers. AlexNet performed better with an accuracy of 93.5%. Furthermore, Sanga *et al.* (2020) presented a model that is based on deep CNN architectures (VGG16, ResNet18, ResNet50, ResNet152 with InceptionV3) for detecting black sigatoka and Fusarium wilt race 1, a fungal diseases threatening banana production. All models achieved high classification accuracy ranging from 95.41% to 99.2%. A similar study was conducted by Sibiya *et al.* (2021) on developing convolutional neural networks (CNNs) to detect severities in maize common rust disease. Early stage, middle stage, late stage, and healthy stage are the four categories used to train a VGG-16 network and when evaluated, it had a validation accuracy of 95.63 % and a testing accuracy of 89 %. However, the majority of the research that has been done has used image data that has not been properly processed, curated, or disseminated to a larger community of AI intellects. This work came up with imagery data of maize leaves obtained from the farm field. The data was used to develop a deep learning model for early diagnosis of diseases affecting maize specifically MSV and MLN. The model was deployed in a mobile app for farmers, agriculture experts, and plant pathologists to use in the early identification of diseases for early actions.

1.2 Statement of the Problem

Maize is a crucial crop in Tanzania, contributing significantly to the country's agricultural sector (Maiga, 2024). However, maize leaf diseases, such as Maize Streak Virus and Maize Lethal Necrosis, pose a severe threat to maize production, with the potential to reduce yield (Biswal *et al.,* 2022; Kiruwa *et al.,* 2020; Mahuku *et al.,* 2015; Shepherd *et al.,* 2010). Early detection of these diseases is crucial for implementing timely preventive measures and mitigating yield losses (Boddupalli *et al.,* 2020; Dhau *et al.,* 2018; Haque *et al.,* 2022). Traditional visual analysis methods for disease detection in crops are prone to errors, labor-intensive, and time-consuming. Moreover, these methods have been observed to identify diseases at a later stage, potentially leading to more harm to the crops (Gong *et al.,* 2023; Toseef *et al.,* 2018). They heavily rely on the expertise of farmers, plant pathologists, and agricultural experts. Additionally, the subjective nature of these methods can lead to inconsistent diagnoses among different experts. Furthermore, there have been no studies focusing on building a combined deep-learning model for the detection of MSV and MLN together, and there is no publicly available dataset with images of maize leaves infected by MSV and MLN. Therefore, there is a need for an accurate, efficient, and accessible method for early detection of maize leaf diseases in Tanzania, leveraging modern technologies to assist farmers, agriculture experts, and plant pathologists in identifying and addressing these diseases promptly.

1.3 Rationale of the Study

Maize diseases such as the Maize Streak Virus (MSV) and Maize Lethal Necrosis (MLN) pose a significant threat to maize production in Tanzania. These diseases can lead to severe yield losses, affecting the livelihoods of farmers who depend on maize as their primary source of income. Additionally, reduced maize production can lead to higher food prices and increased food insecurity. The selection of this research topic is justified by the urgent need for effective and accessible disease detection methods in agriculture. Traditional methods of disease detection are often time-consuming and require specialized knowledge. This research aims to provide a practical, cost-effective, and scalable solution to the problem by developing a deep learning model integrated into a mobile application. The use of advanced technology in agriculture is innovative and essential for modernizing farming practices and improving crop management. With the utilization of deep learning technology, farmers and agricultural professionals can effectively detect these diseases and enhance plant health, allowing them to increase productivity.

1.4 Research Objectives

1.4.1 Main Objective

The main objective of this study is to develop a deep-learning model for early detection of maize diseases in Tanzania.

1.4.2 Specific Objectives

The study aimed to achieve the following specific objectives:

(i) To identify requirements for developing a deep learning model

(ii) To develop the deep learning model for early detection of maize diseases.

(iii) To deploy the developed deep learning model on a mobile-based application.

(iv) To validate the developed mobile-based application for early detection of maize diseases.

1.5 Research Questions

The study intended to answer the following questions:

(i) What are the requirements for developing a deep learning model?

(ii) How will the deep learning model for early detection of maize diseases be developed?

(iii) How will the developed deep learning model be deployed on a mobile-based application?

(iv) How will the mobile-based application for early detection of maize diseases be validated?

1.6 Significance of the Study

Agriculture is a significant economic sector in Tanzania, accounting for roughly 29.1% of GDP, 67% of total employment, and serving as the primary food source, basic raw materials for industries, and foreign exchange profits (Ministry of Agriculture, 2016). This study aimed to develop a deep-learning model for the early detection of MSV and MLN diseases in maize. Early detection of diseases can allow farmers and extension officials to make well-informed decisions about how to manage the disease to increase maize output and prevent farmers from losing money every year. In so doing, the agricultural custodians being the Ministry of Agriculture, companies and other organizations will also benefit, hence raising the national Gross Domestic Product (GDP) and ensuring food security in the nation. Moreover, this research can add up to the achievement of sustainable development goal number 2 zero hunger by providing food and humanitarian relief and establishing sustainable food production.

1.7 Delineation of the Study

The study is conducted in Tanzania, particularly in regions where maize farming is predominant and maize diseases are commonly reported. Image datasets of maize plants were collected through physical observation by plant pathologists. The dataset includes images classified into four categories: Healthy, MLN, MSV, and WRONG. Two models, a Convolutional Neural Network (CNN) and a Vision Transformer (ViT), were developed and trained using the collected dataset. The performance of both models was evaluated to determine their accuracy in detecting the specified maize diseases. The study involves developing a mobile-based application that integrates the CNN model due to its smaller size and lower computational requirements than the ViT model. The mobile application was validated through field testing and feedback collected from farmers and agricultural experts using questionnaires. The dataset used for training the models may not encompass all possible variations of maize diseases, potentially affecting the generalizability of the model. The performance of the model in real-world conditions can be influenced by varying environmental factors such as lighting, background, and image quality. The mobile application was optimized for devices with limited computational resources, which may limit the complexity and accuracy of the deployed model compared to more powerful, server-based solutions. The effectiveness of the mobile application depends on its adoption by farmers and their willingness to integrate this technology into their farming practices. The study is specifically focused on detecting MSV and MLN in maize plants. Other diseases or plant health issues are not addressed. The mobile application is designed for Android devices, which may limit its accessibility to farmers using other types of mobile operating systems.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

This chapter starts by discussing important key terms in the context of the study. Then the chapter dives into a theoretical literature review to discuss the Maize leaf diseases, including brief details on maize streak virus and maize lethal necrosis. It proceeds to discuss deep learning also provides insights into deep learning models, particularly convolutional neural networks and vision transformers, and elaborates on the theories used to support this study. Further, the chapter dives into an empirical Literature Review to discuss the related works on the use of deep learning models developed for maize diseases and the use of mobile applications in plant disease detection. Finally, the chapter ends by highlighting a research gap and a conceptual framework.

2.2 Definition of Key Terms

(i) **Artificial intelligence (AI)** is the study and creation of computer systems that perform activities that would normally require human ability, such as sight, voice recognition, decision-making, and language processing (Dictionary, 2017).

(ii) **Machine learning** is the application and development of computer systems that can adjust without specific guidelines by studying and inferring from patterns in data using various algorithms methods (Batta, 2018).

(iii) **Deep learning (DL)** is a sophisticated machine learning area that has accomplished significant progress in a variety of fields (Yapici *et al.,* 2019). It allows the machines to automatically learn how to comprehend collected data objects through feature extraction and representation for classification, localization, and recognition (Singh *et al.,* 2016).

(iv) **Computer vision** is a branch of AI that allows users and systems to extract useful data from online photos, videos, as well as other sensory inputs and act or offer suggestions based on those facts (Engn *et al.,* 2003).

(v) **Image classification** is a basic activity that entails attempting to comprehend a complete picture. The goal is to give the image a label so that it can be classified. The term "image classification" is widely used to explain pictures that include just one object and is evaluated (Wang *et al.,* 2019).

(vi) **Convolutional neural network** is a sort of image classification and processing artificial neural network that is created specially to analyze pixels (Margaret Rouse, 2018).

(vii) **Vision Transformer or ViT** is an image classification model that makes use of a Transformer-like structure across image patches (Dosovitskiy *et al.,* 2020).

(viii) **Training data** is the data used to build a machine-learning model (Cappi *et al.,* 2021).

(ix) **Test data** is the data used to evaluate the effectiveness of the method you're using to train the model, such as correctness or effectiveness (Cappi *et al.,* 2021).

(x) **Validation data** is the data utilized to evaluate and inform the method and parameter choices for the model you're creating (Cappi *et al.,* 2021).

(xi) **Smart Disease Detector** is a mobile application developed for the early detection of MSV and MLN.

2.3 Theoretical Literature Review

2.3.1 Technology Acceptance Model

This is a theoretical framework that explains how users came to accept and use a technology. It is widely used to predict user acceptance and use of technology (Mugo *et al.,* 2017). The theory of technology acceptance model has been used in this study to examine farmers' and all agricultural stakeholders' attitudes and behavioral intentions about using the proposed tool. This has been addressed during the stakeholder validation of the mobile application in the field.

2.3.2 Diffusion of Innovation Theory

This is a theoretical framework that describes the rate and manner in which a new idea, solution, or tools are adopted in a society (Ismail, 2016). The study has applied the diffusion of innovation theory to determine how quickly consumers are accepting the new tool during the field validation of the mobile application with agricultural stakeholders

2.3.3 Convolutional Neural Network and Vision Transformer

(a) Convolutional Neural Network

For the past several years, deep learning achievement on computer vision tasks has highly depended on Convolutional Neural Networks (CNN) (Raghu *et al.,* 2021). Additionally, CNN has been prevailing in the domain of computer vision as a groundwork for various applications such as image classification where the aim is to categorize an image into one of several pre-defined groups (Atila *et al.,* 2021; Chen *et al.,* 2021; Darwish *et al.,* 2020; Haque *et al.,* 2022; Liu *et al.,* 2021; Sibiya *et al.,* 2019a; Syarief *et al.,* 2020), object detection where the goal is to locate and classify features in an image (Liu *et al.,* 2021; Maxwell *et al.,* 2021; Roy *et al.,* 2022; Zhang *et al.,* 2020) and image segmentation (Gayatri *et al.,* 2021; Liu *et al.,* 2021; Loyani *et al.,* 2021; Maxwell *et al.,* 2021; Sibiya *et al.,* 2021). Convolutional neural network architecture consists of components such as a convolutional layer, a pooling layer, and a fully connected layer, activation functions (Bharali *et al.,* 2019; Francis *et al.,* 2019; Jasim *et al.,* 2020). The input to a CNN is often an image, and the network is made up of convolutional layers, pooling layers, and fully connected layers. The convolutional layers process the input image through a set of learnable filters, creating a set of feature maps (Voulodimos *et al*., 2018). The pooling layers down sample the feature maps to reduce their spatial size, whereas the fully connected layers classify the features into separate classes. Convolutional neural network architecture is demonstrated in Fig. 1.

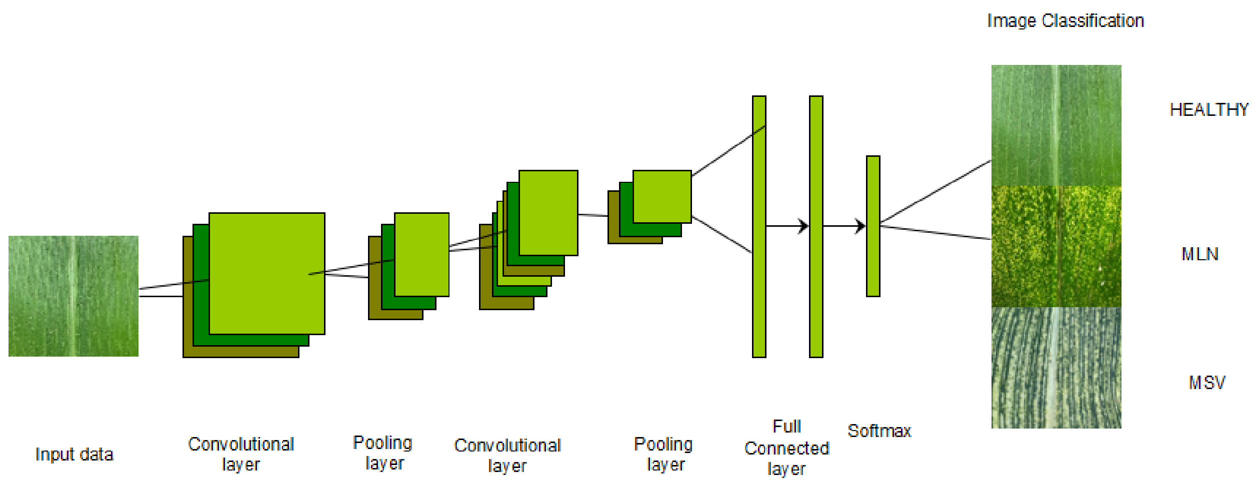


Figure 1: Convolutional neural network architecture

(b) Vision Transformer

Transformer is a novel neural network. It primarily employs the self-attention mechanism to obtain the basic attributes and provide correlations between various features (Vaswani *et al.,* 2017). It has a high potential for widespread use in applications that utilize AI (Han *et al.,* 2021, 2023). Transformers are commonly applied in Natural Language Processing (NLP), the initially reported transformer was based on an attention approach for machine translation and English constituency parsing tasks (Vaswani *et al.,* 2017). Another set of transformers frequently utilized in natural language processing consists of models like Bidirectional Encoder Representations from Transformer (BERT). In addition, BERT pre-trains on unlabeled text and generates deep bidirectional representations, concurrently considering both the left and right context across all layers (Devlin *et al.,* 2019). Meanwhile Generative Pre-trained Transformer 3 (GPT-3) uses deep learning to generate huge amounts of text that appear as though they were composed by people (Brown *et al.,* 2020). As a result of major achievements in transformers on NLP, there was a motivation to look at the use of transformers for visual tasks.

The Vision Transformer (ViT), is an image classification model that implements a Transformer-like architecture over image patches. An image is divided into fixed-size patches, which are then linearly embedded. Position embeddings are then added, and the resulting vector sequence is fed into a standard Transformer encoder. The standard approach of adding an extra learnable "classification token" to the sequence is used to perform classification (Dosovitskiy *et al.,* 2020; Vaswani *et al.,* 2017).

The sequence of the 1D input array is passed to the transformer structure. To process 2D images, 2D patches are extracted from them first, and then they are reshaped to create 1D arrays, which are suitable for ViT structure. They are added to the positional encoder to finish preparing the patch embedding for the next layer.

The positional encoder aids the network in remembering the relative position of the patches in relation to one another. The inputs are then normalized with the normalization layer before entering the transformer block. The multi-head attention layer is the most important aspect of this block. The multi-head attention layer calculates weights to assign higher values to the more important areas. In other words, it focuses network attention on the most important parts. The multi-head attention layer's output is a linear combination of each head (Borhani *et al.,* 2022). Figure 2 shows ViT architecture that was inspired by Vaswani *et al.* (2017).

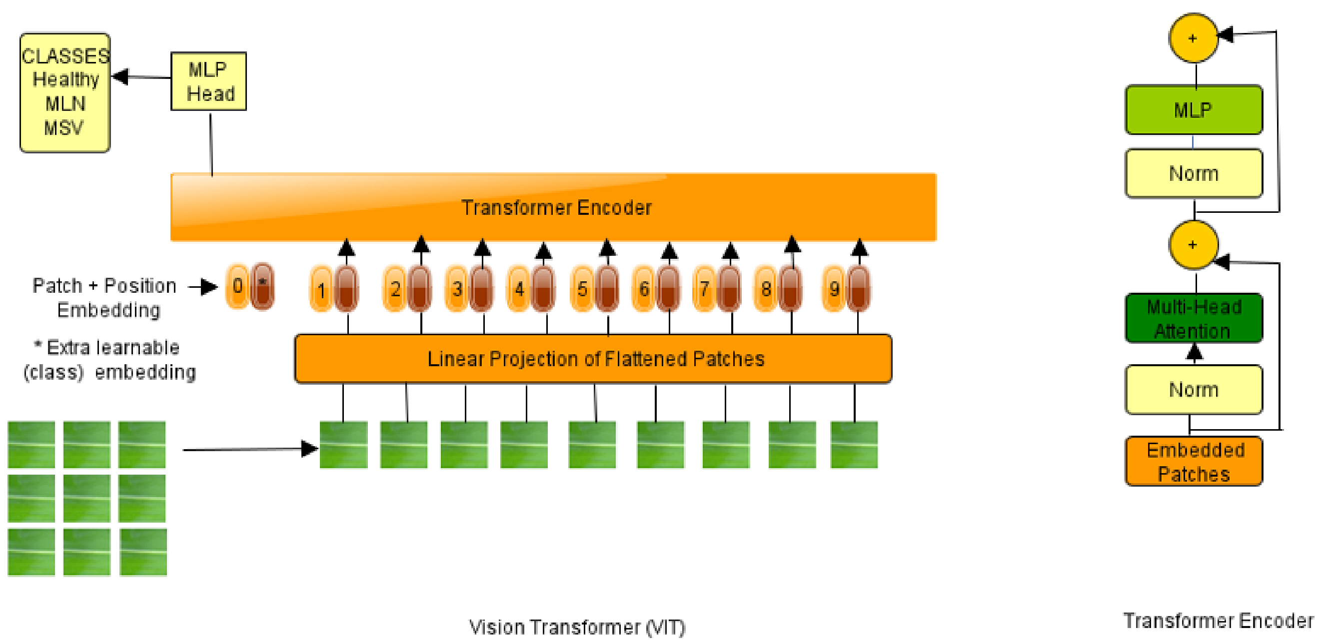


Figure 2: Vision transformer architecture

2.4 Empirical Literature Review

Farmers worldwide are striving to produce healthy crops for better production and improve their lives in every way. Scientists and researchers are also concentrating their efforts on identifying factors affecting plants. Several ways have been employed to identify these factors and severity to provide effective solutions. Deep learning is one of the proposed techniques that has shown significant promise; this approach has been used by numerous researchers to detect and quantify plant diseases (Singh *et al.,* 2016).

2.4.1 Maize Leaf Diseases

Numerous diseases affect maize leaves, encompassing conditions such as Anthracnose, Cercospora Leaf Spot (commonly referred to as Gray Leaf Spot), Common Corn Rust, Downy Mildew, Northern Leaf Blight, Southern Corn Leaf Blight, Bacterial Leaf Blight/Stripe, Bacterial Leaf Streak Diseases, Goss's Bacterial Blight, Holcus Spot, Stewart's Wilt, Maize Dwarf Mosaic, Maize Lethal Necrosis, and Maize Streak Virus (Cimmyt *et al.,* 2004). In the African context, Maize Lethal Necrosis and Maize Streak virus are reported to be the dominant threat, moreover the most serious viral disease of maize (Martin *et al.,* 2008; Monjane *et al.,* 2020). These diseases are the most significant and prevalent viral disease that diminishes maize production and poses a threat to food security in Africa (Shepherd *et al.,* 2010; Tembo *et al.,* 2020). These diseases were first reported in East Africa and have recently widely spread to many other African countries (Cimmyt *et al.,* 2004).

(i) Maize Streak Virus

Maize streak virus disease is a common disease in maize plants caused by a geminivirus (Ilyas *et al.,* 2014; Tembo *et al.,* 2020). It is continuously spread by a leafhopper belonging to the cicadulina genus (Family Cicadellidae, Order Hemiptera), and this insect acts as a mobile carrier that normally transmits the virus for most of its life after feeding on an infected plant. The vulnerable kinds of maize are exposed to the virus through the insect leafhopper cicadulina spp, which is commonly found in fields where maize is planted later in the season (Monjane *et al.,* 2020). The initial signs of the disease manifest approximately one week after infection. These symptoms include tiny, circular spots scattered across the newest leaves. As the plant continues to grow, the number of these spots increases, and they expand along the leaf veins. Eventually, the spots become more concentrated near the base of the leaves and are particularly noticeable on the youngest ones. When the leaves reach full elongation, they exhibit a yellowing effect with discontinuous yellow streaks along the veins, creating a stark contrast with the usual dark green hue of healthy foliage. In severe cases of infection, the plants experience stunted growth, and they may either perish prematurely or fail to produce any fruit. The virus responsible for this disease finds refuge in various cereal crops and wild grasses, serving as its reservoir, and is transmitted by specific vectors (Cimmyt *et al.,* 2004; Emeraghi *et al.,* 2021).

(ii) Maize Lethal Necrosis

Maize lethal necrosis (MLN) is a devastating maize illness caused by the co-infection of two viruses: Maize Chlorotic Mottle Virus (MCMV) and a combination of either Maize Dwarf Mosaic Virus (MDMV) or Wheat Streak Mosaic Virus (WSMV). If only MDMV and WSMV would coexist, no lethal necrosis would develop. Plants that have been infected tend to have stunted growth, the foliage exhibits chlorosis leading to plants demise around the period of flowering. Plants infected during the early stages of growth do not develop ears (Cimmyt *et al.,* 2004; Stewart, 2022). The disease manifests throughout all stages of crop growth. Its primary symptom is the emergence of chlorotic spotting on the leaves, originating from the lower part and progressing upwards. Additionally, the leaves display necrosis along the edges, which subsequently extends to the midrib, resulting in the entire leaf drying up. Necrosis in the young leaves at the center gives rise to the 'dead heart' symptom. Other indications include premature plant demise, shorter male inflorescences with fewer spikes, and/or stunted, deformed, and partially filled ears. The virus is primarily transmitted by vectors, such as maize thrips, aphids, rootworms, and leaf beetles, as well as through infected seeds. Kenya was the first to report this disease on the African continent in 2011 (Mahuku *et al.,* 2015). The disease then spreads to neighbouring nations like Tanzania, Uganda, and South Sudan (Adams *et al.,* 2014).

2.4.2 Deep Learning Models for Maize Diseases Detection

Plant diseases continue to be a threat to the global food supply, causing enormous losses in food and revenue, if proper control measures are not taken, plant leaf diseases may present a severe danger to global food security (Sibiya *et al.,* 2019b). This section reviews related works concerning DL done by various researchers globally, in Africa and Tanzania, and highlights the research gap.

A convolutional neural network deep learning model was developed to diagnose images of plant leaves that are healthy and unhealthy. A total open database of 87 848 images with 25 distinct plants containing 58 distinct categories with healthy and unhealthy images was trained using five model architectures AlexNet, AlexNetOWTBn, GoogLeNet, Overfeat, and VGG. In addition, VGG resulted as the most performing architecture with a success rate of 99.53% in detecting the plant's diseases. Implementation was done using the Torch71 machine learning computational framework, which uses the LuaJIT programming language. The model's exceptionally excellent performance makes it suitable as a vital early alert or advising tool (Ferentinos, 2018). This study was conducted in Athens, Greece, and was done to detect many plant diseases and not specifically for the detection of maize streak virus and maize lethal necrosis.

Another deep-learning model was developed to detect maize diseases in Indonesia. The study used a classification approach to detect 3 diseases cercospora, northern leaf blight, and common rust using CNN. Support vector machine, k-nearest neighbour, and decision tree were used to classify maize leaf images, and seven other CNN architectures were used to analyze maize leaf images. Architectures used include ResNet50, GoogleNet, VGG19, AlexNet, Inception-V3, VGG16, ResNet110 and VGG19. Data consisted of 200 images which were divided into 4 classes, 50 images per class with size of size of 256x256 pixels. Even though AlexNet and SVM were the best methods for feature extraction and image classification of maize leaf diseases. This study used fewer samples (200 images) which were collected in Asia (Syarief *et al.,* 2020).

Also, a Mobile-DANet model was developed to identify 8 maize crop diseases, gibberella ear rot, maize eyespot, crazy top, grey leaf spot, Goss's bacterial wilt, common smut, phaeosphaeria spot, and southern rust. Except for several samples, the results of the Mobile-DANet model demonstrated that the majority of images and maize diseases were correctly identified. Mobile-DANet correctly detected samples with phaeosphaeria spots with a probability of 0.71. Similarly, the model accurately detects samples of gibberella ear rot and southern rust disease, with probabilities of 0.83 and 0.93, respectively. China served as the study's location and this study focused on other maize images other than MSV and MLN. The model employed by the study is Mobile-DANet (Chen *et al.,* 2021).

Furthermore, another study from India proposed a deep convolutional neural network to detect healthy and disease images of maize leaves. The dataset contained 5939 images of maize leaves, the dataset consisted of images of three diseases Maydis leaf blight (MLB), Sheath blight (BLSB), Turcicum leaf blight (TLB), and Banded leaf as well as healthy maize leaves. Using the Inception-v3 network structure, three different models were developed using the normal training procedure. The top-performing model demonstrated an astounding 95.99% accuracy in the overall classification and an average recall of 95.96% when tested on a separate test dataset (Haque *et al.,* 2022).

In Cairo Egypt, a classification model for the identification of Common rust, Northern leaf blight healthy maize leaves, and gray leaf spots was developed. To identify plant diseases, an ensemble model made up of two pre-trained convolutional neural networks, VGG19 and VGG16 was employed, to distinguish between the leaves in healthy photos and the leaves in unhealthy ones. The outcomes show how well the suggested strategy works, outperforming alternative methods by 96.7% for VGG19. Even though the created model performed well, this study struggled with the categorization of unbalanced data, and the dataset it employed lacked sufficient images to properly train CNNs that were created from scratch (Darwish *et al.,* 2020).

A model for recognition of common rust (Puccinia sorghi), gray leaf spot (Cercospora), and northern corn leaf blight (Exserohilum) out of healthy leaves was developed due to these diseases impact on the majority of the maize plantations in South Africa. Neuroph was used for training convolution neural networks, to recognize and classify images of maize. In addition, CNN was quite correct in identifying these diseases, 99.9% accuracy rate for northern corn leaf blight, a 91% accuracy rating for gray leaf spot an 87% accuracy rate for common rust, and a 93.5% accuracy rate for healthy. The research was restricted to the neuroph framework of the Java neural network, which is an integrated environment for developing and deploying neural networks to Java programs, despite the model's strong performance (Sibiya *et al.,* 2019).

A similar study was conducted by Sibiya *et al.* (2021) to develop a CNN deep learning model. The amount of diseased leaf area was calculated using segmentation by threshold on diseased images of leaves of maize impacted by the Common Rust disease. This information was used to create ambiguous decision guidelines in assigning Common Rust images to severity groups with images created using this proposed approach. The VGG-16 network, trained with images made using this suggested method, achieved a testing accuracy of 89% and a validation accuracy of 95.63% when tested on photos of the Common Rust illness in 4 stages of severity (Early stage, Middle stage, Late Stage, and Healthy stage). Despite the good performance results of the developed model, this study was only limited to the image segmentation approach which tends to partition a digital image into multiple segments. Furthermore, the study used CNN architecture which lacked a detailed description.

Also, Arnaud *et al.* (2022) from Kenya came up with a deep learning model to examine in contrast 6 convolutional neural network architectures. Transfer learning was employed for model training, architectures used include EfficientNet b7, VGG19, SqueezeNet, GoogleNet, AlexNet, and DenseNet. The study analyzed four hyperparameters that are batch size, learning rate, number of epochs, and optimizers. An open-source dataset with 4082 images were used. DenseNet121 outperformed the rest of them by achieving an accuracy of 98.34% and a f1-score of 97.37%. The DenseNet121 was trained with batch 32, a learning rate of 0.01, and the optimizer used was Stochastic Gradient Descent (SGD).

2.4.3 Use of Mobile Apps in the Detection of Crop Diseases

Intelligent mobile plant disease diagnostic system has become beneficial due to its use in the early diagnosis and identification of plant illnesses utilizing leave images, even when competent and adequate experts are unavailable in such situations (Adedoja *et al.,* 2022). This section reviews related works concerning mobile apps for disease detection done by various researchers.

In Nigeria a mobile application for the detection of cassava diseases was developed, the app was specifically for the detection of Cassava Bacterial Blight Disease and Cassava Mosaic Disease. In the implementation of a mobile application, two machine-learning models were deployed. The two models were selected because of the best performance they had achieved. The model responsible for health diagnosis had an accuracy of 83.9% and the model for disease detection had an accuracy of 61.6%.

To assist smallholder farmers in Tanzania with the early diagnosis of Fusarium wilt race 1 and black Sigatoka banana illnesses, a smartphone application was developed. Using a dataset of 3000 photos of banana leaves. A deep learning model was trained on Inceptionv3 and Resnet152 Convolution Neural Network architectures. While the Inceptionv3 scored a 95.41% accuracy, the Resnet152 achieved a 99.2% accuracy. As Inceptionv3 requires less memory than Resnet152, it was chosen for deployment via Android smartphones. The mobile application identified the two diseases in real life with a 99% confidence level of the leaf area that was collected. This research indicates that using a disease detection tool can help smallholder farmers increase their banana yield (Sanga *et al.,* 2020).

Another smartphone app that uses deep learning to identify Northern Corn Leaf Blight crop diseases in maize was developed. The application employs a convolutional neural network trained on a dataset of images of Northern Corn Leaf Blight and healthy maize plant leaves. The trained model was deployed on a smartphone app that can be used in the field to detect maize crop disease on the fly. The approach seeks to allow for early diagnosis of plant diseases and proactive corrective procedures to avoid major production loss. The model's classification results demonstrate great accuracy, with an F1 score of 0.99 and a recall of 1.00. Farmers may use the app to identify diseases in maize crops at a low cost and on the go (Richey *et al.,* 2020).

Moreover, in Tanzania, a mobile application based on deep learning titled Tuta Segmenter was developed for the detection of Tuta Absoluta disease in tomato plant leaves. A dataset of tomato leaf images was employed for training the CNN model. The model performed well, with an IoU of 78.60% and a dice coefficient of 82.86%. The model was then deployed in Tuta Segmenter, a mobile application that allows users to capture or upload photos of tomato plants and detect and segment Tuta mines. The tool delivers real-time pest diagnostics and early detection, allowing farmers to conduct proper disease control measures (Loyani *et al.,* 2021).

Furthermore, a mobile application for plant disease detection was developed. The application utilizes NASNet-Mobile, a lightweight convolutional neural network (CNN), for plant disease detection specifically on plant leaf images. The mobile application was developed for both Android and iOS smartphones. The app functions over a web service, where leaf images are received and disease detection is performed by the NASNet-Mobile CNN model. The suggested

NASNet-Mobile CNN model plant disease detection achieved an accuracy of 99.31% (Adedoja *et al.,* 2022).

2.5 Research Gap

Various techniques for detecting plant diseases have been proposed in general. These techniques have shown good performance; however, there have been no studies focusing on building a combined deep-learning model for the detection of MSV and MLN together, and there is no publicly available dataset with images of maize leaves infected by MSV and MLN. As a result, this study developed a combined deep learning model for MSV and MLN detection based on images collected directly from the field, allowing the model to be trained with real data. The dataset was made available in open source to the research community for future studies on MLN and MSV infections. Moreover, this study introduces an approach that enhances the effectiveness and efficiency of MSV and MLN diseases in maize. Prior studies have developed methods for disease detection in plants using image datasets and leveraging Transfer Learning that are prone to overfitting and generalization. This research offers a more efficient, automated, and accurate solution by developing a CNN architecture trained on our maize dataset from scratch.

2.6 Research Conceptual Framework

To help farmers avoid the inconveniences of manual disease detection and to have a user-friendly suggestion, this study has used deep learning techniques to develop an early detection tool for maize diseases with different combinations of image collection methods, image processing methods, and image classification methods. The rapid advancement of mobile device technology has given people new ways to detect crop diseases rather than the use of human experience. Therefore, the developed tool will allow farmers, agricultural experts, and plant pathologists to early detect MLN and MSV diseases for early control and management of these diseases.

2.6.1 Description of the Research Framework

Images of healthy and diseased maize leaves were acquired from farms. The image datasets were pre-processed and divided into training and testing. The model was then trained and tested to test the results of the created model. The best model was then deployed in a mobile application to allow smallholder farmers to easily interact with the developed solution. A description of the conceptual framework is presented in Fig. 3.

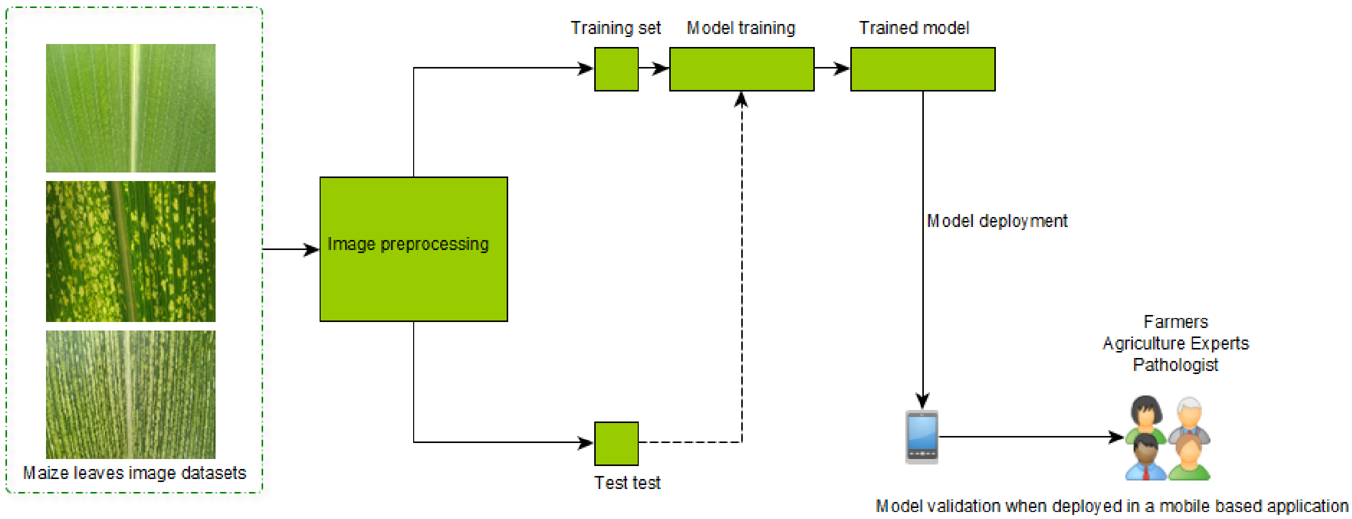


Figure 3: Research conceptual framework

CHAPTER THREE

MATERIALS AND METHODS

3.1 Introduction

This section encompasses the material and methods used to conduct this study. It presents the study area, research design, research methods, the target population, sampling techniques sampling size, data collection methods, data analysis, model and system development, and lastly the ethical consideration.

3.2 Study Area

This study was conducted in the Arusha, Kilimanjaro, and Manyara regions situated in the Northern part of Tanzania. The selection of these areas was based on maize availability and disease prevalence (Card *et al*., 2020). This study focused on various maize farmers across the country. The selected study areas maximize the reliability and applicability of the developed model, the resulting developed model can easily be used in any farm by any farmer or agriculture expert to easily detect diseases early enough. The regions in which the study was conducted are shown in Fig. 4 on the map of Tanzania.

Figure 4: The map of Tanzania with highlighted study areas

3.3 Research Design

Design Science Research (DSR) approach was used to govern this study. The approach was chosen because of its ability to address problems with an emphasis on the development and study of technological objects. Since the major goal of this study was to develop a deep learning model and early detection tool for maize diseases in Tanzania, DSR provided the required foundation for putting the developed study into action. The research was conducted using Vaishnavi and Kuechler's iterative design cycle (2015), as shown in Fig. 5.

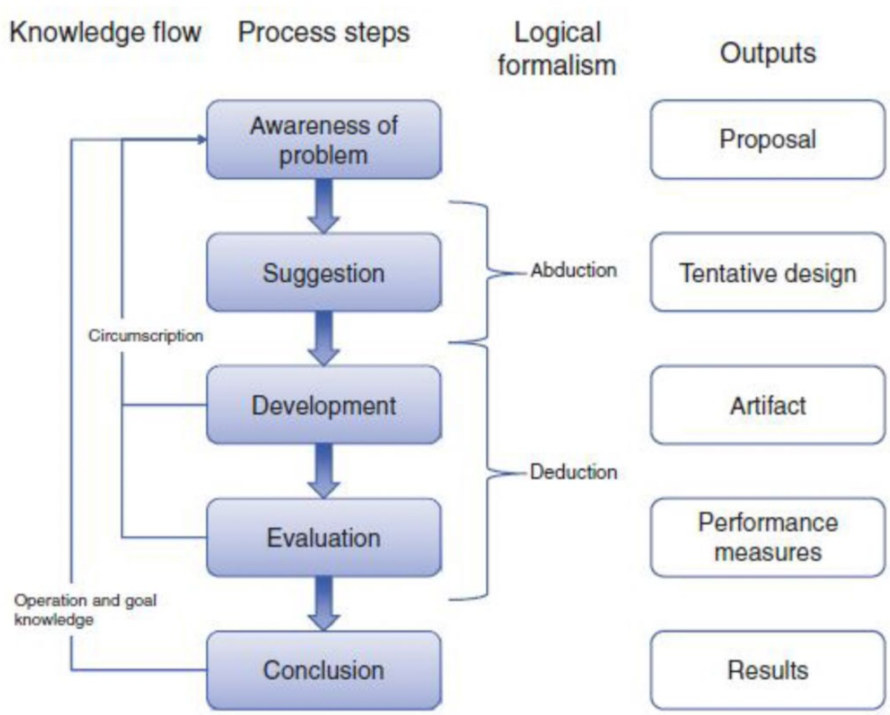


Figure 5: Design science research model

3.4 Research Methods

This study has employed a quantitative research method. The data for gathering requirements for system development and system validation was collected using questionnaires which fall under quantitative methodology.

3.5 Target Population

The target population for this study was all stakeholders of agriculture including farmers, agricultural experts, and plant pathologists.

3.6 Sampling Technique and Sampling Size

3.6.1 Sample Size

The sample used for this study consisted of 27 588 imagery datasets. The data was grouped into four different classes: Healthy, MLN, MSV, and WRONG. Healthy included image datasets taken from the maize leaves that are not diseased. The Maize Lethal Necrosis image dataset is taken from images that are affected by Maize Lethal Necrosis, the MSV image dataset is taken from images that are affected by Maize Streak Virus, WRONG images were taken from a variety of things such as people images, flowers, trees, mountains, all kinds of nature and different devices. The sample included different images to ensure that the model learn to distinguish between different images thereby improving its ability to accurately identify and detect the specified diseases.

3.6.2 Sampling Technique

A non-random sampling technique was used to govern the study. The imagery dataset was selected based on the predefined classes. The primary aim of the study was to develop a deep-learning model for the early detection of maize diseases. Therefore, imagery datasets were collected specifically from maize leaves affected by MLN, MSV, and Healthy maize leaves. The WRONG class was added to capture images from the surrounding environment of the maize farm, enhancing the model's ability to differentiate between target and non-target images.

3.7 Data Collection Methods

A tool called Open Data Kit (ODK) was developed to assist in the collection of the maize leaves' imagery datasets for model development. The data for gathering requirements for system development and system validation were obtained through questionnaires.

3.8 Data Analysis

Maize leaf images were labeled based on the appropriate category (healthy, infected by MLN, infected by MSV, and WRONG). Data cleaning and preprocessing were conducted and the dataset was divided into training and test sets. The training set was used to train the model and the test set was used to evaluate the performance of the developed model. The well-performed model was then deployed on a mobile application.

3.8.1 Data Cleaning and Preprocessing

This is a very crucial stage, where all the collected data is cleaned and ensured it is free of any erroneous or fraudulent information. This process normally uses various tools and software (Lee *et al.,* 2021). In data cleaning the following tasks were applied:

(i) Removing Duplicates

In this step, duplicate images from the three classes, Healthy, MSV, and MLN were removed using the VisiPics tool (Arora *et al.,* 2016). Table 1 displays all the images after and before the duplicates were deleted.

Table 1: Number of images before and after duplicates

S/N	Classes	Numbers of images before removing duplicates	Number of images after removing duplicates
1.	Healthy	9145	8615
2.	MLN	8604	8578
3.	MSV	9911	9720
4.	WRONG	675	675
Total		28335	27588

(ii) Labeling

The labeling process was done with the help of a tool named bulk rename utility to fasten the labeling process. Image labeling was done by naming the imagery data to the corresponding available classes. These images were ensured to be saved in a jpg format to work during model development.

(iii) Resizing

The image dataset was resized to uniform pixels using the Keras library with TensorFlow. Imagery data used to train and test the CNN model were resized to a uniform pixel of 256*256 and images that were used to train and test the Vit model were resized to 200*200. But before resizing the dataset consisted of images in JPG format, the images were in different sizes.

(iv) Image Augmentation

This is a procedure that is used to add the number of datasets by producing new versions of the original data. The model can learn to be more resistant to changes in the input data and to generalize to new, previously unseen data. The conversion performed to the original data does not modify their meaning, but instead generates more data that is added to the training dataset. Data augmentation is crucial to avoid overfitting when you have a small dataset for training the model, this is also done using the Keras library with TensorFlow. The first CNN model used a small dataset of 1500 images, the training data set was increased by image random rotation and image random flipping vertically and horizontally. Image flipping involves flipping an image randomly either horizontally or vertically and image rotation involves rounding an image to a certain level such as 90 degrees to create a new version of the image.

(v) Experimental Setup

The study's experiment was carried out using a Windows 10 computer equipped with an Intel(R) Core ™i5-4200U CPU @ 1.60GHz and 2.30GHz, 8GB of installed RAM, and a 64-bit operating system. In addition, GPU was the hardware accelerator having Python 3 as the run-time. Google Collab was employed for the models online. TensorFlow was used on the backend in conjunction with the Keras library throughout the development. Python was the selected language because of its ability to provide a variety of freely available machine-learning libraries.

3.9 Model Development

3.9.1 Convolutional Neural Network Model 1

A Convolutional Neural Network (CNN) model was meticulously developed from scratch, utilizing a substantial dataset of 1500 samples across three distinct classes (HEALTHY, MLN, and MSV). The dataset was split 80% for training, 10% for validation, and 10% for testing. 10% of validation data was used at the end of every epoch to do validation. 10% of test data was used after the 50th epoch to measure the accuracy of the model and also test the performance of the model before deploying. A sequential model was used that defined 6 convolutional layers each followed by a max pooling layer. The first convolutional layer had 32 filters, the second layer up to the sixth layer had 64 filters each, followed by flatten layer and then a dense layer with 64 neurons. All convolutional layers used a Rectified Linear Unit (ReLu) as an activation function. The output-dense layer had 3 neurons which is the number of our classes and softmax activation. The images were rescaled by 1.0/255 and they were resized to 512 x 512 pixels. From a dataset of 1500 images, 1200 images were used for training and 150 images for validation, 150 images for the test set for the three classes.

3.9.2 Convolutional Neural Network Model 2

Another CNN model was developed from scratch using a total amount of 27 588 datasets belonging to four classes (HEALTHY, MLN, MSV, and WRONG). The dataset was split into 80% for the training set, and 20% for the testing set. Because of the large number of images, the model was trained in four groups of batches where the output weights that were utilized in training the first batch were employed as input in training the second batch, then the same thing for the third and the fourth batch. For the first three batches, each contained 6000 datasets, the datasets were split into a 4800 training set and a 1200 test set for each batch Healthy, MSV, and MLN, but for WRONG images train set contained 540 images and the test set 135 images maintaining an 80:20 ratio for each class. For the fourth batch, the model was trained using the remaining 8913 datasets. The dataset was again split into an 80:20 ratio for the training set and the test set, resulting in 7131 samples for training and 1782 samples for testing for Healthy, MSV, and MLN, where the WRONG image class number remained the same.

A sequential model was employed in this implementation that defined 5 convolutional layers and each layer was followed by a max pooling layer. The first convolution layer had 16 filters; the second convolution layer had 32 filters, and the third up to the fifth layer had 64 filters. These were then followed by a flattened layer and then a dense layer with 512 neurons. Rectified Linear Unit (ReLu) was employed as an activation function in all convolutional layers. The number of the classes was represented by the output dense layer which had 4 neurons with a softmax activation function. The images were rescaled by 1.0/255 and they were resized to 256 x 256 pixels. When training the models in this study, different hyper-parameters were used, which include the following:

(i) Epoch

It refers to the sum of the number of repetitions of all training data in one round for training the model.

(ii) Batch size

It is defined as the number of training instances used in one iteration; it represents the number of samples required to update the model parameters. The number of batch sizes can vary from 16, 32, and 64 up to 128 depending on the parameters such as dataset quantity, model complexity, and computational resources available.

(iii) Optimizer

This is a function or method that modifies neural network parameters such as weights and learning rates during training. Examples of optimizers include Stochastic gradient descent, Adam, Root Mean Square Propagation, Adaptive gradient, and others.

(iv) Learning rate

It is a hyperparameter that governs the number of steps or pace at which the parameters of a model are updated throughout training. Hyperparameters used for training CNN model 1 and CNN model 2 with their values are shown in Table 2 and Table 3:

Table 2: Hyperparameters used for training CNN model 1

S/N	Parameter	Value
1.	Epoch	50
2.	Batch size	32
3.	Optimizer	Adam
4.	Losses	Categorical_crossentropy
5.	Metric	Accuracy

Table 3: Hyperparameters Used for Training CNN Model 2

S/N	Parameter	Value
1.	Epoch	50
2.	Batch size	32
	Steps per epoch	167
3.	Optimizers	Adam
4.	Losses	Categorical_crossentropy
5.	Metrics	Accuracy, Precision, Recall, F-measure

3.9.3 Evaluation of the CNN Models

Model evaluation refers to the process of examining the performance and quality of a model. It entails determining how well the model generalizes to new, previously unknown data and how well it handles the specific problem for which it was trained. There are numerous typical strategies for model evaluation that are being utilized depending on the type of problem and the form of the data. In this study, four metrics were used to evaluate the models' performance which include accuracy, precision, recall, and f-measure. During model training from the 1st epoch to the 50th epoch the following metrics were observed:

(i) Accuracy

Accuracy is the degree of correctness or precision in measurements, data, or information. It measures how close a value or result is to the true or expected value (Maxwell *et al.,* 2021). The formula for obtaining accuracy is shown in Equation 1.

$$Accuracy = \frac{TP+FN}{TP+TN+FP+FN} \tag{1}$$

Where:

True Positives (TP) cases that are correctly predicted as positive.

False Negatives (FN) cases that are predicted as negative but were positive.

True Negatives (TN) cases that were correctly predicted as negative.

False Positives (FP) cases that are predicted as positive but were negative.

(ii) Training Accuracy

This refers to the level of accuracy attained by a model when training on training data. It represents the extent to which the model fits or predicts the training data to which it has been subjected. It was used to show how well the model is learning the patterns in the training data.

(iii) Training Loss

This is also referred to as training error on training data during the training phase. It was used to minimize training loss during the training phase.

(iv) Validation Accuracy

This is the level of accuracy accomplished by a model using a different validation dataset. It was employed to denote how well the model performs on previously unseen data that it was not exposed to during model training.

(v) Validation Loss

This is the error that was returned after passing the validation set of data through the trained data. In model training, this loss is monitored to help in understanding how well the model generalizes to new unseen data.

(vi) Precision

It refers to the division of correct obtained values (TP) over the total of correct obtained values (TP) and incorrect obtained values (FP) as seen in Equation 2. It is concerned with the accuracy of correct obtained value predictions and measures how well the model avoids incorrect obtained value.

$$Precision = \frac{TP}{TP+FP} \tag{2}$$

Where:

True Positives (TP) cases that are correctly predicted as positive.

False Positives (FP) cases that are predicted as positive but were negative.

(vii) Recall

This is the proportion of true positives (TP) to the total number of true positives (TP) and false negatives (FN) as seen in Equation 3.

$$Recall = \frac{TP}{TP+FN} \tag{3}$$

Where:

True Positives (TP) cases that are correctly predicted as positive.

False Negatives (FN) cases that are predicted as negative but were positive.

(viii) F-measure

This combines precision and recall into one measure that strikes a balance between the two (Equation 4).

$$F - measure = 2\frac{Precision*Recall}{Precision+Recall} \quad (4)$$

3.9.4 Vision Transformer Model

The Vision Transformer (ViT) model was developed using a dataset consisting of a total of 6675 samples belonging to four classes (HEALTHY, MLN, MSV and WRONG). The images were resized to a uniform size of 200x200 pixels. The ViT model architecture comprises patch embedding, positional embedding, 12 transformer layers, and a classification head. Each transformer layer includes 12 attention heads in the multi-head attention mechanism, and the feed-forward neural networks in the transformer have a dimensionality of 3072. Each patch in the image has a size of 25, and the number of output classes is 3, corresponding to the number of classes in the dataset. The hidden dimensionality of the transformer model is 768, and a dropout rate of 0.1 was applied. The activation function used in this model was the Gaussian Error Linear Unit (GELU). Hyperparameters used for training the ViT model as shown in Table 4.

Table 4: Hyperparameters used for training ViT model

S/N	Parameters	Value
1.	Epoch	50
	Steps per epoch	154
2.	Batch size	32
3.	Optimizer	Adam
4.	Metric	Accuracy
	Learning rate	0.0001
5.	Losses	categorical-Crossentropy

3.10 System Development Methodology

3.10.1 System Development Approach

The technique employed by the study was Agile software development methodology based on the Extreme programming (XP) agile approach. When compared to other Agile techniques, the XP Agile approach makes it easier to handle changes within iterations and saves time and money in development (Gren *et al.,* 2020).

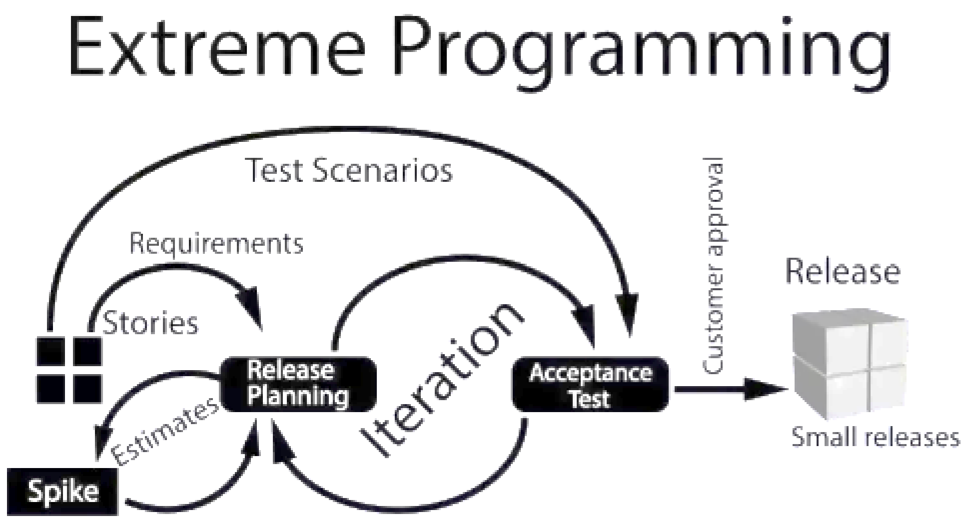


Figure 6: A diagram of extreme programming methodology

Source: https://chef-de-projet.fr/extreme-programming/

3.10.2 System Requirements

The main goal of this phase is to gather the requirements and services that the mobile application should provide. It entails recognizing the system users, understanding what they do and wish to accomplish with the system, and being aware of their environment.

3.10.3 System Design

The design diagram for the development of mobile-based applications was visualized using Unified Modeling Language diagrams (UML) specifically use-case diagrams, activity diagrams, and sequence diagrams.

(i) Use Case Diagram

A use case diagram is normally used to illustrate the interaction between users and the system, showing how users interact with the system's components their inputs and expectations, and the steps the system must take to accomplish its goals. It illustrates the actors, use cases, and their interactions (Aleryani *et al.,* 2016). Figure 7 depicts the use case diagram for the mobile-based application and Table 5 presents the descriptions of the use cases.

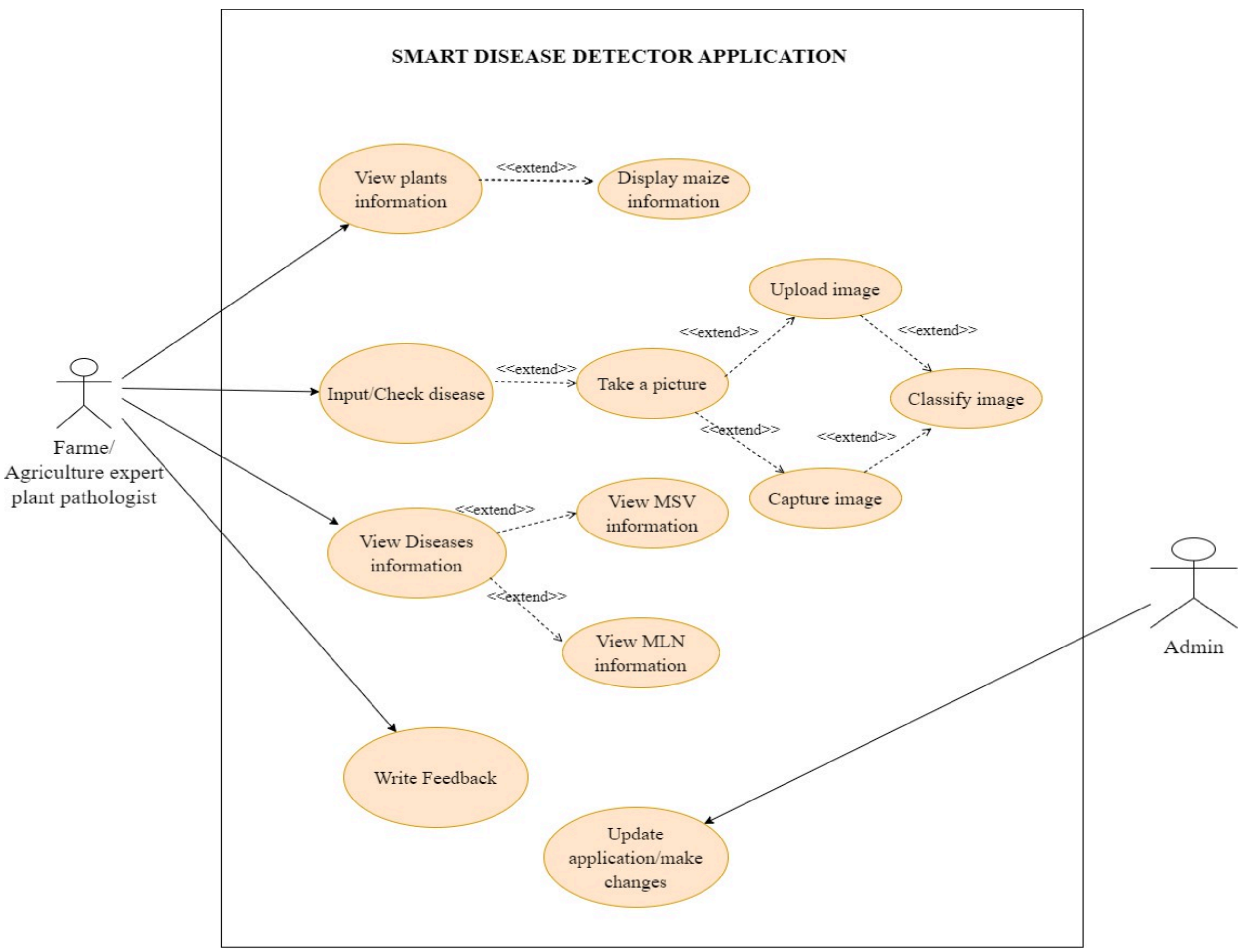


Figure 7: Use case diagram for mobile application

Table 5: Description of the use case

Use case	Description	Actors
1. View plants information	The Actors can get general information about maize plants such as the scientific name, production statistics, and planting information.	Farmer or Agriculture expert or Plant pathologist
2. View Disease information	The Actors can view general information about MSV and MLN.	Farmer or Agriculture expert or Plant pathologist
3. Input/ Check disease	The Actors can access their mobile phone's camera to capture an image or upload an image from the phone gallery. After displaying an image, they can now detect the diseases.	Farmer or Agriculture expert or Plant pathologist
4. Provide Feedback	The Actors can give feedback on the system for better improvement	Farmer or Agriculture expert or Plant pathologist

(ii) Activity Diagram

An activity diagram is a behavioral diagram, meaning it shows how a system behaves. An activity diagram shows the numerous decision routes that are available throughout the execution of an activity by depicting the control flow from a start point to an endpoint (Ahmad *et al.,* 2019). Figure 8 represents the activity diagram for the mobile-based application.

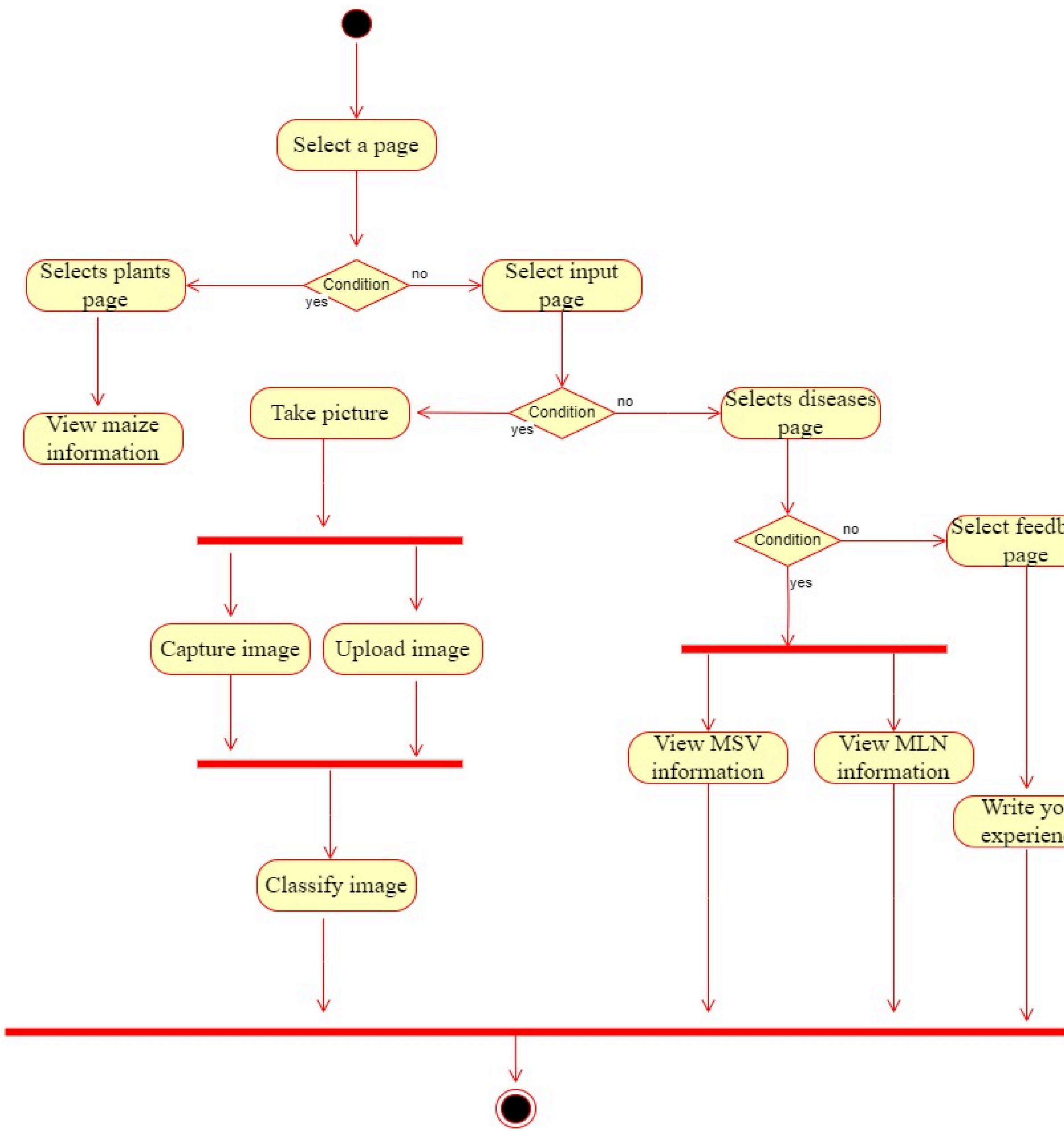


Figure 8: The activity diagram for the mobile application

(iii) Sequence Diagram

The sequence diagram is generally used to display item interactions in the order in which they take place. The dynamic behavior of a system, in particular how elements work together to achieve a certain capability or to satisfy a use case, is visualized and documented using sequence diagrams (Micskei *et al,* 2011). Figure 9 depicts the sequence diagram for the mobile-based application.

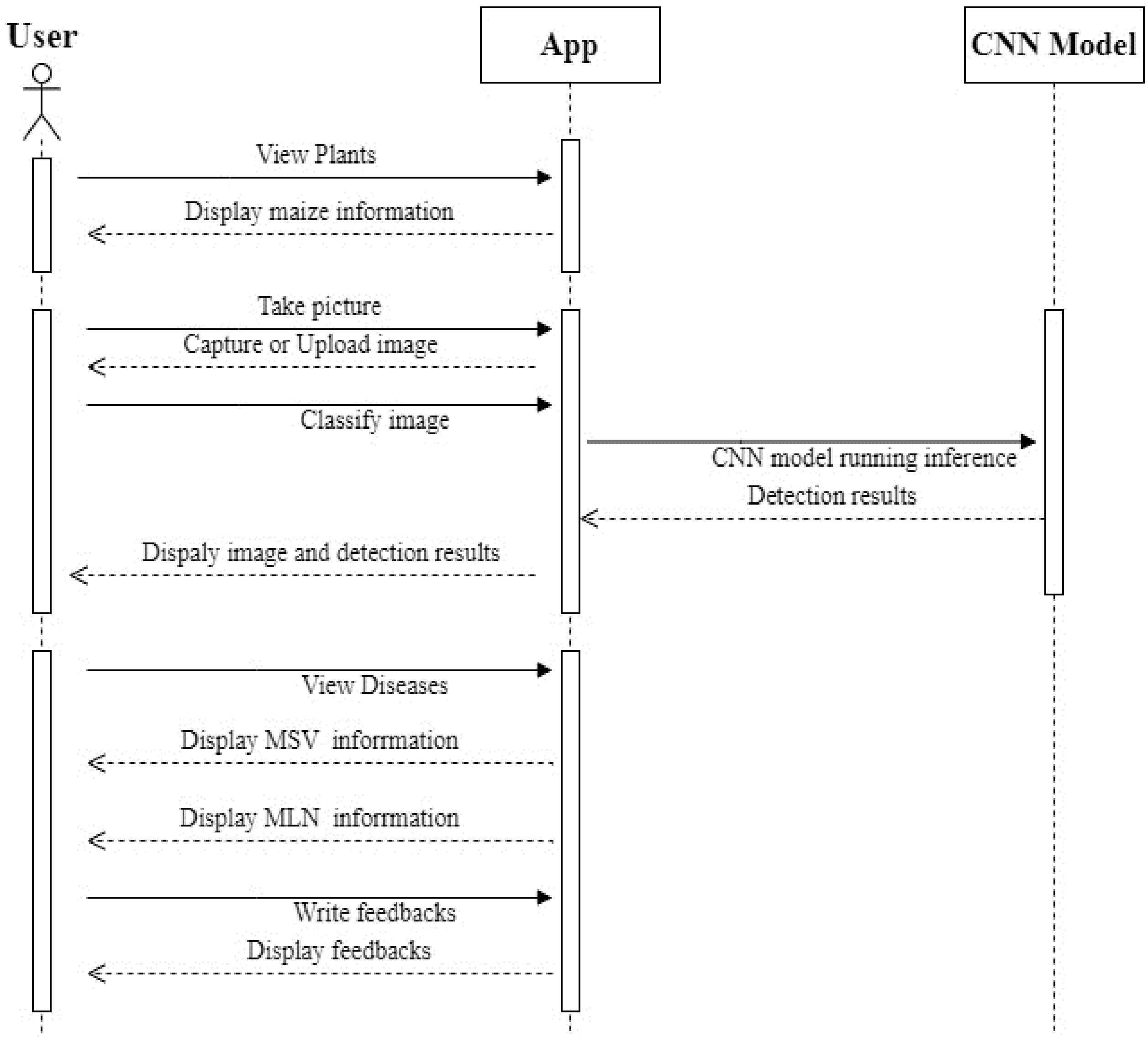


Figure 9: Sequence diagram for mobile application

3.10.4 System Development

(i) Software Requirements

The software used for the development of the mobile application was Android Studio IDE, Operating System, Database, flutter framework, and Java Development Kit, described as follows;

(a) Programming IDEs: The integrated development environments used are Android Studio and Visual Studio Code/Visual Studio.

(b) Tools and Frameworks: The tools and frameworks used are the Flutter framework by Google and Python/Flask. The databases used are SQLite while hosting is done using Render.

(ii) Hardware Requirements

(a) Laptop Intel core i5, RAM 8, 64-bit operating system, 500GB of hard disk drive and 2GB of Graphics, and 15.6 display.

(b) Smartphone Android smartphone.

3.11 Model Deployment

This involves how a machine learning model is set into action to fulfil the purpose for which it was developed. This approach renders the results of the model accessible to users, developers, or systems, allowing them to make data-driven business decisions and get the opportunity to interact with the application (Heymann *et al.,* 2022). The CNN model was selected for deployment because of its small size and it requires less memory and computational power compared to ViT. The CNN model was deployed and hosted on an online server known as render, an Application Programming Interface (API) was developed to bring about communication between the smartphone android application package (APK) and the online server where the model is hosted which acts as a front end. When agriculture stakeholders interact with the application in disease detection the APK sends information to the online server the API receives the requests and then interacts with the deployed model to bring out the results. The results are now sent back to the APK and that is how it works.

3.12 System Validation

User experience and feedback are very essential to ensure that the requirements have been met. Moreover, it helps researchers enhance the capabilities of system features and on a higher level determine the system's applicability to the targeted users (Mgala, 2016). A set of questions was prepared for 10 agriculture stakeholders including 5 farmers, 3 agriculture experts, and 2 plant pathologists from the Arusha region to fill in, according to their experience of the application as part of the survey. They were required to agree or disagree if the requirement was met or not met after they had tried the application on an actual farm. The questionnaire used is presented in Appendix 2.

3.13 Ethical Consideration

This study obtained approval from the respective organs. Each participant was given a consent form before starting data collection, which introduced the overall research idea and requested approval to continue with the data collection process. Furthermore, the study ensured the confidentiality of all respondents.

CHAPTER FOUR

RESULTS AND DISCUSSION

4.1 Introduction

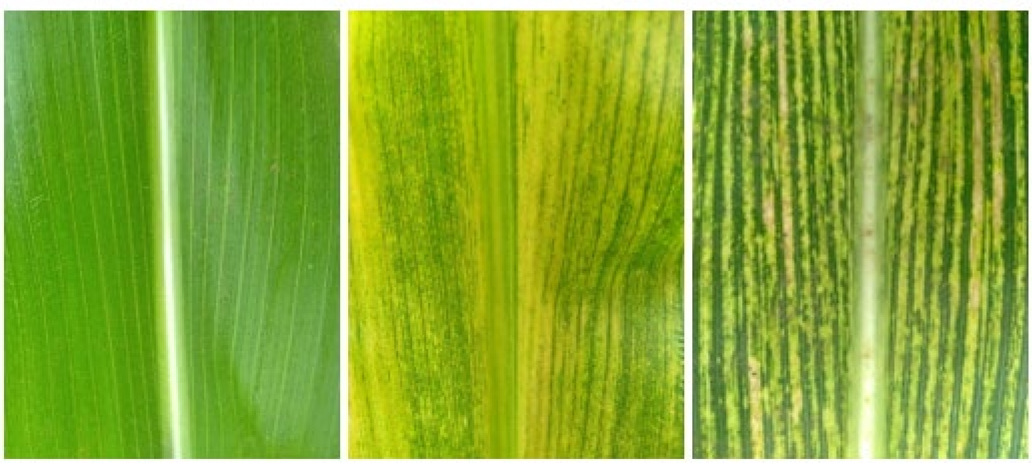
In this chapter, we present the results of the research and provide a comprehensive discussion of their implications. The results are organized to address the research questions and objectives outlined earlier in the dissertation. We also compare our findings with existing literature, analyze their significance, and discuss any potential limitations or implications for future research.

4.2 Results from the Data Collection Methods

4.2.1 Results using ODK Tool

Maize image datasets were collected for a period of six (6) months starting from February 2021 to July of that same year from small-scale farmers in Manyara, Arusha, and Kilimanjaro. Farms with diseased plants were identified with the help of plant pathologists in the regions Different interfaces were part of the Tools design. To capture images iPhone XS Max was utilized to capture clear images. All captured images were in the format of Joint Photographic Group (JPG). The study managed to acquire a total of 27 660 images from the field, this number of images comprised 3 category classes labeled, 9145 healthy images, 8604 MLN images, and 9911 MSV images. Moreover, 675 images were downloaded from the internet for the WRONG class label. Figure 10 shows the researcher collecting data in the field and Fig. 11 shows the sample of image data samples from the three classes that were collected from the field.

Figure 10: Researcher collecting imagery leaves in maize farms



a. b. c.

Figure 11: Sample images from maize dataset where (a) Healthy (b) MLN (c) MSV

4.2.2 Results from the Questionnaires

The system requirements for developing the mobile-based application for the early detection of maize diseases were collected from 5 individuals which include farmers, agricultural experts, plant pathologists, and technical experts. The questions used to get the requirement are presented in Appendix 1. The requirements were divided into two different categories: functional requirements and non-functional requirements.

(i) Functional Requirements

These are the requirements that entail what the mobile-based application can be able to achieve or perform. The following is a list of the features that the mobile-based will be able to perform:

(a) It should provide an option to view general information about the maize plants and an overview of maize streak virus and maize lethal necrosis diseases affecting maize plants.
(b) It should provide an option to capture images of maize leaves that are health-affected with either MSV or MLN directly from the farm.
(c) It should provide an option to upload images of maize leaves that are health-affected with either MSV or MLN from the gallery.
(d) It should provide an option to detect the captured or uploaded images by classifying in which category they belong as either healthy or infected.
(e) It should be able to classify and display the results depending on the captured or uploaded image when the model inference is running on the backend.
(f) It should provide an option to share comments depending on the user experience.

(ii) Non-functional Requirements

These requirements enlist the standards by which a system's performance can be assessed. They indicate the characteristics or requirements that the system must adhere to. The following are non-functional requirements for the mobile-based application:

(a) Availability

The developed mobile application should be available and accessible all the time after it has been installed.

(b) Usability

The developed mobile application should be straightforward to interact with without any instructions or user guidelines.

(c) Performance

The developed mobile application should respond to user interaction within a few seconds, also should not consume more than 20MB of memory on a typical device, moreover, it should be able to handle many users in user load without any delay in response time.

(d) Compatibility

The developed mobile application should be compatible with every mobile device running the Android operating system; however, it should work flawlessly in a variety of settings to increase user satisfaction and adoption.

(e) Flexibility

The developed mobile application should have the ability to undergo modification to accommodate future changes in its requirements

4.3 Model Development Results

4.3.1 Convolutional Neural Networks 1 Results

The model achieved a validation accuracy of 98.44% and a validation loss of 6.33%. The model was then tested using the remaining 10% of the testing set and the results for accuracy were 94.79% and Loss was 14.12%. The rest of the entire result for model training observed during training, from the 1^{st} epoch to the 50^{th} epoch is plotted in Fig. 12.

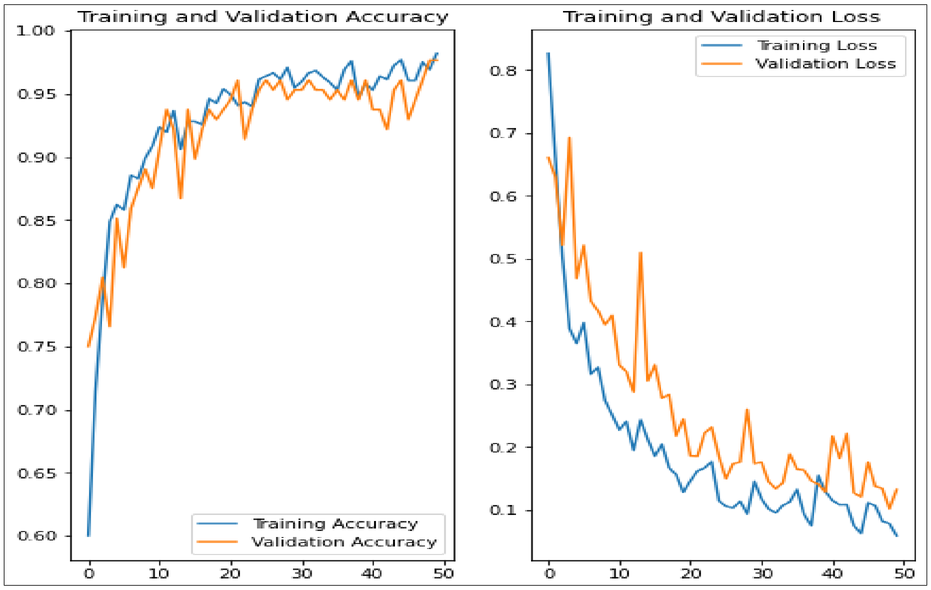


Figure 12: Training and validation plots for the CNN 1 model

4.3.2 Convolutional Neural Networks 2 Results

The model training results show that the second batch got the highest validation accuracy of 97.91% and a low validation loss of 14.65%. The average of the validation accuracy for the entire training for all datasets from all 4 batches is 0.90965. The results for model performance recorded during the 1st to the 50th epoch for each of the four batches are summarized in Table 6. Figure 12 on the left shows the CNN training accuracy and loss curve of over 50 epochs. The results for accuracy over the epoch graph show that the validation accuracy increased rapidly up to the 5th epoch, then remained steady at around 90% exhibiting fluctuations up to the 16th epoch where it dropped to 0.8824 on the 17th epoch and went high again remaining steady in the 90% with fluctuations up to the last epoch and reaching a peak of 0.9790. Meanwhile, the training accuracy increased rapidly up to the 12th epoch and followed a similar trend of remaining steady at 90% with fluctuations hitting a maximum accuracy of 0.9998 surpassing the validation accuracy without any significant fluctuations. This indicates that the model exhibited effective generalization. On the loss over epoch graph in Fig. 13 on the right, the results demonstrate that the training loss decreases rapidly from the 1st epoch to the 10th epoch, after which it starts to fluctuate slightly, exhibiting periodic increases and decreases until the end. Meanwhile, the validation loss shows a rapid decrease from the outset until the 5th epoch, followed by a pattern of fluctuation with periodic increases and decreases until the final epoch. This suggests that the model aligns closely with the characteristics of the dataset throughout both the initial and final phases of the training process.

Table 6: Convolutional Neural Network 2 Model Performance

Batches	Validation accuracy	Validation loss	Precision	Recall	F measure
Batch 1	0.9581	0.3436	1.0000	1.0000	1.0
Batch 2	0.9790	0.1465	0.9998	0.9998	0.9998
Batch 3	0.8135	1.9335	0.9882	0.9872	0.9880
Batch 4	0.8878	0.5497	0.9672	0.9625	0.9648

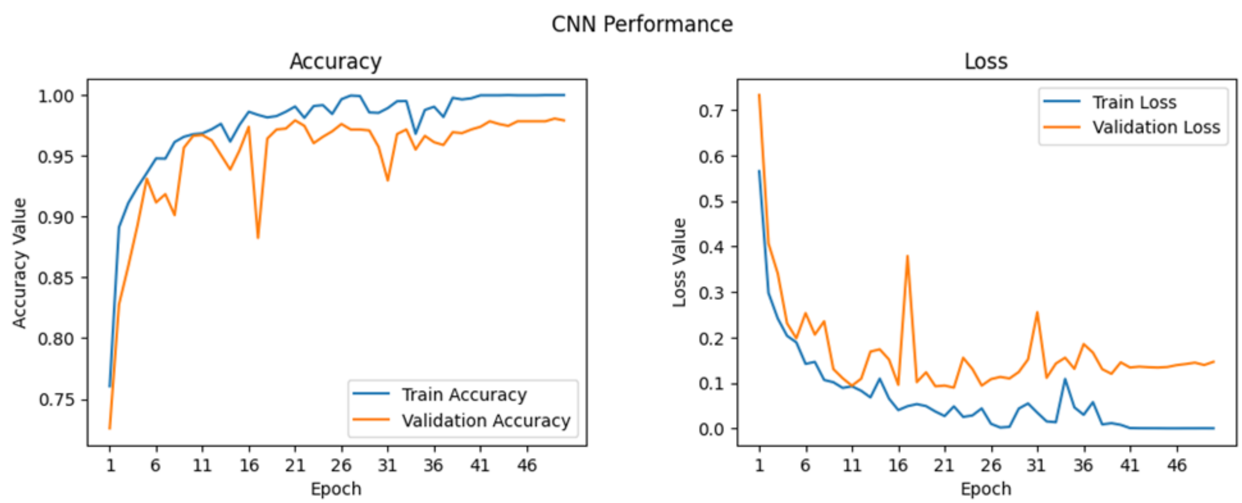


Figure 13: Performance of CNN 2 model

4.3.3 Vision Transformer Results

The model achieved a validation accuracy of 93.10% and a validation loss of 33.71%. The results for model performance recorded during the 1st to the 50th epoch is plotted in Fig. 14. The results for accuracy over the epoch graph show that the validation accuracy increased rapidly up to the 4th epoch, then remained steady at around 80%, and then 90% exhibiting fluctuations up to the 26th epoch where it dropped to 0.8606 on the 27th epoch, and went high again remaining steady in the 90% with fluctuations but dropped again in 40th epoch and went up to the last epoch and reaching a peak of 0.9310. Meanwhile, the training accuracy increased rapidly up to the 10th epoch and followed a similar trend of remaining steady at 90% with fluctuations hitting a maximum accuracy of 0.9777 surpassing the validation accuracy without any significant fluctuations. This indicates that the model exhibited effective generalization. On the loss over epoch graph in Fig. 14 on the right, the results demonstrate that the training loss decreases rapidly from the 1st epoch to the 5th epoch, after which it starts to fluctuate slightly, exhibiting periodic increases and decreases until the end. Meanwhile, the validation loss shows a drop-down from the outset to the 4th epoch, followed by a pattern of fluctuation with periodic increases and decreases until the final epoch. This suggests that the model aligns closely with the characteristics of the dataset throughout both the initial and final phases of the training process.

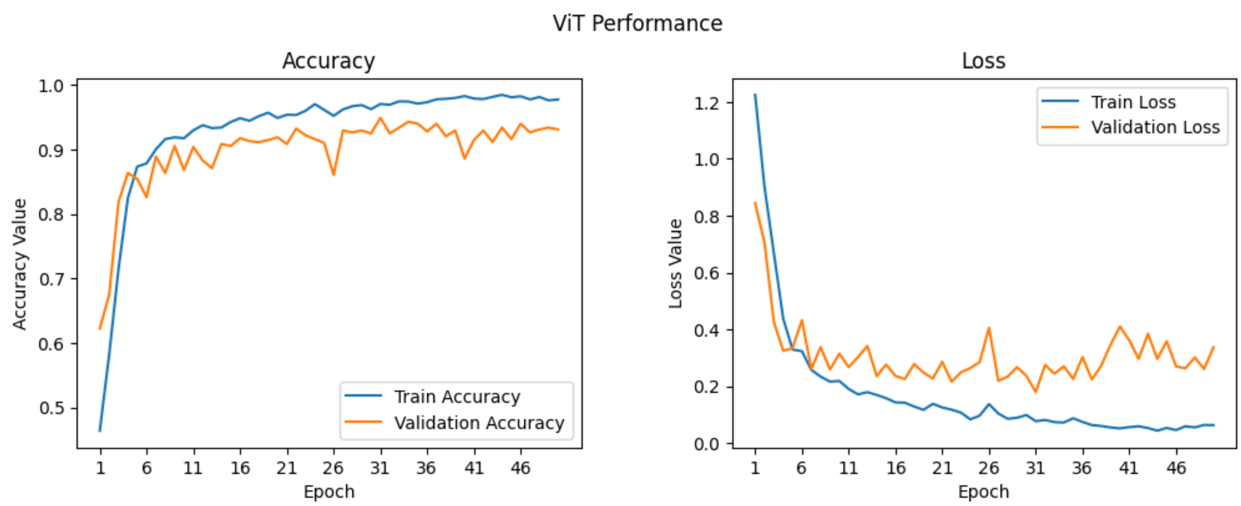


Figure 14: Training and validation plots for ViT model

4.4 System Development Results

A straightforward and easy to use Android mobile-based application was developed to enable early detection of MSV and MLN for farmers, agriculture experts, and plant pathologists. The application will be freely available in the Play Store, the users may simply search for the app with the name "Smart Disease Detector" and then download it and install it on their smartphones. Immediately after installation, the application is ready for use. The applications welcome the user with a splash screen containing a maize image with the title of the application that appears for just a few seconds before taking the user to the contents of the mobile-based application. The mobile-based application consists of four main pages, the plants page, the input page, the disease page, and the feedback page. The first page of the application titled “Plants” consists of the information about maize plant. The splash screen and the first page of plants of the application are shown in Fig. 15.

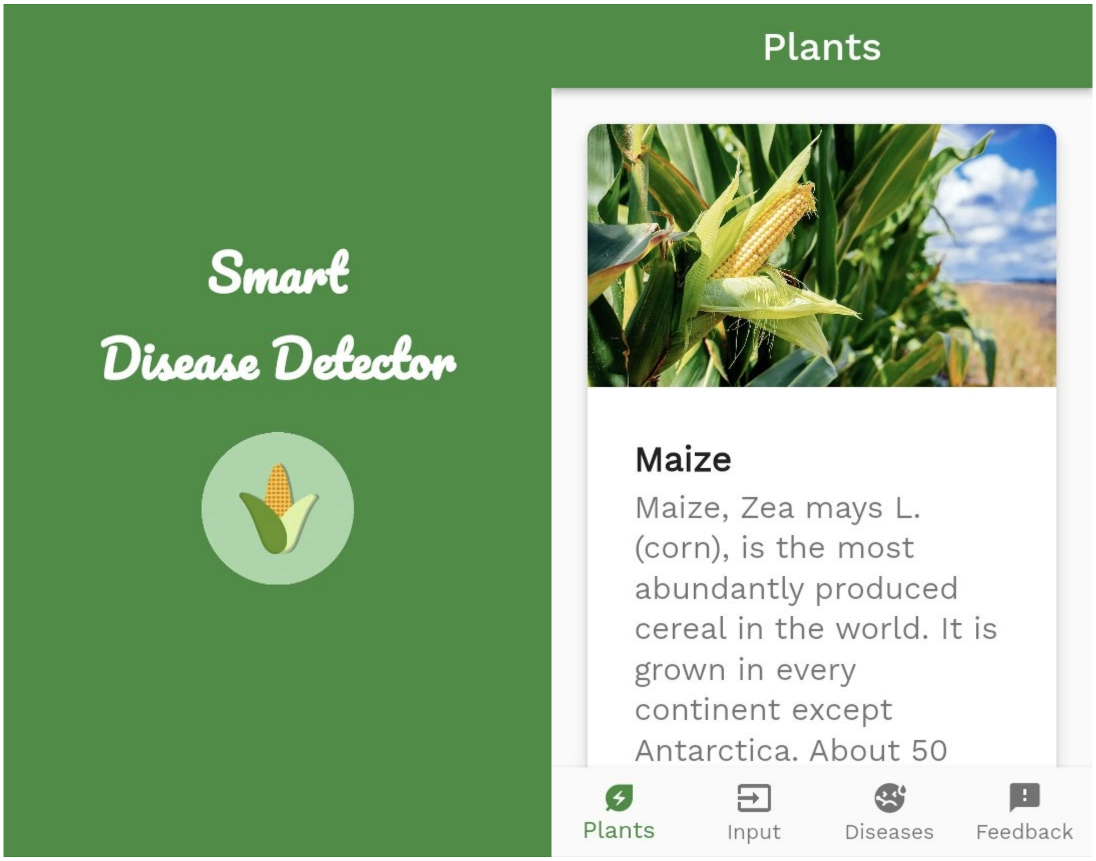


Figure 15: Splash screen page and Plants page

The second page “Input” also checks the disease page which carries the goal of the study which is detection of the maize diseases. On this page you find an option for taking a picture, the option will directly take you to another two options of either capturing an image or uploading an already available image in the gallery. The input page and check disease page are shown in Fig. 16.

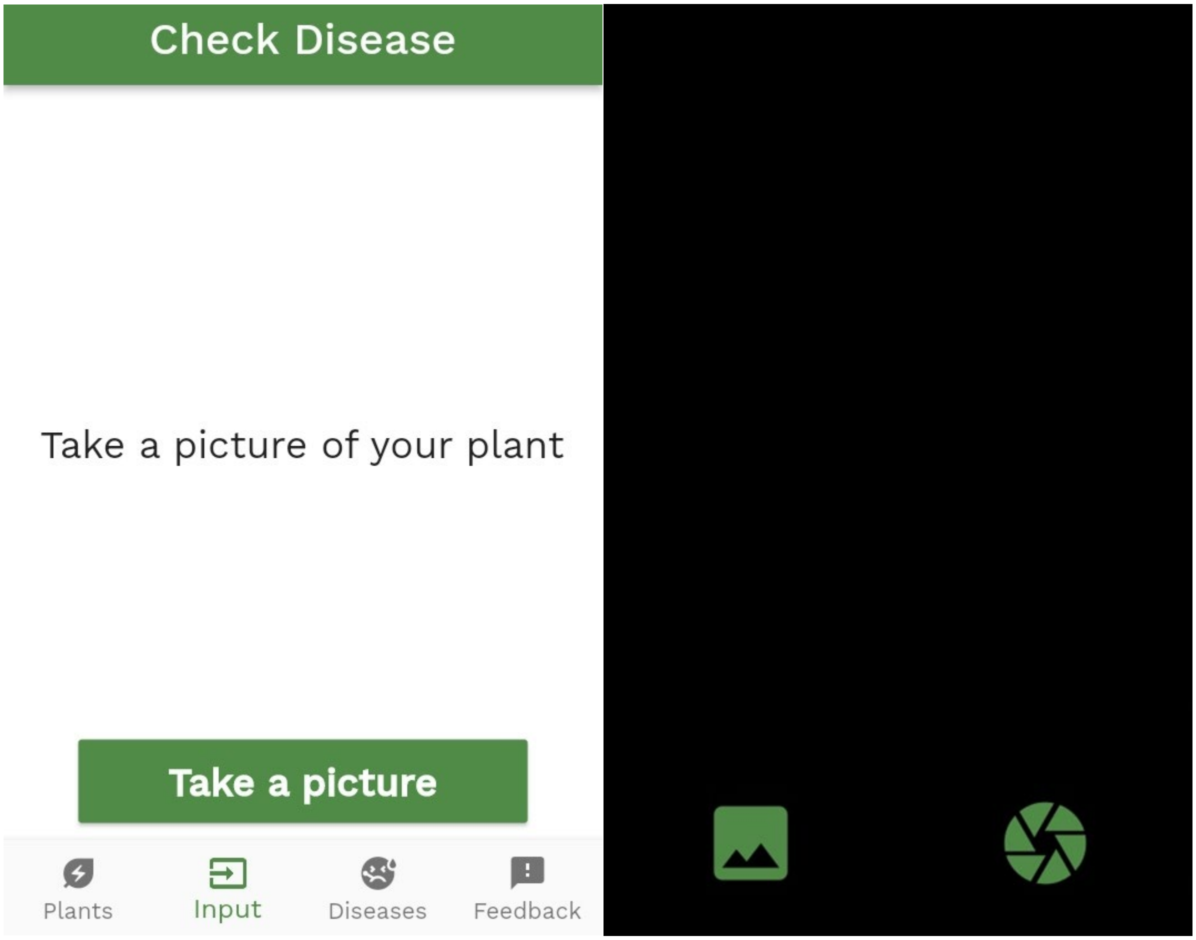


Figure 16: Mobile app landing page including Input and Check disease page

The captured or uploaded image can now be classified with the model running on the back end to get results. Figure 17 shows an uploaded image of maize leaf affected with maize streak virus, maize lethal necrosis after being detected and the results model provided after running inference.

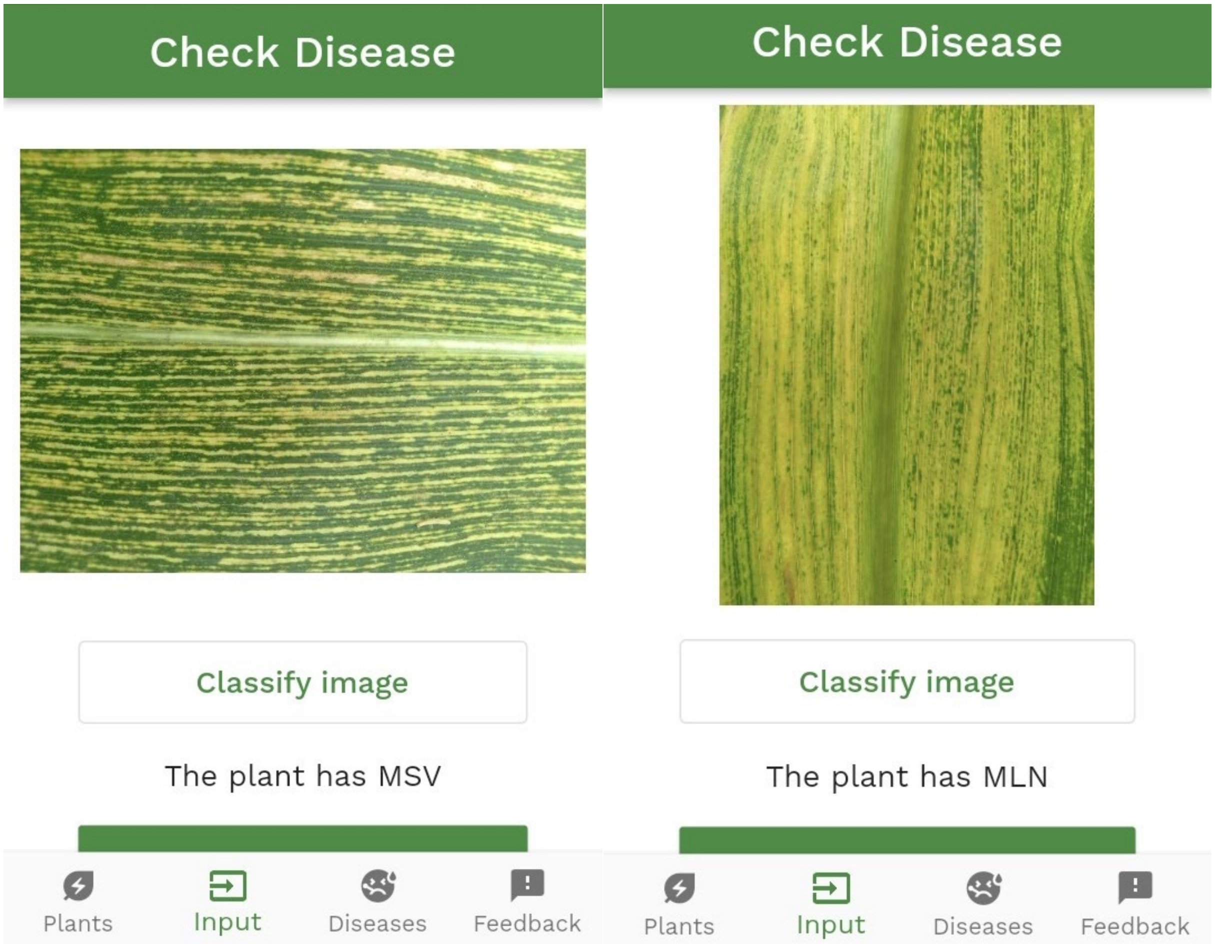


Figure 17: Detected images of maize leaf affected by MSV and MLN

An image of a healthy maize leaf and an image of something different from other maize leaf images were also uploaded to see the result, and the model was able to provide results as shown in Fig. 18.

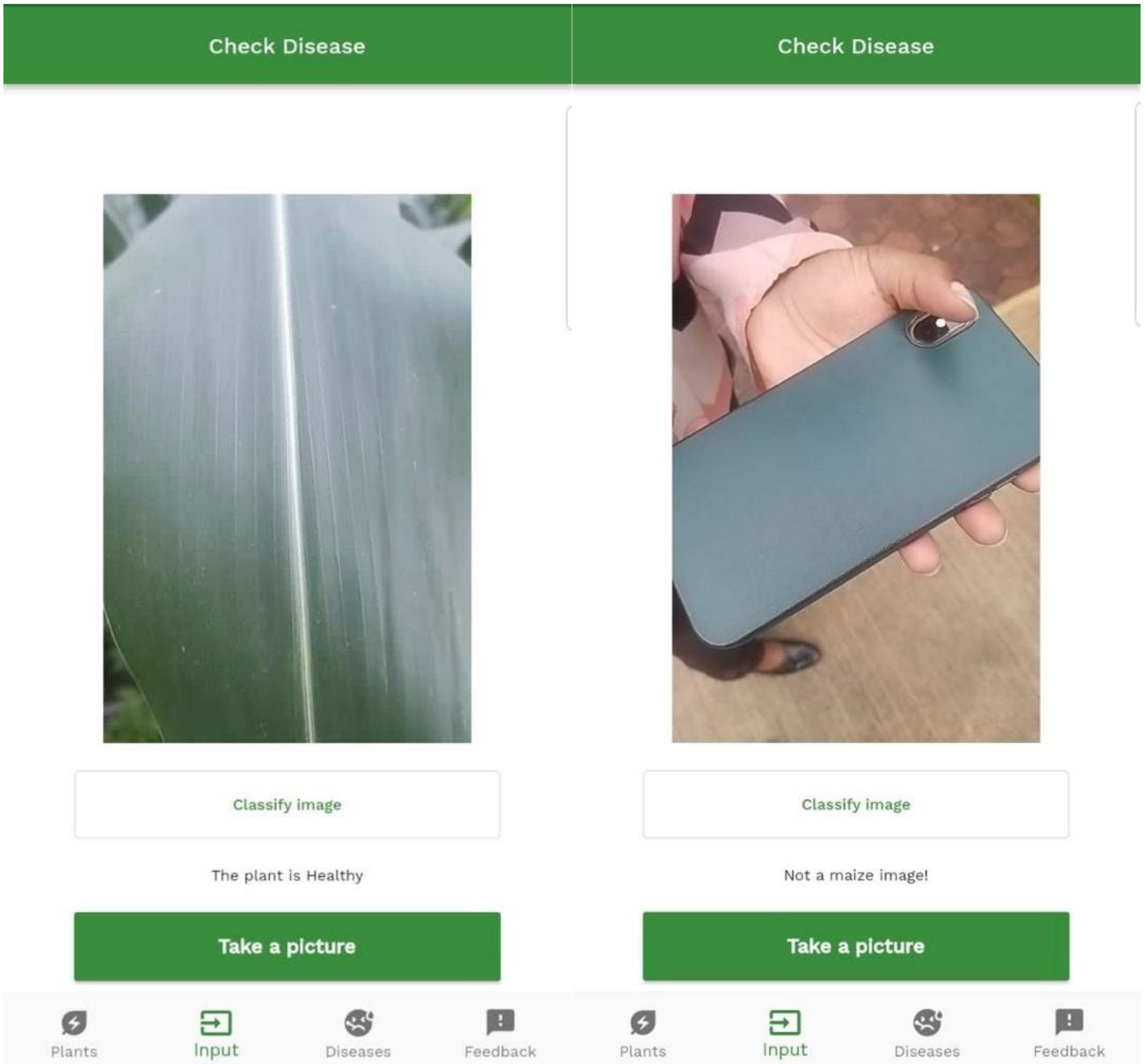


Figure 18: Detection of different images

The third page "Diseases" consists of information concerning the maize diseases this study is worked on. This page provides information about maize streak virus and information about maize lethal necrosis as shown in Fig. 19.

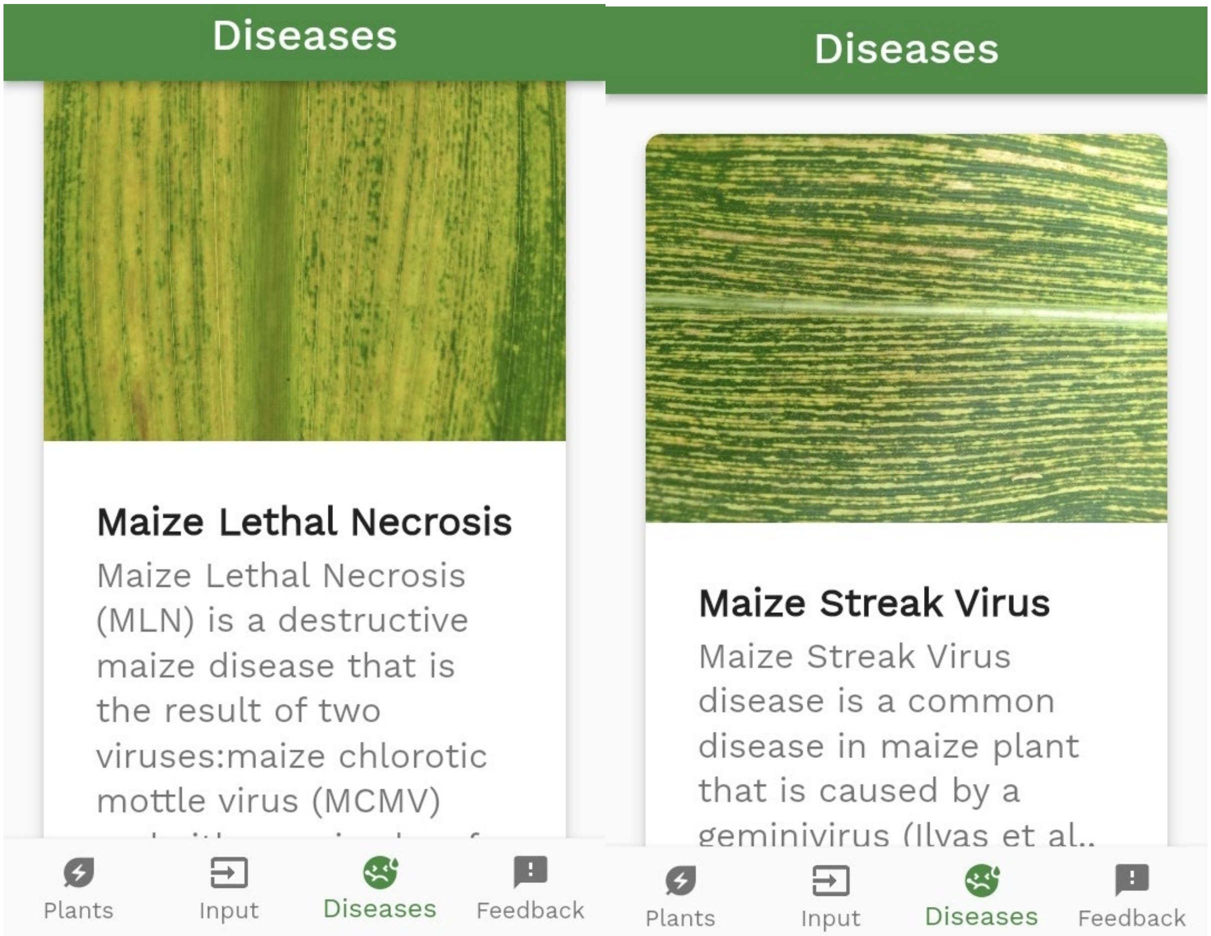


Figure 19: Disease page containing information on MLN and MSV

The last page of the application is called "Feedback" this page is special for the users of the application to write their feedback and experience on how they have seen the application but also the application usage. It is through this page that many improvements will be made because of the user's feedback and comments as shown in Fig. 20.

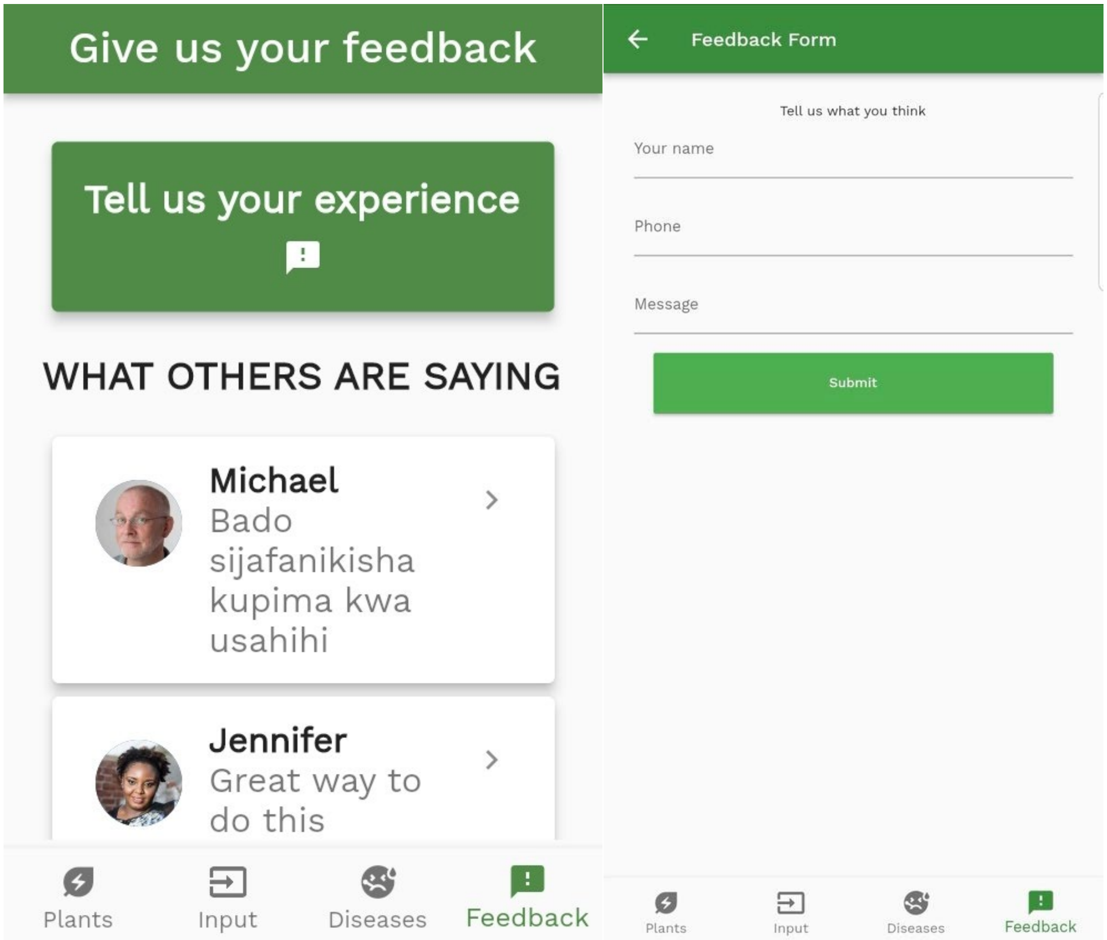


Figure 20: Feedback page

4.5 System Validation Results

Questionnaires were used to validate the model performance. The farmers were asked to utilize the application on actual farms when the researcher visited them again. The farmers were then prompted to respond to questions about their use of the app and their experience with it. Table 7 provides a summary of the responses obtained from the agricultural stakeholders in the questionnaire used for validating the model performance. Both farmers and agriculture experts found the application to be highly interactive and straightforward, with 75% expressing positive feedback. The majority of farmers owned feature phones and had limited understanding of the English language, with 50% appearing to be unfamiliar with smartphones. All features were working properly, as agreed upon by both farmers and the agriculture expert, with 100% expressing satisfaction. Users with featured phones claimed to need assistance when using the application meanwhile users with smartphones claimed not to require assistance when interacting with the application. For learning purposes, all respondents said that the system appears quick to learn. Lastly, the respondents thought it was sophisticated to detect disease using smartphones. Furthermore, the concept of utilizing smartphone devices to predict diseases was observed to be entirely novel by the majority of the respondents. Table 10 summarizes the responses from 10 agriculture stakeholders including 5 farmers, 3 agriculture experts, and 2 plant pathologists from the Arusha region.

Table 7: Summary responses from the agriculture stakeholders

S/N	Questions	Agree (%)	Neutral	Disagree (%)
1.	Does the application provide an option to view general information about the maize plants and an overview of maize streak virus and maize lethal necrosis diseases affecting maize plants?	100	0	0
2.	Does the application provide an option to capture images of maize leaves that are healthy or affected with either MSV or MLN directly from the farm?	100	0	0
3.	Does the application provide an option to upload images of maize leaves that are healthy or affected by either MSV or MLN from the gallery?	100	0	0
4.	Does the application provide an option to detect the captured or uploaded images by classifying in which category they belong as either healthy or infected?	100	0	0
5.	Is the application able to classify and display the results depending on the captured or uploaded image?	100	0	0
6.	Does the application provide an option to share comments depending on the user experience?	100	0	0
7.	Is the application interactive and straight forward	75	0	25
8.	Are you familiar with using smartphone?	50	0	50

4.6 Discussion

This study developed two deep learning models, CNN and ViT. Both models performed well in detecting MSV and MLN diseases in maize plants. The ViT model achieved a validation accuracy of 93.1%, whereas the CNN model achieved an overall average validation accuracy of 90.965%. These results suggest that both models are capable of detecting the presence of diseases in maize plants. Furthermore, these results are considered to be among the best examples of a good model, as a good model is expected to have an accuracy greater than 70% (Maxwell *et al.,* 2021). However, deep learning models also perform very well when trained with larger datasets. The CNN model for this study was trained with 27 588 data samples compared to Syarief *et al.* (2020) who used a few data samples (200) for model training in the detection of maize diseases. The majority of the studies have employed transfer learning to train deep learning models for maize diseases detection and their scope is not focused on Tanzania (Arnaud *et al.,* 2022; Chen *et al.,* 2021; Darwish *et al.,* 2020; Haque *et al.,* 2022; Syarief *et al.*, 2020), unlike the study where both CNN and ViT deep learning models were developed from scratch and the study area is Tanzania. Another study by Sibiya *et al.* (2021) developed a deep learning model for early detection of maize disease using a segmentation approach, while the approach of the study for our case was classification. Furthermore, none of the studies has come up with a combined deep-learning model for the early detection of MSV and MLN diseases in maize. Additionally, when the developed deep learning models were compared, the ViT model had somewhat greater accuracy than the CNN model. According to Dosovitskiy *et al.* (2020), the ViT model's ability to capture global dependencies through self-attention mechanisms gives it an advantage in detecting and classifying various plant diseases with higher accuracy than the CNN model.

The primary function of the mobile-based application is the detection of the maize plant's diseases. It is capable of detecting diseases in maize leaves that are affected by MSV and MLN accurately with a confidence level of 99% of the image. The procedure of manually detecting plant diseases is laborious and prone to mistakes. It may not always be an accurate way to recognize plant diseases and stop them from spreading (Shoaib *et al.,* 2023). Moreover, as of right now, the oldest and most often utilized technique regarding disease classification is the visual analysis of humans. Since the detection is dependent on the agricultural or extension worker's experience, the procedure leaves a lot of space for error. In addition, the process of identifying crops according to visually detectable traits is laborious and time-consuming. Because of their differing experiences, various specialists identify a similar disease to be a distinct condition, resulting in incorrect diagnoses. The mobile application is more accurate than traditional methods because of the exact image used to train the model, unlike the use of vision which cannot be accurate due to maize being infected with many diseases that have a close similarity in terms of colors and streaks.

The importance of application in performing early detection of the diseases includes protecting the plants from severe, permanent, or recurring damages from harvested crops and reducing the severity and spread before it is too late but also potentially halting the propagation of disease (Harakannanavar *et al.,* 2022; Orchi *et al.,* 2022). For the sake of accessibility the application is accessible to all agricultures stakeholders who has access to an Android smart phone, The choice of Android is because many people in Tanzania own an android smartphone (Statcounter, 2023). As of now the application use only English language, farmers who do not know English can only use it when explained by anyone fluent in the English language

CHAPTER FIVE

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

In conclusion, this study has made significant strides in the early detection of maize diseases in Tanzania, specifically focusing on the Maize Streak Virus (MSV) and Maize Lethal Necrosis (MLN). The main objective of this research was to develop deep learning models integrated into a mobile application for early prediction and early control of these devastating diseases. This conclusion summarizes the key findings and contributions of the study:

The study began by identifying the requirements for model development and addressing a gap in the literature, the absence of a combined deep learning model for MSV and MLN detection. This endeavor resulted in the collection of a substantial dataset comprising 26 913 field-acquired images and 675 images downloaded from the internet. The dataset's availability as an open-source resource will facilitate further research on MSV and MLN infections. Deep learning models, namely Convolutional Neural Networks (CNN) and Vision Transformers (ViT), were developed to address the challenge of early disease detection. Both models were developed from scratch, with CNN showcasing its ability to extract local image features, while ViT demonstrated proficiency in understanding global image context. Vision Transformer achieved a validation accuracy of 93.10%, while CNN 90.96%. This highlights the value of deep learning models in the early diagnosis of plant diseases in maize. An Android mobile application was created and a CNN model was selected for deployment. This application represents a crucial bridge between cutting-edge AI technology and practical use in the field of agriculture. Validation of the model's performance through questionnaires administered to farmers and agricultural experts yielded highly positive feedback, indicating the enthusiastic reception and potential impact of the mobile application. Despite these achievements, the study acknowledges its limitations, including a dataset limited to specific regions and a focus on only two maize diseases. Future research should consider expanding accessibility by developing an offline system catering to individuals without smartphones. Moreover, enhancements such as disease control recommendations, plant pathologist contacts, pesticide purchasing guidance, and coverage of diseases affecting other crops can further enrich the system's utility. In sum, this study serves as a commendable step forward in the field of automated plant disease diagnosis, by providing information into the selection and functionality of deep learning models while providing a practical solution to a critical issue in Tanzanian agriculture. Looking to the future, this research sets the stage for continued innovation and the potential for even more profound impacts on food security and sustainable agriculture in the region and beyond.

5.2 Recommendations

5.2.1 Implications to Policy Makers

The findings of this study have several implications for policymakers in the agriculture sector such as promoting AI technologies in agriculture practices to enhance crop health management and productivity. Support for digital infrastructure; Whereby investing in improving digital infrastructures in villages so that farmers can get access to mobile technologies with internet connectivity. Furthermore, allocating funds for further research and development of AI-based agriculture tools to continue improving the accuracy, efficiency, and accessibility of the developed tools.

5.2.2 Implications to Practitioners

For agricultural practitioners and farmers, this study offers practical insights and recommendations to adopt AI-based mobile applications for prompt diagnosis of diseases to improve the health of pants and crop yield, participate in training programs to effectively use digital tools and understand their benefits and limitations, lastly to provide feedback to developers to improve the usability and functionality of AI applications based on the real-life experience.

5.2.3 Future Research

A number of recommendations are provided for improving the efficacy and impact of the developed maize disease detection system and related works in the coming years. Considering the results as well as the implications of this research work. This study recommends future research focus on expanding the dataset used for training the deep learning models. This includes collecting images from a broader range of regions to account for geographical variations in disease patterns and including additional maize diseases for comprehensive coverage.

Moreover, this study recommends an offline version of the mobile application to ensure accessibility to a wider audience, particularly in remote areas. This will enable individuals without smartphones to benefit from the disease detection system. But also, in regions with limited internet access, integrating the system with SMS services can provide valuable information and alerts to farmers, further enhancing their ability to combat maize diseases.

However, the study recommends enhancing the utility of the mobile application and considering incorporating features that provide farmers with comprehensive agricultural support. This may include disease control recommendations, access to plant pathologist contacts, guidance on pesticide purchase, and information on other diseases affecting maize and other crops. Furthermore, the study recommends, expanding the scope of the system to cover other crops of economic importance in addition to maize, addressing a wider range of agricultural challenges. By implementing these recommendations, future endeavors in the field of agricultural technology and disease detection can achieve an even greater impact, contributing to food security, sustainable agriculture, and economic growth in Tanzania and beyond.

REFERENCES

Adams, I. P., Harju, V. ., Hodges, T., Hany, U., Skelton, A., Rai, S., Deka, M. K., Smith, J., Fox, A., Uzayisenga, B., Ngaboyisonga, C., Uwumukiza, B., Rutikanga, A., Rutherford, M., Ricthis, B., Phiri, N., & Boonham, N. (2014). First report of maize lethal necrosis disease in Rwanda. *New Disease Reports*, *29*(1), 22–22. https://doi.org/10.5197/j.2044-0588.2014.029.022

Adedoja, A. O., Owolawi, P. A., Mapayi, T., & Tu, C. (2022). Intelligent Mobile Plant Disease Diagnostic System Using NASNet-Mobile Deep Learning. *IAENG International Journal of Computer Science*, *49*(1), 216–231.

Ahmad, T., Iqbal, J., Ashraf, A., Truscan, D., & Porres, I. (2019). Model-based testing using UML activity diagrams: A systematic mapping study. *Computer Science Review*, *33*, 98–112. https://doi.org/10.1016/j.cosrev.2019.07.001

Aleryani, A., & Aleryani, A. Y. (2016). Comparative Study between Data Flow Diagram and Use Case Diagram Some of the authors of this publication are also working on these related projects: 1000 Researchers FCIT View project A Model to Measure the Impact of Culture on E-Readiness for E-Government. *International Journal of Scientific and Research Publications*, *6*(3), 124. www.ijsrp.org

Arnaud, S. E., Rehema, N., Aoki, S., & Kananu, M. L. (2022). Comparison of Deep Learning Architectures for Late Blight and Early Blight Disease Detection on Potatoes. *Open Journal of Applied Sciences*, *12*(05), 723–743. https://doi.org/10.4236/ojapps.2022.125049

Arora, R., Trelogan, J., & Ba, T. N. (2016). *Using High Performance Computing for Detecting Duplicate, Similar and Related Images in a Large Data Collection*. 1–9.

Atila, Ü., Uçar, M., Akyol, K., & Uçar, E. (2021). Plant leaf disease classification using EfficientNet deep learning model. *Ecological Informatics*, *61*(September 2020), 101182. https://doi.org/10.1016/j.ecoinf.2020.101182

Batta, M. (2018). Machine Learning Algorithms: A Review. *International Journal of Science and Research*, *18*(8), 381–386. https://doi.org/10.21275/ART20203995

Bharali, P., Bhuyan, C., & Boruah, A. (2019). Plant disease detection by leaf image classification using convolutional neural network. *Communications in Computer and Information Science*, *1025 CCIS*, 194–205. https://doi.org/10.1007/978-981-15-1384-8_16

Biswal, A. K., Alakonya, A. E., Mottaleb, K. A., Hearne, S. J., Sonder, K., Molnar, T. L., Jones, A. M., Pixley, K. V., & Prasanna, B. M. (2022). Maize Lethal Necrosis disease: Review of molecular and genetic resistance mechanisms, socio-economic impacts, and mitigation strategies in sub-Saharan Africa. *BMC Plant Biology*, *22*(1), 1–21. https://doi.org/10.1186/s12870-022-03932-y

Boddupalli, P., Suresh, L. M., Mwatuni, F., Beyene, Y., Makumbi, D., Gowda, M., Olsen, M., Hodson, D., Worku, M., Mezzalama, M., Molnar, T., Dhugga, K. S., Wangai, A., Gichuru, L., Angwenyi, S., Alemayehu, Y., Grønbech-Hansen, J., & Lassen, P. (2020). Maize lethal necrosis (MLN): Efforts toward containing the spread and impact of a devastating transboundary disease in sub-Saharan Africa. *Virus Research*, *282*(October 2019), 197943. https://doi.org/10.1016/j.virusres.2020.197943

Borhani, Y., Khoramdel, J., & Najafi, E. (2022). A deep learning based approach for automated plant disease classification using vision transformer. *Scientific Reports*, *12*(1), 1–10. https://doi.org/10.1038/s41598-022-15163-0

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., … Amodei, D. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, *2020-Decem*.

Cappi, C., Chapdelaine, C., Gardes, L., Jenn, E., Lefevre, B., Picard, S., & Soumarmon, T. (2021). *Dataset Definition Standard (DDS)*. http://arxiv.org/abs/2101.03020

Chen, J., Wang, W., Zhang, D., Zeb, A., & Nanehkaran, Y. A. (2021). Attention embedded lightweight network for maize disease recognition. *Plant Pathology*, *70*(3), 630–642. https://doi.org/10.1111/ppa.13322

Cimmyt, T., & Program, M. (2004). *Maize Diseases: A Guide for Field*.

D.Dewbre, C. J., Soglo, Production, F., Cervantes-Godoy, J., PIN, Amegnaglo, Y. Y., Akpa, A. F., Bickel, M., Sanyang, S., Ly, S., Kuiseu, J., Ama, S., Gautier, B. P., Officer, E. S., Officer, E. S., Eberlin, R., Officer, P., Branch, P. A., Oduro-ofori, E., … Swanson, B. E. (2014). The future of food and agriculture: trends and challenges. In *The Future of Food and Agriculture: Trends and Challenges* (Vol. 4, Issue 4).

Darwish, A., Ezzat, D., & Hassanien, A. E. (2020). An optimized model based on convolutional neural networks and orthogonal learning particle swarm optimization algorithm for plant diseases diagnosis. *Swarm and Evolutionary Computation*, *52*, 100616. https://doi.org/10.1016/j.swevo.2019.100616

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, *1*(Mlm), 4171–4186.

Dhau, I., Adam, E., Mutanga, O., & Ayisi, K. K. (2018). Detecting the severity of maize streak virus infestations in maize crop using in situ hyperspectral data. *Transactions of the Royal Society of South Africa*, *73*(1), 8–15. https://doi.org/10.1080/0035919X.2017.1370034

Dictionary, O. (2017). *Artificial Intelligence (AI): What is it and how does it work?," Lexology,*. https://www.lexology.com/library/detail.aspx?g=5424a424-c590-45f0-9e2a-ab05daff032d

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2020). *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. http://arxiv.org/abs/2010.11929

Emeraghi, M., Achigan-Dako, E. G., Nwaoguala, C. N. C., & Oselebe, H. (2021). Maize streak virus research in Africa: An end or a crossroad. *Theoretical and Applied Genetics*, *134*(12), 3785–3803. https://doi.org/10.1007/s00122-021-03914-y

Engn, E., & Fletcher, L. (2003). What is Computer Vision ? *Image Processing*, *September*, 1–10. https://www.ibm.com/topics/computer-vision

Ferentinos, K. P. (2018). Deep learning models for plant disease detection and diagnosis. *Computers and Electronics in Agriculture*, *145*(September 2017), 311–318. https://doi.org/10.1016/j.compag.2018.01.009

Francis, M., & Deisy, C. (2019). Disease Detection and Classification in Agricultural Plants Using Convolutional Neural Networks: A Visual Understanding. *2019 6th International Conference on Signal Processing and Integrated Networks, SPIN 2019*, 1063–1068. https://doi.org/10.1109/SPIN.2019.8711701

Gayatri, K., Kanti, R. D., Sekhar Rao Rayavarapu, V. C., Sridhar, B., & Rama Gowri Bobbili, V. (2021). Image processing and Pattern Recognition based Plant Leaf diseases Identification and Classification. *Journal of Physics: Conference Series*, *1804*(1). https://doi.org/10.1088/1742-6596/1804/1/012160

Gong, X., & Zhang, S. (2023). An Analysis of Plant Diseases Identification Based on Deep Learning Methods. *Plant Pathology Journal*, *39*(4), 319–334. https://doi.org/10.5423/PPJ.OA.02.2023.0034

Gren, L., & Lindman, M. (2020). What an Agile Leader Does: The Group Dynamics Perspective. In *Lecture Notes in Business Information Processing: Vol. 383 LNBIP*. https://doi.org/10.1007/978-3-030-49392-9_12

Han, K., Wang, Y., Chen, H., Chen, X., Guo, J., Liu, Z., Tang, Y., Xiao, A., Xu, C., Xu, Y., Yang, Z., Zhang, Y., & Tao, D. (2023). A Survey on Vision Transformer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *45*(1), 87–110. https://doi.org/10.1109/TPAMI.2022.3152247

Han, K., Xiao, A., Wu, E., Guo, J., Xu, C., & Wang, Y. (2021). Transformer in Transformer. *Advances in Neural Information Processing Systems*, *19*(NeurIPS), 15908–15919.

Haque, M. A., Marwaha, S., Deb, C. K., Nigam, S., Arora, A., Hooda, K. S., Soujanya, P. L., Aggarwal, S. K., Lall, B., Kumar, M., Islam, S., Panwar, M., Kumar, P., & Agrawal, R. C. (2022). Deep learning-based approach for identification of diseases of maize crop. *Scientific Reports*, *12*(1), 1–14. https://doi.org/10.1038/s41598-022-10140-z

Harakannanavar, S. S., Rudagi, J. M., Puranikmath, V. I., Siddiqua, A., & Pramodhini, R. (2022). Plant leaf disease detection using computer vision and machine learning algorithms. *Global Transitions Proceedings*, *3*(1), 305–310. https://doi.org/10.1016/j.gltp.2022.03.016

Heymann, H., Kies, A. D., Frye, M., Schmitt, R. H., & Boza, A. (2022). Guideline for Deployment of Machine Learning Models for Predictive Quality in Production. *Procedia CIRP*, *107*(March), 815–820. https://doi.org/10.1016/j.procir.2022.05.068

Ilyas, M., Rafique, K., Ahmed, S., Zulfiqar, S., Afzal, F., Khalid, M., Gul Kazi, A., & Mujeeb-Kazi, A. (2014). Preventing Potential Diseases of Crop Plants Under the Impact of a Changing Environment. In *Emerging Technologies and Management of Crop Stress Tolerance* (Second Edi, Vol. 2). Elsevier Inc. https://doi.org/10.1016/B978-0-12-800875-1.00009-0

Ismail, S. (2016). Detailed review of Rogers' diffusion of innovations theory and educational technology-related studies based on Rogers' theory and educational technology-related studies based on rogers'. *The Turkish Online Journal of Educational Technology*, *5*(2), 13–23.

Jasim, M. A., & Al-Tuwaijari, J. M. (2020). Plant Leaf Diseases Detection and Classification Using Image Processing and Deep Learning Techniques. *Proceedings of the 2020 International Conference on Computer Science and Software Engineering, CSASE 2020*, 259–265. https://doi.org/10.1109/CSASE48920.2020.9142097

Kiruwa, F. H., Mutiga, S., Njuguna, J., Machuka, E., Senay, S., Feyissa, T., Ndakidemi, P. A., & Stomeo, F. (2020). Status and epidemiology of maize lethal necrotic disease in Northern Tanzania. *Pathogens*, *9*(1), 1–15. https://doi.org/10.3390/pathogens9010004

L., S., Machuve, Jomanga, K. (2020). Mobile-based Deep Learning Models for Banana Disease Detection. *Engineering, Technology & Applied Science Research*, *10*(3), 5674–5677. https://doi.org/10.48084/etasr.3452

Lee, G. Y., Alzamil, L., Doskenov, B., & Termehchy, A. (2021). *A Survey on Data Cleaning Methods for Improved Machine Learning Model Performance*. *September*. http://arxiv.org/abs/2109.07127

Liu, J., & Wang, X. (2021). Plant diseases and pests detection based on deep learning: A review. *Plant Methods*, *17*(1), 1–18. https://doi.org/10.1186/s13007-021-00722-9

Loyani, L., & Machuve, D. (2021). A Deep Learning-based Mobile Application for Segmenting Tuta Absoluta's Damage on Tomato Plants. *Engineering, Technology & Applied Science Research*, *11*(5), 7730–7737. https://doi.org/10.48084/etasr.4355

Mahuku, G., Lockhart, B. E., Wanjala, B., Jones, M. W., Kimunye, J. N., Stewart, L. R., Cassone, B. J., Sevgan, S., Nyasani, J. O., Kusia, E., Kumar, P. L., Niblett, C. L., Kiggundu, A., Asea, G., Pappu, H. R., Wangai, A., Prasanna, B. M., & Redinbaugh, M. G. (2015). Maize lethal necrosis (MLN), an emerging threat to maize-based food security in sub-Saharan Africa. *Phytopathology*, *105*(7), 956–965. https://doi.org/10.1094/PHYTO-12-14-0367-FI

Maiga, Y. (2024). Temporal Forecast of Maize Production in Tanzania: An Autoregressive Integrated Moving Average Approach. *Journal of Agricultural Studies*, *12*(2), 118. https://doi.org/10.5296/jas.v12i2.21679

Margaret Rouse. (2018). *What is convolutional neural network? - Definition from WhatIs.com*. https://searchenterpriseai.techtarget.com/definition/convolutional-neural-network

Martin, D. P., Shepherd, D. N., & Rybicki, E. P. (2008). Maize Streak Virus. *Encyclopedia of Virology*, 263–272. https://doi.org/10.1016/B978-012374410-4.00707-X

Maxwell, A. E., Warner, T. A., & Guillén, L. A. (2021). Accuracy assessment in convolutional neural network-based deep learning remote sensing studies - Part 1: Literature review. *Remote Sensing*, *13*(13). https://doi.org/10.3390/rs13132450

Mgala, M. (2016). *Investigating Prediction Modelling of Academic Performance for Students in Rural Schools in Kenya MVURYA MGALA*. 243.

Micskei, Z., & Waeselynck, H. (2011). The many meanings of UML 2 Sequence Diagrams: A survey. *Software and Systems Modeling*, *10*(4), 489–514. https://doi.org/10.1007/s10270-010-0157-9

Ministry of Agriculture. (2016). *Annual Agriculture Sample Survey Crop and Livestock Report*.

Monjane, A. L., Dellicour, S., Hartnady, P., Oyeniran, K. A., Owor, B. E., Bezeidenhout, M., Linderme, D., Syed, R. A., Donaldson, L., Murray, S., Rybicki, E. P., Kvarnheden, A., Yazdkhasti, E., Lefeuvre, P., Froissart, R., Roumagnac, P., Shepherd, D. N., Harkins, G. W., Suchard, M. A., … Martin, D. P. (2020). Symptom evolution following the emergence of maize streak virus. *ELife*, *9*, 1–19. https://doi.org/10.7554/eLife.51984

Mugo, D., Njagi, K., Chemwei, B., & Motanya, J. (2017). The Technology Acceptance Model (TAM) and its Application to the Utilization of Mobile Learning Technologies. *British Journal of Mathematics & Computer Science*, *20*(4), 1–8. https://doi.org/10.9734/bjmcs/2017/29015

Orchi, H., Sadik, M., & Khaldoun, M. (2022). On using artificial intelligence and the internet of things for crop disease detection: A contemporary survey. *Agriculture (Switzerland)*, *12*(1). https://doi.org/10.3390/agriculture12010009

Oxfordbusinessgroup. (2018). *Already Tanzania's largest sector, agriculture continues to post positive trends | Tanzania 2018 | Oxford Business Group*. https://oxfordbusinessgroup.com/overview/growth-ground-already-largest-economic-sector-agriculture-continues-see-positive-trends-many-areas

Panigrahi, K. P., Das, H., Sahoo, A. K., & Moharana, S. C. (2020). Maize Leaf Disease Detection and Classification Using Machine Learning Algorithms. In H. Das, P. K. Pattnaik, S. S. Rautaray, & K.-C. Li (Eds.), *Progress in Computing, Analytics and Networking* (pp. 659–669). Springer Singapore. https://doi.org/10.1007/978-981-15-2414-1_66

Raghu, M., Unterthiner, T., Kornblith, S., Zhang, C., & Dosovitskiy, A. (2021). Do Vision Transformers See Like Convolutional Neural Networks? *Advances in Neural Information Processing Systems*, *15*(NeurIPS), 12116–12128.

Richey, B., Majumder, S., Shirvaikar, M. V., & Kehtarnavaz, N. (2020). *Real-time detection of maize crop disease via a deep learning-based smartphone app*. *April*, 10. https://doi.org/10.1117/12.2557317

Robertson, A. (2021). *Potential disease problems in corn following corn | Integrated Crop*

Management. https://crops.extension.iastate.edu/encyclopedia/potential-disease-problems-corn-following-corn

Roy, A. M., Bose, R., & Bhaduri, J. (2022). A fast accurate fine-grain object detection model based on YOLOv4 deep neural network. *Neural Computing and Applications*, *34*(5), 3895–3921. https://doi.org/10.1007/s00521-021-06651-x

Sanga, S. L., Machuve, D., & Jomanga, K. (2020). Mobile-based Deep Learning Models for Banana Disease Detection. *Engineering, Technology & Applied Science Research*, *10*(3), 5674–5677. https://doi.org/10.48084/etasr.3452

Savary, S., & Willocquet, L. (2020). Modeling the Impact of Crop Diseases on Global Food Security. *Annual Review of Phytopathology*, *58*, 313–341. https://doi.org/10.1146/annurev-phyto-010820-012856

Shepherd, D. N., Martin, D. P., Van Der Walt, E., Dent, K., Varsani, A., & Rybicki, E. P. (2010). Maize streak virus: An old and complex "emerging" pathogen. *Molecular Plant Pathology*, *11*(1), 1–12. https://doi.org/10.1111/j.1364-3703.2009.00568.x

Shoaib, M., Shah, B., EI-Sappagh, S., Ali, A., Ullah, A., Alenezi, F., Gechev, T., Hussain, T., & Ali, F. (2023). An advanced deep learning models-based plant disease detection: A review of recent research. *Frontiers in Plant Science*, *14*(March), 1–22. https://doi.org/10.3389/fpls.2023.1158933

Sibiya, M., & Sumbwanyambe, M. (2019a). A Computational Procedure for the Recognition and Classification of Maize Leaf Diseases Out of Healthy Leaves Using Convolutional Neural Networks. *AgriEngineering*, *1*(1), 119–131. https://doi.org/10.3390/agriengineering1010009

Sibiya, M., & Sumbwanyambe, M. (2019b). An Algorithm for Severity Estimation of Plant Leaf Diseases by the Use of Colour Threshold Image Segmentation and Fuzzy Logic Inference: A Proposed Algorithm to Update a "Leaf Doctor" Application. *AgriEngineering*, *1*(2), 205–219. https://doi.org/10.3390/agriengineering1020015

Sibiya, M., & Sumbwanyambe, M. (2021). Automatic fuzzy logic-based maize common rust disease severity predictions with thresholding and deep learning. *Pathogens*, *10*(2), 1–17.

https://doi.org/10.3390/pathogens10020131

Singh, A., Ganapathysubramanian, B., Singh, A. K., & Sarkar, S. (2016). Machine Learning for High-Throughput Stress Phenotyping in Plants. *Trends in Plant Science*, *21*(2), 110–124. https://doi.org/10.1016/J.TPLANTS.2015.10.015

Statcounter. (2023). *Mobile Operating System Market Share United Republic Of Tanzania | Statcounter Global Stats*. https://gs.statcounter.com/os-market-share/mobile/tanzania

Stewart, L. (2022). Maize lethal necrosis. *CABI Compendium*, *CABI Compe*(May 2016). https://doi.org/10.1079/cabicompendium.119663

Syarief, M., & Setiawan, W. (2020). Convolutional neural network for maize leaf disease image classification. *Telkomnika (Telecommunication Computing Electronics and Control)*, *18*(3), 1376–1381. https://doi.org/10.12928/TELKOMNIKA.v18i3.14840

Tanzaniainvest. (2021). *Tanzania Agriculture -*. https://www.tanzaniainvest.com/agriculture

Tembo, M., Adediji, A. O., Bouvaine, S., Chikoti, P. C., Seal, S. E., & Silva, G. C. (2020). A quick and sensitive diagnostic tool for detection of Maize streak virus. *Scientific Reports*, *10*(1), 1–10. https://doi.org/10.1038/s41598-020-76612-2

Toseef, M., & Khan, M. J. (2018). An intelligent mobile application for diagnosis of crop diseases in Pakistan using fuzzy inference system. *Computers and Electronics in Agriculture*, *153* (February), 1–11. https://doi.org/10.1016/j.compag.2018.07.034

Trade.gov. (2021). *Tanzania - Agriculture and Agricultural Processing*. https://www.trade.gov/country-commercial-guides/tanzania-agriculture-and-agricultural-processing

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, *2017-Decem* (Nips), 5999–6009.

Wang, S., & Su, Z. (2019). *Metamorphic Testing for Object Detection Systems*.

http://arxiv.org/abs/1912.12162

Yapıcı, M. M., Tekerek, A., & Topaloğlu, N. (2019). Literature Review of Deep Learning Research Areas. *Gazi Mühendislik Bilimleri Dergisi*, *5*(3), 188–215. https://www.researchgate.net/publication/338238434_Literature_Review_of_Deep_Learning_Research_Areas

Zhang, Y., Song, C., & Zhang, D. (2020). Deep Learning-Based Object Detection Improvement for Tomato Disease. IEEE Access, 8, 56607–56614. DOI: 10.1109/ACCESS.2020.2982456

APPENDICES

Appendix 1: Survey questionnaire for functional and non-functional requirements

THE NELSON MANDELA AFRICAN INSTITUTION OF SCIENCE AND TECHNOLOGY

(NM-AIST)

Research Title: A Deep Learning Model for Early Detection of Maize Diseases in Tanzania

1. Which specific features and functionality should the application contain?
2. How should a user interact with the application?
3. Which specific information do you want to be available in the application?
4. What should happen when the application detects a disease?
5. How should a user access the application?
6. Which information don't you need in the application?

Appendix 2: Questionnaire survey for the validation of the Mobile Application

THE NELSON MANDELA AFRICAN INSTITUTION OF SCIENCE AND TECHNOLOGY

(NM-AIST)

Research Title: A Deep Learning Model for Early Detection of Maize Diseases in Tanzania

Dear Participant

I am Flavia Stephen Mayo, taking a master's degree in information and communication science and engineering. I have worked on research on AI in Agriculture where I have developed a smartphone application that will help farmers and all stakeholders of agriculture for early detection of maize streak virus and maize lethal necrosis diseases in maize leaves plants. The application is called Smart Disease Detector. The study was carried out in the Northern regions of Tanzania; Kilimanjaro, Manyara, and Arusha

I have prepared questions meant to be used to validate the Mobile Application.

I hope you feedbacks on the app usage will bring clarity to the developed App.

Your name ……………………………………………………………………….

Title/Occupation ……………………………………………………………….

Instructions:

Select the correct response by either ticking Agree, Neutral or Disagree.

☐ By ticking 'Yes,' you agree to participate in this survey and consent to the collection and use of your responses for research purposes.

S/N	Questions/Maswali	Agree/Kubali
1.	Does the application provide an option to view general information about the maize plants and an overview of maize streak virus and maize lethal necrosis diseases affecting maize plants? Je, program ya simu inatoa taarifa kuhusu mmea wa maindi na taarifa za ugonjwa wa MSV na MLN?	
2.	Does the application provide an option to capture images of maize leaves that are healthy or affected with either MSV or MLN directly from the farm? Je, program ya simu in sahemu ya kupiga picha za majani ya maindi yenye ugonjwa na yasiyo na ugonjwa kutoka shambani?	
3.	Does the application provide an option to upload images of maize leaves that are healthy or affected with either MSV or MLN from the gallery? Je, program ya simu in sehemu ya kupakia picha za majani ya maindi zilizo tayari kwenye simu?	
4.	Does the application provide an option to detect the captured or uploaded images by classifying in which category they belong to as either healthy or infected? Je, program ya simu ina sehemu ya kugundua kama picha ilopakiwa au kupigwa ina ugonjwa au haina ugonjwa?	
5.	Does the application provide an option to share comments depending on the user experience?	

S/N	Questions/Maswali	Agree/Kubali
	Je, program ya simu in sehemu ya kutoa mapendekezo kulingana uzoefu wa mtumiaji?	
6.	Does the application provide an option to share comments depending on the user experience? Je, program ya simu in sehemu ya kutoa mapendekezo kulingana uzoefu wa mtumiaji?	
7.	Is the application interactive and straight forward? Je, program ya simu ni rahisi kutumia?	
8.	Are you familiar with using smartphone? Je, una uzoefu wa kutumia smartphone?	

Thank you for your valuable time!

Appendix 3: Convolutional Neural Network 1 model source code

```
#MOUNT GOOGLE TO DRIVE
from google.colab import drive
drive.mount('/content/drive')
#IMPORT ALL IMPORTANT DEPENDENCIES
import tensorflow as tf
from tensorflow.keras import models, layers
import matplotlib.pyplot as plt
#SETTING IMPORTANT CONSTANTS
IMAGE_SIZE = 512
BATCH_SIZE = 32
CHANNELS = 3
EPOCHS = 50
#LOADING IMAGES INTO THE TENSORFLOW
dataset=tf.keras.preprocessing.image_dataset_from_directory(
 "/content/drive/MyDrive/Training/MAIZE/",
  seed = 123,
  shuffle =True,
  image_size = (IMAGE_SIZE,IMAGE_SIZE),
  batch_size = BATCH_SIZE )
#TO KNOW CLASS NAMES
class_names = dataset.class_names
class_names
len(dataset)
for image_batch, label_batch in dataset.take(1):
```

```
    print(image_batch.shape)
    print(label_batch.numpy())
for image_batch, label_batch in dataset.take(1):
    print(image_batch[0].shape)
#VISUALIZE SOME OF THE IMAGES FROM THE DATASET
for image_batch, label_batch in dataset.take(1):
    plt.imshow(image_batch[0].numpy().astype("uint8"))
    #SHOW LABEL OF IMAGE
    plt.title(class_names[label_batch[0]])
    plt.axis("off")
plt.figure(figsize=(10,10))
for image_batch, label_batch in dataset.take(1):
    for i in range(10):
      ax=plt.subplot(3,4,i+1)
      plt.imshow(image_batch[i].numpy().astype("uint8"))
#SHOW LABEL OF IMAGE
      plt.title(class_names[label_batch[i]])
      plt.axis("off")
len(dataset)
train_size = 0.8
len(dataset)*train_size
train_ds = dataset.take(37)
len(train_ds)
test_ds = dataset.skip(37)
len(test_ds)
```

```
val_size = 0.1

len(dataset)*val_size

val_ds = test_ds.take(4)

len(val_ds)

test_ds = test_ds.skip(4)

len(test_ds)

def get_dataset_partitions_tf(ds, train_split=0.8, val_split=0.1, test_split=0.1, shuffle=True, shuffle_size=10000):

    ds_size=len(ds)

    if shuffle:

        ds=ds.shuffle(shuffle_size, seed=12)

    train_size=int(train_split*ds_size)

    val_size=int(val_split*ds_size)

      train_ds=ds.take(train_size)

        val_ds=ds.skip(train_size).take(val_size)

    test_ds=ds.skip(train_size).skip(val_size)

        return train_ds, val_ds, test_ds

train_ds, val_ds, test_ds = get_dataset_partitions_tf(dataset)

len(train_ds)

len(val_ds)

len(test_ds)

#CACHE, SHUFFLE AND PREFETCH THE DATASET

train_ds=train_ds.cache().shuffle(1000).prefetch(buffer_size=tf.data.AUTOTUNE)

val_ds=val_ds.cache().shuffle(1000).prefetch(buffer_size=tf.data.AUTOTUNE)

test_ds=test_ds.cache().shuffle(1000).prefetch(buffer_size=tf.data.AUTOTUNE)

#BUILDING THE MODEL
```

```
resize_and_rescale= tf.keras.Sequential([
    layers.experimental.preprocessing.Resizing(IMAGE_SIZE,IMAGE_SIZE),
    layers.experimental.preprocessing.Rescaling(1.0/255) ])
#DATA AUGMENTATION
#WE HAVE CREATED LAYERS FOR PREPROCESSING
data_augmentation= tf.keras.Sequential([
 layers.experimental.preprocessing.RandomFlip("horizontal_and_vertical"),
 layers.experimental.preprocessing.RandomRotation(0.2) ])
train_ds=train_ds.map(
    lambda x,y: (data_augmentation(x, training=True), y)
).prefetch(buffer_size=tf.data.AUTOTUNE)
input_shape = (BATCH_SIZE, IMAGE_SIZE,IMAGE_SIZE, CHANNELS)
n_classes = 3
model=models.Sequential([
    resize_and_rescale,
    data_augmentation,
    layers.Conv2D (32, (3,3), activation='relu', input_shape = input_shape),
    layers.MaxPooling2D ((2, 2)),
    layers.Conv2D (64, kernel_size=(3,3),activation='relu'),
    layers.MaxPooling2D ((2,2)),
    layers.Conv2D (64, (3,3),activation='relu'),
    layers.MaxPooling2D ((2,2)),
    layers.Conv2D (64, (3,3),activation='relu'),
    layers.MaxPooling2D ((2,2)),
    layers.Conv2D (64, (3,3),activation='relu'),
```

```
    layers.MaxPooling2D ((2,2)),
    layers.Conv2D (64, (3,3),activation='relu'),
    layers.MaxPooling2D ((2,2)),
    layers.Flatten(),
    layers.Dense(64,activation='relu'),
    layers.Dense(n_classes,activation='softmax'),])
model.build(input_shape=input_shape)
model.summary()
#COMPILING THE MODEL
#We use adam Optimizer, SparseCategoricalCrossentropy for losses, accuracy as a metric
model.compile(
    optimizer='adam',
    loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False),
    metrics=['accuracy'])
history = model.fit(
    train_ds,
    batch_size=BATCH_SIZE,
    validation_data=val_ds,
    verbose=1,
    epochs=50,
scores= model.evaluate(test_ds)
scores
history
history.params
history.history.keys()
```

```
len(history.history['accuracy'])
acc=history.history['accuracy']
val_acc=history.history['val_accuracy']
loss=history.history['loss']
val_loss=history.history['val_loss']
#PLOT GRAPHS
plt.figure(figsize=(8,8))
plt.subplot(1, 2, 1)
plt.plot(range(EPOCHS), acc, label="Training Accuracy")
plt.plot(range(EPOCHS),val_acc, label='Validation Accuracy')
plt.legend(loc='lower right')
plt.title('Training and Validation Accuracy')
plt.subplot(1, 2, 2)
plt.plot(range(EPOCHS), loss, label="Training Loss")
plt.plot(range(EPOCHS),val_loss, label='Validation Loss')
plt.legend(loc='upper right')
plt.title('Training and Validation Loss')
plt.show()
#MAKING PREDICTION
import numpy as np
for images_batch, labels_batch in test_ds.take(1):
  first_image=images_batch[0].numpy().astype('uint8')
  first_label=labels_batch[0].numpy()
  print("first image to predict")
  plt.imshow(first_image)
```

```
  print("actual label:",class_names[first_label])
  batch_prediction=model.predict(images_batch)
  print("predicted label:",class_names[np.argmax(batch_prediction[0])])
  #plt.axis('off')
np.argmax([3.5631895e-04, 9.9629670e-01, 3.3469382e-03])
#CHECKING FOR CONFIDENCE LEVEL
def predict(model,img):
  img_array=tf.keras.preprocessing.image.img_to_array(images[i].numpy())
  img_array=tf.expand_dims(img_array,0) #Create a batch
  predictions=model.predict(img_array)
  predicted_class=class_names[np.argmax(predictions[0])]
  confidence=round(100*(np.max(predictions[0])), 2)
  return predicted_class, confidence
plt.figure(figsize=(15,15))
for images, labels in test_ds.take(1):
  for i in range(8):
    ax=plt.subplot(3,3,i+1)
    plt.imshow(images[i].numpy().astype("uint8"))
    predicted_class, confidence=predict(model, images[i].numpy())
    actual_class = class_names[labels[i]]
    plt.title(f"Actual: {actual_class},\n Predicted:{predicted_class}.\n Confidence: {confidence}%")
    plt.axis("off")
model.save("../Maize.h5")
```

Appendix 4: Convolutional Neural Network 2 Model Source Code

```
# Import Tensorflow
import tensorflow as tf
import os
import time
import matplotlib.pyplot as plt
import numpy as np
# Testing if you are assigned a GPU
tf.test.gpu_device_name()
# First attach your google drive to the colab
from google.colab import drive
drive.mount('/content/drive', force_remount=True)
# change folder
%cd /content/drive/MyDrive/TRAINING/BATCH1
# Directory with our training MSV pictures
train_MSV_dir = os.path.join('/content/drive/MyDrive/TRAINING/BATCH4/TRAIN/MSV')
# Directory with our training MLN pictures
train_MLN_dir = os.path.join('/content/drive/MyDrive/TRAINING/BATCH4/TRAIN/MLN')
# Directory with our training Healthy pictures
train_HEALTHY_dir =
os.path.join('/content/drive/MyDrive/TRAINING/BATCH4/TRAIN/HEALTHY')
#Total number of images in the directories
print('total training MSV images:', len(os.listdir(train_MSV_dir)))
print('total training MLN images:', len(os.listdir(train_MLN_dir)))
print('total training HEALTHY images:', len(os.listdir(train_HEALTHY_dir)))
# from tensorflow.keras import models, layers
Height_size = 256
Width_size = 256
Batch_Size = 32
#Building the model
model = tf.keras.models.Sequential([
  # resize_and_rescale,
```

```
    # Note the input shape is the desired size of the image 512x512 with 3 bytes color
    # This is the first convolution
    tf.keras.layers.Conv2D(16, (3,3), activation='relu', input_shape=(Height_size, Width_size, 3)),
    tf.keras.layers.MaxPooling2D(2, 2),
    # The second convolution
    tf.keras.layers.Conv2D(32, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    # The third convolution
    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    # The fourth convolution
    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    # The fifth convolution
    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    # Flatten the results to feed into a CNN
    tf.keras.layers.Flatten(),
    # 512 neuron hidden layer
    tf.keras.layers.Dense(512, activation='relu'),
    # 3 output neurons.
    tf.keras.layers.Dense(3, activation='softmax')  ])
from keras.callbacks import ModelCheckpoint
model_filename = 'maize_cnn.h5'
callback_checkpoint = ModelCheckpoint(
    model_filename,
    verbose=1,
    monitor='val_loss',
    save_best_only=True,)
from tensorflow.keras.optimizers import RMSprop
model.compile(loss=                    tf.keras.losses.CategoricalCrossentropy(from_logits=True,
name="categorical_crossentropy"),
            optimizer='adam',
```

```
            metrics=['accuracy',                              tf.keras.metrics.Precision(thresholds=None),
tf.keras.metrics.Recall(thresholds=None)])
from PIL import ImageFile
ImageFile.LOAD_TRUNCATED_IMAGES = True
from tensorflow.keras.preprocessing.image import ImageDataGenerator
# All training images will be rescaled by 1./255
train_datagen = ImageDataGenerator(rescale=1/255)
#Selecting batch size
# Flow training images in batches of 256 using train_datagen generator
train_generator = train_datagen.flow_from_directory(
        '/content/drive/MyDrive/TRAINING/BATCH4/TRAIN/',
        target_size=(Height_size, Width_size),
        batch_size=Batch_Size,
        # Since we use sparse_categorical_crossentropy loss, we need categorical labels
        class_mode='categorical')
# All validation images will be rescaled by 1./255
validation_datagen = ImageDataGenerator(rescale=1/255)

validation_generator = train_datagen.flow_from_directory(
    '/content/drive/MyDrive/TRAINING/BATCH4/TEST/',
    target_size=(Height_size, Width_size),
    # Since we use categorical_crossentropy loss, we need categorical labels
    class_mode='categorical')
start = time.time()
history = model.fit(
      train_generator,
      steps_per_epoch=150,
      epochs=50,
      validation_data=validation_generator,
      verbose=1,
      callbacks=[callback_checkpoint])
end = time.time()
Training_time = end - start
```

```
print(f"Execution time: {round(Training_time, 5)} seconds")
precision = 0.9883
recall = 0.9877
F_measure = 2 * ((precision * recall) / (precision + recall))
print("F-measure: ", F_measure)
#Plot CNN Performance
f, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 4))
t = f.suptitle('CNN Performance', fontsize=12)
f.subplots_adjust(top=0.85, wspace=0.3)
max_epoch = len(history.history['accuracy'])+1
epoch_list = list(range(1,max_epoch))
ax1.plot(epoch_list, history.history['accuracy'], label='Train Accuracy')
ax1.plot(epoch_list, history.history['val_accuracy'], label='Validation Accuracy')
ax1.set_xticks(np.arange(1, max_epoch, 5))
ax1.set_ylabel('Accuracy Value')
ax1.set_xlabel('Epoch')
ax1.set_title('Accuracy')
l1 = ax1.legend(loc="best")
ax2.plot(epoch_list, history.history['loss'], label='Train Loss')
ax2.plot(epoch_list, history.history['val_loss'], label='Validation Loss')
ax2.set_xticks(np.arange(1, max_epoch, 5))
ax2.set_ylabel('Loss Value')
ax2.set_xlabel('Epoch')
ax2.set_title('Loss')
l2 = ax2.legend(loc="best")
model.summary()
```

Appendix 5: Vision Transformer Model Source Code

```
from google.colab import drive

drive.mount('/content/drive')

from google.colab import files

files.upload()

from vit import ViT

pip install patchify

!pip install import-ipynb

import os

os.environ["TF_CPP_MIN_LOG_LEVEL"] = "2" # This to avoid tensorflow message

import numpy as np

import cv2

import matplotlib.pyplot as plt

from glob import glob

from sklearn.utils import shuffle

from sklearn.model_selection import train_test_split

from patchify import patchify #In transformer image is converted into patches then flattened

import tensorflow as tf

from tensorflow.keras.callbacks import ModelCheckpoint, ReduceLROnPlateau, CSVLogger,
EarlyStopping

import import_ipynb

from vit import *

#from vit_tensorflow import ViT

import pandas as pd

#HYPERPARAMETERS
```

```
hp = {}
hp["image_size"] = 200
hp["num_channels"] = 3
hp["patch_size"] = 25
hp["num_patches"] = (hp["image_size"]**2) // (hp["patch_size"]**2)
hp["flat_patches_shape"] = (hp["num_patches"], hp["patch_size"]*hp["patch_size"]*hp["num_channels"])
hp["batch_size"] = 32
hp["lr"] = 1e-4 #Learning rate
hp["num_epochs"] = 50
hp["num_classes"] = 3
hp["class_names"] = ["MSV","MLN","HEALTHY"]
hp["num_layers"] = 8
hp["hidden_dim"] =512
hp["mlp_dim"] = 2048
hp["num_heads"] = 8
hp["dropout_rate"] = 0.1
def create_dir(path):
    if not os.path.exists(path):
        os.makedirs(path)
def load_data(path, split=0.1):
    images = shuffle(glob(os.path.join(path,"*","*.jpg")))
    split_size = int(len(images) * split)
    train_x, valid_x = train_test_split(images, test_size=split_size, random_state = 42)
    train_x, test_x =  train_test_split(train_x, test_size=split_size, random_state = 42)
    return train_x, valid_x, test_x
```

```
def process_image_label(image, label, hp):
  print(image.shape)
  image = cv2.resize(image, (hp["image.size"], hp["image_size"]))
def process_image_label(path):
    #READING IMAGES
    path = path.decode()
    image = cv2.imread(path, cv2.IMREAD_COLOR)
    image = cv2.resize(image, (hp["image_size"], hp["image_size"]))
    image = image/255.0
    #PREPROCESSING TO PATCHES
    patch_shape = (hp["patch_size"], hp["patch_size"], hp["num_channels"])
    patches = patchify(image, patch_shape, hp["patch_size"])
    #patches = np.reshape(patches,(64,25,25,3))
    #for i in range(64):
     #   cv2.imwrite(f"files/{i}.png", patches[i])
    patches = np.reshape(patches, hp["flat_patches_shape"])
    patches = patches.astype(np.float32)
    #LABEL
    class_name = path.split("/")[-2]
    class_idx = hp["class_names"].index(class_name)
    class_idx = np.array(class_idx, dtype=np.int32)
    return patches, class_idx
def parse(path):
    patches, labels= tf.numpy_function(process_image_label, [path], [tf.float32, tf.int32])
    labels = tf.one_hot(labels, hp["num_classes"])
```

```
    patches.set_shape(hp["flat_patches_shape"])
    labels.set_shape(hp["num_classes"])
    return patches, labels
def tf_dataset(images, batch=32):
    ds = tf.data.Dataset.from_tensor_slices((images))
    ds = ds.map(parse).batch(batch).prefetch(8)
    return ds
if __name__ == "__main__":
    #SEEDING
    np.random.seed(42)
    tf.random.set_seed(42)
    #DIRECTORY FOR STORING THE
    create_dir("files")
    #PATHS
    dataset_path = "/content/drive/MyDrive/TRANSFORMERS/maize6000"
    model_path = os.path.join("files", "model.h5")
    csv_path = os.path.join("files", "log.csv")
    #DATASET
    train_x, valid_x, test_x = load_data(dataset_path)
    print(f"Train: {len(train_x)} - Valid: {len(valid_x)} - Test: {len(test_x)}")
    train_ds = tf_dataset(train_x, batch=hp["batch_size"])
    valid_ds = tf_dataset(valid_x, batch=hp["batch_size"])

    #MODEL
    model = ViT(hp)
```

```
    model.compile(
        loss="categorical_crossentropy",
        optimizer=tf.keras.optimizers.Adam(hp["lr"], clipvalue=1.0),
        metrics=["accuracy"])
    callbacks = [
        ModelCheckpoint(model_path, monitor='val_loss', verbose=1, save_best_only=
        ReduceLROnPlateau(monitor='val_loss', factor=0.1, patience=10, min_lr=1e-1)),
        CSVLogger(csv_path),
        EarlyStopping(monitor='val_loss', patience=50, restore_best_weights=False)]
    history = model.fit(
                train_ds,
                epochs=hp["num_epochs"],
                validation_data=valid_ds,
                callbacks=callbacks)
#Plot ViT graph
f, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 4))
t = f.suptitle('ViT Performance', fontsize=12)
f.subplots_adjust(top=0.85, wspace=0.3)
max_epoch = len(history.history['accuracy'])+1
epoch_list = list(range(1,max_epoch))
ax1.plot(epoch_list, history.history['accuracy'], label='Train Accuracy')
ax1.plot(epoch_list, history.history['val_accuracy'], label='Validation Accuracy')
ax1.set_xticks(np.arange(1, max_epoch, 5))
ax1.set_ylabel('Accuracy Value')
ax1.set_xlabel('Epoch')
```

```
ax1.set_title('Accuracy')
l1 = ax1.legend(loc="best")
ax2.plot(epoch_list, history.history['loss'], label='Train Loss')
ax2.plot(epoch_list, history.history['val_loss'], label='Validation Loss')
ax2.set_xticks(np.arange(1, max_epoch, 5))
ax2.set_ylabel('Loss Value')
ax2.set_xlabel('Epoch')
ax2.set_title('Loss')
l2 = ax2.legend(loc="best")
#Arrangment of classes
train_dataset = tf.keras.preprocessing.image_dataset_from_directory(
    "/content/drive/MyDrive/TRANSFORMERS/maize6000",
    shuffle = True,
    #image_size = (Height_size,Width_size),
    #batch_size = Batch_Size)
class_names = train_dataset.class_names
class_names
```

RESEARCH OUTPUTS

(a) Research Paper

Mayo, F. S., & Mduma, N. (2024). *Convolutional Neural Network Deep Learning Model for Early Detection of Streak Virus and Lethal Necrosis in Maize: A Case of Northern-Highlands, Tanzania*. 87–93. https://doi.org/10.1007/978-3-031-56576-2_8

(b) Poster Presentation