Project Report

On

Real-time Plant Disease Detection and Pesticide Application System Using IoT and CNN

Submitted as partial fulfilment for the award of

BACHELOR OF TECHNOLOGY DEGREE

Session 2023-24

in

Electrical and Electronics Engineering

By

Aakash Kumar Singh (2100320219001)
Ashmit Raghav (2000320210019)
Bhanu Pratap Singh (2000320210021)
Anivesh Kumar Singh (2100320219003)

Under the guidance of

Mr. Vivek Kumar Verma, Sr. Assistant Professor

ABES ENGINEERING COLLEGE, GHAZIABAD
(Affiliated to DR. A.P.J. ABDUL KALAM TECHNICAL UNIVERSITY, U.P., LUCKNOW)

June 2024

Project Report

on

Real-time Plant Disease Detection and Pesticide Application System Using IoT and CNN

Submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY

DEGREE

Session 2023-24
in

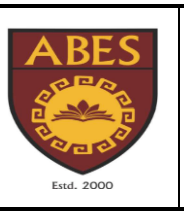
Electrical and Electronics Engineering

By
Aakash Kumar Singh (2100320219001)
Ashmit Raghav (2000320210019)
Bhanu Pratap Singh (2000320210021)
Anivesh Kumar Singh (2100320219003)

Under the guidance of

Mr. Vivek Kumar Verma
Sr. Assistant Professor

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING
ABES ENGINEERING COLLEGE, GHAZIABAD



AFFILIATED TO
DR. A.P.J. ABDUL KALAM TECHNICAL UNIVERSITY, U.P., LUCKNOW
(Formerly UPTU)

Student's Declaration

We hereby declare that the work being presented in this report entitled **"Real-time Plant Disease Detection and Pesticide Application System Using IoT and CNN"** is an authentic record of our own work carried out under the supervision of **Mr. Vivek Kumar Verma, Associate Professor, Department of Electrical and Electronics Engineering.**

The matter embodied in this report has not been submitted by us for the award of any other degree.

Date:

Signature of student
(Name: Aakash Kumar Singh)
(Roll No. 2100320219001)
Department: EN

Signature of student
(Name: Ashmit Raghav)
(Roll No. 2000320210019)
Department: EN

Signature of student
(Name: Bhanu Pratap Singh)
(Roll No. 2000320210021)
Department: EN

Signature of student
(Name: Anivesh Kumar Singh)
(Roll No. 2100320219003)
Department: EN

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Signature of HOD
Name: Prof. Pragati Shrivastava Deb
HOD EN Department
EN Department

Signature of Supervisor
Name: Mr. Vivek Kumar Verma
Sr. Assistant Professor
EN Department

Date: .....

Acknowledgement

We extend our sincere gratitude to Mr. Vivek Kumar Verma for their guidance and mentorship throughout this project. Special thanks to ABES Engineering College Ghaziabad for providing resources that facilitated our research endeavors. We also appreciate the unwavering support of our families, whose encouragement motivated us to overcome challenges and persevere towards our goals.

We would like to express our appreciation to our fellow group members for their dedication and teamwork. Lastly, we thank all who contributed directly or indirectly to this project. Your assistance was invaluable. Together, we have achieved a significant milestone, and we look forward to future collaborations and discoveries.

Signature of student
(Name: Aakash Kumar Singh)
(Roll No. 2100320219001)

Signature of student
(Name: Ashmit Raghav)
(Roll No. 2000320210019)

Signature of student
(Name: Bhanu Pratap Singh)
(Roll No. 2000320210021)

Signature of student
(Name: Anivesh Kumar Singh)
(Roll No. 2100320219003)

Table of Contents

Page No.

STUDENT'S DECLARATION..... i

ACKNOWLEDGEMENT..... ii

LIST OF FIGURES..... v

LIST OF TABLES..... vi

LIST OF ABBREVIATIONS..... vii

ABSTRACT..... viii

1. INTRODUCTION..... 1

1.1 BACKGROUND OF THE STUDY..... 1

1.2 CURRENT CHALLANAGES..... 2

1.3 LITERATURE SURVEY..... 4

2. PROJECT STATEMENT 5

3. PROJECT OBJECTIVE..... 6

4. CONVOLUTIONAL NEURAL NETWORKS 7

4.1 ARCHITECTURE OF CNN..... 7

4.2 2-D CNN..... 9

5. PROJECT METHODOLOGY..... 12

5.1 TRAINING OF CNN MODEL..... 12

5.2 IMPLEMENTATION ON RASPBERRY PI..... 17

6. DETAILS OF PROJECT WORK..... 21

6.1 HARDWARE COMPONENTS..... 21

6.2 SOFTWARE USED.....33

6.3 CONNECTION DIAGRAM.....36

6.4 HARDWARE SETUP……37

7. RESULTS AND DISCUSSIONS 38

8. CONCLUSION AND FUTURE SCOPE 42

9. REFERENCES..... 43

10. RESEARCH PAPER..... 44

11. APPENDIX-I (CODING) 50

List of Figures

Figure No.	Figure Name	Page No.
FIG 4.2	ARCHITECTURE OF CNN	11
FIG 5	FLOWCHART OF THE METHODOLOGY	20
FIG 6.1.1a	RASPBERRY PI 2 MODEL B	22
FIG 6.1.1b	RASPBERRY PI 2 PIN DESCRIPTION	23
FIG 6.1.2	12V DC MOTOR PUMP	24
FIG 6.1.3	16X2 LCD WITH I2C	26
FIG 6.1.4	5V SINGLE CHANNEL RELAY	27
FIG 6.1.5	ULTRASONIC SENSOR	28
FIG 6.1.6	MOTOR DRIVER	29
FIG 6.1.8	2V DC MOTOR	31
FIG 6.1.9	12V BATTERY PACK	32
FIG 6.2.1	JUPYTER NOTEBOOK ENVIRONMENT	33
FIG 6.2.2	GEANY ENVIRONMENT	35
FIG 6.3.1	CONNECTION DIAGRAM OF THE MODEL	36
FIG 6.3.2	CONNECTION DIAGRAM OF THE WHEELS	36
FIG 6.4	FINAL HARDWARE SETUP	37
FIG 7.1.1	VISUALIZATION OF ACCURACY RESULT	38
FIG 7.1.2a	CALCULATION OF PRECISION AND RECALL	40
FIG 7.1.2b	PLANT DISEASE PREDICTION CONFUSION MATRIX	40
FIG 7.2a	HEALTHY PLANT DETECTED	41
FIG 7.2b	DISEASED PLANT DETECTED	41

List of Tables

Table No.	Title	Page No.
1.3	Difference between the conventional method and automatic spraying method	4

List of Abbreviations

CNN	CONVOLUTIONAL NEURAL NETWORK
RPN	REGION PROPOSAL NETWORKS
GPIO	GENERAL PURPOSE INPUT / OUTPUT
LCD	LIQUID CRYSTAL DISPLAY
I2C	INTER INTEGRATED CIRCUIT
VCC	VOLTAGE COMMON COLLECTOR
GND	GROUND

Abstract

The agricultural sector faces a critical challenge: the inefficiency and health hazards associated with manual pesticide application. This outdated method leads to farmer fatigue, potential health risks, and uneven application that jeopardizes crop health and yield. Inadequate or excessive pesticide use also raises environmental concerns. To address these issues, this report proposes a groundbreaking solution leveraging advanced technologies to revolutionize crop management.

For accurate plant disease identification and automated pesticide administration, this research focuses on combining a 2-Dimensional Convolutional Neural Network (CNN) model with a Raspberry Pi platform. The CNN model uses deep learning techniques to correctly identify diseases using image recognition after being trained on large datasets of plant pictures. With its powerful processor, the Raspberry Pi makes it possible to analyze photos in real time and manage the application of pesticides in response to suspected plant illnesses.

This creative strategy seeks to advance sustainable crop management methods and simplify farming operations in a number of ways. First off, automation greatly lessens the need for human labor, which relieves farmers of the physical strain and possible health hazards. Second, the technology optimizes pesticide consumption, reducing waste and environmental contamination by exclusively targeting diseased plants. Thirdly, accurate treatment rates based on disease severity are made possible by real-time analysis, which maximizes pest control effectiveness.

Chapter-1

Introduction

The substantial financial losses linked to plant diseases emphasize the critical necessity for effective pest and disease management techniques in modern agricultural operations in order to guarantee crop health and output. Traditional techniques, which sometimes include physical effort, can be labor-intensive and incorrect, producing less-than-ideal results. Furthermore, using pesticides carelessly endangers both human health and the sustainability of the environment.

The Real-time Plant Disease Detection and Pesticide Application System Using IoT and CNN project, which is based on the Raspberry Pi platform, combines cutting-edge technology to provide a unique answer to these problems. This project attempts to transform pesticide application methods by fusing a Raspberry Pi's processing capacity with a 2-D Convolutional Neural Network (CNN). The Smart Pesticide Sprayer provides a proactive and effective method of crop management by combining the power of real-time processing capabilities of the Raspberry Pi with the precision of deep learning algorithms for plant disease identification. This clever technology optimizes pesticide usage and minimizes manual work, which not only expedites the administration of pesticides but also supports sustainable agriculture practices.

1.1 Background of the Study:

The increased demand for food due to the growing global population is placing immense pressure on our agriculture industry. However, plant diseases continue to pose a threat to this important industry. These stealthy saboteurs

stealthily enter fields, causing severe losses in productivity, much like cunning thieves. Most of these attacks are directed on farmers, who suffer greatly from lower yields and consequent financial difficulty.

Early diagnosis is the key to reducing these losses. Early infection detection allows farmers to change growing conditions or apply targeted fungicides with rapid action. Unfortunately, a significant part of conventional methods for identifying plant diseases involves eye inspection by agricultural specialists. These experts are very knowledgeable, but their methods have several shortcomings.

These checks take a long time, to start with. Many farmers lack the luxury of time, as it can take days to thoroughly survey big fields, plant by plant, looking for minute indicators of illness. Second, opinions are subjective by definition. A novice's sight might miss a noticeable degradation on a single leaf, or fatigue could lead to errors. Owing to its subjectivity, there is a potential for error, which could cause therapy to be delayed or administered incorrectly. A false positive might have catastrophic consequences, raising the possibility of disease spread and jeopardizing crop health.

Recognizing these limitations, scientists have been aggressively looking into innovative solutions. This project is a result of the need for a more trustworthy and efficient plant disease detection system. It offers a state-of-the-art tactic that leverages the revolutionary potential of artificial intelligence (AI).

1.2 Current Challenges in Pesticide Application:

Achieving sustainable practices in agriculture is severely hampered by the way pesticides are currently applied. The most common method, manual spraying, is very labor-intensive. Farmers use handheld equipment for extended periods

of time, which can be taxing on their bodies and pose health hazards, especially if they don't have the appropriate protective gear. In addition, hand spraying frequently leads to an inconsistent pesticide distribution throughout the crop canopy. Certain regions receive an excessive quantity, which could waste resources and put workers at needless risk. More significantly, an incomplete application makes the entire treatment useless by allowing pests to flourish in untreated regions. Because of this inequity, insect populations become increasingly resistant to pesticides, which feeds a vicious cycle whereby greater pest populations require even more frequent and intense treatment.

In addition to the human cost, conventional techniques have major environmental drawbacks. Fields release pesticide runoff that contaminates bodies of water, endangering aquatic life and lowering drinking water quality. These substances can also build up in the soil, where they can damage biodiversity and soil health over time by lingering in the ecosystem. The delicate equilibrium of natural ecosystems is upset by the over use of pesticides. The natural food chain is upset and long-term pest control efforts are further jeopardized when beneficial creatures like pollinators and naturally occurring insect predators perish. This lowers crop yields and makes agricultural systems less resilient overall.

The drawbacks of conventional techniques emphasize how urgently plant disease control has to adopt a more effective, focused, and ecologically responsible strategy. This is where technological innovations like automated systems and machine learning come into play, offering promising answers for a more sustainable agriculture in the future.

1.3 Literature Survey:

Our investigation revealed that a great deal of research has been done in this area because agriculture plays a significant role in the economy. Therefore, pesticides have a function in protecting crops against illness and undesired insects. However, the typical method—using a backpack sprayer—promotes uneven pesticide spraying on crops, which has been harmful to crops [1].

This project is a fresh idea since previous methods did not combine the use of CNN with Raspberry Pi [] for this type of hardware project. This will assist the farmer in farming profitably because it is a low power operating strategy that may be applied in situations where the condition is complex.

The previous models use a camera to capture the plant image and the manpower is required to take the decision whether plant is infected or not by seeing the images and then they decide to spray. But this model automates this process, where the camera takes the picture and the deep learning model identify whether the plant is diseased or not. Table 1.3 shows the cost difference between the conventional method and automatic spraying method:

Description	Conventional Method (10 Acre)	Automatic Spraying Machine (10 Acre)
Pesticides	5 Litre	4 Litre
Pesticide Cost	5*2000 = Rs.10000	4*2000 = Rs.8000
Labor Charge	Rs. 300 / Day	Rs. 300 / Day
Nos. of Labors	3	1
Nos. of Working Days	8	2
No. of Cycle	5	5
Total Charges of Labor	300*3*8*5 = Rs36000	300*1*2*5 = Rs 3000
Machine cost	3000*3 = Rs.9000	Rs.35000
Working Charges of Battery	-	Rs.120 / Day for 5 no. of Cycle = Rs 600
Total Cost	10000+36000+9000= Rs.55000	8000+3000+35000+600= Rs.46000

Table 1.3 Difference between the conventional method and automatic spraying method

Chapter-2

Problem Statement

Traditional agricultural pesticide spraying techniques are fraught with environmental hazards and inefficiencies. The labor-intensive procedure of manual spraying frequently leads to unequal distribution. Research indicates that as much as 70% of pesticides used may fail to reach their intended target, resulting in resource waste and inadequate control of pests. Furthermore, the careless use of pesticides contaminates soil and water, endangering biodiversity and possibly posing a risk to human health. Additionally, the concerning increase in pesticide-resistant species of pests calls for ongoing adaptation and the investigation of substitute approaches.

The Smart Pesticide Sprayer project offers an innovative solution that uses automation and deep learning to address these issues. This project combines an automated spraying system based on a Raspberry Pi with a 2-D Convolutional Neural Network (CNN) model for precise plant disease diagnosis. The Smart Pesticide Sprayer has the potential to drastically cut pesticide usage (estimated at 50%) and lessen environmental effect by exclusively targeting unhealthy plants. To further improve system efficiency, the Raspberry Pi's computing capacity enables real-time decision-making and on-device analysis.

In the future, features like real-time data collecting and analysis will allow this project to grow even further. By using this data, targeted spray maps could be made, subsequent applications could be optimized, and sustainable pest management techniques could progress.

Chapter-3

Project Objective

This project stands out as a light of intelligent and automated solutions in the face of agriculture's ineffective and unsustainable pesticide application procedures. In order to construct a system that smoothly integrates a 2-dimensional Convolutional Neural Network (2-D CNN) model with an automated system based on a Raspberry Pi, this project makes use of cutting-edge technologies. This integration's main purpose is to accomplish multiple important objectives in order to modernize the application of pesticides.

Initially, the goal of the project is to create a CNN model that is incredibly precise. This program learns to recognize different plant illnesses through image recognition after being trained on large datasets of plant pictures. With this additional capability, farmers can reduce the need for needless chemical use by acting quickly to address problems and applying pesticides selectively to impacted regions.

Secondly, the Raspberry Pi acts as the automation system's brain, enabling control and real-time decision-making over the application process. The Raspberry Pi may dynamically modify the application rates by evaluating the CNN model's data. By adjusting for the intensity and existence of the identified disease, this modification guarantees that the exact amount of pesticide is applied, maximizing resource utilization and reducing waste.

Chapter-4

Convolutional Neural Networks

Deep learning has made Convolutional Neural Networks (CNNs) the standard paradigm. CNNs are especially good at applications involving visual input, such as image detection. This chapter explores the intricate workings of CNNs, including their functions, architectural underpinnings, and inherent benefits in the field of image identification.

4.1 Architecture of CNN:

A typical CNN architecture consists of a number of linked layers working together in a hierarchical manner to extract features and, in the end, identify or classify objects in a picture. Here's an explanation of the main components that enable this amazing ability:

- **Convolutional Layers:**

  These layers serve as a CNN's foundation. They use filters, sometimes referred to as kernels, which move through the input image in a sliding window manner. Every filter is painstakingly created to identify particular visual elements in the picture, such lines, forms, edges, or color patterns. The filter works by performing a mathematical operation between its filter weights and the corresponding values of the relevant pixels in the picture as it passes across the image. The output of this process is a feature map that indicates the locations and existence of the features in the image that the filter was intended to identify. A single convolutional layer can have multiple filters applied to it, producing a series of feature maps that each highlight a different aspect of the image.

- **Pooling Layers:**

Pooling layers, which come after convolutional layers, are essential for lowering the dimensionality of feature maps. This improves computing efficiency while also aiding in the management of overfitting, a condition in which the model performs badly on unobserved data due to its excessive acuity with the training set. By applying a pooling function (such as max pooling or average pooling) to a predetermined area of the feature map, pooling layers achieve down sampling. While average pooling determines the average value, max pooling chooses the maximum value from this region. The most important features are retained while the spatial dimensions of the feature maps are decreased by this method.

- **Activation Layers:**

These layers give the network a vital feature called non-linearity. Acquiring knowledge of intricate patterns in picture data requires this. Similar to basic addition, linear activation functions are limited to modeling linear relationships. The network can learn more complex characteristics and their interactions by introducing non-linear decision limits through activation functions like Sigmoid or ReLU (Rectified Linear Unit). These functions introduce non-linear activation based on a threshold or sigmoid function, so transforming the output of the preceding layer.

- **Fully-Connected Layers:**

Fully-connected layers are commonly used in the later stages of a CNN. These layers act in a manner akin to conventional neural networks, in which every neuron in one layer is linked to every other neuron in the layer above it. Fully-connected layers, on the other hand, handle the output that is

flattened from the convolutional and pooling layers (turned into a single vector), in contrast to convolutional layers that work on specific areas of the feature maps. These layers use sophisticated reasoning to identify things in the image or classify it. For classification tasks, fully-connected layers usually use activation functions such as Softmax, where the output indicates the likelihood that the input belongs to a particular class.

4.2 2-D Convolutional Neural Networks

The most common architecture for image detection tasks is represented by 2D CNNs. They interpret images as two-dimensional arrays of pixel values, which are commonly shown as RGB (Red, Green, and Blue) channels. This is a thorough explanation of how well 2D CNNs do image detection:

- **Input Layer:**

  The pre-processed image data is received by the input layer, which starts the journey. Usually, this data is represented as a three-dimensional tensor, with the third dimension representing the RGB color channels and the first two dimensions representing the image width and height.

- **Feature Extraction through Convolutional Layers:**

  Upon entering the network, the pre-processed image is presented with multiple convolutional layers. A set of feature maps is produced by the numerous filters that each convolutional layer applies to the input. These filters begin by identifying low-level features such as edges and lines, and as the network advances through deeper layers, they grow increasingly complicated. This enables the network to extract more and more complex information from the image, such as textures, contours, and object sections.

- **Dimensionality Reduction with Pooling Layers:**

To lower the dimensionality of the feature maps, pooling layers are put between convolutional layers in a deliberate manner. In addition to increasing computing efficiency for training and inference, this compression also lessens the effects of overfitting. As previously indicated, pooling layers accomplish this by putting a pooling function—such as average or max pooling—to a predetermined area of the feature map. The most noticeable features are retained while the feature maps' spatial dimensions—width and height—are decreased by this technique. The network can avoid overfitting to unimportant characteristics in the image by focusing on the most important components of the retrieved features by including pooling layers at regular intervals.

- **Object Detection:**

The main goal of image detection jobs is to identify the kind and location of objects within the image in addition to classifying the content of the image. Additional layers are usually added following the feature extraction phases in order to do this. These layers use a variety of methods to locate and identify objects, including:

  - Region Proposal Networks (RPNs):

    These specialized layers are essential for producing potential object-containing regions in the image. Using a sliding window technique, RPNs work with the feature maps that the convolutional layers generate to suggest rectangular regions (bounding boxes) that are probably going to contain objects. By doing this, the search space for object detection is

greatly reduced, directing the network's attention to portions of the image that show promise.

- Classification Layers:

Classification layers take over once RPNs provide candidate regions. These layers identify two important factors by analyzing the attributes within each suggested bounding box: (1) Is there an object inside the bounding box? (2) If so, what kind of thing is it? Typically, Softmax activation functions are used in classification layers. The output of this function indicates the likelihood that the proposed region belongs to a particular item class.

- Bounding Box Regression:

Bounding box regression refines the position and size of the bounding boxes, while classification layers determine the existence and kind of objects within specified regions. The discovered items will be localized more precisely thanks to this phase. Bounding box regression layers generally tighten the fit around the image's actual object by modifying the coordinates and dimensions of the suggested bounding boxes using methods like linear regression.

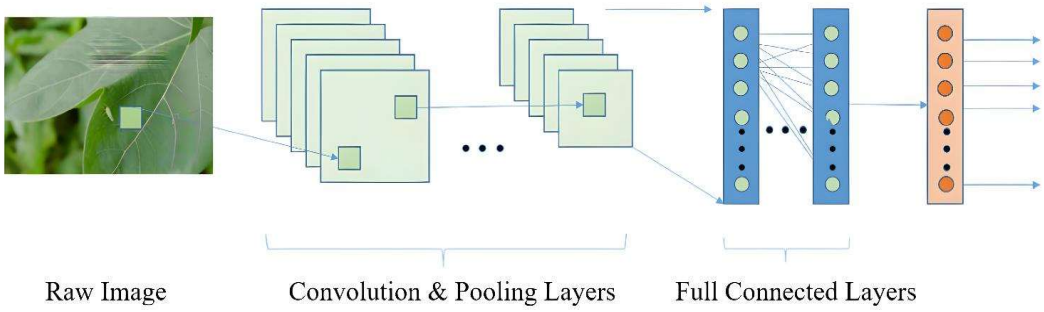


Fig.4.2. Architecture of CNN

Chapter-5

Project Methodology

This chapter outlines the comprehensive methodology for training a Convolutional Neural Network (CNN) model for image detection and deploying it on a Raspberry Pi.

5.1 Training the CNN Model:

5.1.1 Dataset Selection:

The "New Plant Disease Dataset" from Kaggle is a publicly accessible dataset. It has about 87,000 RGB pictures of both healthy and damaged crop leaves divided into 38 different classifications. With a separate directory containing 33 test photos for assessing model performance on untested data, this dataset is split 80/20 for training and validation. This choice is in line with the goal and provides a significant amount of data for training, while also taking into account the constraints of the Raspberry Pi 2 and the requirement for potentially resource-efficient training methods. The 38 classes make sure that trainees are exposed to a wide variety of crop diseases, and keeping the directory structure intact during the split makes it easier to retrieve data.

5.1.2 Data Preprocessing:

Once the "New Plant Disease Dataset" has been chosen, careful data preprocessing is necessary to get the data ready for the Raspberry Pi 2's CNN model training. This step guarantees that the model collects high-quality features from the pictures, which will ultimately result in the best possible illness detection performance. An overview of the key data preparation methods used on this project is provided below.

- **Image Dimension Standardization:**

One of the most important things is to make sure that every image is the same size and fits into the selected CNN architecture. This uniformity guarantees that every image is handled consistently in both the training and inference (prediction) stages. The degree of detail necessary for precise illness diagnosis, the processing constraints of the Raspberry Pi 2, and model complexity all have to be carefully balanced in the selection of image size.

- **Pixel Value Normalization:**

Techniques for normalization are essential for making training easier. This entails scaling pixel values within a range that has been predetermined (e.g., -1 to 1). By ensuring that all pixel values fall on the same scale, this normalization helps the resource-constrained device's model to converge more quickly and train more efficiently. Normalization methods that are frequently used are standardization and min-max scaling.

- **Data Augmentation:**

It is optional to use data augmentation techniques to artificially increase the dataset's size and improve the model's resilience. In order to do this, the training images are subjected to random changes such rotation, jittering of color, brightness, contrast, and saturation, cropping, and flipping (horizontal and vertical). By using these methods, the model is encouraged to acquire strong features and improve its ability to generalize to new images that contain variances that could occur in actual crop disease situations. But it's important to use augmentation strategies that make sense for the particular

job at hand and stay away from adding irrational distortions that can impair model performance.

- **Strategic Data Splitting**

A deliberate division of the preprocessed dataset into test, validation, and training sets is made. Model learning is based on the training set, which is usually the largest piece. A smaller subset of the data, the validation set, serves as an essential training-day performance monitor for the model, assisting in preventing overfitting. Finally, the model's ultimate performance on real-world data is assessed using the test set, which consists of completely unseen photos. For training, validation, and test sets, a typical split ratio is 80/10/10. The ideal ratio, however, can change based on the size of the dataset and the particular needs of the project.

5.1.3 Choosing a CNN Architecture:

For real-time agricultural disease diagnosis on a Raspberry Pi 2, the project makes use of a pre-trained ResNet architecture. Getting the vanishing gradient issue fixed and accomplishing effective deployment on the resource-constrained device are the two main requirements that will determine this decision.

ResNet architectures help to overcome a common challenge in deep neural networks: the vanishing gradient problem. When training, this issue makes it more difficult to learn in early stages. For complicated applications like crop disease detection, ResNet's skip connections enable gradients to pass directly across the network, supporting fast learning even in deeper structures.

5.1.4 Model Training:

In order to obtain high accuracy in real-time crop disease detection tasks using Raspberry Pi 2, this critical phase entails tweaking the model's parameters. Below is a summary of the main actions involved in this process:

- **Pre-trained Knowledge:**

  We start with the weights of the pre-trained ResNet. These frozen weights offer a reliable foundation for the classification of diseases. On the particular dataset, only the last layers are refined.

- **Learning through Optimization:**

  The training procedure is directed by a loss function, similar to categorical cross-entropy. Based on the training data, the model's parameters are changed (for example, by using the Adam optimizer) to minimize the loss and learn to distinguish between healthy and diseased leaves as well as different disease classes.

- **Strategic Data Split:**

  Two sets of preprocessed data are separated: training and validation. The validation set (20%) keeps track of performance throughout training to assist prevent overfitting, while the training set (usually 80%) is used to train the model.

- **Monitoring and Early Stopping:**

  On both sets, training measures like loss and accuracy are tracked. If the training loss keeps going down but the validation loss goes up, early halting can be employed to stop overfitting.

- **Fine-tuning for Efficiency:**

  The last layers are optimized for disease classification, however using pre-trained weights cuts down on training time. But given the restrictions of the Raspberry Pi 2, methods like lowering the learning rate or utilizing a learning rate scheduler might be required for effective deployment on the device.

5.1.5 Model Evaluation

The model's efficacy for real-time disease detection is evaluated after the training phase. The main statistic, accuracy, is assessed using the hidden validation set. This evaluates how well the model can distinguish between healthy and unhealthy leaves in a variety of characteristics. Furthermore, memory and precision offer more profound insights. Recall shows the percentage of true positives recognized out of all real diseases, whereas precision measures the true positives (diseases accurately diagnosed out of all positive predictions). By examining both, possible biases or restrictions in the diagnosis of particular diseases can be found. Areas where the model has trouble with particular illness kinds are further revealed by visualization approaches such as confusion matrices.

Lastly, an independent, hidden test set is used to assess the model's performance. The model's capacity to generalize to real-world data with possible deviations from the training data is evaluated in this final assessment. Recalculated metrics like as recall, accuracy, and precision provide a practical metric for in-the-moment deployment. These thorough techniques offer a thorough grasp of the model's advantages and disadvantages, assisting in judgments regarding the model's applicability for Raspberry Pi 2 real-time disease detection as well as possible future optimization initiatives.

5.2 Implementation on Raspberry Pi:

The deployment procedure for a Raspberry Pi 2 device-based real-time crop disease detection system with integrated automated spraying capabilities is described in this section. Because of its limited resources, the Raspberry Pi 2 requires careful planning before implementation in order to guarantee accurate and efficient performance. The main procedures are delineated in this section.

5.2.1 Model Optimization:

The Convolutional Neural Network (CNN) model that is selected must be optimized for effective deployment due to the Raspberry Pi 2's low processing capabilities. Here, we examine three essential optimization methods.

- **Quantization:**

  This method converts 32-bit floats into 8-bit integers, hence decreasing the model's weights and activations' precision. Although there may be a tiny drop in accuracy as a result, there are substantial advantages:

  - Smaller variant: Less storage space is needed for lower precision formats, which results in a lighter variant that is simpler to install on the Raspberry Pi.
  - Faster Inference Speed: As lower precision calculations are usually easier to execute on the hardware of the device, real-time disease detection predictions can be made more quickly.

- **Pruning:**

  This method involves deleting unnecessary filters or connections from the CNN design. Either during or after the model has been trained, pruning can be done. Finding and removing connections that barely affect the model's performance is the aim. This may result in:

- Smaller Model Size: By eliminating pointless connections, the model becomes smaller overall and is more suited for deployment on the device with limited resources.
- Faster Inference Speed: The model may require less computing for inference, which could result in faster predictions, if there are fewer connections to process.

5.2.2 Framework and Library Selection:

Choosing the right deep learning framework is essential for a successful Raspberry Pi 2 deployment. Embedded device deployment-specific optimized libraries are available through frameworks such as TensorFlow Lite and Core ML. These libraries offer features and resources to guarantee effective model execution on hardware with limited resources. The framework of choice should provide the required functionality for model conversion, inference, and integration with the Raspberry Pi 2 environment, while also being compatible with the format of the pre-trained and fine-tuned CNN model.

5.2.3 Model Conversion:

The pre-trained and refined CNN model (such as TensorFlow.h5 or PyTorch.pt) must then be optimized and formatted so that it can be used with the Raspberry Pi 2. Conversion tools are available in TensorFlow Lite and other frameworks, which can change the model into a more compact and lightweight format that is suitable for embedded and mobile devices. In order to decrease the size of the model and increase the speed of inference on the Raspberry Pi 2, this conversion procedure frequently uses quantization.

5.2.4 Raspberry Pi Setup with Motor Pump Control:

It is necessary to set up a development environment before the Raspberry Pi 2 can be deployed. Installing the required libraries and frameworks is part of this process. The Raspberry Pi 2 will be able to run the adapted CNN model for real-time disease prediction with the help of these tools and frameworks.

Installation of the Motor Pump Control Library: Install a library that works with the Raspberry Pi 2 and the selected motor pump. This library will offer features for managing the motor and maybe keeping track of its condition (such as turning it on or off).

5.2.5 Image Acquisition and Preprocessing:

To take pictures of crops in order to detect diseases, the system needs to be integrated with a camera module that is fastened to the Raspberry Pi 2. It is necessary to create a script or application in order to:

• Take pictures using the camera module.

• To ensure accurate predictions, preprocess the collected photos (e.g., resize, normalize) to match the format the model expects. Preprocessing methods make sure that the input photos have the same size and format as the training data that was used to create the model.

5.2.6 Model Inference:

The model analyzes the image features and outputs a prediction on the health status of the crop (e.g., healthy or diseased).

5.2.7 Motor Pump Control based on Inference Results:

If the model predicts a specific disease requiring pesticide application:

The motor pump gets activated for a predefined duration or until a sensor reading indicates sufficient coverage.

Otherwise for healthy crop the motor pump will remain off.

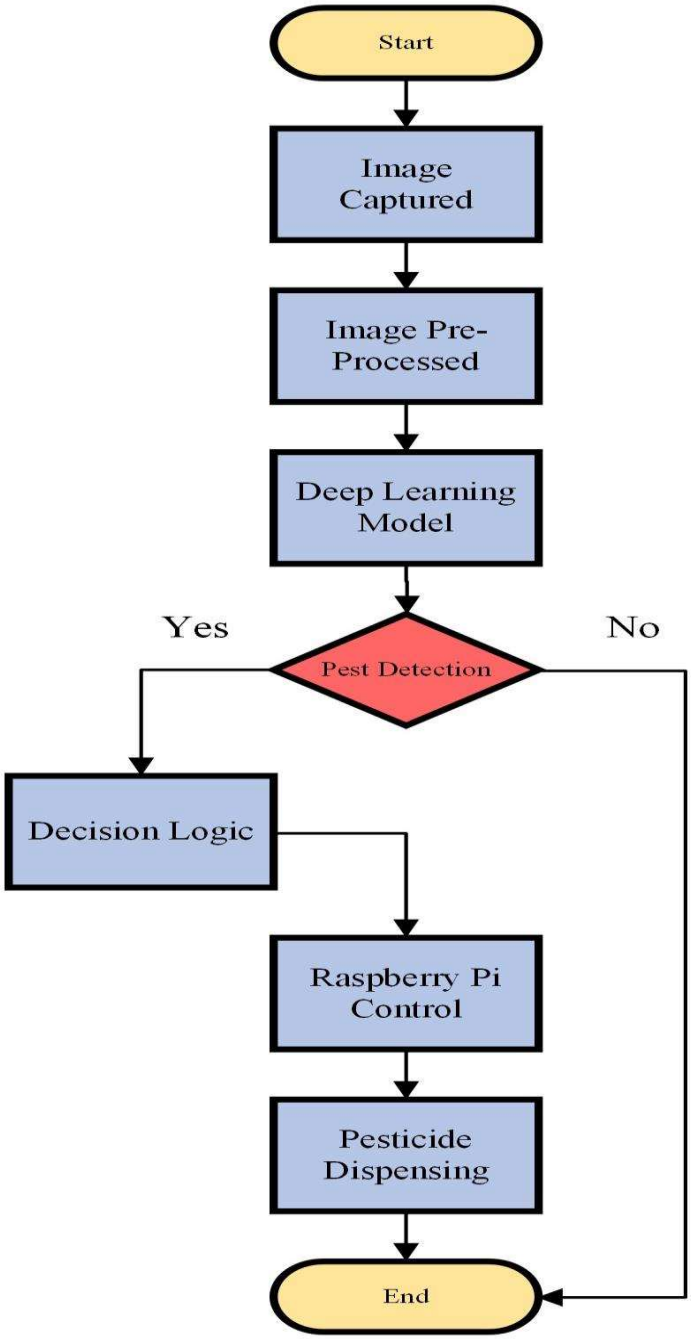


Fig.5. Flowchart of the methodology

Chapter-6

Details of Project Work

6.1 Hardware Components used:

6.1.1 Raspberry Pi 2 Microcontroller

The Raspberry Pi 2 Model B is the central component of the real-time plant disease detection and pesticide application system. The brains of the system are housed in this credit card-sized computer, which provides the processing capacity and adaptability required for image processing, CNN model inference, and control logic execution.

A quad-core ARM Cortex-A7 CPU operating at 900 MHz powers the Broadcom BCM2836 System-on-Chip (SoC) found in the Raspberry Pi 2 Model B. For computationally demanding workloads like real-time picture pre-processing and illness detection utilizing the trained CNN model, this multi-core architecture offers enough power. Furthermore, the system's 1GB of LPDDR2 RAM enables effective data management and multitasking. This amount of RAM guarantees that the trained CNN model loads smoothly, that intermediate processing results are stored, and that relevant software applications are executed.

The Raspberry Pi 2 Model B is well-equipped in terms of connectivity. Peripherals such as a keyboard and mouse for human interaction during setup and monitoring, an external storage device for data logging, and a camera for picture capture can all be connected via four USB 2.0 ports. Wired network connectivity is made possible with a 10/100 Base T Ethernet port, which enables the Raspberry Pi to link to a local network for possible cloud storage, remote monitoring, and integration with an IoT platform for more extensive

communication features. By connecting the HDMI and composite video outputs to a display, the system may also be seen visually.

The Raspberry Pi 2 Model B is better suited for this project because of its extra features. The operating system, software programs, and trained CNN model are all stored primarily on a Micro SD card slot. Plant disease detection can be facilitated by connecting a camera module through the use of the Camera Serial Interface (CSI). Lastly, by integrating with external sensors or actuators, the GPIO (General Purpose Input/Output) header offers a set of programmable pins for potential system extensions in the future.

All things considered, the Raspberry Pi 2 Model B is an excellent option for the computational requirements of the project since it hits the optimal mix of processor power, memory capacity, and pricing. The system's portability and energy efficiency are further enhanced by its small size and low power usage.at minimum cost and effort.

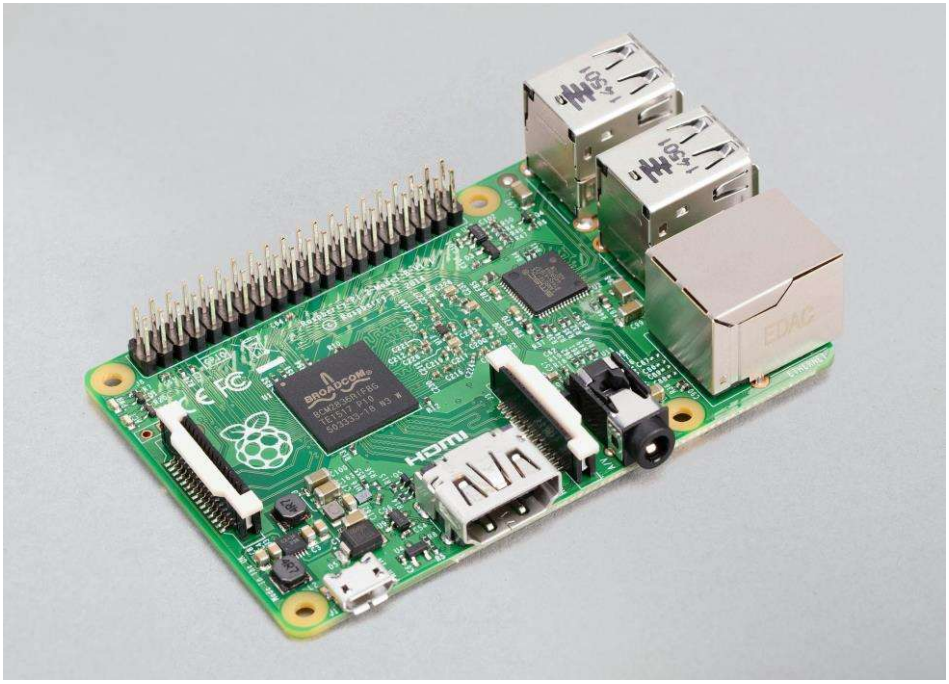


Fig.6.1.1a. Raspberry Pi 2 Model B

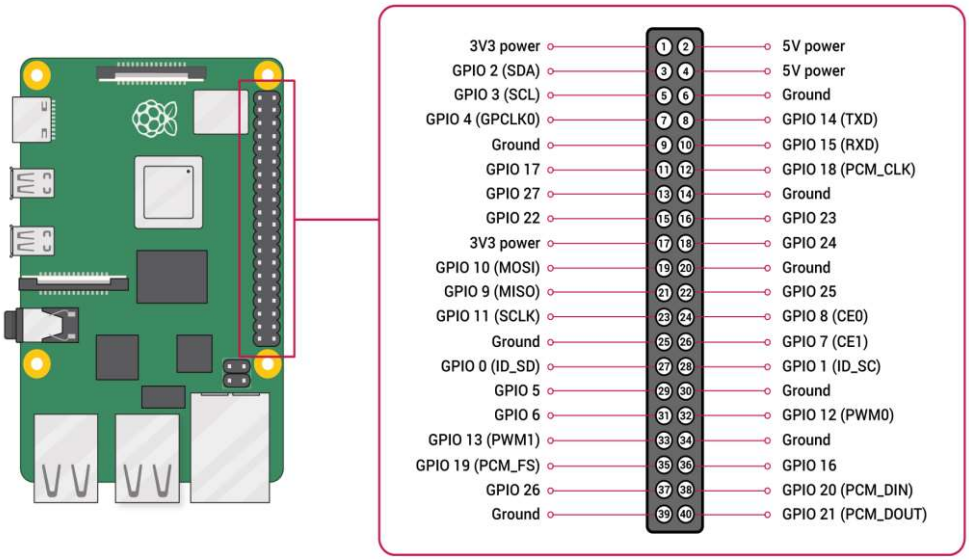


Fig.6.1.1b. Raspberry Pi 2 Pin Description

One of the Raspberry Pi 2 Model B's strongest features is its 40-pin GPIO header. Because these ports may be programmed, the Raspberry Pi can communicate with a wide range of external sensors and devices. They can be set up as outputs to operate actuators like motors or LEDs, or as inputs to receive signals from sensors like light detectors and temperature gauges. Features including sensor-based environmental data collecting, sensor-readout-based device control, and I2C-based device-to-device communication are made possible by this adaptability. Developers can use the GPIO header's programmability to make the Raspberry Pi capable of more than just basic computing, turning it into a platform for creative projects that engage with the outside world.

6.1.2 Motor Pump

In order to enable features like automated irrigation based on real-time disease detection, the project includes a 12V DC diaphragm pump. Diaphragm pumps are beneficial in this application in a number of ways. Because they are self-priming and can start pumping without human assistance, they are perfect for automated systems. They work more silently than some other pump types, and their small size makes them ideal for installations with limited space.

The two most important parameters to take into account when choosing a particular pump model are flow rate and pressure. A small-scale irrigation system might be able to use the suggested pump, which has a flow rate of 210 L/hr (3.5 L/min) according to the materials supplied. To select a pump with sufficient pressure to overcome system resistance elements like hose friction and elevation fluctuations, use the given "head" of 4-5 meters as a guide.

Fig.6.1.2. 12V DC Motor Pump

A sufficient 12V DC power supply that can power the Raspberry Pi and the pump is required for system integration. According to the data, the pump needs a minimum of two amps of current. Even though the Raspberry Pi requires less

power, the capacity of the power supply must also account for the Raspberry Pi's power consumption.

It is possible to program the Raspberry Pi code to operate the pump in response to disease detection. For example, by turning on the pump in certain locations known to house specific diseases, targeted irrigation can be started. Choosing the right sprinkler or nozzle type and tubing (such as 7mm silicone tube) is also necessary to achieve the required watering pattern and coverage area. These factors are properly taken into account, and the Raspberry Pi system is connected with the selected 12V diaphragm pump to produce a dynamic automatic watering system that reacts to real-time disease detection data.

6.1.3 LCD with I2C

The plant disease detection device has a 16x2 character LCD display with a strong yellow backlight for real-time feedback. When combined with an I2C Serial Interface Adapter Module, this display provides a practical and easy-to-use communication solution.

Picture a little informational TV screen with a bright yellow backdrop. Clear black text with 16 characters per line spread across two lines is displayed on this LCD panel. Via the I2C adapter, the Raspberry Pi, the system's central processing unit, controls the LCD and dynamically updates the data presented according to how the system is operating.

Quick feedback is essential for making well-informed decisions. With notifications like "Disease detected!" or "Healthy plant" shown instantly, the LCD offers insightful information about the health of plants. Notifications such

as "System idle" or "Pump activated" are clearly visible on the bright yellow screen, making it easy for users to keep an eye on system operations.

The Raspberry Pi and the LCD display can communicate more easily thanks to the I2C Serial Interface Adapter Module. Devices can communicate using just two wires—a serial data line and a serial clock line—in this condensed communication mode known as I2C (Inter-Integrated Circuit). Wiring complexity

Fig.6.1.3. 16X2 LCD with I2C

is greatly reduced using I2C interfaces as opposed to traditional interfaces that require many data lines.

The I2C adapter functions as a translator, transforming data from the Raspberry Pi into an LCD display-compatible format. This conversion simplifies communication and reduces the number of cables required for connections.

6.1.4 Relay

A 5-volt, 1-channel relay module is integrated into the project to control the operation of the 12V DC motor pump. The Raspberry Pi, the system's central processing unit, and the motor pump are connected by the relay module.

Consider it as a switch that is electronically operated. The relay module receives a control signal from the Raspberry Pi, which is powered by low voltage DC. The relay is triggered by this signal, which opens a circuit channel and permits the motor pump to run at the higher voltage current—12 V DC in this example. By managing the motor pump, this indirect control technique allows the Raspberry Pi to potentially initiate targeted irrigation in response to disease detection results.

Fig.6.1.4. 5V Single Channel Relay

It's important to keep in mind that, even with a safe DC system, electrical literacy and following safety procedures are prerequisites.

For our project, the 5V 1-Channel Relay Module has the following advantages:

• Safe Control: The overall safety of the system is improved by isolating the low voltage control signals from the Raspberry Pi from the higher voltage motor pump circuit.

• Compatibility: The Raspberry Pi is compatible with the relay thanks to its broad control signal range.

• Compact Design: Its tiny size (48 x 29 x 17 mm) and low weight (15 grams) help to create a space-efficient design.

This relay module's integration fills in the communication gap that exists between the Raspberry Pi's control signals and the motor pump's operation. This enables the device to diagnose plant diseases in real time and automate irrigation.

6.1.5 Ultrasonic Sensor

With the help of the HC-SR04 ultrasonic distance sensor, your robotics project or Raspberry Pi can identify and steer clear of obstacles like a spare pair of eyes. This sensor measures the separation between itself and surrounding objects using sonar technology, which is akin to that used by submarines.

The HC-SR04 uses ultrasonic transmitters to send out high-frequency sound waves. These waves radiate outward until they come into contact with a solid object, after which they return to the sensor. The sensor has a receiver built in to pick up on these reverberating echoes. The timing difference between the broadcast and received signals is then measured by the control circuit. This time difference can be used to calculate the distance between the object and the sensor with some smart math.

Fig.6.1.5. Ultrasonic Sensor

Using an HC-SR04 sensor in your Raspberry Pi project has a number of benefits. First of all, it provides measurement that is non-contact, meaning that items do not need to be physically touched by the sensor. Because there is no

chance of breaking fragile objects or the robot itself, it is perfect for safe robot navigation. Second, integration is quite simple. For simple applications, the HC-SR04 can be connected straight to your Raspberry Pi with a few resistors. As an alternative, you can connect many sensors at once with an Ultra Borg shield without taxing the Raspberry Pi's CPU. Lastly, the HC-SR04 is an excellent option for Raspberry Pi applications on a tight budget because it is both economical and energy-efficient. It's crucial to remember that although the sensor uses a 5V power source to function, its output voltage needs to be regulated down to 3.3V to match the Raspberry Pi's operating voltage.

With a precision of ±0.3cm, the HC-SR04 has an outstanding measurement range of 2cm to 400cm (1" to 13ft). It has a conical pattern with a sensor angle of 30 degrees and operates between -15°C and 70°C. The sensor itself is small, with dimensions of only 43 x 20 x 15 mm.

6.1.6 Motor Driver L298N

When it comes to motor control, the L298N integrated circuit (IC) is the best choice, especially for builders and enthusiasts. With its dual H-bridge architecture, this flexible chip can operate a single stepper motor or two DC motors in both directions. Put more simply, it controls the speed and direction (forward or backward) of the linked motors.

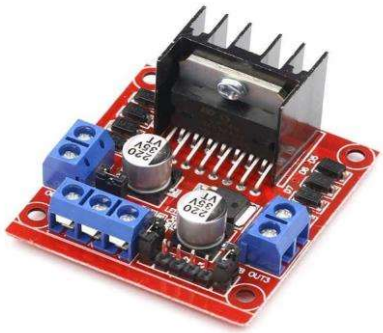


Fig.6.1.6. Motor Driver

The H-bridge configuration holds the key. Consider an electronic circuit that has four switches set up in a "H" configuration. The L298N regulates the current flow to the motor, effectively dictating the direction of rotation, by carefully turning them on and off.

Because of this feature, the L298N is an ideal collaborator for a wide range of Raspberry Pi applications. Although Raspberry Pi's GPIO pins can be used to directly control more basic motors like servos, applications requiring more power or control over several DC motors are best suited for the L298N. The L298N, for example, would allow the Raspberry Pi to control the direction and speed of the two DC motors powering the robot's wheels, enabling it to navigate a line based on sensor readings. Take the example of a line-following robot.

The L298N has a number of features that are easy to use. It is appropriate for a range of projects since it can manage motor supply voltages between 5V and 46V and currents up to 2A per motor. Additionally, it operates the H-bridges using independent low-voltage control signals (usually 5V) to ensure Raspberry Pi compatibility. This makes the L298N a well-liked option for building Raspberry Pi controlled robots or other motor-driven projects, in addition to its affordability and sturdy architecture.

6.1.7 ESP-32:

The ESP32-WROOM-32 development board is a developer-friendly platform specifically designed for creating Internet of Things (IoT) projects. The ESP32 microcontroller, with its dual-core CPU for effective data handling, is at the heart of it all. By incorporating necessary parts like amplifiers and antenna switches, this chip reduces the size of the board. For maximum performance without compromising battery life, the ESP32 itself makes use of low-power

technologies. The ESP32-WROOM-32's extensive feature set is designed to satisfy developers. Smooth wireless communication is made possible by built-in Bluetooth and Wi-Fi, and versatile network configurations are made possible by the multiple Wi-Fi operational modes (access point, station, or both).

In conclusion, the ESP32-WROOM-32 development board offers a small and effective way to build feature-rich Internet of Things devices by combining a potent microcontroller with necessary parts. Its software support, several operating modes, and integrated wireless connectivity make it a flexible platform for developers to start their Internet of Things projects. Additionally, the ESP32 linked with an L298N motor driver to allow for remote wheel control.

6.1.8 12V DC Motor:

This is a 100RPM 12V DC motor. These are straightforward DC motors with shaft gears to get the best possible performance characteristics. Because the shaft of its gearbox assembly passes through its center, they are referred to as center shaft DC geared motors.

Fig.6.1.8. 12V DC Motor

Utilizing these DC motors in standard size is quite simple. This motor, which has a voltage of between 5 and 35V DC, can be utilized with the L298N H-bridge module with inbuilt voltage regulator motor driver. Numerous robotic

applications, including all-terrain robots, can make use of this DC motor with a speed of 100 RPM and 12 V. These motors feature an easy-to-connect 3 mm threaded drill hole in the middle of the shaft for attaching to the wheels. An internally threaded shaft with a nut and threads makes it simple to attach the shaft to the wheels. These heavy-duty DC motors feature a sturdy metal or plastic gearbox.

6.1.9 12V DC Rechargeable Battery Pack:

Three linked 18650 Li-ion cells make up the rechargeable 12V Lithium-ion battery pack used in this project. With a nominal voltage of 11.1V and a maximum voltage of 12.6V, the pack gives the system a portable and effective power supply.

Fig.6.1.9 12V Li-ion Battery Pack

High energy density, or the ability to store a substantial amount of power in a comparatively small amount of space, is what makes lithium-ion technology perfect for portable applications.

6.2 Software Used:

6.2.1 Jupyter Notebook:

The foundation for creating and refining the Convolutional Neural Network (CNN) model utilized in this project was Jupyter Notebook. This interactive development environment was very good at visualizing and manipulating data. It made operations like loading and examining picture collections with crop samples—both healthy and diseased—easier. In order to prepare the data for the CNN model, Jupyter Notebook also expedited data pre-processing tasks like image scaling, normalization, and potential data augmentation strategies.

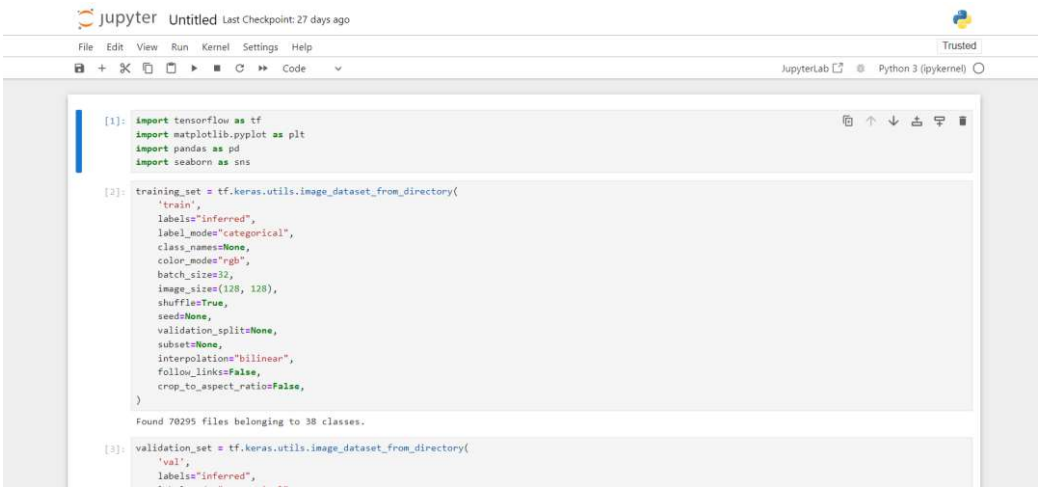


Fig.6.2.1. Jupyter Notebook Environment

The interactive coding environment of Jupyter Notebook is one of its strongest points. This made it perfect for employing libraries like TensorFlow or PyTorch to define and construct the CNN model architecture. Additionally, it made it possible to experiment with different model architectures, visualize the training process, and tune hyperparameters like learning rate, optimizer, and activation functions. Using Jupyter Notebook to plot training and validation loss curves

made it easier to evaluate model convergence and see any overfitting problems that might have arisen during training.

Jupyter Notebook provided benefits in code reusability and documentation beyond its technical capabilities. Code segments for pre-processing data or evaluating models could be readily repurposed in various trials. To further record the model architecture, training details, and development process, notes and explanations were inserted within the notebook cells. This is essential for sharing work with partners or going back to it later.

After training the CNN model successfully in Jupyter Notebook, it was quickly transformed into a TensorFlow Lite model. The model is optimized throughout this translation process for deployment on low-resource mobile and embedded devices, such as the Raspberry Pi used in this project. On the Raspberry Pi in the deployed system, the TensorFlow Lite format guarantees effective inference for real-time crop disease diagnosis.

6.2.2 Geany:

Geany is a particularly strong and effective code editor for Raspberry Pi real-time plant disease detection systems. Its cross-platform compatibility, which runs well on Raspbian or Ubuntu on Raspberry Pis and enables cross-platform programming, demonstrates its adaptability. Geany's lightweight design reduces resource use, which is important given the Raspberry Pi's constrained RAM and processing power. Geany's project management tools streamline project organization and make it easier to navigate by grouping relevant code files together.

Geany's extensibility enables plugin modification. These plugins include debugging tools, syntax highlighting for languages like Python and C++, and

code completion. This gives you the ability to customize Geany to meet the unique requirements of your project, possibly incorporating features for machine learning frameworks or image processing libraries. Moreover, Geany's intuitive interface reduces clutter so you can concentrate on writing code for your plant disease detection system.

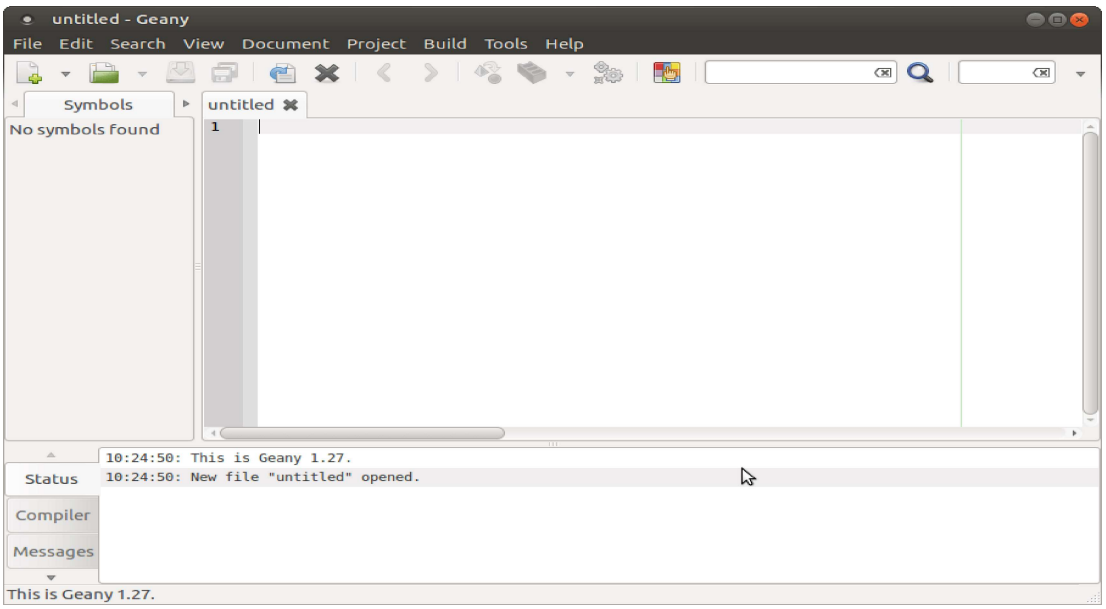


Fig.6.2.2. Geany Environment

Beyond its technical advantages, Geany has a sizable and vibrant community with a wealth of online resources and discussion boards. This encourages troubleshooting and knowledge exchange, which can help you use Geany for your Raspberry Pi project more successfully. Since Geany is an open-source project, you can use it for free and it is always being updated with new features and bug fixes to ensure the maintainability of your project. All things considered, Geany's lightweight architecture, cross-platform interoperability, extensibility, project management, and user-friendliness combine to make it an effective tool for creating and implementing real-time plant disease detection systems on Raspberry Pi.

6.3 Connection Diagram:

6.3.1 Connection Diagram of the Model:

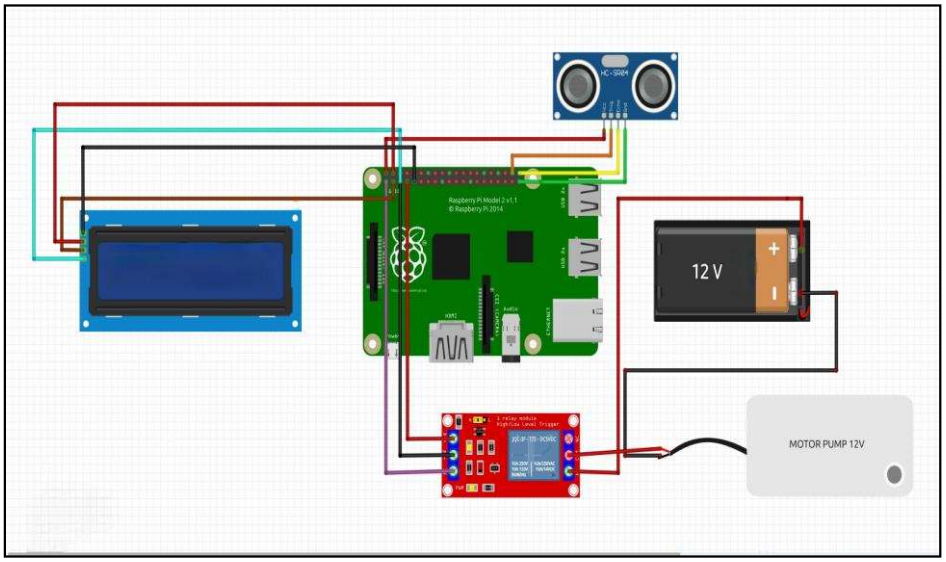


Fig.6.3.1. Connection Diagram of the Model

6.3.2 Connection Diagram for Wheel Operation:

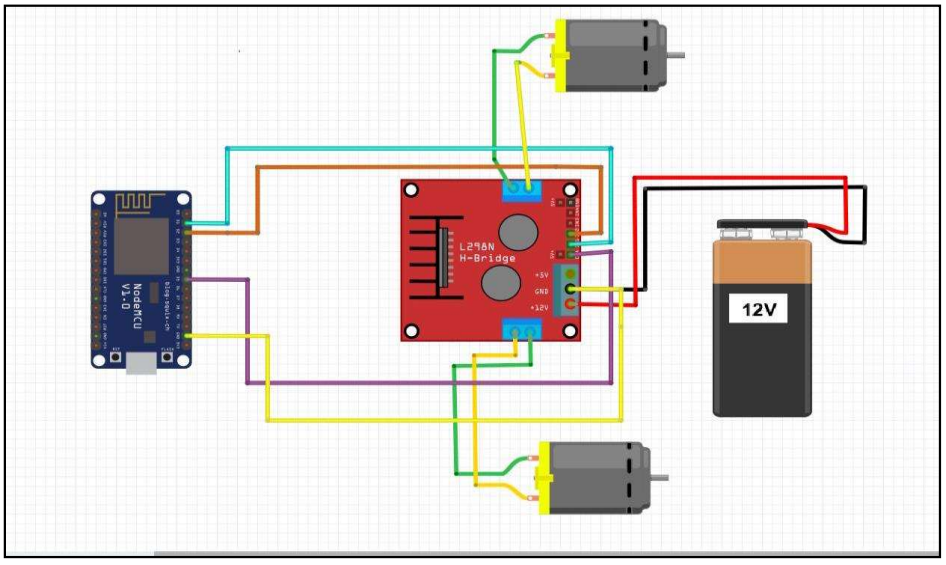


Fig.6.3.2. Connection Diagram for Wheel Operation

6.4 Hardware Setup:

The successful training of a convolutional neural network (CNN) model and its deployment on a Raspberry Pi module using the TensorFlow Lite library mark important milestones. The next crucial step is to assemble a functional hardware model by integrating these components. This model consists of a frame constructed from welded iron rods with cardboard attached over it.

The entire system will be mobile thanks to wheels powered by an ESP32 microcontroller and a motor driver the connection of which is shown above. The upper part of the model consists of Raspberry Pi, Motor Pump, Battery pack, LCD, Ultrasonic sensor, Camera, tank, Sprayer which is connected accordingly as per the connection shown in the above section of the report.

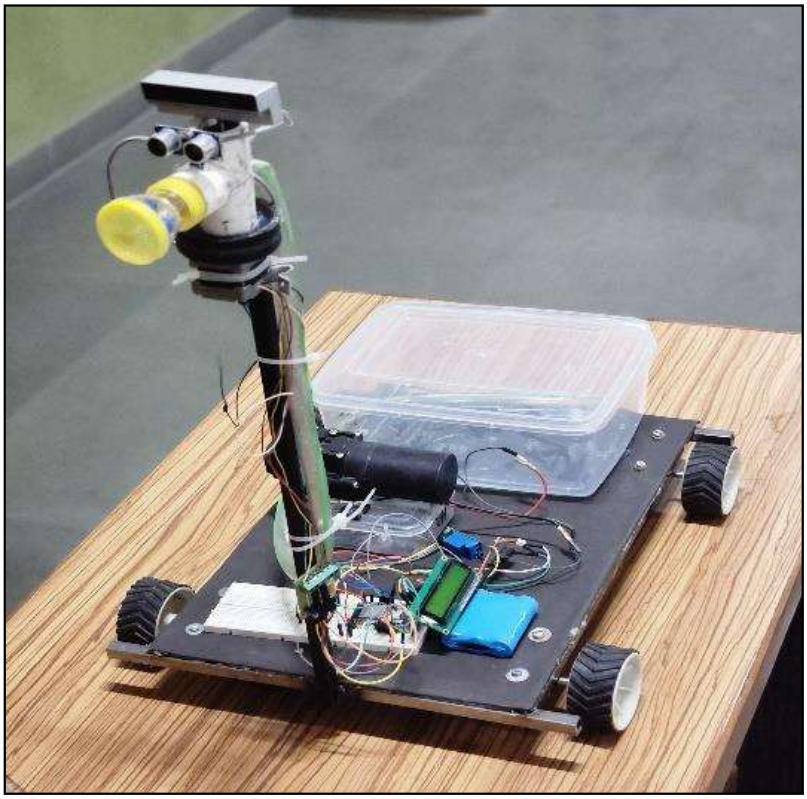


Fig.6.4. Final Hardware Setup

Chapter-7

Result and Discussions

This project effectively used a Raspberry Pi 2 device to develop an automated spraying system with real-time crop disease detection capabilities. For illness prediction, the system used a pre-trained and optimized CNN model, and the motor pump control logic made sure that pesticide application was targeted according to the disease that was detected.

7.1 CNN Model Performance:

The effectiveness of a CNN model hinges on its ability to learn and generalize from the provided data.

7.1.1 Model Accuracy:

The plot titled "Visualization of Accuracy Result" shows the training accuracy (red line) steadily increasing with each epoch. This suggests the model is effectively learning from the training data. The validation accuracy (blue line) also increases but at a slower rate, indicating the model is generalizing reasonably well.

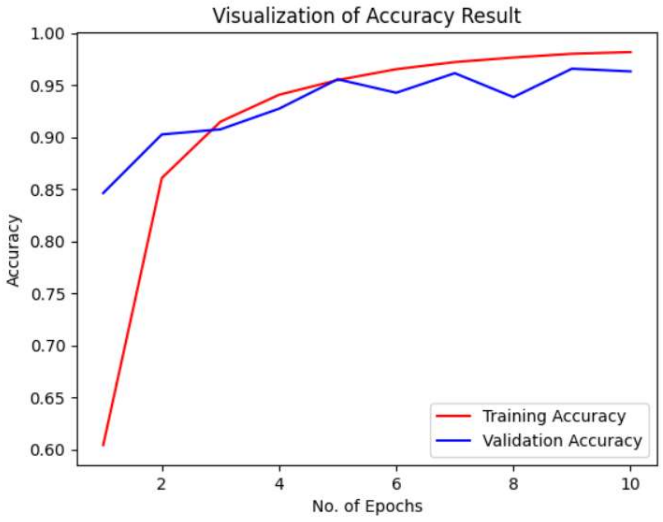


Fig.7.1.1. Visualization of Accuracy Result

The "Validation Accuracy" plot depicts the model's performance on unseen data during training. While the training efficiency reached a high of 98%, indicating strong learning from the training data, the validation accuracy shows a fluctuating pattern. This could suggest potential overfitting, where the model memorizes specifics of the training data and struggles with variations. Alternatively, a small validation set size might contribute to the fluctuations. To address this, we will monitor the training loss for signs of early plateauing and consider techniques like dropout or adjusting the learning rate to promote smoother validation accuracy while maintaining learning efficiency.

7.1.2 Confusion Matrix:

The confusion matrix provides a class-by-class breakdown of our model's performance for 38 categories (0-37). Rows represent actual classes, and columns represent predicted classes. Each cell shows how often the model predicted a class (column) for data points that truly belong to another (row). Ideally, high values concentrate on the diagonal, indicating accurate classifications. Deviations from this pattern reveal confusion between classes. By analyzing these trends, we can pinpoint areas for improvement, such as classes where the model consistently misclassifies one for another.

Additionally, calculating precision (correctly predicted positives) and recall (correctly identified actual positives) for each class offers a more nuanced understanding. This can highlight classes where the model struggles with either over-predicting a category (low precision) or under-predicting it (low recall), guiding further training or data adjustments.

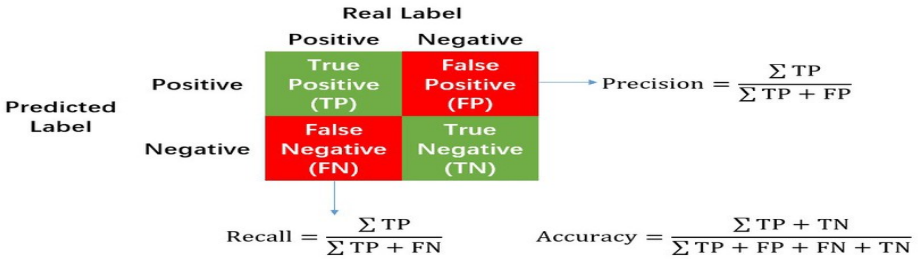


Fig.7.1.2a. Calculation of Precision and Recall

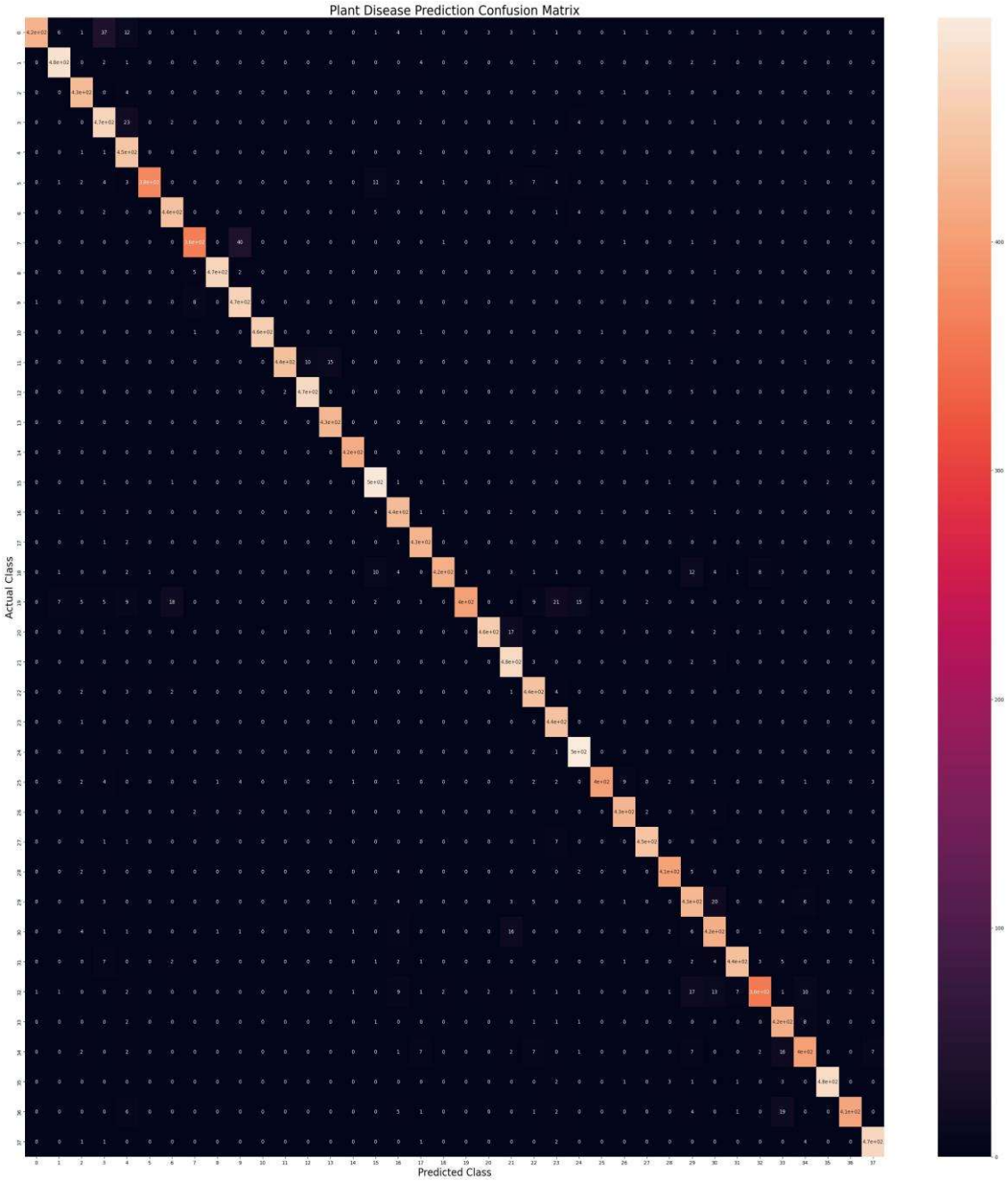


Fig.7.1.2b Plant Disease Prediction Confusion Matrix

7.1 Detection and Pesticide Application:

This section explores the performance of the CNN based system for disease detection and its integration with Raspberry Pi for the targeted pesticide application system.

Our plant disease prediction model, implemented on a Raspberry Pi using a convolutional neural network (CNN) architecture, was evaluated using images of both healthy and diseased leaves. To identify healthy specimens, an image of a healthy leaf was presented. The model, built on a convolutional neural network (CNN) architecture, successfully classified the leaf as healthy shown in fig.7.2a. But, when the model detects a diseased leaf shown in fig.7.2b, the Pi triggers the motor pump for pest application over the diseased plant.

Fig.7.2a. Healthy Plant Detected

Fig.7.2b. Diseased Plant Detected

Chapter-8

Conclusion and Future Scope

In this project, a Raspberry Pi 2 was used to successfully implement an automated spraying system with real-time crop disease detection. Crop diseases are accurately predicted in the field using a CNN model that has been pre-trained and optimized. The predicted disease is integrated with the motor pump control by the system, which applies the targeted pesticide only when required. The environmental effect and total amount of pesticides used could be greatly reduced using this strategy.

This project made two significant progresses. It first successfully adjusted a CNN model to run on a Raspberry Pi 2 constrained resources. This indicates the viability of putting such systems into practice on devices with limited resources, opening up the technology for broader usage. Secondly, the project combined real-time motor pump control with disease prediction, automating the spraying process according to crop health. In addition to saving labor and time, this guarantees accurate application, reducing waste and possible environmental harm.

In the future, the project presents intriguing opportunities for growth. Including other sensors, such as soil moisture sensors, could give a more complete picture of crop health and enable even more precise spraying decisions. In order to reduce pesticide drift, real-time weather data integration may allow the system to modify spraying schedules or cease application during strong winds. Additionally, a mobile application could improve system management and user experience by enabling remote monitoring, data visualization, and possible human control overrides.

References

[1] Ghafar AS, Hajjaj SS, Gsangaya KR, Sultan MT, Mail MF, Hua LS. Design and development of a robot for spraying fertilizers and pesticides for agriculture. Materials Today: Proceedings. 2023 Jan 1;81:242-8.

[2] Richardson M, Wallace S. Getting started with raspberry PI. " O'Reilly Media, Inc."; 2012 Dec 10.

[3] Kumar PD, Suhasini A, Anand D. Crop disease detection using 2d cnn based deep learning architecture. International Journal of Intelligent Systems and Applications in Engineering. 2023 Feb 17;11(2):461-70.

[4] Teixeira AC, Ribeiro J, Morais R, Sousa JJ, Cunha A. A Systematic Review on Automatic Insect Detection Using Deep Learning. Agriculture. 2023 Mar 19;13(3):713.

[5] Vasanth N, Akash G, Srikanth KR, Pavan ST, Sinha R. Solar powered automatic pesticides sprayer. In2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS) 2017 Aug 1 (pp. 3438-3441). IEEE.

[6] Vallejo-Gómez D, Osorio M, Hincapié CA. Smart Irrigation Systems in Agriculture: A Systematic Review. Agronomy. 2023 Jan 25;13(2):342.

[7] Chohan M, Khan A, Chohan R, Katpar SH, Mahar MS. Plant disease detection using deep learning. International Journal of Recent Technology and Engineering. 2020 May;9(1):909-14.

[8] Petrov N, Dobrilovic D, Kavalić M, Stanisavljev S. Examples of raspberry pi usage in internet of things. InProceedings of 6th International Conference on Applied Internet and Information Technologies AIIT2016 2016 Jun (pp. 03-04).

Real-time Plant Disease Detection and Pesticide Application System Using IoT and CNN

Vivek Kumar Verma
Department of Electrical and Electronics
A.B.E.S. Engineering College
Ghaziabad, India
vivek.verma@abes.ac.in

Abhishek Kumar Gupta
Department of Electrical and Electronics
A.B.E.S. Engineering College
Ghaziabad, India
abhishek.gupta@abes.ac.in

Aakash Kumar Singh
Department of Electrical and Electronics
A.B.E.S. Engineering College
Ghaziabad, India
aakash.21b0213006@abes.ac.in

Ashmit Raghav
Department of Electrical and Electronics
A.B.E.S. Engineering College
Ghaziabad, India
ashmit.20b0211079@abes.ac.in

Bhanu Pratap Singh
Department of Electrical and Electronics
A.B.E.S. Engineering College
Ghaziabad, India
bhanu.20b0211033@abes.ac.in

Anivesh Kumar Singh
Department of Electrical and Electronics
A.B.E.S. Engineering College
Ghaziabad, India
anivesh.21b0213005@abes.ac.in

***Abstract*—Globally, farmers use pesticides for improved crop yield and quality. The manual method of applying pesticides cause fatigue, harm to farmers health and application of pesticides in an indefinite proportion on the crops may be less or more in quantity then requirement. This project is aimed to build a 2-D CNN model using deep learning algorithm for plant disease detection and also this research paper outlines the idea of integrating ESP-32 microcontroller with the plant disease detection model to automate the process of pesticide dispensing. This will use image recognition model to identify the damaged crop. By using this technique, the time and work of the farmer will be reduced.**

Keywords— ESP32, CNN- Convolutional Neural Networks, Deep Learning

I. INTRODUCTION

A lot of challenges are there in agriculture which happens to be one of the most important pillars of our food supply. There is an ever-increasing world's population that demands more foods while conserving the environment. Fortunately, this is where technology helps in. This research is concerned with imaging plants pests using a machine and then spraying these plants with pesticides by an automatic devices. Such machines have a possibility of changing farming by bringing efficiency and the environmentally friendly way of doing things. Farmers have utilized the application of pesticide for a lengthy period to prevent their crops' exposure to pests and diseases. But here's the problem: at times they overuse, and this is harmful for the environment [1]. At other instances, they use less than sufficient quantities, which damages the crops as well.

This research paper develops the idea of using 2-D Convolutional Neural Networks model using deep learning algorithm which identifies the plant diseases [2] [3]. Further, this model is integrated with ESP-32 microcontroller to design a bot which dispense pesticide over the infected plant.

The use of 2-D CNN based deep learning model with ESP-32 microcontroller have an added advantage that it is a low power operation to precisely identify the plant disease and give instructions in real-time to apply pesticide over the infected area of the plant.

A. Research Motivation

The motivation behind this research is to make farming practices safer for the farmers and for the consumers also who consume food produced using old traditional methods. This project helps the farmers in producing food without harming the environment. By this project we are intending to develop such a technology that will use deep learning algorithms in such a way that only the affected part of the crop will get the pesticide ensuring in enhancement of growth for the crop and at the same time makes the environment clean.

II. LITERATURE REVIEW

In our study, it was found that extensive research has been carried out in this field as agriculture is a major part of economy. So, to protect crops from unwanted insects and disease the role of pesticides come into play. But the use of conventional method i.e., farmer using backpack type sprayer promotes uneven application of pesticides on crops which proven to be hazardous for crops. By carrying these heavy sprayers, they cause fatigue to the farmers [4] [5].

This project uses CNN with ESP-32 which is a new concept as the older techniques doesn't use both of them together [6]. This will help the farmer to do farming in economic way as this is a low power operation method and can be used where the condition is complex.

The previous models use a camera to capture the plant image and the manpower is required to take the decision whether plant is infected or not by seeing the images and then they decide to spray . But this model automate this process, where the camera takes the picture and the deep learning model identify whether the plant is diseased or not, if the plant is having some kind of disease then this model will automatically spray over the infected area of the plant.

As we can see from the below table. 1., there is lot of wastage of money and labor required is more in conventional method. The previous research shows that there are new technologies that are automating the pesticide spraying process but no one had used ESP-32, also there is lack of level indicator sensor in the dispenser.

Description	Conventional Method (10 Acre)	Automatic Pesticides Spraying Machine (10 Acre)
Pesticides	5Litre	4Litre
Pesticides cost	5*2000=10000 Rs	4*2000=8000 Rs
Labor Charge	Rs. 300 / Day	Rs. 300 / Day
Nos. of labors	3	1
Nos. of working days	8	2
No of cycle	5	5
Total charges of labor	300*3*8*5=36000Rs	300*1*2*5=3000 Rs
Machine cost	3000*3=9000	35000
Working charges of battery	-	Rs. 120/day For 5 no of cycle =Rs 600
Total cost	10000+36000+9000=55000	8000+3000+35000+600 =46600
We save Rs.8400/ year		

Table. 1. Comparison between conventional method and automated pesticide dispenser method [7]

III. Description of The Proposed System

The project has robotic vehicle, the movement of which is controlled by ESP-32 micro-controller.

A. *ESP-32 Micro-controller*

This is the heart of the system it controls the movement of the robotic vehicle, the pesticide dispenser, and the servo motor for spray direction. With the help of sensors like GPS and ultrasonic sensor allows ESP-32 in decision making. This help increasing efficiency and accuracy.

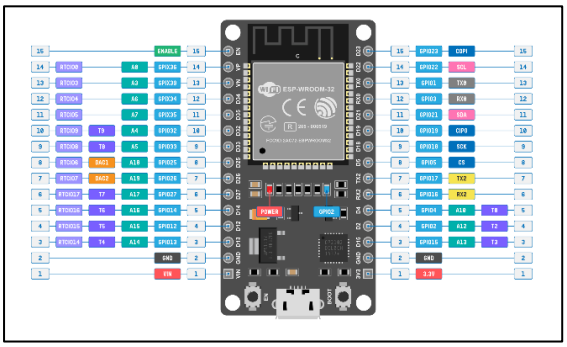


Fig. 1. Pin Diagram of ESP-32

The prime features of the ESP-32 micro-controller is that it has inbuilt Wi-Fi and Bluetooth for communication and internet access [8].

B. *Robotic Vehicle*

- **Frame:** It is the chassis part on which clamps for motor, supporting plate, vertical supports for the nozzles, battery, pump, and tank are attached. It is made up of mild steel. Its specifications are as follows: Length: 46 cm; Width: 30 cm.
- **Motor**: We can use DC motors to rotate the wheels at 40 rpm. These motors are powered by DC battery.
- **Battery**: The power required to run the proposed robotic vehicle have rating of 12v and 9amp DC.

C. *Pump*

The pump that can be used in the proposed project is powered by 12v DC supply. The pump is started when it gets command from ESP-32 microcontroller.

D. *Transmitter Block Diagram*

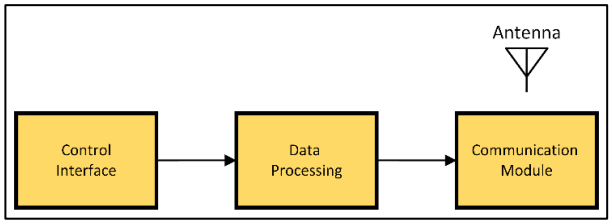


Fig. 2. Block Diagram of Transmitter

1) Control Interface

User interacts with app to send movement, spray parameters and other command.

2) Data Processing

Software interprets commands and formats data for transmission.

3) Communication Module

Sends data to receiver using antenna [9].

E. *Receiver Block Diagram*

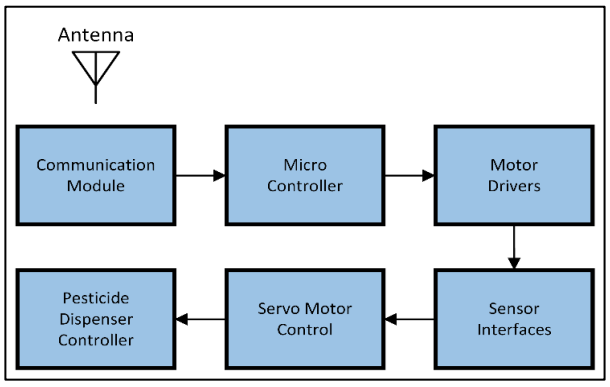


Fig. 3. Block Diagram of Receiver

1) Communication Module

Receives data from transmitter which further fetched by microcontroller.

2) Microcontroller

Decodes commands and send signals to motor drivers, sensor interfaces, servo motor controller, and pesticide dispenser controller.

3) Motor Drivers

Regulate power to motors for movement and steering [9].

4) Sensor Interfaces

Receive and process data from sensors like GPS, line sensors, and ultrasonic sensors.

5) Servo Motor Control

Adjusts servo motor position based on commands for targeted pesticide application.

6) Pesticide Dispenser Controller

Controls the flow of pesticide based on commands and sensor data.

IV. Methodology

A. Problem Definition

This research project intends to propose a CNN based image recognition system through ESP32 microcontroller. The aim is to find out what kind of pests there are as well as the prevalent diseases or crop conditions to apply pesticides appropriately in agriculture. This section provides objectives for the image recognition task and enumerates the recognized targets.

B. Convolutional Neural Networks

It is a type of neural network designed to classify data as they are good classifier. In this project we use 2D CNN as it is particularly designed for image data which has two-dimensional grid like structure. The main operation of this model is to filter the layers across the input image using convolution and calculating the dot products, this helps in extracting features [10]. Pooling is done to sample down the features maps which results in reducing the data size. To train this model, techniques like backpropagation and optimization are used to update the network weights in accordance with loss function.

CNNs are especially effective at image classification because they are able to automatically learn the spatial hierarchies of features, such as edges, textures, and shapes, which are important for recognizing objects in images.

C. Structure of CNN

As seen in Fig. 1, it consists of three layers: convolutional, pooling, and a fully linked layer. It is a kind of neural networks that analyzes data with a structure resembling a grid. The CNN building piece that is primarily in charge of computation is the convolution layer. Pooling minimizes the number of calculations needed while also decreasing the spatial size of the representation. On the other hand, the Fully Connected Layer is linked to the previous and most recent layers.

The convolution layer can be formulated by [11]:

$$x_j^l = g\left(\sum_{i \epsilon M_j} x_i^{l-1} * k_{ij}^l + b_j^l\right) \quad \text{... (1)}$$

In the above formula, b_j^l denotes a term and g(.) an activation function; the hyperbolic tangent function, which has a range of [-1,1], or the Sigmoid function are frequently used in neural networks.

The pooling layer can be calculated by the given formula:

$$x_j^l = down(x_i^{l-1}) \quad \text{... (2)}$$

The loss function calculates a neural network model's effectiveness in completing a given job. The loss function is shown in the formula below:

$$J(\theta) = -\frac{1}{N}\left(\sum_{i=1}^{N}\sum_{i=c}^{C} 1\{y_i = t\} log \frac{e^{\theta_t^T x_i}}{\sum_{l=1}^{T} e^{\theta_t^T x_i}}\right) \quad \text{... (3)}$$

In this case, 1{.} indicates an indicative function; the value equals 1 only in the event that the text's i^{th} image was accurate.

D. Image Classification

1) Pre-Processing

The captured image is firstly pre-processed i.e., resized, normalized, and converted to appropriate format as per CNN.

2) Covolutional layers

It is the main block of CNN. There are layers through which the image is passed. The layer performs separate operations, there tasks is to apply filters to the image. Each filter convolves with the image and creates an activation map [12].

3) Pooling Layer

This layer function is to reduce the size of feature map, enhance feature invariance, control the model complexity and encapsulate the features in the region covered by the filter that is applied by convolution layer as in [13].

4) Fully Connected Layer

Their function is to learn between the features and ultimately output the class probabilities [14].

5) Training

An extensive collection of labelled images is used to train CNN. To lessen the discrepancy between the actual and anticipated class labels, the network learns by modifying its weights and biases.

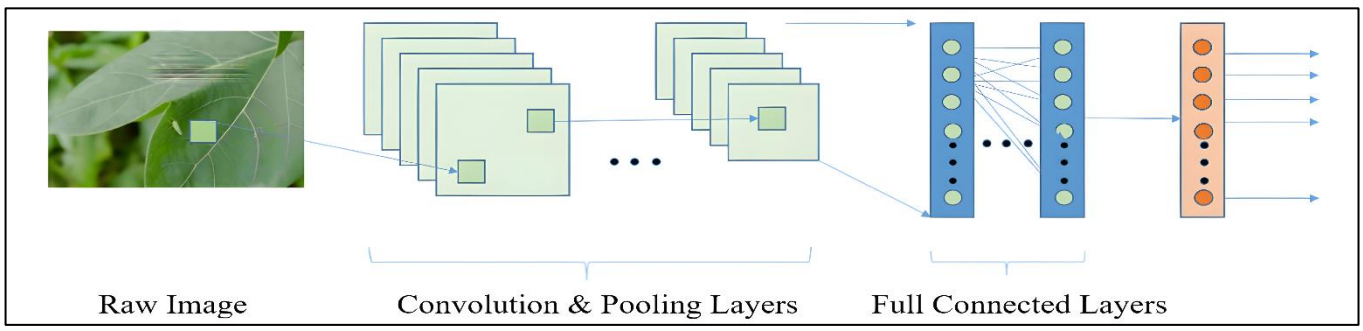


Fig. 4. Structure of CNN [11]

6) Evaluation

After the model is trained, a thorough assessment is carried out. When the dataset testing is done then the performance of the model is evaluated on parameters like accuracy, precision, recall and also the computation of F1 score.

E. Deployment on ESP-32

When the evaluation is completed, the CNN model is converted into a format that can be easily integrated with ESP-32 application. This format is TensorFlow lite [15]. After integration it enables real time image recognition.

F. Hardware Setup

A camera is connected to ESP32 microcontroller to take the image of the pest affected area. This helps in image acquisition for recognition purposes. Configuration of the ESP32 microcontroller is undertaken to manage image processing and interfacing with the deep learning model.

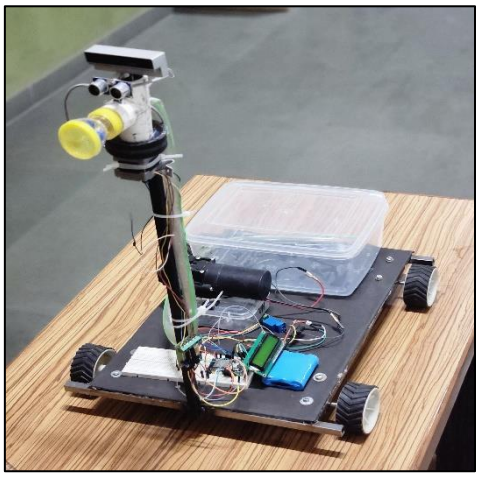


Fig. 5. Hardware Model

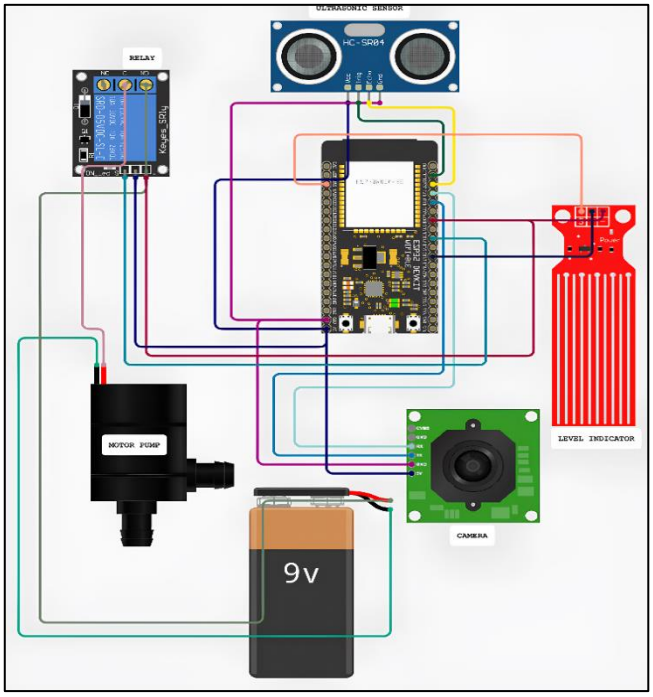


Fig. 6. Circuit Diagram of the model

G. Real-time Image Recognition

This system is enabled through the ESP32 microcontroller application code. This system is set to detect pests, diseases, and crop condition, giving essential input concerning decision on pesticide percentage that is to be used.

H. Testing and Validation

It undergoes testing and verification in simulated conditions as well as on-farm situations. This paper evaluates whether automatic pesticide dispensing system makes correct diagnosis and administration in regard to reliability, accuracy, and precision.

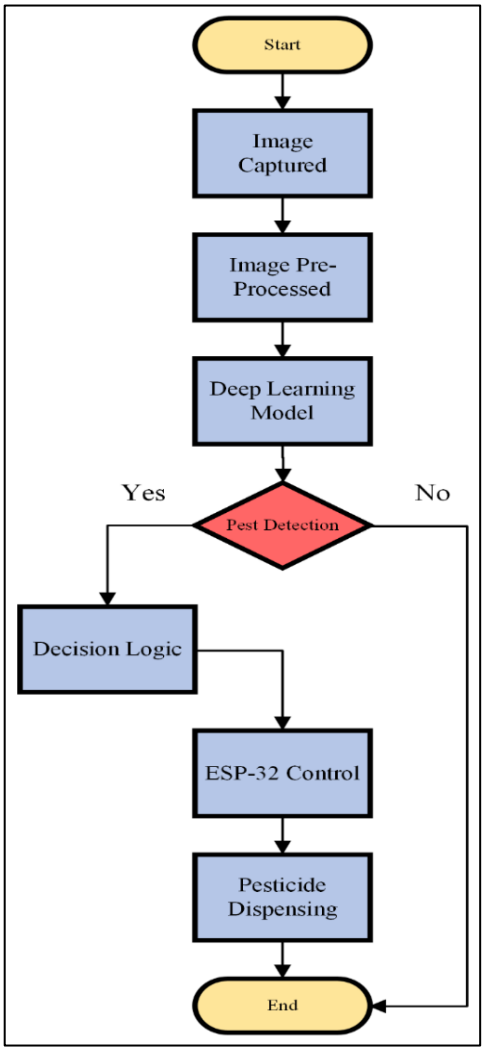


Fig. 7. Flowchart of the model

Firstly, the process will start when the camera fitted will capture the image of the plant. The captured image is then pre-processed after which the CNN model of deep learning is selected which will help in extracting the feature of the pl leaf. When all the features of the plant leaf are extracted then according to the data extracted, the deep learning model will decide whether the plant leaf is damaged or in good condition. If the plant leaf is in good condition, then the process will instantly end. On the other hand, if the plant leaf is damaged then the Esp-32 give instructions to the pesticide dispenser and the dispenser will spray on the affected part of the plant.

V. Experiments And Results

In this project, the motor is fed the DC current from the battery, this battery can also be used to power up the robotic vehicle and all the components. At the same time, the camera captures the image in real time and the model build using 2-D CNN predicts that whether the plant is having disease or not and if the plant is infected then the automated model will spray pesticides on the affected region.

To measure the accuracy of the CNN model we have plotted the accuracy and loss graph at the given epoch as shown in fig. 9 and fig. 10.

A. About Dataset

To train our model we have taken the data set which contains about 86K RGB images of both healthy and affected crop leaves. This data set is categorized in 36 non-identical classes. This dataset is divided into 80/20 ratio in order to train and test the dataset to check the accuracy of the model.

B. Results and Output

For calculating the accuracy of the system, we tested the system with different samples of potato, tomato and corn plant leaves with healthy and infected conditions as shown in fig. 8.

When the training of the model is completed, we got the final validation accuracy of 88.48% for the sample plant taken into consideration which is positive as the model indicates the accuracy is increasing when the model is evaluating the results again and again at given epoch.

The plotted graph as shown in fig. 9. puts light on the fact that each time the accuracy is increasing when the model is training on the dataset.

C. Implementation of result of CNN model with ESP-32 microcontroller

Now, when the camera which is used in the proposed model will capture the image and fed images to the 2D CNN trained model in real time which will help to identify the diseased crop and ESP-32 is provided with the data and it will turn on the pump and the infected plant will get the pesticides in appropriate quantity.

The CNN model when optimized with ESP-32 achieved an accuracy of 85.64%, there is a small decrease when compared to its training accuracy which was 88.48% on the PC. When the system is powered on, it takes few seconds to capture an image by the camera then pre-process it, perform inference with the CNN model, and generate a classification result. This processing time ensures real-time decision-making capabilities for spraying activation.

The plant chosen to test the hardware model are potato, corn and tomato. There leaves are taken both in healthy and diseased condition.

The CNN and ESP32 integration work well in the case of tomato and potato leaves as the model is able to identify between the healthy and unhealthy leaves and sprayed over the unhealthy leaves. But in the case of corn leaves, the model unable to distinguish between the healthy and unhealthy leaves and the result received in this case was opposite.

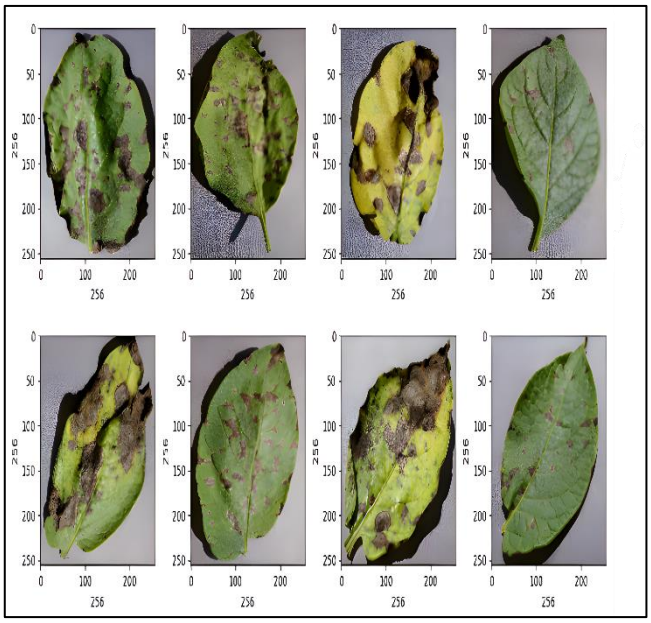


Fig. 8. Sample Plant Leaves

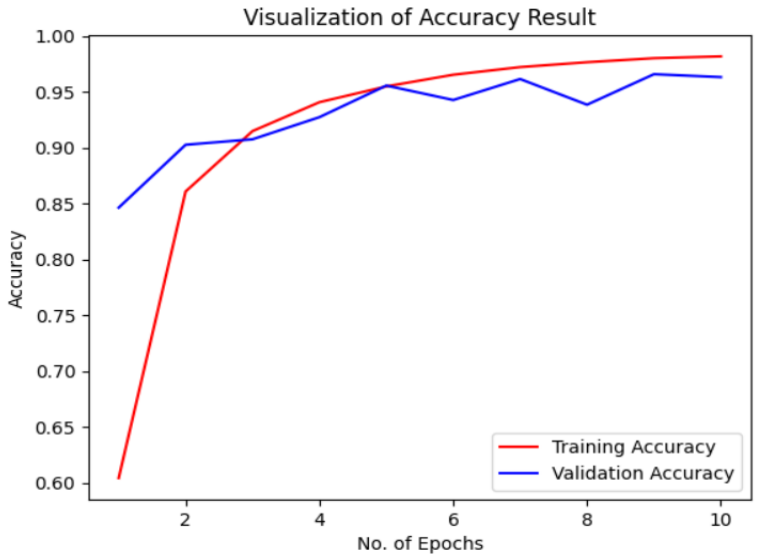


Fig. 9. Accuracy of Final Model

VI. Conclusion and Future Work

Smart Agricultural system is implemented as an independent agricultural solution to automate the process of spraying pesticide. It utilizes with immediate recognition of disease next to controlled spraying of pesticide. Also, labor cost will be reduced. Both the farmer's health and the principles of resource management were taken into consideration when developing this system. In future, solar panel can be used to charge battery which will end the dependency on electricity other than this there are numerous plants on which research can be done to produce new datasets which will enhance the bot for larger no. of disease detection. This will bring advancement in agricultural methods which is least cost-effective solution.

Acknowledgment

With deep appreciation, we would like to thank ABES Engineering College, Ghaziabad, our guide Mr. Vivek Kumar Verma and our co-guide Mr. Abhishek Kumar Gupta for their invaluable assistance and direction during our research project. Their assistance was very important to the project's success.

References

[1] Ghafar AS, Hajjaj SS, Gsangaya KR, Sultan MT, Mail MF, Hua LS. Design and development of a robot for spraying fertilizers and pesticides for agriculture. Materials Today: Proceedings. 2023 Jan 1;81:242-8.

[2] Kumar PD, Suhasini A, Anand D. Crop disease detection using 2d cnn based deep learning architecture. International Journal of Intelligent Systems and Applications in Engineering. 2023 Feb 17;11(2):461-70.

[3] Teixeira AC, Ribeiro J, Morais R, Sousa JJ, Cunha A. A Systematic Review on Automatic Insect Detection Using Deep Learning. Agriculture. 2023 Mar 19;13(3):713.

[4] Vasanth N, Akash G, Srikanth KR, Pavan ST, Sinha R. Solar powered automatic pesticides sprayer. In2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS) 2017 Aug 1 (pp. 3438-3441). IEEE.

[5] Vallejo-Gómez D, Osorio M, Hincapié CA. Smart Irrigation Systems in Agriculture: A Systematic Review. Agronomy. 2023 Jan 25;13(2):342.

[6] Chohan M, Khan A, Chohan R, Katpar SH, Mahar MS. Plant disease detection using deep learning. International Journal of Recent Technology and Engineering. 2020 May;9(1):909-14.

[7] Kumbhare DN, Singh V, Waghmare P, Ansari A, Tiwari V, Gorle RD. Fabrication of automatic pesticides spraying machine. International Research Journal of Engineering and Technology (IRJET). 2016 Apr;3(4).

[8] Hercog D, Lerher T, Truntič M, Težak O. Design and implementation of ESP32-based IOT devices. Sensors. 2023 Jul 27;23(15):6739.

[9] Cameron N. Remote Control an ESP32-CAM Robot Car. InESP32 Formats and Communication: Application of Communication Protocols with ESP32 Microcontroller 2023 Jul 1 (pp. 577-603). Berkeley, CA: Apress.

[10] Taye MM. Theoretical understanding of convolutional neural network: Concepts, architectures, applications, future directions. Computation. 2023 Mar 6;11(3):52.

[11] Cheng X, Zhang Y, Chen Y, Wu Y, Yue Y. Pest identification via deep residual learning in complex background. Computers and Electronics in Agriculture. 2017 Sep 1;141:351-6.

[12] Yamashita R, Nishio M, Do RK, Togashi K. Convolutional neural networks: an overview and application in radiology. Insights into imaging. 2018 Aug;9:611-29.

[13] Özdemir C. Avg-topk: A new pooling method for convolutional neural networks. Expert Systems with Applications. 2023 Aug 1;223:119892.

[14] Basha SS, Dubey SR, Pulabaigari V, Mukherjee S. Impact of fully connected layers on performance of convolutional neural networks for image classification. Neurocomputing. 2020 Feb 22;378:112-9.

[15] David R, Duke J, Jain A, Janapa Reddi V, Jeffries N, Li J, Kreeger N, Nappier I, Natraj M, Wang T, Warden P. Tensorflow lite micro: Embedded machine learning for tinyml systems. Proceedings of Machine Learning and Systems. 2021 Mar 15;3:800-11.

Appendix-I

Coding

Code for Motor Pump Operation:

```
import cv2

import numpy as np

import tensorflow as tf

import RPi.GPIO as GPIO

import time


# Load the TensorFlow Lite model for plant disease detection

interpreter = tf.lite.Interpreter('converted_model.tflite')

interpreter.allocate_tensors()


# Set up the camera

cap = cv2.VideoCapture(0)  # Use 0 for the default camera


# Set up the GPIO pins

PUMP_PIN = 4

TRIG_PIN = 20

ECHO_PIN = 21


GPIO.setmode(GPIO.BCM)

GPIO.setup(PUMP_PIN, GPIO.OUT)

GPIO.setup(TRIG_PIN, GPIO.OUT)
```

```
GPIO.setup(ECHO_PIN, GPIO.IN)

# Define a function to preprocess the image
def preprocess_image(image):
    image = cv2.resize(image, (224, 224))
    image = image.astype(np.float32) / 255.0  # Normalize pixel values
    return np.expand_dims(image, axis=0)

# Function to start the motor pump
def start_motor_pump():
    GPIO.output(PUMP_PIN, GPIO.HIGH)  # Turn on the motor pump
    time.sleep(5)  # Spray for 5 seconds
    GPIO.output(PUMP_PIN, GPIO.LOW)  # Turn off the motor pump

# Function to measure distance using ultrasonic sensor
def measure_distance():
    # Send a 10us pulse to TRIG_PIN
    GPIO.output(TRIG_PIN, GPIO.HIGH)
    time.sleep(0.00001)
    GPIO.output(TRIG_PIN, GPIO.LOW)

    # Measure the pulse duration on ECHO_PIN
    while GPIO.input(ECHO_PIN) == 0:
        pulse_start = time.time()
```

```
    while GPIO.input(ECHO_PIN) == 1:
        pulse_end = time.time()

    pulse_duration = pulse_end - pulse_start

    # Calculate the distance in cm
    distance = pulse_duration * 17150
    distance = round(distance, 2)
    return distance

# Main loop for live plant disease detection
while True:
    distance = measure_distance()
    if distance < 20:  # Check if the plant is within 20 cm range
        ret, frame = cap.read()
        if not ret:
            print("Error: Failed to capture frame from camera.")
            break

        # Preprocess the image
        input_data = preprocess_image(frame)

        # Set the input tensor
        input_details = interpreter.get_input_details()
        interpreter.set_tensor(input_details[0]['index'], input_data)
```

```
# Run inference
interpreter.invoke()

# Get the output tensor
output_details = interpreter.get_output_details()
output_data = interpreter.get_tensor(output_details[0]['index'])

# Get the class index with the highest probability
max_index = np.argmax(output_data)
# Get the corresponding class name
class_name = {
    0: 'Apple___Apple_scab',
    1: 'Apple___Black_rot',
    2: 'Apple___Cedar_apple_rust',
    3: 'Apple___healthy',
    4: 'Blueberry___healthy',
    5: 'Cherry_(including_sour)_Powdery_mildew',
    6: 'Cherry_(including_sour)_healthy',
    7: 'Corn_(maize)_Cercospora_leaf_spot Gray_leaf_spot',
    8: 'Corn_(maize)Common_rust',
    9: 'Corn_(maize)_Northern_Leaf_Blight',
    10: 'Corn_(maize)_healthy',
    11: 'Grape___Black_rot',
    12: 'Grape__Esca(Black_Measles)',
```

13: 'Grape__Leaf_blight(Isariopsis_Leaf_Spot)',

14: 'Grape___healthy',

15: 'Orange__Haunglongbing(Citrus_greening)',

16: 'Peach___Bacterial_spot',

17: 'Peach___healthy',

18: 'Pepper,bell__Bacterial_spot',

19: 'Pepper,bell__healthy',

20: 'Potato___Early_blight',

21: 'Potato___Late_blight',

22: 'Potato___healthy',

23: 'Raspberry___healthy',

24: 'Soybean___healthy',

25: 'Squash___Powdery_mildew',

26: 'Strawberry___Leaf_scorch',

27: 'Strawberry___healthy',

28: 'Tomato___Bacterial_spot',

29: 'Tomato___Early_blight',

30: 'Tomato___Late_blight',

31: 'Tomato___Leaf_Mold',

32: 'Tomato___Septoria_leaf_spot',

33: 'Tomato___Spider_mites Two-spotted_spider_mite',

34: 'Tomato___Target_Spot',

35: 'Tomato___Tomato_Yellow_Leaf_Curl_Virus',

36: 'Tomato___Tomato_mosaic_virus',

37: 'Tomato___healthy'

```
        }

        class_name = class_name[max_index]

        # Check if the class name includes "healthy"
        if "healthy" in class_name.lower():
            output = "healthy"
        else:
            output = "diseased"
            # Start the motor pump
            start_motor_pump()

        print("Output:", output)

        # Display the frame
        cv2.imshow('Plant Disease Detection', frame)

    # Check for key press to exit
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

# Release resources
cap.release()
cv2.destroyAllWindows()
GPIO.cleanup()
```

Code for Remote Controlled Operation:

```
#define BLYNK_PRINT Serial
#define BLYNK_TEMPLATE_ID "TMPL3oaotcwxp"
#define BLYNK_TEMPLATE_NAME "Pesticide robot"
#define BLYNK_AUTH_TOKEN "Ci5CGTZwutcRMRbqfO7sKtk3RWfRbK-V"
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
char auth[] = BLYNK_AUTH_TOKEN;

char ssid[] = "Aakash's S22";  // type your wifi name
char pass[] = "12345678";      // type your wifi password

int M1PWM = 14;  // PWM pin for motor speed control
int M1F = 13;    // Forward control pin
int M1R = 12;    // Reverse control pin
int pinValue1;   // Motor speed value from Blynk

BLYNK_WRITE(V1) {
  pinValue1 = param.asInt();      // Read the motor speed value from Blynk
  analogWrite(M1PWM, abs(pinValue1));  // Set motor speed (0-255)

  // Determine direction based on pinValue1
  if (pinValue1 > 0) {
    digitalWrite(M1F, HIGH);  // Set forward pin HIGH
    digitalWrite(M1R, LOW);   // Set reverse pin LOW
```

```
  } else if (pinValue1 < 0) {
    digitalWrite(M1F, LOW);   // Set forward pin LOW
    digitalWrite(M1R, HIGH);  // Set reverse pin HIGH
  } else {
    digitalWrite(M1F, LOW);   // Stop the motor
    digitalWrite(M1R, LOW);   // Stop the motor
  }
  Blynk.virtualWrite(V1, pinValue1);  // Update Blynk slider value
  Serial.print("V1 Slider Value is ");
  Serial.println(pinValue1);
}
void setup() {
  pinMode(M1PWM, OUTPUT);
  pinMode(M1F, OUTPUT);
  pinMode(M1R, OUTPUT);

  Serial.begin(9600);
  Blynk.begin(auth, ssid, pass);
}

void loop() {
  Blynk.run();
}
```

pesticide

ORIGINALITY REPORT

7%	4%	2%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to ABES Engineering College Student Paper	1%
2	1library.net Internet Source	1%
3	www.coursehero.com Internet Source	1%
4	biblio.ugent.be Internet Source	1%
5	Submitted to KIET Group of Institutions, Ghaziabad Student Paper	<1%
6	"Information and Communication Technology for Competitive Strategies (ICTCS 2020)", Springer Science and Business Media LLC, 2022 Publication	<1%
7	Haifa Ghabri, Mohammed S. Alqahtani, Soufiene Ben Othman, Amal Al-Rasheed et al. "Transfer Learning for Accurate Fetal Organ Classification from Ultrasound Images: A	<1%

	Potential Tool for Maternal Healthcare Providers", Research Square Platform LLC, 2023 Publication	
8	etd.aau.edu.et Internet Source	<1 %
9	www.slideshare.net Internet Source	<1 %
10	Submitted to University of Sunderland Student Paper	<1 %
11	digitallibrary.usc.edu Internet Source	<1 %
12	Luca Catarinucci, Danilo De Donno, Luca Mainetti, Luigi Patrono, Maria Laura Stefanizzi, Luciano Tarricone. "An IoT-aware Architecture to improve Safety in Sports Environments", Journal of Communications Software and Systems, 2017 Publication	<1 %
13	tuprints.ulb.tu-darmstadt.de Internet Source	<1 %
14	Submitted to German University of Technology in Oman Student Paper	<1 %
15	www.mdpi.com Internet Source	<1 %

16	Submitted to Sankalchand Patel University Student Paper	<1 %
17	Submitted to The Kyoto College of Graduate Studies for Informatics Student Paper	<1 %
18	Submitted to University of Auckland Student Paper	<1 %
19	Kola Balavani, Dasari Sriram, Malla Bhavani Shankar, Devavarapu Sri Charan. "An Optimized Plant Disease Classification System Based on Resnet-50 Architecture and Transfer Learning", 2023 4th International Conference for Emerging Technology (INCET), 2023 Publication	<1 %
20	Submitted to University of Perpetual Help Sytem Laguna Student Paper	<1 %
21	Submitted to University of Technology, Sydney Student Paper	<1 %
22	Submitted to Curtin University of Technology Student Paper	<1 %
23	Submitted to The University of the West of Scotland Student Paper	<1 %
24	Submitted to University of Glasgow Student Paper	

		<1 %
25	Submitted to University of Sydney Student Paper	<1 %
26	Submitted to National School of Business Management NSBM, Sri Lanka Student Paper	<1 %
27	jums.academy Internet Source	<1 %

Exclude quotes	On	Exclude matches	< 10 words
Exclude bibliography	On		