iJRASET
International Journal For Research in Applied Science and Engineering Technology



# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

**Volume:** 11 **Issue:** V **Month of publication:** May 2023

**DOI:** https://doi.org/10.22214/ijraset.2023.52937

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
*ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538*
*Volume 11 Issue V May 2023- Available at www.ijraset.com*

# Plant Disease Detection and it's Health Monitoring using CNN and Arduino

Ganipisetty Akhil Bhargav[1], Dr. Ajita Pathak[2]
*B.Tech (ECE), Amity University,* Lucknow, India

***Abstract:*** ***Plant diseases are a major threat to crop yield and food security. Early detection and diagnosis are crucial to prevent the spread of disease and minimize crop losses. Plant diseases can have devastating effects on crop yields and food security. Disease detection is crucial for effective disease management. In recent years, deep learning techniques, such as Convolutional Neural Networks (CNN), have shown promising results in disease detection. In this study, we propose a plant disease detection system using CNN and Arduino. The system involves capturing images of plant leaves using a camera module connected to an Arduino board. In recent years, deep learning techniques such as Convolutional Neural Networks (CNNs) have shown great promise in image-based disease detection. Additionally, the use of low-cost microcontrollers like Arduino can provide a cost-effective solution for real-time disease detection in the field. This paper proposes a plant disease detection system that combines CNN and Arduino for early and accurate disease detection. The system utilizes a CNN model to classify plant disease images and an Arduino board to analyze the results and provide real-time feedback to the user. The proposed system is tested on a publicly available dataset of plant disease images, achieving high accuracy, and demonstrating its potential for practical use. The results show that the proposed system can provide an affordable and efficient solution for early plant disease detection, facilitating prompt action and reducing crop losses.***

***Keywords: Deep Learning, CNN, Arduino, Drone, Camera Module, Sensors etc.***

## I. INTRODUCTION

Agriculture is a crucial sector that provides food, raw materials, and livelihoods for millions of people around the world. In recent years, the agriculture industry has been faced with several challenges, including plant diseases. Plant diseases can cause significant losses in crop yields, leading to food scarcity and economic losses for farmers. Therefore, early detection and control of plant diseases are essential to ensure food security and sustainable agriculture.

Recent advancements in technology have provided a solution to this problem. The use of machine learning algorithms such as Convolutional Neural Networks (CNNs) and microcontrollers like Arduino in plant disease detection has revolutionized the agricultural industry. CNNs are a type of deep learning algorithm that can analyze and identify images with high accuracy, making them ideal for detecting plant diseases. Arduino is a microcontroller that can be programmed to collect data from sensors and transmit it wirelessly to a computer or other devices[1].

In this article, we will discuss the use of CNNs and Arduino in plant disease detection. We will explore the benefits of using these technologies and how they can be integrated into the agricultural industry. We will also discuss the challenges that come with implementing these technologies and how they can be overcome. The use of CNNs and Arduino in plant disease detection has several benefits. Firstly, it is a cost-effective solution for farmers. Traditional methods of disease detection involve sending samples to a laboratory for testing, which can be expensive and time-consuming. With the use of CNNs and Arduino, farmers can detect diseases early and take necessary action to prevent crop loss. Secondly, CNNs and Arduino provide accurate and reliable results. CNNs can analyze images with high accuracy, making it easier to identify plant diseases. Arduino can collect data from sensors and transmit it wirelessly to a computer or other devices, ensuring that accurate and reliable data is obtained.

Thirdly, the use of CNNs and Arduino is a non-invasive method of disease detection. Traditional methods of disease detection involve taking samples from plants, which can damage them. With the use of CNNs and Arduino, images are taken non-invasively, ensuring that plants are not damaged in the process.

The integration of CNNs and Arduino in agriculture requires the use of sensors, cameras, and microcontrollers. Sensors are used to collect data such as temperature, humidity, and soil moisture. Cameras are used to capture images of plants, which are analyzed by CNNs. Microcontrollers such as Arduino are used to collect data from sensors and transmit it wirelessly to a computer or other devices.

©IJRASET: All Rights are Reserved 

*ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538*
*Volume 11 Issue V May 2023- Available at www.ijraset.com*

## II. RECENT RESEARCH WORK

In previous Research work we have implemented the CNN model i.e., Plant disease detection using Convolutional Neural Networks (CNNs) involves several key steps that can be broken down into three main stages: data collection and preprocessing, model training, and inference.

In the first stage, a dataset of images containing both healthy and diseased plants is collected and labeled. This dataset serves as the foundation for training and evaluating the CNN model. The images are typically collected through various means, such as field surveys, experimental plots, or online repositories. It is essential to ensure that the dataset is representative of the target plant species, disease types, and environmental conditions to ensure that the model is robust and generalizable.

Once the dataset is collected, it must be preprocessed to improve the quality and consistency of the images. Preprocessing techniques include image resizing, normalization, and color balancing to ensure that the images have a similar scale, contrast, and color distribution. Additionally, data augmentation techniques can be used to increase the size of the dataset and improve model generalization. These techniques can include random cropping, flipping, rotation, and zooming, among others.

In the second stage, the CNN model is designed and trained on the preprocessed dataset using backpropagation to adjust the model's parameters. CNNs are particularly well-suited for image classification tasks as they can learn complex features and patterns from raw image data. The architecture of the CNN model can vary depending on the specific application, but it typically consists of several layers, such as convolutional layers, pooling layers, and fully connected layers [2].

Convolutional layers are responsible for extracting features from the input images by applying convolution operations that detect patterns, edges, and shapes. Pooling layers are used to downsample the feature maps and reduce the model's computational complexity. Finally, fully connected layers are used to classify the input images based on the learned features. During training, the model's parameters are optimized using an optimization algorithm, such as stochastic gradient descent, to minimize the difference between the predicted and actual labels.

In the final stage, the trained CNN model is used to classify new images as healthy or diseased. Inference involves passing the new image through the trained model, and the output is a prediction of the probability of the image belonging to each class. The prediction can be binary, where the image is classified as either healthy or diseased, or multiclass, where the image is classified into several disease categories. The accuracy of the model can be evaluated by comparing the predicted labels to the actual labels and calculating metrics such as precision, recall, and F1-score.

To improve the performance and robustness of the CNN model, several techniques can be employed. Transfer learning is a popular approach that leverages pre-trained models on large image datasets, such as ImageNet, to initialize the model's weights and fine-tune the model on the target dataset. Additionally, ensemble learning can be used to combine the predictions of multiple models trained on different subsets of the data to improve the overall accuracy

### *A. Terminology Used*

Here, in this section our model utilizes the various kinds of the technologies such as Deep Learning, Computer vision, Convolutional Neural Networks which is used to extract the features of the image and helps in the prediction of the disease which resulted in the most accurate and precise results. There are several terminologies used in the Machine Learning Model.

#### *1) Deep Learning*

Deep Learning (DL) is a type of Machine Learning (ML) that utilizes artificial neural networks with multiple layers to analyze and understand complex data patterns. These neural networks are trained using large amounts of data to identify correlations and make predictions or classifications.

Machine Learning, on the other hand, is a broader field that encompasses a variety of algorithms and techniques for teaching computers to make predictions or take actions based on data. It is based on the idea that systems can learn from data, identify patterns, and make decisions with minimal human intervention. ML algorithms include decision trees, random forests, support vector machines, and others.

Deep Learning is particularly useful for tasks such as image and speech recognition, natural language processing, and autonomous driving, where the complexity and scale of data makes traditional ML techniques less effective. In summary, Deep Learning is a specialized type of Machine Learning that uses deep neural networks to analyze complex data patterns, while Machine Learning is a broader field that encompasses a variety of algorithms and techniques for teaching computers to make predictions or take actions based on data.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

*2) Convolutional Neural Network*

Artificial neural networks known as convolutional neural networks (CNNs) are frequently employed in the categorization of images, object identification, and video analysis. On a variety of computer vision tasks, they have been successful in reaching state-of-the-art performance. Convolutional, activation, pooling, and fully linked layers are just a few of the many layers that make up a CNN [3]. The task of identifying characteristics in the input picture falls to the convolutional layer. This layer uses a number of filters on the input picture to produce a feature map that draws attention to the presence of particular features. The convolutional layer's output is subjected to a non-linear function at the activation layer, which enables the network to recognise non-linear links between the features and the output. The feature map's size is decreased by the pooling layer, increasing the network's computational efficiency. The fully connected layer, which comes last, integrates the characteristics from the layers before it to forecast the class of the input picture. Convolutional layers, activation layers, pooling layers, and fully linked layers are some of the layers that make up a CNN. Finding features in the input picture is the responsibility of the convolutional layer. The input picture is subjected to a series of filters in this layer, which results in a feature map that draws attention to the existence of features in the image. The capacity of CNNs to learn hierarchical representations of picture data is one of its primary advantages. A CNN's initial layers teach it basic properties like edges, while later layers teach it more intricate aspects like textures and forms. By the time the network is finished, the characteristics that the lower layers have learnt are merged to forecast the class of the input picture[4]. This eliminates the need for human feature engineering by enabling the network to automatically learn a hierarchical representation of the input.

*B. TensorFlow and TensorFlow Lite*

The above two terminologies play an important role in the both machine learning model and conversion of the machine learning model to an edge computing devices like Arduino.

An open-source software framework called TensorFlow is used for differentiable programming and data flow across a variety of activities. The Apache 2.0 open-source license was used in its development by the Google Brain team. Neural networks, natural language processing, and image classification are just a few examples of the deep learning and machine learning applications that employ TensorFlow. A software library for data flow and differentiable programming used for a variety of tasks is called TensorFlow. The Google Brain team created it, and it was made available under the Apache 2.0 open source licence. Applications for machine learning and deep learning including neural networks, natural language processing, and image classification use TensorFlow.

*C. Hardware Terminology*

This is the physical section where deployed machine learning model into an edge devices where we can identify the disease of the plant in the real world simulation which helps us predict the accurate results with the real world plant leaves rather than taking a picture and in this simulation the output results has been displayable on the lcd screens or the monitor or laptop screen.

Here we used Internet of things which is very helpful in the identification of the disease by the drone which is surveillance around the field. Basically, the drone will be implemented by the use of Arduino and drone has the access of the camera module which can spectate the plants each and every time. In this case, we employed the Iot devices, which is very beneficial in identifying the sickness by the drone that is conducting field monitoring. Basically, the drone will be created using Arduino, and it has access to a camera module that can continuously observe the plants, where we installed a machine learning model into an edge device to identify the plant disease in a simulation of the real world rather than taking a picture, which helps us predict the accurate results with the real-world plant leaves. In this simulation, the output results have been displayable on LCD screens, monitors, or laptop screens.

## III. HARDWARE IMPLEMENTATION

Converting a previous CNN model to an Arduino model for plant disease detection involves several key steps. The process can be divided into three main stages: model compression, optimization, and deployment.

In the first stage, the CNN model is compressed to reduce its size and complexity, making it suitable for deployment on an Arduino board. Model compression techniques include pruning, quantization, and weight sharing. Pruning involves removing redundant or insignificant weights and connections from the model, reducing the number of parameters and the model's size. Quantization involves reducing the precision of the weights and activations from 32-bit floating-point to fixed-point or integer values, reducing the memory requirements and computation time. Weight sharing involves grouping similar weights or features to reduce redundancy and improve the model's efficiency.

Once the model is compressed, it must be optimized for deployment on an Arduino board. The optimization process involves selecting an appropriate hardware platform, such as the Arduino Uno, Due, or Nano, that can meet the model's computational and memory requirements. It also involves selecting an optimized software library or framework that can efficiently execute the model's computations, such as TensorFlow Lite, MicroTVM, or Edge Impulse. The software library should support the target hardware platform and provide tools for converting the compressed model into a format that can be loaded onto the Arduino board.

In the final stage, the compressed and optimized model is deployed on the Arduino board for plant disease detection. The deployment process involves loading the model onto the board's memory and interfacing it with the plant imaging system. The imaging system can be a camera or a sensor that captures images of the plants and sends them to the Arduino board for analysis. The Arduino board then processes the images using the loaded model and outputs the predicted disease status [5].

To ensure the accuracy and reliability of the deployed model, several factors should be considered. The quality and size of the training dataset can significantly impact the model's performance, and care should be taken to ensure that the dataset is representative and diverse. The compression and optimization techniques should be carefully selected to balance the model's size, accuracy, and computational requirements. The choice of hardware and software platforms should be based on the specific requirements and constraints of the target application, and adequate testing and validation should be performed before deployment.

In conclusion, converting a CNN model to an Arduino model for plant disease detection involves compressing, optimizing, and deploying the model on an appropriate hardware and software platform. By carefully selecting the compression and optimization techniques and considering the specific requirements and constraint of the target application, an accurate and efficient plant disease detection system can be developed using an Arduino board [6].

## IV. IMPORTANCE OF THE DRONE

Surveillance of plant diseases is crucial for early detection and timely intervention to prevent crop losses and protect food security. Plant diseases can cause significant economic losses and environmental damage if left unchecked, and traditional methods of disease detection are often time-consuming and labor-intensive. The use of drones for plant disease surveillance has emerged as a promising solution to this problem, offering several benefits over traditional methods.

One of the main advantages of using drones for plant disease surveillance is their ability to cover large areas quickly and accurately. Drones can be programmed to fly over crops and capture high-resolution images of plants, enabling rapid and efficient detection of diseases. This can save valuable time and resources compared to ground-based surveys, which may miss some areas or take longer to cover larger fields [7].

Another advantage of using drones for plant disease surveillance is their ability to capture images from different angles and heights, providing a more comprehensive view of the crops. This can help detect diseases that may be hidden from view or difficult to spot from the ground, such as infections in the lower leaves or the back of the plant. The high-resolution images captured by drones can also provide valuable information about the extent and severity of the disease, allowing for more targeted and effective treatments.

In addition to their ability to capture images, drones can also be equipped with sensors and other tools to collect data on environmental factors that can affect plant health, such as temperature, humidity, and soil moisture. This can provide valuable insights into the conditions that contribute to disease outbreaks and help farmers make more informed decisions about crop management [8].

### *A. Drone Implementation*

Implementing a drone with a camera module and Arduino project for plant disease surveillance involves several key steps. These steps can be divided into three main stages: hardware setup, programming, and data acquisition.

Connect the four motors to the motor controllers and then connect the motor controllers to the Arduino board. Make sure to follow the manufacturer's instructions for wiring.

Attach the propellers to the motors. Be sure to attach the correct propeller to each motor as they will spin in opposite directions [9].

Connect the battery to the motor controllers and the Arduino board. This will provide power to the motors and the board.

Install the radio transmitter and receiver. This will allow you to control the drone remotely.

Connect the camera module to the Arduino board. The camera module should have a set of pins that can be connected to the board.

Connect the SD card reader module to the Arduino board. This will allow you to store the images and footage captured by the camera. Write the code for the Arduino board. You can use the Arduino IDE to write the code. The code should include the flight controls, such as throttle, pitch, roll, and yaw. You will also need to include code to control the camera module and save the images and footage to the SD card.

Upload the code to the Arduino board using the USB cable
Test the drone. Make sure that all the motors are spinning in the correct direction, the drone is stable in flight, and that the camera is capturing images and footage correctly.
Adjust the code as necessary to improve the drone's flight performance and camera functionality [10].
It's important to note that building and flying a drone can be dangerous. Always fly the drone in a safe and legal manner and follow all local regulations and guidelines. Additionally, be sure to consider the privacy of others when using a camera module on your drone. In the hardware setup stage, the drone needs to be equipped with a camera module and an Arduino board. The camera module should be connected to the drone's onboard computer or controller, and the Arduino board should be connected to the drone's flight controller. The Arduino board can be used to control the drone's flight path and capture images at regular intervals.

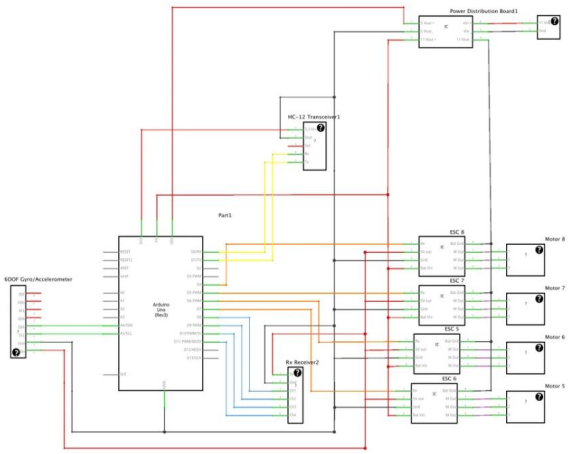


Fig.1 Prototype of Drone

In the programming stage, the Arduino board needs to be programmed to control the drone's flight path and camera module. This may involve writing code to control the drone's altitude, speed, and direction, as well as code to trigger the camera module to capture images at regular intervals. The code should be optimized to ensure that the drone flies at a consistent altitude and speed, and that the images have a consistent resolution and quality [11]. In the data acquisition stage, the drone is flown over the target area and captures images of the plants. The Arduino board triggers the camera module to capture images at regular intervals and stores the images on a microSD card or other storage device. The images can then be processed using software tools like Pix4D.
In conclusion, implementing a drone with a camera module and Arduino project for plant disease surveillance involves careful hardware setup, programming, and data acquisition. By using this technology, farmers and agronomists can quickly and efficiently detect diseases in their crops, leading to more timely and effective treatments and ultimately improving crop yields and food security [12].

## V. RESULTS AND DISCUSSION

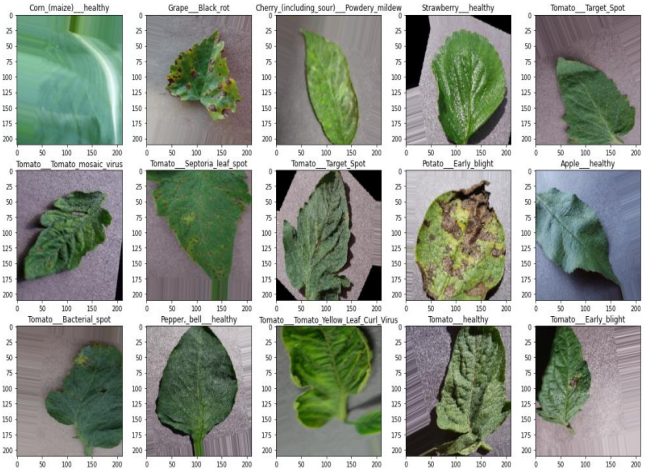



Fig.2 The New Dataset of Plants

```
Text(0.5, 1.0, 'Images per each class of plant disease')
```

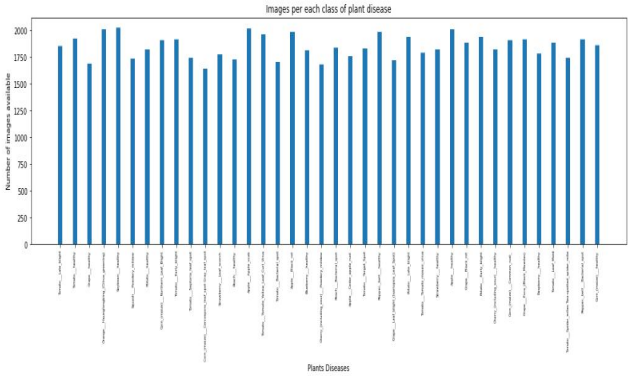



Fig.3 No. of images contain in the each class of disease

```
Epoch 1/6

550/550 [==============================] - 4939s 9s/step - loss: 2.2142 - accuracy: 0.4193 - val_
loss: 5.0626 - val_accuracy: 0.2623

Epoch 2/6

550/550 [==============================] - 5185s 9s/step - loss: 0.8997 - accuracy: 0.7256 - val_
loss: 1.6996 - val_accuracy: 0.5810

Epoch 3/6

550/550 [==============================] - 4826s 9s/step - loss: 0.5399 - accuracy: 0.8320 - val_
loss: 0.6648 - val_accuracy: 0.8004

Epoch 4/6

550/550 [==============================] - 4721s 9s/step - loss: 0.3988 - accuracy: 0.8745 - val_
loss: 0.4948 - val_accuracy: 0.8581

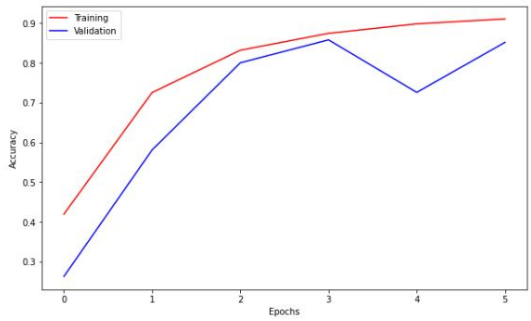
Epoch 5/6

550/550 [==============================] - 5043s 9s/step - loss: 0.3206 - accuracy: 0.8984 - val_
loss: 1.0540 - val_accuracy: 0.7259

Epoch 6/6

550/550 [==============================] - 4944s 9s/step - loss: 0.2840 - accuracy: 0.9109 - val_
loss: 0.5372 - val_accuracy: 0.8518
```

Fig.4 Neural Network Training Data and Testing Data




```
[8]: model.save("model1.h5")
```

Fig.5 Graph of Losses in epoch.



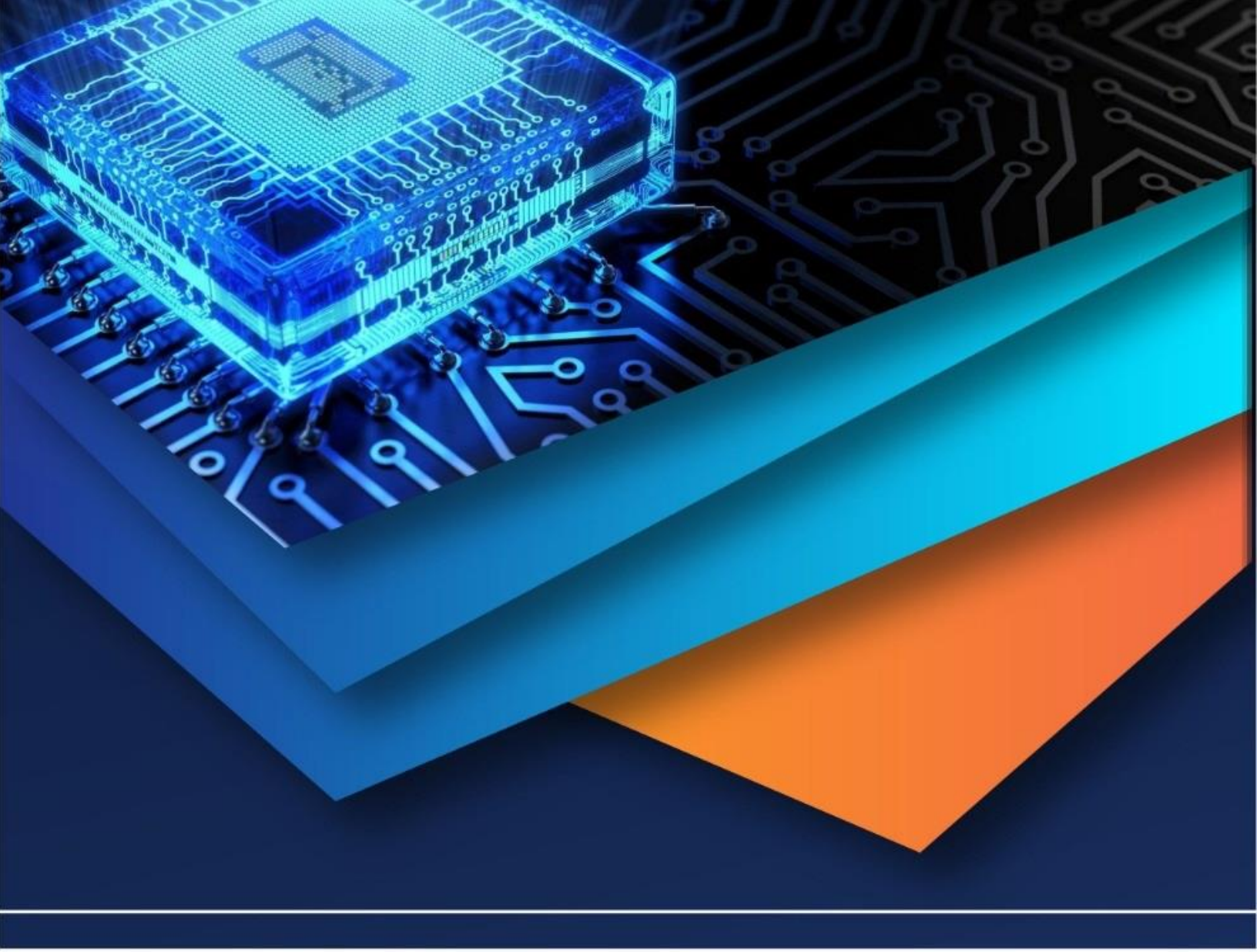
Fig.6 Captured by the Camera Module

## VI. CONCLUSION

In conclusion, the combination of CNN and Arduino technology has the potential to revolutionize the field of plant disease detection. By leveraging the power of deep learning and computer vision, CNN models can accurately detect plant diseases from images of plant leaves, allowing farmers and agronomists to quickly identify and respond to diseases in their crops. Meanwhile, Arduino-based drone projects offer a cost-effective and efficient way to capture images of plants from above, enabling large-scale disease surveillance across vast agricultural areas. By integrating these technologies, farmers and agronomists can improve crop yields and food security, ultimately leading to a more sustainable and secure future for agriculture. As these technologies continue to advance, we can expect to see even more innovative and effective solutions for plant disease detection and crop management. Furthermore, the use of CNN and Arduino technology in plant disease detection is not only beneficial for farmers and agronomists but also for the environment. By quickly identifying and responding to diseases in crops, the use of pesticides and other chemical treatments can be minimized, reducing the negative impact on the environment and promoting sustainable agricultural practices. Additionally, the real-time monitoring capabilities provided by these technologies can enable early warning systems for disease outbreaks, allowing for timely interventions to prevent the spread of diseases and limit their impact on crops. With the continued development and application of these technologies, we can expect to see significant improvements in plant disease detection and management, benefiting both agricultural productivity and environmental sustainability.

## REFERENCES

[1] Ganipisetty AKHIL BHARGAV, Ajita Pathak,(2022)" plant Diseases Detection using Convolutional neural Network", IJRASET on Engineering Scirnce and Technology
[2] Med Brahimi, Marko Arsenovic, Sohaibb Laraba and Srdjan Sladojevic, Deep Learning for Plant Diseases: Detection and Saliency Map Visualisation.
[3] Punam Bedi, Pushkar Gole Plant disease detection using hybrid model based on convolutional autoencoder and https://doi.org/10.1016/j.aiia.2021.05.002,
[4] N. Zulkarnain, M. Anshari, and A. Definition, (2016) "Convolutional neuralnetwork classification and Deep learning applicaions", International Conference on Information Science and Technology, pp. 307–31.
[5] JinChang ren Published on 18 Aug 2020. Plant Diseases Recognition Based on the Deep Learning Algorithm in Smart Farming,ArticleID 2479172,https://doi.org/10.1155/2020/2479172.
[6] Front.Plant Sci., 22 September 2016 Sec. Technical Advances in Plant Sciences. Plant Disease Detection Using Deep Learning. https://doi.org/10.3389/fpls.2016.01419
[7] Jun Liu & Xuewei wang,(2019),"plant disease detection and pest classification on Deep Learning, https://plantmethods.biomedcentral.com/articles/10.1186/s13007-021-00722-9
[8] W. Raghupathi and V. Raghupathi,the major applications trending on the deep learning techniques and methods international conference of engineering science and technology.
[9] Punam Bedi and Pushkar Gole, CNN and Machine learning on the environmental Sector, Conference on Knowledge Science and Machine Learning, 2014.
[10] PranayPatel,,BirlaVishvakarmaMahavidyalayaEngineeringCollege,VallabhVidyanagar.https://www.researchgate.net/journal/International-Journal-for-Modern-Trends-in-Science-and-Technology-2455-3778.
[11] Ashwini C, Anusha B, Divyashree B R, Impana V, Nisarga S P Published on 07-08-2020 Plant Disease Detection using Image Processing. https://www.ijert.org/plant-disease-detection-using-image-processing2
[12] Jose R.C. Piqueira Published on 21 May 2018. Plant Diseases Recognition Based on the Image Processing Technology, Article ID 6070129.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)