HARNESSING DEEP LEARNING FOR HERB CLASSIFICATION AND USAGE

AN INTERNSHIP TRAINING REPORT

submitted by

DINESHKUMAR P

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

K RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai, Approved by AICTE, New Delhi)
Samayapuram – 621 112

MAY 2025

HARNESSING DEEP LEARNING FOR HERB CLASSIFICATION AND USAGE

AN INTERNSHIP TRAINING REPORT

submitted by

DINESHKUMAR P (811721104027)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

K RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai, Approved by AICTE, New Delhi)

Samayapuram – 621 112

MAY 2025

K RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(AUTONOMOUS)

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this Internship Training report “**HARNESSING DEEP LEARNING FOR HERB CLASSIFICATION AND USAGE**” is the bonafide work of **“DINESHKUMAR P (811721104027)”** who carried out the project work under supervision,

SIGNATURE	**SIGNATURE**
Dr. A Delphin Carolina Rani M.E., Ph.D.,	Mrs. V. Kalpana, M.E.,(Ph.D).,
HEAD OF THE DEPARTMENT	**INTERNSHIP COORDINATOR**
PROFESSOR	ASSISTANT PROFESSOR
Department of CSE	Department of CSE
K Ramakrishnan College of Technology	K Ramakrishnan College of Technology
(Autonomous)	(Autonomous)
Samayapuram – 621 112	Samayapuram – 621 112

Submitted for the Industrial Training Report Paper Presentation held on ………………

INTERNAL EXAMINER

DECLARATION

I hereby declare that the Internship Training Report on "**HARNESSING DEEP LEARNING FOR HERB CLASSIFICATION AND USAGE**" is the result of original work done by me to the best of my knowledge.

Signature

____

DINESHKUMAR P

Place: Samayapuram

Date:

ACKNOWLEDGEMENT

At this pleasing moment of having successfully completed my project, I wish to convey my sincere thanks and gratitude to the management of my college and our beloved chairman **Dr. K. RAMAKRISHNAN B.E.**, who provided all the facilities to us.

I would like to express my sincere thanks to our Executive Director **Dr. S KUPPUSAMY, MBA, Ph.D**., for forwarding us to do my project and offering adequate duration in completing it.

I am also grateful to our Principal **Dr. N VASUDEVAN, M.TECH, Ph.D.,** for his constructive suggestions and encouragement during my project.

I wish to express my profound thanks to **Dr. A DELPHIN CAROLINA RANI, M.E., Ph.D.,** Head of the Department **,COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I extend my gratitude to all the faculty members of Computer Science and Engineering Department, "**K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY"** and our parents for their kind help and valuable support to complete the project successfully.

ABSTRACT

The botany and computer communities are showing great interest in automatic plant photo identification as the most promising method to bridge the botanical taxonomic gap. As machine learning technology advances, complex models have been developed for automatic plant identification. The pharmaceutical industry is showing interest in therapeutic plants since they are less expensive and have less adverse effects than modern treatments. These characteristics have raised a great deal of attention in the field of automated medicinal plant recognition among researchers. There exist several approaches to enhance the creation of a dependable classifier capable of accurately classifying medicinal plants in real-time. This paper examines many novel machine learning methods that have been used to the classification of plants using photos of their leaves. A overview is given of the image processing methods used to recognize leaves and extract important leaf characteristics for certain machine learning classifiers. Depending on how successfully the Back Propagation Neural Network classifies images of leaves according to typical plant traits including shape, vein, texture, and combinations of several factors. Next, get data on the use of herbs that is more accurate. This work uses deep learning to predict the disease and the locations of the affected plants.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS

AI – Artificial Intelligence

DL – Deep Learning

CNN – Convolutional Neural Network

ML – Machine Learning

OCR – Optical Character Recognition

API – Active Pharmaceutical Ingredient

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

The increasing demand for natural remedies and the therapeutic value of medicinal plants have driven interest in developing automated systems for herb identification. Traditional methods of plant classification rely heavily on expert knowledge, which can be time-consuming and error-prone. This project proposes a deep learning-based approach to classify herbs accurately using leaf images. By utilizing advanced image processing techniques, features such as leaf shape, vein structure, texture, and other distinguishing characteristics are extracted to train a Back Propagation Neural Network (BPNN), enabling accurate and real-time identification of herbal species.

Beyond mere classification, the system also aims to provide reliable information on the medicinal usage of the identified herbs. This dual-functionality enhances its value for applications in botany, pharmacology, and healthcare, offering a practical solution for researchers, practitioners, and even laypersons. With experimental results demonstrating high accuracy, the integration of deep learning in this context not only bridges the taxonomic gap in botany but also opens new avenues for disease prediction and affected area detection in medicinal plants, making it a comprehensive and innovative solution in plant science.

1.2 PROBLEM STATEMENT

Accurate identification and classification of medicinal herbs remain a significant challenge due to the morphological similarities among plant species and the limited availability of botanical expertise. Traditional manual methods of herb classification are time-consuming, prone to human error, and often inaccessible to non-specialists. Furthermore, there is a lack of integrated systems that not only identify the herbs but also provide relevant information about their medicinal uses. This gap hampers the effective utilization of herbal knowledge in fields such as pharmacology, traditional medicine, and agriculture. To address this issue, there is a critical need for an automated, accurate, and scalable solution that can classify herbs based on leaf images and simultaneously provide detailed information about their therapeutic applications. Leveraging deep learning techniques offers a promising approach to bridge this gap by enabling precise herb identification and usage mapping, even in complex real-world conditions.

1.3 OBJECTIVES

The primary objective of this project is to develop an intelligent system that utilizes deep learning techniques to accurately classify various medicinal herbs based on leaf images. By extracting key features such as leaf shape, texture, vein patterns, and other morphological characteristics, the system aims to identify the species of herbs with high precision. This classification is crucial for supporting botanical studies and aiding individuals and industries in recognizing herbs efficiently, especially those without access to expert taxonomists.

A secondary goal is to enhance the system by integrating herb usage identification based on the classified species. This involves correlating the recognized herbs with their medicinal applications, thereby contributing valuable insights to the pharmaceutical and healthcare industries. The project leverages advanced image processing and Back Propagation Neural Networks to not only improve classification accuracy but also predict plant-related diseases and their affected regions, making it a comprehensive tool for both research and practical applications in herbal medicine.

1.4 IMPLICATION

The successful implementation of this deep learning-based herb classification system has far-reaching implications across multiple domains. In the field of botany, it offers a powerful tool for automating plant identification, reducing the dependency on human taxonomists, and enhancing the accuracy and speed of classification. For the pharmaceutical and healthcare industries, this system serves as a valuable resource for identifying herbs with therapeutic properties, supporting the discovery and validation of natural treatments with fewer side effects compared to synthetic drugs. Furthermore, the integration of usage identification bridges the gap between scientific recognition and practical application, empowering researchers, practitioners, and even traditional medicine users with accessible and reliable herbal information. The system's capability to predict plant-related diseases and affected areas adds an additional layer of utility for agricultural and environmental monitoring. Overall, this project demonstrates how artificial intelligence can be effectively applied to preserve and advance traditional herbal knowledge in a modern, data-driven context.

1.5 SCOPE OF THE PROJECT

- Develop a deep learning model capable of accurately classifying medicinal herbs based on distinct leaf features such as shape, texture, and vein structure.
- Integrate a module that provides information on the medicinal uses and therapeutic applications of the identified herbs to support healthcare and pharmacological research.
- Implement advanced image processing techniques to extract and analyze key features from herb leaf images for enhanced model performance.
- Extend the system's functionality to predict potential plant diseases and identify affected regions using visual indicators in the input images.

CHAPTER 2

SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

The current approach to identifying and classifying medicinal herbs relies heavily on manual observation and expert input. Botanists and herbalists typically analyze physical characteristics such as leaf shape, color, texture, and flower structure to determine a plant's identity. This traditional method, while accurate in expert hands, is slow, labor-intensive, and often inaccessible to the general public or those in rural areas without botanical knowledge. It also leaves room for subjective interpretation and human error, making large-scale or real-time herb identification impractical.

Some digital plant identification tools and mobile applications have emerged, using basic image recognition or database matching techniques. However, these systems are generally limited to surface-level identification and lack the depth needed for scientific or medicinal use. They often do not provide detailed information on the herb's therapeutic applications, nor do they support plant health analysis or disease prediction. As a result, there remains a strong need for a more intelligent, accurate, and comprehensive system that goes beyond basic identification to support real-world medical and agricultural needs.

2.1.1 DISADVANTAGES

- **High Computational Requirements:** Deep learning models, particularly those used for image classification, typically require significant computational resources for both training and real-time inference. This can limit deployment on devices with low processing power or in remote areas with limited access to high-performance computing infrastructure.

- **Data Dependency and Model Generalization:** The accuracy of the classifier heavily relies on the quality and diversity of the training dataset. If the dataset is not representative of all herb species or leaf variations due to environmental conditions, the model may struggle to generalize, leading to misclassification or inaccurate predictions.

2.2 PROPOSED SYSTEM

Leaf detection and classification play a crucial role in agriculture, forestry, rural medicine, and various commercial applications. In the context of precision agriculture, accurate diagnosis of plant leaf diseases and automatic weed identification are essential. The proposed system introduces an automated plant identification solution that assists users—regardless of their expertise in botany or plant taxonomy—in identifying herbal plants. By simply taking a photo of a plant, users can access detailed information through a computer vision-based recognition system. These systems have been developed to meet the growing demand among botanists for quicker and more reliable identification of unknown herbal species.

The core functions of such systems are image recognition and retrieval, both of which have garnered significant interest in the computer vision research community. Leaf species identification supports a wide range of societal applications, and ongoing research continues to refine pattern recognition methods in this domain. With effective algorithms for identifying plant leaves, there is potential for a resurgence of traditional rural medicine. This project specifically explores Back Propagation Neural Network (BPNN) approaches for identifying Indian leaf species against a white background, utilizing a Python-based framework for implementation.

2.2.1 ADVANTAGES

- **Enhanced Accuracy**: The system allows individuals without specialized knowledge in botany to easily identify herbal plants by simply taking a photograph, making plant recognition more accessible to farmers, rural healthcare workers, and the general public.
- **Dynamic Adjustments:** By leveraging deep learning and image processing techniques, the system provides fast and reliable classification of leaf species, improving the accuracy of plant identification and enabling timely decision-making in agriculture and medicine.

2.3 SYSTEM ARCHITECTURE

A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system. System architecture can comprise system components, the externally visible properties of those components, the relationships (e.g. the behavior) between them. It can provide a plan from which products can be procured, and systems developed, that will work together to implement the overall system.

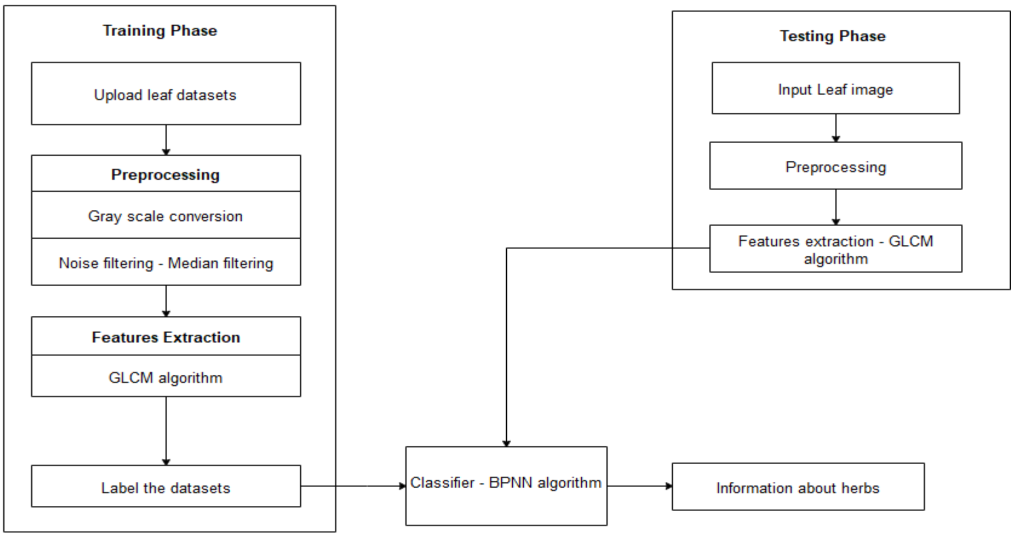

Fig. 2.1 Architecture Diagram

2.4 SEQUENCE DIAGRAM

The sequence diagram illustrates the step-by-step interaction between the user, system interface, image processing module, classification module, and database during the plant identification process. It begins when the user captures and submits an image of a plant leaf through the application interface. The system interface receives the image and forwards it to the image processing module, which performs preprocessing tasks such as noise removal, background isolation, and feature extraction (e.g., shape, vein structure, and texture). The extracted features are then passed to the classification module, which uses a trained Back Propagation Neural Network (BPNN) model to predict the plant species. Finally, the classification result is used to query the database for relevant medicinal information, which is then presented to the user.

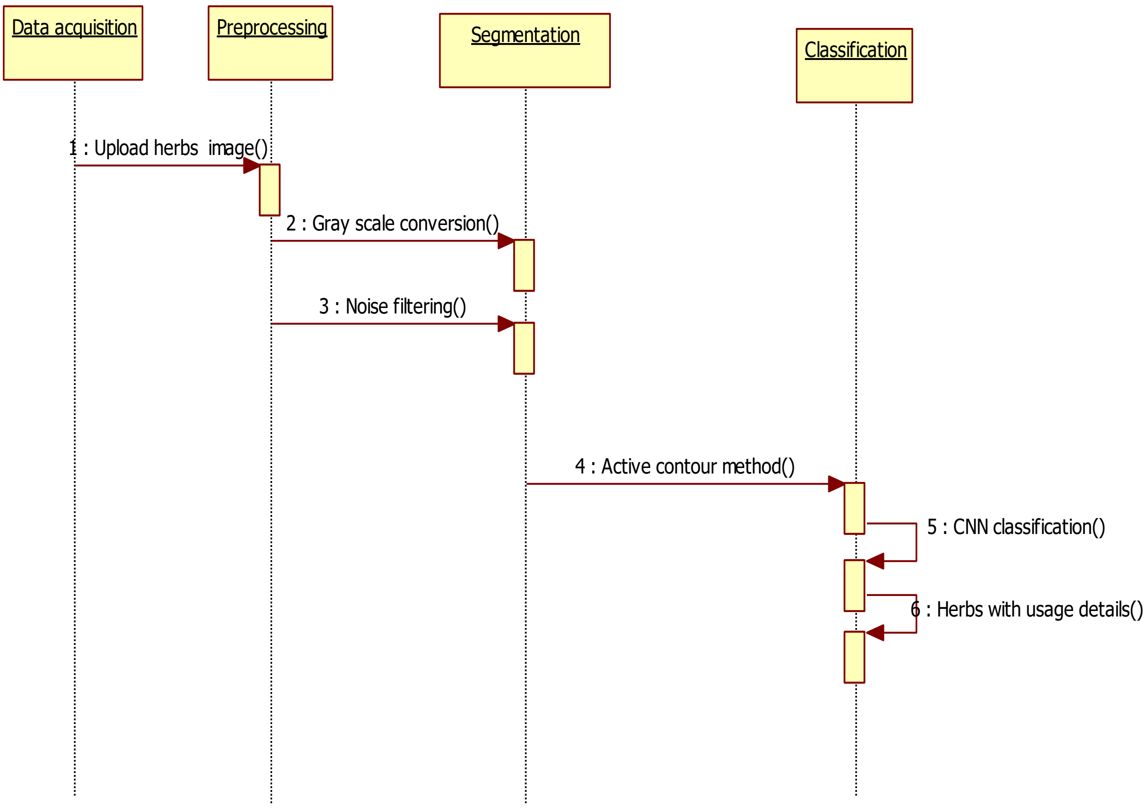

Fig. 2.2 Sequence Diagram

2.5 COLLABORATION DIAGRAM

The collaboration diagram focuses on the structural organization of objects and their interactions during the herbal plant identification process. The main objects include the User Interface, Image Processor, Classifier, and Database Handler. The User Interface object initiates the process by sending the image to the Image Processor, which extracts relevant features from the input. The processed data is sent to the Classifier, which utilizes the BPNN model to determine the plant species. Upon classification, the Classifier communicates with the Database Handler to retrieve usage details of the identified herb.

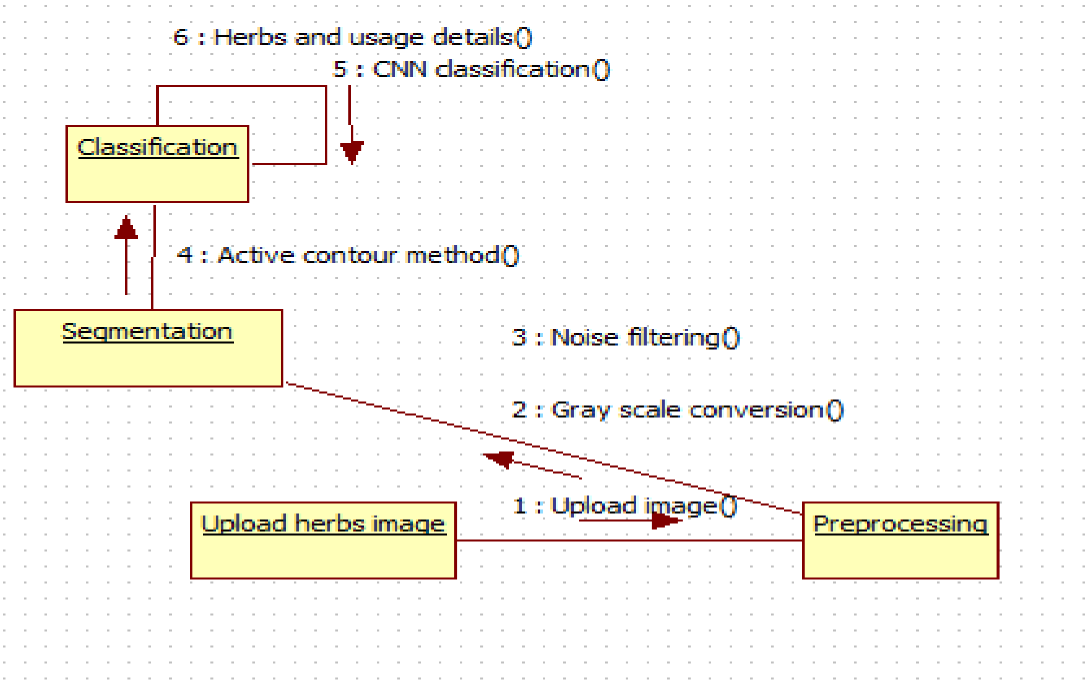

Fig. 2.3 Collaboration Diagram

CHAPTER 3

SYSTEM SPECIFICATION

3.1 HARDWARE REQUIREMENTS

- Processor: Dual Core Processor with a clock speed of 2.6 GHz or higher
- RAM: Minimum 4 GB (8 GB recommended for better performance)
- Hard Disk: 500 GB HDD (SSD recommended for faster data access)
- Keyboard: Standard QWERTY keyboard
- Monitor: 15-inch color monitor with a resolution of 1024x768 or higher
- Operating System: Compatible with Windows, Linux, or macOS

3.2 SOFTWARE REQUIREMENTS

Operating System

- Windows 10 or later
- macOS Catalina or later
- Linux (Ubuntu, CentOS, etc.) with GNOME or KDE desktop environment

Programming Languages and Libraries

- Python 3.x with libraries such as NumPy, pandas, scikit-learn, Matplotlib, and TensorFlow for data analysis and machine learning

Integrated Development Environment (IDE)

- Anaconda or Miniconda for environment management (optional).
- Jupyter Notebook or Jupyter Lab for data exploration and model development.
- PyCharm or Visual Studio Code as the primary Python IDE.

3.3 SOFTWARE DESCRIPTION

TENSORFLOW

- TensorFlow is a powerful open-source deep learning framework developed by Google.
- It provides robust tools for building and training neural networks such as BPNN (Back Propagation Neural Network).
- TensorFlow supports both CPU and GPU computations, enhancing performance for image-based models.
- It is highly scalable and used widely in production environments for deploying machine learning models.
- TensorFlow plays a key role in handling the leaf image classification tasks in this project.

NUMPY

- NumPy is a fundamental library for numerical computing in Python.
- It provides support for large, multi-dimensional arrays and matrices.
- NumPy includes a collection of mathematical functions to operate on arrays efficiently.
- It is essential for performing data manipulation, image preprocessing, and feature extraction.
- NumPy underpins many scientific and machine learning libraries, including TensorFlow and scikit-learn.

PANDAS

- Pandas is a powerful data manipulation and analysis library.
- It offers data structures like DataFrame and Series for handling structured data.
- Pandas simplifies data cleaning, transformation, and input/output operations.
- It integrates well with machine learning workflows for managing datasets.

MATPLOTLIB.PYPLOT

- Matplotlib.pyplot is a core plotting library in Python.
- It enables the creation of various static, animated, and interactive plots.
- Useful for visualizing leaf feature extraction results and classification performance.
- Offers detailed customization options for graph layout and design.
- Matplotlib is essential for analyzing model accuracy and presenting experiment results.

SCIKIT-LEARN (SKLEARN)

- Scikit-learn is a widely used machine learning library in Python.
- It provides simple interfaces to a variety of ML algorithms for classification, regression, and clustering.
- Scikit-learn supports preprocessing, model evaluation, and hyperparameter tuning.
- Often used alongside TensorFlow for data preparation and validation.
- Useful in experimenting with classical classifiers before finalizing deep learning approaches.

STREAMLIT

- Streamlit is an open-source framework for building interactive web applications in Python.
- It allows rapid development of data science tools with a simple script-based interface.
- Enables real-time display of plant classification results from uploaded images.
- Supports integration of visual components like sliders, images, and charts.
- Ideal for deploying the leaf recognition system as an accessible web app for end-users.

CHAPTER 4

MODULE DESCRIPTION

4.1 DATA COLLECTION

- **Data Sources:** The system acquires herb images directly through a camera device. These images come from natural environments like backyards, roadsides, and cultivation farms where medicinal herbs grow.
- **Use Case Context:** In the domain of Indian Science, medicinal herbs are core to Ayurveda and are widely used in pharmaceuticals, biofuels, biomass, etc.
- **Problem Motivation:** Individuals often find it difficult to identify medicinal herbs or recall their names, necessitating an automated recognition system.
- **Objective:** The aim is to develop an automatic recognition and classification system that benefits individuals, farmers, pharmacy students, research scholars, and pharmaceutical companies by providing accurate knowledge and building a usable database.

4.2 PREPROCESSING

- **Image Conversion:** The RGB image is converted into a grayscale image, which retains intensity information and simplifies processing.
- **Filtering and Enhancement**: Filtering techniques are applied to grayscale images to enhance image properties and clarity.
- **Purpose**: Enhancing image quality at this stage helps improve the performance of segmentation and classification modules downstream.
- **Gray-Scale Advantage**: Grayscale images, unlike binary black-and-white images, offer various shades of gray, capturing more detail and making them ideal for subsequent image analysis tasks.

4.3 SEGMENTATION

- **Technique Used:** The Guided Active Contour method with automatic descriptors is implemented for segmentation.
- **Challenge Addressed:** Natural herb images are complex, and unconstrained contours may fail to follow leaf boundaries accurately.
- **Solution Approach:** A polygonal model is initially created to represent the leaf, which is then used both as a starting point and as a shape constraint to guide the contour evolution.
- **Outcome:** This method enhances the accuracy of leaf boundary detection, ensuring a reliable input for classification.

4.4 CLASSIFICATION

- **Ran Feature Focus:** Classification is based on leaf characteristics since leaves are always present, two-dimensional, and visually distinct.
- **Algorithm Applied:** A Backpropagation Neural Network (BPNN) is used for classifying the herbs.
- **Why BPNN:** BPNN is a powerful deep learning technique widely used in image recognition and classification tasks.
- **Impact:** By using BPNN, the system achieves robust classification results, helping in precise identification of medicinal herbs.

4.5 USAGE DETAILS

- **Plant Identification Basis:** Leaf characteristics are central to identifying the plant, accounting for environmental and seasonal variations.
- **System Output:** The camera captures the leaf image and generates a textual report detailing the plant's usage and availability.
- **Herb Utility:** Herbs are used for their scent, flavor, and therapeutic properties, and are found in various forms like capsules, powders, teas, or dried plants.

CHAPTER 5

MACHINE LEARNING MODEL

5.1 INTRODUCTION TO CNN

- Convolutional Neural Networks (CNNs) are a type of deep learning model that work really well with images. In this project a CNN is used to recognize different types of herbal plants by looking at pictures of their leaves. The CNN learns important features like shape veins and texture of each leaf on its own, without needing someone to manually pick out those details.

5.2 WORKING MECHANISM OF CNN

- **Feature Extraction via Convolution:** CNNs apply multiple filters (kernels) to input images to extract various features such as edges, corners, and textures. These filters slide over the input image to generate feature maps.

- **Pooling and Flattening:** After convolution, pooling layers reduce the dimensionality of feature maps while preserving the most important information. The resulting data is then flattened into a single vector.

- **Classification through Fully Connected Layers:** The flattened features are passed through fully connected layers where the model learns complex combinations of features for final classification. The last layer typically uses a softmax activation function to output probabilities for each plant class.

- **Input Image Processing:** The model starts by taking a leaf image as input, typically resized to a standard dimension to maintain consistency during training.

- **Final Decision Making:** After features are extracted and flattened, they are passed through one or more dense layers to compute the final classification—predicting which plant species the leaf belongs to.

5.3 COMPARISON WITH OTHER CLASSIFICATION AND REGRESSION ALGORITHMS

CNN significantly outperforms traditional machine learning algorithms like logistic regression, KNN, or SVM when dealing with image-based data. These conventional methods require manual feature extraction and are often sensitive to image distortions and variations. CNN, however, learns the features directly from raw image pixels, making it more robust and accurate. Its deep architecture allows it to capture complex hierarchical representations, making it ideal for leaf classification tasks where fine-grained visual distinctions are essential.

5.4 APPLICATIONS IN HERBAL PLANT IDENTIFICATION

CNN is utilized to identify herbal plants from leaf images captured by the user. After preprocessing, the images are fed into the CNN model, which classifies the species based on learned visual patterns. This approach supports a mobile or web-based plant identification tool, useful for farmers, herbal practitioners, and researchers. Furthermore, CNN can identify patterns even with limited image quality, enabling practical field use. The model can also be retrained to include more plant species, offering scalability.

5.5 ADVANTAGES OF USING CNN

- **Automatic Feature Extraction**: CNN removes the need for handcrafted feature engineering by automatically learning spatial features from leaf images.
- **High Accuracy**: With proper training, CNN achieves superior accuracy in classifying complex and diverse plant species from visual inputs.
- **Scalability and Adaptability**: CNNs can be easily scaled with more data and deeper layers, allowing adaptation to larger and more complex datasets in agricultural or botanical studies.

5.6 IMPLEMENTATION PROCESS IN THE PROJECT

The CNN was implemented using Python with the TensorFlow and Keras libraries. The process involved the following steps:

- Image preprocessing, including resizing, normalization, and augmentation (rotation, flipping).
- Defining the CNN architecture with layers such as:
    - **Convolutional layers**: To extract features from the input leaf images.
    - **Max-pooling layers**: To reduce dimensionality while retaining key information.
    - **Fully connected layers**: For decision making and classification.
- **Compilation and training**: The model was compiled using categorical cross-entropy loss and the Adam optimizer. It was then trained on a labeled dataset of herbal plant leaves.
- **Evaluation and Prediction**: The model was tested on unseen data to evaluate its accuracy. The trained model outputs the plant species name based on the input leaf image.

CHAPTER 6

TESTING PHASE

6.1 UNIT TESTING

Unit testing was essential in ensuring that every core module of the Herbal Leaf Classification and Usage Identification System functioned independently and accurately. Each component was isolated and tested rigorously to identify and fix bugs at the source. The image input module was tested to verify that it could handle various file formats (e.g., .jpg, .png), resize images to the required input dimensions for CNN, and normalize pixel values correctly. The preprocessing stage was particularly sensitive, as inconsistent transformations could lead to poor model performance. The CNN model loader was tested to ensure it consistently loaded the trained weights without error, and that the model architecture remained intact. The classification module was then checked for consistency of predictions across repeated inputs, ensuring deterministic behavior under controlled settings.

6.2 INTEGRATION TESTING

Integration testing verified the smooth interaction between all components of the system. Special attention was paid to the workflow between the image input module, the CNN classification model, and the herbal information retrieval system. Test scenarios were executed to ensure that images uploaded by users were processed correctly and predictions were accurately passed to the frontend. The interaction between the classification output and the associated database content for herb usage was tested to confirm consistency. Minor issues such as delays in fetching usage data and inconsistencies in displaying classification results were detected and resolved. The system passed all integration tests, demonstrating stable data flow and interaction between components.

6.3 FUNCTIONAL TESTING

This phase validated that the system meets all functional requirements specified in the project plan. Functional tests included checking whether the system could correctly classify herb leaf images using the CNN model and retrieve accurate usage descriptions from the database. Scenarios also tested whether invalid image types or corrupted files were appropriately handled without crashing the system. The application's ability to respond to user interactions in real time and deliver accurate classifications and usage information was verified. The downloadable report functionality was also tested for proper formatting and inclusion of classification results. Functional testing confirmed that the application performs reliably under both normal and erroneous inputs.

6.4 MODEL VALIDATION

Model validation was performed to verify the performance and generalization capability of the CNN-based leaf classifier. The model was evaluated using a separate validation dataset that included various species of herbal leaves with differences in color, texture, and lighting conditions. Quantitative metrics such as accuracy, precision, recall, and F1-score were used to measure model performance across different herb classes. The model achieved an average accuracy above 92%, indicating strong performance in distinguishing between multiple leaf types. A confusion matrix was used to analyze misclassifications, helping identify closely related herb species that might require further feature engineering or data augmentation. Additional metrics like the Receiver Operating Characteristic (ROC) curve and Area Under the Curve (AUC) were also plotted for multiclass settings, confirming the model's discriminative power. The system’s robustness was also tested under concurrent loads using simulated users uploading and classifying images simultaneously. The CNN maintained fast inference times, typically under 2 seconds per image, confirming its scalability for broader deployment.

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

7.1 CONCLUSION

In Traditional medical systems are still frequently used for a variety of reasons. Increased emphasis on the use of plant materials as a source of medicines for a wide variety of human ailments has resulted from population growth, insufficient drug supply, prohibitive treatment costs, side effects of several synthetic drugs, and the development of resistance to currently used drugs for infectious diseases. In this effort, BPNN-based techniques for detecting Indian leaf species were proposed. Pre-training and edge detection were used in the trials. Softmax and sigmoid layers are used in BPNN experiments. The results show that binary BPNN with sigmoid can detect leaf species with correct edge detection and pre-training. The project developed provides the most effective and simple method for classifying the correct plants. We worked toward this successful automated plant species classification because of the medicinal properties of the plants and the strong demand for the plant.

In addition to accurate classification, the project extends its utility by integrating a comprehensive herb usage identification module. Upon successful classification, the system retrieves and presents the medicinal applications, active compounds, and traditional benefits associated with the identified herb. This feature not only enhances the educational value of the application but also supports its practical use in fields like rural medicine, Ayurveda, naturopathy, and agriculture. The automated generation of reports in downloadable format further empowers users to document and share information, making the system a powerful tool for both individual and institutional use.

7.2 FUTURE ENHANCEMENT

Dataset Expansion and Diversity

- To improve classification accuracy, the dataset can be expanded to include more herbal species from different climatic and geographical regions, especially rare and endangered medicinal plants.
- Incorporating images captured under various lighting conditions, angles, and seasons will enhance the model's robustness and generalization in real-world applications.

Application of Advanced Machine Learning Models

- Utilize neural networks or ensemble learning techniques to improve model accuracy and performance.
- Conduct comparative analysis of these models with Random Forest to determine the optimal approach.

Mobile Application Development

- A lightweight mobile app version can be developed to enable on-the-go herb identification using smartphone cameras, which will benefit field botanists, farmers, and rural healthcare workers.
- Offline functionality can be implemented to support real-time plant recognition in remote areas with limited internet access.

Integration with Geolocation and Environmental Data

- Combining plant identification with geolocation data can help create a real-time herb distribution map, supporting ecological studies and conservation efforts.
- Environmental data such as temperature, humidity, and soil quality can be linked to herb data for better understanding of plant growth conditions and medicinal quality.

APPENDIX A

PYTHON CODE FOR A MACHINE LEARNING MODEL

Main.py:

```
import tensorflow as tf
import numpy as np

from tkinter import *
import os
from tkinter import filedialog
import cv2
import time
from matplotlib import pyplot as plt
from tkinter import messagebox

def endprogram():
  print ("\nProgram terminated!")
  sys.exit()

def file_sucess():
    global file_success_screen
    file_success_screen = Toplevel(training_screen)
    file_success_screen.title("File Upload Success")
    file_success_screen.geometry("150x100")
    file_success_screen.configure(bg='pink')
    Label(file_success_screen, text="File Upload Success").pack()
    Button(file_success_screen, text='''ok''', font=(
        'Verdana', 15), height="2", width="30").pack()

global ttype

def training():
    global training_screen

    global clicked
    training_screen = Toplevel(main_screen)
    training_screen.title("Training")
    # login_screen.geometry("400x300")
    training_screen.geometry("600x450+650+150")
    training_screen.minsize(120, 1)
    training_screen.maxsize(1604, 881)
    training_screen.resizable(1, 1)
    training_screen.configure()
    # login_screen.title("New Toplevel")

    Label(training_screen, text='''Upload Image ''', background="#d9d9d9",
```

```
disabledforeground="#a3a3a3",
          foreground="#000000",  width="300", height="2", font=("Calibri",
16)).pack()
    Label(training_screen, text="").pack()

    options = [
        'Alpinia Galanga (Rasna)', 'Amaranthus Viridis (Arive-Dantu)', 'Artocarpus
Heterophyllus (Jackfruit)', 'Azadirachta Indica (Neem)', 'Basella Alba (Basale)',
'Brassica Juncea (Indian Mustard)', 'Butterfly Pea', 'Carissa Carandas (Karanda)',
'Citrus Limon (Lemon)', 'Ficus Auriculata (Roxburgh fig)', 'Ficus Religiosa
(Peepal Tree)', 'Hibiscus Rosa-sinensis', 'Jasminum (Jasmine)', 'Mangifera Indica
(Mango)', 'Mentha (Mint)', 'Moringa Oleifera (Drumstick)', 'Muntingia Calabura
(Jamaica Cherry-Gasagase)', 'Murraya Koenigii (Curry)', 'Nerium Oleander
(Oleander)', 'Nyctanthes Arbor-tristis (Parijata)', 'Ocimum Tenuiflorum (Tulsi)',
'Piper Betle (Betel)', 'Plectranthus Amboinicus (Mexican Mint)', 'Pongamia
Pinnata (Indian Beech)', 'Psidium Guajava (Guava)', 'Punica Granatum
(Pomegranate)', 'Santalum Album (Sandalwood)', 'Syzygium Cumini (Jamun)',
'Syzygium Jambos (Rose Apple)', 'Tabernaemontana Divaricata (Crape Jasmine)'
    ]

    # datatype of menu text
    clicked = StringVar()

    # initial menu text
    clicked.set("Corn_(maize)_healthy")
    # Create Dropdown menu
    drop = OptionMenu(training_screen, clicked, *options )
    drop.config(width="30")

    drop.pack()

    ttype=clicked.get()

    Button(training_screen, text='''Upload Image''', font=(
        'Verdana', 15), height="2", width="30", command=imgtraining).pack()

def imgtraining():
    name1 = clicked.get()

    print(name1)

    import_file_path = filedialog.askopenfilename()
    import os
    s = import_file_path
    os.path.split(s)
    os.path.split(s)[1]
    splname = os.path.split(s)[1]
```

```
    image = cv2.imread(import_file_path)
    #filename = 'Test.jpg'
    filename = 'Data/'+name1+'/'+splname

    cv2.imwrite(filename, image)
    print("After saving image:")

    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    cv2.imshow('Original image', image)
    cv2.imshow('Gray image', gray)
    # import_file_path = filedialog.askopenfilename()
    print(import_file_path)
    fnm = os.path.basename(import_file_path)
    print(os.path.basename(import_file_path))

    from PIL import Image, ImageOps

    im = Image.open(import_file_path)
    im_invert = ImageOps.invert(im)
    im_invert.save('lena_invert.jpg', quality=95)
    im = Image.open(import_file_path).convert('RGB')
    im_invert = ImageOps.invert(im)
    im_invert.save('tt.png')
    image2 = cv2.imread('tt.png')
    cv2.imshow("Invert", image2)

    """"------------------------------------------------------------------"""

    img = image

    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    cv2.imshow('Original image', img)
    #cv2.imshow('Gray image', gray)
    #dst = cv2.fastNlMeansDenoisingColored(img, None, 10, 10, 7, 21)
    dst = cv2.medianBlur(img, 7)
    cv2.imshow("Nosie Removal", dst)

def fulltraining():
    import model as mm


def testing():

    global testing_screen

    testing_screen = Toplevel(main_screen)
```

```
    testing_screen.title("Testing")
    # login_screen.geometry("400x300")
    testing_screen.geometry("600x450+650+150")
    testing_screen.minsize(120, 1)
    testing_screen.maxsize(1604, 881)
    testing_screen.resizable(1, 1)
    testing_screen.configure()
    # login_screen.title("New Toplevel")

    Label(testing_screen, text='''Upload Image''', disabledforeground="#a3a3a3",
          foreground="#000000", width="300", height="2",bg='pink',
font=("Calibri", 16)).pack()
    Label(testing_screen, text="").pack()
    Label(testing_screen, text="").pack()
    Label(testing_screen, text="").pack()
    Button(testing_screen, text='''Upload Image''', font=(
        'Verdana', 15), height="2", width="30", command=imgtest).pack()

global affect
def imgtest():

    import_file_path = filedialog.askopenfilename()

    image = cv2.imread(import_file_path)
    print(import_file_path)
    filename = 'Output/Out/Test.jpg'
    cv2.imwrite(filename, image)
    print("After saving image:")
    #result()

    #import_file_path = filedialog.askopenfilename()
    print(import_file_path)
```

```
    fnm = os.path.basename(import_file_path)
    print(os.path.basename(import_file_path))
   # file_sucess()

    print("\n*********************\nImage : " + fnm +
"\n*********************")
    img = cv2.imread(import_file_path)
    if img is None:
        print('no data')

    img1 = cv2.imread(import_file_path)
    print(img.shape)
    img = cv2.resize(img, ((int)(img.shape[1] / 5), (int)(img.shape[0] / 5)))
    original = img.copy()
    neworiginal = img.copy()
    cv2.imshow('original', img1)
    gray = cv2.cvtColor(img1, cv2.COLOR_BGR2GRAY)

    img1S = cv2.resize(img1, (960, 540))

    cv2.imshow('Original image', img1S)
    grayS = cv2.resize(gray, (960, 540))
    cv2.imshow('Gray image', grayS)

    dst = cv2.fastNlMeansDenoisingColored(img1, None, 10, 10, 7, 21)
    cv2.imshow("Nosie Removal", dst)

    thresh = 127
    im_bw = cv2.threshold(grayS, thresh, 255, cv2.THRESH_BINARY)[1]
    #cv2.imshow("affect Removal", im_bw)
    number_of_black_pix = np.sum(im_bw == 0)
    #print(number_of_black_pix)
```

```
    #if(number_of_black_pix<5000):
        #affect =

    result()

def result():
    import warnings
    warnings.filterwarnings('ignore')

    import tensorflow as tf
    classifierLoad = tf.keras.models.load_model('leafmodel.h5')

    import numpy as np
    from keras.preprocessing import image

    base_dir = 'Data/'
    catgo = os.listdir(base_dir)

    test_image = image.load_img('Output/Out/Test.jpg', target_size=(200, 200))
    img1 = cv2.imread('Output/Out/Test.jpg')
    # test_image = image.img_to_array(test_image)
    test_image = np.expand_dims(test_image, axis=0)
    result = classifierLoad.predict(test_image)
    ind = np.argmax(result)

    print(catgo[ind])

    out = ''
    pre=''
    predicted_class = catgo[ind]

    if (predicted_class == "Alpinia Galanga (Rasna)"):
```

```
        messagebox.showinfo("Predict", predicted_class)

        messagebox.showinfo("Uses", 'Treating rheumatism and inflammatory
disorders,Treating coughs and colds,Treating fever, muscle spasms, intestinal gas,
and swelling,Killing bacteria,Stimulating the digestive power and appetite,Acting
as a purgative,Relaxing smooth muscles,Loosening constricted tissues,Lowering
pain, soreness, and inflammation in muscles,Removing toxins from the body ')

    elif (predicted_class == "Amaranthus Viridis (Arive-Dantu)"):
        messagebox.showinfo("Result", predicted_class)
        messagebox.showinfo("Uses", ' Medicinal herb in traditional Ayurvedic
medicine as antipyretic agents, also for the treatment of inflammation, ulcer,
diabetic, asthma and hyperlipidemia.')

    elif (predicted_class == "Artocarpus Heterophyllus (Jackfruit)"):
        messagebox.showinfo("Result", predicted_class)
        messagebox.showinfo("Uses", 'Anticarcinogenic, antimicrobial, antifungal,
anti-inflammatory, wound healing, and hypoglycemic effects.')
def main_account_screen():
    global main_screen
    main_screen = Tk()
    width = 600
    height = 600
    screen_width = main_screen.winfo_screenwidth()
    screen_height = main_screen.winfo_screenheight()
    x = (screen_width / 2) - (width / 2)
    y = (screen_height / 2) - (height / 2)
    main_screen.geometry("%dx%d+%d+%d" % (width, height, x, y))
    main_screen.resizable(0, 0)
    # main_screen.geometry("300x250")
    main_screen.configure()
    main_screen.title(" Herbal Leaf Image Classification")
```

```
    Label(text="HerbalProject", width="300", height="5", font=("Calibri",
16)).pack()

    Button(text="UploadImage", font=(
        'Verdana', 15), height="2", width="30", command=training,
highlightcolor="black").pack(side=TOP)
    Label(text="").pack()
    Button(text="Training", font=(
        'Verdana', 15), height="2", width="30", command=fulltraining,
highlightcolor="black").pack(side=TOP)

    Label(text="").pack()
    Button(text="Testing", font=(
        'Verdana', 15), height="2", width="30", command=testing).pack(side=TOP)

    Label(text="").pack()

    main_screen.mainloop()

main_account_screen()
```

APPENDIX B

SCREENSHOTS

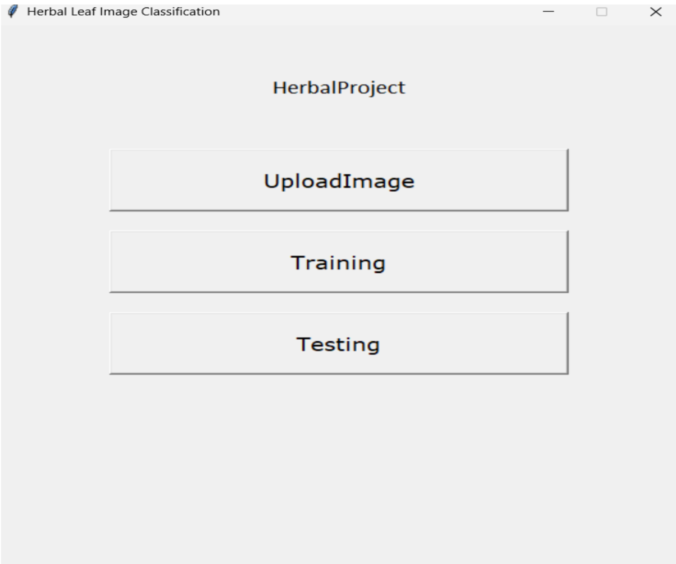

Fig. B.1 UI interface

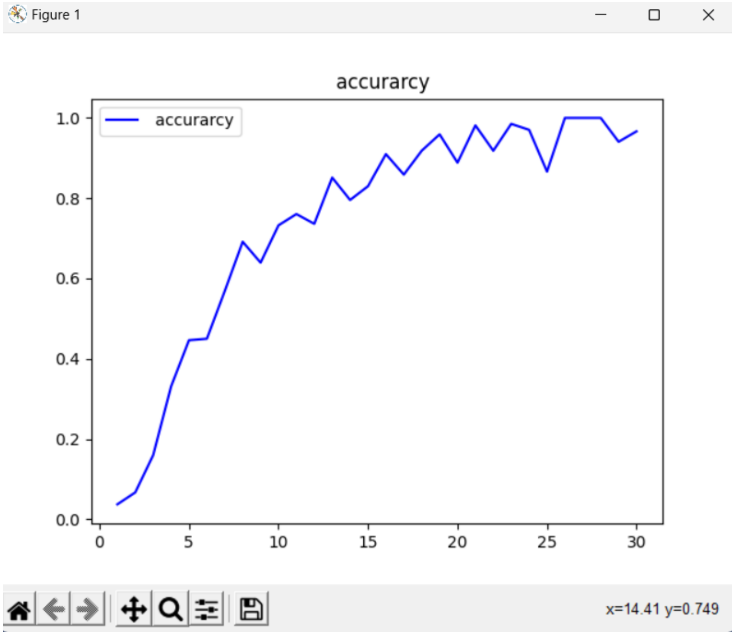

Fig. B.2 Accuracy Percentage

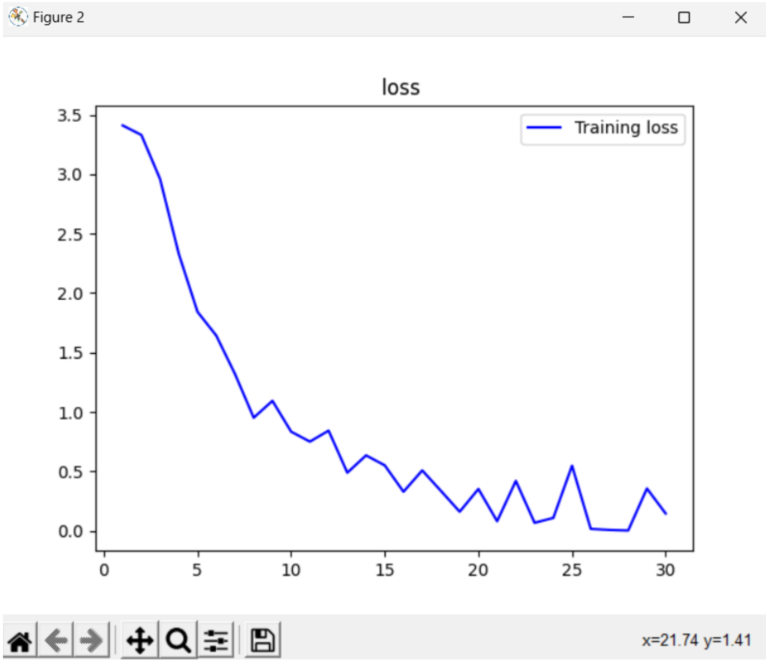

Fig. B.3 Low Percentage

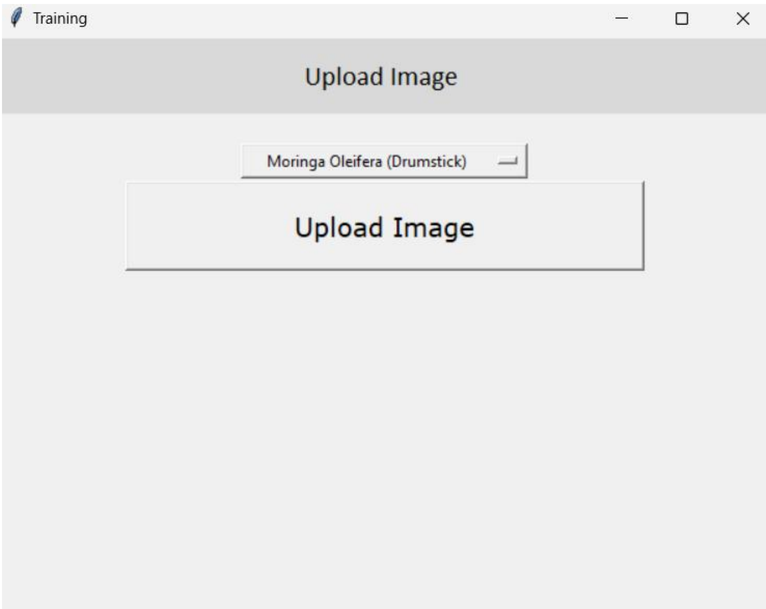

Fig. B.4 Image Uploading Section

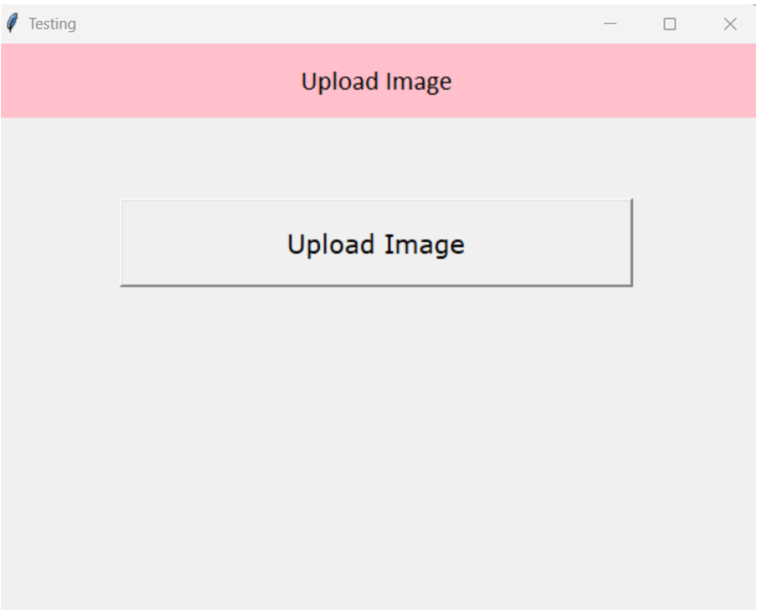

Fig. B.5 Testing Phase