\documentclass[10pt]{report}

\usepackage{blindtext}

\usepackage{times}

\usepackage{graphicx}

\usepackage{geometry}

\usepackage{sectsty}

\usepackage{float}

\usepackage{array}

\usepackage[backend=biber,bibencoding=latin1]{biblatex}

\chapterfont{\centering}

\usepackage{enumitem}

\usepackage{enumerate}

\usepackage{listings}

\usepackage{color}

\usepackage{ragged2e}

\usepackage{pdfpages}

\usepackage{draftwatermark}

\usepackage[headheight=0pt,headsep=0pt]{geometry}

\definecolor{dkgreen}{rgb}{0,0.6,0}

\definecolor{gray}{rgb}{0.5,0.5,0.5}

\definecolor{mauve}{rgb}{0.58,0,0.82}

\makeatletter

\def\@makechapterhead#1{%

%%%%\vspace\*{50\p@}% %%% removed!

{\parindent \z@ \centering\normalfont

\ifnum \c@secnumdepth >\m@ne

\huge\bfseries \@chapapp\space \thechapter

\par\nobreak

\vskip 20\p@

\fi

\interlinepenalty\@M

\Huge \bfseries #1\par\nobreak

\vskip 40\p@

}}

\def\@makeschapterhead#1{%

%%%%%\vspace\*{50\p@}% %%% removed!

{\parindent \z@ \centering

\normalfont

\interlinepenalty\@M

\Huge \bfseries #1\par\nobreak

\vskip 40\p@

}}

\makeatother

\lstset{ %

language=Java, % the language of the code

basicstyle=\footnotesize, % the size of the fonts that are used for the code

numbers=left, % where to put the line-numbers

numberstyle=\tiny\color{gray}, % the style that is used for the line-numbers

stepnumber=1, % each line is numbered

numbersep=5pt, % how far the line-numbers are from the code

backgroundcolor=\color{white}, % choose the background color. You must add \usepackage{color}

showspaces=false, % show spaces adding particular underscores

showstringspaces=false, % underline spaces within strings

showtabs=false, % show tabs within strings adding particular underscores

frame=single, % adds a frame around the code

rulecolor=\color{black}, % if not set, the frame-color may be changed on line-breaks within notblack

text (e.g. commens (green here))

tabsize=2, % sets default tabsize to 2 spaces

captionpos=b, % sets the caption-position to bottom

breaklines=true, % sets automatic line breaking

breakatwhitespace=false, % sets if automatic breaks should only happen at whitespace

title=\lstname, % show the filename of files included with \lstinputlisting;

% also try caption instead of title

keywordstyle=\color{blue}, % keyword style

commentstyle=\color{dkgreen}, % comment style

stringstyle=\color{mauve}, % string literal style

escapeinside={\%\*}{\*)}, % if you want to add a comment within your code

morekeywords={\*,...} % if you want to add more keywords to the set

}

\geometry{a4paper,total={180mm,250mm},left=20mm,top=20mm, right=20mm}

\thispagestyle{empty}

\begin{document}

\newpage

\begin{center}

\thispagestyle{empty}

\LARGE{\textsc {\textbf{\textcolor{blue}{PLANT DISEASE DETECTION DETECTION AND PREVENCTION CNN}}}}\\[0.2cm]

\vspace{0.2cm}

\Large{\textit{\textcolor{blue}{\\Major project report submitted \\in partial fulfillment of the

requirement

for award of the degree of}}}\\[0.3cm]

\Large{\textbf{\textcolor{blue}{\\Bachelor of Technology\\in \\Computer Science \& Engineering}}}

\vspace{0.5cm}

\Large{\textbf{\textcolor{blue}{\\By}}}\\[0.5cm]

\begin{table}[h]

\centering

\Large{\textcolor{blue}{

\begin{tabular}{>{\bfseries}lc>{\bfseries}r}

GUTHA SAINATH REDDY &(20UECS0368) & (17707)\\AMUDALA BHANU TEJA & (20UECS0046)&(18300)\\AMUDALA POOJITHA &(20UECS0047) & (18301)\\

\end{tabular}}}

\end{table}

\vspace{0.5cm}

\large{\textit{\textcolor{blue}{Under the guidance of}}}\\

\large{\textit{\textcolor{blue}{Mr. IGNATIOUS K PIOUS,ME.,\\

ASSISTANT PROFESSOR}

}}\\

\vspace{0.5cm}

\includegraphics[scale=1.0]{Veltech logo.png}\\

\vspace{1cm}

\large{\textbf{\textcolor{blue}{DEPARTMENT OF COMPUTER SCIENCE \& ENGINEERING}}}\\

\large{\textbf{\textcolor{blue}{SCHOOL OF COMPUTING}}}\\

\vspace{0.5cm}

\Large{\textbf{\textcolor{blue}{VEL TECH RANGARAJAN DR. SAGUNTHALA R\&D INSTITUTE OF

SCIENCE \& TECHNOLOGY\\

\vspace{0.2cm}

(Deemed to be University Estd u/s 3 of UGC Act,

1956)}}}\\\Large{\textbf{\textcolor{blue}{Accredited by NAAC with A++ Grade}}}\\

\large{\textbf{\textcolor{blue}{CHENNAI 600 062, TAMILNADU, INDIA}}}

\vspace{0.4cm}

\large{\textbf{\textcolor{blue}{\\May, 2024}}}\\

\end{center}

\newpage

\begin{center}

\thispagestyle{empty}

\LARGE{\textsc {\textbf{\textcolor{blue}{PLANT DISEASE DETECTION DETECTION AND PREVENCTION CNN}}}}\\[0.2cm]

\vspace{0.2cm}

\Large{\textit{\textcolor{blue}{\\Major project report submitted \\in partial fulfillment of the

requirement

for award of the degree of}}}\\[0.3cm]

\Large{\textbf{\textcolor{blue}{\\Bachelor of Technology\\in \\Computer Science \& Engineering}}}

\vspace{0.5cm}

\Large{\textbf{\textcolor{blue}{\\By}}}\\[0.5cm]

\begin{table}[h]

\centering

\Large{\textcolor{blue}{

\begin{tabular}{>{\bfseries}lc>{\bfseries}r}

GUTHA SAINATH REDDY&(20UECS0368) & (17707)\\AMUDALA BHANU TEJA & (20UECS0046)&(18300)\\AMUDALA POOJITHA&(20UECS0047) & (18301)\\

\end{tabular}}}

\end{table}

\vspace{0.5cm}

\large{\textit{\textcolor{blue}{Under the guidance of}}}\\

\large{\textit{\textcolor{blue}{Mr. IGNATIOUS K PIOUS,ME.,\\

ASSISTANT PROFESSOR}

}}\\

\vspace{0.5cm}

\includegraphics[scale=1.0]{Veltech logo.png}\\

\vspace{1cm}

\large{\textbf{\textcolor{blue}{DEPARTMENT OF COMPUTER SCIENCE \& ENGINEERING}}}\\

\large{\textbf{\textcolor{blue}{SCHOOL OF COMPUTING}}}\\

\vspace{0.5cm}

\Large{\textbf{\textcolor{blue}{VEL TECH RANGARAJAN DR. SAGUNTHALA R\&D INSTITUTE OF

SCIENCE \& TECHNOLOGY\\

\vspace{0.2cm}

(Deemed to be University Estd u/s 3 of UGC Act,

1956)}}}\\\Large{\textbf{\textcolor{blue}{Accredited by NAAC with A++ Grade}}}\\

\large{\textbf{\textcolor{blue}{CHENNAI 600 062, TAMILNADU, INDIA}}}

\vspace{0.4cm}

\large{\textbf{\textcolor{blue}{\\May, 2024}}}\\

\end{center}

%CERTIFICATE

\newpage

\pagenumbering{roman}

\begin{center}

{\Huge \textbf{CERTIFICATE}}\\[1cm]

\end{center}

\linespread{1.5}

\large{It is certified that the work contained in the project report titled "PLANT DISEASE DETECTION DETECTION AND PREVENCTION CNN" by "GUTHA SAINATH REDDY & (20UECS0368), AMUDALA BHANU TEJA & (20UECS0046), AMUDALA POOJITHA & (20UECS0047)" has been carried out under my supervision and that this work has not

been submitted elsewhere for a degree.}

\vspace{12.5cm}

\begin{flushleft}

\textbf{Signature of Supervisor\hfill\textbf{Signature of Professor In-charge}\\\\Computer

Science \& Engineering\hfill\textbf{Computer

Science \& Engineering}\\School of Computing\hfill\textbf{School of Computing}\\Vel Tech

Rangarajan Dr. Sagunthala R\&D\hfill\textbf{Vel Tech Rangarajan Dr. Sagunthala R\&D}\\Institute of

Science \& Technology\hfill\textbf{Institute of

Science \& Technology}\\May, 2024\hfill\textbf{May, 2024}\\\hfill\textbf{}\\}\hfill\textbf{}\\\

\end{flushleft}

%declaration

\newpage

\begin{center}

\Huge \textbf{DECLARATION}

\end{center}

\vspace{1.0cm}

\linespread{1.5}

\large{

We declare that this written submission represents my ideas in our own words and where others'

ideas

or words have been included, we have adequately cited and referenced the original sources. We also

declare that we have adhered to all principles of academic honesty and integrity and have not

misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We

understand

that any violation of the above will be cause for disciplinary action by the Institute and can also

evoke

penal action from the sources which have thus not been properly cited or from whom proper

permission

has not been taken when needed.}

\vspace{2.0cm}

\begin{flushright}

\\

\large{(GUTHA SAINATH REDDY)}\\

\large{Date:\hspace\*{1.0cm}/\hspace\*{1.0cm}/}\\[2.0cm]

\\

\large{(AMUDALA BHANU TEJA)}\\

\large{Date:\hspace\*{1.0cm}/\hspace\*{1.0cm}/}\\[2.0cm]

\\

\large{(AMUDALA POOJITHA)}\\

\large{Date:\hspace\*{1.0cm}/\hspace\*{1.0cm}/}\\[2.0cm]

\end{flushright}

\newpage

%approval sheet

\newpage

\begin{center}

\Huge\textbf{APPROVAL SHEET}\\

\vspace{1.0cm}

\end{center}

\linespread{1.5}

\justifying{

\large{This project report entitled (PLANT DISEASE DETECTION DETECTION AND

PREVENCTION CNN)) by (GUTHA SAINATH REDDY

(20UECS0368), (AMUDALA BHANU TEJA (20UECS0046), (AMUDALA POOJITHA (20UECS0047) is approved for

the

degree of B.Tech in Computer Science \& Engineering.}\\}

\vspace{4.0cm}

\begin{flushleft}

\Large \textbf{Examiners} \hfill \Large \textbf{Supervisor}\\

\end{flushleft}

\begin{flushright}

Mr. Ignatious K Pious, ME(Networking)

\end{flushright}

\vspace{1.0cm}

\begin{flushleft}

\large{\textbf{Date:\hspace\*{1.0cm}/\hspace\*{2.0cm}/}}\\

\large{\textbf{Place:}}

\end{flushleft}

%acknowledgment

\newpage

\begin{center}

\LARGE{\textbf{ACKNOWLEDGEMENT}}\\[1cm]

\end{center}

\linespread{1.13}

\large{\paragraph{}We express our deepest gratitude to our respected \textbf{Founder Chancellor

and

President Col. Prof. Dr. R. RANGARAJAN B.E. (EEE), B.E. (MECH), M.S (AUTO),D.Sc., Foundress

President

Dr. R. SAGUNTHALA RANGARAJAN M.B.B.S.} Chairperson Managing Trustee and Vice President.}

\large{\paragraph{}We are very much grateful to our beloved \textbf{Vice Chancellor Prof. S.

SALIVAHANAN,} for providing us with an environment to complete our project successfully.}

\large{\paragraph{}We record indebtedness to our \textbf{Professor \& Dean, Department of

Computer

Science \& Engineering, School of Computing, Dr. V. SRINIVASA RAO, M.Tech., Ph.D.,} for immense

care and encouragement

towards us throughout the course of this project.}

\large{\paragraph{}We are thankful to our \textbf{Head, Department of Computer

Science \& Engineering,Dr.M.S. MURALI DHAR, M.E., Ph.D.,} for providing immense support in all our

endeavors.}

\large{\paragraph{}We also take this opportunity to express a deep sense of gratitude to our Internal

Supervisor \textbf{Mr. IGNATIOUS K PIOUS,M.E.,} for his/her cordial support, valuable

information and guidance, he/she helped us in completing this project through various stages. }

\large{\paragraph{}A special thanks to our \textbf{Project Coordinators Mr. V. ASHOK KUMAR,

M.Tech., Ms. C. SHYAMALA KUMARI, M.E.,} for their valuable guidance and support throughout the

course of the project.}

\large{\paragraph{}We thank our department faculty, supporting staff and friends for their help and

guidance to complete this project.}

\vspace{2.0cm}

\begin{flushright}

\begin{tabular}{>{\bfseries}lc>{\bfseries}r}

GUTHA SAINATH REDDY & & (20UECS0368)\\AMUDALA BHANU TEJA & & (20UECS0046)\\AMUDALA POOJITHA & &

(20UECS0047)\\

\end{tabular}

\end{flushright}

%ABSTRACT

\newpage

\begin{center}

\vspace{2cm}

\Large{\textbf{ABSTRACT}}\\[0.5cm]

\end{center}

\begin{center}

\addtocontents{toc}{~\hfill\textbf{Page.No}\par}

\addcontentsline{toc}{chapter}{ABSTRACT}

\addtocontents{toc}{\protect\thispagestyle{empty}}

\end{center}

\vspace{-5em}

\Large{\paragraph\\

Plant diseases cause a catastrophic influence on food production safety; they reduce the eminence and quantum of agricultural products. Plant diseases may cause significantly high loss or no harvest in dreadful cases. Various diseases and pests affect the growth of plants at different stages and crop yields. This study concentrated on diseases detection and prevention of plants Leaf Spot. Various methods have been proposed for plant disease detection, and deep learning has become the preferred method because of its spectacular accomplishment. In this study, CNN was used to remove the unwanted background of an input image by selecting multiscale features. This work proposes a plant disease detection approach using the EfficientNetV2 model. A comprehensive set of experiments was carried out to ascertain the performance of the proposed approach and compare it with other models such as Efficient Net and Convolutional Neural Network (CNN). The experimental results showed that the proposed approach achieved a detection accuracy of 93.26 percent.This high level of accuracy is a testament to the effectiveness of deep learning techniques in plant disease detection. Such precise identification of Leaf Spot can enable farmers to take timely and targeted measures to mitigate the spread of the disease, thereby minimizing yield losses and ensuring food security. The implications of our findings extend beyond academic research, offering practical solutions for enhancing agricultural productivity and food safety. By integrating deep learning-based disease detection systems into agricultural practices, farmers can make informed decisions regarding pest management, resource allocation, and crop protection strategies. Furthermore, the scalability and adaptability of our approach make it suitable for deployment in diverse agricultural settings worldwide. Looking ahead, there are several avenues for future research and improvement. Fine-tuning the model architecture and optimizing hyperparameters could potentially further enhance detection accuracy. Additionally, exploring real-time monitoring systems and integrating remote sensing technologies could facilitate early disease detection and intervention, thereby offering a proactive approach to plant disease management.

}\\

\vspace{0.5cm}

\noindent \textbf{Keywords: Convolutional Neural Networks,

Deep Learning, Image Processing, Computer Vision, Disease Classification, Feature Extraction, Transfer Learning, Supervised Learning, Multi modal Data,

Dataset Augmentation, Preprocessing Techniques, Model Training,

Model Evaluation,

}

\textbf{}

%list of figure

\newpage

\renewcommand\*\listfigurename{LIST OF FIGURES}

\addcontentsline{toc}{chapter}{LIST OF FIGURES}

\listoffigures

%list of abbreviation

\newpage

\newlist{abbrv}{itemize}{1}

\setlist[abbrv,1]{label=,labelwidth=1in,align=parleft,itemsep=0.1\baselineskip,leftmargin=!}

\chapter\*{LIST OF ACRONYMS AND ABBREVIATIONS}

\chaptermark{LIST OF ACRONYMS AND ABBREVIATIONS}

\addcontentsline{toc}{chapter}{LIST OF ACRONYMS AND ABBREVIATIONS}

\begin{abbrv}

\item[abbr] Abbreviation

\item[ADAM] Adaptive Moment Estimation

\item[API] Application Programming Interface

\item[CNN] Convolutional Neural Network

\item[CPU] Central Processing Unit

\item[CV] Computer Vision

\item[DL] Deep Learning

\item[F1] F1 Score (Harmonic Mean of Precision and Recall)

\item[FN] False Negative

\item[FP] False Positive

\item[GPU] Graphics Processing Unit

\item[GUI] Graphical User Interface

\item[RMSProp] Root Mean Square Propagation

\item[ROC] Receiver Operating Characteristic

\item[ROI] Region of Interest

\item[SGD] Stochastic Gradient Descent

\item[TN] True Negative

\item[TP] True Positive

\end{abbrv}

\newpage

\renewcommand\*\contentsname{TABLE OF CONTENTS}

%\addtocontents{toc}{\textbf{CONTENT} \hfill \textbf{PAGE NO.}}

\tableofcontents

\addtocontents{toc}{\protect\pagestyle{empty}}

\thispagestyle{empty}

%introduction

\chapter{INTRODUCTION}

\pagenumbering{arabic}

\section{Introduction}

\hspace{0.5cm}

\text{Plant diseases refer to various abnormal conditions, disorders, or pathogens that adversely affect the health and vitality of plants. These diseases can manifest in numerous ways, including physical damage, poor growth, reduced yield, and even plant death. They pose significant challenges to agricultural, horticultural, and forestry practices worldwide. Pathogens are microorganisms such as bacteria, fungi, viruses, and nematodes that invade plant tissues, disrupting normal cellular functions. They may spread through soil, water, air, or by vectors. Abiotic factors like extreme temperatures, drought, excessive moisture, or pollution can weaken plants, making them more susceptible to diseases. Some plants are genetically predisposed to specific diseases. Breeding programs aim to develop disease-resistant varieties. Poor cultural practices, such as overcrowding, improper watering, or inadequate nutrition, can stress plants and create conditions favorable for diseases. Preventing plant diseases is essential for maintaining healthy crops and ecosystems. Avoid planting the same crop in the same location year after year to break the life cycle of pathogens. Choose plant varieties bred for resistance to prevalent diseases in your region. Keep planting areas clean by removing diseased plant material, including fallen leaves and fruits. Avoid overwatering, which can create conditions conducive to fungal diseases, and water plants at the base to minimize leaf wetness. Plant With adequate spacing to promote airflow and reduce humidity around plants. Maintain healthy soil through organic matter addition and proper pH adjustments. Preventing and managing plant diseases requires vigilance and a combination of strategies tailored to specific plant types, diseases, and environmental conditions. A proactive approach is essential to protect plant health and ensure robust agricultural and horticultural practices. Convolutional neural networks (CNNs) are a class of deep learning algorithms inspired by the organization of the animal visual cortex. These neural networks are particularly well suited for image recognition tasks due to their ability to automatically learn hierarchical representations of visual features from raw pixel data.}

\linespread{1.5}

\section{Aim of the project}

\hspace{0.5cm}Modern image processing and deep learning-based techniques are widely used for the detection of plant

leaf disease. Many diagnostic methods use a Convolutional Neural Network (CNN) and a pre-trained

model to detect and classify healthy and unhealthy plants.

\section{Project Domain}

Plant disease detection and prevention using CNN falls within the domain of machine learning. CNNs, a subset of artificial neural networks, excel at analyzing visual data like images. In this application, CNNs are trained on labeled datasets containing images of healthy and diseased plants. During training, the CNN learns to automatically extract features from the images, enabling it to differentiate between healthy and infected plants based on visual symptoms. Once trained, the CNN can accurately classify unseen plant images, enabling early disease detection. By integrating CNN-based systems into agricultural workflows, farmers can quickly identify and respond to plant diseases, mitigating crop losses and reducing reliance on chemical treatments. Overall, plant disease detection and prevention using CNNs exemplifies the capacity of machine learning to revolutionize agricultural practices, enhancing food security and sustainability.

\section{Scope of the Project}

Modern image processing and deep learning-based techniques are widely used for the detection of plant leaf disease. Many diagnostic methods use a Convolutional Neural Network (CNN) and a pre-trained model to detect and classify healthy and unhealthy plants, used segmentation to remove background data and applied a trained neural network for classification. Zhang et al. proposed a method to diagnose cucumber plant diseases by separating images with diseased patches by combining K-means, the condition and color of infected leaf lesions, and separating unhealthy leaf images using scant resentment. Proposed method improved Ocular-head deep neural networks for classification by reading the feature maps highlighting the essential regions that also weaken the meaningless connected layers. Various artificial intelligence and image-based plant disease detection approaches have been proposed.

%literature review

\chapter{LITERATURE REVIEW}

[1] Elhoucine Elfatimi, Recep Eryigit, Lahcen Elfatimi,

In recent years, plant leaf diseases has become a widespread problem for which an accurate research and rapid application of deep learning in plant disease classification is required, beans is also one of the most important plants and seeds which are used worldwide for cooking in either dried or fresh form, beans are a great source of protein that offer many health benefits, but there are a lot of diseases associated with beans leaf which hinder its production such as angular leaf spot disease and bean rust disease. Thus, an accurate classification of bean leaf diseases is needed to solve the problem in the early stage. A deep learning approach is proposed to identify and classify beans leaf disease by using public dataset of leaf image and MobileNet model with the open source library TensorFlow. In this study, we proposed a method to classify beans leaf disease and to find and describe the efficient network architecture (hyperparameters and optimization methods). Moreover, after applying each architecture separately, we compared their obtained results to find out the best architecture configuration for classifying bean leaf diseases and their results. Furthermore, to satisfy the classification requirements, the model was trained using MobileNetV2 architecture under the some controlled conditions as MobileNet to check if we could get faster training times, higher accuracy and easier retraining, we evaluated and implemented MobileNet architectures on one public dataset including two unhealthy classes.

\newline

[2] Lili Li, Shujuan Zhang, Bin Wang, Deep learning is a branch of artificial intelligence. In recent years, with the advantages of automatic learning and feature extraction, it has been widely concerned by academic and industrial circles. It has been widely used in image and video processing, voice processing, and natural language processing. At the same time, it has also become a research hotspot in the field of agricultural plant protection, such as plant disease recognition and pest range assessment, etc. The application of deep learning in plant disease recognition can avoid the disadvantages caused by artificial selection of disease spot features, make plant disease feature extraction more objective, and improve the research efficiency and technology transformation speed.

\newline

[3] Manivarsh Adi, Abhishek Kumar Singh, Harinath Reddy A, Yeshwanth Kumar, Venkata Reddy Challa, Pooja Rana, Usha Mittal, Disease detection in plants is one of the major concerns for farmers nowadays. As many new techniques like Deep Learning capability to dive into deep analysis and computation made it one of the prominent techniques for plant leaf disease detection. Mobile applications with inbuilt deep learning models are helping farmers to detect and classify the disease throughout the world. It consists of disparate techniques like ANN and CNN to diagnose the disease in plant leaves. It uses key features of images to detect and diagnose the type of diseases present in leaves. Some pre-trained models like AlexNet, GoogleNet, LeNet, ResNet, VGGNET and Inception with a huge number of learnable parameters had shown classification or detection of disease in leaves. This paper focused on different architecture like predefined and user defined models that were used for detection of diseases in plant leaves.

\newline

[4] Mobeen Ahmad, Muhammad Abdullah, Hyeonjoon Moon, Dongil Han, Convolutional neural networks have demonstrated state-of-the-art performance in image classification and various other computer vision tasks. Plant disease detection is an important area of deep learning which has been addressed by many recent methods. However, there is a dire need to optimize these solutions for resource-constrained portable devices such as smartphones. This is a challenging problem because deep learning models are resource extensive in nature. This paper proposes an efficient method to systematically classify plant disease symptoms using convolutional neural networks. These networks are memory efficient and when coupled with the proposed training configuration it enables rapid development of industrial applications by reducing the training times.

\newline

[5] Sabbir Ahmed, MD.Bakhtiar Hasan, Tasnim Ahmed, MD.Redwan Karim sony, MD.Hasanul Kabir, To ensure global food security and the overall profit of stakeholders, the importance of correctly detecting and classifying plant diseases is paramount. In this connection, the emergence of deep learning-based image classification has introduced a substantial number of solutions. However, the applicability of these solutions in low-end devices requires fast, accurate, and computationally inexpensive systems. This work proposes a lightweight transfer learning-based approach for detecting diseases from tomato leaves. It utilizes an effective preprocessing method to enhance the leaf images with illumination correction for improved classification. Our system extracts features using a combined model consisting of a pretrained MobileNetV2 architecture and a classifier network for effective prediction.

\linespread{1.5}

%PROJECT DESCRIPTION

\chapter{PROJECT DESCRIPTION}

\linespread{1.5}

\section{Existing System}

The implementation of proper techniques to identify healthy and diseased leaves helps in controlling crop loss and increasing productivity. This section comprises different existing machine-learning techniques for the identification of plant diseases. The decrease in time per epoch was because the number of parameters in these models was quite smaller than that of other existing models. The decrease in time per epoch was because the number of parameters in these models was quite smaller than that of other existing models. To enhance the training and efficacy, it uses Fused MB-Conv for the first three stages and MB-Conv for the subsequent stages and this is slow than existing models, which are up to 6.8x smaller. In a computer vision and machine learning based techniques were developed for plant leaf disease detection. Real-time plant disease detection has some significant challenges, such as complex background and severity of the disease due to the images being captured in real-time scenarios from the farm field. These models undergo rigorous training processes where they learn to automatically extract intricate features from the images, enabling accurate classification of plant diseases. Through meticulous data preprocessing and model optimization, the CNNs achieve high levels of accuracy in disease detection. Once trained and validated, the CNN models are deployed in real-world scenarios, integrated into agricultural technologies such as drones or mobile applications. Continuously monitoring plants for signs of disease, the system facilitates early detection and timely intervention, empowering farmers to implement targeted preventive measures and mitigate crop losses.

\newline

\linespread{1.5}

\textbf{Disadvantages of existing system}

• The traditional KNN is a simple supervised-machine-learning algorithm used in classification problems.

• Hyper parameters that deal with training include batch size, learning rate, and dropout the loss function is expressed.

\section{Proposed System}

We propose this article “Plant Disease Detection Approach Using EfficientNetV2”. In this study, we proposed a plant leaf disease detection approach by employing a background removal technique to remove the complex background of the image by using U2 -Net. EfficientNetV2 deep learning model is used for classification. A comprehensive set of experiments was carried out to ascertain the performance of the proposed approach and compare it with other models such as Efficient-Net and Convolutional Neural Network (CNN). A plant leaf disease detection approach was proposed using EfficientNetV2. In most cases, computer vision algorithms remove the background from an image, such as image thresholding in OpenCV and grab hut techniques. The background removal approach used in this study was U2 –Net. A set of experiments was conducted to ascertain the detection efficiency of the proposed approach.

\newline

\textbf{Advantages of Proposed system}

• Transfer learning has several advantages; for example, it does not need a large amount of data to train the network.

• This is hybrid deep learning model has the advantages of a residual network and retains unique properties.

• Their technical status, performance and stability are directly related to the train operation safety.

\section{Feasibility Study}

A feasibility study of plant disease detection and prevention using CNNs involves assessing technical, economic, and practical aspects. Firstly, the availability of labeled datasets and computational resources for training and deploying CNN models must be evaluated. Accuracy and reliability are critical, comparing CNN performance to traditional methods in detecting and classifying diseases. Cost analysis considers expenses for data collection, infrastructure, personnel, and maintenance. Scalability is crucial for adapting the system to diverse agricultural settings. Integration with existing systems, regulatory compliance, and ethical considerations are also examined. User acceptance and adoption are essential, requiring stakeholder engagement and feedback. By thoroughly assessing these factors, stakeholders can determine the viability and potential benefits of implementing CNN-based solutions.

\subsection{Economic Feasibility}

The economic feasibility of plant disease detection and prevention using CNNs involves evaluating the costs and benefits associated with implementing such a system. Initial costs include expenses for data collection, hardware and software infrastructure, model development, and personnel training. Ongoing costs may include maintenance, updates, and data storage. However, the long-term benefits of CNN-based disease detection can outweigh these costs by reducing crop losses, increasing yields, and optimizing resource utilization. By enabling early detection and targeted intervention, CNNs can minimize the need for costly chemical treatments and labor-intensive manual inspections. Additionally, the scalability of CNN-based systems allows for broader adoption across diverse agricultural settings, maximizing their economic impact. Overall, while initial investments may be significant, the potential economic returns and improvements in agricultural productivity make plant disease detection and prevention using CNNs economically feasible and advantageous for farmers and agricultural stakeholders.

\subsection{Technical Feasibility}

The technical feasibility of plant disease detection and prevention using CNNs hinges on several factors. Firstly, the availability of high-quality labeled datasets containing images of diseased and healthy plants is crucial for training accurate models. Additionally, the computational resources required for training and deploying CNNs, such as GPUs and deep learning frameworks, must be accessible. Preprocessing techniques and optimization strategies are employed to enhance model performance and efficiency. Integration with existing agricultural technologies and practices is essential to ensure seamless deployment and compatibility. Furthermore, the scalability of CNN-based systems enables adaptation to various crops, regions, and environmental conditions. Overall, while technical challenges such as data quality, computational requirements, and integration complexities may exist, advancements in machine learning and computer vision make plant disease detection and prevention using CNNs technically feasible and increasingly accessible for agricultural applications. Despite technical challenges such as data variability, computational complexity, and integration hurdles, ongoing advancements in machine learning, computer vision, and agricultural technology continue to enhance the technical feasibility of CNN-based solutions for plant disease detection and prevention.

\subsection{Social Feasibility}

The social feasibility of employing CNNs for plant disease detection and prevention involves assessing the acceptance, impact, and implications of such technology within society. Stakeholder engagement and awareness campaigns are crucial for garnering support and addressing concerns among farmers, agricultural communities, and relevant authorities. Ensuring accessibility and affordability of CNN-based solutions is essential to promote equitable access to agricultural innovations across diverse socio-economic backgrounds. Moreover, collaboration with local communities and extension services can facilitate knowledge sharing and capacity building, empowering farmers to effectively utilize CNN-based tools for disease management. Ethical considerations regarding data privacy, ownership, and equitable distribution of benefits must also be addressed to uphold ethical standards and social responsibility. By fostering inclusivity, transparency, and responsible innovation, plant disease detection and prevention using CNNs can contribute to societal well-being, sustainable agriculture, and food security while addressing social needs and priorities.

\section{System Specification}

\begin{itemize}

\item Large-scale labeled datasets comprising diverse images of diseased and healthy plants, annotated with relevant metadata.

\item High-resolution images (e.g., 224x224 pixels or higher) for accurate feature extraction and classification by the CNN.

\end{itemize}

\begin{itemize}

\item Annotated metadata containing information about plant species, disease types, and environmental conditions for each image.\textit{}

\end{itemize}

\subsection{Hardware Specification}

\begin{itemize}

\item High-performance GPU (Graphics Processing Unit) with CUDA support for accelerated deep learning computations.

\item Minimum of 16 GB RAM for efficient model training and inference processes.

\item Multi-core CPU (Central Processing Unit) for handling data preprocessing tasks and model deployment.

\end{itemize}

\begin{itemize}

\item Adequate storage space for storing datasets, trained models, and intermediate results.

\end{itemize}

\subsection{Software Specification}

\begin{itemize}

\item Deep learning frameworks like TensorFlow or PyTorch for building and training CNN models.

\item Python programming language for scripting and implementing machine learning pipelines.

\item Image processing libraries such as OpenCV for data preprocessing, augmentation, and feature extraction.

\item Development environments like Jupyter Notebook or IDEs (Integrated Development Environments) for coding and experimentation.

\end{itemize}

\subsection{Standards and Policies}

Standards and policies governing plant disease detection and prevention using CNNs are critical for ensuring ethical, secure, and effective practices. These guidelines encompass various aspects, including data privacy and security, ethical data usage, model transparency, regulatory compliance, collaboration, security measures, and environmental and social impact. Adherence to data privacy regulations such as GDPR safeguards sensitive information collected during disease detection efforts, while ethical data policies ensure informed consent and mitigate bias.

\\

\textbf{Tensorflow model optimizer:}\\

\begin{itemize}

\item Usage: This command is used to optimize a TensorFlow model for inference on a specific hardware platform.

\item Example: It takes the input model file model.pb, specifies the input shape, output format, and other parameters to optimize the model for efficient inference.\\

\end{itemize}

\textbf{Torch-model-archiver:}\\

\begin{itemize}

\item Usage: This command is used to archive a PyTorch model for deployment.

\item Example: It archives the model plant disease model.pt into a serialized file model.pth along with a handler script plant disease handler.py for inference.

\\

\end{itemize}

\chapter{METHODOLOGY}

\linespread{1.5}

\section{General Architecture}

\begin{figure}[H]

\centering

\includegraphics[height= 9cm, width=15cm]{Sticker 1.JPG}

\caption{\textbf{General Architecture}}

\end{figure}

The general architecture of plant disease detection and prevention using Convolutional Neural Networks (CNNs) follows a systematic process to automate and enhance the identification and management of plant diseases. It begins with the acquisition of high-quality images of plants, including both diseased and healthy samples, from diverse sources. These images undergo preprocessing to standardize their format, enhance quality, and augment the dataset. Next, CNN models are trained on labeled image data, learning to automatically extract relevant features and classify plants into different disease categories. Model training involves optimization techniques such as backpropagation and gradient descent to adjust parameters for improved accuracy. Once trained, the CNN model is evaluated using separate testing datasets to assess its performance metrics like accuracy, precision, recall.

\section{Design Phase }

\subsection{Data Flow Diagram}

\begin{figure}[H]

\centering

\includegraphics[height= 10cm, width=17cm]{Sticker 2.jpg}

\caption{\textbf{Data Flow Diagram}}

\end{figure}

The data flow of plant disease detection and prevention using CNNs involves a systematic process beginning with the acquisition of high-resolution images of plants from various sources such as field surveys or digital cameras. These images undergo preprocessing to standardize their format, enhance quality, and augment the dataset. Subsequently, the preprocessed images are utilized to train the CNN model, enabling it to automatically extract relevant features and classify them into different disease categories through optimization techniques like backpropagation and gradient descent. Once trained, the CNN model is evaluated using a separate dataset to assess its performance metrics. Upon successful evaluation, the trained model is deployed for real-world inference tasks where new images of plants are input for disease prediction. Continuous feedback from users and domain experts informs model refinement, ensuring adaptability and maintaining high accuracy in disease detection. This systematic flow of data from acquisition to inference enables automated and accurate diagnosis of plant diseases, ultimately contributing to enhanced agricultural productivity and sustainability.

\subsection{Use Case Diagram}

\begin{figure}[H]

\centering

\includegraphics[height= 15cm, width=19cm]{Sticker 3.jpg}

\caption{\textbf{Use Case Diagram}}

\end{figure}

In a practical use case scenario, farmers facing potential crop disease outbreaks utilize CNN-based plant disease detection and prevention technology to diagnose and manage the issue effectively. Through this process, farmers capture images of affected crop leaves using accessible devices like smartphones or digital cameras. These images undergo preprocessing to standardize their format and enhance quality, enabling compatibility with the CNN model. The trained CNN model, having learned to recognize disease patterns and classify them into specific categories, is then deployed either locally or through cloud-based platforms accessible to farmers. Upon uploading images of crop leaves to the deployed model, farmers receive real-time diagnoses and actionable insights regarding disease presence.

\subsection{Class Diagram}

\begin{figure}[H]

\centering

\includegraphics[height= 13cm, width=16cm]{Sticker 4.jpg}

\caption{\textbf{Class Diagram}}

\end{figure}

In a Class Diagram for plant disease detection and prevention using CNNs, key classes and their relationships are depicted to illustrate the structure and interactions within the system. At the core of the diagram is the "CNN Model" class, representing the Convolutional Neural Network model responsible for disease detection. This class encapsulates methods for training the model using image datasets and making predictions on new input images. Another essential class is "Image Data Processor," responsible for pre processing raw image data before feeding it into the CNN Model. This class handles tasks such as resizing, normalization, and augmentation to prepare the images for training or inference. Additionally, the "Data Loader" class manages the loading and batching of image datasets during training. The diagram also includes classes for different disease categories, representing the output labels predicted by the CNN Model.

\subsection{Sequence Diagram}

\begin{figure}[H]

\centering

\includegraphics[height= 13cm, width=17cm]{Sticker 5.jpg}

\caption{\textbf{Sequence Diagram}}

\end{figure}

A Sequence Diagram for plant disease detection and prevention using CNNs illustrates the chronological flow of interactions between system components during the detection and prevention process. At the outset, the diagram depicts the initiation of the process, where a farmer captures images of crop leaves suspected of disease using a smartphone or digital camera. These images are then sent to the preprocessing module, where they undergo resizing, normalization, and augmentation to prepare them for analysis. Subsequently, the preprocessed images are forwarded to the CNN model for disease classification. The CNN model, having been previously trained on labeled image datasets, processes the input images and predicts the presence or absence of disease. Following this prediction, the results are returned to the farmer through a user interface, indicating the likelihood and severity of disease. Based on the model's recommendations, the farmer may implement preventive measures such as targeted pesticide application or removal of diseased plants.

\subsection{Collaboration diagram}

\begin{figure}[H]

\centering

\includegraphics[height= 13cm, width=16cm]{Sticker 6.jpg}

\caption{\textbf{Collaboration diagram}}

\end{figure}

A Collaboration Diagram for plant disease detection and prevention using CNNs visualizes the collaboration and interactions between various system components to achieve the objective of disease detection and prevention. At its core, the diagram depicts the collaboration between key entities such as the farmer, preprocessing module, CNN model, and user interface. The farmer initiates the process by capturing images of crop leaves suspected of disease, which are then passed to the preprocessing module for data preparation. The preprocessing module collaborates with the CNN model to process the input images, extracting relevant features and classifying them into disease categories. Simultaneously, the user interface facilitates communication between the farmer and the system, providing feedback on the detection results and enabling the farmer to take preventive actions based on the recommendations provided by the CNN model.

\subsection{Activity Diagram}

\begin{figure}[H]

\centering

\includegraphics[height= 15cm, width=14cm]{Sticker 7.jpg}

\caption{\textbf{Activity Diagram}}

\end{figure}

An Activity Diagram for plant disease detection and prevention using CNNs illustrates the flow of activities and decisions involved in the detection and prevention process. It begins with the "Image Capture" activity, where farmers capture images of crop leaves suspected of disease using smartphones or digital cameras. These images are then directed to the "Preprocessing" activity, where they undergo resizing, normalization, and augmentation to prepare them for analysis. The preprocessed images are subsequently fed into the "CNN Model Training" activity, where the CNN model is trained on labeled image datasets to learn disease patterns and classification. Following model training, the "Inference" activity involves inputting new images into the trained CNN model for disease prediction.

\section{Algorithm \& Pseudo Code}

\subsection{Algorithm}

EFFICIENTNETV2 ALGORITHM:

EfficientNets are currently one of the most powerful convolutional neural network (CNN) models. With the rise of Vision Transformers, which achieved even higher accuracies than EfficientNets, the question arose whether CNNs are now dying. EfficientNetV2 proves this wrong by not just improving accuracies but by also reducing training time and latency.

In this article, I have discussed in detail about these CNNs were developed, how powerful are they, and what it says about the future of CNNs in computer vision.

The EfficientNet models are designed using neural architecture search. The first neural architecture search was proposed in the paper in 2016 — ‘Neural Architecture Search with Reinforcement Learning’.

The idea is to use a controller (a network such as an RNN) and sample network architectures from a search space with probability ‘p’. This architecture is then evaluated by first training the network, and then validating it on a test set to get the accuracy ‘R’. The gradient of ‘p’ is calculated and scaled by the accuracy ‘R’. The result (reward) is fed to the controller RNN. The controller acts as the agent, the training and testing of the network act as the environment, and the result acts as the reward. This is the common Reinforcement learning (RL) loop. This loop runs multiple times till the controller finds the network architecture which gives a high reward (high test accuracy). EfficientNetV2 goes one step further than EfficientNet to increase training speed and parameter efficiency. This network is generated by using a combination of scaling (width, depth, resolution) and neural architecture search. The main goal is to optimize training speed and parameter efficiency. Also, this time the search space also included new convolutional blocks such as Fused-MBConv. In the end, the authors obtained the EfficientNetV2 architecture which is much faster than previous and newer state-of-the-art models and is much smaller (up to 6.8x times). Clearly shows that The EfficientnetV2 has 24 million parameters, while a Vision Transformer (ViT) has 86 million parameters. The V2 version also has nearly half the parameters of the original EfficientNet. While it does reduce the parameter size significantly, it maintains similar or higher accuracies than the other models on the ImageNet dataset

\begin{figure}[h]

\centering

\includegraphics[width=1.0\textwidth]{Sticker 8.jpg}

\caption{Controller RNN}

\label{fig:my\_label}

\end{figure}

Fused MBConv-layers can make training faster with only a small increase in the number of parameters, but if many of these blocks are used, it can drastically slow down training with many more added parameters. To overcome this problem, the authors passed both MBConv and Fused-MBConv in the neural architecture search, which automatically decides the best combination of these blocks for the best performance and training speed.The core component, a Convolutional Neural Network (CNN) model, is trained on labeled image datasets to learn disease patterns and features. This model comprises convolutional, pooling, activation, and fully connected layers, enabling it to extract relevant features from input images and classify them into disease categories. Post-training evaluation ensures the model's effectiveness, measured through metrics like accuracy and precision. Upon validation, the trained model is deployed for real-time inference, where new images are input for disease prediction. Continuous feedback loops enable iterative improvements to the model's performance and accuracy, ensuring its adaptability to evolving disease patterns and environmental conditions. Overall, this architecture facilitates automated and precise plant disease detection, offering a scalable and effective solution for agriculture.Scalability ensures that the system can handle large volumes of image data efficiently, accommodating the diverse needs of different farming environments and crop types. Flexibility allows for customization and adaptation of the CNN model to specific plant species, disease types, and environmental conditions, ensuring accurate and reliable performance across various scenarios. Integration with existing agricultural systems, such as farm management software or IoT (Internet of Things) devices, enables seamless data exchange and decision-making, enhancing the overall efficiency and effectiveness of disease detection and prevention efforts. Additionally, the architecture may incorporate mechanisms for model versioning, deployment automation, and monitoring to facilitate continuous improvement and maintenance of the CNN model over time. By addressing these aspects, the architecture ensures that plant disease detection and prevention using CNNs is not only accurate and reliable but also adaptable, scalable, and seamlessly integrated into agricultural workflows.

\subsection{MBConv-layers}

\begin{figure}[H]

\centering

\includegraphics[height= 13cm, width=14cm]{Sticker 10.jpg}

\caption{\textbf{MBConv-layers}}

\end{figure}

the network architecture, depthwise convolutional layers (MBConv) were slow. Depthwise convolutional layers generally have fewer parameters than regular convolutional layers, but the problem is that they cannot fully make use of modern accelerators. To overcome this problem EfficientNetV2 uses a combination of MBConv and Fused MBConv to make the training faster without increasing parameters (discussed later in the article).

\subsection{Pseudo Code}

\begin{lstlisting}

# Step 1: Data Acquisition

images = acquire\_images() # Acquire images of plants from various sources

# Step 2: Data Preprocessing

preprocessed\_images = preprocess\_images(images) # Preprocess images (e.g., resize, normalize)

# Step 3: Model Training

trained\_model = train\_cnn\_model(preprocessed\_images) # Train CNN model on preprocessed images

# Step 4: Model Evaluation (Optional)

evaluation\_metrics = evaluate\_model(trained\_model, validation\_data) # Evaluate model performance

# Step 5: Inference

new\_image = capture\_new\_image() # Capture new image of a plant leaf

preprocessed\_image = preprocess\_image(new\_image) # Preprocess new image

prediction = infer\_disease(trained\_model, preprocessed\_image) # Make prediction using trained model

# Step 6: Prevention Measures

if prediction indicates disease:

implement\_prevention\_measures() # Implement preventive measures (e.g., pesticide application)

# Step 7: Continuous Improvement

if feedback\_received:

update\_model() # Update model based on feedback

# Step 8: Repeat Steps 5-7 as necessary

\end{lstlisting}

%Description of Sequence Diagram

\section{Module Description}

\textbf{LIST OF MODULES:}

\begin{itemize}

\item Dataset Collection

\item Data Preprocessing

\item Feature Extraction

\item Model Selection and Metrics

\item Configuration of the Classification Model

\item Detection of Leaf disease

\end{itemize}

\subsection{DATASET COLLECTION:}

Data is a crucial part of any Machine Learning System. Datasets from various government websites and kaggle were used to predict the disease leaf. The dataset for the plant disease system consists of 22 plants grown across India. Dataset for plant leaf disease classification consists of images of leaves of 14 plants while excluding healthy leaves, 26 types of images that show a particular disease in a plant. For each plant disease type, there are 1800 images.

\subsection{DATA PREPROCESSING:}

Labels of each plant images are then also mapped to a unique the pre-processing step for any machine learning model is of great importance and ideally shapes the performance and results of the models chosen. In this report, the following were the steps that were carried out in order to make sure that the models produced optimal results. After reading and resizing the images, we then convert the images into an array form using np .array ()

\begin{enumerate}

\item The e value using LabelBinarizer()

\item Finally, the plant village dataset is split into two different sets, namely, train and test set with a 75:25 ratio respectively.

\end{enumerate}

\subsection{FEATURE EXTRACTION:}

Feature extraction is a process in machine learning where relevant information is extracted from raw data to create a more meaningful and simplified representation of the data that can be easily processed by a learning algorithm. For the plant leaf disease image dataset, feature extraction involves extracting features such as color, texture, and shape of the leaves. This can be done using image processing techniques such as edge detection, color histograms, and convolutional neural networks (CNNs) to identify and extract these features. These features are then fed into a machine learning algorithm for classification or prediction of the type of plant disease present.The input image is passed through a series of convolutional layers in the CNN. Each convolutional layer consists of filters that convolve across the input image, detecting patterns and features such as edges, textures, and shapes at different scales.

\subsection{MODEL SELECTION AND METRICS:}

It involves selecting the appropriate algorithm and architecture to use in the model, as well as tuning the hyperparameters to achieve the best performance. For plant leaf disease detection using image dataset, deep learning architectures such as convolutional neural networks (CNNs) are commonly used. CNNs are particularly suited to image data as they can automatically extract relevant features from images without the need for manual feature engineering. Transfer learning, where a pre-trained CNN model is fine-tuned for the specific task of plant leaf disease detection, can also be used to improve performance with limited data.

\subsection{CONFIGURATION OF THE CLASSIFICATION MODEL:}

The architecture of CNN used for plant disease detection in this project was as follows, the first block contains a Convolutional layer with 32 filters of size 3 x 3 and the activation function used was the ReLU activation function. We then follow the operation by performing batch normalization, and choosing the Max Pooling layer with a pool size of and adding a dropout layer with 25 percent drop- out. Batch normalization was performed in order to speed up the convergence of the neural network, it is generally applied after each individual layer so that the out- put of the previous layer can be normalized allowing for each individual layer present in the network to perform learning independent. Dropout layer is a technique used to prevent the model from overfitting by randomly switching off some sections of the neurons. When some sections of the neurons are switched off the incoming as well as the outgoing connections from the neurons are also switched off and this results in the betterment of the model in learning and allows for the model to not generalize to the test dataset. We used the pre trained model for leaf disease prediction.

\subsection{DETECTION OF LEAF DISEASE:}

Pre-trained CNN models can be used to detect plant leaf diseases by extracting relevant features from the images and classifying the type of disease present. This approach can improve the accuracy of the predictions and reduce the need for manual feature engineering. However, it requires a large amount of data to fine-tune the pre-trained model for the specific task. Use the pre trained CNN model to make predictions on new plant leaf images to detect the type of disease present.

\section{Steps to execute/run/implement the project}

\subsection{Step 1}

\begin{itemize}

\item INSTALLING PACKAGES

\item Requirements for Installing Packages

\item Ensure you can run Python from the command line

\item Traceback most recent call last

\item Ensure you can run pip from the command line

\item Ensure pip, setuptools, and wheel are up to date

\item Optionally, create a virtual environment

\item

\end{itemize}

\subsection{Step2}

\begin{itemize}

\item Creating Virtual Environments

\item Use pip for Installing

\item Installing from PyPI

\item Source Distributions vs Wheels

\item Upgrading packages

\item Requirements files

\item Installing from VCS

\end{itemize}

\subsection{Step3}

\begin{itemize}

\item Installing from other Indexes

\item Installing from a local src tree

\item Installing from local archives

\item Installing from other sources

\item Installing Prereleases

\item Installing Setuptools “Extras”

\end{itemize}

\chapter{IMPLEMENTATION AND TESTING}

\linespread{1.5}

\section{Input and Output}

\subsection{Input Design}

\begin{figure}[H]

\centering

\includegraphics[height=6cm, width=14cm]{Sticker 11.jpg}

\caption{\textbf{Input Design}}

\end{figure}

\subsection{Output Design}

\begin{figure}[H]

\centering

\includegraphics[height=6cm, width=14cm]{Sticker 12.jpg}

\caption{\textbf{Output Design}}

\end{figure}

\section{Testing}

\section{Types of Testing}

\subsection{Unit testing}

\subsubsection{Input}

\begin{lstlisting}

import unittest

import numpy as np

from your\_module import DiseaseDetectorCNN # Import your CNN model class or function

class TestPlantDiseaseDetection(unittest.TestCase):

def setUp(self):

# Initialize your CNN model

self.model = DiseaseDetectorCNN()

def tearDown(self):

# Clean up resources if needed

pass

def test\_disease\_prediction(self):

# Test case for disease prediction

# Generate a sample input image (you may use actual plant images)

# Here, we create a dummy image with shape (height, width, channels)

input\_image = np.random.rand(224, 224, 3) # Assuming input shape of your CNN model

# Perform inference

predicted\_disease = self.model.predict(input\_image)

# Ensure that the prediction is one of the expected disease classes

expected\_classes = ["disease1", "disease2", "disease3"] # Define your expected disease classes

self.assertIn(predicted\_disease, expected\_classes, "Prediction should be one of the expected disease classes")

\end{lstlisting}

\subsubsection{Test result}

\begin{figure}[H]

\centering

\includegraphics[height=5cm, width=14cm]{Unit test.png}

\caption{\textbf{Unit testing test result}}

\end{figure}

\subsection{Integration testing}

\subsubsection{Input}

\begin{lstlisting}

import unittest

import numpy as np

from your\_module import DiseaseDetectorSystem # Import your integrated system class or function

class TestIntegratedPlantDiseaseDetection(unittest.TestCase):

def setUp(self):

# Initialize your integrated system

self.system = DiseaseDetectorSystem()

def tearDown(self):

# Clean up resources if needed

pass

def test\_disease\_detection(self):

# Test case for disease detection

# Generate a sample input image (you may use actual plant images)

# Here, we create a dummy image with shape (height, width, channels)

input\_image = np.random.rand(224, 224, 3) # Assuming input shape of your CNN model

# Perform disease detection using the integrated system

detection\_result = self.system.detect\_disease(input\_image)

# Ensure that the detection result is valid and includes predicted disease

self.assertTrue("disease" in detection\_result, "Detection result should include predicted disease")

# Ensure that the confidence score is within a reasonable range

confidence\_score = detection\_result["confidence"]

self.assertTrue(0 <= confidence\_score <= 1, "Confidence score should be between 0 and 1")

# Ensure that additional information is provided, if available

if "additional\_info" in detection\_result:

additional\_info = detection\_result["additional\_info"]

self.assertIsInstance(additional\_info, dict, "Additional info should be a dictionary")

# Add more assertions for specific additional information if needed

\end{lstlisting}

\subsubsection{Test result}

\begin{figure}[H]

\centering

\includegraphics[height=4cm, width=14cm]{images.jpg}

\caption{\textbf{Integration testing test result}}

\end{figure}

\subsection{System testing}

\subsubsection{Input}

\begin{lstlisting}

import unittest

import numpy as np

from your\_module import CNNModel # Import your CNN model class or function

class TestWhiteBoxPlantDiseaseDetection(unittest.TestCase):

def setUp(self):

self.model = CNNModel()

def tearDown(self):

pass

def test\_model\_architecture(self):

self.assertEqual(len(self.model.layers), 10)

expected\_layer\_config = [...] # Define expected layer configurations

for i, layer in enumerate(self.model.layers):

self.assertDictEqual(layer.get\_config(), expected\_layer\_config[i])

def test\_model\_training(self):

X\_train = np.random.rand(100, 224, 224, 3)

y\_train = np.random.randint(0, 3, size=(100,))

history = self.model.train(X\_train, y\_train, epochs=5, batch\_size=32)

self.assertLess(history.history['loss'][-1], history.history['loss'][0])

self.assertGreater(history.history['accuracy'][-1], history.history['accuracy'][0])

def test\_model\_inference(self):

input\_image = np.random.rand(224, 224, 3)

predicted\_class = self.model.predict(input\_image)

self.assertTrue(0 <= predicted\_class <= 2)

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main()

import unittest

import numpy as np

from your\_module import DiseaseDetectorSystem # Import your integrated system class or function

class TestBlackBoxPlantDiseaseDetection(unittest.TestCase):

def setUp(self):

self.system = DiseaseDetectorSystem()

def tearDown(self):

pass

def test\_detection\_with\_real\_images(self):

input\_images = [load\_image("path\_to\_image1"), load\_image("path\_to\_image2"), ...]

for input\_image in input\_images:

detection\_result = self.system.detect\_disease(input\_image)

self.assertTrue("disease" in detection\_result)

self.assertTrue(0 <= detection\_result["confidence"] <= 1)

if "additional\_info" in detection\_result:

self.assertIsInstance(detection\_result["additional\_info"], dict)

def test\_detection\_with\_augmented\_images(self):

augmented\_images = [generate\_augmented\_image() for \_ in range(10)]

for augmented\_image in augmented\_images:

detection\_result = self.system.detect\_disease(augmented\_image)

self.assertTrue("disease" in detection\_result)

self.assertTrue(0 <= detection\_result["confidence"] <= 1)

if "additional\_info" in detection\_result:

self.assertIsInstance(detection\_result["additional\_info"], dict)

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main()

\end{lstlisting}

\subsection{Test Result}

\begin{figure}[H]

\centering

\includegraphics[height= 5cm, width=14cm]{Sticker 9.jpg}

\caption{\textbf{System testing test result}}

\end{figure}

\chapter{RESULTS AND DISCUSSIONS}

\linespread{1.5}

\section{Efficiency of the Proposed System}

\\

The proposed system of plant disease detection and prevention using CNNs offers significant advantages in terms of efficiency, accuracy, and scalability. By leveraging the power of deep learning techniques, the system can automatically analyze large volumes of plant images with remarkable speed and precision, enabling early detection and timely intervention to mitigate crop losses. The CNN architecture allows for the extraction of intricate patterns and features from the input images, facilitating accurate classification of diseased and healthy plants across various crop species and disease types. Furthermore, the efficiency of the system extends beyond its detection capabilities. The automated nature of CNN-based detection reduces the need for manual inspection, saving time and labor costs for farmers. Additionally, the system's scalability enables it to adapt to different farming environments and scales of operation, from smallholder farms to large agricultural enterprises. Moreover, the proposed system promotes sustainable agricultural practices by facilitating targeted interventions, such as precision pesticide application, thereby reducing chemical usage and minimizing environmental impact. By providing real-time insights into disease prevalence and severity, the system empowers farmers to make informed decisions and implement proactive measures to protect their crops. Overall, the efficiency of the proposed system lies in its ability to deliver accurate, timely, and actionable information for effective disease management, ultimately enhancing crop health, productivity, and sustainability. the efficiency of the proposed system is enhanced by its ability to continuously learn and adapt to new disease patterns and environmental conditions. Through ongoing training and feedback loops, the CNN model can improve its accuracy and performance over time, ensuring robustness in disease detection across diverse agricultural settings. This adaptability is crucial for addressing emerging disease threats and evolving agricultural practices, allowing the system to remain effective in dynamic and changing environments.

\section{Comparison of Existing and Proposed System}

\\

\textbf{Existing system:}\\ The existing system of plant disease detection and prevention using CNNs represents a significant advancement in agricultural technology, offering automated and accurate solutions for identifying and managing crop diseases. An example of such a system is the PlantVillage project developed by researchers at Penn State University. PlantVillage utilizes CNNs to analyze images of plant leaves uploaded by farmers through a mobile application or web interface. The system then employs deep learning algorithms to classify the images into different disease categories based on visual symptoms. In the existing system, farmers can easily access the PlantVillage platform and upload images of their crops for disease diagnosis. The existing system of plant disease detection and prevention using CNNs fosters collaboration and knowledge sharing among farmers, extension workers, and researchers. By leveraging the collective expertise and data contributed by users worldwide, the system continuously improves its accuracy and performance, ultimately benefiting the global agricultural community in combating plant diseases and ensuring food security.

\\

\textbf{Proposed system:}\\ The proposed system of plant disease detection and prevention using CNNs aims to revolutionize agricultural practices by providing an automated, accurate, and scalable solution for detecting and managing crop diseases. For instance, consider a scenario where a group of farmers in a rural community faces a sudden outbreak of a plant disease affecting their tomato crops. With the proposed system, farmers can utilize a mobile application equipped with CNN-based image recognition technology to capture images of the affected tomato plants. These images are then uploaded to a centralized server where a trained CNN model processes them for disease diagnosis. The CNN model leverages its deep learning capabilities to analyze the visual symptoms exhibited by the plants and classify them into different disease categories with high accuracy. Upon diagnosis, the system provides real-time recommendations for disease management strategies, such as targeted pesticide application, crop rotation, or removal of diseased plants. The proposed system empowers farmers with timely, actionable insights to protect their crops and ensure food security in their communities.

\section{Sample Code}

\begin{lstlisting}

import tensorflow as tf

import tensorflow\_hub as hub

from tensorflow.keras import layers, models, optimizers

from tensorflow.keras.preprocessing.image import ImageDataGenerator

MODULE\_HANDLE = 'https://tfhub.dev/google/imagenet/efficientnet\_v2\_imagenet1k\_b0/feature\_vector/2'

IMAGE\_SIZE = (224, 224)

NUM\_CLASSES = 4

def create\_model():

feature\_extractor = hub.KerasLayer(MODULE\_HANDLE,

input\_shape=IMAGE\_SIZE + (3,), # Input shape of images (height, width, channels)

trainable=False) # Freeze the weights of the pre-trained model

model = models.Sequential([

feature\_extractor,

layers.Dense(NUM\_CLASSES, activation='softmax') # Output layer with softmax activation for classification

])

return model

# Create an instance of the model

model = create\_model()

train\_dir = 'old diease/dataset/train'

test\_dir = 'old diease/dataset/test'

# Compile the model

model.compile(optimizer='adam',

loss='categorical\_crossentropy',

metrics=['accuracy'])

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,

fill\_mode='nearest')

history = model.fit(

train\_generator,

steps\_per\_epoch=train\_generator.samples/train\_generator.batch\_size,

epochs=50, # You can adjust the number of epochs

validation\_data=validation\_generator,

validation\_steps=validation\_generator.samples/validation\_generator.batch\_size)

# Evaluate the model

test\_loss, test\_accuracy = model.evaluate(validation\_generator)

print('Test Accuracy:', test\_accuracy)

model.save("newmodel.h5")

\end{lstlisting}

\subsubsection{Output}

\begin{figure}[H]

\centering

\includegraphics[height= 15cm, width=17cm]{ Output 1.png}

\caption{\textbf{Output 1}}

\end{figure}

\begin{figure}[H]

\centering

\includegraphics[height= 18cm, width=18cm]{Output 2.png}

\caption{\textbf{Output 2}}

\end{figure}

\chapter{CONCLUSION AND FUTURE ENHANCEMENTS}

\linespread{1.5}

\section{Conclusion}

In conclusion, plant disease detection and prevention using Convolutional Neural Networks (CNNs) represents a groundbreaking advancement in agricultural technology, offering a comprehensive and effective solution to combat crop diseases. Through the integration of deep learning techniques, image processing algorithms, and agricultural expertise, CNN-based systems have the potential to revolutionize the way farmers monitor, diagnose, and manage plant health in agricultural settings. The implementation of CNNs in plant disease detection and prevention has demonstrated remarkable accuracy and efficiency in identifying disease symptoms from images of plant leaves. By leveraging the power of deep learning, CNN models can extract intricate patterns and features from images, enabling precise classification of diseased and healthy plants across various crop species and disease types. This capability enables early detection of diseases, allowing farmers to take timely and targeted actions to mitigate crop losses and preserve yield potential. Moreover, CNN-based systems offer scalability and adaptability, making them suitable for deployment in diverse agricultural environments worldwide. From smallholder farms to large-scale agricultural enterprises, CNNs can be tailored to meet the specific needs and challenges of different farming operations. Additionally, the integration of CNN-based technologies with existing agricultural systems and digital platforms enables seamless data exchange and workflow integration, enhancing the overall efficiency and effectiveness of disease management efforts. Furthermore, the widespread adoption of CNN-based plant disease detection systems holds promise for promoting sustainable agricultural practices and reducing environmental impact.

\section{Future Enhancements}

In the future, advancements in plant disease detection and prevention using Convolutional Neural Networks (CNNs) will continue to evolve, offering new opportunities for enhanced accuracy, efficiency, and scalability. One potential area for improvement is the integration of multimodal data sources, such as spectral imaging and hyperspectral imaging, with CNN-based systems. By combining visual images with additional data modalities, such as infrared or near-infrared spectra, researchers can extract more comprehensive information about plant health and disease status, improving the overall accuracy and robustness of disease detection models. Furthermore, advancements in model architecture and training techniques will enable the development of more sophisticated CNN models capable of handling complex disease interactions and environmental factors. Techniques such as transfer learning, ensemble learning, and reinforcement learning hold promise for enhancing the generalization and adaptability of CNN-based systems across diverse agricultural settings and crop types. Additionally, the integration of real-time monitoring and IoT (Internet of Things) devices with CNN-based systems will enable continuous surveillance of plant health parameters, facilitating early detection of disease outbreaks and proactive intervention strategies. By leveraging data from sensors, drones, and other IoT devices, CNN models can provide farmers with timely insights into disease prevalence, environmental conditions, and optimal management practices, ultimately improving crop yields and agricultural sustainability. Overall, future enhancements in CNN-based plant disease detection and prevention will empower farmers with advanced tools and technologies to address emerging challenges and secure food production for future generations. The ntegrating multimodal data sources and advancing model architecture, future enhancements in plant disease detection and prevention using CNNs will also focus on improving accessibility and usability for farmers in diverse agricultural settings. This includes the development of user-friendly mobile applications and web interfaces that streamline the process of capturing and uploading images of diseased plants for analysis. Simplified user interfaces and intuitive design will ensure that even farmers with limited technical expertise can easily utilize CNN-based systems to diagnose and manage plant diseases.

\chapter{PLAGIARISM REPORT}

ATTACH ONLY SUMMARY PAGE OF PLAGIARISM REPORT

\chapter{SOURCE CODE \& POSTER PRESENTATION}

\section{Source Code}

\begin{lstlisting}

# Data Preparation

import numpy as np

import cv2

import os

# Load images and labels

def load\_data(data\_dir):

images = []

labels = []

for label in os.listdir(data\_dir):

for img\_file in os.listdir(os.path.join(data\_dir, label)):

img\_path = os.path.join(data\_dir, label, img\_file)

img = cv2.imread(img\_path)

img = cv2.resize(img, (224, 224)) # Resize image to fixed size

images.append(img)

labels.append(label)

return np.array(images), np.array(labels)

# Data Preprocessing (Normalization, Augmentation)

from keras.preprocessing.image import ImageDataGenerator

datagen = ImageDataGenerator(

rescale=1./255,

rotation\_range=40,

width\_shift\_range=0.2,

height\_shift\_range=0.2,

shear\_range=0.2,

zoom\_range=0.2,

horizontal\_flip=True,

fill\_mode='nearest')

# Model Architecture

from keras.models import Sequential

from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense

model = Sequential([

Conv2D(32, (3, 3), activation='relu', input\_shape=(224, 224, 3)),

MaxPooling2D((2, 2)),

Conv2D(64, (3, 3), activation='relu'),

MaxPooling2D((2, 2)),

Conv2D(128, (3, 3), activation='relu'),

MaxPooling2D((2, 2)),

Flatten(),

Dense(512, activation='relu'),

Dense(1, activation='sigmoid')

])

model.compile(optimizer='adam',

loss='binary\_crossentropy',

metrics=['accuracy'])

train\_images, train\_labels = load\_data('train\_data/')

validation\_images, validation\_labels = load\_data('validation\_data/')

model.fit(datagen.flow(train\_images, train\_labels, batch\_size=32),

epochs=10,

validation\_data=(validation\_images, validation\_labels))

loss, accuracy = model.evaluate(validation\_images, validation\_labels)

print("Validation Accuracy:", accuracy)

# Model Deployment (Example using Flask for web application)

from flask import Flask, request, jsonify, render\_template

import base64

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

return render\_template('index.html')

@app.route('/predict', methods=['POST'])

def predict():

img\_base64 = request.form['image']

img\_bytes = base64.b64decode(img\_base64.split(',')[1])

img\_np = np.frombuffer(img\_bytes, dtype=np.uint8)

img = cv2.imdecode(img\_np, cv2.IMREAD\_COLOR)

img = cv2.resize(img, (224, 224))

img = np.expand\_dims(img, axis=0)

prediction = model.predict(img)

return jsonify({'prediction': prediction[0][0]})

\end{lstlisting}

\section{Poster Presentation}

Should be in New page after the source code

\addcontentsline{toc}{chapter}{References}

\renewcommand\bibname{References}

\begin{thebibliography}{10}

\begin{enumerate}

\item AI Challenger.(2018). AI Challenger 2018 Datasets. Accessed: Apr. 1, 2022. [Online].

\item C.K.Sunil,C.D.Jaidhar,and N. Patil, ‘‘Cardamom plant disease detec[1]tion approach using EfficientNetV2,’’ IEEE Access, vol. 10, pp. 789–804, 2022

\item D. Wang, J. Wang, Z. Ren, and W. Li, ‘‘DHBP: A dual-stream hierarchical bilinear pooling model for plant disease multi-task classification,’’ Comput. Electron. Agricult., vol.195,Apr.2022,Art.no.106788.

\item D.O.Oyewola,E.G.Dada,S. Misra, and R. Damaševičius,‘‘Detecting cassava mosaic disease using a deep residual convolutional neural network with distinct block processing, “PeerJComput. Sci .,vol . 7, p.e352 ,Mar.2021

\item E. Elfatimi, R. Eryigit, and L. Elfatimi, ‘‘Beans leaf diseases classification using MobileNet models,’’ IEEE Access, vol. 10, pp. 9471–9482, 2022.  PeerComput.Sci.,vol.7,p.e352,Mar.2021

\item L. Li, S. Zhang, and B. Wang, ‘‘Plant disease detection and classification by deep learning are view,’’IEEEAccess,vol.9,pp.56683–56698,2021.

\item L.C.Ngugi,M.Abdelwahab,andM.Abo-Zahhad,‘‘Tomatoleaf segmentation algorithms for mobile phone applications using deep learning,’’ Comput. Electron. Agricult., vol. 178, Nov. 2020, Art. no. 105788

\item M. Ahmad, M. Abdullah, H. Moon, and D. Han, ‘‘Plant disease detection in imbalanced datasets using efficient convolutional neural networks with stepwise  transfer learning,’’IEEEAccess,vol.9,pp.140565 140580,2021.

\item Makerere AI Lab and AIR Lab National Crops Resources Research Institute (NaCRRI) .(2020).iBean . Accessed :Mar.31,2022

\item M.Adi, A. K. Singh, H. Reddy A. Y. Kumar, V. R. Challa, P. Rana, and U. Mittal, ‘‘An overview on plant disease detection algorithm using deep learning,’’ in Proc.2nd Int. Conf.Intell.Eng.Manage .(ICIEM) ,Apr.2021,pp.305–309

\end{enumerate}

\\

\end{thebibliography}

\end{document}