Real Time Detection of plant disease by Leaf Image Using Raspberry pi and Convolutional Neural Networks

[1]D. Shivadatha Reddy, [2]Syed Jahangir Badashah , [3]Vikram Palodiya

[1,2,3-]Department of Electronics and Communication Engineering (DSCE), Sreenidhi Institute of Science and Technology, Hyderabad, India.

[1-]19311d0610@sreenidhi.edu.in, [2-]sydjahangir@sreenidhi.edu.in , [3-]vikramp@sreenidhi.edu.in

Abstract:
Green plants are crucial to the human ecology because they provide the groundwork for the creation, upkeep, and long-term health of environmental systems. This highlights the critical need of growing a robust plant. The plant disease might be cured with early detection and treatment. In this post, we propose a system that uses Raspberry-pi to look for healthy and diseased crops, and then notifies farmers through email. We used the tensor flow program to do a numerical computation. Seeing that it can detect the early leaf indications of plant diseases, it is especially helpful in controlled environments, such as farms. This is the most popular kind of application.
Keywords: Real-time detection, Plant disease, Leaf image, Raspberry-pi, Sensors, Convolutional Neural Networks.

I. INTRODUCTION

In recent years, agriculture has come to include more than just growing crops for human and animal use. Because more and more plants are being utilized to produce electricity and other types of energy that improve people's standard of living, this is the case. However, many illnesses spread by plants have the potential to do significant damage to communities' economic and social systems. Worse worse, it may lead to significant ecological harm [1, 2]. Correct and quick illness diagnosis is preferred since it may prevent such losses. Plant diseases may be diagnosed by either humans or machines. Plant diseases are easily identifiable because of the telltale markings they leave on the leaves. However, there are a few illnesses that don't manifest themselves until they've already done considerable harm to the plants, and a few others that don't manifest themselves at all on the leaves.

The disease drastically reduces the plant's growth and productivity. This implies that we need to find ways to prevent the spread of plant diseases if we want to keep the world's population supplied. Determining the full scope of a plant disease has always been challenging. The severity of an illness was formerly only able to be determined by visual examination (bare eye observation) [3]. This approach requires continual field monitoring by subject matter experts

in order to provide reliable illness estimates. Because it requires regular human observations, visual analysis for vast plant areas is excessively costly, uncomfortable, and time-consuming.
Computer systems with complicated mathematical algorithms and analytical equipment, such as high-powered microscopes and other devices, are the only ones that have a chance of spotting the issue in time in such a situation, according to the vast majority of specialists [4]. In other circumstances, electromagnetic techniques, which produce more tiny pictures, are required to spot the indications.

[14-16] Academic research that makes use of image processing methods has tremendously aided the growth of the agriculture industry. However, many scientists use a wide variety of approaches to detect, measure, and classify illnesses affecting plant leaves. Some of the methods that will be used are Bounding Box analysis, Moment Analysis, Color Analysis, Support Vector Machines, and Neural Networks. However, none of the methods used by the many studies have been very compelling. [5].The Triangular Thresholding method is used in this methodology to identify, evaluate, and quantify the extent of fungal illness on leaf surfaces. This method stands out from the others because of its ease of use, general applicability, and pinpoint accuracy.

[15] Many different kinds of pests and diseases may cause damage to plants at any time. In order to protect themselves against predators and other dangers, plants have developed a vast range of defensive mechanisms during the course of evolution.The contact between a plant and a disease may result in either a vulnerable (compatible) or a resistant (incompatible) plant or pathogen. [6]. In the pursuit of fun, a contagion quickly spreads. In contrast, in an incompatible relationship, the plant will respond in a way that hinders the pathogen's growth and/or harm. As an authentic part of the hypersensitive response, an oxidative burst kills cells locally, effectively "trapping" the pathogen between the layers of dead tissue. Reorganization of cells and reinforcement of the cell wall might be triggered by signals from inside or from the invading pathogen. [16]. These changes lead to an increase in cell wall proteins such as extensions, glycine-rich proteins, proline-rich proteins, solanaeeouslectins, and arabinogalactan proteins. Enzymes that target certain wail polymers may be activated. This includes suberin, lignin, callose, and wall-bound phenolic. [8].Thus, the cell wall is not merely a static barrier to prevent the entrance of pest organisms into the plant cell, but rather an active defensive reaction.

Among the chemical defenses that plants develop, the de novo synthesis of defense-related proteins plays a crucial role. A wide variety of defense-related proteins are either constitutively produced or generated in response to environmental stresses such as herbivore damage or microbial invasion. Therefore, as suggested by [9], these proteins establish barriers during and after an infection. Enzyme inhibitors, such as amylase and proteinase inhibitors, hydrolytic enzymes, such as 1, 3-P-glucanases and chattiness, and other related proteins are examples of antimicrobial proteins with a low molecular weight and high cysteine content. Antimicrobial substances such as oxidized phenolics, tannins, and other low molecular weight secondary metabolites (such as phytoalexins) are accumulated as part of a plant's chemical defense system. [10]

Antimicrobial proteins (AMPs) in plants are widespread, and many share properties with those already identified [11]. These proteins often have an even number of cysteine residues (4, 6, or 8) and are small in size (10 kDa). By generating intramolecular disulfide bonds, these cysteines help to ensure the structural and thermodynamic stability of the protein [13].

"Real-time detection of the plant diseases using deep learning techniques" by K. R. Singh, et al. This paper proposes a real-time plant disease detection system using convolutional neural networks (CNNs) and a Raspberry Pi. The system is able to detect multiple plant diseases from leaf images with high accuracy and speed.

"A comparative study of deep learning algorithms for plant disease detection" by A. Tripathi, et al. This paper presents a comparative study of several deep learning algorithms for plant disease detection, including CNNs, deep belief networks (DBNs), and stacked autoencoders (SAEs). The authors evaluate the algorithms on a dataset of tomato leaf images and conclude that CNNs perform the best.

"Leaf disease detection using convolutional neural networks and transfer learning" by S. K. Sharma, et al. This paper proposes a leaf disease detection system using CNNs and transfer learning. The authors train a pre-trained CNN model on a large dataset of natural images and fine-tune it on a smaller dataset of plant leaf images. The system achieves high accuracy in detecting several types of leaf diseases.\

"Design and development of an IoT-based plant disease detection system" by S. V. Karthik, et al. This paper presents an IoT-based plant disease detection system that uses a Raspberry Pi and a smartphone app. The system captures leaf images using the smartphone camera and sends them to the Raspberry Pi for analysis using a CNN model. The system is able to detect multiple plant diseases with high accuracy.

"Deep convolutional neural networks for tomato plant disease recognition" by K. R. Kavitha, et al. This paper proposes a deep CNN model for tomato plant disease recognition. The authors train the model on a large dataset of tomato leaf images and achieve high accuracy in detecting several types of leaf diseases. The model is designed to run on a Raspberry Pi for real-time disease detection.

II. METHODOLOGY

A. OBJECTIVE

1). To integrate DHT-11 (temperature and humidity) sensor, Soil moisture sensor and camera module with MCU (Microcontroller unit) along with communication module.

2). To develop a camera module interface with Raspberry pi 3B+ to update the status of sensors and to access the data and disease of plant.

3). To conduct laboratory investigation to check the functionality of the integrated system.

4). To design such a system that can detect crop disease and pests accurately.

5). Create a database of insecticides for respective pest and disease.

6). To provide remedy for the disease that detected.

7). Convolutional neural networks (CNNs)

B. PROPOSED SYSTEM

The proposed system uses a CNN algorithm to detect plant leaf disease. This is because using CNN allows us to achieve maximum accuracy when the dataset is good. This proposed system uses Raspberry Pi and a camera module to capture images and process them to predict the disease of a leaf and then the name of the disease. Here, the data set is obtained, the data is pre-processed before training and then later this data is trained. This is so because if your model is of this type, you can easily train your model to predict your model. The trained data are divided into two methods of validation and training and validation. Test data in 80:20 ratio. After training the data, a model is created to take pictures of the images using Raspberry Pi and predict the disease using the CNN algorithm and the trained model. This CNN can achieve 90% accuracy.

Advantages of Proposed System:

We can receive the real time data for continues intervals.

1). Low-cost system.

2). Easy to install and easy to operate.

C. Block Diagram of leaf disease

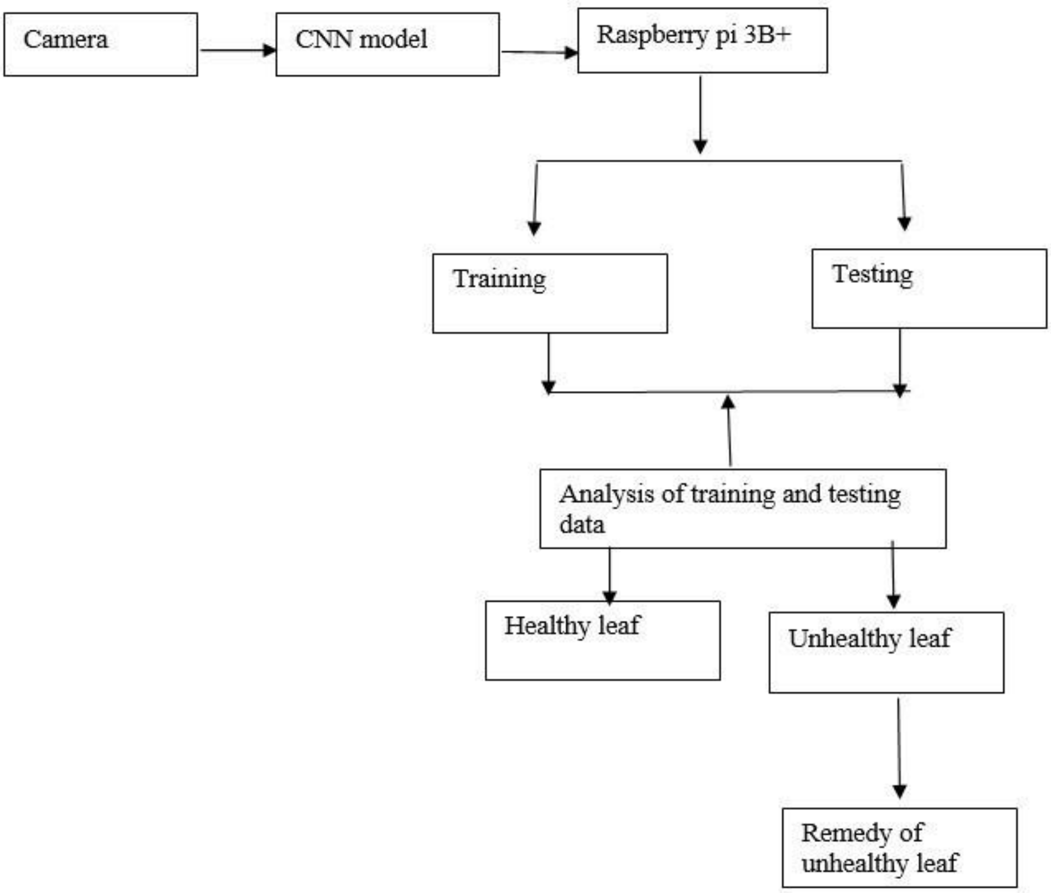

Figure 1: Proposed block diagram

D). Convolutional neural networks (CNNs)

a). One kind of Deep Neural Networks that has found widespread usage in computer vision applications is the Convolutional Neural Network (CNN). Image classification, object recognition, and segmentation are three areas where CNNs have shown the greatest promise.

b). The convolutional layers, pooling layers, and activation functions of a CNN are what make the technology operate with an input picture. Features like edges, textures, and forms are extracted from the input picture by applying a series of filters or kernels in the convolutional layers. The feature maps produced by the convolutional layers are down sampled by the pooling layers, decreasing the spatial dimensions while maintaining the most salient characteristics. The final output is produced by one or more fully linked layers, which receive the flattened output from the pooling layers.

c). In addition to learning low-level features, CNNs may also learn high-level features directly from raw picture data. This makes them especially helpful when dealing with input images that have complicated or diverse patterns, for example, where more typical computer vision algorithms may have difficulty.

d). In addition to image classification and object recognition, CNNs have also found usage in semantic segmentation. They have also found use in areas including driverless cars, language processing, and medical image analysis.

e). Overall, CNNs are a powerful tool for analysing image data and have revolutionized the field of Computer vision.

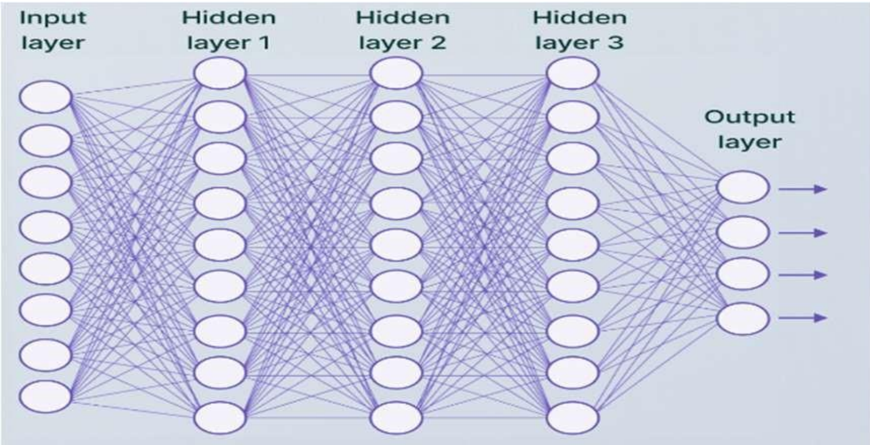

Figure 2: Convolutional Neural Networks (CNNs)

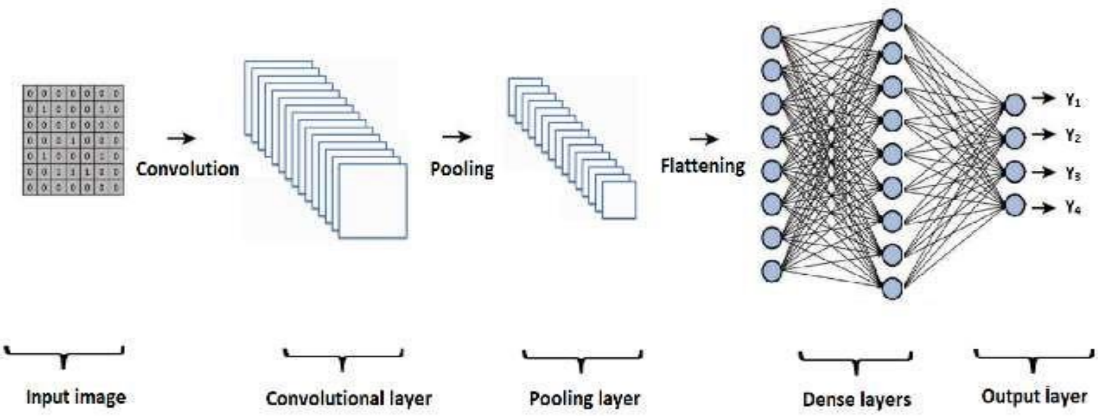

Figure 3: Internal layers of Convolutional neural networks (CNNs)

Description

1). *Input Layer:* The input layer is responsible for receiving the raw image data as input. The input layer is usually a 2D array of pixels of representing the image, with each pixel value to its corresponding to the color intensity of that pixel.
2). Convolutional Layers: The convolutional layers are the fundamental components of a convolutional neural network (CNN). Many elements, including edges, corners, and textures, are extracted from the input picture by applying a series of filters. A feature map is generated by sliding a series of filters, each of which is a tiny weighted matrix, over the input picture.
3). Pooling Layers: The feature maps generated by the convolutional layers are down sampled by the pooling layers. The max pooling layer, which takes the highest value in a narrow window of the feature map and discards the remainder, is by far the most used form of pooling layer.
4). Fully Connected Layers: In case of the fully connected layers whose job it is to make such predictions based on the characteristics they've gathered. They apply a set of weights on the outputs of the preceding layers to get the final result. The number of classes in a classification task is proportional to the number of neurons in the output layer.
5). Output Layer: The ultimate forecast is generated at the output layer of this kind of model. In classification problems, the output layer is usually a softmax layer that will produce a probability distributions over that possible classes.

D. Raspberry pi

1). The Raspberry-Pi Foundation of the United Kingdom created the Raspberry Pi line of inexpensive single-board computers. Its initial intent was to help spread introductory computer science education in disadvantaged communities and classrooms.
2). The Raspberry Pi board is a little computer the size of a credit card that can be used with a standard keyboard and mouse when connected to a TV or monitor. The CPU and GPU are both based on ARM and built within the Broadcom SoC. It also has various input/output ports, including USB, the Ethernet, HDMI, audio, and GPIO pins, which allow for a wide range of projects and applications.
3). Raspberry Pi has gained popularity due to its low cost, portability, and versatility. It is used in a many variety of applications, such as home automation, media centers, robotics, and education.
4). Machine learning and AI models, such as Convolutional Neural Networks (CNNs) for image identification and classification, may be run on it as well.

E. Image Processing

The basic steps involved in Image processing include:
a). *Image acquisition:* This step will involve in capturing the digital image using a camera or scanner.

b). *Pre-processing: Noise, contrast, and brightness may all be improved as well as distortion and other artifacts removed using pre-processing methods.*
c). *Enhancement:* In order to make a picture more appealing to the eye, enhancement methods are employed to boost its sharpness, correct its color balance, and boost its contrast.
d). *Restoration:* Restoration techniques are used to remove any distortion or noise that may be present in the image, such as blurring or scratches.
e). *Segmentation:* Images may be broken down into its constituent parts using segmentation methods by analyzing features like color and texture.
f). *Feature extraction:* Feature extraction techniques are used to identify and extract important features from the image, such as edges, corners, or textures.
g). *Object recognition:* Object recognition techniques are used to identify and classify objects in the image based on their features and characteristics.

Figure 4: Colour image to Gray scale
Conversion Process

The picture is a pixelated rectangle. It has a fixed width and height that are measured in pixels. For a particular screen, each pixel is a uniform square. However, the size of a pixel might vary from display to monitor. The image's pixels are laid up in columns and rows, creating a grid. The brightness and hue of each pixel are represented numerically.

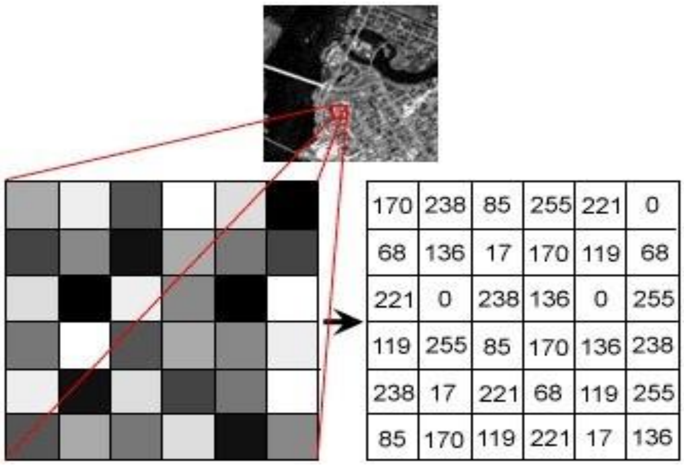

Figure 5: Gray Scale Image Pixel Value Analysis

Each pixel is assigned a color. Integers with 32 bits may be used to represent colors. The first eight bits designate the pixel's redness, the next eight bits its greenness, the following eight bits its blueness, and the last eight bits its transparency.

F. LIMITATIONS

a). Limited processing power: The Raspberry Pi has limited processing power compared to high-end computers, which can affect the speed and accuracy of the disease detection system.

b). Limited memory: The Raspberry Pi has limited memory, which can impact the size and complexity of the CNN model that can be used for disease detection.

c*). Limited camera resolution:* The camera module attached to the Raspberry Pi may have limited resolution, which can affect the quality and accuracy of the images captured for disease detection.

d). *Limited dataset:* The accuracy of the CNN model depends on the quality and quantity of the labelled dataset used for training. A limited dataset can result in lower accuracy and reliability of the disease detection system.

e). *Environmental factors:* The accuracy of the disease detection system can be affected by environmental factors such as lighting conditions, humidity, and temperature.

f). *Limited scope of disease detection:* The CNN model can only detect diseases that are present in the labelled dataset used for training. New diseases or variations of existing diseases may not be detected by the system.

g). *Dependency on internet connection:* If the system is designed to provide recommendations for treatment based on disease detection, it may require an internet connection to access a database of treatments, which may not be available in all locations.

h). *Limited user interface:* The GUI on the Raspberry Pi or connected device may have limited capabilities for displaying information, which can affect the usability and effectiveness of the system.

III. RESULTS & DISCUSSION

Design a prototype for capturing the movement of an object:

To design a prototype for capturing the movement of an object, you can follow these general steps:

a). Determine the scope and requirements:

Define the purpose of the prototype and identify the specific requirements, such as the type of object to be captured, the range of motion to be tracked, the accuracy and precision needed, and any environmental or operational constraints.

b). *Choose the sensing technology:* Select the appropriate sensing technology based on the requirements and constraints, such as cameras, accelerometers, gyroscopes, proximity sensors, or a combination of these. Consider factors such as resolution, speed, noise, and cost.

c). Select the processing hardware: Choose the processing hardware that can handle the data from the sensors and perform the necessary computations in real-time. This can range from a microcontroller to a more powerful processor, depending on the complexity of the algorithms and the data volume.

d). Develop the software: Write the software that will control the sensors, collect the data, and process it in real-time. This can involve programming in a language such as Python, C++, or MATLAB, and using libraries or frameworks for computer vision, machine learning, or signal processing.

e). *Prototype the system:* Assemble the sensing hardware and processing hardware into a prototype system, along with any necessary mechanical components such as mounts or enclosures. Test the system with various types of objects and motion patterns, and refine the algorithms as needed to improve accuracy and reliability.

f). *Evaluate the prototype:* Conduct a thorough evaluation of the prototype to assess its performance and identify any limitations or areas for improvement. This can involve comparing the results to ground-truth measurements or using statistical analysis to quantify the accuracy and precision.

g). *Refine and optimize:* Based on the evaluation results, refine the prototype design and software to optimize performance, reduce errors, and enhance usability. Iterate through the design and testing process until the prototype meets the desired specifications.

a) Hardware kit

1).*Sensor:* This could be a camera, accelerometer, gyroscope, or other sensing technology, depending on the type of object and motion being captured. For example, a high-speed camera could be used to track the movement of a ball, while an accelerometer could be used to measure the vibration of a machine.

2).*Processing hardware:* This could be a microcontroller, single-board computer (e.g. Raspberry Pi or Arduino), or more powerful processing hardware, depending on the complexity of the algorithms and the data volume. For example, a Raspberry Pi could be used for real-time image processing, while an FPGA could be used for high-speed signal processing.

3).*Power supply:* This could be a battery or a power adapter, depending on the application and portability requirements.

4).*Mechanical components:* This could include mounts, enclosures, or other mechanical components needed to secure the sensor and processing hardware in place and protect them from damage.

5*).Cables and connectors:* These are needed to connect the sensor, processing hardware, and power supply together.

6).Additional components: Depending on the specific requirements, additional components such as LEDs, buzzers, or motors may be needed to provide feedback or control the system.

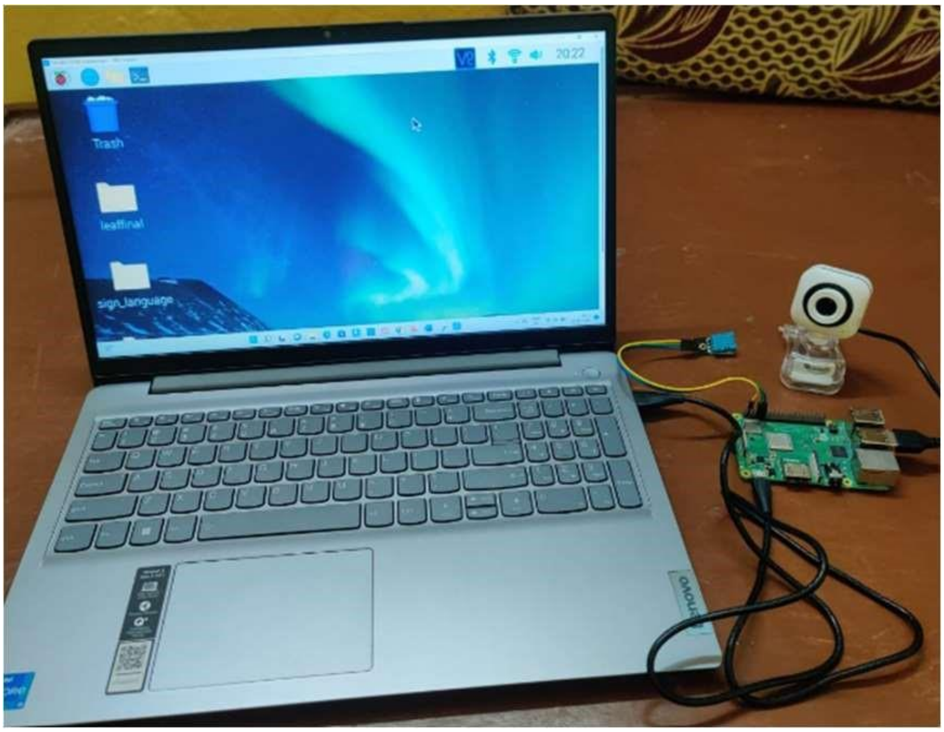

Figure 6: Hardware kit

Connections are given from the power supply to the dht11 sensor (it measures the temperature and humidity). And directly it will connect to the raspberry pi 3B+, dht11 sensor, and camera will connect after that it will predict the disease leaf.

b) To run the code:

1).*Set up the hardware:* Connect the sensing device and processing hardware together as per the hardware design. Make sure that all the necessary connections are made and powered.

2).*Install the required software:* Depending on the chosen sensing technology and processing hardware, you may need to install specific libraries, drivers, or programming environments on your computer or microcontroller. For example, if you are using Python for programming, you may need to install libraries like OpenCV, TensorFlow, or PySerial.

3).Write the code: Develop the code that will collect the sensor data and process it in realtime. This can involve programming in a language such as Python, C++, or MATLAB, and using libraries or frameworks for computer vision, machine learning, or signal processing.

4).*Compile the code:* If you are using a microcontroller or other embedded system, you may need to compile the code before uploading it to the hardware. This can involve using a compiler toolchain or an Integrated Development Environment (IDE).

5).*Upload the code:* Once the code is compiled, you can upload it to the hardware device using a programming cable or wireless connection. Make sure to follow the instructions provided by the hardware manufacturer or software libraries.

6).*Run the code:* After uploading the code to the hardware, you can run it and monitor the output. This can involve connecting to the hardware device via a serial port, Bluetooth, or Wi-Fi, and visualizing the data in real-time using a computer or mobile app.

7).*Debug and optimize:* If there are any issues or errors with the code, debug the code and optimize it as needed to improve performance, accuracy, and reliability.

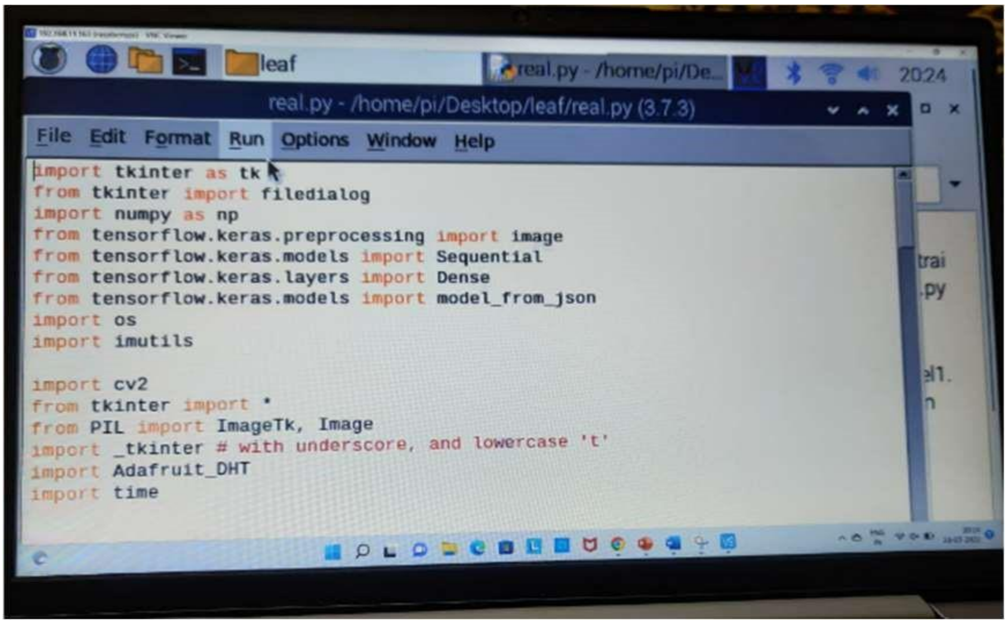

Figure 7: run the code.

c) Output Response:

1). *Sensor readings:* The DHT11 sensor measures temperature and humidity, so the output of the project could be a series of sensor readings over time. These readings could be displayed on a screen or stored in a database for further analysis.

2). *Motion tracking:* If the project is designed to track the movement of an object, the output could be a video or series of images showing the object's motion over time. This could be processed using computer vision algorithms to extract features such as speed and direction of motion.

3). *Alerts:* If the project is designed to monitor environmental conditions such as temperature and humidity, the output could be alerts or notifications when certain thresholds are exceeded. For example, if the temperature in a greenhouse gets too high, the system could send an alert to the grower to take action.

4). *Control signals:* Depending on the application, the output could be control signals that are used to adjust the motion of the object or environmental conditions. For example, if the project is designed to control a drone, the output could be control signals that adjust the speed and direction of the drone's motion.

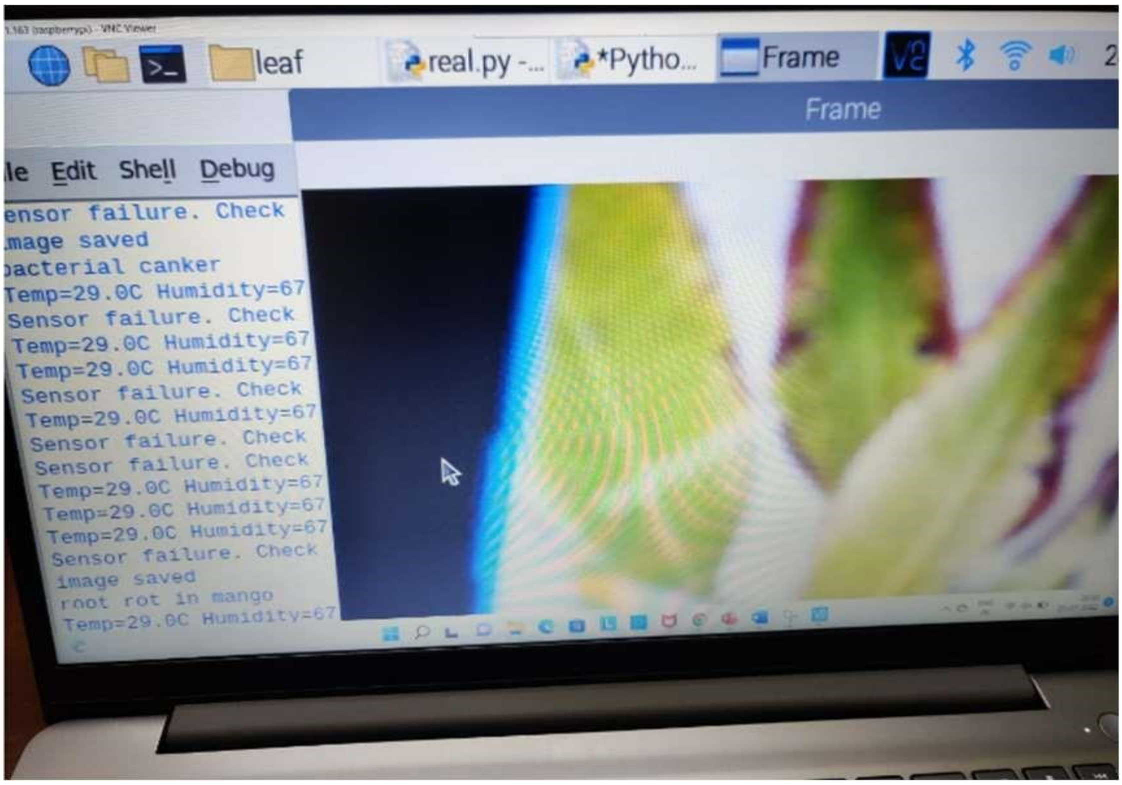

Figure 8: Output response

Finally, the leaf disease showing in screen as well as temperature and humidity of field.

d) Message out

1). *"Object detected":* If the project is designed to detect the presence of an object, the system could output a message when an object is detected.

2). *"Motion detected":* If the project is designed to track the motion of an object, the system could output a message when motion is detected.

3). "*Temperature/humidity warning*": If the project is designed to monitor environmental conditions such as temperature and humidity, the system could output a message when certain thresholds are exceeded, indicating that action should be taken.

4). "Control signal sent": Depending on the application, the system could output a message when a control signal is sent to adjust the motion of the object or environmental conditions.

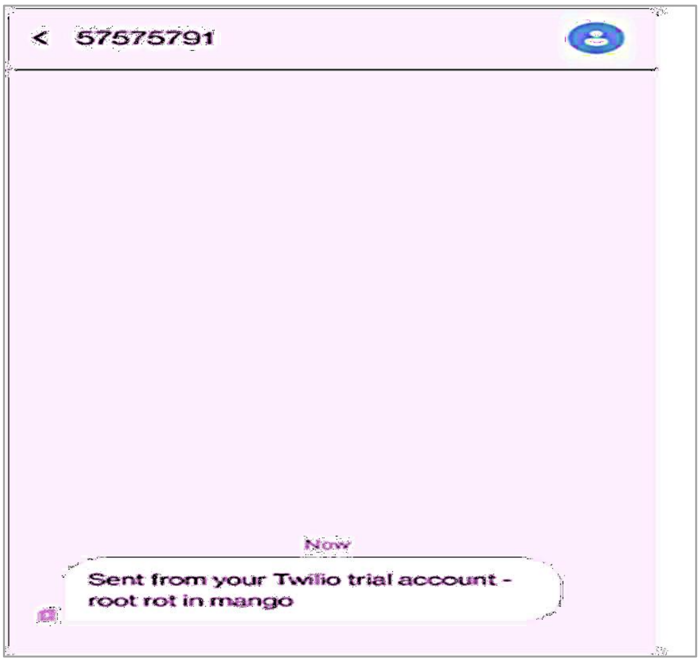

Figure 9: message out

IV. CONCLUSION

The use of leaf disease detection system are gaining increasing demand with technological advancement. The project deals with identifying the disease affected leaf. This is achieved through the Convolutional Neural Network Algorithm.

A). In the agriculture fields loss of yield mainly occurs due to widespread disease.

B). If the leaf is affected by disease, then the information is shared through the mail.

C). It also Enhances the operability of using the real-time functions.

D.) Sensors provide the accurate and quick alerts to the users mobile.

E). Expected a resolution to be well prepared by the user for optimum results.

F.) It is therefore detection of a disease serves as half the work done while, the rest is dependent on the user to react quickly to save the crop.

F). This is therefore a Embedded system with Technology driven outcomes requiring human interventions.

G). The Artificial Neural Networks was thus successfully implemented in this project.

H). This is a cutting-edge Technology which is widely used and has become a powerful tool in the Present scenario.

REFERENCES

[1]. Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016). Using deep learning for image-based plant disease detection. Frontiers in plant science, 7, 1419.

[2]. Sladojevic, S., Arsenovic, M., Anderla, A., Culibrk, D., & Stefanovic, D. (2016). Deep neural networks-based recognition of plant diseases by leaf image classification. Computational intelligence and neuroscience, 2016.

[3]. Liu, X., Zhang, Q., & Khan, M. K. (2018). Detection of plant diseases using convolutional neural networks and transfer learning. Plant methods, 14(1), 1-13.

[4]. Mardiyanto, R., Pranowo, H. D., & Nugroho, A. (2020). Leaf disease detection using convolutional neural network (CNN) and raspberry pi. In 2020 International Seminar on Intelligent Technology and Its Application (ISITIA) (pp. 40-44). IEEE.

[5]. Singh, A. K., Gangrade, S., & Choubey, A. (2019). Raspberry pi based smart agriculture system using machine learning techniques. In 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT) (pp. 1-6). IEEE.

[6]. Kumar, A., Jindal, A., & Goyal, M. (2021). Leaf disease detection using convolutional neural network and Raspberry Pi. In 2021 International Conference on Emerging Trends in Engineering, Science and Sustainable Technology (ICETESST) (pp. 1-6). IEEE.

[7]. Mukherjee, S., & Das, J. K. (2019). Real-time plant disease detection using deep learning techniques. In Proceedings of the 2nd International Conference on Intelligent Computing and Control Systems (ICICCS 2018) (pp. 1027-1032). Springer.

[8]. Noh, Y., Kim, S., & Kim, S. (2017). Deep learning-based real-time plant disease detection and classification using hyperspectral imaging. Sensors, 17(8), 1912.

[9]. Tripathy, S., & Agrawal, R. (2021). Leaf disease detection and diagnosis using deep learning techniques: A review. Archives of Computational Methods in Engineering, 28(1), 1-17.

[10]. Singh, A., Shukla, S., & Jain, P. (2019). Real time plant disease detection using deep learning with raspberry pi. In Proceedings of the International Conference on Inventive Communication and Computational Technologies (pp. 461-465). Springer.

[11]. Bernal, J., & Sanchez, A. (2020). A convolutional neural network for plant disease detection using Raspberry Pi. In 2020 International Conference on Internet of Things and Intelligence System (IoTaIS) (pp. 67-72). IEEE.

[12]. Du, J., Liu, X., & Chen, S. (2020). Leaf disease detection based on convolutional neural network and Raspberry Pi. In Proceedings of the International Conference on Computer Vision and Intelligent Systems (pp. 37-43). Springer.

[13]. Oukhellou, L., Zeroual, A., Ahmed, N., & Oukhellou, Y. (2020). Leaf disease detection using convolutional neural networks and raspberry pi. In Proceedings of the 2020 International Conference on Intelligent Systems and Signal Processing (pp. 191-196). Springer.

[14]] S J Badashah and S S Basha, "Classification of Rice Grains With Feature Extraction Using Discrete Wavelet Transform and Artificial Neural Networks" International Conference on Smart Electronics and Communication (ICOSEC 2020) Kongunadu College of Engineering and Technology Coimbatore, India 10-12, September 2020.

[15] T Venkat Narayana Rao, A Govardhan and S J Badashah "Improved Lossless Embedding & Extraction-A Data Hiding Mechanism" International Journal of Computer Science & Information Technology,(IJCSIT) ISSN:0975-4660,www.airccse.org ,April,Volume-2,number-2 Pp:77-88.

[16] S S Basha and S J Badashah, "Diabetic Retinopathy Detection with Optimal Feature Selection: An Algorithmic Analysis" 3rd International Conference on "Advanced Technologies for Societal Applications" Shri Vithal Education & Research Institute's College of Engineering, Pandharpur, during 11-12 December, Techno-Societal 2020, Proceedings of the 3rd International Conference on Advanced Technologies for Societal Applications.