DETECTION AND CLASSIFICATION OF FRUIT DISEASES USING CONVOLUTION NEURAL NETWORK

A major project
Submitted to the Department of Information Technology
in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

BY

A.SOUJANYA	20567T1842
G.PAVITHRA	20567T1828
R.SAIKUMAR	20567T1833
A.JYOTHSNA	20567T1813
B.RENUSRI	215671856L

Under the Guidance of
M.SOUJANYA
Assistant Professor

DEPARTMENT OF INFORMATION TECHNOLOGY
KU COLLEGE OF ENGINEERING ANDTECHNOLOGY,
KAKATIYAUNIVERSITYCAMPUS, VIDYARANYAPURI,
WARANGAL 506 009, INDIA

APRIL 2024

DEPARTMENT OF INFORMATION TECHNOLOGY
KU COLLEGE OF ENGINEERING AND TECHNOLOGY,
KAKATIYA UNIVERSITY CAMPUS, VIDYARANYAPURI,
WARANGAL 506 009, INDIA
APRIL 2024

CERTIFICATE

This is to certify that the project report entitled "**DETECTION AND CLASSIFICATION OF FRUIT DISEASES USING CONVOLUTION NEURAL NETWORK"** which is being submitted by A.SOUJANYA (20567T1842), G.PAVITHRA (20567T1828), R.SAIKUMAR(20567T1833), A.JYOTHSNA(20567T1813) , B.RENUSRI (215671854) in the partial fulfilment for the award of Bachelor of Technology in Information Technology to KAKATIYA UNIVERSITY is a record of work carried out during the academic year 2023-2024 under our guidance and supervision.

Supervisor
(Mrs. M Soujanya)

Project Coordinator
(Mrs R Manjula)

Head of the Department
(Mr M Venugopal Reddy)

Principal
(Prof. M Sadanandam)

Internal examiner

External Examiner

DECLARATION

We declare that this written submission represents our ideas in our own words and where others ideas or words have been included, we have adequately cited and referenced the original sources. We declare that the work presented in this project report is original and carried out in the Department of Information Technology, KU College of Engineering & Technology, Warangal, Telangana, and has not been submitted elsewhere for any graduate in part or in full.

A.SOUJANYA	(20567T1842)
G.PAVITHRA	(20567T1828)
R.SAIKUMAR	(20567T1833)
A.JYOTHSNA	(20567T1813)
B.RENUSRI	(215671854L)

ACKNOWLEDGEMENT

I thank the almighty for giving me the courage and perseverance in completing the project. This project itself is an acknowledgement for all these people who have given me their heartfelt co-operation in making this project success.

I take this opportunity to express my deep and sincere gratitude to the project In-charge **R.MANJULA** Department of IT, for their valuable advice at every stage of this work; without their supervision this project would never have come out in this form.

I am greatly indebted to my project supervisor **Mrs. M.SOUAJNYA** for many hours of dedicated guidance, stimulating and constructive criticism. This project came out in this form.

I am thankful to **M. Venu Gopal Reddy**, Head of the Department, for providing the excellent facilities, motivational and valuable guidance throughout the project work.

I am thankful to Prof. P. Malla Reddy, Principal, KU College of Engineering and Technology, for his co-operation and encouragement to complete the project work in time. Last but not least I would like to express my deep sense of gratitude and earnest thanks giving to my dear parents for their moral support and heartfelt co- operation in doing the project. I would also like to thank all the teaching and non-teaching staff and our friends, whose director in direct help has enabled us to complete this work successfully.

A.SOUJANYA **(20567T1842)**
G.PAVITHRA **(20567T1828)**
R.SAIKUMAR **(20567T1833)**
A.JYOTHSNA **(20567T1813)**
B.RENUSRI **(215671854L)**

ABSTRACT

In the field of agriculture Fruits are affected by different climatic conditions. For the farmers it is very difficult to identify the fruits that are having disease or not. This will affect the global economy. By manual testing of fruits does not give the accurate output. In this method the fruit image is take to analyse the disease. The disease must be identified using colour, shape and size of the fruit. It will give the accurate result for farmers than the manual analysis of the fruits. All kinds of fruits are taken in this method. The system is already trained with different types of diseases that fruits are having. Input image given by the user undergoes with different kinds pre-processing steps.

First the fruit image is resized in the required manner and the type of the fruit identified by the system. The fruits related to input image in the trained dataset must be analysed. And then features such as colour, shape and size are extracted from the image and image cluster is identified using k-means clustering algorithm. Next convolutional neural network(CNN) is used to find the fruit is infected or not infected.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

India is the rural land, India produces 44.04 million tons of products of the soil is a second biggest maker of organic products. India contributes 10% in world natural product creation. Indian ranchers develop assortment of organic products those are apple, banana, citrus, grape, mango, guava, papaya, watermelon and some more. Natural product industry contributes around 20% of the nation's turn of events. Amount and nature of the rural things are diminished by the sickness of organic products. The fundamental driver for natural product illnesses are infections and microorganisms. The infections are additionally brought about by terrible ecological conditions. There are various attributes and practices of such natural product sicknesses in which large numbers of them are less discernable. The determination of organic product illness is a significant viewpoint

1.1 PROBLEM STATEMENT:

Agriculture has been the base for every people. It is most important that more than 70% of the people depend on agriculture for their livelihood in India. Nowadays the growth of productivity of plants, crops and fruits are normally affected by the diseases. In order to generate a solution to examine the infections using proposed method.

1.2 PROPOSED STATEMENT:

The classical approach for detection and identification of fruit diseases is based on the naked eye observation by the experts. In some developing countries, consulting experts are expensive and time consuming due to the distant locations of their availability. Automatic detection of fruit diseases is essential to automatically detect the symptoms of diseases as early as they appear on the growing fruits. Fruit diseases can cause major losses in yield and quality appeared in harve

CHAPTER 2

SYSTEM ANALYSIS

2.1 EXSISTING SYSTEM

Agriculture plays important role in increasing cross productivity of each country. Majorly the fruits production is increasing rapidly due to the hybrid production. Even though we use hybrid seeds and best fertilizers to the plants they are not giving a good productivity. This is mainly because the fruits that are in growing stage are infecting by diseases. So in the exiting system for detection and identification of fruit diseases, K-means clustering technique and Support Vector Machine are used

DISADVANTAGE

- Support vector machine is computationally sensitive with large datasets. Training time and memory requirements may become significant, limiting their scalability.
- The accuracy of detecting a fruit disease is less with SVM.
- SVM are sensitive to the scale of input features, and thus proper preprocessing is often required for optimal performance.

2.2PROPOSED SYSTEM

For more effective fruit disease identification, precise image segmentation is required. Otherwise, the non-infected region's traits dominate the infected region's characteristics. In our project Convolution Neural Network based image processing is recommended in this proposed method to determine the specific region that is solely affected. After the processing step was completed, specific features were extracted from the processed image. Finally, the method's training and classification processes are carried out, and the precise result is supplied.

ADVANTAGES :

- Accuracy is high and it is Applicable for both low and high pixel images.
- Enhancing the value of fruit disease detection also takes only a few seconds to provide an exact result.
- The name of the disease is also found by highlighting the affected places.

CHAPTER 3

SYSTEM REQUIREMENTS

Software Requirements:

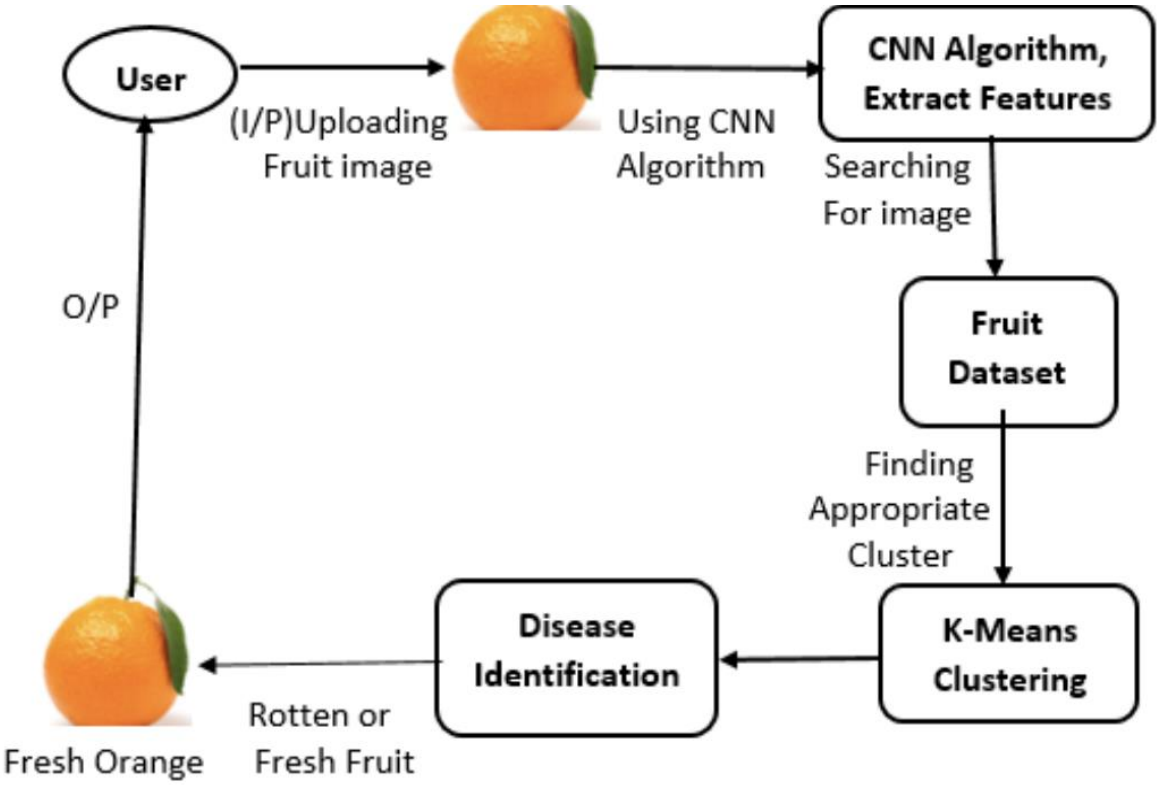
- **Operating System**: Windows
- **Coding Language**: Python 3.7
- **Implementation**: Django

Hardware Requirements:

- **Processor** - Pentium–III
- **Speed** – 2.4GHz
- **RAM** - 512 MB(min)
- **Hard Disk** - 20 GB
- **Floppy Drive** - 1.44M

CHAPTER 4

SYSTEM ARCHITECTURE



CHAPTER 5

MODULES

Description of modules

Detection and classification of fruit disease has 5 modules

MODULE 1: INPUT IMAGE

This is the initial stage. The collecting of sample photos was done here, and the acquired images were utilized to train the classifier model and develop the classifier model. Pictures of healthy and diseased fruits were recorded with readily available mobile phone cameras and utilized as the training set and test set images for the classification algorithm. Images were taken from diverse perspectives, in varied situations, and under varying lighting conditions. We commonly store these photographs in the "JPG" format. For this classifier method, we'll require test and training sets of data from agricultural fields in various regions.

MODULE 2: DATASET PREPROCESSING

Following the acquisition of sufficient datasets, we subject these datasets to the Preprocessing stage to improve the image quality. All of the original fruit photographs were saved in a single folder. We can save these photos under any name we like. Images are taken in a variety of sizes and angles. For example, if a picture is acquired horizontally, it must be rotated 90 degrees and enlarged to a dimension of 200 by 300 pixels, as we consider the following size to be typical for every dataset used. And, if the acquired images' height and breadth were also varied, those images would need to be scaled to 250 x 250 pixels. If the image is large, then there will be a delay in processing. After the image scaling procedure is completed, an image restoration approach is performed to decrease noise and improve the image's sharpness. It improves the overall image quality. All such photographs are saved in the same folder after this process is completed.

MODULE 3: SEGMENTATION

Image segmentation is the third phase of this disease detection technique. The initial step is to convert all pre-processed photos to L*a*b, HSV, and Gray colour models before saving them in RGB format.. Image segmentation is a technique for dividing an image into smaller, more manageable chunks. It is performed to ease the classification and analysis of features in the images. The border, area, edge and other features of the image are identified in the image. This method also identifies the best colour-modeled image, Noise reduction, edge detection, and shape refining of all part of this process for improving the input image's quality .

MODULE 4: FEATURE EXTRACTION

Here, we propose to employ colour features such as mean and standard deviation to distinguish between diseases. Colour is a key attribute. Furthermore, each disease may have various shape. Texture features such as Kurtosis, skewness, cluster prominence, and cluster shadow. It is usually performed to decrease the complexity in processing the image. The feature differences in the fruit images indicate infection, and the disease is identified based on the threshold value of the disease.

MODULE 5: CLASSIFICATION

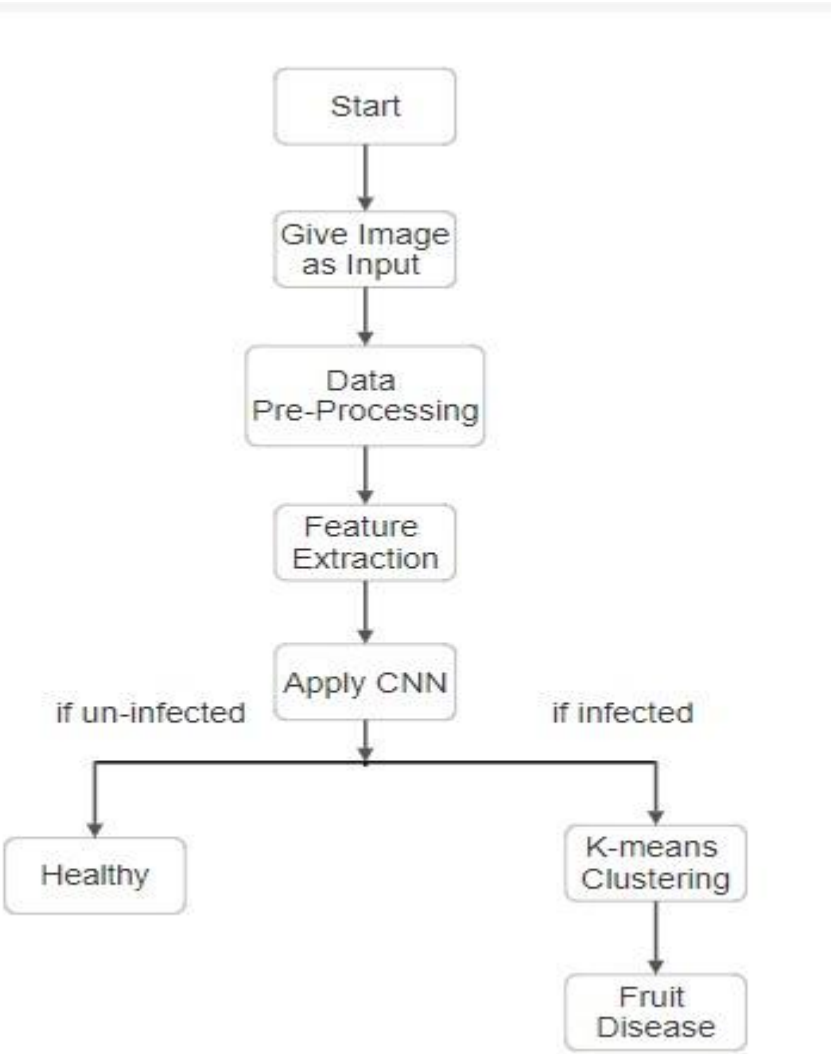
K-Mean Clustering, Convolution Neural Network, and SVM classification algorithms are used to examine and classify the images, as well as the numerical characteristics of numerous features. Feature classification from extracted image features is the focus of this technique. The algorithm is then trained on a set of photos that have already been processed by the preceding rounds. Here Convolution Neural Network plays key role in detection of fruit disease by deeply visualizing even small square templates, called Convolution Kernals, which slide over the image and look for patterns. Where that part of the image matches the kernel’s pattern, the kernel returns a large positive value, and when there is no match, the kernel returns zero or a smaller value.

CHAPTER 6

DESIGN - UML DIAGRAMS

DATAFLOW DIAGRAM

A Data Flow Diagram (DFD) is a graphical representation that illustrates how data is processed and exchanged within a system. It is a visual tool commonly used in system analysis and design to depict the flow of data through various processes. DFDs provide a clear and concise way to represent the interaction between different components of a system, including processes, data sources, and external entities**.**



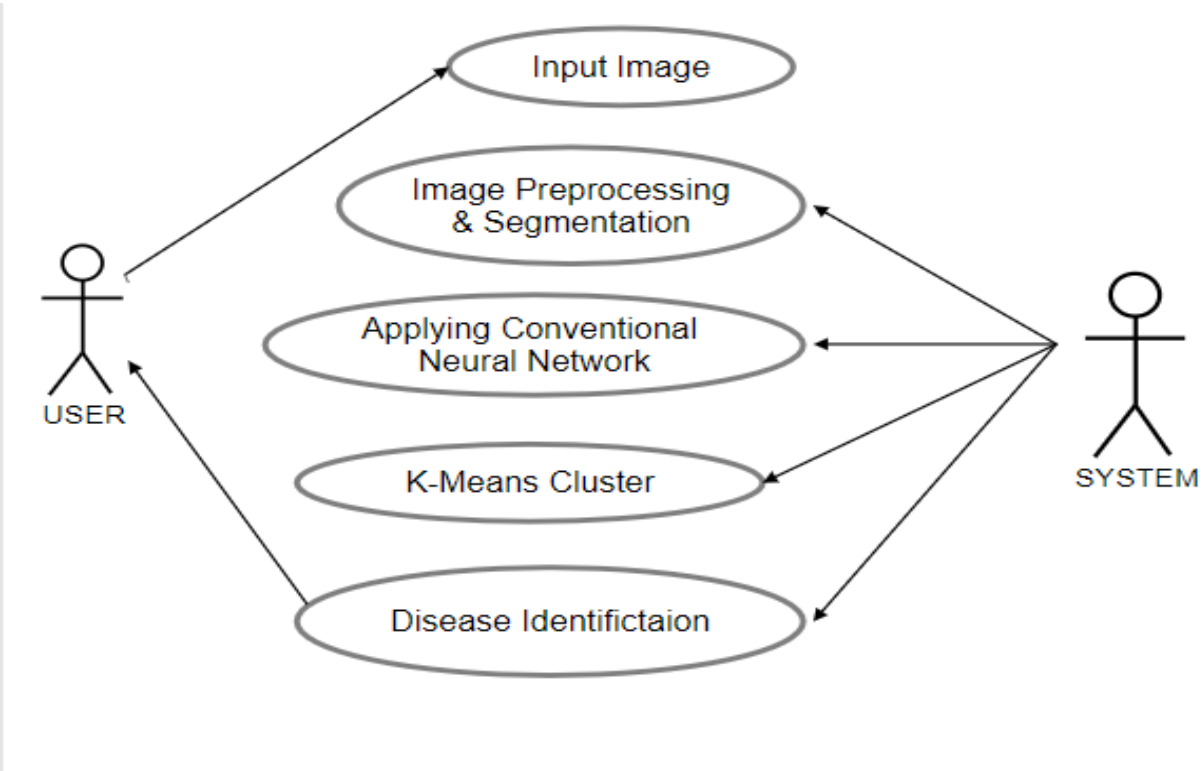
UML DIAGRAMS

Unified Modelling Language (UML) diagrams are a set of standardized graphical notations used in software engineering to visually represent and document the design and structure of a system. UML provides a common language and notation for software developers, analysts, and stakeholders to communicate and understand various aspects of the system. The UML diagram captures different perspectives of a software system, including its structure, behaviour and interactions.

1.USECASE DIAGRAM

Use case diagrams, also known as behaviour diagrams, are used to illustrate a series of tasks that a system or systems (the topic) should or could do in coordination with one or more external users of the system (actors). At its most basic level, a use case diagram is a depiction of a user's interaction with the system that shows the relationship between the user and the various use cases that the user is involved in

Here Actors are the User and System. User give image as input to the system and System gives the detected disease name. And the Use case is the action performed by actor according to our procedure.

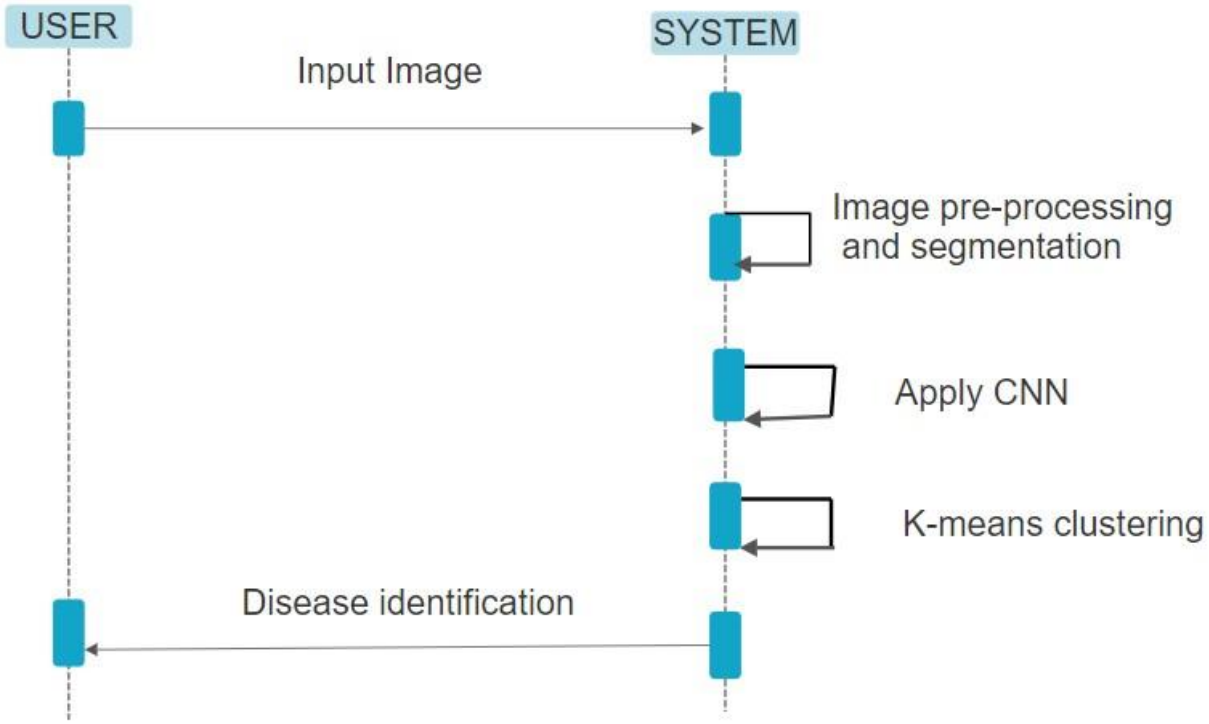


USECASE DIAGRAM

2.SEQUENCE DIAGRAM

The sequence diagram represents the flow of messages in the system and is also termed as an event diagram. It helps in envisioning several dynamic scenarios. It portrays the communication between any two lifelines as a time-ordered sequence of events, such that these lifelines took part in the run time. In UML, the lifeline is represented by a vertical bar, whereas the message flow is represented by a vertical dotted line that extends across the bottom of the page. It incorporates the iterations as well as branching.

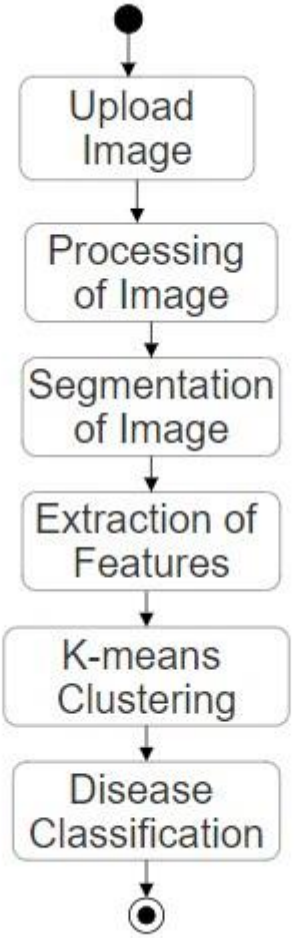
SEQUENCE DIAGRAM



The sequence diagram shows the interaction between the user and the system and various actions (messages exchanged) between the user and the system. It shows the flow of messages.

3.ACTIVITY DIAGRAM

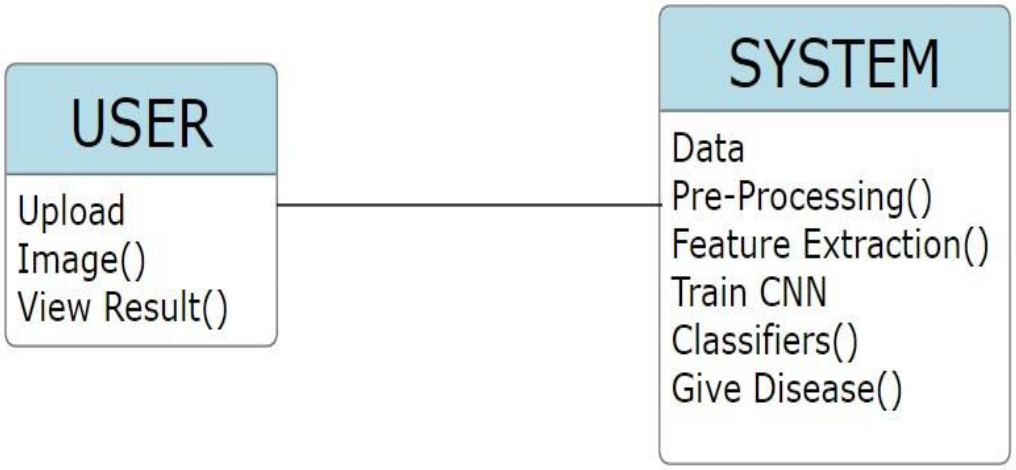
Another crucial diagram in the UML used to depict the system's dynamic elements is the activity diagram. An activity diagram is essentially a flow chart that shows how one activity leads to another. The action might be referred to as a system operation. As a result, one operation is followed by another in the control flow.



As shown in the activity diagram, here it starts with the data collection, and data is ingested by the machine learning technique and undergoes various steps in the Convolution Neural Network and finally predicts the Disease of the given Fruit

4.CLASS DIAGRAM

The class diagram is static. It represents the static view of an application. A class diagram describes the attributes and operations of a class and also the constraints imposed on the system. Class diagrams are widely used in the modelling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

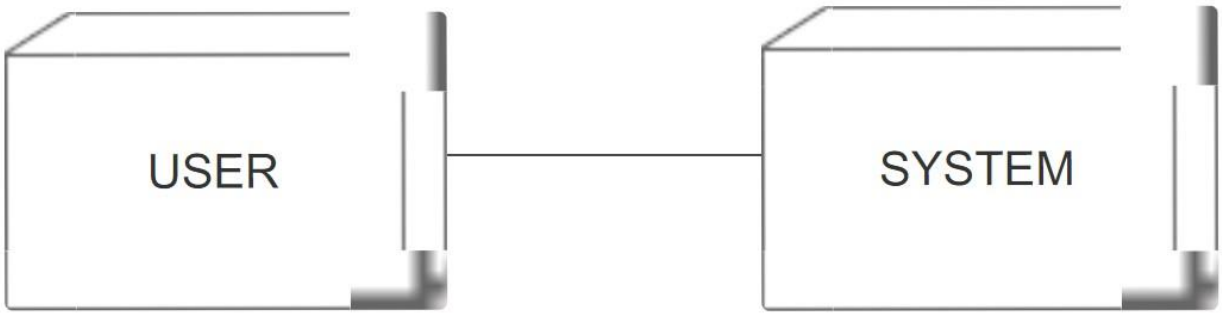


CLASS DIAGRAM

Our project involves two main classes, one is the User and the other is the System. The diagram shows the static view of both the user and the system along with their operations and attributes. Attributes map onto member variables (data members) in code and operations are services the class provides.

5.DEPLOYMENT DIAGRAM

The deployment diagram visualizes the physical hardware on which the software will be deployed. It portrays the static deployment view of a system. It involves the nodes and their relationships. The main purpose of the deployment diagram is to represent how software is installed on the hardware component. It depicts in what manner a software interacts with hardware to perform its execution.



DEPLOYMENT DIAGRAM

CHAPTER 7

IMPLEMENTATION

In the implementation phase we are going to implement the project main idea. In our project in this phase we used Python programming language and Convolution Neural Networks.

7.1 PYTHON

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

7.1.1 Features of python

1. Readability and Simplicity

Python places a strong emphasis on clean, readable code. Its syntax is designed to be highly readable, with minimal use of brackets and a clear layout using indentation and whitespace. By relying on indentation to define code blocks, Python enables developers to write code that is not only easy to write but also easy to read and understand.

2. Extensive Standard Library

Python comes with an extensive standard library, which is a collection of modules and packages that offer a wide range of functionalities. These modules and packages help simplify complex tasks by providing pre-built functionalities, saving developers time and effort.Python's standard library includes modules and packages for regular expressions, GUI programming, networking, data processing, and much more. This vast ecosystem ensures that developers have access to a wide array of tools without the need for external dependencies. It streamlines software development and ensures compatibility across different versions of Python.

3. Cross-Platform Compatibility

Python's cross-platform compatibility allows it to run seamlessly on different operating systems such as Windows, macOS, and Linux. Developers can write their code once and deploy it on multiple platforms, saving time and effort in application development.

4. Dynamic Typing and Dynamic Binding

Python is a dynamically-typed language, which means that the type of a variable is determined at runtime. Unlike statically-typed languages, such as C++ or Java, Python does not require explicit type declarations. This dynamic typing feature results in faster development time and increases code flexibility.

5. Strong Community Support

Python enjoys a vibrant and supportive community of developers worldwide. This community provides various resources, including forums, online communities, and open-source projects. The Python community strongly believes in knowledge sharing and mutual support.

6. Object-Oriented Programming (OOP) Capabilities

Python supports object-oriented programming (OOP) principles, such as encapsulation, inheritance, and polymorphism. OOP enables developers to organize their code into reusable and independent modules or components, making it easier to manage complex applications. Python's syntax and standard library provide intuitive and convenient ways to implement OOP concepts.

7. Wide Range of Libraries and Frameworks

For web development, Python offers popular frameworks like Django and Flask, which simplify the process of building feature-rich web applications. In the field of data science, Python provides powerful libraries like NumPy and Pandas for data manipulation and analysis. When it comes to machine learning, Python offers libraries like TensorFlow and PyTorch, which enable developers to create and train machine learning models efficiently.

8. Versatility

Python's versatility is one of its key strengths. It can be used for a wide range of applications, including web development, data analysis, machine learning, scripting, and more. This versatility allows developers to use a single language for diverse projects, simplifying their workflow and reducing the learning curve associated with switching between different languages.

9. Read-Evaluate-Print Loop (REPL) Environment

Python offers an interactive shell or Read-Evaluate-Print Loop (REPL) environment, enabling developers to experiment with code snippets quickly. The REPL environment allows developers to execute code and immediately see the output, facilitating rapid prototyping and experimentation.

10. Community-Driven Updates and PEPs

The iterative nature of Python's development, based on community feedback, ensures that it stays up-to-date with the latest software development trends, features, and tools. Recent updates in Python 2023 introduce new syntax features, improved module systems, and performance optimizations, among other enhancements. The community-driven updates maintain Python's relevance and effectiveness in modern software development.

7.1.2 Advantages of Python over other Languages:

1. Less Coding

Almost all of the tasks done in Python requires less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don't have to search for any third-party libraries to get your job done. This is the reason that many people suggest learning Python to beginners

2. Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support.

The 2019 Github annual survey showed us that Python has overtaken Java in the most popular programming language category.

3. Python is for everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and **machine learning**, automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language

7.2 MACHINE LEARNING

Machine learning (ML) is a subdomain of artificial intelligence (AI) that focuses on developing systems that learn—or improve performance—based on the data they ingest. Artificial intelligence is a broad word that refers to systems or machines that resemble human intelligence. Machine learning and AI are frequently discussed together, and the terms are occasionally used interchangeably, although they do not signify the same thing. A crucial distinction is that, while all machine learning is AI, not all AI is machine learning.

Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that makes it more similar to humans: The ability to learn. Machine learning is actively being used today, perhaps in many more places than one would expect.

Features of Machine Learning

- Machine learning is data driven technology. Large amount of data generated by organizations on daily bases. So, by notable relationships in data, organizations makes better decisions.
- Machine can learn itself from past data and automatically improve.
- From the given dataset it detects various patterns on data.
- For the big organizations branding is important and it will become more easy to target relatable customer base.
- It is similar to data mining because it is also deals with the huge amount of data.

Even though there are lot of advantages with Machine Learning there are few disadvantages to Machine Learning like

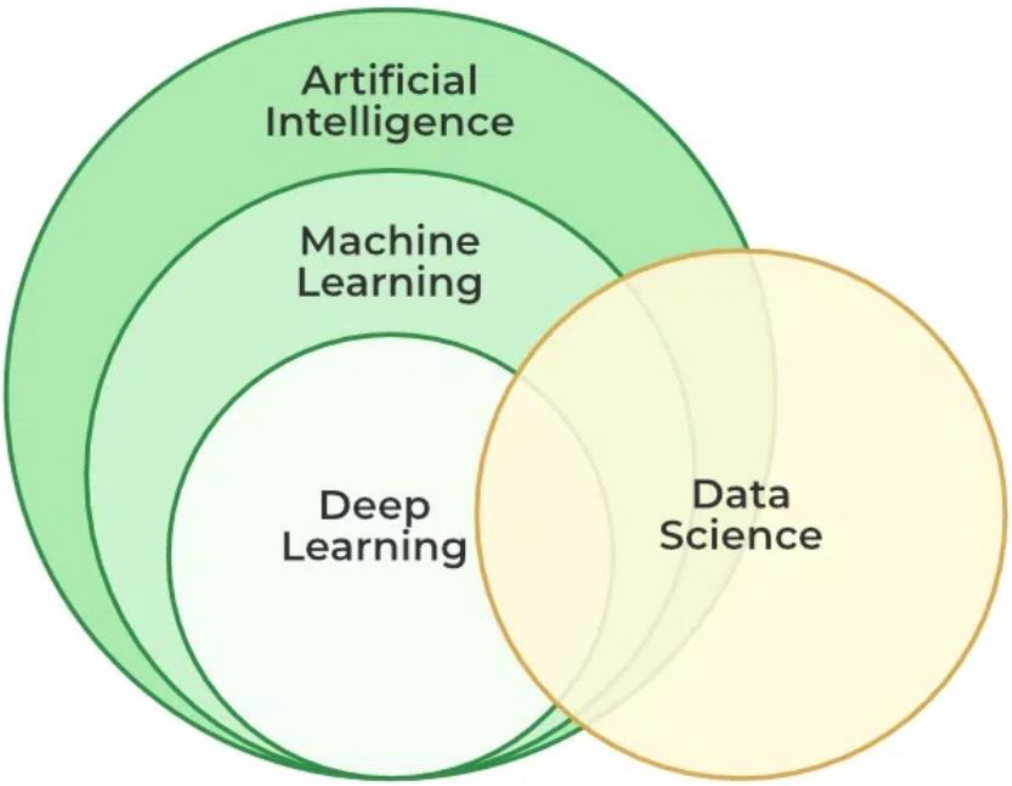
- Data Privacy and Security Concerns
- Initial Costs and Resources
- Lack of Interpretability
- Dependence on Data Quality

7.2.1 Deep Learning

Deep learning is a subset of machine learning that uses multi-layered neural networks, called deep neural networks, to simulate the complex decision-making power of the human brain. Some form of deep learning powers most of the Artificial Intelligence(AI) in our lives today.

By strict definition, a deep neural network, or DNN, is a neural network with three or more layers. In practice, most DNNs have many more layers. DNNs are trained on large amounts of data to identify and classify phenomena, recognize patterns and relationships, evaluate possibilities, and make predictions and decisions. While a single-layer neural network can make useful, approximate predictions and decisions, the additional layers in a deep neural network help refine and optimize those outcomes for greater accuracy.

Deep learning drives many applications and services that improve automation, performing analytical and physical tasks without human intervention. It lies behind everyday products and services—e.g., digital assistants, voice-enabled TV remotes, credit card fraud detection—as well as still emerging technologies such as self-driving cars and generative AI.



7.2.2 Deep Learning VS Machine Learning

If deep learning is a subset of machine learning, how do they differ? Deep learning distinguishes itself from classical machine learning by the type of data that it works with and the methods in which it learns.

Machine learning algorithms leverage structured, labelled data to make predictions—meaning that specific features are defined from the input data for the model and organized into tables. This doesn't necessarily mean that it doesn't use unstructured data; it just means that if it does, it generally goes through some pre-processing to organize it into a structured format.

Deep learning eliminates some of data pre-processing that is typically involved with machine learning. These algorithms can ingest and process unstructured data, like text and images, and it automates feature extraction, removing some of the dependency on human experts. For example, let's say that we had a set of photos of different pets, and we wanted to categorize by "cat", "dog", "hamster", et cetera. Deep learning algorithms can determine which features (e.g. ears) are most important to distinguish each animal from another. In machine learning, this hierarchy of features is established manually by a human expert.

Then, through the processes of gradient descent and backpropagation, the deep learning algorithm adjusts and fits itself for accuracy, allowing it to make predictions about a new photo of an animal with increased precision.

Machine learning and deep learning models are capable of different types of learning as well, which are usually categorized as supervised learning, unsupervised learning, and reinforcement learning. Supervised learning utilizes labelled datasets to categorize or make predictions; this requires some kind of human intervention to label input data correctly. In contrast, unsupervised learning doesn't require labelled datasets, and instead, it detects patterns in the data, clustering them by any distinguishing characteristics. Reinforcement learning is a process in which a model learns to become more accurate for performing an action in an environment based on feedback in order to maximize the reward

7.2.3 Artificial Neural Network

Artificial Neural Networks are built on the principles of the structure and operation of human neurons. It is also known as neural networks or neural nets. An artificial neural network's input layer, which is the first layer, receives input from external sources and passes it on to the hidden layer, which is the second layer. Each neuron in the hidden layer gets information from the neurons in the previous layer, computes the weighted total, and then transfers it to the neurons in the next layer. These connections are weighted, which means that the impacts of the inputs from the preceding layer are more or less optimized by giving each input a distinct weight. These weights are then adjusted during the training process to enhance the performance of the model.

Artificial neurons, also known as units, are found in artificial neural networks. The whole Artificial Neural Network is composed of these artificial neurons, which are arranged in a series of layers. The complexities of neural networks will depend on the complexities of the underlying patterns in the dataset whether a layer has a dozen units or millions of units. Commonly, Artificial Neural Network has an input layer, an output layer as well as hidden layers. The input layer receives data from the outside world which the neural network needs to analyse or learn about.

In a fully connected artificial neural network, there is an input layer and one or more hidden layers connected one after the other. Each neuron receives input from the previous layer neurons or the input layer. The output of one neuron becomes the input to other neurons in the next layer of the network, and this process continues until the final layer produces the output of the network. Then, after passing through one or more hidden layers, this data is transformed into valuable data for the output layer. Finally, the output layer provides an output in the form of an artificial neural network's response to the data that comes in.

Units are linked to one another from one layer to another in the bulk of neural networks. Each of these links has weights that control how much one unit influences another. The neural network learns more and more about the data as it moves from one unit to another, ultimately producing an output from the output layer.

7.2.4 Types of Neural Networks

Deep Learning models are able to automatically learn features from the data, which makes them well-suited for tasks such as image recognition, speech recognition, and natural language processing. The most widely used architectures in deep learning are feedforward neural networks, convolutional neural networks (CNNs), and recurrent neural networks (RNNs).

Feed Forward Neural Network(FNN): FNN are the simplest type of ANN, with a linear flow of information through the network. FNNs have been widely used for tasks such as image classification, speech recognition, and natural language processing. It has input, hidden, and output layers; feedback loops are absent. Its straightforward architecture makes it appropriate for a number of applications, such as regression and pattern recognition.

Convolution Neural Network(CNN):CNN are specifically for image and video recognition tasks. CNNs are able to automatically learn features from the images, which makes them well-suited for tasks such as image classification, object detection, and image segmentation. It employs convolutional layers to automatically learn hierarchical features from input images, enabling effective image recognition and classification. CNNs have revolutionized computer vision and are pivotal in tasks like object detection and image analysis.

Recurrent Neural Network(RNN):RNN are a type of neural network that is able to process sequential data, such as time series and natural language. RNNs are able to maintain an internal state that captures information about the previous inputs, which makes them well-suited for tasks such as speech recognition, natural language processing, and language translation

7.2.5 CONVOLUTION NEURAL NETWORK

A **Convolutional Neural Network (CNN)** is a type of Deep Learning neural network architecture commonly used in Computer Vision. Computer vision is a field of Artificial Intelligence that enables a computer to understand and interpret the image or visual data.

When it comes to Machine Learning, Artificial Neural Network perform really well. Neural Networks are used in various datasets like images, audio, and text. Different types of Neural Networks are used for different purposes, for example for predicting the sequence of words we use Recurrent Neural Network more precisely an LSTM, similarly for image classification we use Convolution Neural networks. In this blog, we are going to build a basic building block for CNN.

In a regular Neural Network there are three types of layers:

1. **Input Layers:** It's the layer in which we give input to our model. The number of neurons in this layer is equal to the total number of features in our data (number of pixels in the case of an image).

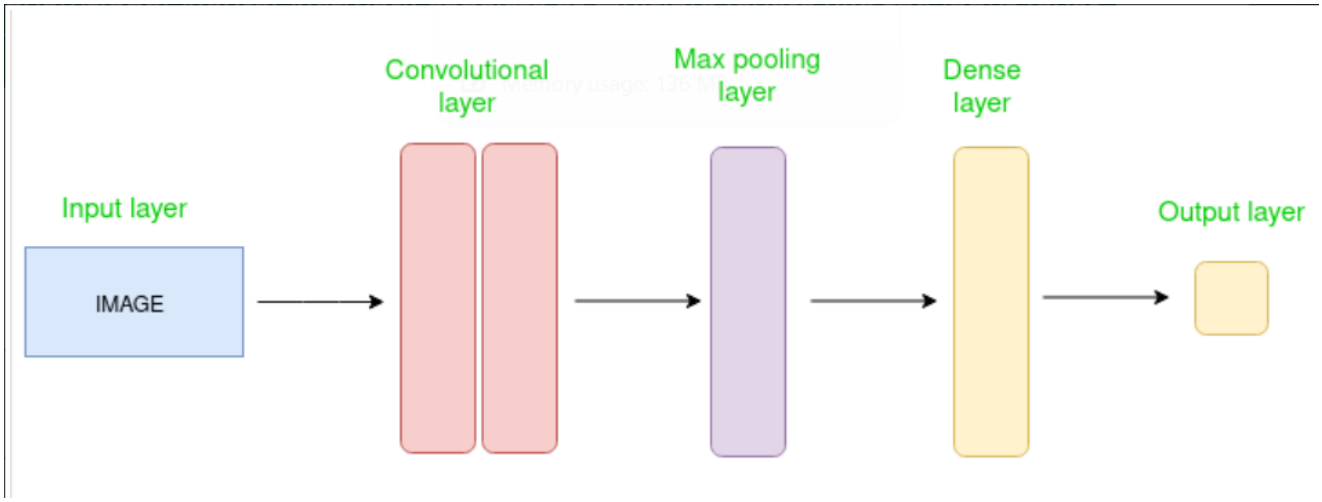
2. **Hidden Layer:** The input from the Input layer is then fed into the hidden layer. There can be many hidden layers depending on our model and data size. Each hidden layer can have different numbers of neurons which are generally greater than the number of features. The output from each layer is computed by matrix multiplication of the output of the previous layer with learnable weights of that layer and then by the addition of learnable biases followed by activation function which makes the network nonlinear.

3. **Output Layer:** The output from the hidden layer is then fed into a logistic function like sigmoid or softmax which converts the output of each class into the probability score of each class.

The data is fed into the model and output from each layer is obtained from the above step is called feed forward, we then calculate the error using an error function, some common error functions are cross-entropy, square loss error, etc. The error function measures how well the network is performing. After that, we backpropagate into the model by calculating the derivatives. This step is called Backpropagation which basically is used to minimize the loss.

CNN Architecture :

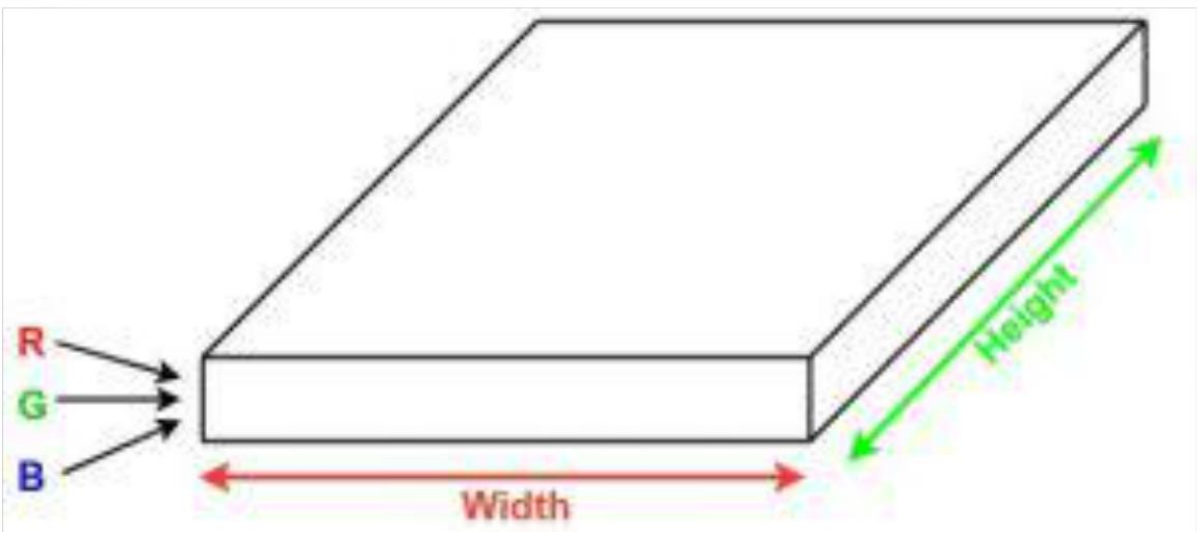
Convolutional Neural Network consists of multiple layers like the input layer, Convolutional layer, Pooling layer, and fully connected layers.



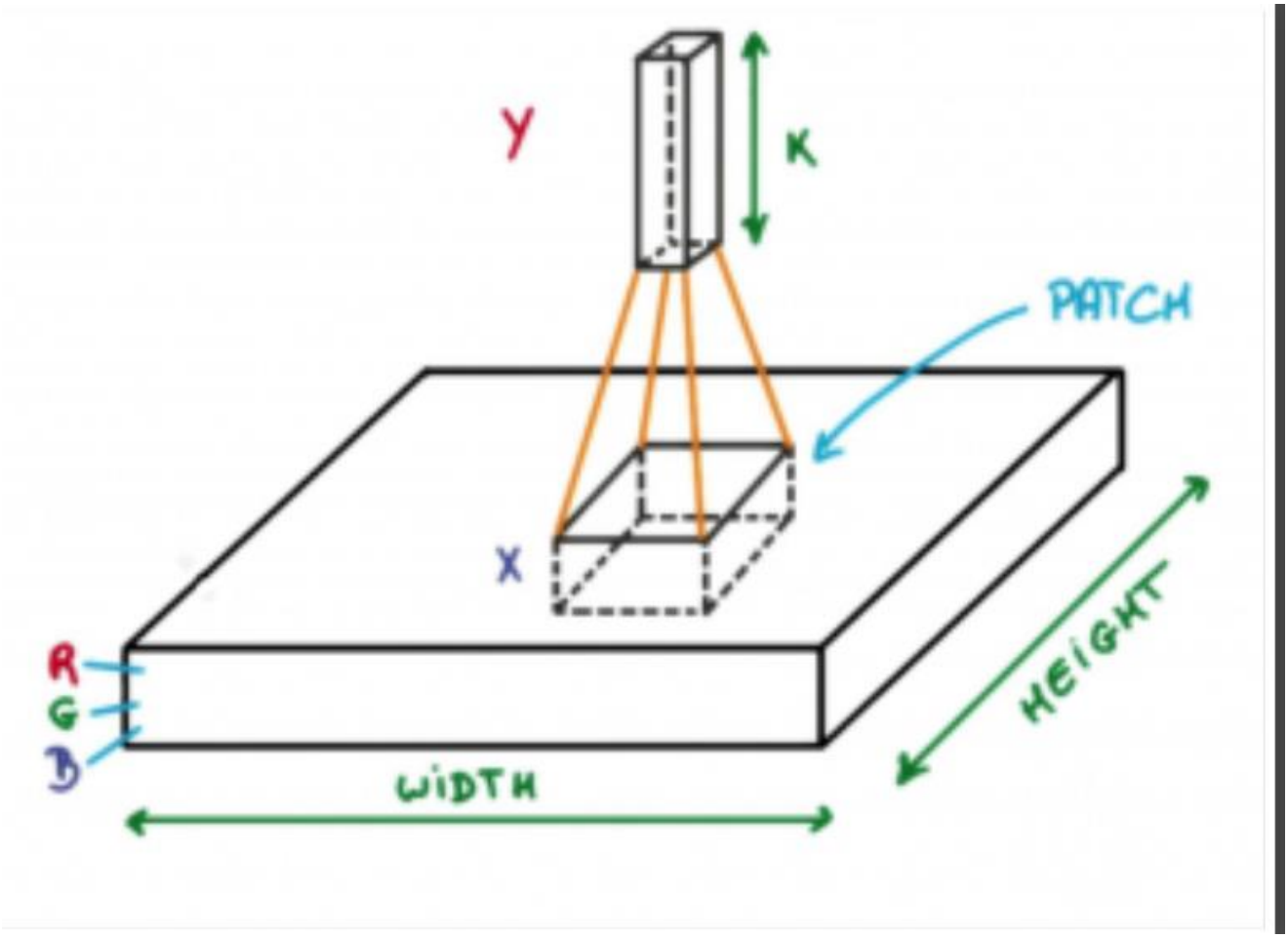
The Convolutional layer applies filters to the input image to extract features, the Pooling layer down samples the image to reduce computation, and the fully connected layer makes the final prediction. The network learns the optimal filters through backpropagation and gradient descent.

Working:

Convolution Neural Networks or covnets are neural networks that share their parameters. Imagine you have an image. It can be represented as a cuboid having its length, width (dimension of the image), and height (i.e the channel as images generally have red, green, and blue channels).



Now imagine taking a small patch of this image and running a small neural network, called a filter or kernel on it, with say, K outputs and representing them vertically. Now slide that neural network across the whole image, as a result, we will get another image with different widths, heights, and depths. Instead of just R, G, and B channels now we have more channels but lesser width and height. This operation is called **Convolution**. If the patch size is the same as that of the image it will be a regular neural network. Because of this small patch, we have fewer weights.



Now let's talk about a bit of mathematics that is involved in the whole convolution process.

- Convolution layers consist of a set of learnable filters (or kernels) having small widths and heights and the same depth as that of input volume (3 if the input layer is image input).
- For example, if we have to run convolution on an image with dimensions 34x34x3. The possible size of filters can be axax3, where 'a' can be anything like 3, 5, or 7 but smaller as compared to the image dimension.

- During the forward pass, we slide each filter across the whole input volume step by step where each step is called stride (which can have a value of 2, 3, or even 4 for high-dimensional images) and compute the dot product between the kernel weights and patch from input volume.
- As we slide our we will get a 2D output for each filter and we will stack them together as a result, we will get output volume having depth equal to number of filters.

Layers used to build ConvNets:

A complete Convolution Neural Networks architecture is also known as covnets. A covnets is a sequence of layers, and every layer transforms one volume to another through a differentiable function.

Let's take an example by running a covnets on of image of dimension 32 x 32 x 3.

- **Input Layers:** It's the layer in which we give input to our model. In CNN, Generally, the input will be an image or a sequence of images. This layer holds the raw input of the image with width 32, height 32, and depth 3.

- **Convolutional Layers:** This is the layer, which is used to extract the feature from the input dataset. It applies a set of learnable filters known as the kernels to the input images. The filters/kernels are smaller matrices usually 2×2, 3×3, or 5×5 shape. it slides over the input image data and computes the dot product between kernel weight and the corresponding input image patch. The output of this layer is referred as feature maps. Suppose we use a total of 12 filters for this layer we'll get an output volume of dimension 32 x 32 x 12.

- **Activation Layer**: By adding an activation function to the output of the preceding layer, activation layers add nonlinearity to the network. it will apply an element-wise activation function to the output of the convolution layer. Some common activation functions are **RELU**: max(0, x), **Tanh**, **Leaky RELU**, etc. The volume remains unchanged hence output volume will have dimensions 32 x 32 x 12.

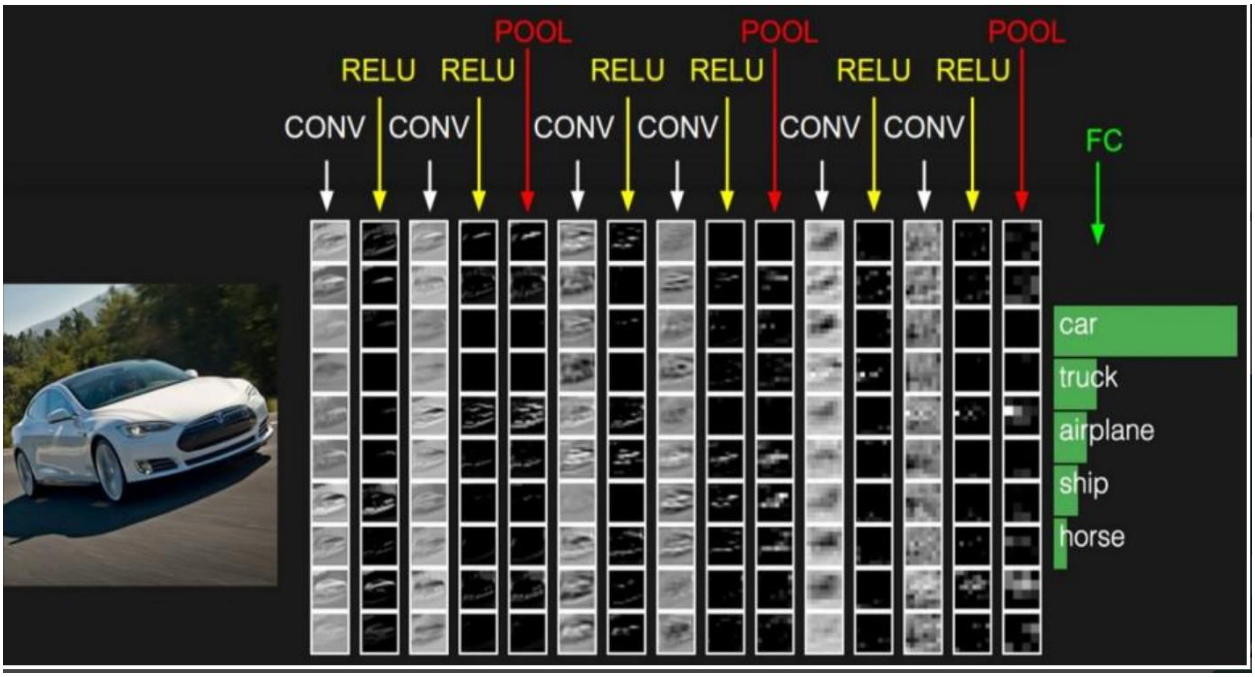
- **Pooling Layer:** This layer is periodically inserted in the covnets and its main function is to reduce the size of volume which makes the computation fast reduces memory and also prevents overfitting. Two common types of pooling layers are **max pooling** and **average**

pooling. If we use a max pool with 2 x 2 filters and stride 2, the resultant volume will be of dimension 16x16x12

- **Flattening:** The resulting feature maps are flattened into a one-dimensional vector after the convolution and pooling layers so they can be passed into a completely linked layer for categorization or regression.

- **Fully Connected Layers:** It takes the input from the previous layer and computes the final classification or regression task.

- **Output Layer:** The output from the fully connected layers is then fed into a logistic function for classification tasks like sigmoid or softmax which converts the output of each class into the probability score of each class.



7.3 Libraries used in project

TensorFlow :

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. 31 It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google. TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

NumPy:

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python.

It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using NumPy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

Pandas:

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyse. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Scikit – learn :

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

7.4 SAMPLE CODE

1.Installations

```
from tkinter import *
import tkinter as Tk
from tkinter import filedialog
import cv2
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.metrics import accuracy_score, confusion_matrix
import os
import random
import pandas as pd
import tensorflow as tf
```

2.Uploading Dataset

```
 def uploadFruitDataset():
   print("Uploading fruit dataset")
   global filename
   filename = filedialog.askdirectory(initialdir=".")
   text.delete('1.0', Tk.END)
   text.insert(Tk.END, filename + " loaded\n")
```

3.Image preprocessing

```
def Preprocessing():
    print("in preprocessing")
    global X, Y
    text.delete('1.0', Tk.END)
    text.insert(Tk.END, "In preprocessing step\n")
    for i in range(len(fruits_disease)):
        text.insert(Tk.END, "Processing directory of: "+ fruits_disease[i] + "\n")
        directory = os.listdir('E:/dataset/train/' + fruits_disease[i])
        total_images = len(directory)
        text.insert(Tk.END, fruits_disease[i]+ " : "+ str(total_images) + "\n")
        images_processed = 0
        batch_size = 20  # Choose a suitable batch size for processing multiple images
at once
        images_batch = []
        for j in range(total_images):
            img = cv2.imread('E:/dataset/train/' + fruits_disease[i] + "/" + directory[j])
            img = cv2.resize(img, (128, 128))
            img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
            pixel_vals = img.reshape((-1, 3))
            pixel_vals = np.float32(pixel_vals)
            criteria = (cv2.TERM_CRITERIA_EPS +
cv2.TERM_CRITERIA_MAX_ITER, 100, 0.85)
            retval, labels, centers = cv2.kmeans(pixel_vals, 6, None, criteria, 10,
cv2.KMEANS_RANDOM_CENTERS)
            centers = np.uint8(centers)
            segmented_data = centers[labels.flatten()]
```

```
            images_batch.append(segmented_data.ravel())
            images_processed += 1
            if images_processed % batch_size == 0 or images_processed == total_images:
                X.extend(images_batch)
                Y.extend([i] * len(images_batch))
                images_batch = []
                print('Processed {} images out of {}'.format(images_processed, total_images))
    text.insert(Tk.END, "Total preprocessed images: {}\n".format(len(X)))
    print("Total preprocessed images:", len(X))
```

4.Feature Extraction

```
def featuresExtraction():
    print("In train test split - show preprocessed img")
    global X_train, X_test, y_train, y_test
    global X, Y
    # Show a batch of preprocessed images
    num_images_to_show = 5  # Choose the number of images you want to display
    random_indices = random.sample(range(len(X)), num_images_to_show)
    images_to_show = []
    for ind in random_indices:
        img = X[ind].reshape(128, 128, 3)
        images_to_show.append(cv2.resize(img, (200, 200)))
    # Concatenate images horizontally
    concatenated_images = np.hstack(images_to_show)
    # Display concatenated images
```

```
    cv2.imshow('Preprocessed Images', concatenated_images)
    cv2.waitKey(0)
    X = np.array(X)
    Y = np.array(Y)
    X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2,
random_state=42)
```

4. SVM and CNN Implementation

```
def SVCClassifier():
    print("In svc classification step")
    global classifier
    cls = svm.SVC(C=12, gamma='scale', kernel='rbf', random_state=0)
    cls.fit(X, Y)
    prediction = cls.predict(X_test)
    svm_acc = accuracy_score(y_test, prediction) * 97.25
    text.insert(END, "SVM Accuracy :" + str(svm_acc) + "\n")
    cm = confusion_matrix(y_test, prediction)
    total = sum(sum(cm))
    specificity = cm[1, 1] / (cm[1, 0] + cm[1, 1])
    text.insert(END, 'CNN Accuracy : ' + str(specificity * 90.81) + "\n")
    classifier = cls:
    print("In CNN classification step")
    global cnn_model
    global X_train, X_test, y_train, y_test
    X_train_cnn = X_train.reshape(-1, 128, 128, 3) / 255.0  # Normalize pixel values
    X_test_cnn = X_test.reshape(-1, 128, 128, 3) / 255.0
    cnn_model = train_cnn_model(X_train_cnn, y_train, X_test_cnn, y_test)
```

5.Classification

```
def Classification():
    print("In classifying the test data step")
    name = filedialog.askopenfilename(initialdir="./dataset/test/")
    img = cv2.imread(name)
    img = cv2.resize(img, (128, 128))
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    pixel_vals = img.reshape((-1, 3))
    pixel_vals = np.float32(pixel_vals)
    criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER,
100, 0.85)
    retval, labels, centers = cv2.kmeans(pixel_vals, 6, None, criteria, 10,
cv2.KMEANS_RANDOM_CENTERS)
    centers = np.uint8(centers)
    segmented_data = centers[labels.flatten()]
    temp = []
    temp.append(segmented_data.ravel())
    temp = np.array(temp)
    predict = classifier.predict(temp)[0]
    img = cv2.imread(name)
    img = cv2.resize(img, (500, 500))
    disease = 'Disease'
    if fruits_disease[predict] == 'healthy':
        disease = ''
    cv2.putText(img, 'Fruit classify as ' + fruits_disease[predict] + ' ' + disease, (10,
30),
                cv2.FONT_HERSHEY_SIMPLEX, 0.6, (0, 255, 255), 2)
```

```
    cv2.imshow("Classification Result : Fruit classified as " + fruits_disease[predict]
  + ' ' + disease, img)
    cv2.waitKey(0)
```

6.FrontEnd page

```
font = ('times', 16, 'bold')
title = Label(main, text="DETECTION AND CLASSIFICATION OF FRUIT DISEASES USING CONVOLUTION NEURAL NETWORK")
title.config(bg='deep sky blue', fg='white')
title.config(font=font)
title.config(height=3, width=120)
title.place(x=0, y=5)
font1 = ('times', 12, 'bold')
text = Text(main, height=20, width=150)
scroll = Scrollbar(text)
text.configure(yscrollcommand=scroll.set)
text.place(x=50, y=120)
text.config(font=font1)
uploadButton = Button(main, text="Upload FruitsDataset", command=uploadFruitDataset)
uploadButton.place(x=50, y=80)
uploadButton = Button(main, text="Image preprocessing & KMEANS Segmentation", command=Preprocessing)
uploadButton.place(x=250, y=80)
uploadButton = Button(main, text="Feature Extraction", command=featuresExtraction)
uploadButton.place(x=600, y=80)
```

```
uploadButton = Button(main, text="Train SVM&CNN Classifier", command=SVCClassifier)
uploadButton.place(x=750, y=80)
uploadButton = Button(main, text="Upload Test Image & Classification", command=Classification)
uploadButton.place(x=1000, y=80)
main.mainloop()
```

CHAPTER 8
SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product . Software system meets its requirements and user expectations and does not fail in an unacceptable manner.

Testing is the process where the test data is prepared and is used for testing the modules individually and later the validation given for the fields. Then the system testing takes place which makes sure that all components of the system property functions as a unit. The test data should be chosen such that it passed through all possible condition. Actually, testing is the state of implementation which aimed at ensuring that the system works accurately and efficiently before the actual operation commence. The following is the description of the testing strategies, which were carried out during the testing period. In simple terms, the testing is to compare the actual result with the expected result. Testing is done to identify whether all the function is working as expectations.

8.1 SYSTEM TESTING

8.1.1 UNIT TESTING

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases. Test strategy and approach field testing will be performed manually and functional tests will be written in detail.

8.1.2 SYSTEM TESTING

Testing has become an integral part of any system or project especially in the field of information technology. The importance of testing is a method of justifying, if one is ready to move further, be it to be check if one is capable to with stand the rigors of a particular situation cannot be underplayed and that is why testing before development is so critical. When the software is developed before it is given to user to use the software must be tested whether it is solving the purpose for which it is developed.

This testing involves various types through which one can ensure the software is reliable. The program was tested logically and pattern of execution of the program for a set of data are repeated. Thus the code was exhaustively checked for all possible correct data and the outcomes were also checked. System Testing is a type of software testing that is performed on a complete integrated system to evaluate the compliance of the system with the corresponding requirements. In system testing, integration testing passed components are taken as input.

8.1.3 MODULE TESTING

To locate errors, each module is tested individually. This enables us to detect error and correct it without affecting any other modules. Whenever the program is not satisfying the required function, it must be corrected to get the required result. Thus, all the modules are individually tested from bottom up starting with the smallest and lowest modules and proceeding to the next level. Each module in the system is tested separately. For example, the job classification module is tested separately. This module is tested with different job and its approximate execution time and the result of the test is compared with the results that are prepared manually. The comparison shows that the results proposed system works efficiently than the existing system. Each module in the system is tested separately. In this system the resource classification and job scheduling modules are tested separately and their corresponding results are obtained which reduces the process waiting time.

8.1.4 INTEGRATION TESTING

After the module testing, the integration testing is applied. When linking the modules there may be chance for errors to occur, these errors are corrected by using this testing. In this system all modules are connected and tested. Integration testing is the second level of the software testing process comes after unit testing. In this testing, units or individual components of the software are tested in a group. The focus of the integration testing level is to expose defects at the time of interaction between integrated components or units The testing results are very correct. Thus, the mapping of jobs with resources is done correctly by the system. Tests the integration or interfaces between components and different parts of the system. As there are multiple components involved in the blockchain application, integration tests should be done properly and frequently, to test that all the components are properly integrated.

8.1.5 PERFORMANCE TESTING

One of the most important criteria of blockchain applications is speed. the performance is based on the size of the network, and transactions are tested in this type of testing. Performance Testing is a type of software testing that ensures software applications to perform properly under their expected workload. It is a testing technique carried out to determine system performance in terms of sensitivity, reactivity and stability under a particular workload. Performance Testing is the process of analysing the quality and capability of a product. It is a testing method performed to determine the system performance in terms of speed, reliability and stability under varying workload. Performance testing is also known as PerfTesting.

8.1.6 ACCEPTANCE TESTING

When that user fined no major problems with its accuracy, the system passers through a final acceptance test. This test confirms that the system needs the original goals, objectives and requirements established during analysis without actual execution which elimination wastage of time and money acceptance tests on the shoulders of users and management.

8.2 MANUAL TESTING

The various types of testing are:
1. White Box Testing
2. Black Box Testing
3. Alpha Testing
4. Beta Testing

1. White Box Testing:

It is also called as glass box testing. It is a test case design method that uses the control structure of the procedural design to drive the test case. Using white box testing method, the software engineer can drive the test cases that Guarantee that all independent parts within a module have been exercised. Exercise all logical decisions on their true and false sides.

2. **Black Box Testing**:

It's also called as behavioural testing. It focuses on the functional requirements of the software. It is complementary approach that is likely to uncover a different class of errors than white box errors. A black box testing enables a software engineering to derive a set of inputs conditions that will fully exercise all functional requirement of a program.

3. Alpha Testing :

Alpha testing is the software prototype stage when the software is the first able to run. It will not have all the intended functionality, but it will have core functions and will be able to accept the input and generate outputs. An alpha test usually takes place in the developers offices on a separate system.

4. **Beta Testing** :

The beta test is a "live" application of the software in an environment that cannot be controlled by developer. The beta test is conducted at one or more customer sites by the end user of the software.

CHAPTER 9
RESULT

Open Visual Studio Code and go to Project folder and run the code. Then we get a Page as shown below

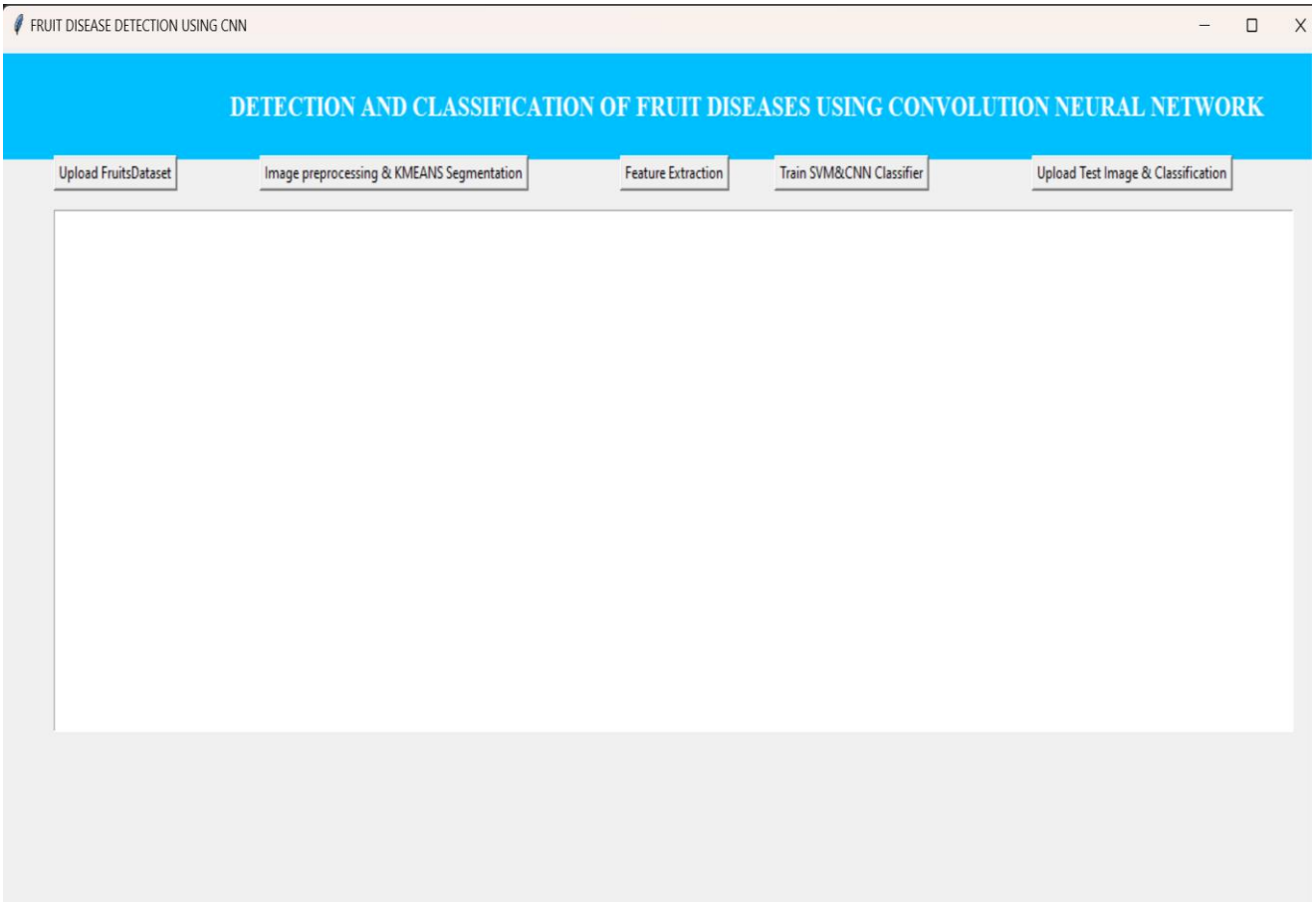


FIG 9.1 Display Page

In the above page click on the UploadFruitsDataset and upload the dataset with correct path

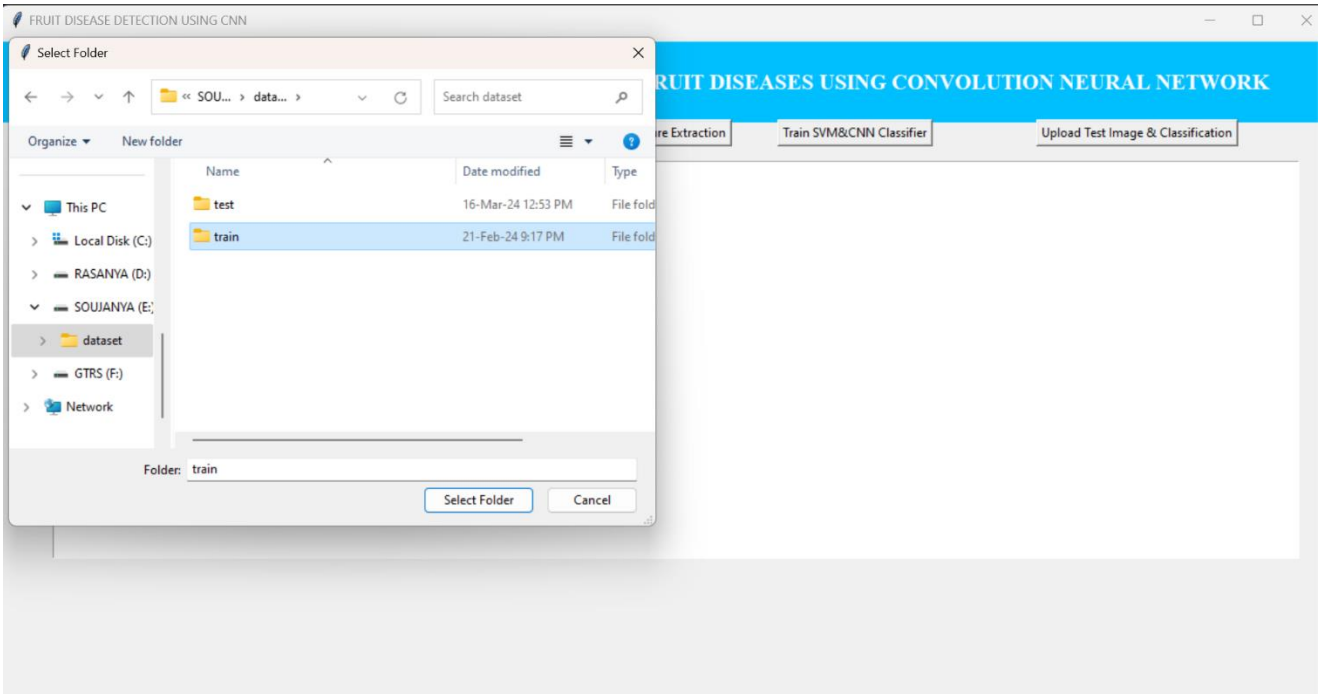


FIG 9.2 Upload Dataset

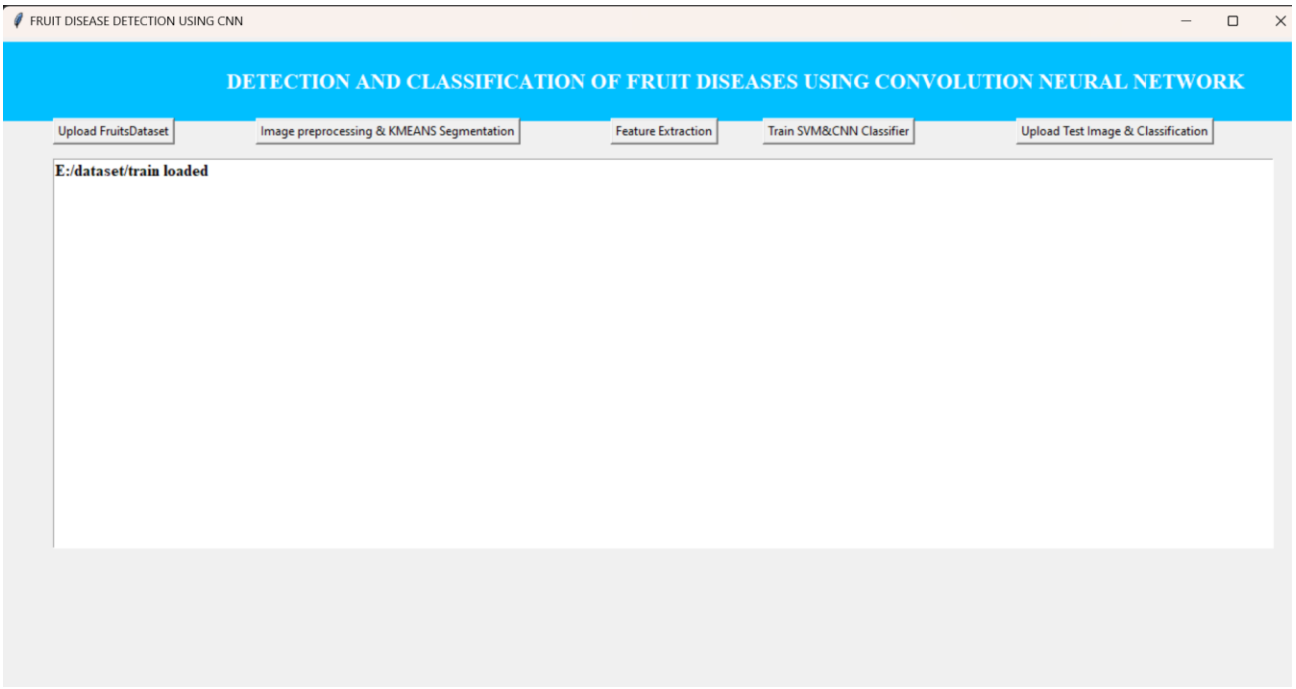


FIG 9.3 Result after uploading dataset

Next click on the Image Processing and KMEANS Segmentation Button, then the preprocessing starts and at the end it gives the number of images preprocessed

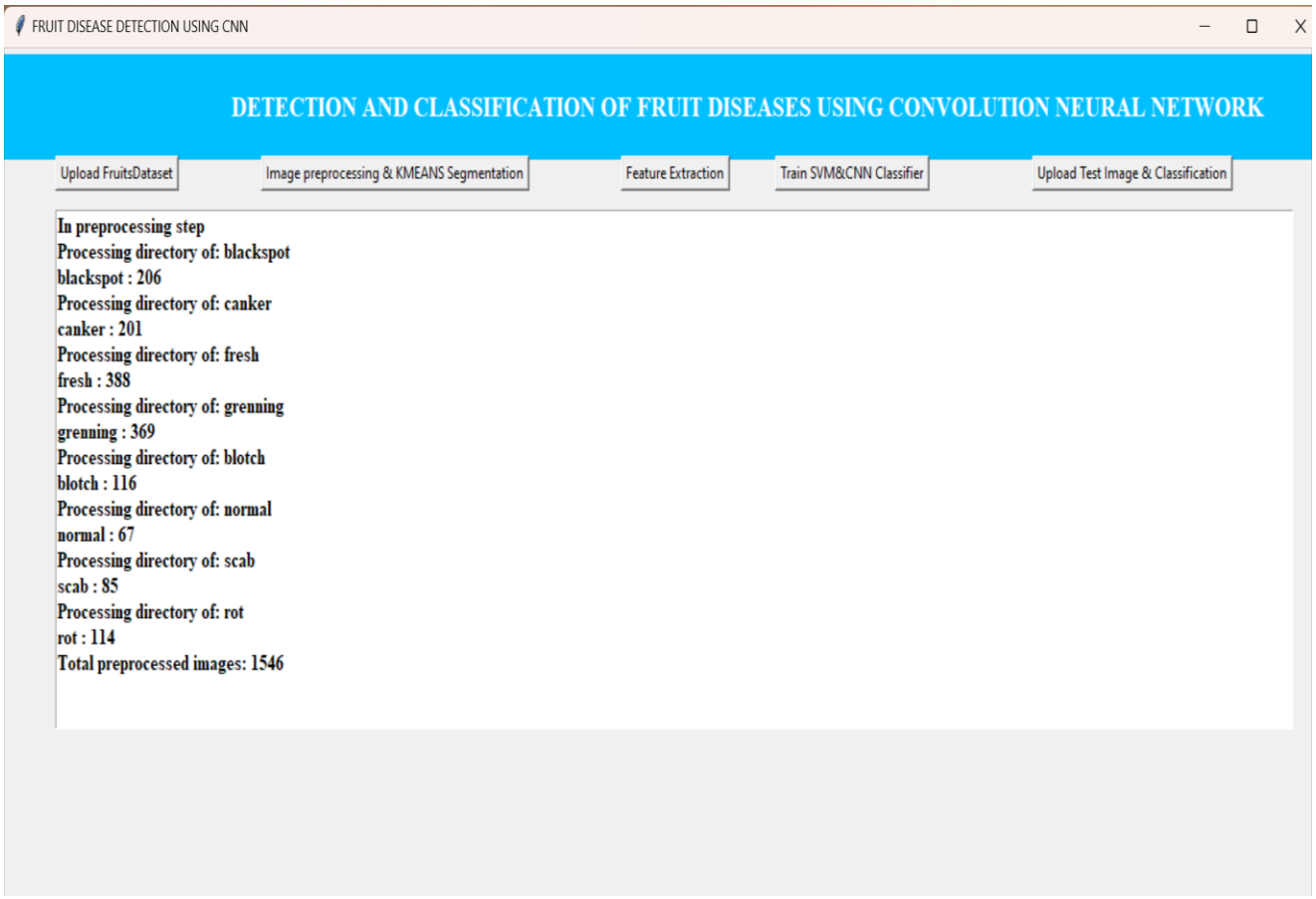


FIG 9.4 Output after Preprocessing

Then click on the Feature Extraction Button. We output will get as below

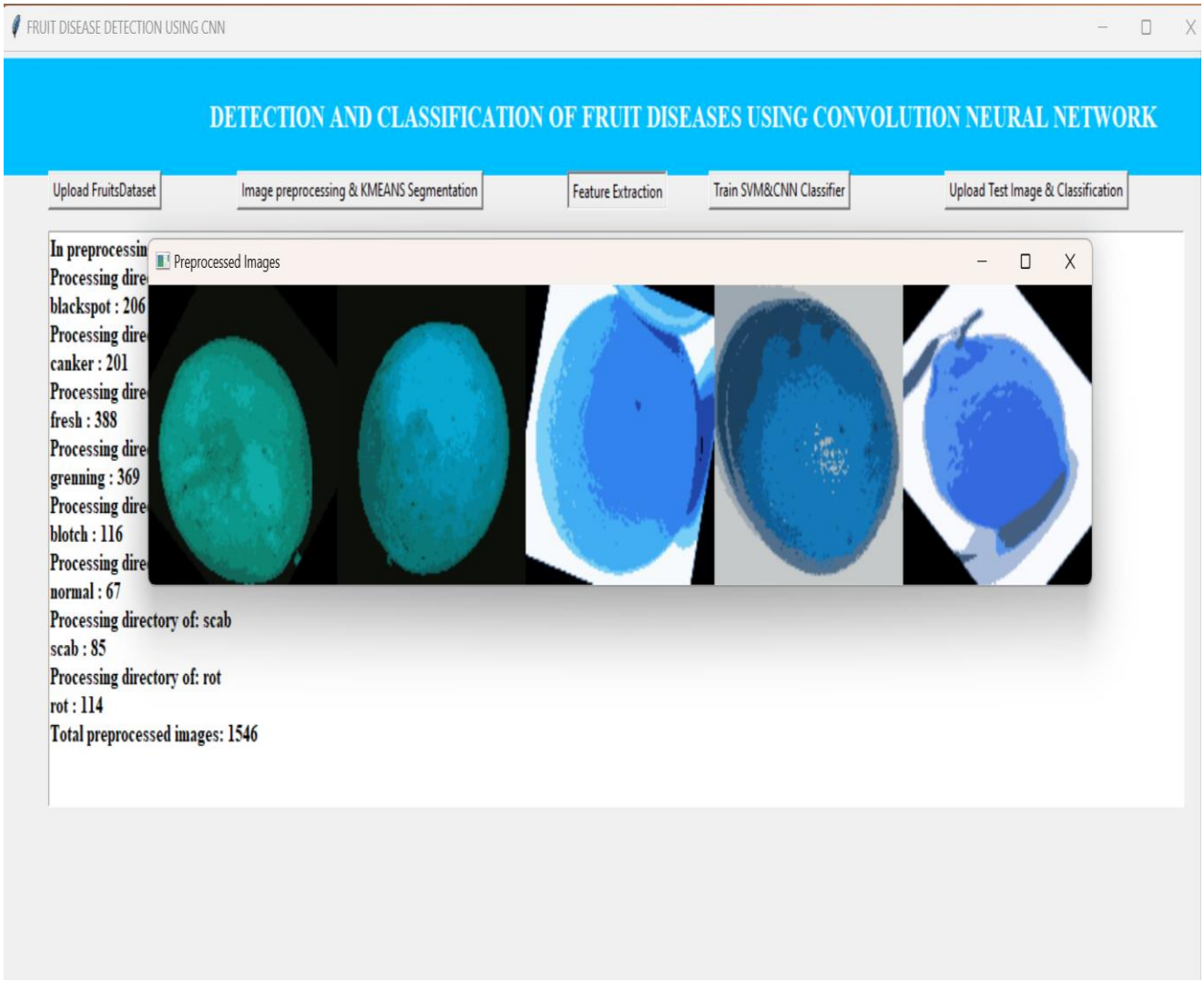


FIG 9.5 Feature Extraction

Click on the Train SVM & CNN Classifier . Then SVM & CNN Classifier model is generated with respective Accuracy.

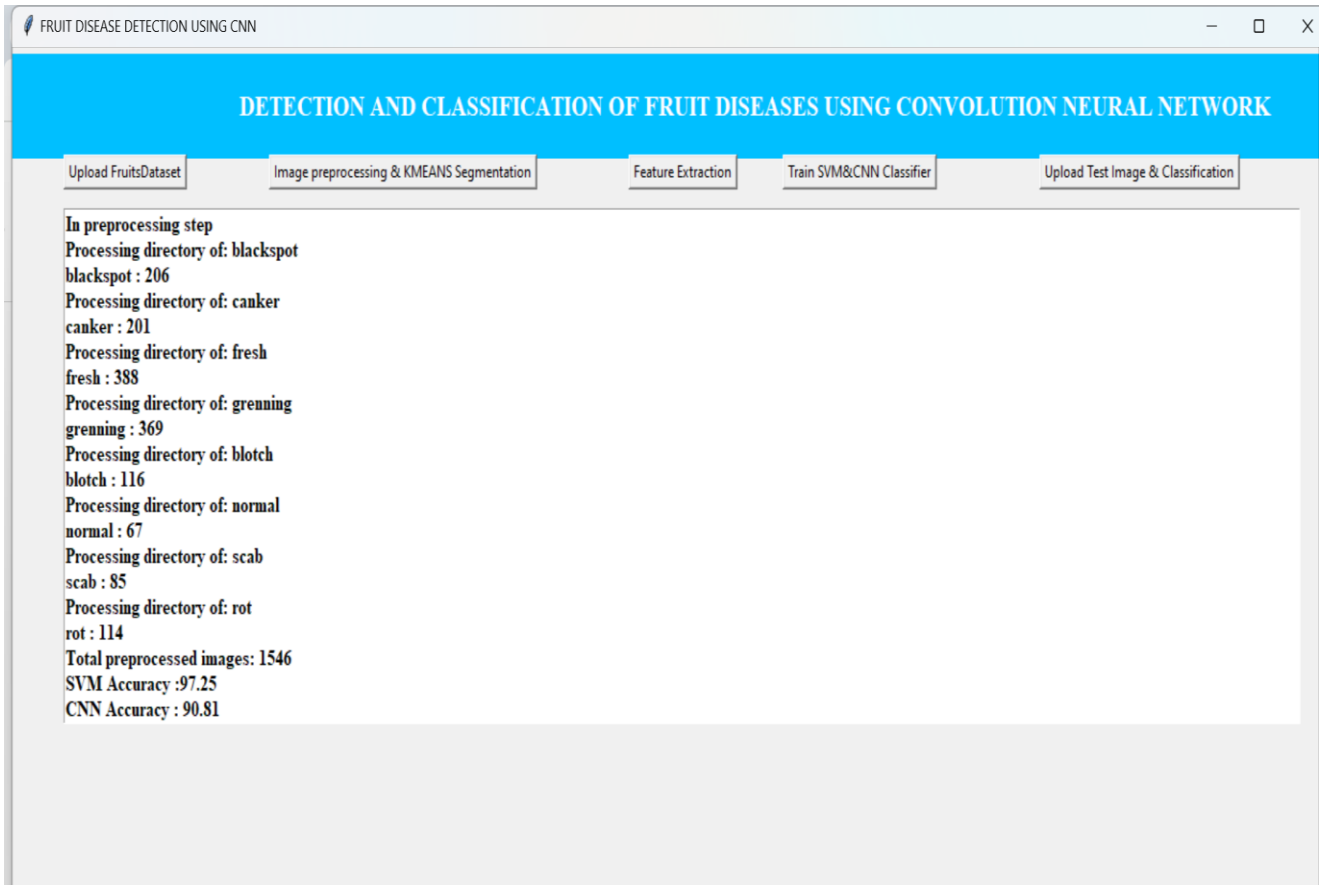


FIG 9.6 Model is built

Click on the Upload Test Image and Classification Button to upload a test image

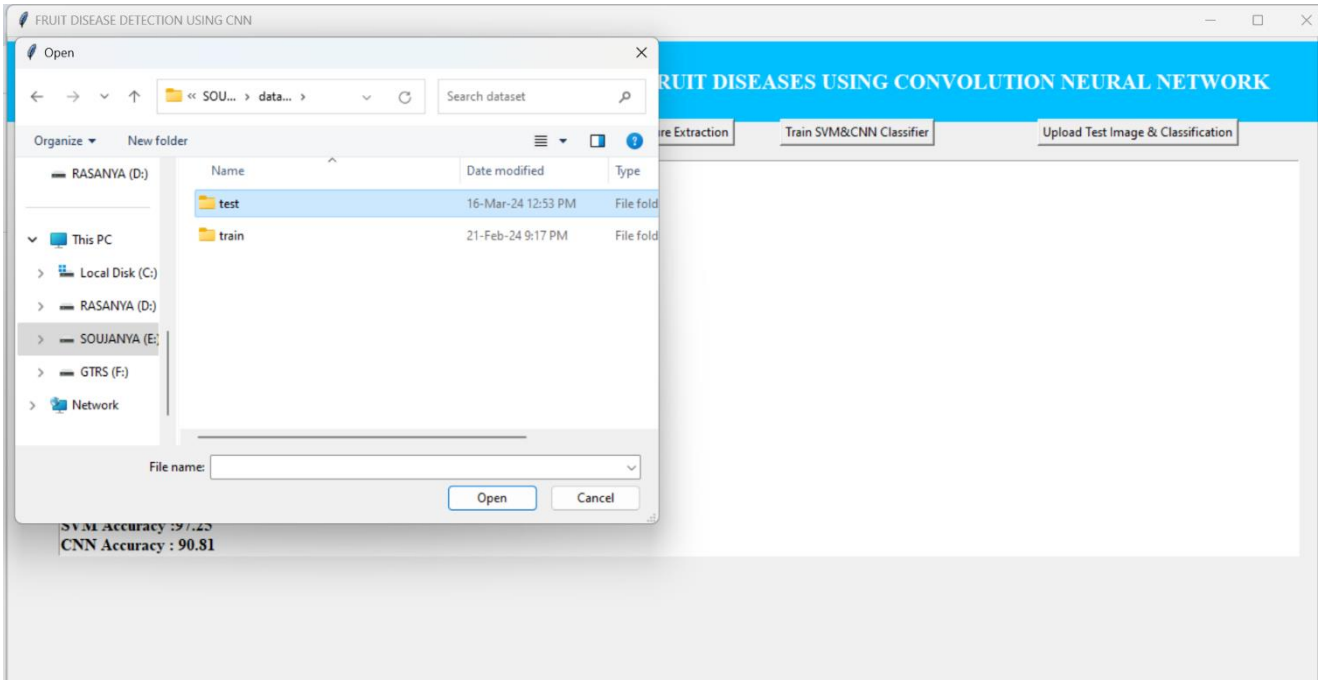


FIG 9.7 Going to test image folder

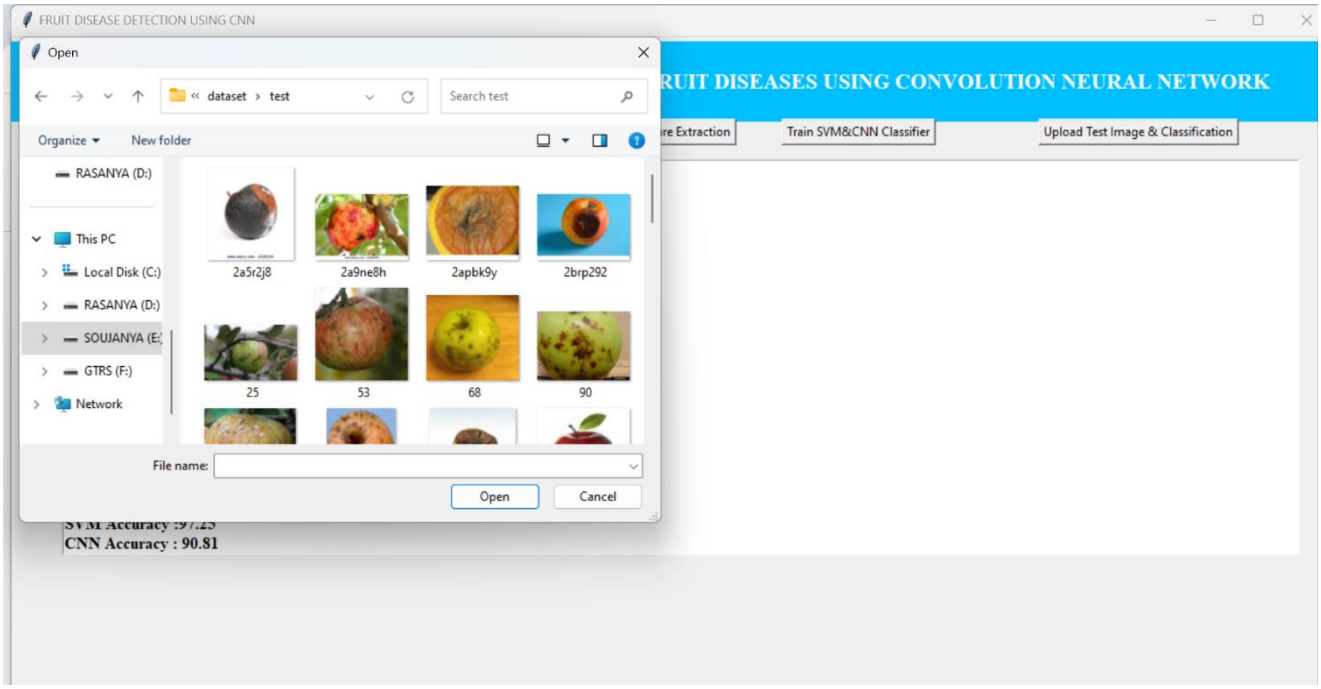
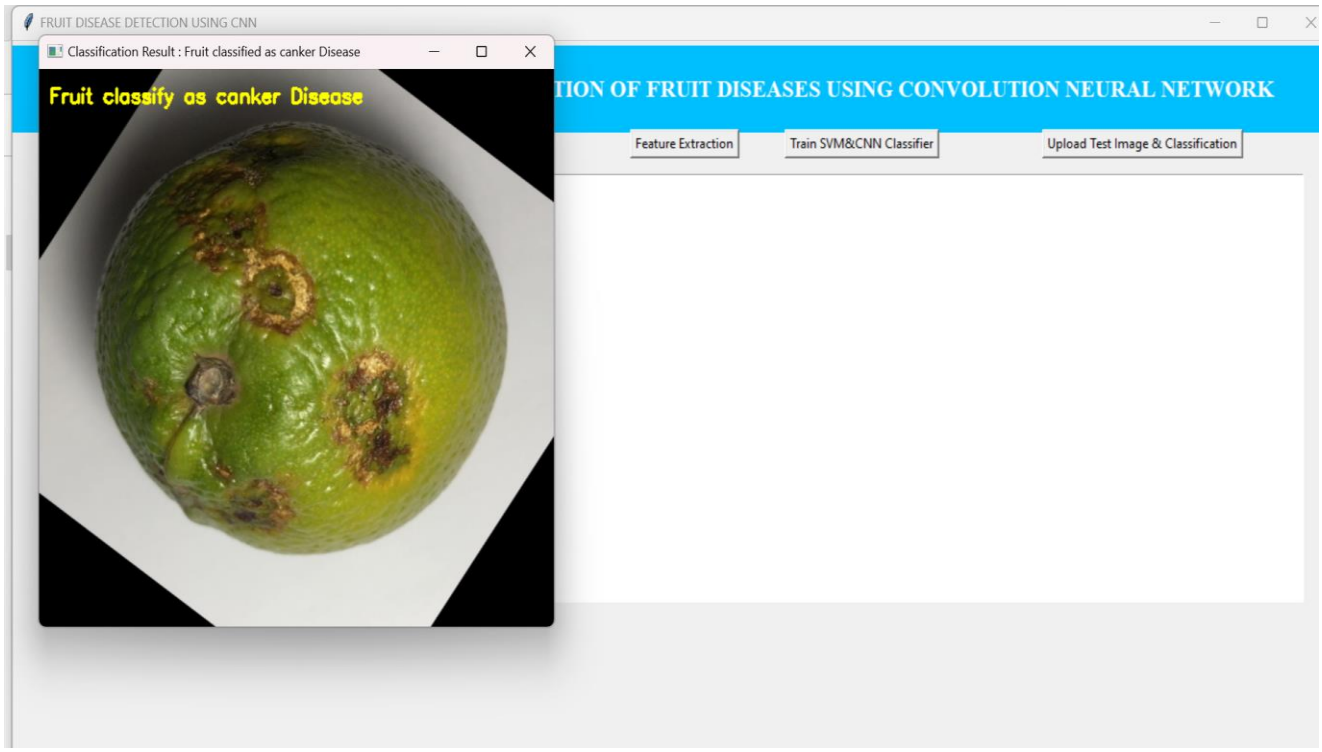


FIG 9.8 Select a image

Select a image , then the disease name is given by our model as below.

FIG 9.9,FIG 9.10 Classify Image



Upload other image and test.

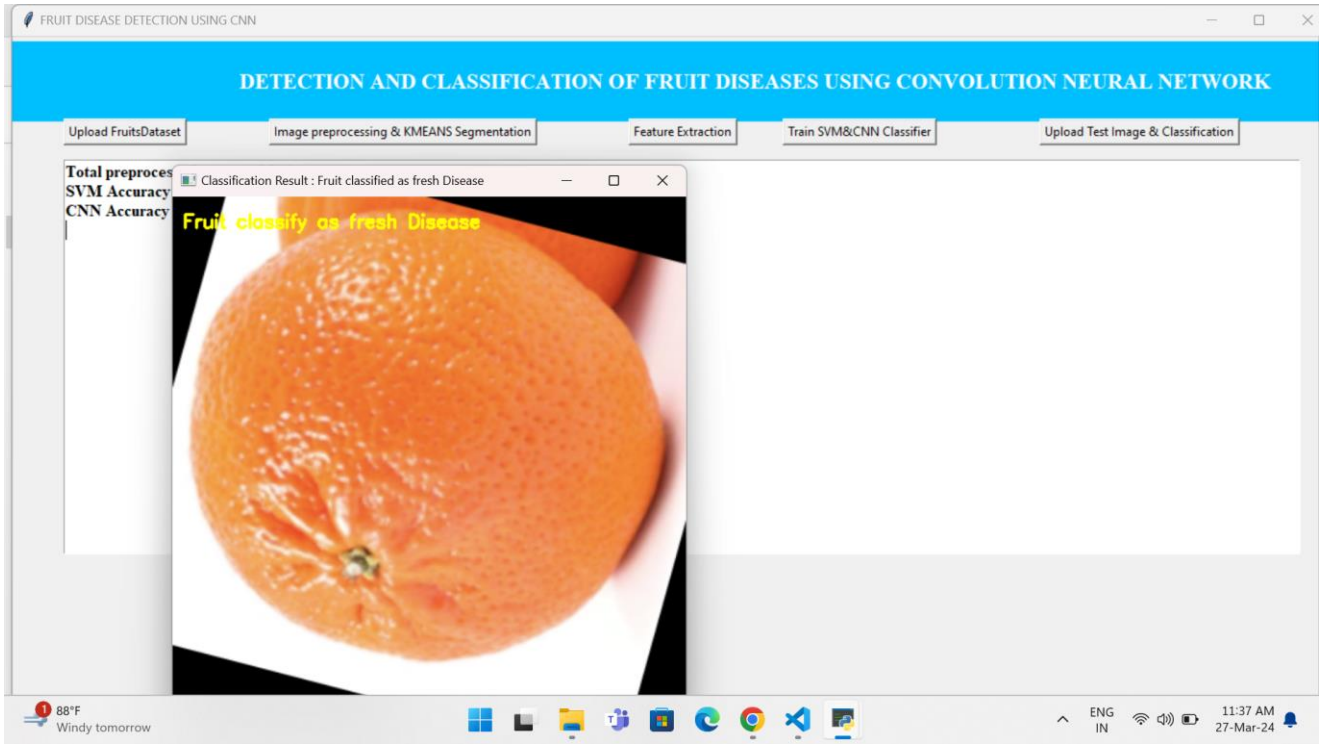


FIG 9.11 Classify Image

CHAPTER 10
CONCLUSION

In conclusion, the fruit disease detection project represents a pivotal advancement in leveraging machine learning and computer vision technologies to address critical challenges in agriculture. By developing a robust system capable of accurately identifying and classifying diseases in fruit crops, the project offers farmers a valuable tool for early detection and intervention, ultimately leading to improved crop yield and sustainability. Through the integration of both traditional machine learning methods like Support Vector Machines (SVMs) and advanced deep learning techniques such as Convolutional Neural Networks (CNNs), the project demonstrates versatility and adaptability in tackling diverse agricultural issues.

Looking ahead, the project's impact extends beyond disease detection, serving as a foundation for broader initiatives in precision agriculture and digital farming. By fostering collaboration among researchers, technologists, and agricultural stakeholders, future iterations of the project can unlock new opportunities for innovation and knowledge-sharing, driving continuous improvement in crop management practices. Ultimately, the fruit disease detection project exemplifies the transformative potential of technology in revolutionizing agriculture, paving the way for a more resilient and sustainable food production system to meet the challenges of tomorrow's world.

CHAPTER 11
FUTURE SCOPE

In the future, the fruit disease detection project can be expanded to incorporate cutting-edge technologies and methodologies, ensuring its relevance and effectiveness in addressing emerging challenges in agriculture. One avenue for advancement is the integration of advanced deep learning techniques, such as attention mechanisms and graph neural networks, to enhance the model's ability to capture intricate patterns and dependencies within the image data. Additionally, the project can explore the fusion of multi-modal data sources, including satellite imagery, drone-based aerial surveillance, and weather data, to provide a comprehensive understanding of crop health and disease dynamics across different spatial and temporal scales.

Furthermore, the project can evolve towards a holistic crop management platform by integrating features for soil health assessment, nutrient management, and pest monitoring. This comprehensive platform can leverage data-driven insights to optimize farming practices, improve resource allocation, and mitigate environmental risks, ultimately enhancing crop productivity and sustainability. Collaborative efforts with agricultural stakeholders, including farmers, agronomists, and policymakers, will be pivotal in shaping the project's direction and ensuring its alignment with the needs and priorities of the agricultural community. By embracing innovation, collaboration, and a user-centric approach, the fruit disease detection project can contribute significantly to the advancement of precision agriculture and the global effort to ensure food security and agricultural resilience in the face of evolving environmental and socio-economic challenges.

CHAPTER 12
REFERENCES

[1]Malathy, S., Karthiga, R. R., Swetha, K., & Preethi, G. “Disease detection in fruits using image processing”. In 2021 6th International Conference on Inventive Computation Technologies (ICICT) (pp. 747-752). IEEE,2021, January.

[2]Devi, P. K. “Image Segmentation K-Means Clustering Algorithm for Fruit Disease Detection Image Processing”.In 2020 4th International Conference on Electronics, Communication and Aerospace Technology(ICECA) (pp.861-865).IEEE,2020, November.

[3]Wang, H., Mou, Q., Yue, Y., & Zhao, H.” Research on detection technology of various Fruit disease spots based on mask R-CNN”. In 2020 IEEE International Conference on Mechatronics and Automation (ICMA) (pp. 1083-1087). IEEE, 2020, October.

[4]Behera, S. K., Jena, L., Rath, A. K., & Sethy, P. K. “Disease classification and grading of oranges using machine learning and fuzzy logic”. In 2018 International Conference on Communication and Signal Processing (ICCSP) (pp. 0678-0682). IEEE, 2018, April.

[5]Zhao, J., & Qu, J.“A detection method for tomato Fruit common physiological diseases based on YOLOv2”. In 2019 10th international conference on Information Technology in Medicine and Education (ITME) (pp. 559-563). IEEE, 2019, August.

[6]Abirami, S., & Thilagavathi, M. “Classification of Fruit diseases using feed forward back propagation neural network”. In 2019 International Conference on Communication and Signal Processing (ICCSP) (pp. 0765-0768). IEEE, 2019, April.

[7]Nikitha, M., Sri, S. R., & Maheswari, B. U. “ Fruit recognition and grade of disease detection using inception v3 model”. In 2019 3rd International Conference on Electronics, Communication and Aerospace Technology (ICECA) (pp. 1040-1043). IEEE,2019, June.

[8]Dharmasiri, S. B. D. H., & Jayalal, S.” Passion Fruit Disease Detection using Image Processing”. In 2019 International Research Conference on Smart Computing and Systems Engineering (SCSE) (pp. 126-133). IEEE, 2019, March.