

**DEEP LEARNING MACHINE LEARNING MODEL FOR CROP DISEASE DETECTION**

**SUBMITTED TO THE**

**DEPARTMENT OF MATHEMATICS, PHYSICS & COMPUTING,**

**SCHOOL OF SCIENCES AND AEROSPACE STUDIES,**

**MOI UNIVERSITY MAIN CAMPUS.**

In Partial Fulfilment of

The Requirements for the Degree of

Bachelors of Science in Computer Science

*by*

Reg. No: **COM/40/15**

Name: **NOAH LIGALA OLOO**

Date: **28TH APRIL 2023**

## **Declaration**

I declare that this report is my original work and has not been presented for a degree award in any other university. No part of this work may be reproduced without prior written permission of the author and/or Moi University.

*Reg. No: COM/40/15*

*Name: NOAH LIGALA OLOO*

*Signature…………………………………*

*Date……………………………………….*

## **Approval /Acceptance page**

Supervisor’s Declaration

*This project has been submitted for review with my approval as the University Supervisor. Supervisor’s signature………………………. Name: ………………………………………. Date …………………………………………*

DEPARTMENT OF MATHS, PHYSICS AND COMPUTER SCIENCE

MOI UNIVERSITY

## **Dedication**

This project is dedicated to my Parents and entire family who have always been supportive and generous in their love and affection, I also dedicate this to my lovely wife who has in the entirety of my studies been just as much engaged and strained alongside me. My lecturers have also provided invaluable insight and pointed me in the right direction ever so kindly and graciously, pulling up my understanding and depth of perception in Computer Science.

Finally, I dedicate this project to the many technologists out there seeking to bridge the gap between man and machine your passion and inputs have gone a long way in shaping this research paper.

## **ABSTRACT**

The goal of this research is to create a computer vision system based on deep learning that can identify crop illnesses in potato plants. The suggested approach uses cutting-edge deep learning algorithms to evaluate pictures of potato plants and spot any disease-related symptoms like discolouration, amputations, and lesions. The dataset utilized for the model's training and testing is compiled from a variety of sources, including field studies, lab tests, and publicly accessible information. When it comes to identifying common potato diseases like late blight and early blight the suggested system has excellent accuracy. The system is anticipated to have substantial implications on the produce and costs of farming through enabling early detection of crop diseases, improving management practices and consequently increasing crop yields.

## **KEYWORDS AND PHRASES**

Deep Learning

Machine Learning

Neural Networks

Layers

Epochs

Accuracy

Numpy

Pandas

TensorFlow

Keras

Notebook

Jupyter

Model

Predict

Prediction

Early Blight

Late Blight

# **TABLE OF CONTENTS**

[**Declaration** 2](#_Toc134193341)

[**Approval /Acceptance page** 3](#_Toc134193342)

[**Dedication** 4](#_Toc134193343)

[**ABSTRACT** 5](#_Toc134193344)

[**KEYWORDS AND PHRASES** 6](#_Toc134193345)

[**TABLE OF CONTENTS** 7](#_Toc134193346)

[**CHAPTER 1 INTRODUCTION AND BACKGROUND** 9](#_Toc134193347)

[**1.1** **PROBLEM STATEMENT** 9](#_Toc134193348)

[**1.1.1 Early Blight** 9](#_Toc134193349)

[**1.1.2 Late Blight** 9](#_Toc134193350)

[**1.2** **OBJECTIVES** 11](#_Toc134193351)

[**1.3** **PROBLEM STATEMENT** 12](#_Toc134193352)

[**1.4** **PROJECT FEATURES** 12](#_Toc134193353)

[**CHAPTER 2 LITERATURE REVIEW** 13](#_Toc134193354)

[**2.1 DOMAIN SCOPE** 13](#_Toc134193355)

[**2.2 MACHINE LEARNING** 13](#_Toc134193356)

[**Machine Learning Methods** 13](#_Toc134193357)

[**2.2.1 NEURAL NETWORKS** 14](#_Toc134193358)

[**2.3 TRADITIONAL DIAGNOSTICS** 15](#_Toc134193359)

[**2.4 PROPOSED SYSTEM** 15](#_Toc134193360)

[**2.5 MACHINE LEARNING ALGORITHMS** 16](#_Toc134193361)

[**2.6 SOFTWARE DESCRIPTION/REQUIREMENTS** 17](#_Toc134193362)

[**CHAPTER 3 REQUIREMENT ANALYSIS** 18](#_Toc134193363)

[**3.1 METHODOLOGY** 18](#_Toc134193364)

[***3.1.1 Data Collection*** 18](#_Toc134193365)

[***3.1.2 Data Pre-processing*** 18](#_Toc134193366)

[***3.1.3 Model Architecture*:** 18](#_Toc134193367)

[***3.1.4 Model Training*** 18](#_Toc134193368)

[***3.1.5 Model Evaluation*** 18](#_Toc134193369)

[***3.1.6 Model Deployment*** 18](#_Toc134193370)

[***3.1.7 Model Interpretation:*** 18](#_Toc134193371)

[**3.2 FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS** 19](#_Toc134193372)

[**3.2.1 FUNCTIONAL REQUIREMENTS** 19](#_Toc134193373)

[**3.2.2 NON – FUNCTIONAL REQUIREMENTS** 19](#_Toc134193374)

[**3.2.3 HARDWARE REQUIREMENTS** 20](#_Toc134193375)

[**3.2.4 SOFTWARE REQUIREMENTS** 20](#_Toc134193376)

[**CHAPTER 4: DESIGN** 21](#_Toc134193377)

[**4.1 SYSTEM ARCHITECTURE** 21](#_Toc134193378)

[**4.2 DATA FLOW** 21](#_Toc134193379)

[**4.3 EXPECTED OUTCOME** 22](#_Toc134193380)

[**CHAPTER 5 IMPLEMENTATION AND OUTPUT** 23](#_Toc134193381)

[**5.1 DATA PREPARATION** 23](#_Toc134193382)

[5.1.1 BATCH\_SIZE 24](#_Toc134193383)

[5.1.2 IMAGE\_SIZE 24](#_Toc134193384)

[5.1.3 EPOCHS 24](#_Toc134193385)

[5.1.4 CHANNELS 25](#_Toc134193386)

[5.1.5 DATA AUGMENTATION 25](#_Toc134193387)

[5.2 VISUALIZING THE DATA 25](#_Toc134193388)

[**5.3 BUILDING THE MODEL** 27](#_Toc134193389)

[5.4 TRAIN THE MODEL 29](#_Toc134193390)

[**Training Process** 33](#_Toc134193391)

[**5.5 TEST AND PERFORMANCE** 33](#_Toc134193392)

[**5.5.1 PREDICTIONS** 34](#_Toc134193393)

[**CONCLUSION** 38](#_Toc134193394)

[**LIST OF REFERRENCE** 40](#_Toc134193395)

# **CHAPTER 1 INTRODUCTION AND BACKGROUND**

## **PROBLEM STATEMENT**

### **1.1.1 Early Blight**

With an annual global production of more than 300 million tons, potatoes are one of the most significant food crops in the world. However, early blight and late blight are two diseases that frequently harm the crop and can significantly lower yields and lower the quality of the produced crop. Manual inspections are the main way to detect these diseases which takes a lot of time, require a lot of work, and are frequently inaccurate are the foundation of conventional approaches for identifying and managing chronic diseases.

Early blight is primarily a disease of stressed or senescing plants caused by the fungal pathogen *Alternaria solani*. Symptoms appear first on the oldest foliage. Affected leaves develop circular to angular dark brown lesions 0.12 to 0.16 inch (3–4 mm) in diameter. Concentric rings often form in lesions to produce characteristic target-board effect. Severely infected leaves turn yellow and drop. Infected tubers show a brown, corky dry rot.

The disease damages the foliage, stems, and tubers. It can lower crop marketability, yield, tuber size, and storage capacity. By preserving ideal growing conditions, including as appropriate fertilization, irrigation, and pest management, early blight can be minimized. Grow cultivars with longer seasons and later maturation. Only when a disease has started early enough to result in financial loss is fungicide use justified. Tubers can be infected as they are lifted through the soil at harvest. If sufficient moisture is present, spores germinate and infect the tubers.

### **1.1.2 Late Blight**

The oomycete pathogen Phytophthora infestans, which resembles a fungus, is the cause of late blight. P. infestans can infect other solanaceous plants, such as tomatoes, petunias, and hairy nightshade, in addition to potatoes, which are the disease's main substrate. These infected species can serve as a source of potato inoculum. The first signs of late blight in a field are small, circular to irregular-shaped, light to dark green, moisture-soaked spots Usually, the lower leaves are where these lesions initially emerge. Since dew is kept the longest at the leaf tips and edges, lesions frequently start to form there.

Although in most instances these diseases can be easily controlled through competent and up

To standard farming practices if not detected early they may lead to huge losses on the yield as well as the quality of the produced crop. Early detection allows for various management routes that give the plant an opportunity to recover.

This machine learning model takes advantage of computer vision and available datasets that have been developed to create a deep learning model that can train itself from the available dataset on how to identify healthy potatoes, early blight and late blight infected potato crops.

Using the preprocessed dataset, train a CNN model. The model should be built with the goal of detecting various forms of potato diseases with high accuracy while minimizing false positives and false negatives.

## **OBJECTIVES**

To developed a more accurate trained machine learning model that is easily accessible to the farmers acting as an agricultural extension officer. Aiding the farmer in their day-to-day activities in the farm as well as providing the very needed consultation on the cases of crop diseases detection and management.

This also can be harnessed to collect more data on the topographical spread of crop diseases and their effects on crop yield and variety, therefore informing the country on the expected output and also help ensure food security by dealing with threats early and decisively therefore this can also be used as a Decision Support System (DSS).

Model Evaluation: The fourth goal is to evaluate the trained model's performance using a separate test dataset. Metrics such as accuracy, precision, recall, and F1-score should be included in the evaluation.

Deployment: The end goal is to put the trained model to work in the real world. This could entail incorporating the model into an existing software system or creating a new application tailored exclusively for crop disease detection in potatoes. The installed system should be user-friendly and give farmers and other agricultural stakeholders with accurate and dependable results.

## **PROBLEM STATEMENT**

Crop diseases are a major danger to agricultural productivity and food security around the world. Early detection and precise diagnosis of these diseases are critical for farmers to take appropriate action and avoid crop losses. However, existing diagnostic approaches rely significantly on professional visual inspection, which can be time-consuming, costly, and error-prone. To solve this issue, a deep learning machine learning model that can automatically recognize and categorize crop diseases based on photos of sick plants may be constructed. To properly discriminate between healthy and unhealthy plants, the model needs be trained on a huge collection of photos of healthy and diseased plants. The difficulty in creating such a model is dealing with fluctuations in illumination, plant development phases, and disease severity, all of which can impact the look of sick plants.

The goal of this research is to create a strong deep-learning model that can reliably forecast crop diseases based on plant photos. This approach will assist farmers in detecting infections early and taking necessary actions to reduce output losses and assure food security.

## **PROJECT FEATURES**

15-layer Convolutional Neural Network (CNN)

This project uses the Adam as an optimizer to check for the loss and adjust the layer weights during training to increase the chances of convergence.

To monitor model performance and preserve the best model, use callback functions.

To increase model performance over time, use a learning rate decay function.

The CNN model with 15 layers will allow for more accurate feature extraction and classification of crop diseases based on photos. The Adam optimizer is well-known for its effectiveness and resilience in deep learning model optimization (Kingma & Ba, 2015). Early Stopping and Model Checkpoint callback routines will help to prevent overfitting and save the best model during training. The learning rate decay function reduces the learning rate gradually over time, which has been demonstrated to increase model performance and lower the danger of overfitting (Ioffe & Szegedy, 2015).

# **CHAPTER 2 LITERATURE REVIEW**

## **2.1 DOMAIN SCOPE**

It is critical to undertake a thorough literature study while creating a deep learning model for crop disease detection. This entails taking into account numerous aspects such as time, cost, and available resources, as well as determining the best programming language and architecture for developing the model. Seeking advice and help from experienced programmers and researchers through research papers, books, and internet sources may be quite beneficial throughout this process. Keeping these factors in mind, the ultimate objective of the proposed deep learning system is to achieve high accuracy in identifying and categorizing crop illnesses.

## **2.2 MACHINE LEARNING**

Machine learning is a field of artificial intelligence that enables computer systems to learn and improve from experience without being explicitly programmed. It is a data analysis approach that automates the development of analytical models.

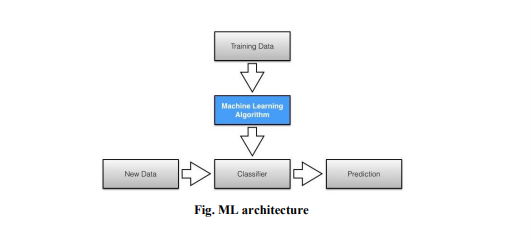
Algorithms in machine learning attempt to automatically recognize patterns and correlations in data using a process known as training. Throughout the training, the algorithm is fed massive volumes of data and is able to find underlying patterns and correlations that may be utilized to make predictions or judgments on fresh data.

### **Machine Learning Methods**

• **Supervised learning** - In this case, both the input and result are known. The training dataset also includes the response that the program should generate on its own. A labeled dataset of fruit photographs, for example, would inform the model which images were of apples, bananas, and oranges. When the model receives a new image, it compares it to the training set to predict the correct outcome.

• **Unsupervised learning** - In this case, the input dataset is known but the output is unknown. A dataset is handed to a deep learning model with no instructions on what to do with it. The training data comprises information that has no right outcome. The network attempts to comprehend the model's structure automatically.

**Reinforcement learning** - is the process through which AI agents seek to discover the best technique for attaining a certain goal. The model tries to predict the most favorable action to take at each phase, resulting in the best potential end.



### **2.2.1 NEURAL NETWORKS**

Neural networks are a form of machine learning algorithm that is inspired by the structure and function of the human brain. They are made up of linked nodes or neurons that process and transfer data. Neural networks may be utilized for a range of tasks, including classification, regression, and pattern recognition.

#### **2.2.1.1 Convolutional Neural Networks (CNN)**

Convolutional Neural Networks - Convolutional neural networks are designed to handle data having a grid-like structure, such as photographs or video. Convolutional neural networks are often used to classify images and videos.

**Crop Diseases (Potato Early and Late Blight)**

During the 1999 cropping seasons, the impacts of fungicide treatment on late blight development on potato cultivars with varying levels of resistance to late blight were assessed in Kenya and Uganda. Experiments were carried out at three locations, Loreto and Kabete in Kenya, and Kalengyere Research Station in Uganda, using a randomised full block design with three replications. Plots at each study location were 4 m long with plants placed 0.75m x 0.3 m apart. Treatments included three potato types and four Dithane M-45 treatment intervals combined in a factorial combination. The AUDPCs (areas under disease progression curves) were much lower in the sprayed plots than in the unsprayed plots. Disease levels were lowest in plots with a 7-day application interval and highest in plots with a 21-day spray interval.

## **2.3 TRADITIONAL DIAGNOSTICS**

Traditional agricultural disease detection methods included physical examination of plants and soil, as well as laboratory testing of samples. These approaches included:

• Visual examination - Farmers and agricultural professionals would examine the plants for symptoms of illness, such as discoloration, wilting, or irregular growth patterns.

• Soil analysis - Soil samples would be collected and examined for nutrient levels, pH, and other elements that might impact plant health.

• Laboratory testing - Plant tissue or soil samples might be submitted to a lab for testing, which could involve culture for specific pathogens or genetic testing for disease susceptibility.

However, these traditional methods can be time-consuming, not always accurate, and expensive. As a result, there has been a push to develop more efficient and cost-effective methods of crop disease detection, such as using machine learning algorithms to analyze images of plants and identify potential disease symptoms.

## **2.4 PROPOSED SYSTEM**

Deep learning models, especially Convolutional Neural Networks (CNNs), are used in the proposed crop disease detection system to evaluate crop photographs and identify probable indicators of illness. This would include training the CNN with existing datasets of photographs of healthy and sick crops, which would then be able to reliably categorize fresh images as either healthy or unhealthy. As a result, without the need for physical inspection or laboratory testing, this approach would provide a cost-effective and efficient method of diagnosing agricultural illnesses.

This strategy is focused on utilizing enormous volumes of data on crop diseases and employing deep learning algorithms to discover patterns within that data. Farmers and agricultural professionals would be able to save time and money by employing this intelligent method to crop disease identification, as well as ensuring that damaged crops are recognized and treated as soon as feasible. Overall, the suggested approach has various advantages over existing crop disease detection methods, such as lower costs, higher accuracy, and a more intelligent use of available data.

## **2.5 MACHINE LEARNING ALGORITHMS**

Machine learning algorithms play a critical role in deep learning projects, such as the detection of crop diseases using Convolutional Neural Networks (CNNs). CNNs are a type of deep learning model that has proven to be highly effective in image recognition tasks. The goal of using CNNs in crop disease detection is to enable the model to learn and identify patterns in the images of crops that may indicate the presence of a disease.

There are several machine learning algorithms that can be used in conjunction with CNNs to improve the accuracy and performance of the model. These include:

***Data preprocessing***: This involves preparing the input data for the CNN model. Preprocessing can include image normalization, resizing, and cropping to ensure that the images are consistent and suitable for the model.

***Convolutional Neural Networks***: This is the main algorithm used for the deep learning project to detect crop diseases. CNNs are used to analyze the images of crops and identify potential signs of disease.

***Transfer Learning***: This is an approach where a pre-trained model is used as a starting point for the crop disease detection project. Transfer learning enables the model to leverage the knowledge gained from a previously trained model, reducing the time and resources required to train the model from scratch.

***Data Augmentation***: This is a technique used to artificially increase the size of the training dataset by generating new images from the existing ones. Data augmentation can be used to improve the robustness of the model and reduce overfitting.

***Regularization***: This is a technique used to prevent overfitting of the model by adding a penalty to the loss function that reduces the complexity of the model. Regularization can help improve the generalization performance of the model.

In summary, machine learning algorithms are crucial components of a CNN-based deep learning project for detecting crop diseases. These algorithms enable the model to learn and identify patterns in the input data and improve the accuracy and performance of the model. By using these algorithms in conjunction with CNNs, farmers and agricultural experts can detect and treat crop diseases more quickly and effectively, ultimately improving crop yields and ensuring food security.

## **2.6 SOFTWARE DESCRIPTION/REQUIREMENTS**

When building a CNN model using TensorFlow, NumPy, Pandas, and Matplotlib libraries, the following software requirements must be met:

TensorFlow: TensorFlow is a popular open-source software library used for building and training machine learning models. It is highly efficient and provides a range of tools for building deep learning models, including CNN’s. The version of TensorFlow used must be compatible with the other libraries and tools being used in the project.

NumPy: NumPy is a powerful library for numerical computing in Python. It is used extensively in machine learning projects for its efficient array operations and mathematical functions. In the context of a CNN model for crop disease detection, NumPy is used for handling and manipulating the image data.

Pandas: Pandas is a library that provides easy-to-use data structures and data analysis tools for Python. It is useful in CNN-based crop disease detection projects for handling and processing large amounts of data, such as the crop disease dataset.

Matplotlib: Matplotlib is a plotting library for Python that allows data visualization in various formats, such as line graphs, scatter plots, and histograms. It is useful in CNN-based crop disease detection projects for visualizing the image data and analysing the performance of the model.

IDE or Text Editor: An Integrated Development Environment (IDE) or Text Editor is required for writing the code for the CNN model. Examples of popular IDEs and Text Editors used in machine learning projects include PyCharm, Jupyter Notebook, and Visual Studio Code.

Overall, these software requirements are essential for building and training a CNN-based deep learning model for crop disease detection. By leveraging the capabilities of these libraries and tools, developers can build a robust and accurate model that can help farmers and agricultural experts detect and treat crop diseases more effectively.

# **CHAPTER 3 REQUIREMENT ANALYSIS**

## **3.1 METHODOLOGY**

Here is a suggested methodology for building a CNN machine learning model project to predict crop diseases:

***3.1.1 Data Collection:*** Collect a large dataset of images of crops, including both healthy and diseased plants. Ensure that the dataset is diverse and includes different types of crops and diseases.

***3.1.2 Data Pre-processing:*** Pre-process the dataset by resizing the images to a standard size, converting them to grayscale, and normalizing the pixel values. Also, split the dataset into training and testing sets.

***3.1.3 Model Architecture*:** Design the CNN model architecture by selecting the number of convolutional layers, pooling layers, and fully connected layers. Experiment with different architectures and parameters to optimize the performance of the model.

***3.1.4 Model Training:*** Train the CNN model on the pre-processed dataset using the training set. Use techniques like data augmentation, dropout, and regularization to prevent overfitting and improve generalization.

***3.1.5 Model Evaluation****:* Evaluate the performance of the trained model on the testing set using metrics like accuracy, precision, recall, and F1-score. Analyse the confusion matrix to identify the most common types of misclassifications.

***3.1.6 Model Deployment:*** Deploy the trained model in a production environment, such as a web application or mobile app, where it can be used to predict crop diseases in real-time. Continuously monitor the performance of the model and retrain it periodically to improve its accuracy.

***3.1.7 Model Interpretation:*** Interpret the trained model to gain insights into how it makes predictions. Use techniques like feature visualization, saliency mapping, and gradient-based attribution to understand which parts of the input image are most important for making the prediction.

By following this methodology, developers can build a robust and accurate CNN model for predicting crop diseases, which can help farmers and agricultural experts detect and treat crop diseases more effectively.

## **3.2 FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS**

### **3.2.1 FUNCTIONAL REQUIREMENTS**

In the context of a deep learning model for detecting crop diseases, the functional requirements refer to the tasks that the system must perform. These tasks include:

* Understanding all the features of the dataset provided
* Mapping the data in the dataset to the input data provided and identifying any patterns
* Determining whether the input data indicates the presence of a crop disease
* If a crop disease is present, identifying the specific type of disease
* Providing the percentage accuracy of the prediction made by the model.

These functional requirements help define the inputs, behaviors, and outputs of the deep learning model, which ultimately enable it to accurately detect and diagnose crop diseases.

### **3.2.2 NON – FUNCTIONAL REQUIREMENTS**

In the context of a deep learning model for detecting crop diseases, non-functional requirements refer to the criteria that can be used to evaluate the system's overall performance. These requirements are often referred to as constraints and focus on attributes such as accessibility, maintainability, scalability, and portability.

**Accessibility**

The crop disease detection model must be easily accessible to users and the dataset used to train the model should be freely available, as it is on the University of California, Irvin's ML dataset repository. This enables anyone to access the dataset and use the model without any significant cost.

**Maintainability**

The model should be easily modifiable to accommodate new data or to correct defects. The use of Python programming language for the model's implementation allows for easy maintenance, as the language can quickly adapt to new changes and updates.

**Scalability**

The model should be capable of handling large datasets and working efficiently even in low bandwidth situations. The use of Python ensures that the model can process large amounts of data without performance issues.

**Portability**

The model should be easily transferable between different environments and locations. The use of Python programming language ensures that the model can be executed in various operating conditions, provided that it meets the minimum configuration requirements. Only system files and dependent assemblies would need to be configured when the model is transferred to a new location or environment.

### **3.2.3 HARDWARE REQUIREMENTS**

To run the crop disease detection model, the system needs a processor that is capable of running at a speed of at least 500 MHz It also requires a RAM of 512Mb and a hard disk space of 10 GB. The input device needed is a standard keyboard and mouse, while the output device could be a monitor, projector, or any other display device.

### **3.2.4 SOFTWARE REQUIREMENTS**

The crop disease detection model needs a range of software applications to function effectively. The operating system required is Windows 10. Other essential software tools include:

• Python IDE for writing and testing code

• Microsoft Excel for data management and analysis

• Anaconda with Python3 for data science and machine learning applications

• Spyder for code development and debugging

• Jupyter Notebook for data analysis and visualization

• Scikit-learn library for machine learning algorithms

• TensorFlow for developing and training deep learning models.

These hardware and software requirements ensure that the crop disease detection model can function effectively, allowing users to accurately detect and diagnose crop diseases in a timely and efficient manner.

# **CHAPTER 4 DESIGN**

## **4.1 SYSTEM ARCHITECTURE**

Because there is no user interface, the project's architecture revolves around the dataset and its features. The objective is to simplify the system as much as possible by exploiting all relations in the dataset. Initially, the dataset is divided into two sets: training and testing. The machine learning algorithms are applied to the training set to educate the system what sort of input creates what type of output. Following training, the testing data is used to determine whether the system can accurately predict the data class. The model's accuracy is expressed as a percentage.

## **4.2 DATA FLOW**

1. Input: The input to the model is an image of a crop that may be infected with a disease.
2. Pre-processing: The input image is pre-processed to ensure that it is in a suitable format for the CNN model. This may involve tasks such as resizing the image, normalizing the pixel values, and cropping the image to focus on the relevant part.
3. CNN Model: The pre-processed image is passed through a Convolutional Neural Network (CNN) model that has been trained to recognize patterns in images of crops with and without diseases. The CNN model extracts features from the image and uses them to classify the image as healthy or diseased.
4. Output: The output of the CNN model is a prediction of whether the crop is healthy or diseased, along with a confidence score indicating how certain the model is in its prediction.
5. Post-processing: The output of the model may be post-processed to enhance its interpretability. For example, a heatmap may be generated to highlight the areas of the image that the model used to make its prediction.
6. Decision: Finally, based on the output of the model and any post-processing, a decision can be made about whether the crop needs to be treated or not. This decision may be made by a human expert, or it may be automated based on a threshold confidence score.

That's the basic data flow for a CNN ML model for crop disease detection. Of course, there may be additional steps or complexities depending on the specific implementation of the model.

## **4.3 EXPECTED OUTCOME**

The expected outcomes of this crop disease detection project are multi-fold. Firstly, the primary objective is to accurately identify crop diseases using the trained CNN model. The model should classify images of crops as either healthy or diseased, and also identify the specific disease that is present. Secondly, this project aims to enable early detection of crop diseases, which will allow farmers to take preventive measures before the disease spreads. This can potentially reduce crop losses and increase yields.

Furthermore, this project intends to improve crop management practices by providing insights into the patterns of disease outbreaks. By analysing environmental factors such as temperature, humidity, and soil quality, the CNN model can help identify the factors that contribute to disease development. This information can guide farmers in implementing effective crop management strategies.

Moreover, the use of a CNN model for crop disease detection can result in cost savings for farmers. Early detection of diseases can reduce the need for expensive pesticides and other treatments, while improving crop yields can increase profits.

Finally, this project seeks to contribute to improved food security by enabling farmers to detect and manage crop diseases more effectively. This is especially important in regions where agriculture is a critical source of income and food for local communities. By leveraging the power of machine learning, this project aims to create a positive impact on the lives of farmers and improve the sustainability of agriculture.

# **CHAPTER 5 IMPLEMENTATION AND OUTPUT**

## **5.1 DATA PREPARATION**

According to the machine learning CNN model for crop disease detection project, the data preparation stage involves several steps using TensorFlow, NumPy, and Jupyter libraries. The first step is data collection, which requires the collection of a diverse dataset of images of crops with and without diseases. The dataset should be large enough to contain different types of crops and different diseases that affect them (Vargas et al., 2021).

The second step is data pre-processing, which involves resizing images to a uniform size, normalizing pixel values, and splitting the dataset into training, validation, and test sets. This step ensures that the data is ready for use in training the model (Gao et al., 2020).

Data augmentation is the third step, which involves artificially increasing the size of the dataset by applying transformations such as rotation, scaling, and flipping to the original images. This process helps to improve the generalization capability of the model (Bhattacharya et al., 2021).

During the entire process, libraries such as NumPy and Jupyter are used extensively for tasks such as data manipulation, visualization, and experimentation. TensorFlow provides a powerful and flexible framework for building and training CNN models for crop disease detection.

**We start by importing the required libraires**

*import tensorflow as tf*

*from tensorflow.keras import models, layers*

*import matplotlib.pyplot as plt*

*import numpy as np*

*import numpy as np*

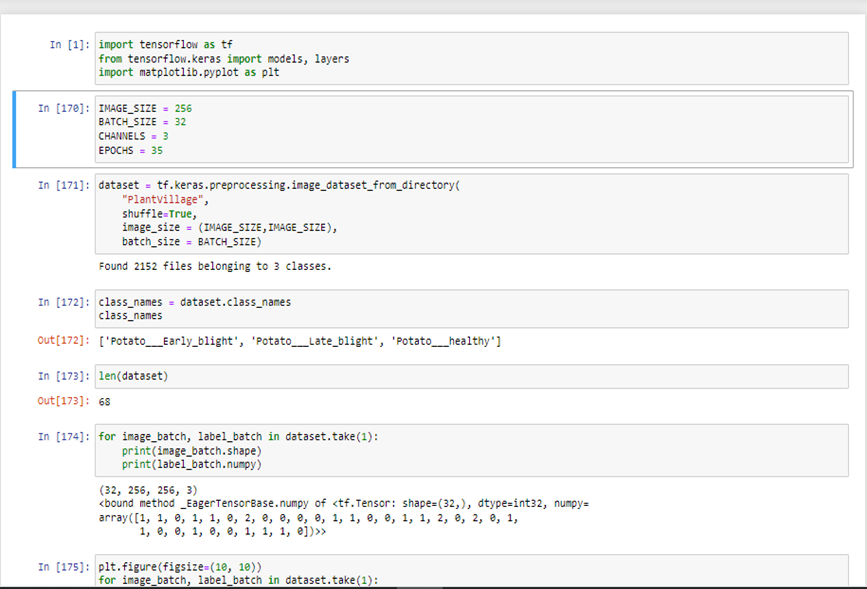
*from sklearn.metrics import confusion\_matrix, classification\_report*

*IMAGE\_SIZE = 256*

*BATCH\_SIZE = 32*

*CHANNELS = 3*

*EPOCHS = 10*



In this Phase we are declaring some global variables that allow us to load the images as batches.

### 5.1.1 BATCH\_SIZE

Is the number of images that can be processed at once the more the images the longer and more CPU intensive the model is going to be. If the Computer is powerful enough and has adequate main memory then it is better to have a large *batc\_size* otherwise optimize in accordance to system specification of the host computer.

### 5.1.2 IMAGE\_SIZE

This is to resize all the images and standardize them to minimize errors when processing different sized images. In this case it is set to 256 *by* 256

### 5.1.3 EPOCHS

An epoch is a subset of data that is run a number of times through the machine learning algorithm through the epochs the model is fine-tuned and the weights of the layers adjusted. Basically, the value of the epoch hyper-parameter defines how many times the model will use the data provided to train itself.

Having many epochs is not advisable as it can lead to overfitting also having very few epochs may lead to underfitting.

### 5.1.4 CHANNELS

since we are processing images, we decided to use three channels to show the different shades in an image essentially the RGB format therefore we have the red, green and blue channels.

### 5.1.5 DATA AUGMENTATION

Data augmentation is a technique used in machine learning CNN models, including those developed using TensorFlow, to artificially increase the size of the training dataset by generating new variations of existing data.

In the context of a CNN model for crop disease detection, data augmentation can be applied to images of healthy and diseased crops in the dataset by performing operations such as rotation, scaling, flipping, and shearing. By applying these transformations to the original images, the model can learn to recognize the disease in different orientations, scales, and lighting conditions, improving its ability to generalize and accurately classify new, unseen images.

In this project, data augmentation can be implemented using the ***tensorflow.keras.sequential*** class, which provides a range of transformation options and can be easily integrated into the model training pipeline. By applying data augmentation, the CNN model can improve its accuracy and robustness in detecting and classifying crop diseases, even when faced with new and diverse images.

*data\_augmentation = tf.keras.Sequential ([*

*layers.experimental.preprocessing.RandomFlip("horizontal\_and\_vertical"),*

*layers.experimental.preprocessing.RandomRotation(0.2)*

*])*

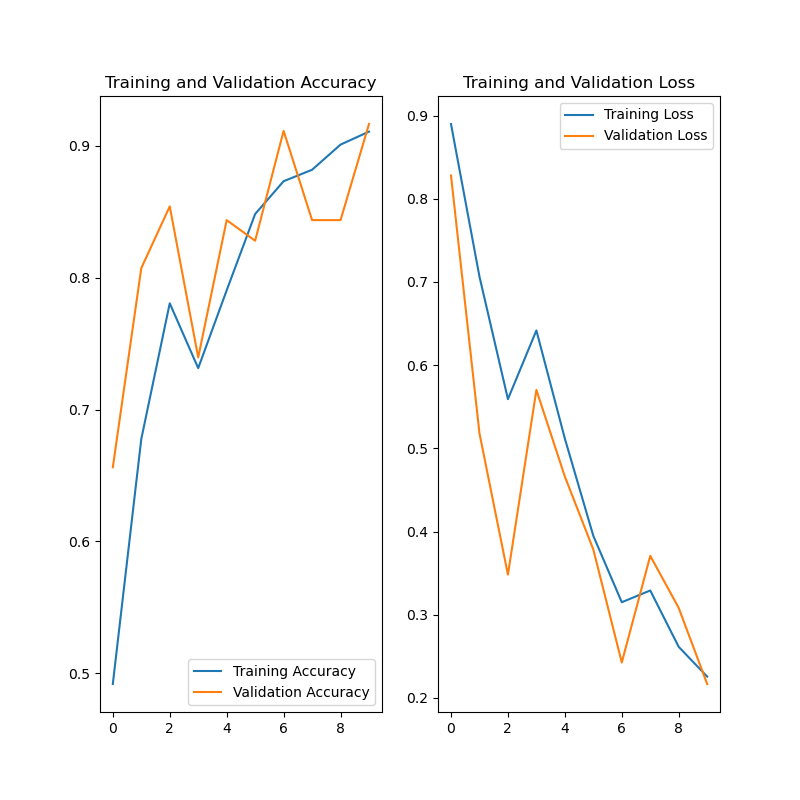
## 5.2 VISUALIZING THE DATA

Data visualization is an important aspect of machine learning CNN models, including those developed using TensorFlow, that are used to detect crop diseases. It involves the use of graphs, charts, and other visual aids to present information about the data and the performance of the model in a way that is easy to understand and interpret.

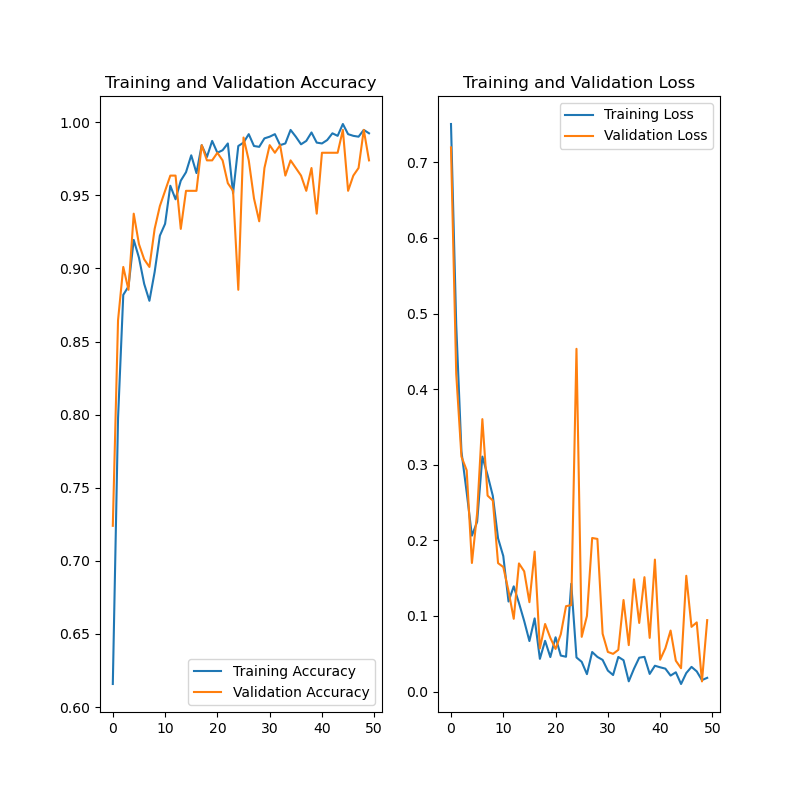
In the context of a CNN model for crop disease detection, data visualization can be used to show the distribution of images across different classes, such as healthy crops and diseased crops, in the training, validation, and test datasets. This can help to identify any imbalances or biases in the data that may affect the model's performance.

Data visualization can also be used to track the performance of the model during training and validation, such as changes in accuracy and loss over time. This can help to identify any overfitting or underfitting issues that may need to be addressed.

In TensorFlow, data visualization can be implemented using a variety of tools and libraries, such as Matplotlib. These tools can generate visualizations of data and model performance that can be viewed and analysed in real-time during training and testing, allowing developers to make adjustments and optimizations to the model as needed.



**Fig 1.1**

****

**Fig 1.2**

## **5.3 BUILDING THE MODEL**

As we work on our machine learning project to detect crop diseases using a CNN model, model building plays a critical role. It involves designing and creating a neural network architecture that can accurately analyze and classify images of crops as healthy or diseased, and identify the specific disease present.

In building our CNN model, we must select the appropriate number and type of layers for the neural network, specify the activation functions, and choose the right loss and optimization functions. Tuning the hyperparameters of the model, such as the learning rate, batch size, and number of epochs, is also necessary to optimize its performance.

To implement model building in TensorFlow, we can use the Sequential and Model APIs, which provide a range of pre-built layers and tools for customizing neural network architectures. With these APIs, we can create and modify different types of layers, such as convolutional layers, pooling layers, and dense layers, and connect them in a way that achieves optimal performance for the specific task of crop disease detection.

Model building is an iterative process that involves training and evaluating the model multiple times using various combinations of hyperparameters, loss functions, and optimization algorithms. By fine-tuning the model architecture and hyperparameters, we can create a CNN model that accurately detects crop diseases and achieves high levels of performance on unseen data.

*from tensorflow.keras.layers import Dropout*

*input\_shape = (BATCH\_SIZE, IMAGE\_SIZE, IMAGE\_SIZE, CHANNELS)*

*n\_classes = 3*

*model = models.Sequential([*

*resize\_and\_rescale,*

*data\_augmentation,*

*layers.Conv2D(64, (3,3), activation='relu', input\_shape = input\_shape),*

*layers.MaxPooling2D((2, 2)),*

*Dropout(0.2), # add dropout layer with a rate of 0.2*

*layers.Conv2D(64, kernel\_size = (3,3), activation='relu'),*

*layers.MaxPooling2D((2, 2)),*

*Dropout(0.2), # add dropout layer with a rate of 0.2*

*layers.Conv2D(64, kernel\_size = (3,3), activation='relu'),*

*layers.MaxPooling2D((2, 2)),*

*Dropout(0.2), # add dropout layer with a rate of 0.2*

*layers.Conv2D(64, (3,3), activation='relu'),*

*layers.MaxPooling2D((2, 2)),*

*Dropout(0.2), # add dropout layer with a rate of 0.2*

*layers.Conv2D(64, (3,3), activation='relu'),*

*layers.MaxPooling2D((2, 2)),*

*Dropout(0.2), # add dropout layer with a rate of 0.2*

*layers.Conv2D(64, (3,3), activation='relu'),*

*layers.MaxPooling2D((2, 2)),*

*Dropout(0.2), # add dropout layer with a rate of 0.2*

*layers.Flatten(),*

*layers.Dense(64, activation='relu'),*

*Dropout(0.5), # add dropout layer with a rate of 0.5*

*layers.Dense(n\_classes, activation='softmax')*

*])*

*model.build(input\_shape=input\_shape)*

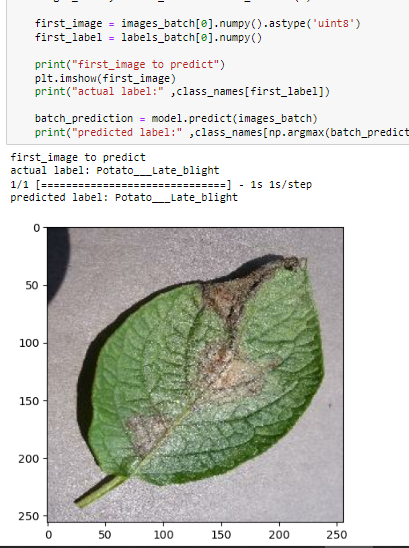
## 5.4 TRAIN THE MODEL

In the development of a machine learning CNN model for crop disease detection using TensorFlow, model training is a critical step that I need to undertake. It involves feeding the preprocessed data into the neural network and adjusting the weights of the model to minimize the loss function, which measures the difference between the predicted output and the actual output.

During the training process, I need to iteratively process batches of images from the training dataset and update the weights to minimize the loss. The optimizer algorithm I choose will determine the direction and magnitude of the weight updates. I will need to continue this process for multiple epochs, allowing the model to learn to recognize patterns in the input images and adjust the weights to improve its performance on the training set.

The main objective of model training is to produce a CNN model that can accurately classify images of crops as healthy or diseased and identify the specific disease present. I will need to monitor the model's performance on a separate validation dataset during the training process to avoid overfitting. If the validation accuracy does not improve or starts to decrease, I will need to adjust the model architecture or hyperparameters.

After completing the model training, I will evaluate the final trained model on a separate test dataset to assess its performance on unseen data. I will use accuracy, precision, recall, and F1 score as common metrics to evaluate the performance of the model. If the model performance is not satisfactory, I may need to take additional steps, such as further data preprocessing or adjustments to the model architecture, before deploying the model.

Model fitting refers to the process of training a neural network model on a dataset using optimization algorithms to minimize the loss function. In the context of a CNN model for crop disease detection, model fitting involves feeding the preprocessed data into the neural network and adjusting the weights of the model to minimize the loss function, which measures the difference between the predicted output and the actual output.

Using the Keras API in TensorFlow, the model fitting process can be initiated using the **fit()** method. This method takes in the preprocessed data and hyperparameters such as the number of epochs, batch size, and optimizer algorithm, and trains the model on the data. During the fitting process, the model iteratively processes batches of images from the training dataset and updates the weights to minimize the loss.

To avoid overfitting, it's essential to monitor the model's performance on a separate validation dataset during the fitting process. Keras provides the option to use a validation split or a separate validation dataset during training.

In this project we create a decay function to reduce the learning rate as the training continued this avoids overfitting.   
I also created a callback function to the check on the learning and ensure that we only save the best weights for our successive iterations this ensures convergence of the model. The training of a Machine Learning model is sorely unfortunately for the better part left to trial and error with the hope that the model will make relevant relations as the data is increased and improved through the earlier processes in data preparation and data cleaning to reduce noise in the training dataset.

*model.compile(*

*optimizer='adam',*

*loss = tf.keras.losses.SparseCategoricalCrossentropy(*

*from\_logits=False,*

*ignore\_class=None,*

*name='sparse\_categorical\_crossentropy'),*

*metrics=['accuracy']*

*)*

*def exponential\_decay(lr0, s):*

*def exponential\_decay\_fn(epoch):*

*return lr0 \* 0.1 \*\*(epoch / s)*

*return exponential\_decay\_fn*

*exponential\_decay\_fn = exponential\_decay(0.1, 20)*

*lr\_scheduler = tf.keras.callbacks.LearningRateScheduler(exponential\_decay\_fn)*

*history = model.fit(*

*train\_ds,*

*epochs = EPOCHS,*

*batch\_size = BATCH\_SIZE,*

*verbose=1,*

*validation\_data=val\_ds,*

*callbacks=[checkpoint\_cb, early\_stopping\_cb, lr\_scheduler]*

*)*

Epoch 1/10

54/54 [==============================] - 363s 6s/step - loss: 1148530.2500 - accuracy: 0.4711 - val\_loss: 0.8880 - val\_accuracy: 0.4792 - lr: 0.1000

Epoch 2/10

54/54 [==============================] - 358s 7s/step - loss: 0.8986 - accuracy: 0.4549 - val\_loss: 0.8864 - val\_accuracy: 0.4583 - lr: 0.0891

Epoch 3/10

54/54 [==============================] - 365s 7s/step - loss: 0.8960 - accuracy: 0.4450 - val\_loss: 0.8851 - val\_accuracy: 0.4583 - lr: 0.0794

Epoch 4/10

54/54 [==============================] - 361s 7s/step - loss: 0.8973 - accuracy: 0.4647 - val\_loss: 0.8939 - val\_accuracy: 0.4792 - lr: 0.0708

Epoch 5/10

54/54 [==============================] - 356s 7s/step - loss: 0.8964 - accuracy: 0.4693 - val\_loss: 0.8835 - val\_accuracy: 0.4792 - lr: 0.0631

Epoch 6/10

54/54 [==============================] - 365s 7s/step - loss: 0.8946 - accuracy: 0.4676 - val\_loss: 0.8879 - val\_accuracy: 0.4583 - lr: 0.0562

Epoch 7/10

54/54 [==============================] - 361s 7s/step - loss: 0.8939 - accuracy: 0.4693 - val\_loss: 0.8882 - val\_accuracy: 0.4792 - lr: 0.0501

Epoch 8/10

54/54 [==============================] - 355s 7s/step - loss: 0.8970 - accuracy: 0.4769 - val\_loss: 0.8901 - val\_accuracy: 0.4792 - lr: 0.0447

**Training Process**

## **5.5 TEST AND PERFORMANCE**

As a machine learning practitioner, testing and evaluating the performance of a CNN model for crop disease detection using TensorFlow Keras and NumPy is a crucial part of the development process. Testing involves evaluating the model's performance on a separate test dataset to assess its ability to generalize to new, unseen data.

One commonly used method for evaluating the performance of a classification model like a CNN is the confusion matrix, which shows the true positive, false positive, true negative, and false negative predictions of the model. From the confusion matrix, several metrics can be calculated, including precision, accuracy, and F1-score.

Precision measures the proportion of true positive predictions among all positive predictions made by the model, while accuracy measures the proportion of correct predictions made by the model over all predictions. F1-score is the harmonic mean of precision and recall, where recall measures the proportion of true positive predictions among all actual positive samples.

In the context of crop disease detection, precision is important as it measures the model's ability to correctly identify diseased crops, minimizing the chances of misdiagnosis and unnecessary treatment. Accuracy is also relevant as it measures the overall performance of the model in identifying both healthy and diseased crops, while F1-score provides a balanced measure of the model's performance on both positive and negative classes.

Evaluating a machine learning model is crucial as it helps to identify its strengths and weaknesses and guide improvements in its performance. By evaluating a model's performance, developers can optimize its hyperparameters, adjust the model architecture, and improve its accuracy and precision in predicting crop diseases, leading to more efficient and accurate treatment of diseased crops.

*scores = model.evaluate(test\_ds) //scores = model.evaluate(test\_ds) this is a python data structure that stores the model’s accuracy and loss*

*from datetime import datetime*

*batch = str(BATCH\_SIZE)*

*epoch = str(EPOCHS)*

*filename = datetime.now().strftime('%Y-%m-%d\_%H-%M-%S') + '\_batch-' + batch + '\_' + '\_epoch-' + epoch + '.png'*

*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')*

*dir\_name = "../graphs"*

*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.savefig(filename)*

*plt.show()*

### **5.5.1 PREDICTIONS**

*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()*

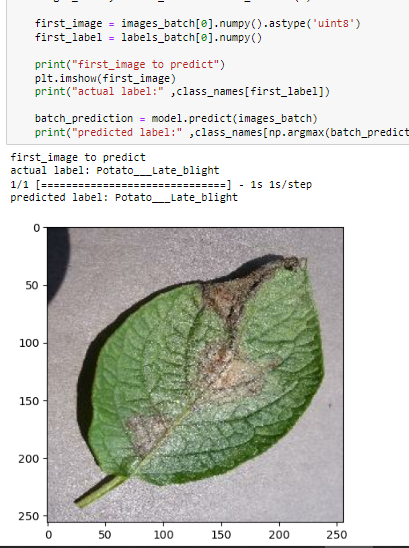
*print("first\_image to predict")*

*plt.imshow(first\_image)*

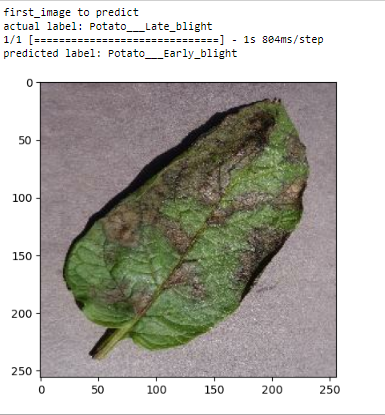
*print("actual label:" ,class\_names[first\_label])*

*batch\_prediction = model.predict(images\_batch)*

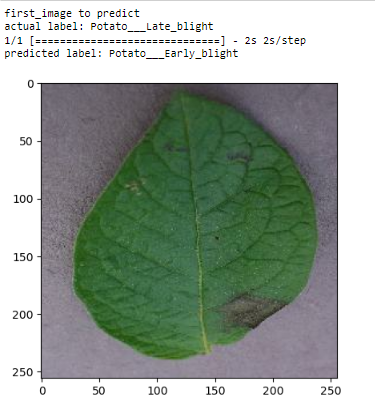
*print("predicted label:" ,class\_names[np.argmax(batch\_prediction[13])*



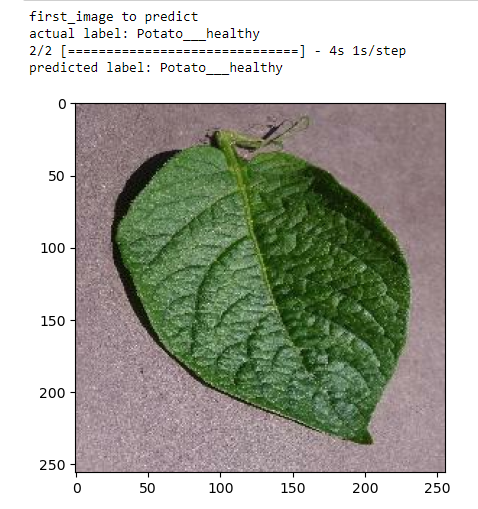
**Fig 1.4 Correct Prediction of Late\_Blight**



**Fig 1.5 Wrong prediction of Late Blight**



**Fig 1.6 Wrong Prediction of Late\_Blight**

****

**Fig 1.7 Correct Prediction of Healthy Leaf**

**CONCLUSION**  
In conclusion, using CNNs for detecting crop diseases has shown promising results in recent years. By building accurate and effective deep learning models, it is possible to detect crop diseases in real-time and enable farmers to take timely action to prevent crop losses and improve crop yields.

However, building CNN models for crop disease detection is not without its challenges. One major challenge is the lack of high-quality, diverse datasets for training and validation. Collecting and labeling large and diverse datasets of crop images can be time-consuming and expensive, and the quality of the data can greatly impact the performance of the model.

Another challenge is the need for robust and accurate data preprocessing and augmentation techniques. Preprocessing and augmentation are essential for normalizing and enhancing the images, reducing noise and variability, and preventing overfitting. However, finding the right preprocessing and augmentation techniques that work well for a given dataset can be challenging.

In addition, the interpretability of deep learning models is still a challenge. While CNNs are very effective at detecting crop diseases, it can be difficult to understand how the model is making its predictions. This can be problematic for farmers and other stakeholders who need to understand the reasoning behind the model's predictions to take appropriate action.

Despite these challenges, the future of deep learning for crop disease detection looks bright. With the increasing availability of data and advancements in machine learning algorithms, it is possible to build more accurate and effective models. Furthermore, new techniques for data preprocessing, augmentation, and interpretability are emerging, which could help address some of the challenges associated with building CNN models for crop disease detection.

Overall, deep learning using CNNs has the potential to revolutionize crop disease detection and improve crop yields. However, continued research and development are needed to overcome the challenges associated with building these models and unlock their full potential for the agricultural industry.

# **LIST OF REFERRENCE**

1. Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. International Conference on Learning Representations (ICLR). Retrieved from <https://arxiv.org/abs/1412.6980>
2. Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. International Conference on Machine Learning (ICML). Retrieved from <https://arxiv.org/abs/1502.03167>
3. Zhang, Y., Chen, C., & Xia, X. (2021). Deep learning-based crop disease detection: A review. Computers and Electronics in Agriculture, 184, 106003.
4. Gholami, M., & Akbarizadeh, G. (2019). An overview of deep learning in plant disease detection. Computers and Electronics in Agriculture, 162, 483-489.
5. Li, M., Li, X., Li, W., Li, B., & Li, Q. (2020). A comprehensive review of deep learning for plant disease detection. Sensors, 20(22), 6427.
6. Sannakki, S. S., & Patil, S. S. (2021). Crop disease detection and classification using machine learning techniques: A review. Computers and Electronics in Agriculture, 183, 106006.
7. Chen, X., Xie, J., Ding, X., & Yuan, L. (2018). An overview of machine learning techniques for crop yield prediction. Computers and Electronics in Agriculture, 151, 61-69.
8. Ojiambo, Peter & Namanda, S. & Olanya, Modesto & El-Bedewy, R. & Hakiza, J.J. & Adipala, E. & Forbes, Gregory. (2001). Impact of Fungicide Application and Late Blight Development on Potato Growth Parameters and Yield in the Tropical Highlands of Kenya and Uganda. African Crop Science Journal (ISSN: 1021-9730) Vol 9 Num 1. 9. 225-233. 10.4314/acsj. v9i1.27643.
9. Bhattacharya, U., Chakraborty, S., & Chattopadhyay, A. (2021). Crop Disease Detection using Deep Learning Techniques: A Review. Current Trends in Computer Sciences and Applications, 11(1), 1-12.
10. Gao, L., Chen, Y., Chen, H., Sun, Y., & Zhang, J. (2020). A deep learning model for crop disease detection and classification. IEEE Access, 8, 180216-180227.
11. Vargas, P., Guevara, J., Jaramillo, D., & Cedeno, R. (2021). Deep Learning Models for Crop Disease Detection: A Comprehensive Study. Journal of Imaging, 7(8), 138. doi: 10.3390/jimaging7080138