Sudan University of Science And Technology

College of Engineering

Electronics Engineering Department

Crop Health Monitoring Using Convolutional Neural Network (CNN)

A Research Submitted In Partial fulfillment for the Requirements of the Degree of B.Sc. in Electronics Engineering

Prepared By:

1- **Abdalkareem Abdalmajed Mohmed Othman**

2- **Khalid Elwalid Omer Hag Elzaki**

3- **Mohammed Osama Babiker**

4- **Omer Ibrahim Ali**

Supervised By :

Prof. Dr. Rashid Saeed

Feb/2025

إستهلال

قال تعالى:

(وَفَوْقَ كُلِّ ذِي عِلْمٍ عَلِيم)

سورة يوسف (76)

DEDICATION

To our beloved families for their support, to our mentors for their guidance, despite the challenges the country is going through.

ACKNOWLEDGMENT

The completion of this research would have been impossible without the assistance of everyone who supported and contributed towards it. First of all, We express our very great gratitude to our academic advisor [**Prof.Dr.Rashid Saeed**] who was there throughout the project, supporting us in many ways, throwing light through thought-provoking comments, and cheering us on to the end.

The staff at [**SUDAN University of Science and Technology**] gave foundational aid that helped us an awful lot. Further, We extend special gratitude to our friends throughout the center, all of whom contributed expertise for the betterment of the project in one way or another.

In addition, We'd like to extend wishes of gratitude to the open-source community that supplied basics needed for producing underlying datasets used and software that would not have been possible whatsoever without this assistance.

Finally, We further thank our aunts, parents, and siblings for their never-ending support, tolerance, and belief in our work.

ABSTRACT

Crop diseases threaten global food security, requiring rapid and accurate detection to maintain agricultural productivity. Traditional methods for monitoring crop health are costly and inefficient. This research explores the use of convolutional neural network (CNN), a type of deep learning algorithm, to automate plant disease detection and classification through image analysis. Leveraging publicly available datasets of plant leaf images, a robust model is developed and trained to recognize different stages of crop diseases. The CNN model is improved using techniques such as data augmentation and evaluated through metrics such as precision, accuracy, recall, and F1 score.

The results demonstrate the potential of CNN to improve the accuracy and effectiveness of disease detection, contributing to sustainable agriculture and improved crop management practices. Limitations and future work include expanding the dataset, improving real-time data processing, using CNN fast Fourier transform, increasing the used layers for training, and generalizing the model to a wider range of crops and diseases.

Keywords: Crop Health Monitoring ,Convolutional Neural network (CNN),Image Classification, Plant Disease Detection.

الخلاصة

تهدد أمراض المحاصيل الأمن الغذائي العالمي، وتتطلب الكشف السريع والدقيق للحفاظ على الإنتاجية الزراعية. الطرق التقليدية لمراقبة صحة المحاصيل مكلفة وغير فعالة. يستكشف هذا البحث استخدام الشبكات العصبية التلافيفية CNN، وهو نوع من خوارزميات التعلم العميق، لأتمتة اكتشاف أمراض النبات وتصنيفها من خلال تحليل الصور. من خلال الاستفادة من مجموعات البيانات المتاحة للجمهور لصور أوراق النبات، يتم تطوير نموذج قوي وتدريبه للتعرف على مراحل مختلفة من أمراض المحاصيل. يتم تحسين نموذج CNN باستخدام تقنيات مثل زيادة البيانات وتقييمه من خلال مقاييس مثل الدقة والدقة والتذكير ودرجة F1. توضح النتائج إمكانات CNN لتحسين دقة وفعالية اكتشاف الأمراض، والمساهمة في الزراعة المستدامة وتحسين ممارسات إدارة المحاصيل. تشمل القيود والأعمال المستقبلية توسيع مجموعة البيانات، وتحسين معالجة البيانات في الوقت الفعلي، واستخدام تحويل فوري و سريعCNN ، وزيادة الطبقات المستخدمة للتدريب، وتعميم النموذج على مجموعة أوسع من المحاصيل والأمراض.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER ONE

INTRODUCTION

1.1 Overview

As the global population grows, food production depends on advances in farming to continue growing. Sustainable farming is essential. Plants are sources of food and air that makes the environmental quality better and improving the soil quality. Yet, crop diseases can easily affect agricultural productivity. It can affect the quality and quantity in harvest. Traditionally, monitoring crop health and the detection of diseased crops involves checks by experts. But, it is time-consuming, expensive, and requires special skills.

To overcome the obesity and malnourishment limitations, technology, especially Precision Agriculture (PA) technology, has been gaining prominence in the agri-food sector. PA uses the powers of computer vision, image processing, and deep learning to improve all farming practices, including land preparation, monitoring plant health, and detecting plants diseases (Mohanty et al., 2016)[1]. Among these methods, convolutional neural network helps to automatically detect and classify plant diseases effectively (Ferentinos, 2018)[2]. Convolution neural network can examine undeveloped images of plant leaves to identify and classify diseases images without requiring manual feature extraction. Using automated systems powered by large datasets and accelerated through (GPUs and High-Performance Computing (HPC), it has the potential to enhance the precision and efficiency of crop disease diagnostic.

The current work is related to the study of deep learning in general and, specifically, CNN for automatic plant disease classification across various crops. The various technologies being leveraged in the study will eventually help in the model development that shall monitor and manage crop health efficiently, thus addressing the bigger challenge of sustainable agriculture.

1.2 Problem Statement:

Diseases can seriously threaten food security and agricultural productivity across the world. To manage and control the diseases, their timely and accurate detection is essential. But, old methods of diagnosing crop disease are labor intensive, time-consuming, expensive and require expertise. Moreover, they may not work for early detection of disease or for testing on a larger agricultural scale (Sladojevic et al., 2016)[3].

In the last few years a lot of research has done using machine learning, deep learning and remote sensing techniques for better detection and classification of crop diseases. But there be quite a few issues when we apply these techs which include the need for large annotation data sets, high variability in disease symptoms and more (Too et al., 2019)[4].

The main objective of the research is to create and test innovative ways of detecting and classifying crop disease at early stages. This task involves tackling several challenges, including building algorithms to analyse images, creating large datasets of samples, and integrating data to improve disease management. The purpose of this study is to improve the existing system for detecting disease to lessen the manual effort in the future. Ultimately, this will help in increasing the health of crops and overall agricultural productivity.

1.3 Aim and Objectives

This research, therefore, aims to develop a structured and efficient methodology for crop monitoring using CNN, which will lead to the detection of diseases and stress factors with a view to improving agricultural productivity and sustainability.

1.3.1 Objectives :

1-To collect and preprocess diverse crop images.

2-To use the performance metrics and optimization techniques like data augmentation and transfer learning to evaluate and fine tune the model.

3-To Build a CNN based model for identifying accurate crop health.

4-To Build a real time crop monitoring mobile or web app for users to use

1.4 Scope

The problem concerns the development of Convolutional Neural network (CNN) for real-time monitoring and assessment of crop health with an emphasis on the diagnosis of diseases from leaves alone. The project will also include choosing appropriate crops and their corresponding diseases as well as gathering, cleaning, and annotating data, creating and training convolutional neural network (CNN), and developing a proof-of-concept application that would utilize these models in practice.

This project does not include the capture of audio video data in the fields on the instantaneous basis, the viewing of diseases in the context of other plant organs (like stems or roots), or the effects of external environmental parameters like soil type, ecosystem, and weather, and pest infestations. Also, the limitations are practical with respect to the existence of sufficient quality images and requirements of resource for computation for learning the deep models. Additionally, the transferability of the results achieved by the model may be only to the diseases of and crops included in the first dataset which restricts its use for other crops and diseases.

1.5 Methodology

This research targets establishing a powerful system of identification and classification of plant diseases with the help of Convolutional Neural network (CNN). The study is organized into three main phases: the first phase is the phase of gathering and preparing the data set, the second phase depicts the design and training of a CNN model and the third one is the evaluation phase which includes prototyping.

In the first phase, for this Research, a large number of pixel data was obtained from publicly available leaf images databases for agriculture. This dataset contains images depicting different species of crops and corresponding diseases of plants hence ensuring ample data for model training. The dataset that was collected was subjected to thorough preprocessing in order to enhance its quality and usefulness. Such procedures included purging the database of duplicate or non-essential images and bringing the labeling into coherence.

Equalization of different disease categories was also included. Images were standardized through rescaling the images to a certain degree of resolution and/or normalizing the pixel values. After these steps, further measures were taken to enhance the effects of trained models including several techniques of data augmentations such as rotations, flips and change of color values.

In the second step, CNN models were developed through the use of Python programming language altogether with Tensorflow and Keras deep learning frameworks. The architecture was formulated in a way that will allow capturing distinct characteristics from the leaf images upwards, involving several of convolutional and pooling layers as well as activation functions. The training of the models was done using some portions of the prepared dataset and hyper parameters although there were limits such as the learning rate, batch size, and number of epochs which were optimized over various iterations. The training was performed on resources with high performance such as GPUs so as to be able to deal with the resource-intensive utilization of training the model on a large scale. The testing was also performed on the developed models whereby other metrics were used such as accuracy, precision, recall, and F1 score so as to make sure that the models were useful in the classification of the plant diseases and made predictions on new data.

In the final phase, the CNN models were rigorously evaluated using a separate test dataset to assess their accuracy and reliability in practical scenarios. Evaluation metrics were thoroughly analyzed to determine model performance and identify areas for improvement. Additionally, a prototype application was developed to demonstrate the practical utility of the trained models in agricultural settings. This application integrated the CNN models to provide actionable insights and support decision-making in crop health management. The prototype serves as a validation tool for the CNN-based approach and showcases its potential for real-world implementation.

1.6 Thesis Outline

The organization of the work is ensured in the following way:

Chapter 1: Introduction : In this chapter the readers are acquainted with the core of the research, they understand exactly what the challenge is, what the aims and objectives are, who and what is needed for this to be accomplished, what approach was taken, and the structure of the thesis.

Chapter 2: Literature Review : Covers the literature on the general aspects on crop monitoring, provides a review of the works done on imaging for the purposes of identifying diseases in plants and stating what is lacking in these area.

Chapter 3: Methodology : Focuses on how the research was planned in terms of the collection of data, how such data was prepared for analysis, how models were created and the criteria used to assess their performance.

Chapter 4: Results and Discussion : Discusses the results obtained including classifiers accuracy and applicability of proper n of the sequential models for the crops health monitoring systems.

Chapter 5: Conclusion and Future Work : Restates the essential aspects of the research conducted, assesses the results of the work done in this case the most important ones, and suggests other possible lines of investigation for the follow up research.

CHAPTER TWO

LITERATURE REVIEW

2.1 TECHNICAL BACKGROUND

2.1.1 Precision agriculture

Field-level management for crop farming depends on site-specific crop management technology which professionals call precision agriculture and satellite farming. (Ahmad et al. 2024)[5] .

To achieve sustainable agriculture, Precision agriculture (PA) is used and it is the technology that enhances farming techniques. It prepares the land before planting, ensures an equally fertile vegetation across the field, monitors plantations during in-season growth, detects of early onset of pest and diseases, right amount, right time and right application of farm input resources to the right location through harvest and postharvest processes . PA involves remotes sensing, the use of Geographical Information System (GIS), Global Positioning System (GPS) and data analysis. This technology enables the collection of field data in a known- destructive way and making the data available for analysis and implementation on the field .

The technology is made necessary due to a spatial and temporal variability of the field revealing information as, patterns and spatial relationships. The stages of remote sensing involve energy interaction with the atmosphere and interaction with the target, then the interaction of the energy with the sensor (camera), data transmission and processing, image analysis and finally the application of results to treatment areas required .

Simple applications on the farm involve determining the location of sampling sites, plotting maps for use in the field, examining the distribution of soil types in relation to yields and productivity. Other applications take advantage of the analytical capabilities of GIS and RS software for vegetation classification to predict crop yield, environmental impacts, modeling of surface water drainage patterns, tracking animal migration patterns and other ranges of applications (Abdullah et al. 2017)[6].

PA techniques are employed to increase yield, reduce production costs, and pressure on the environment.

2.1.2 Remote sensing

Remote sensing refers to the method of collecting information about an object or phenomenon without direct physical contact. Engineers use remote sensing as a data collection method that operates from distances by deploying sensor technologies on both satellites and aircraft.

2.1.2.1 Main types of remote sensing:

Passive Remote Sensing: relies on natural energy which includes sunlight both in its reflective and emitted forms for object observation. Photography along with infrared imaging represent typical instances of passive remote sensing.

Active Remote Sensing: sends energy towards objects for measuring the energy which bounces back from the observation target. The most typical examples of these systems are RADAR and LiDAR.

Table 2.1 Sentinel-2 satellite channels used

Spectral Area Band-Use	Wavelength (nm)	Description
Band 1 – Coastal aerosol	443.9 - 479.1	This range can be used to detect chlorophyll in plants and assess their health. Changes in the spectral signature of this band may indicate stress, including disease and pests.
Band 2 – Blue	490.6 - 516.5	This range is also used to analyze chlorophyll in plants and can help determine the health of crops
Band 3 – Green	559.8 - 597.6	The green stripe is used to assess the condition and health of plants. Changes in this spectral range may also indicate stress in crops
Band 4 – Red	650.0 - 680.0	The red spectral range is useful for analyzing plant pigmentation and assessing their physiological state
Band 5 – Vegetation red edge	698.5 - 712.5	This range can be used to analyze the condition of plants and identify stress factors
Band 6 – Vegetation red edge	733.5 - 747.5	This range can help identify diseases and stress in plants
Band 7 – Vegetation red edge	773.1 - 793.1	This range is also useful for plant health analysis and disease detection
Band 8 – NIR	793-842	Primary applications include vegetation health analysis and assessment of plant biophysical parameters

2.1.2.2 Sentinel-2

Sentinel-2 is a space-based Earth observation system mission that provides data in a variety of spectral bands listed in (Table 2.1). The Sentinel-2 Vegetation Red Edge channels are positioned in the red edge region of the spectrum, which is between the Red and Near Infrared regions. This spectral range is particularly useful with regard to the assessment of the state of vegetation and identification of changes in crops associated with infestations and diseases.

Through Vegetation Red Edge channels one can monitor vegetation health to detect chlorophyll changes which indicate plant health and stress conditions. Any infestation that affects plants will change their spectral characteristics which allows detection of their presence. Vegetation conditions on Earth's surface get monitored through the analysis of spectral data from the Vegetation Red Edge channel combined with vegetation index metrics including NDVI as well as NDWI and SAVI and ARVI and EVI and GNDVI.

These indices are based on the assessment of health, density, and moisture content among other vegetation characteristics, and involve the use of remote sensing spectral data. For more information about the indices, refer to (Tussupov et al. 2024) [7] . Analysis of Formal Concepts for Verification of Pests and Diseases of Crops Using Machine Learning Methods. IEEE Access, 12, pp. 19902–19910.].

The general model for verifying pests, for instance, provides a clear, step-by-step procedure in identifying and confirming the presence of the pest within the crops.

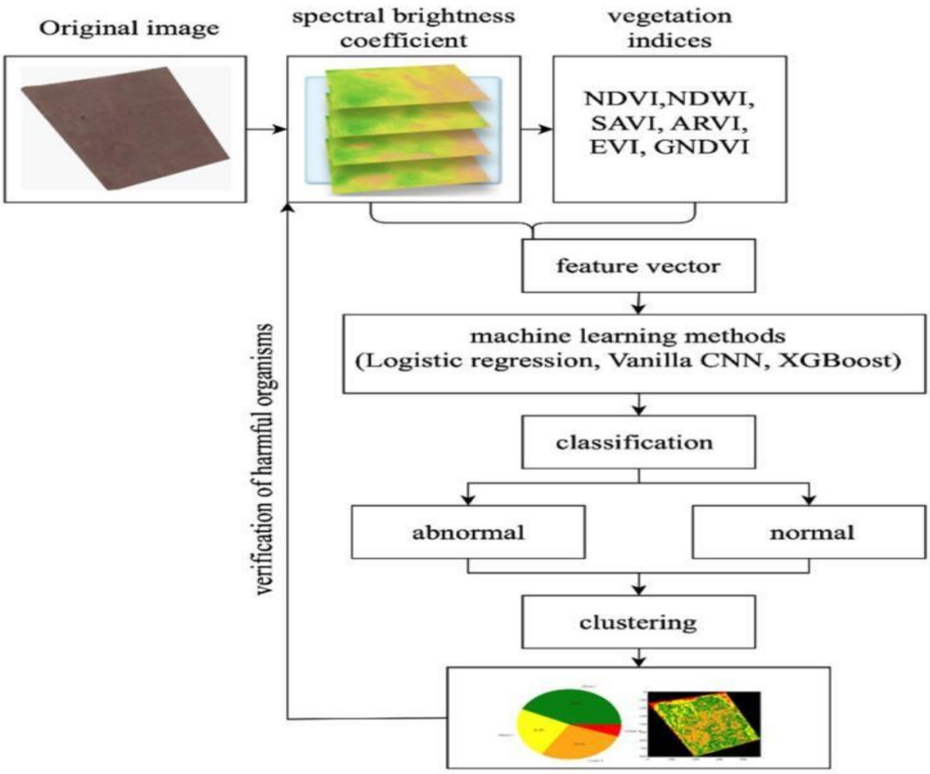

Figure 2.1 Malicious organism verification model

Multiple sequential steps along with diverse techniques are used in this method to achieve both efficiency and accuracy in the process. The method begins with Sentinel-2 satellite data along with other sources to study vegetation spectral properties. The evaluation seeks to detect irregularities linked to dangerous microorganisms. Database integration of pest and pathogen data takes a central role in the process of data collection and analysis. Machine learning algorithms including classification and clustering along with neural network function as automatic tools to analyze spectral data.

The algorithms detect specific traits which point towards potential pest occurrences. Picture processing methods serve as a standard tool for scientists to locate regions where pests could cause damage. The data verification process receives support from expert systems alongside the data classification methods. The systems validate machine learning algorithm results by incorporating expert knowledge and contextual information. A final system emerges for automated choices that utilize the findings from data evaluation.

Spectral radiance coefficients represent numerical values used to explain the interactions between objects or surfaces and electromagentic radiation throughout the spectrum. The identification of weeds versus crop species can be achieved through analyzing reflected or emitted light using spectral characteristics. (Ahmad et al. 2024) [5].

2.1.3 Digital image processing

Computer algorithms perform digital image processing operations on digital images through a set of programmed instructions..

2.1.3.1 Basics of Digital Image Processing:

Image Acquisition: Digital devices like cameras or scanners utilize the process to record images. A computer utilizes digital format to process images that result from capturing.

Image Enhancement: The application of techniques enables users to improve image visual quality or transform images into analysis-ready formats. The process includes modifying brightness together with contrast and eliminating noise from the image.

Image Restoration: Image enhancement requires the removal of both distortions and degradations to improve visual appearance. The two popular image enhancement techniques for distortion and degradation elimination are deploring and delousing.

Image Compression: The technique reduces image file size yet preserves its quality levels effectively. Images need efficient storage and transmission mechanisms where this method plays a crucial role.

Image Segmentation: Dividing an image into its constituent parts or objects. This is often used in medical imaging to isolate regions of interest, such as tumors.

Image Representation and Description: The processed image requires conversion into a form that enables further analysis of features and pattern recognition. (Mohammed 2020)[7].

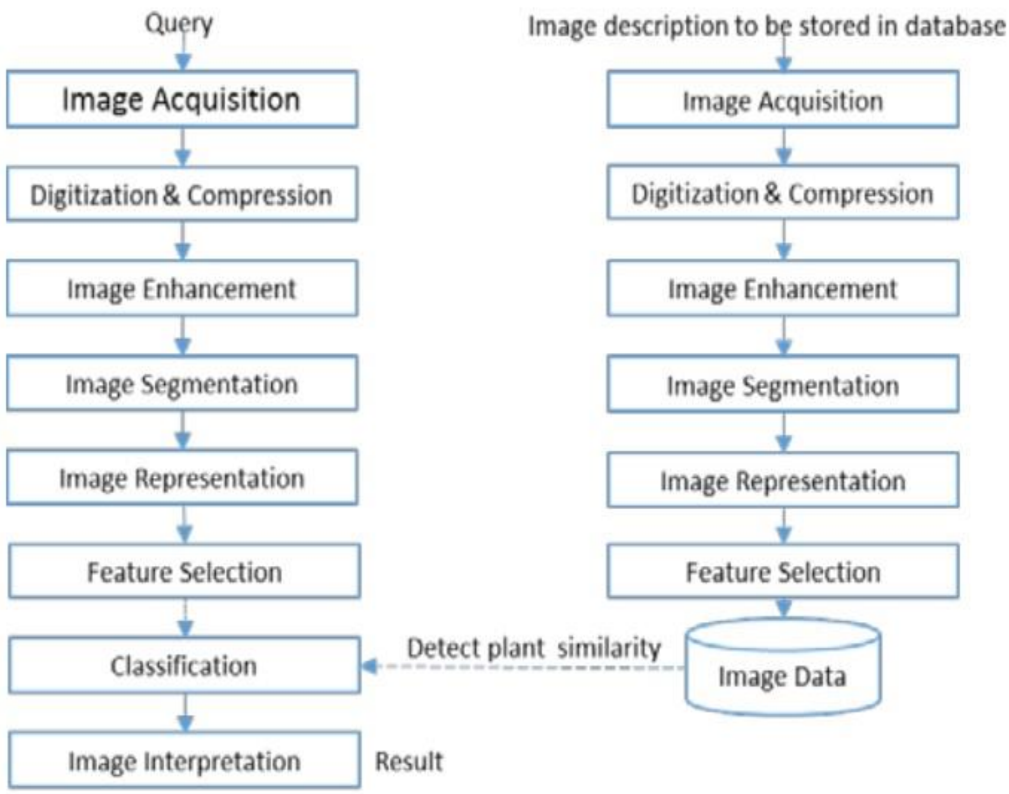

Figure 2.2 Stages of Image analysis (Abdullah et al. 2017)

Digital image processing is used in various fields, including medical imaging, remote sensing, industrial inspection, and more.

2.1.4 Artificial Intelligence (AI)

2.1.4.1 Definition:

The field of AI functions to develop systems that execute duties which need human mental capabilities. AI performs five fundamental operations consisting of reasoning alongside learning, problem-solving and perception with language understanding capabilities.

2.1.4.2 Scope:

Artificial Intelligence contains multiple technological approaches such as rule-based systems and expert systems and neural network.2.1.4.3

Applications:

AI is used in various applications such as natural language processing (e.g., chatbots), robotics, and autonomous vehicles. (Fonseca et al. 2019)[8].

2.1.5 Machine Learning (ML)

Machine Learning (ML) is a subset of artificial intelligence (AI) that focuses on developing algorithms that enable computers to learn from and make decisions based on data.

2.1.5.1 Types of Learning:

Supervised Learning: The training process requires labeled information where each input example receives a specific output label. Linear regression together with decision trees and support vector machines represent commonly used algorithms. (Kubat 2021)[9].

Unsupervised Learning: The algorithm is used to find patterns and relationships in data without any labeled responses. Examples include clustering algorithms like k-means and hierarchical clustering.(Bhattacharyya et al. 2019)[10].

Reinforcement Learning: An algorithm operates by taking part in environmental activities through which it gets either positive feedback

or negative penalties. The algorithm applies this process in robotics and game playing applications. (Nilsson 1998)[11] .

2.1.5.2 Features and Labels: The input variables known as features help predict output variables labeled as labels.

2.1.5.3 Training and Testing: The dataset gets divided into two sections for model training through the training set and performance evaluation through the testing set.

2.1.5.4 Model Evaluation: A machine learning model gets evaluated using accuracy and precision together with recall and the F1 score (Kubat 2021)[10].

2.1.5.5 The Difference Between AI And ML:

Breadth: The concept of Artificial Intelligence (AI) includes multiple processes and technologies but Machine Learning (ML) represents its implementation of algorithms to analyze data through applied AI.

Functionality: The purpose of AI is to develop the capability to execute virtually all intelligent deeds. In contrast, the purpose of ML is to make it possible for systems to learn from data and become progressively more efficient.

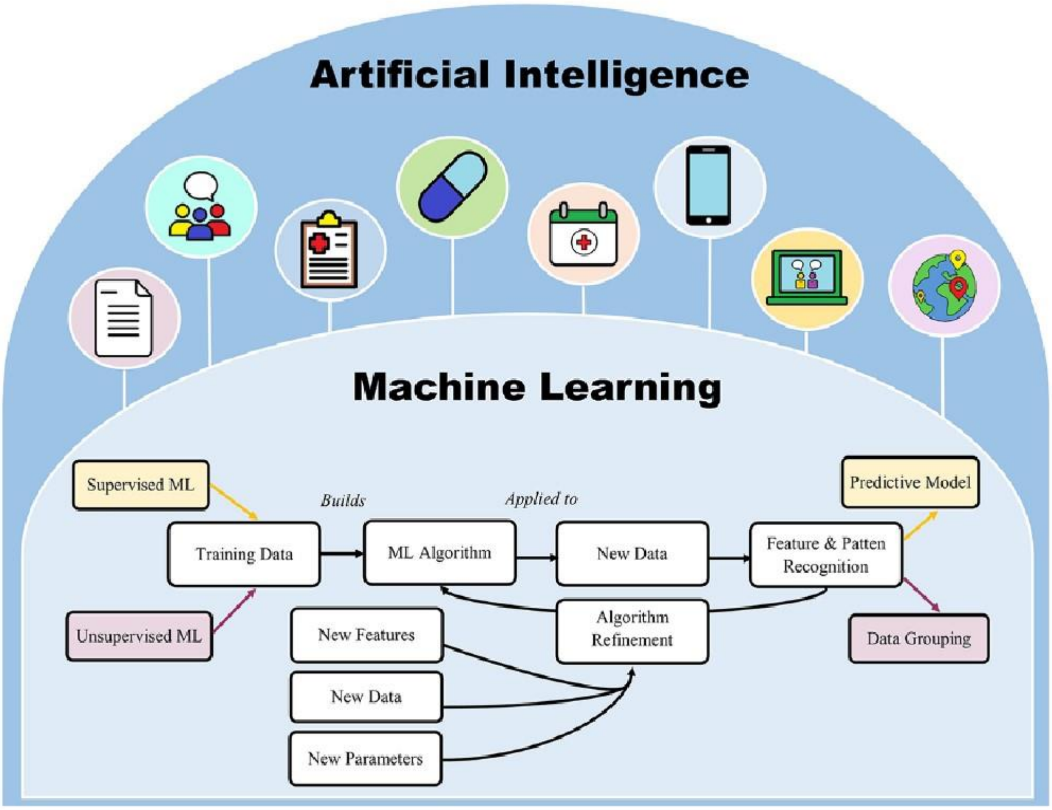

Figure 2.3 The relationship between machine learning and artificial intelligence. Differences (Fonseca et al. 2019)[9]

2.1.6 Deep learning

Deep learning (DL): A machine learning and artificial intelligence (AI) sub-field focusing on the deep neural network with multiple layers so it can learn complex patterns from data.

2.1.6.1 Key Concepts of DL

Artificial Neural network: Smallest unit which falls under Deep learning gives layers and nodes(network of these nodes that are literally interconnected], neuron; the nodes input one are neuron, these process data into place downstream. parent.

Layers: Most deep learning models will come equipped with several layers, that is an input layer followed by zero or more hidden layers and ending in an output layer.

Learning: Deep learning models are trained with the help of big data. The training is on adjusting the weights of neurons connections for lower prediction error.

Backpropagation: Back prop fixes weights through a backward propagation of error throughout neural network training.

Activation Functions: The model becomes capable of detecting advanced patterns through non-linear transformations achieved by functions such as ReLU (Rectified Linear Unit) and Sigmoid. (Shiloah-Perl and Geries 2023)[12] (Georgievich and Terblanche 2019)[13].

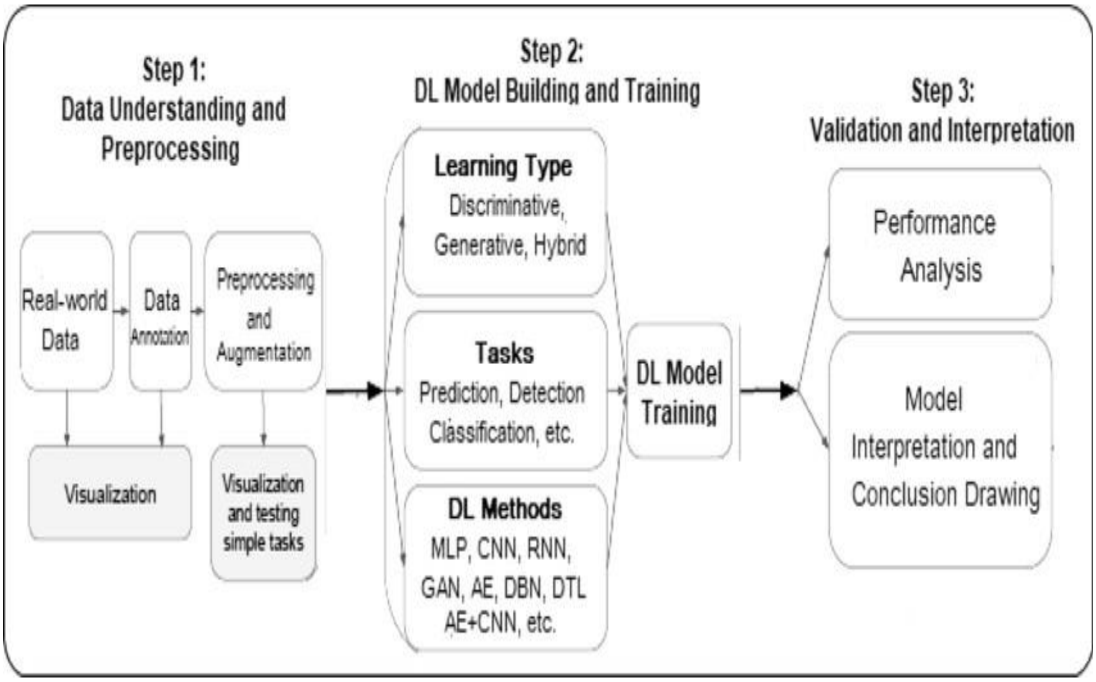

Figure 2.4 Illustrates a deep learning (DL) workflow with three stages: data preprocessing, model training, and result validation.(Sarker 2021)[14]

2.1.7 Convolutional Neural network (CNN)

The fundamental aspect of deep learning algorithms which identify images and videos utilizes Convolutional Neural networks (CNN). Points to remember:

2.1.7.1 Basics of Convolutional Neural Network Architecture:

Multiple layers make up CNN in which convolutional layers, pooling layers and fully connected layers are included. Through convolutional layers images receive filters that generate feature maps to identify patterns including edges and textures and shapes. (O'Shea 2015)[15].

Convolution Operation: The main operation in CNN functionally relies on the A feature map results from applying a filter across the input image through the convolution operation. The data contains spatial hierarchies which can be captured through this operation. (Salman Khan el.at., 2018)[16].

Pooling: The computation becomes more efficient while overfitting risks decrease because pooling layers decrease the number of features in the maps. Max pooling and average pooling represent the standard methods used in pooling operations. (Salman Khan el.at., 2018)[17]

Activation Functions: Models utilize ReLU (Rectified Linear Unit) activation functions for non-linear operations because they produce non-linearities which allow the model to detect complex patterns. (O'Shea 2015)[16]

Training: Backpropagation together with gradient descent serves as the training method for CNN. The network modifies its weight values through error calculations between predicted outputs and actual outputs to improve accuracy progressively. (Salman Khan el.at., 2018)[17]

CNN are widely used in various applications, including image classification, object detection, facial recognition, and medical image analysis.(Sarvamangala and Kulkarni 2022)[17]

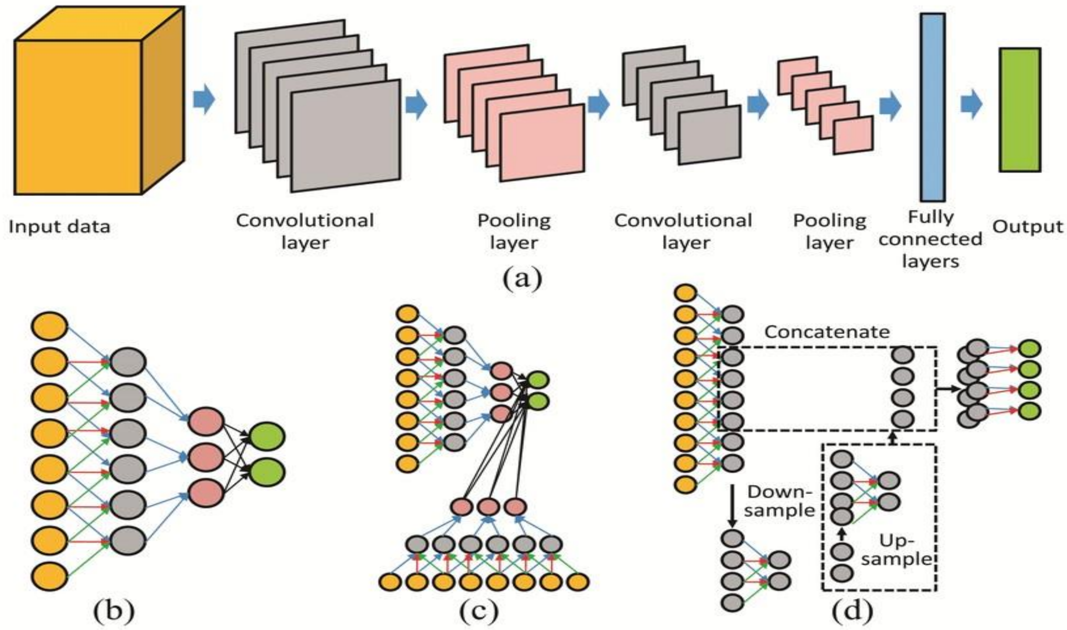

Figure 2.5 Common CNN architectures in medical imaging, including classical, multi-stream, and segmentation models.(Munkem et al. 2019)[18]

2.1.7.2 Overfitting

This occurs when the number of input samples is too small compared to the learning capacity of the network. Overfitting does not allow to learn the general characteristics of the classes and instead captures the noise of the training set (Srivastava et al. 2014)[19]. This leads to a model with high accuracy during training but that is unable to generalize (i.e., it does not achieve high test accuracy)(Bulent et al. 2019)[20] .techniques have been developed to improve the performance of the test set, even if it means reducing the performance of the training set. These techniques are called regularization techniques (Goodfellow et.al., 2016)[21] . One of these is data augmentation.

2.1.7.3 Data augmentation

Data augmentation consists of the transformation of the geometry or intensity of the original images to make them seem like new images. The operations are often simple: rotation, mirroring, random cropping, zooming, or even adding noise, changing contrast, and brightness values as well as simulating new backgrounds. The size of the dataset and its diversity are therefore artificially increased. The augmentation operations can be performed in different ways, using one or several (possibly randomly chosen) transformations per image.

It is important to carry out these augmentation operations once the sets have been defined to ensure that an image and its duplicates are in the same set. Also augmentation operations have proven to be effective against overfitting (Bulent et. al., 2019)[21].

2.1.8 Image Classification

Image classification is an essential computer vision task of classifying images by their visual content in an automated way. This transformation allows computers to (eventually) understand images, which opens up a world of applications.

2.1.8.1 Important Keywords :

Image Data: The digital representation of images contains numerical arrays that show pixel intensity data.

Features: The identification of classes relied on extracted image characteristics including color, texture, shape and edges.

Labels: Images receive assigned categories which represent the shown objects or environments.

Training Data: This collection comprises labeled pictures that serve as training material for image classification systems.

Models: Mathematical functions that learn patterns from training data and predict labels for new images.

2.1.8.2 Applications:

Object Recognition: In images, detecting particular classes of objects (cars/people/pedestrians or animals etc. self driving cars, X-ray vision) .

Scene Classification : The process of classifying images according to scene they are representing, e.g., landscape, indoor or outdoor environment (e.g., robotics,image search).

Image Retrieval: Image search and retrieval by finding image similar to that based on visual content. (e.g. Online photo databases, medical image archives)

Content Moderation: Distinguishing inappropriate or explicit content in images to let online platforms keep safe environments (Social media platforms, online marketplaces etc.)

2.1.8.3 Approaches:

Traditional Methods: Researchers employed SVMs in addition to Random Forests and hand-crafted features in their approach. Heavy feature engineering is needed by methods to process images but they can develop overfitting when operating on sophisticated image variations.

Deep learning Approaches: The application of Convolutional Neural network (CNN) extracts features from image data automatically and achieves top performance results. The CNN architecture shows exceptional flexibility when dealing with image variations of various complexities.

In image classification, CNN outperform traditional image processing methods in several applications. This general trend is also observed in the automatic identification of crop diseases. Some of the selected studies compared the performance obtained with CNN to that of other methods. In all of these studies, the CNN results are better than the others (Wu et al. 2024)[22].

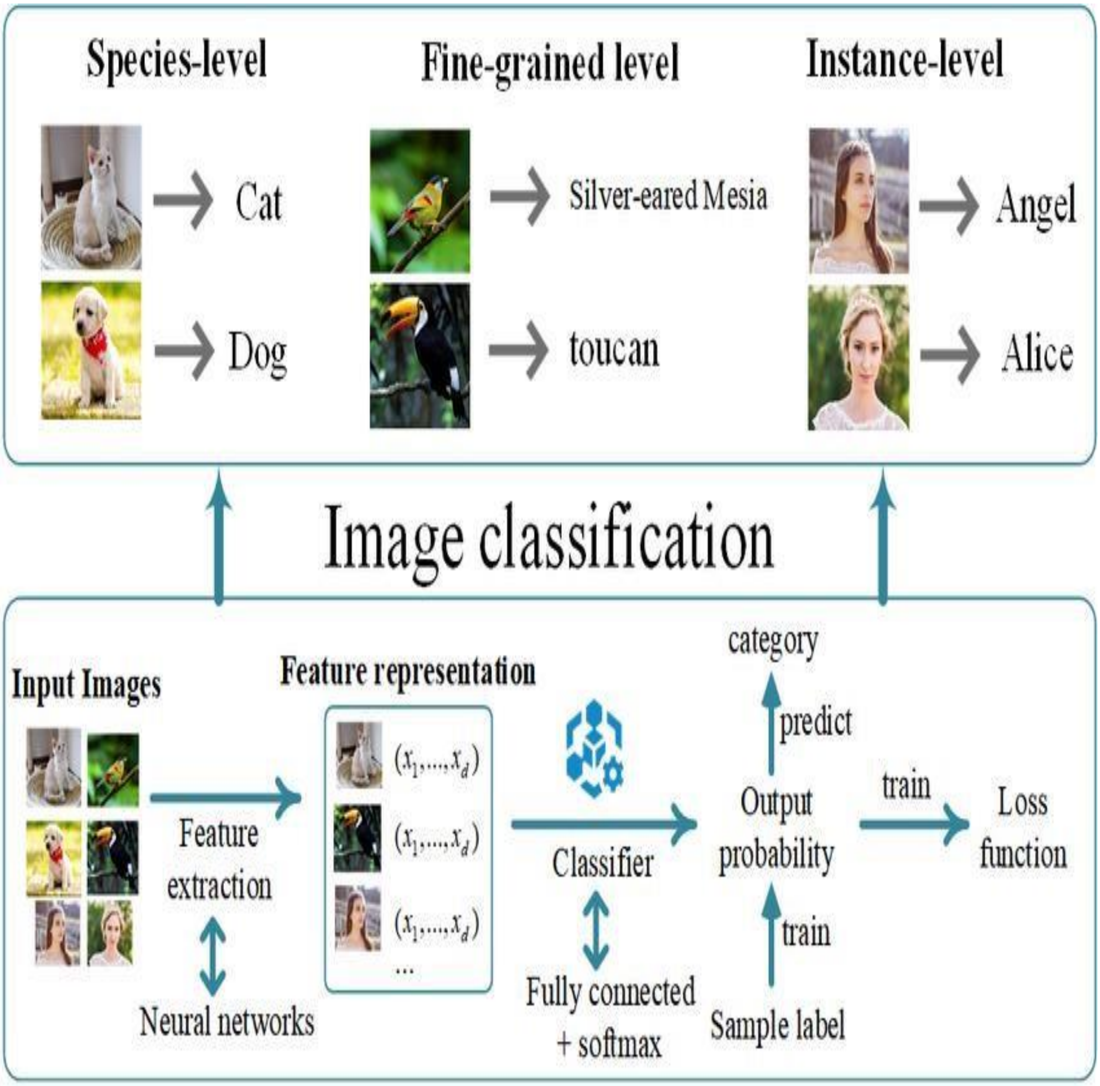

Figure 2.6 Image classification analogy and process schematic (Hoo et al. 2023)[23]

2.2 Related Works

2.2.1 Using Satellite Images and ML:

(Tussupov, et al. 2024) [7], did an analysis of formal represen-tations and verification of pests and pathogens affecting crops using spectral brightness coefficients (SBR) for the period from 2021 to 2023. Their database contains about 10,000 records covering the growing season, types of diseases and pests, as well as their growth phases in a real coordinate system. Their work used machine learning techniques including logistic regression, extreme gradient boosting (XGBoost), and Vanilla convolutional neural network (CNN) to analyze spectral data and classify the presence of pests and diseases in satellite images.

In their study Logistic regression, Vanilla CNN, and XGBoost demonstrate their effectiveness in various aspects of the problem of verifying harmful organisms and pathogens. Logistic regression provides a simple and interpretable method for classifying and assessing the risks associated with pest exposure to crops. On the other hand, Vanilla CNN can efficiently work with image and spectral data, making it a powerful tool for automatically identifying and classifying pathogens and harmful organisms. This method helps identify even complex samples and allows analysis at a higher level of detail.

XGBoost, on the other hand, stands out for its ability to work efficiently with numerical data and time series. This boosting algorithm helps improve the accuracy and reliability of forecasts and can be used to more accurately estimate the proportion of harmful organisms and pathogens .

The combined use of spectral-space data and machine learning techniques in agriculture has enormous potential to improve the sustainability of agricultural systems, optimize resources, and increase productivity. These innovative approaches lead to more precise and efficient farmland management, which is critical in the face of growing populations and changing climate conditions

2.2.2 Using DL and CNN :

Also (A.Hussain et.al., 2018) [24] used a deep-learning framework to classify wheat diseases using images captured in-place by camera devices, with various resolutions. their dataset contains four categories of wheat disease like stem rust, yellow rust, Powderly and normal. Each category contained 2,207 images. To train their classifier they used the Convolutional Neural Network (CNN).

Their results show that deep models and particularly CNN outperform the previous works in wheat diseases classification..but their dataset was under uncontrolled condition and with a focus on a particular organ .their 84.54% accuracy may fall drastically when using an image captured under uncontrolled conditions and without focus on a particular organ like most of the studies reviewed in (Bulent et al. 2019) [17] .

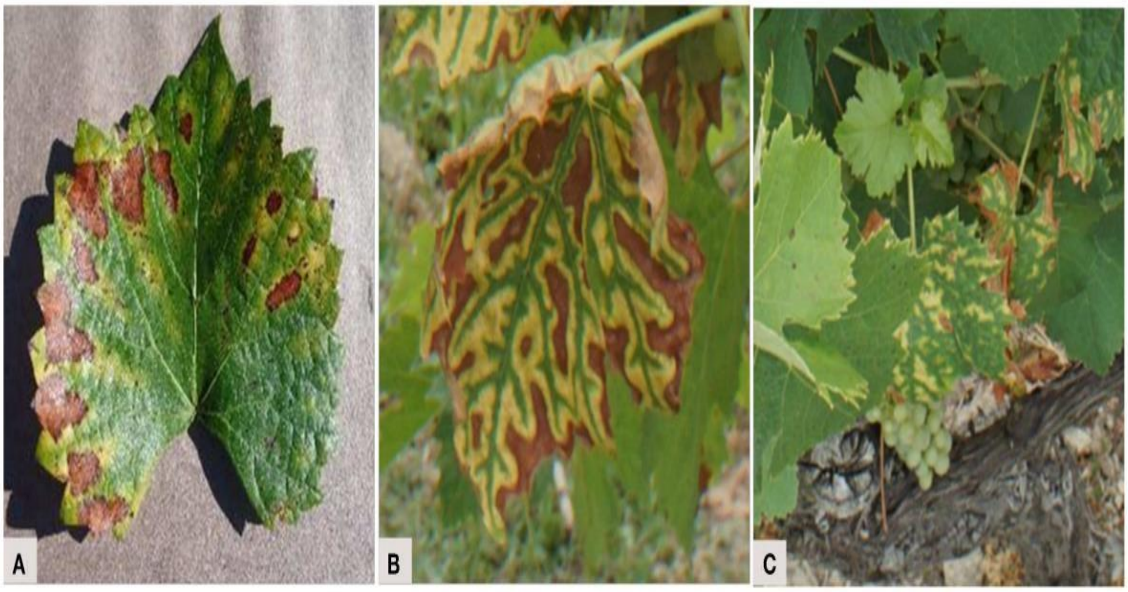

Figure 2.7 Typology of image complexity in datasets, showing Esca grape disease under controlled (A) and uncontrolled conditions (B, C)

2.2.3 Using Compueter vision and IoT:

(Ouhami et al. 2021)[25] explores the use of Computer Vision (CV), Internet of Things (IoT), and Machine Learning (ML) for detecting crop diseases. The paper looks at different CV methods in use such as image processing pulling out features, and sorting algorithms like CNN. It also checks out how IoT gadgets like sensors and drones help gather crop data, including pictures and environmental factors. They looked into how mixing various data types (images, sensor info) leads to better disease spotting (Data fusion).

The paper points out some tricky parts, like not having enough data, differences in how diseases look, and the need to process info . They talked about promising fields such as deep learning blending different types of data, and apps for phones.

The study conducted a thorough review of literature to collect data on current methods. It examines the advantages and disadvantages of various approaches evaluating their effectiveness and usefulness.

The researchers determined that blending computer vision Internet of Things, and machine learning has huge promise to detect crop diseases and . They also pointed out the need to study more in fields such as data enhancement instant processing, and reliable model implementation

2.2.4 Review of CNN-Based Approaches for Crop Disease Identification:

Since 2016, advances in image processing have come from deep learning methods Convolutional Neural network (CNN). This has led to the creation of many tools that identify diseases in crops.To assess the potential of these network for such applications,(J. Boulent et.al., 2019) [21] surveyed 19 studies that relied on CNN to automatically identify crop diseases. they described their profiles, their main implementation aspects and their performance.

They identified some of the major issues and shortcomings of works that used CNN to automatically identify crop diseases. they also provided guidelines and procedures to follow in order to maximize the potential of CNN deployed in real-world applications. Many already-published solutions based on CNN are not currently operational for field use mostly due to a lack of conformity to several important concepts of machine learning. This lack of conformity may lead to poor generalization capabilities for unfamiliar data samples and/or imaging conditions, which lowers the practical use of the trained models. Nevertheless, they said that the studied works show the potential of deep learning techniques for crop diseases identification. And Their findings are definitely promising for the development of new agricultural tools that could contribute to a more sustainable and secure food production.

CHAPTER THREE

METHODOLOGY

3.1 Introduction

This work is to propose and realize a novel crop health monitoring system by using Convolutional Neural network (CNN) features. As the demand for food production keeps growing and as the effects such as climate change, plant diseases and so on are presented in front us, effective monitoring tools are certainly urgent for the precision agriculture. The main purpose of this work is to design and implement a system of plant disease accurate classification and diagnosis from image of the crop leaf. By facilitating the timely identification of crop diseases, this system aims to enhance crop yield and ensure food security. The presented solution overcomes a number of limitations in the conventional approach in agriculture, such as, the manual interrogation and the identification detection latencies.

Using recent advanced deep learning methods, the system can quickly analyze images and generate valuable actions for farmers and/or the agricultural community. For this aim, a Stepwise procedure is followed including all the stages from data collection, data preprocessing, design of CNN model, model training and model evaluation. All stages contribute in an indispensable way to the accuracy and robustness of the final product. The following methodology section gives a detailed description of those steps, outlining the workflow and selection criteria at each of those steps. This systematic methodology not only facilitates a robust monitoring system, but also enables the progress of research in the area of agricultural technology.

3.2 System Model:

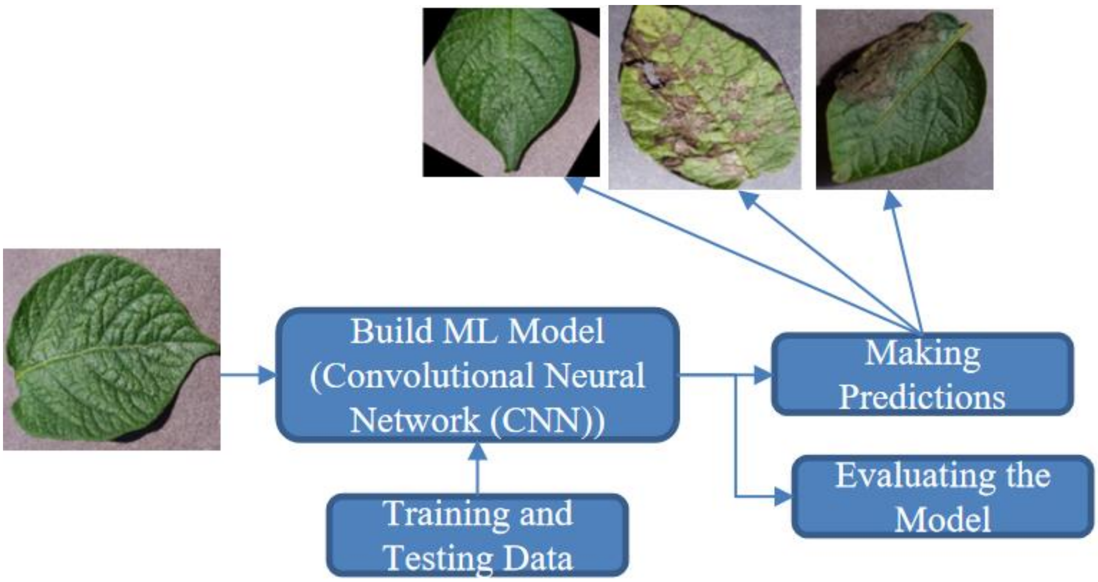

Figure 3.1 Plant disease detecting architecture

The design of the system is organized around the following major stages which when combined generate a coherent work flow for training and testing the crop disease detection models. The pipeline consists of stages such as dataset acquisition, data preprocessing, model architecture design, model training and testing.

3.2.1 Datasets Acquisition

The dataset on which this work is based is the "New Plant Diseases Dataset", which includes 3999 RGB images of potatoes leaves . These images are classified into 3 different classes, which refer to three health conditions of this crop:

Figure 3.2 Potato leaf in good health

Figure 3.3 Potato Leaf images displaying early symptoms of blight, a fungal disease.

Figure 3.4 Potato Leaf images showing advanced stages of the blight infection.

This dataset provided to us, released on Kaggle, is the key to the machine learning model training for the identification and classification of plant diseases. In this research, through choosing an expert curated and balanced dataset the model can be made to generalize well for other health conditions. Data is divided into train, validation and test sets to ensure a valid model development and testing process The below gives an idea of the distribution in terms of dataset.

Table 3.1 Overview of the dataset distribution

Crop Type	Class	Training Images	Validation Images	Test Images
potato	**healthy**	**1000**	**200**	**133**
potato	**early blight**	**1000**	**200**	**133**
potato	**late blight**	**1000**	**200**	**133**

This process standardizes every image to 255x255 pixels so that CNN model can handle similarly sized input. Balanced Classes: to avoid the model from leaning towards any class and having the correct classification fairly balanced.

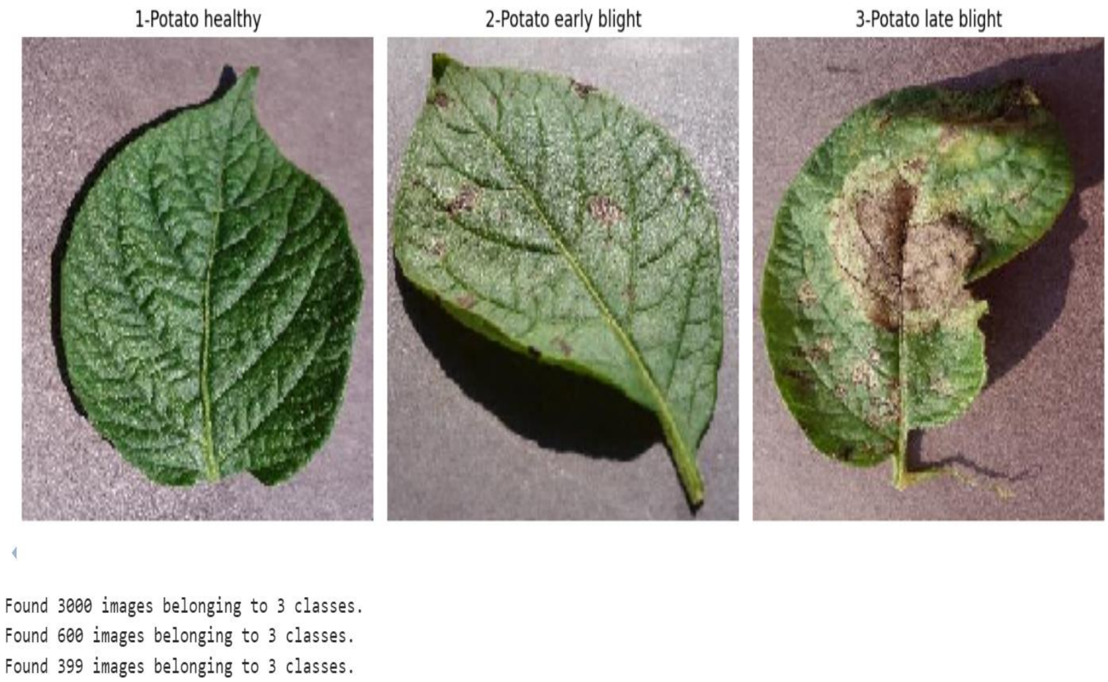

Figure 3.5 Dataset distribution

3.2.2 Data Preprocessing

The data is preprocessed before feeding into the model to make data model trainable. Good preprocessing can help drastically model performance, avoid both overfitting and underfitting.

Key preprocessing steps include: Key preprocessing steps include:

1.**Image Resizing**: All images are resized to 255x255 pixels. This ensures uniform input sizes and reduces computational complexity.

2.**Data Augmentation:** To augment the generalization performance of the model, we perform multiple augmentations on training images during training. This includes transformations like:

-Rotation: Randomly rotating images by up to 20 degrees.

-Shifting: Generating random horizontal and vertical shifts within a 20% range.

- Zooming: Randomly zooming into the images by 20%
- Flipping: Horizontally flipping the images to introduce more diversity.

Augmentation increases the size of the actual training set, which reduces overfitting and increases model robustness.

3.**Normalization**: Pixel values are normalized to the range of [0,1] by pixel values/255. This type of normalization, is characterized by quicker convergence for the model as well because it ensures the training is quicker in this way.

This is the normalization where Training gets faster in this way, allowing the model train faster this kind of normalization.

```python
self.train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True
)
self.validation_datagen = ImageDataGenerator(rescale=1./255)
self.test_datagen = ImageDataGenerator(rescale=1./255)
```

Figure 3.6 Data augmentation and preprocessing

3.3 CNN Model Design

The heart of the crop health monitoring system is the Convolutional Neural Network (CNN). The CNN have the advantage for image classification tasks because of the power to automatically learn and extract features from input images. In this study, the architecture of the convolutional neural network (CNN) model is appropriately formulated for its optimal performance. The CNN model consists of the following components:

1.Convolutional Layers: In these layers filters are applied to extract spatial information from the images. They identify features like edges, texturing and shapes that discriminate between healthy and diseased crops. - Activation Function: The ReLU (Rectified Linear Unit) activation function endows the model with non-linearity, which enables the model to learn high-level patterns. ReLU is widely applied due to its efficiency in computation and stability in preventing such problems as the vanishing gradient problem.

2.Pooling Layers: Down sample the feature maps by MaxPooling, which not only downsample the spatial dimension of the feature maps but also preserves useful characteristics. Pooling not only minimizes the computational complexity of the model, but also the overfitting problem.

3. Dropout Layers: Dropout is used to improve model generalization. While training, dropout randomly turns off half of the neurons in the network, thus requiring the model to learn robust features which are not dependent on single neurons.

4. Fully Connected Layers: At the network end fully connected layers are implemented to integrate the features derived by the convolutional layers and to predict something. The output layer employs softmax activation, for classifying the input into one of the three classes.

```
self.model = models.Sequential([
    layers.Conv2D(32, (3, 3), activation='relu',
     input_shape=(self.img_height, self.img_width, 3)),
    layers.MaxPooling2D((2, 2)),
    layers.BatchNormalization(),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.BatchNormalization(),
    layers.Conv2D(128, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.BatchNormalization(),
    layers.Flatten(),
    layers.Dense(256, activation='relu'),
    layers.Dropout(0.5),
    layers.Dense(len(self.classes), activation='softmax')
```

Figure 3.7 CNN layers

3.4 Model Compilation and Training

The CNN models are compiled and trained using standard techniques to ensure optimal performance. Adam optimizer is used, as this has adaptive learning rate that allows us to minimize the loss function faster.

The models are built on categorical cross-entropy as this is used for multi-class classification and loss function.

```
self.model.compile(optimizer=Adam(learning_rate=0.0001),
                   loss='categorical_crossentropy', metrics=['accuracy'])
```

Figure 3.8 Adam Optimizer

Batch Size: We train the model using batch size as 32, which means after every 32 images that time model get updated in a training.

- Epochs: The models are trained for up to 15 epochs.

```
def train_model(self, max_epochs=15):
    history = self.model.fit(
        self.train_generator,
        steps_per_epoch=self.train_generator.samples // self.batch_size,
        validation_data=self.validation_generator,
        validation_steps=self.validation_generator.samples // self.batch_size,
        epochs=max_epochs
    )
```

Figure 3.9 Model Training Code Snippet

```
Epoch 1/15
93/93 ━━━━━━━━━━━━━━━━━━━━ 148s 2s/step - accuracy: 0.8083 - loss: 1.1911 - val_accuracy: 0.3403 - val_loss: 5.2508
Epoch 2/15
93/93 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.9062 - loss: 0.3585 - val_accuracy: 0.1667 - val_loss: 6.3800
Epoch 3/15
93/93 ━━━━━━━━━━━━━━━━━━━━ 151s 2s/step - accuracy: 0.9209 - loss: 0.3568 - val_accuracy: 0.3264 - val_loss: 11.6093
Epoch 4/15
93/93 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.8438 - loss: 0.7713 - val_accuracy: 0.5000 - val_loss: 8.6552
Epoch 5/15
93/93 ━━━━━━━━━━━━━━━━━━━━ 143s 2s/step - accuracy: 0.9293 - loss: 0.2975 - val_accuracy: 0.3351 - val_loss: 9.0640
Epoch 6/15
93/93 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.8750 - loss: 0.2468 - val_accuracy: 0.2917 - val_loss: 9.7377
Epoch 7/15
93/93 ━━━━━━━━━━━━━━━━━━━━ 143s 2s/step - accuracy: 0.9401 - loss: 0.2041 - val_accuracy: 0.3385 - val_loss: 8.1410
Epoch 8/15
93/93 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 1.0000 - loss: 0.0211 - val_accuracy: 0.2083 - val_loss: 9.7979
Epoch 9/15
93/93 ━━━━━━━━━━━━━━━━━━━━ 144s 2s/step - accuracy: 0.9400 - loss: 0.2125 - val_accuracy: 0.5417 - val_loss: 1.9633
Epoch 10/15
93/93 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.9688 - loss: 0.1091 - val_accuracy: 0.4167 - val_loss: 2.7225
Epoch 11/15
93/93 ━━━━━━━━━━━━━━━━━━━━ 141s 1s/step - accuracy: 0.9506 - loss: 0.1406 - val_accuracy: 0.8299 - val_loss: 0.6657
...
Epoch 14/15
93/93 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.9688 - loss: 0.0832 - val_accuracy: 0.8750 - val_loss: 0.1784
Epoch 15/15
93/93 ━━━━━━━━━━━━━━━━━━━━ 145s 2s/step - accuracy: 0.9463 - loss: 0.1456 - val_accuracy: 0.9149 - val_loss: 0.4844
```

Figure 3.10 Accuracy and Loss per Epoch

3.5 Model Evaluation

The performance of the learnt model is then assessed on the test set. In this stage, a number of parameters are calculated providing insights into the generalization ability and the correct classification of the input images of the model. The following metrics are used:

Accuracy: The percentage of correctly predicted outcomes out of the total predicted outcomes by the model.

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP} \tag{3.1}$$

Precision: is the ratio of correctly predicted positive observations to the total predicted positives. It measures the accuracy of the positive predictions.

$$precision = \frac{TP}{TP + FP} \tag{3.2}$$

Recall: Also known as sensitivity, recall is the ratio of correctly predicted positive observations to all actual positives. It gives an idea of how well the model identifies true positive cases.

$$recall = \frac{TP}{TP + FN} \tag{3.3}$$

F1-Score: This is the harmonic mean of precision and recall. It can provide a single quantitative measure of the trade-off between precision and recall. These metrics offer a global view of the model's performance on disease classes

$$F1 = \frac{2 \times precision \times recall}{precision + recall} \quad (3.4)$$

Besides accuracy, precision and recall, the F1 score can also be calculated to characterize the classification performance for each class, offering more specific control over the model's distinguishability between healthy and diseased leaves.

3.6 Model Testing

In order to evaluate the use of the trained model for the real-world application, a testing phase was performed using a data set consisting of previously unseen images. These test images were specifically chosen to allow the model's predictions to be tested by completely new, and thus unlabeled, data. The model's performance was evaluated by predicting the labels for these test images and comparing them with the ground truth labels.

The model was tested using the test -generator, which loads the test data and applies the same preprocessing as the training and validation sets.The model metrics used include accuracy, precision, recall and F1 score. These measures cover well the performance of the model over real life scenarios. Further, the model predictions were evaluated against the actual labels.

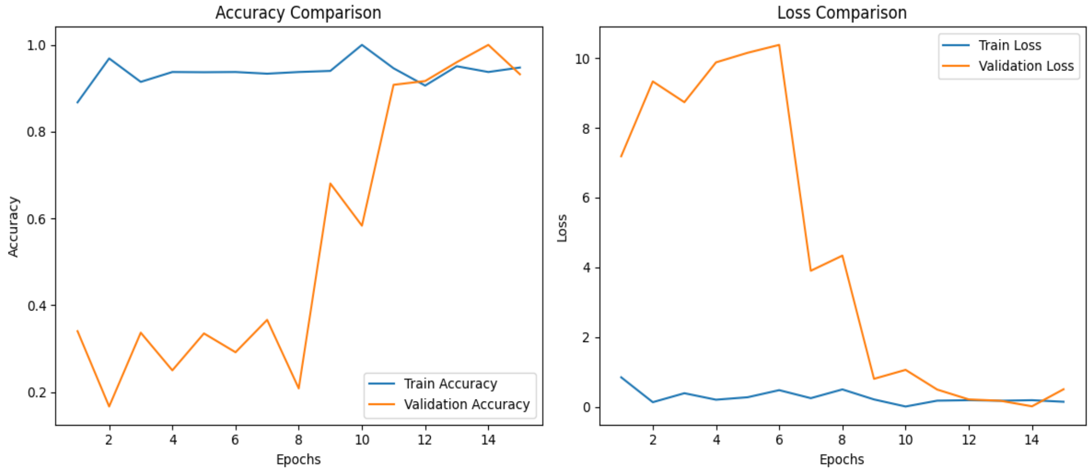

Figure 3.11 Training and validation accuracy

Figure 3.12 Tested Samples of Potato Leaf Health Conditions

Interpretation of Model Predictions healthy, early blight, and late blight. This transformation was critical to translate the model's general numerical results into useful ones. Through the indices of the predicted class labels, and mapping them to those real-world conditions, the model outputs were made more interpretable and practical. This mapping facilitated more accurate decision making, because the predictions could be directly related to real-world agricultural problems, e.g., crop health management. □

3.7 User Interface

This is the interface of the crop health monitoring system (Figure 3.12), where a user can upload an image of a crop leaf for disease detection. The system processes the uploaded image for diagnosis, showing the probability of different diseases. Here, the result shows 100% Late Blight, meaning the system has identified the uploaded leaf to be affected by Late Blight with high confidence.

This interface connects to the backend, where it uses a trained Convolutional Neural Network (CNN) model that classifies crop diseases based on learned features. The clean and simple UI ensures ease of use for farmers or researchers who will analyze crop health.

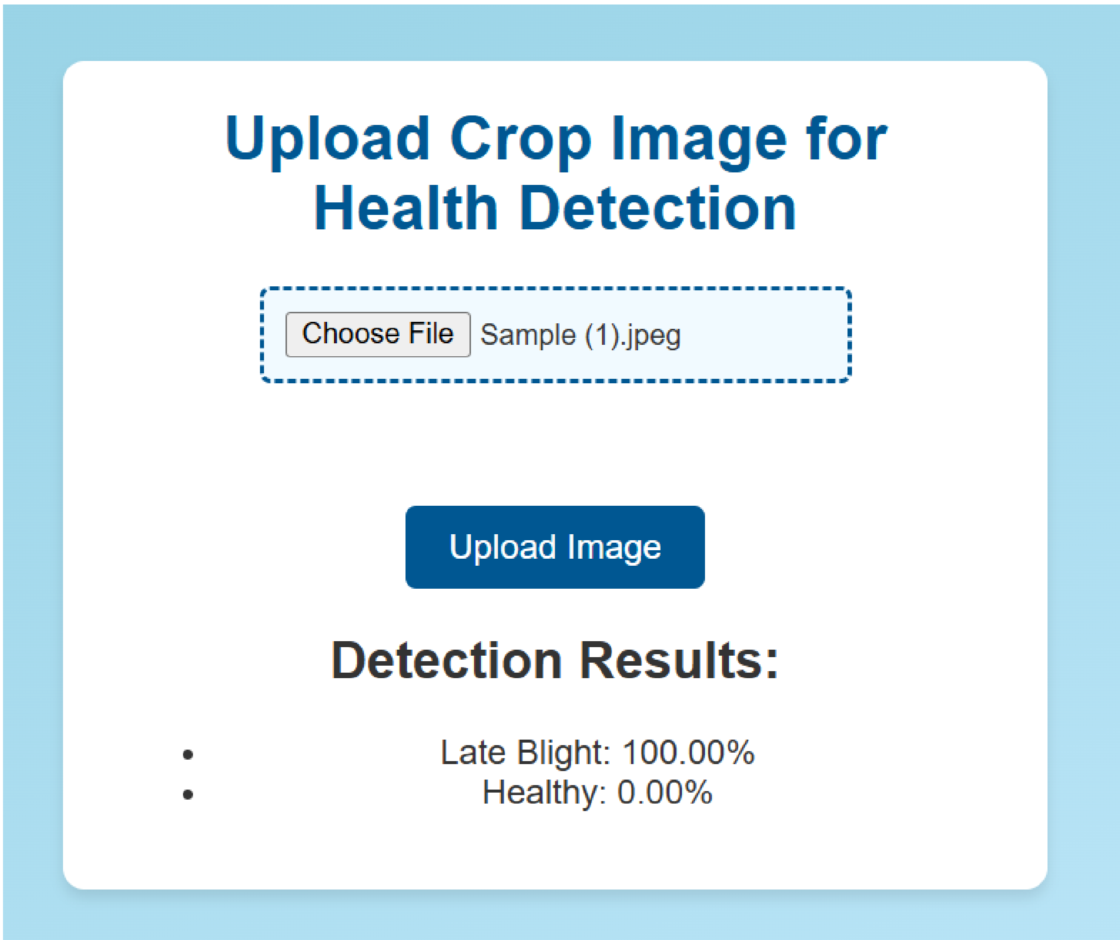

Figure 3.13 User Interface

```python
@app.route('/upload', methods=['POST'])
def upload_image():
    # Check if a file is uploaded
    if 'file' not in request.files:
        return jsonify({"error": "No file part"}), 400

    file = request.files['file']

    # Validate the uploaded file
    if file.filename == '' or not allowed_file(file.filename):
        return jsonify({"error": "Invalid file format. Please upload a valid image (PNG, JPG, JPEG, GIF)."}), 400

    # Save the uploaded file
    if file:
        filepath = os.path.join(app.config['UPLOAD_FOLDER'], file.filename)
        file.save(filepath)

        # Load and preprocess the image
        img = image.load_img(filepath, target_size=(255, 255))  # Resize to match model input
        img_array = image.img_to_array(img)  # Convert to array

        # Use the predict function
        top_two_predictions = predict_top_two(model, img_array)

        # Return prediction results with class names and confidence percentages
        return jsonify({
            "top_two_predictions": [
                {"class": top_two_predictions[0][0], "confidence": f"{top_two_predictions[0][1]:.2f}%"},
                {"class": top_two_predictions[1][0], "confidence": f"{top_two_predictions[1][1]:.2f}%"}
            ]
        })
```

Figure 3.14 Upload and classify image

3.8 Summary

This methodology describes a systematised procedure to build a crop health monitoring system based on CNN. Using Convolutional Neural network (CNN) the system is able to detect and classify plant diseases from leaves images. Under the principle of rigorous data preprocessing, model construction, training and testing, the model shows excellent performance in the field of disease identification for similar crops in different plants.

This solution provides a promising potential tool for today's agricultural system as well, with the potential for potentially more proactive and targeted disease management practices, in turn reducing crop failure and increasing yields. This more elaborated methodology now extends the thinking to include such issues as data augmentation, design choices of the model, training procedures, evaluation metrics, and deployment plans. It explains in detail to serve as a basis for further in-depth research documentation.

CHAPTER FOUR

RESULTS AND DECISIONS

4.1 Introduction

In this chapter, we present the results of the crop disease detection model using image data of crop leaves. The main focus of this model was to see the effectiveness of the model for the identification of diseases, particularly focusing on its ability to detect diseases in different classes such as early blight and late blight. The results are tabulated and discussed with regards to major key performance indicators of accuracy, recall, precision, and F1-score. For a comprehensive assessment of the strength of the model, early and late stages for both early blight and late blight detection were studied.

It was also compared with the ground-truth data to ensure the classification accuracy over the range of disease stages. Furthermore, statistical tools such as confusion matrices and classification reports were put into use for the purpose of showing the model strength in discerning each class. The result of the study gives insight into the strengths and limitations that the proposed system may consider for actual application in the field.

4.2 Training Results

The CNN model was trained using a 75%-15%-10% split for training, validation, and testing, respectively. The training dataset contain 1000 image per class. Throughout the training process, we monitored the model's accuracy and loss using the Adam optimizer, which was selected for its adaptive learning rate and efficient handling of sparse gradients.
The training results showed consistent improvement across epochs, with the model achieving a final training accuracy of 95% after 15 epochs.

Figure 4.1 illustrates the model's training and validation accuracy and loss over the course of training.

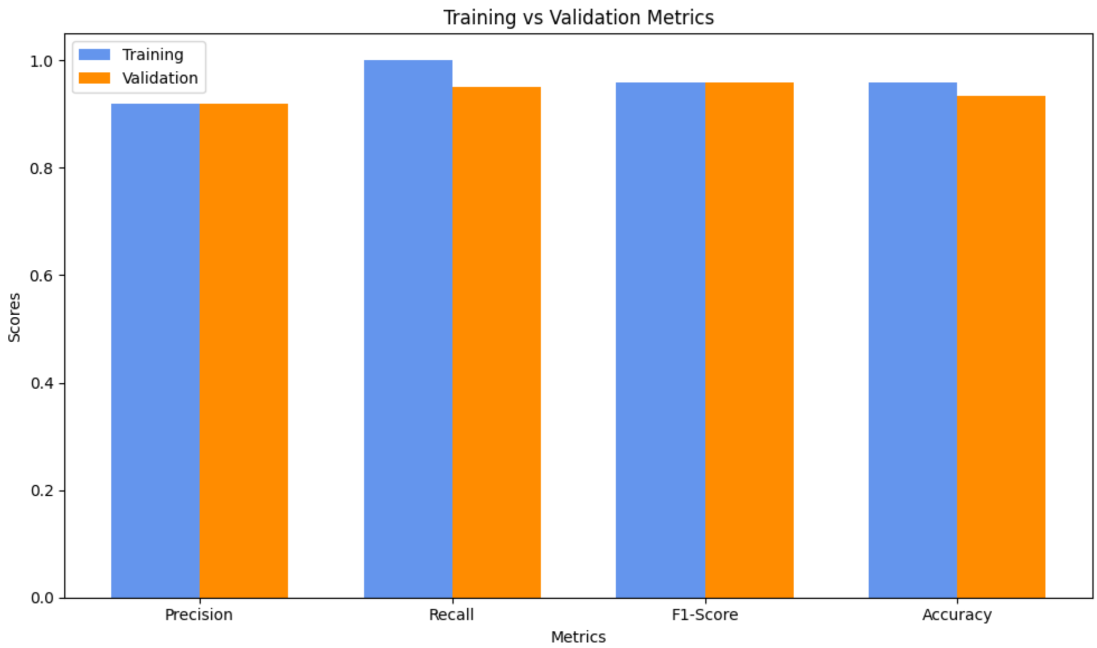

Figure 4.15 Training vs Validation Metric

4.3 Validation and Testing Results

Following the training phase, the model is tested on a validation dataset of 200 photos per class. This step prevents difficulties by fine-tuning the model and reducing overfitting, which is a typical concern in machine learning models. As shown in (Figure 4.2), the CNN model maintained high accuracy on the validation set, hitting 94%, demonstrating significant generalization.

Testing is conducted on a dataset consisting of 133 images per class, and the model demonstrated an accuracy of 97%. This result confirms the reliability of the model's predictive capability when applied to unseen data.

Figure 4.16 Validation and Testing

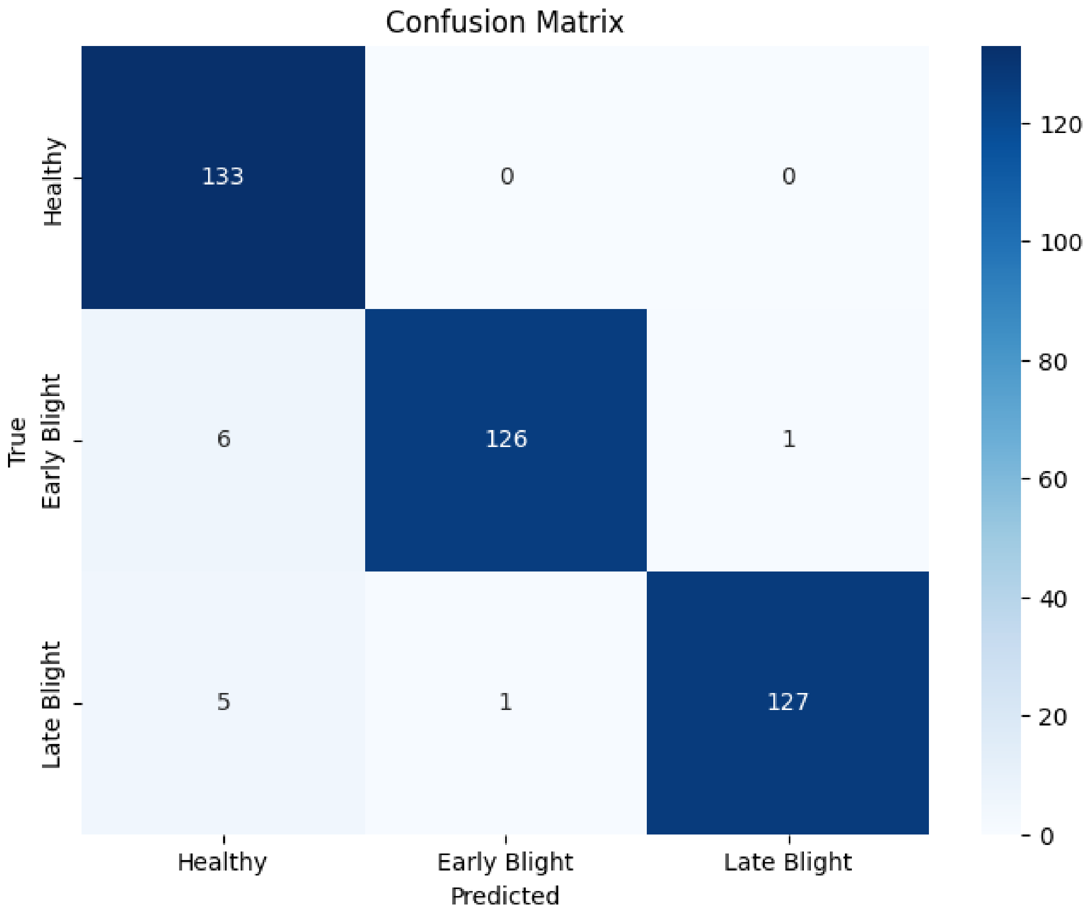

Figure 4.17 Confusion Matrix of Potato Disease Detection Model

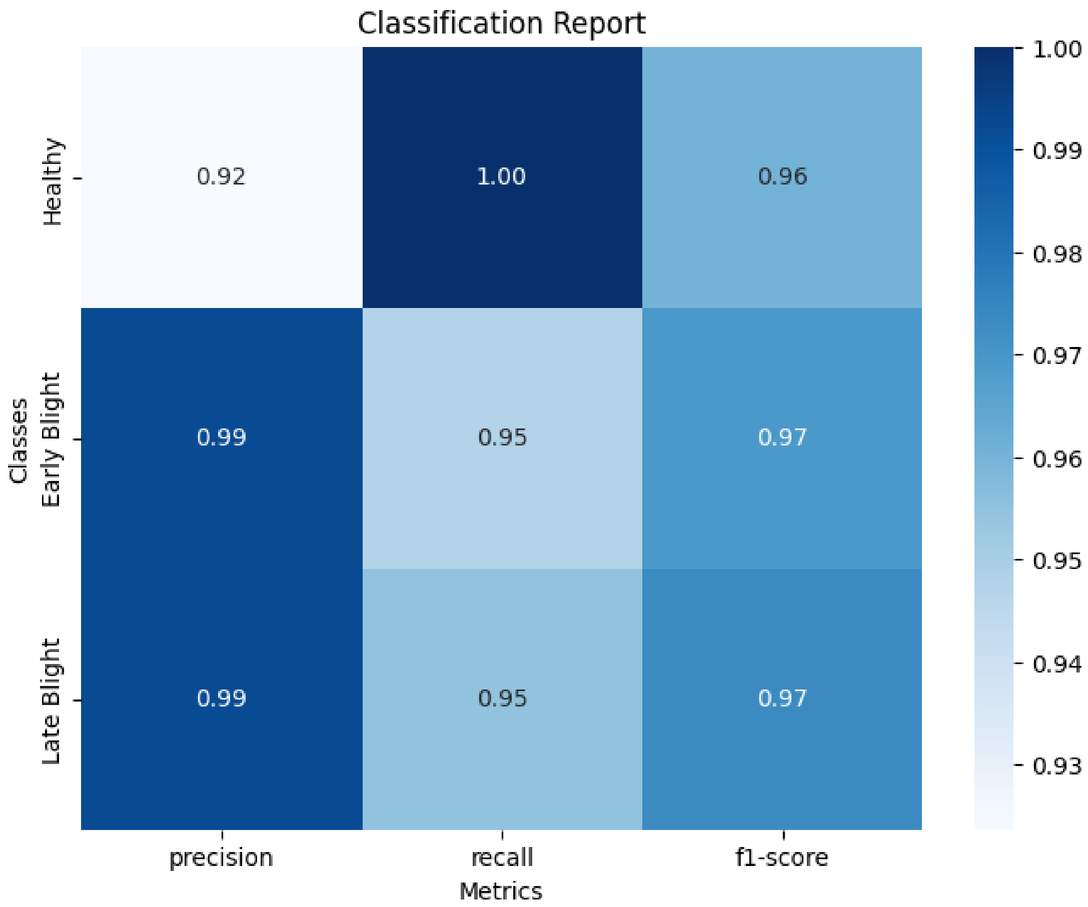

Figure 4.18 Classification Report of Potato Disease Detection

4.4 Confusion Matrix

The confusion matrix (Figure 4.3) shows the true labels (ground truth) versus the predicted labels for each class. The matrix indicates the following results:

-Potato Healthy: All of the 133 images were correctly classified as healthy, resulting in a 100% accuracy for this class.

- Potato Early Blight : Out of 133 images of early blight, the model correctly classified 126 images, with 1 image misclassified as late blight and 6 images misclassified as Healthy. This results in a 94% accuracy for this class.

-Potato Late Blight : The model correctly identified 127 out of 133 late blight images, while 5 images identified as Healthy and one image as Early bligh. This yields of 94% for late blight, indicating that this stage of the disease was more challenging for the model to distinguish, particularly from early blight.

The confusion matrix highlights that the model performed exceptionally well in distinguishing healthy potato plants, while there was some confusion between early and late blight stages with healthy plants.

4.5 Classification Report

The classification report (Figure 4.4) provides a summary of the model's performance using precision, recall, and F1-score for each class. The results for each class as the following: Potato Healthy (Precision:0.92 ,
Recall: 1 , F1-Score:0.96) , for Potato Early Blight: (Precision:0.99,Recall: 0.95 , F1-Score:0.97) ,and Potato Late Blight: (Precision:0.99 ,
Recall: 0.95 , F1-Score:0.97)

To calculate the overall accuracy we take the sum of the True Positives /
Total samples: True Positives = Recall * support
For each class:
Potato Healthy: 0.92*133=122.

Potato Early Blight: 0.99 * 133=131.67

Potato Late Blight: 0.99 * 133=131.6

Sum of true positives:385.70

The accuracy:385.70/399= 0.97

This shows that the model performed well across all classes, achieving an overall accuracy of 97%. The model's highest performance was in identifying the Healthy class with 1.00 but it showed slightly lower performance in identifying Late Blight and Early Blight with a recall of 0.95.

4.6 Discussion of Findings

Thus, The experimental results demonstrate that the developed model performs efficiently at identifying normal potato plants and initial blight stages. The system features obstacles in its ability to distinguish early blight from late blight manifestations. The confusion matrix indicates that model misclassified instances between early and late blight stages because the visual distinctions between these stages were confusing for classification.

The applied model reached 97% accuracy in identification yet five late blight pictures received incorrect identification as healthy results and one early blight image was misdiagnosed as late blight. Implementing additional data enhancement strategies alongside feature extraction improvements may allow the model to achieve better results in detecting late blight. The model demonstrates robustness for identifying early blight because it reaches a 0.95 recall rate during its first detection stage thus enabling early disease management interventions.

The early discovery of diseases proves vital for farmers because it enables them to implement preventive steps before the disease expands. Additional enhancement techniques such as data augmentation and advanced feature extraction techniques with elements like contrast modifications and image transformations should be utilized to boost the model's late and early blight discrimination. The minor performance drawback does not outweigh the model's effective early disease detection capability.

4.7 Summary

In summary, the CNN model developed for potato disease detection achieved an overall accuracy of 97%, with strong performance in detecting healthy plants and early blight. The model can be improved by enhance the ability to differentiate between early blight and late blight classes, though the model’s high recall and F1-scores demonstrate its overall effectiveness. Adding additional data could improve the prediction between different classes, also using a stronger architecture will improve the accuracy.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

This research therefore focused on the CNN model for crop disease detection, especially blight at early and late stage. Combined, the efficiency of this model was at 97% proving its reliability with a 95% recall of Early blight and Late blight. Early disease detection is another reason; This high recall demonstrates how effectively the model efficiency was in detecting infections in their early stages which is critical in relation to treatment and illness control. Further, early to late misclassification rate was 9 percent according to the model. Suggesting that there are the two quite similar looking phases of the advancement of the sickness. This misclassification aggravates the need for enhancing Model through more Using training data diversification or through utilizing advanced image processing that might enhance features differentiation.

Despite this challenge, our proposed model has a macro-average F1-score of 0.96, suggesting a good balance of recall and precision that make Useful more applicable in farming contexts. Additionally it is true Healthy plant classification proves that the model can efficiently Reducing the number of false positives for better controlling plants. This work provides a clear understanding of employability of CNN technology in detection of crop diseases; further studies may be made to improve the accuracy of classification and to expand functionality of the CNN in terms of identifying more diseases. When such advanced technologies are integrated into agricultural practices, the farmers enhance the skill of decision making, processes and subsequently foster sustainable control of crop management.

5.2 Recommendations

Further, to improve the performance and usefulness of the crop As to the Medical ailment Detection fashions, there are a several pointers can be suggested also ,Enhancing plants blight differentiation, Assessing records augmentation Techniques such as illumination comparison which are helpful in identifying such Pictures greater easily, rotation, Another model called Zoom technic can help the model higher Distinguish between early and late blight ,in addition to broadening the amount Of layers in the (CNN) to be able to identify additional intricate characteristics as regards improving Its capability to distinguish between related features.

in disorder degrees. In the same way, to promote high generalization, the database has to be stepped up in a way to feature More diverse photographs, concerning the wide range of environmental situations and Plant increase tiers. Training the version on other illness could also Expand its versatility, that enable it to allow for the identification of a much broader URL Session. many type of Agricultural ailment.

Also overall performance progressed with a larger total of the final product getting to the market, mainly the cuisines using real time tracking.

And finally incorporating the above model into internet of Objects (Internet of Things – IOT) like Drones and Sensors would enable Statements in continuity and the Real Time notification, all of which make for quicker interventions when disease symptoms are detected.

The Field trying out is important to corroborate the feasibility of the version’s performance beneath Real-global factors, given that the element included, varies starting with the lights, climate, And different items in the environment. Finally, to as to motivate a user, It is much more important to up the level of knowledge applications that assists farmers on how to successfully Use the device, ensure that the version is available and most importantly consumer friendly for the day by day Farming practices.

5.3 Future works:

In future studies there exist great prospects for carrying out research using more Kinder types of psychometric assessment put ahead intricate psychometric systems for example geometric models. This Are helpful in enhancing disease categorization thereby increasing the likely hood of an accurate disease classification. Expanding the Model to consider the presence of several diseases Not only others Potato diseases as well as the diseases controlling other crops will even lead Practical value to farmers. In addition, integrating this advanced Model with an automated production system that gives Real – Time information.

The implication is that time insights and first automated solutions can increase accuracy, Manufacturing, Implementation of solutions and main product responsibility Greatly.

REFRENCES & APPENDIX

REFRENCES

[1] Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016). Using deep learning for image-based plant disease detection. Frontiers in Plant Science, 7, 1419. DOI:10.3389/fpls.2016.01419

[2] Ferentinos, K. P. (2018). Deep learning models for plant disease detection and diagnosis. Computers and Electronics in Agriculture, 145, 311–318. DOI:10.1016/j.compag.2018.01.009

[3] Sladojevic, S., Arsenovic, M., Anderla, A., Culibrk, D., & Stefanovic, D. (2016). Deep neural network based recognition of plant diseases by leaf image classification. Computational Intelligence and Neuroscience, 2016, 3289801. DOI:10.1155/2016/3289801

[4] Too, E. C., Yujian, L., Njuki, S., & Yingchun, L. (2019). A comparative study of fine-tuning deep learning models for plant disease identification. Computers and Electronics in Agriculture, 161, 272–279. DOI:10.1016/j.compag.2018.03.032

[5] Ahmad, L., et al. (2024). Precision Agriculture: Fundamentals and Applications of Crop and Climate Science. Cham: Springer Nature Switzerland, pp. 151–161.

[6] Abdullahi, H. S., et al. (2017). Convolution neural network in precision agriculture for plant image recognition and classification. Proceedings of the 2017 Seventh International Conference on Innovative Computing Technology (INTECH).

[8] Muhammad, S. (2020). Introductory Chapter: On Digital Image Processing. In: Digital Imaging. Rijeka: IntechOpen, Ch. 1.

[9] Fonseka, T. M., et al. (2019). The utility of artificial intelligence in suicide risk prediction and the management of suicidal behaviors. Australian & New Zealand Journal of Psychiatry, 53, pp. 954–964.

[10] Kubat, M. (2021). An Introduction to Machine Learning. Springer International Publishing.

[11] Putatunda, S. (2019). Machine Learning: An Introduction. In: A. K. Laha, ed., Advances in Analytics and Applications. Singapore: Springer Singapore, pp. 3–11.

[12] Nilsson, N. J. (1998 (draft version)). Introduction to Machine Learning, Stanford University.

[13] Shiloh-Perl, L. and Giryes, R. (2023). Introduction to Deep Learning. In: Machine Learning for Data Science Handbook: Data Mining and Knowledge Discovery Handbook. Springer, pp. 301–338.

[14] Georgevici, A. I. and Terblanche, M. (2019). Neural network and deep learning: a brief introduction. Intensive Care Medicine, 45(5), pp. 712–714.

[15] Sarker, I. H. (2021). Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. SN Computer Science, 2(6), p. 420.

[16]O'Shea, K. (2015). An introduction to convolutional neural network. arXiv preprint, arXiv:1511.08458.

[17] Salman Khan, H. R., Syed Afaq Ali Shah, and Mohammed Bennamoun (2018). A Guide to Convolutional Neural network for Computer Vision. Cham: Springer Nature Switzerland AG.

[18] Sarvamangala, D. R. and R. V. Kulkarni (2022). "Convolutional neural network in medical image understanding: a survey." Evolutionary Intelligence **15**(1): 1-22.

[19]Monkam, P., et al. (2019). Detection and Classification of Pulmonary Nodules Using Convolutional Neural network: A Survey. IEEE Access, pp. 1–1.

[20]Srivastava, N., et al. (2014). Dropout: A Simple Way to Prevent Neural network from Overfitting. Journal of Machine Learning Research, 15, pp. 1929–1958.

[21] J .Boulent., et al. (2019). Convolutional Neural network for the Automatic Identification of Plant Diseases. Frontiers in Plant Science, 10.

[22] Goodfellow, I. J., Bengio, Y., and Courville, A. (2016). Deep Learning. Cambridge,

MA: MIT Press. Available online at: http://www.deeplearningbook.org

[23] Wu, M., et al. (2024). Deep Learning for Image Classification: A Review. Singapore: Springer Nature Singapore.

[24] Huo, Y., et al. (2023). Vision Transformer (ViT)-based Applications in Image Classification.

[25]Hussain, A., et al. (2018). Automatic Disease Detection in Wheat Crop using Convolution Neural Network.

[26] Ouhami, M., et al. (2021). "Computer Vision, IoT and Data Fusion for Crop Disease Detection Using Machine Learning: A Survey and Ongoing Research." Remote Sensing.

[27] https://www.kaggle.com/datasets/vipoooool/new-plant-diseases-dataset 1\9\2024

APPENDIX: CNN-Based Crop Health Monitoring Code

```
import os
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import tensorflow as tf
from tensorflow.keras import layers, models
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.optimizers import Adam
from sklearn.metrics import classification_report, confusion_matrix
import warnings
# Suppress warnings
warnings.filterwarnings("ignore", category=UserWarning)
# Suppress TensorFlow logging messages
tf.get_logger().setLevel('ERROR')
np.random.seed(42)
tf.random.set_seed(42)
class PotatoDiseaseCNN:
    def _init_(self, train_dir, validation_dir, test_dir, img_height=255,
img_width=255, batch_size=32):
        self.train_dir = train_dir
```

```
        self.validation_dir = validation_dir
        self.test_dir = test_dir
        self.img_height = img_height
        self.img_width = img_width
        self.batch_size = batch_size
        self.classes = ['1-Potato healthy', '2-Potato early blight', '3-Potato late blight']
        # Data augmentation and preprocessing
        self.train_datagen = ImageDataGenerator(
            rescale=1./255,
            rotation_range=20,
            width_shift_range=0.2,
            height_shift_range=0.2,
            shear_range=0.2,
            zoom_range=0.2,
            horizontal_flip=True
        )
        self.validation_datagen = ImageDataGenerator(rescale=1./255)
        self.test_datagen = ImageDataGenerator(rescale=1./255)
    def load_data(self):
        # Load training data
```

```
        self.train_generator = self.train_datagen.flow_from_directory(
            self.train_dir,
            target_size=(self.img_height, self.img_width),
            batch_size=self.batch_size,
            class_mode='categorical'
        )
        # Load validation data
        self.validation_generator =
self.validation_datagen.flow_from_directory(
            self.validation_dir,
            target_size=(self.img_height, self.img_width),
            batch_size=self.batch_size,
            class_mode='categorical'
        )
        # Load test data
        self.test_generator = self.test_datagen.flow_from_directory(
            self.test_dir,
            target_size=(self.img_height, self.img_width),
            batch_size=1,
            class_mode='categorical',
            shuffle=False
```

```
    )
def build_model(self):
    self.model = models.Sequential([
        layers.Conv2D(32, (3, 3), activation='relu',
         input_shape=(self.img_height, self.img_width, 3)),
        layers.MaxPooling2D((2, 2)),
        layers.BatchNormalization(),
        layers.Conv2D(64, (3, 3), activation='relu'),
        layers.MaxPooling2D((2, 2)),
        layers.BatchNormalization(),
        layers.Conv2D(128, (3, 3), activation='relu'),
        layers.MaxPooling2D((2, 2)),
        layers.BatchNormalization(),
        layers.Flatten(),
        layers.Dense(256, activation='relu'),
        layers.Dropout(0.5),
        layers.Dense(len(self.classes), activation='softmax')
    ])
    self.model.compile(optimizer=Adam(learning_rate=0.0001),
                  loss='categorical_crossentropy', metrics=['accuracy'])
    return self.model
```

```
    def train_model(self, max_epochs=15):
        history = self.model.fit(
            self.train_generator,
            steps_per_epoch=self.train_generator.samples // self.batch_size,
            validation_data=self.validation_generator,
            validation_steps=self.validation_generator.samples //
self.batch_size,
            epochs=max_epochs
        )
        # Save the trained model
        model_path = "D:/test/app1.h5"
        os.makedirs(os.path.dirname(model_path), exist_ok=True)
        self.model.save(model_path)
        print(f"Model saved to {model_path}")
        return history
    def plot_training_metrics(self, history):
        epochs = range(1, len(history.history['accuracy']) + 1)
        plt.figure(figsize=(12, 5))
        # Accuracy Plot
        plt.subplot(1, 2, 1)
        plt.plot(epochs, history.history['accuracy'], label='Train Accuracy')
```

```
        plt.plot(epochs, history.history['val_accuracy'], label='Validation
Accuracy')
        plt.title('Accuracy Comparison')
        plt.xlabel('Epochs')
        plt.ylabel('Accuracy')
        plt.legend()
        # Loss Plot
        plt.subplot(1, 2, 2)
        plt.plot(epochs, history.history['loss'], label='Train Loss')
        plt.plot(epochs, history.history['val_loss'], label='Validation Loss')
        plt.title('Loss Comparison')
        plt.xlabel('Epochs')
        plt.ylabel('Loss')
        plt.legend()
        plt.tight_layout()
        plt.show()
    def plot_comparison_bars(self, metrics_1, metrics_2, labels, title):
        width = 0.4  # Bar width
        x = range(len(labels))
        plt.figure(figsize=(10, 6))
        plt.bar([p - width/2 for p in x], metrics_1, width, label='Set 1')
```

```
    plt.bar([p + width/2 for p in x], metrics_2, width, label='Set 2')
    plt.xticks(x, labels)
    plt.xlabel('Metrics')
    plt.ylabel('Scores')
    plt.title(title)
    plt.legend()
    plt.show()
def evaluate_model(self):
    predictions = self.model.predict(self.test_generator)
    y_pred = np.argmax(predictions, axis=1)
    y_true = self.test_generator.classes
    # Confusion Matrix
    cm = confusion_matrix(y_true, y_pred)
    plt.figure(figsize=(8, 6))
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
                xticklabels=self.classes,
                yticklabels=self.classes)
    plt.title('Confusion Matrix')
    plt.xlabel('Predicted')
    plt.ylabel('True')
    plt.show()
```

```
        # Classification Report
        report = classification_report(y_true, y_pred,
target_names=self.classes, output_dict=True)
          # Visualizing Classification Report as Heatmap
        metrics = ['precision', 'recall', 'f1-score']
        report_values = np.array([[report[cls][metric] for metric in metrics]
for cls in self.classes])
        plt.figure(figsize=(8, 6))
        sns.heatmap(report_values, annot=True, cmap='Blues',
xticklabels=metrics, yticklabels=self.classes, fmt='.2f')
        plt.title('Classification Report')
        plt.xlabel('Metrics')
        plt.ylabel('Classes')
        plt.show()
    def predict_random_images(self, num_images=3):
        # Display random test images and their predictions
        for class_name in self.classes:
            print(f"\nRandom {class_name} images:")
            class_dir = os.path.join(self.test_dir, class_name)
            image_files = os.listdir(class_dir)
            selected_images = np.random.choice(image_files, num_images,
replace=False)
```

```
        plt.figure(figsize=(15, 5))  # Set up the figure size
        for i, img_file in enumerate(selected_images):
            img_path = os.path.join(class_dir, img_file)
            img = tf.keras.preprocessing.image.load_img(img_path,
target_size=(self.img_height, self.img_width))
            img_array = tf.keras.preprocessing.image.img_to_array(img) /
255.0
            img_array = np.expand_dims(img_array, axis=0)
            prediction = self.model.predict(img_array)
            predicted_class = self.classes[np.argmax(prediction)]
            # Plot the image
            plt.subplot(1, num_images, i + 1)
            plt.imshow(img)
            plt.title(f"Actual: {class_name}\nPredicted:
{predicted_class}")
            plt.axis('off')
        plt.tight_layout()
        plt.show()
# Main Execution
def main():
    train_dir = 'D:/Dataset/Dataset_training'
    validation_dir = 'D:/Dataset/validation'
```

```
test_dir = 'D:/Dataset/dataset_test'
# Display one image from each class (healthy, early blight, late blight)
classes = ['1-Potato healthy', '2-Potato early blight', '3-Potato late blight']
images = []
for class_name in classes:
    class_dir = os.path.join(train_dir, class_name)
    img_path = os.path.join(class_dir, os.listdir(class_dir)[0])
    img = tf.keras.preprocessing.image.load_img(img_path, target_size=(255, 255))
    img_array = tf.keras.preprocessing.image.img_to_array(img)
    images.append(img_array)
# Stack images horizontally
merged_image = np.concatenate(images, axis=1)  # Concatenate along the width
# Display the merged image with labels
plt.figure(figsize=(12, 5))
for i, image in enumerate(images):
    plt.subplot(1, 3, i + 1)
    plt.imshow(image.astype('uint8'))
    plt.title(classes[i])
    plt.axis('off')
```

```
    plt.tight_layout()
    plt.show()



    classifier = PotatoDiseaseCNN(train_dir, validation_dir, test_dir)
    classifier.load_data()
    classifier.build_model()
    history = classifier.train_model()
    # Plot Training Metrics
    classifier.plot_training_metrics(history)
    # Evaluate Model
    classifier.evaluate_model()
    # Predict random images
    classifier.predict_random_images()
if _name_ == "_main_":
    main()
```