MEDICINAL PLANT IDENTIFICATION

A PROJECT REPORT

Submitted by

PRESHIT SHARMA [Reg No:RA1711004010381]
RASHIKA SHARMA [Reg No:RA1711004010433]
NITIKA BAGGA [Reg No:RA1711004010549]

Under the guidance of

Mr. R. PRITHIVIRAJ

(Assistant Professor, Department of Electronics and communication Engineering)

in partial fulfillment for the award of the

degree of

BACHELOR OF TECHNOLOGY

in

ELECTRONICS AND COMMUNICATION ENGINEERING

Department of Electronics and Communication Engineering

College of Engineering and Technology

SRM Institute of Science and Technology

SRM Nagar, Kattankulathur - 603203, Kancheepuram District, Tamilnadu

MAY 2021

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Deemed to be University u/s 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that this project report titled "**MEDICINAL PLANT IDENTIFICATION**" is the bonafide work of "**PRESHIT SHARMA [RegNo:RA1711004010381], RASHIKA SHARMA [RegNo:RA1711004010433], NITIKA BAGGA [RegNo:RA1711004010549]**" who carried out the project work under my supervision as a batch. Certified further, that to the best of my knowledge the work reported herein does not form any other project report on the basis of which a degree or award was conferred on an earlier occasion for this or any other candidate.

Date:18.05.21 Project Supervisor Head of the Department

Submitted for University Examination held on 18th May 2021 in the Department of Electronics and Communication Engineering, SRM Institute of Science and Technology, Kattankulathur.

Date:18.05.21 Internal Examiner External Examiner

DECLARATION

We hereby declare that the Major Project entitled "**Medicinal Plant Identification**" to be submitted for the Degree of Bachelor of Technology is our original work as a team and the dissertation has not formed the basis of any degree, diploma, associate-ship or fellowship of similar other titles. It has not been submitted to any other University or institution for the award of any degree or diploma.

Place: SRM-IST KTR CHENNAI

Date:18.05.21

PRESHIT SHARMA
[RA1711004010381]

RASHIKA SHARMA
[RA1711004010433]

NITIKA BAGGA
[RA1711004010549]

ABSTRACT

Identification of plants is a difficult task considering the lack of awareness about the local flora. Thus an automated identification system is a highly beneficial and superior option. Information about local plants is essential for plants and herbs based medicine systems like Ayurveda. This project focuses on the utilization of image processing techniques to extract morphological, texture-based, and colour-based features from test images scanned using a camera and to compare these features using a support vector machine (Support Vector Machine (SVM)) for classification to detect leaf species with accuracy higher than 85 percentage.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my guide, Mr. R. PRITHIVIRAJ his valuable guidance, consistent encouragement, personal caring, timely help and providing me with an excellent atmosphere for doing research. All through the work, in spite of his busy schedule, he has extended cheerful and cordial support to me for completing this research work.

Author

LIST OF FIGURES

ABBREVIATIONS

SVM	Support Vector Machine
GLCM	Gray Level Co-Occurrence Matrix
IDM	Inverse Difference Moments
ANN	Artificial Neural Network
KNN	K Nearest Neighbor
URL	Uniform Resource Locator
GUI	Graphical User Interface

LIST OF SYMBOLS

$\sum$	Summation
μ	Mean
σ	Standard deviation
$I(x, y)$	Pixel intensity at (x,y) coordinates
$G(i, j)$	(i,j)th element of GLCM(Gray level co-occurrence matrix)
ln	Natural logarithm

CHAPTER 1

INTRODUCTION

Ayurveda is Indian traditional system of medicine. It is related to preservation of health by maintaining a balance between mind, body, and spirit. It focuses on prevention of disease rather than the treatment. It is used to make the immune system stronger. It is an ancient medical practice that is mostly based on plants using leaves, roots, fruits and seeds of the plant. These medicinal herbs are used to cure a multitude of ailments. The identification of these leaves using naked eyes is tough and at high risk. Methodology to identify the plant species using its leaf image comprises of preprocessing of the query leaf image followed by segregation and differentiation based on various factors. The uses and properties of these herbs include curing of ailments, boosting immunity, homeopathic medicines, etc.

1.0.1 Aim/Objective

Aim of this project is to recognize leaf using contour based edge detection for geometrical features and Gray level Co-occurance matrices for texture features combined with color moments of RGB image.Then to study features using classifier i.e. SVM(Support Vector Machine) which will determine performance of the model.This model will provide botanical name of the leaf and display its uses and properties. This model aims to reduce the man work and make an automated application that uses leaf images to detect its species and results in the uses and properties with the botanical name of that leaf. The uses will also include the contribution of that leaf in Ayurvedic medicine compositions. This model aims to increase the accuracy in identification while having less storage requirements. The uses and properties are not stored but displayed on Google using the browser control feature. Database stored on the cloud storage can be used which in turn shuns the use of hard disk or any external storage device to store huge database.

CHAPTER 2

LITERATURE SURVEY

Plant identification has been proposed using image processing earlier using Artificial Neural Network (ANN), K Nearest Neighbor (KNN), and SVM classifiers extracting features like shape, color, or veins. With a dataset of 1000-1500 images and returning closest matches.

1. Thibaut Beghin, James S Cope et al [3] used incremental classification using only shape and texture based features.While incremental classification increases process time and is computationally intensive,dependence on just contour score and sobel score is only efficient in broader classifications thereby reducing accuracy for wider range of species.The method was only tested on limited dataset.

2. Cem Kalyoncu, Önsen Toygar [2] are highly inclined towards geometrical features.Due to less inter species differences based on just geometrical shape ,model is less feasible for separating species with similar shape.

3. Sulc M., Matas J. [10] relies on histograms for texture on border and interior of leaf and leaf orientation is hurdle for accurate classification.High accuracy was achieved for standard datasets with clear pictures.Pictures scanned using mobile camera cannot provide clear texture information.

4. H.X.Kan ,L.Jin et al.[11] used 10 shape based features and 5 texture based features with support vector machine as classifier to classify 12 different species with accuracy greater than 90 percentage.Use of medicinal plants is TCM(Traditional Chinese Medicine) was considered.Lack of proper database for medicinal plants acts as big hurdle in testing reliability of model.

5. D. Venkataraman and N. Mangayarkarasi [12] used HOG(Histogram of Oriented Gradients) ,texture based features and color moments for classification of medicinal plants using support vector machine as classifier.

From the previous studies it is evident that morphological or geometrical features ,texture based features and color moments are used to identify the plant using sample leaf images.

CHAPTER 3

SYSTEM DESIGN

3.1 Proposed System

• The proposed system architecture consists of different steps involved. The whole process is divided into two stages: Training and testing. The training stage is about storing the dataset and training the classifier on the same dataset.Dataset is a collection of feature vectors for the respective leaf image.These feature vectors are collection all the extracted features from sample image.The trained classifier model is then ready for testing on query images.

• Three main features of leaf i.e. Shape, texture, and color-based features are extracted and combined in a single feature vector. Features of multiple sample images of different leaf species are thus extracted and stored in their respective feature vectors. .In the testing stage same feature extraction is incorporated on query image followed by classification among the various classes or species of leaves. When the feature vector of the query image is fed to the classifier it returns the botanical name of the species.

• The botanical name of the species identified opens the gateway to access information regarding the species from the internet. Webbrowser module in python allows searching of user-defined strings on the internet. E-encyclopedias of medicinal plants and webpages dedicated to Ayurvedic plants are preferred sources for information regarding these plants such as uses, properties, and various local names and better alternative to storing of this information on local system. List of the leaves included in custom dataset used for training the model with common and botanical names is shown in Fig.1 .

3.2 Algorithm

- Upload query image
- Extract features from leaf image
- Store Features in vector
- Feed vector to classifier
- Comparison with already stored feature vectors
- Output as botanical name of leaf species
- Uses and properties of species displayed

Local Name	Botanical Name
Neem	Azadirachta indica
Holy Basil	Ocimum tenuiflorum
Ashoka	Saraca asoca
Papri(Indian Elm)	Holoptelea integrifolia
Ayapan	Eupatorium ayyappana
Great Basil	Ocimum basilicum
Tej patta	Laurus nobilis
Elaichi	Elettaria cardamomum
Bhang	Cannabis
Giloi	Tinospora cordifolia
Bhurat(Sundakkai)	Solanum torvum
Paan	Piper betle
Indrajao(Dhudhi)	Wrightia Tinctoria Pala
Adrak(Ginger)	Zingiber officinale
Marorphali	Helicteres isora
Bamboo	Phyllostachys edulis
Horse Chestnut(Kanor)	Aesculus chinensis
Anhu Barberry	Berberis anhweiensis Ahrendt
Chinese redbud	Cercis chinensis
True indigo	Indigofera tinctoria L.
Nanmu	Phoebe nanmu (Oliv.) Gamble
Japanese maple	Acer Palmatum
Castor aralia	Kalopanax septemlobus (Thunb. ex A.Murr.) Koidz.
Cinnamon	Cinnamomum japonicum Sieb
Goldenrain tree	Koelreuteria paniculata Laxm.
Big fruited Holly	Ilex macrocarpa Oliv
Japanes cheeswood	Pittosporum tobira (Thunb.) Ait. f.
Wintersweet	Chimonanthus praecox L.
Camphor tree	Cinnamomum camphora (L.) J. Presl
Arrowwood	Viburnum awabuki K.Koch
Sweet osmanthus	Osmanthus fragrans Lour.
Deodar	Cedrus deodara (Roxb.) G. Don
Maidenhair	Ginkgo biloba L.
Crape myrtle	Lagerstroemia indica (L.) Pers.
Oleander	Nerium oleander L.
Yew plum pine	Podocarpus macrophyllus (Thunb.) Sweet
Flowering cherry	Prunus serrulata Lindl. var. lannesiana auct.
Glossy Privet	Ligustrum lucidum Ait. f.
Chinese toon	Tonna sinensis M. Roem.
Peach	Prunus persica (L.) Batsch.
Ford Woodlotus	Manglietia fordiana Oliv.
Trident Maple	Acer buergerianum Miq.
Barberry	Mahonia bealei (Fortune) Carr.
Southern magnolia	Magnolia grandiflora L.
Poplar	Populus ×canadensis Moench
Tulip tree	Liriodendron chinense (Hemsl.) Sarg.
Tangerine	Citrus reticulata Blanco

Figure 3.1: Leaf names list

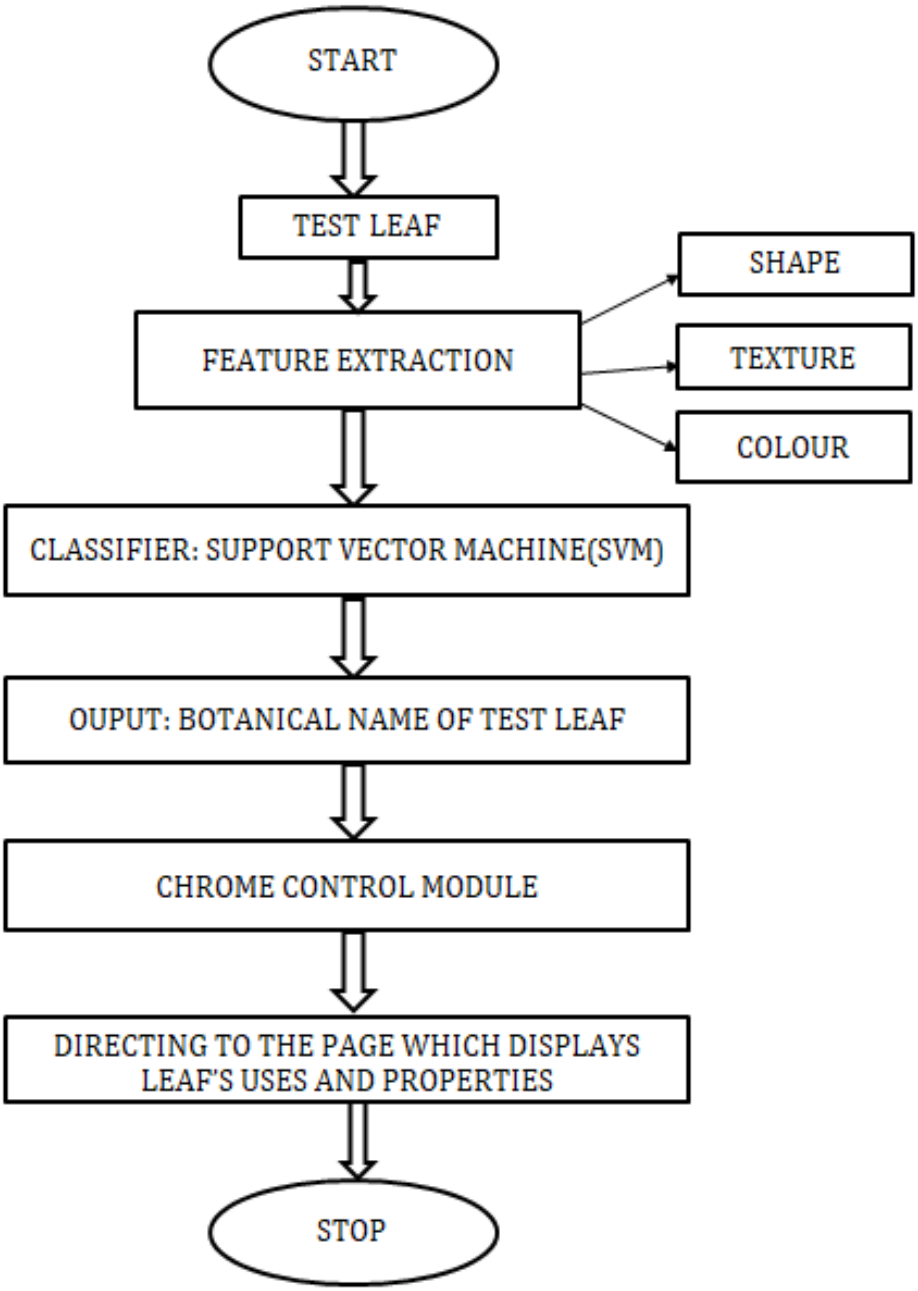

Figure 3.2: Flow chart

CHAPTER 4

METHODOLOGY

4.1 Working Principle

4.1.1 Feature Extraction

Shape

Figure 4.1: Query Image

- Morphological features – Contouring [6][7] aids to find different contours in leaf image enclosing all closed shapes in image from this largest contour is the one enclosing the leaf as depicted in Fig.3(a,b and c). Opencv package of python provides with different functions for several morphological features - Image moments: Image moments aids in calculation center of mass and area of the object in image.Centroid is given by the relations:

$$C_x = \frac{M_{10}}{M_{00}} \tag{4.1}$$

$$C_y = \frac{M_{01}}{M_{00}} \tag{4.2}$$

Area is given by:

$$A = M['m00'] \tag{4.3}$$

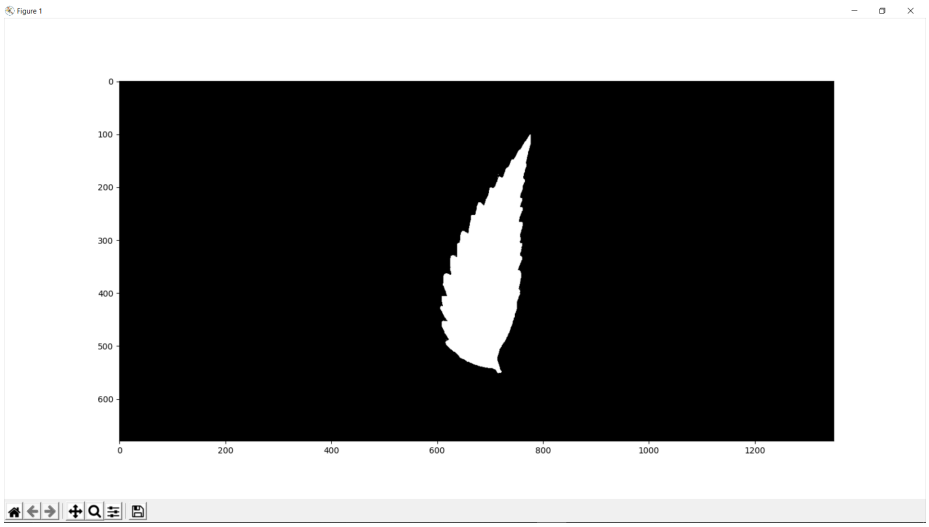

Figure 4.2: Mask to separate leaf from background

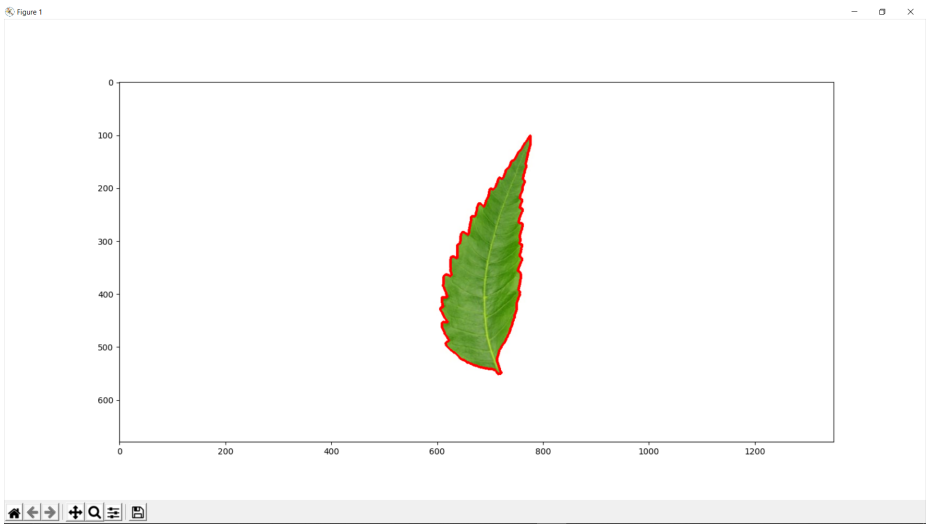

Figure 4.3: Largest contour enclosing the leaf

here Mij is raw moment of image with pixel intensity I(x,y) and can be calculated by

$$M_{ij} = \sum_{x} \sum_{y} x^i y^j I(x, y) \tag{4.4}$$

,area can also be calculated by cv.contourArea() function.

- Contour perimeter: It can be found out using cv.arcLength() function
- Circularity: it can be found using area and arc length.

$$Circularity = \frac{perimeter^2}{area} \tag{4.5}$$

- Rectangularity and aspect ratio: Width and height of bounding rectangle is used

to calculate rectangularity and aspect ratio

$$Aspectratio = \frac{width}{height} \tag{4.6}$$

$$Rectangularity = \frac{width + height}{area} \tag{4.7}$$

Texture

To extract features from leaf image which can provide info of different vein structure and patterns on leaf surface ,image is transformed to grayscale . Haralick features are extracted from Gray Level Co-Occurrence Matrix (GLCM) (Gray level Co-occurrence matrix) of grayscale image. This matrix records how many times two gray-level pixels adjacent to each other appear in an image [4] .

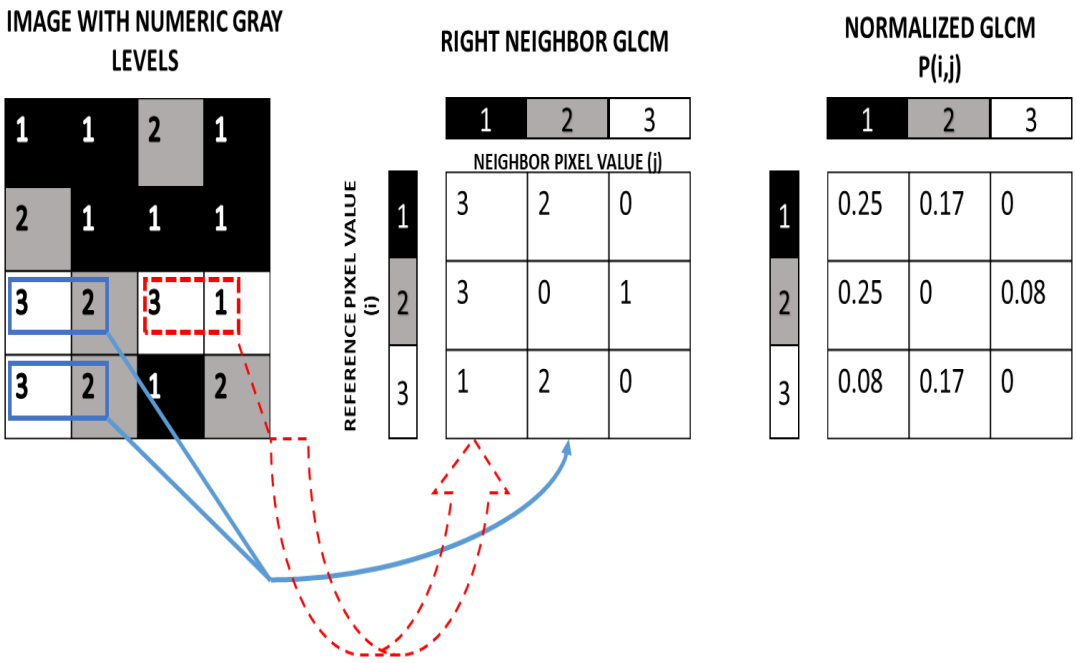

Figure 4.4: Gray Level Co-occurrence Matrix

This allows to identify particular vein pattern and texture on surface of different leaves and makes addition to features on the basis of which leaves are differentiated. For a normalized symmetric GLCM matrix (G) of dimensions N X N where N is number of gray levels and G(i,j) is (i,j)th element of matrix ,various texture features are given by:

- Contrast is a measure of intensity or gray level variations between the reference pixel and its neighbor:

$$Contrast = \sum_i \sum_j (i-j)^2 G(i,j) \tag{4.8}$$

- Correlation feature shows the linear dependency of gray level values in the cooccurrence matrix:

$$Correlation = \sum_i \sum_j G(i,j) \frac{(i-\mu_x)(j-\mu_y)}{\sigma_x \sigma_y} \tag{4.9}$$

where μ_x, μ_y, σ_x and σ_y are the means and standard deviations and are expressed as:

$$\mu_x = \sum_i \sum_j iG(i,j) \tag{4.10}$$

$$\mu_y = \sum_i \sum_j jG(i,j) \tag{4.11}$$

$$\sigma_x = \sqrt{\sum_i \sum_j (i-\mu_x)^2 G(i,j)} \tag{4.12}$$

$$\sigma_y = \sqrt{\sum_i \sum_j (j-\mu_y)^2 G(i,j)} \tag{4.13}$$

- Inverse difference moments (Inverse Difference Moments (IDM)) or Homogeneity is a measure of closeness between the distribution of elements in the GLCM and the diagonal of GLCM:

$$\text{IDM} = \sum_i \sum_j \frac{1}{1+(i-j)^2} G(i,j) \tag{4.14}$$

- Entropy is the degree of disorder present in the image. Entropy shows largest value when all elements of the co-occurrence matrix have similarity and lowest value when elements have inequalities :

$$Entropy = -\sum_i \sum_j G(i,j) lnG(i,j) \tag{4.15}$$

Colour

Colour based features aids to differentiate between plants with different coloured leaves .Image is divided into three channels i.e. (red green and blue) Mean and standard deviation of theses channel values are stored as colour features of leaf image [8].

4.1.2 Classification

For classification of query image among the samples in database Support Vector Machine or SVM classifier is used . SVM classifies datapoints using N dimensional hyperplane where N is number of features. SVM natively is a binary classifier i.e. each datapoint can either lie on one side of the hyperplane .Support vectors are datapoints nearer to hyperplane which defines the margin. The generalized equation of a hyperplane is given by:

$$w^T x = 0 \tag{4.16}$$

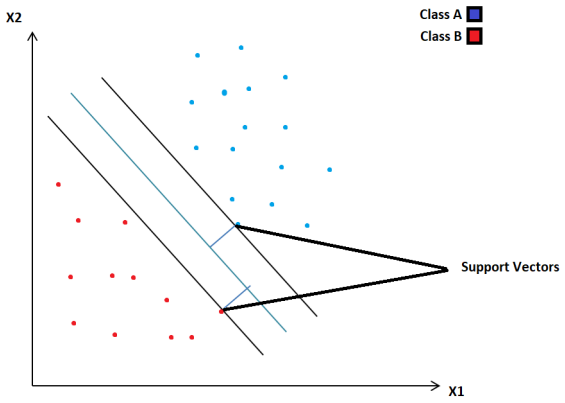

Figure 4.5: Binary SVM Model

For multiclass classification ,binary SVM is modified with the help of algorithms namely one vs all and one vs one which includes binary classifier per each class . In the One-to-Rest approach, hyperplane is used to separate between a class and all others at once while in One vs One approach it ignores all other classes while a hyperplane seperates two classes in current split as shown in Fig.5(a,b).

Normalization of feature vectors is done to match standard range for SVM from 0 to 1 or -1 to 1. After training the classifier on dataset , parameter tuning is used to enhance accuracy testing for different values of C (regularization parameter which deals with misclassification) and gamma and selecting the ideal kernel for the testcase.

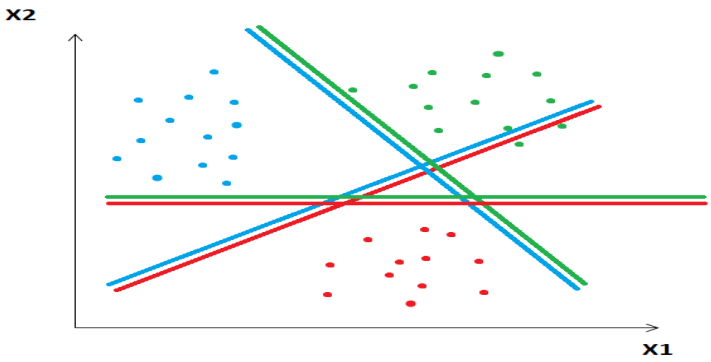

Figure 4.6: One vs One Approach

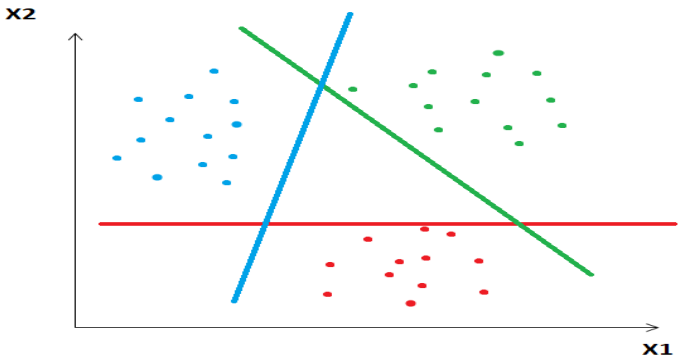

Figure 4.7: One vs All Approach

4.1.3 Websearch

The webbrowser module provides interface to allow users to access web pages and display them. Browser application is set as environment for the code and controller.open(Uniform Resource Locator (URL)) function can be used to open a new window in browser which displays the page corresponding to "URL" string. url = 'http://www.google.com/' comment-Open URL in a new window, if a browser window is already open. webbrowser.open(URL)

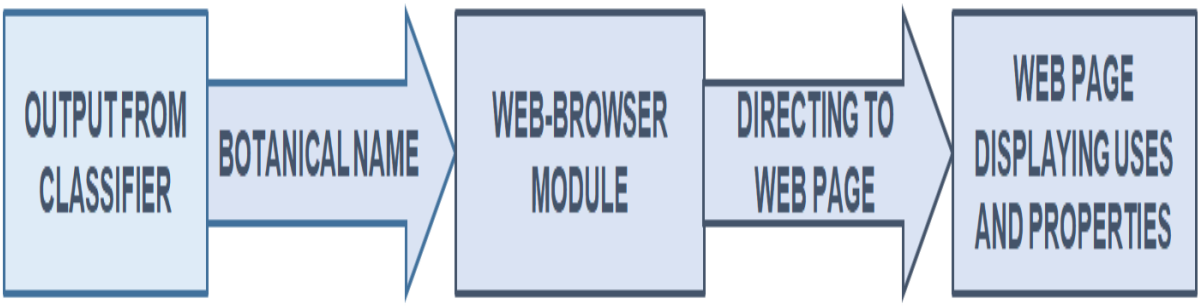

Figure 4.8: Browser Control

CHAPTER 5

CODING/TESTING

5.1 Source Code

Source code for "Identification of Indian Medicinal Plants using Image Processing Techniques" in Python:

5.1.1 Functions involved

Training and Testing phase

Two main functions of program are:

- File named store.py which generates feature vectors for all the leaf sample images stored in local directory and collects all these feature vectors in Excel sheet(.csv file).

```
import pandas as pd
from texture import *
from preprocess import *
from shape import *
from colour import *
import os

titles = [ 'area','perimeter','aspect_ratio','rectangularity','circularity',
        'contrast','correlation','inverse_difference_moments','entropy',
         'mean_r','mean_g','mean_b','std_r','std_g','std_b'
        ]



path  = 'D:\\MDD\\Leaf images - ALL Final'
file_itr = os.listdir(path)
df = pd.DataFrame([], columns=titles)
for file in file_itr:
    imgpath = path + "\\" + file
    image = cv2.imread(imgpath)
    cv2.imshow('leaf',image)
    cv2.waitKey(500)
```

```
    cv2.destroyAllWindows()
    # texture
    fv_texture = texture(image)
    # shape
    fv_shape = shape(image)
    # colour
    fv_colour = colour(image)

    fv_final = fv_shape + fv_texture + fv_colour

    print(fv_final)
    df_temp = pd.DataFrame([fv_final], columns= titles)
    df = df.append(df_temp)

df.to_csv("Leaf_data.csv")
```

- SName() in testing.py which returns botanical name of query image

```

import cv2
import numpy as np
import pandas as pd
import os
import string
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn import metrics
from sklearn.preprocessing import StandardScaler
from preprocess import preprocess
from shape import shape
from colour import colour
from texture import texture
from sklearn.model_selection import GridSearchCV
from sklearn.decomposition import PCA
from sklearn.multiclass import OneVsOneClassifier
from search import search
from GUI import*

def SName(path_img):
    dr = pd.read_csv("leaf_data.csv")

    #adding target column in our dataset

    target = list(range(47))  #for number of  different plant
   species
    K = 10                    # number of samples
    target_n =  [ele for ele in target for i in range(K)]
    #print(target_n)

    X = dr.iloc[:,1:]  #to remove unnamed first column in
   dataframe


    pca = PCA()
    pca.fit(X)
    print(X)

    X_train, X_test, y_train, y_test = train_test_split(X,
```

```
    target_n, test_size=0.33,random_state=42)


    #scaling

    sc_X = StandardScaler()
    X_train = sc_X.fit_transform(X_train)
    X_test = sc_X.transform(X_test)


    #prediction
    #clf = OneVsOneClassifier(svm.SVC(C=100)).fit(X_train,
    y_train)

    clf = svm.SVC(decision_function_shape='ovo',C=100)
    clf.fit(X_train,y_train)

    #parameter tuning

    parameters = [{'kernel': ['rbf'],
                'gamma': [1e-4, 1e-3, 0.01, 0.1, 0.2, 0.5],
                'C': [1, 10, 100, 1000]},
               {'kernel': ['linear'], 'C': [1, 10, 100, 1000]}
              ]


    svm_clf = GridSearchCV(svm.SVC(decision_function_shape='ovo'
    ), parameters,cv=2)
    svm_clf.fit(X_train, y_train)

    print('best para = ',svm_clf.best_params_)
    y_pred_svm = svm_clf.predict(X_test)


    print('score after tuning',metrics.accuracy_score(y_test,
    y_pred_svm))

    print(metrics.classification_report(y_test, y_pred_svm))




    #   TEST ___ IMAGE

    titles = ['area','perimeter','aspect_ratio','rectangularity'
    ,'circularity',
          'contrast','correlation','inverse_difference_moments',
    'entropy',
            'mean_r','mean_g','mean_b','std_r','std_g','std_b'
            ]



    df = pd.DataFrame([], columns=titles)
    img_path = path_img
    img_path = img_path.replace("/", "\\")
    test_image = cv2.imread(img_path)
    #cv2.imshow('Test Image',test_image)
```

```
    #cv2.waitKey(1000)
    #cv2.destroyAllWindows()

    # texture
    fv_texture = texture(test_image)
    # shape
    fv_shape = shape(test_image)
    # colour
    fv_colour = colour(test_image)

    fv_final = fv_shape + fv_texture + fv_colour

    #print(fv_final)
    df_temp_cap = pd.DataFrame([fv_final], columns=titles)
    df_cap = df.append(df_temp_cap)
    X_final = df_cap
    df_cap_trans = sc_X.transform(df_cap)
    X_final  = df_cap_trans
    ########
    print('features of leaf being tested\n',X_final)
    # print(df_cap_trans)
    y_pred_test = svm_clf.predict(X_final)



    print('Prediction: ', y_pred_test[0])

    leaf_names = ['Azadirachta indica',
    'Ocimum tenuiflorum',
    'Saraca asoca',
    'Holoptelea integrifolia',
    'Eupatorium ayyappana',
    'Ocimum basilicum',
    'Laurus nobilis',
    'Elettaria cardamomum',
    'Canabis',
    'Tinospora cordifolia',
    'Solanum torvum',
    'Piper betle ',
    'Wrightia Tinctoria Pala',
    'Zingiber officinale' ,
    'Helicteres isora',
    'Phyllostachys edulis ',
    'Aesculus chinensis',
    'Berberis anhweiensis Ahrendt',
    'Cercis chinensis',
    'Indigofera tinctoria L.',
    'Phoebe nanmu (Oliv.) Gamble',
    'Acer Palmatum',
    'Kalopanax septemlobus (Thunb. ex A.Murr.) Koidz.',
    'Cinnamomum japonicum Sieb',
    'Koelreuteria paniculata Laxm.',
    'Ilex macrocarpa Oliv',
    'Pittosporum tobira (Thunb.) Ait. f.',
    'Chimonanthus praecox L.',
    'Cinnamomum camphora (L.) J. Presl',
    'Viburnum awabuki K.Koch',
    'Osmanthus fragrans Lour.',
```

```
    'Cedrus deodara (Roxb.) G. Don',
    'Ginkgo biloba L.',
    'Lagerstroemia indica (L.) Pers.',
    'Nerium oleander L.',
    'Podocarpus macrophyllus (Thunb.) Sweet',
    'Prunus serrulata Lindl. var. lannesiana auct.',
    'Ligustrum lucidum Ait. f.',
    'Tonna sinensis M. Roem.',
    'Prunus persica (L.) Batsch',
    'Manglietia fordiana Oliv.',
    'Acer buergerianum Miq.',
    'Mahonia bealei (Fortune) Carr.',
    'Magnolia grandiflora L.',
    'Populus xcanadensis Moench',
    'Liriodendron chinense (Hemsl.) Sarg.',
    'Citrus reticulata Blanco'
    ]

    species_name = leaf_names[y_pred_test[0]]

    print('Detected Species = ' , species_name)

    return species_name
```

While training the classifier ,store.py calls functions for shape features, texture features and color moments which are shape.py , texture.py, colour.py .

Shape based features using OpenCV package for drawContours(), shape.py :

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
def shape(img):
    img = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
    #plt.imshow(img)
    #plt.show()

    #mask
    lower = np.array([0,30,0])
    higher = np.array([250,250,250])
    mask = cv2.inRange(img,lower,higher)
    #plt.imshow(mask,'gray')
    #plt.show()

    cont,_ = cv2.findContours(mask,cv2.RETR_EXTERNAL,cv2.
   CHAIN_APPROX_NONE)
```

```
17
18      #seperating contour with max area
19      c = max(cont,key = cv2.contourArea)
20
21
22      c_img = cv2.drawContours(img,c,-1,255,3)
23      #plt.imshow(c_img)
24      #plt.show()
25      #shape features
26      M = cv2.moments(c)
27      area = cv2.contourArea(c)
28      perimeter = cv2.arcLength(c,True)
29      x,y,w,h = cv2.boundingRect(c)
30      aspect_ratio = float(w)/h
31      rectangularity = w*h/area
32      circularity = ((perimeter)**2)/area
33
34
35      vect = [area,perimeter,aspect_ratio,rectangularity,circularity]
36
37      return vect
```

Texture based features using mahotas package for haralick features, texture.py :

```
1
2  import mahotas as mt
3  import numpy as np
4  import cv2
5
6
7  def texture(img):
8      gs  = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
9      #cv2.imshow('gray',gs)
10     #cv2.waitKey(500)
11     #cv2.destroyAllWindows()
12     textures = mt.features.haralick(gs)
13     ht_mean = textures.mean(axis=0)
14     contrast = ht_mean[1]
15     correlation = ht_mean[2]
```

```
    inverse_diff_moments = ht_mean[4]
    entropy = ht_mean[8]
    vect = [contrast,correlation,inverse_diff_moments,entropy]

    return vect
```

Color based features, colour.py :

```

from preprocess import *
def colour(main_img):
    #main_img= preprocess(main_img)
    img = cv2.cvtColor(main_img, cv2.COLOR_BGR2RGB)

    # Colour features
    red_channel = img[:, :, 0]
    green_channel = img[:, :, 1]
    blue_channel = img[:, :, 2]

    blue_channel[blue_channel == 255] = 0
    green_channel[green_channel == 255] = 0
    red_channel[red_channel == 255] = 0
    #mean values
    red_mean = np.mean(red_channel)
    green_mean = np.mean(green_channel)
    blue_mean = np.mean(blue_channel)
    #std values
    red_std = np.std(red_channel)
    green_std = np.std(green_channel)
    blue_std = np.std(blue_channel)
    vect = [red_mean,green_mean,blue_mean,red_std,green_std,blue_std]
    return vect
```

For browser control , webbrower package for Python is used which opens a new window in preferred browser and redirects to web page containing uses and properties of the species.

```
import webbrowser
```

```
def search(species):
    url = 'https://www.google.com/search?q=' + species + ' uses and properties'
    url = url.replace(' ', '+')
    webbrowser.register('chrome',
                        None,
                        webbrowser.BackgroundBrowser(
                            "C:\Program Files\Google\Chrome\Application\chrome.exe"))
    webbrowser.get('chrome').open(url)
```

For user interface tkinter package for Python is used to developed User friendly custom GUI.

```
import tkinter as tk
from PIL import ImageTk,Image
from tkinter import filedialog
path = "path"
from PIL import Image,ImageTk
import tkinter.font as font
from testing import SName
from search import search
species = "species"


def GUImaker():
    #intial steps
    window = tk.Tk()
    window.title("Identification of Ayurvedic Medicinal Plants by Image Processing of Leaf Samples")
    window.attributes('-fullscreen', True)
    window.configure(bg="#c5e0b4")
    my_menu = tk.Menu(window)
    window.config(menu = my_menu)

    #functions
    def open():
        global path
        global my_image
```

```
25        window.filename = filedialog.askopenfilename(initialdir="/",
  title="Select A File",
26                                                     filetypes=(("jpg files
  ", "*.jpg"), ("all files", "*.*")))
27        my_label = tk.Label(window, text=window.filename).grid(row=3,
  column=1)
28        my_image = Image.open(window.filename)
29        my_image = my_image.resize((350, 140), Image.ANTIALIAS)
30        my_image = ImageTk.PhotoImage(my_image)
31
32        my_image_label = tk.Label(image=my_image).grid(row=1,column
  =1)
33        path = window.filename
34
35
36    def Name():
37        species = SName(path)
38        l1.config(text="BOTANICAL NAME = "+species)
39
40    def close():
41        search(SName(path))
42        window.destroy()
43
44    #Creating sections and widgets of app
45    file_menu = tk.Menu(my_menu)
46    my_menu.add_cascade(label="File",menu=file_menu)
47    file_menu.add_command(label="Exit",command=window.quit)
48
49
50    myFont = font.Font(size=15)
51    bot_button = Image.open(r"D:\myapp\Botanicalname.png")
52    bot_button = bot_button.resize((1500,190), Image.ANTIALIAS)
53    bot_button = ImageTk.PhotoImage(bot_button)
54
55    select_button = Image.open(r"D:\myapp\select.png")
56    select_button = select_button.resize((1500,190), Image.ANTIALIAS)
57    select_button = ImageTk.PhotoImage(select_button)
58
59    use_button = Image.open(r"D:\myapp\uses.png")
```

```
60      use_button = use_button.resize((1500,190), Image.ANTIALIAS)
61      use_button = ImageTk.PhotoImage(use_button)
62
63      img = Image.open(r"D:\myapp\logo.png")
64      img = img.resize((1500, 190), Image.ANTIALIAS)
65      bg_img = ImageTk.PhotoImage(img)
66      label =tk.Label(window,image =bg_img,bg="#c5e0b4",height =180,
        width=640)
67      label.grid(row=0,column=0)
68      #my_textlabel = tk.Label(window,text = "Identification of
        Ayurvedic Medicinal Plants by Image Processing of Leaf Samples \n
        Select Query Image",font=("Arial", 25),
69                              #fg="white",bg="green",width =400,height
        =2 ).pack()
70
71      my_btn = tk.Button(window,image =select_button,bg="#c5e0b4",
        height =180,width=640,activebackground="#c5e0b4",borderwidth=0,
        command= open)
72      my_btn['font'] = myFont
73      my_btn.grid(row=1,column=0)
74
75      name_btn = tk.Button(window,image =bot_button,bg="#c5e0b4",height
         =180,width=640,activebackground="#c5e0b4",borderwidth=0, command=
        Name)
76      name_btn['font'] = myFont
77      name_btn.grid(row=2,column=0)
78
79      use_btn = tk.Button(window,image =use_button,bg="#c5e0b4",height
        =180,width=640,activebackground="#c5e0b4",borderwidth=0, command=
        close)
80      use_btn['font'] = myFont
81      use_btn.grid(row=3,column=0)
82
83      l1 = tk.Label(window, text="BOTANICAL NAME = DEFAULT")
84      l1.grid(row=2,column=1)
85
86  #instructions label
87      instruct1 = Image.open(r"D:\myapp\instruct1.png")
88      instruct1 = instruct1.resize((600, 190), Image.ANTIALIAS)
```

```
    instruct1 = ImageTk.PhotoImage(instruct1)
    l2 = tk.Label(window, image =instruct1,bg="#c5e0b4",height =178,
width=640)
    l2.grid(row=0, column=2)

    instruct2 = Image.open(r"D:\myapp\instruct2.png")
    instruct2 = instruct2.resize((600, 190), Image.ANTIALIAS)
    instruct2 = ImageTk.PhotoImage(instruct2)
    l3 = tk.Label(window, image=instruct2, bg="#c5e0b4", height=178,
width=640)
    l3.grid(row=1, column=2)

    instruct3 = Image.open(r"D:\myapp\instruct3.png")
    instruct3 = instruct3.resize((600, 190), Image.ANTIALIAS)
    instruct3 = ImageTk.PhotoImage(instruct3)
    l4 = tk.Label(window, image=instruct3, bg="#c5e0b4", height=178,
width=640)
    l4.grid(row=2, column=2)

    instruct4 = Image.open(r"D:\myapp\instruct4.png")
    instruct4 = instruct4.resize((600, 190), Image.ANTIALIAS)
    instruct4 = ImageTk.PhotoImage(instruct4)
    l5 = tk.Label(window, image=instruct4, bg="#c5e0b4", height=178,
width=640)
    l5.grid(row=3, column=2)

    instruct5 = Image.open(r"D:\myapp\instruct5.png")
    instruct5 = instruct5.resize((600, 190), Image.ANTIALIAS)
    instruct5 = ImageTk.PhotoImage(instruct5)
    l6 = tk.Label(window, image=instruct5, bg="#c5e0b4", height=178,
width=640)
    l6.grid(row=4, column=2)

    window.mainloop()
```

CHAPTER 6

RESULTS

6.1 Classifier Results

Classifier is trained on total of 470 sample images including 47 different plant species with 10 samples each. Dataset is divided in two sets : Training and Testing with 315 and 155 images respectively.

Features	Classifier	Accuracy	Dataset	Training	Testing	Species	Samples
Shape, texture, color	SVM	86.5%	470	315	155	47	10

```
..          ...           ...           ...  ...        ...        ...        ...
465  693327.0  4429.687838      1.346591  ...  36.301181  64.008354  28.171358
466  626601.0  4342.280765      1.378026  ...  44.432803  67.482810  30.603159
467  663465.5  4266.332500      1.430815  ...  46.771753  71.488590  27.488877
468  719563.5  4449.871604      1.397338  ...  41.220509  68.795816  27.017941
469  760375.0  4479.653348      1.410765  ...  37.616911  62.360694  28.105651

[470 rows x 15 columns]
best para =  {'C': 10, 'kernel': 'linear'}
score after tuning 0.8653846153846154
```

Figure 6.1: Classifier Results

6.2 User Interface

For User friendly outlook, a Graphical User Interface (GUI) (graphical user interface) is developed which allows user to select query image,display its botanical name and to find uses and properties of identified plant. Tkinter package for Python provides all required functions and utilities to develop a GUI. Custom buttons and labels are used to match the theme.

- Main page of application containing several buttons to interact with on left side and instruction manual on right Fig 6.2.
- Pressing SELECT button to open file browser for query image selection Fig 6.3.
- Pressing BOTANICAL NAME button to display botanical name of species Fig 6.4.
- Pressing PROPERTIES AND USES button to display uses and properties of identified species Fig 6.5.

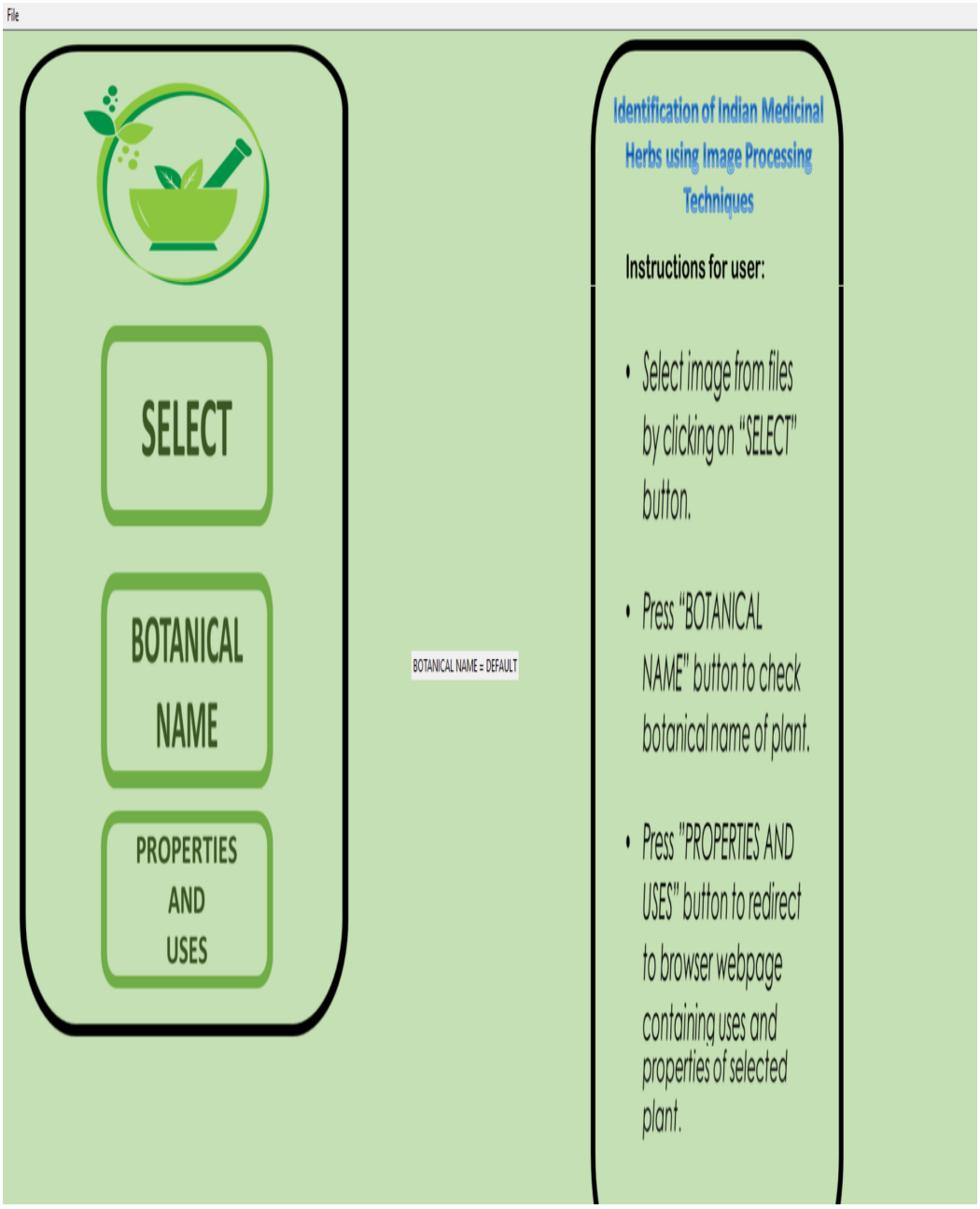

Figure 6.2: Main page of User Interface

Figure 6.3: SELECT button for query image selection

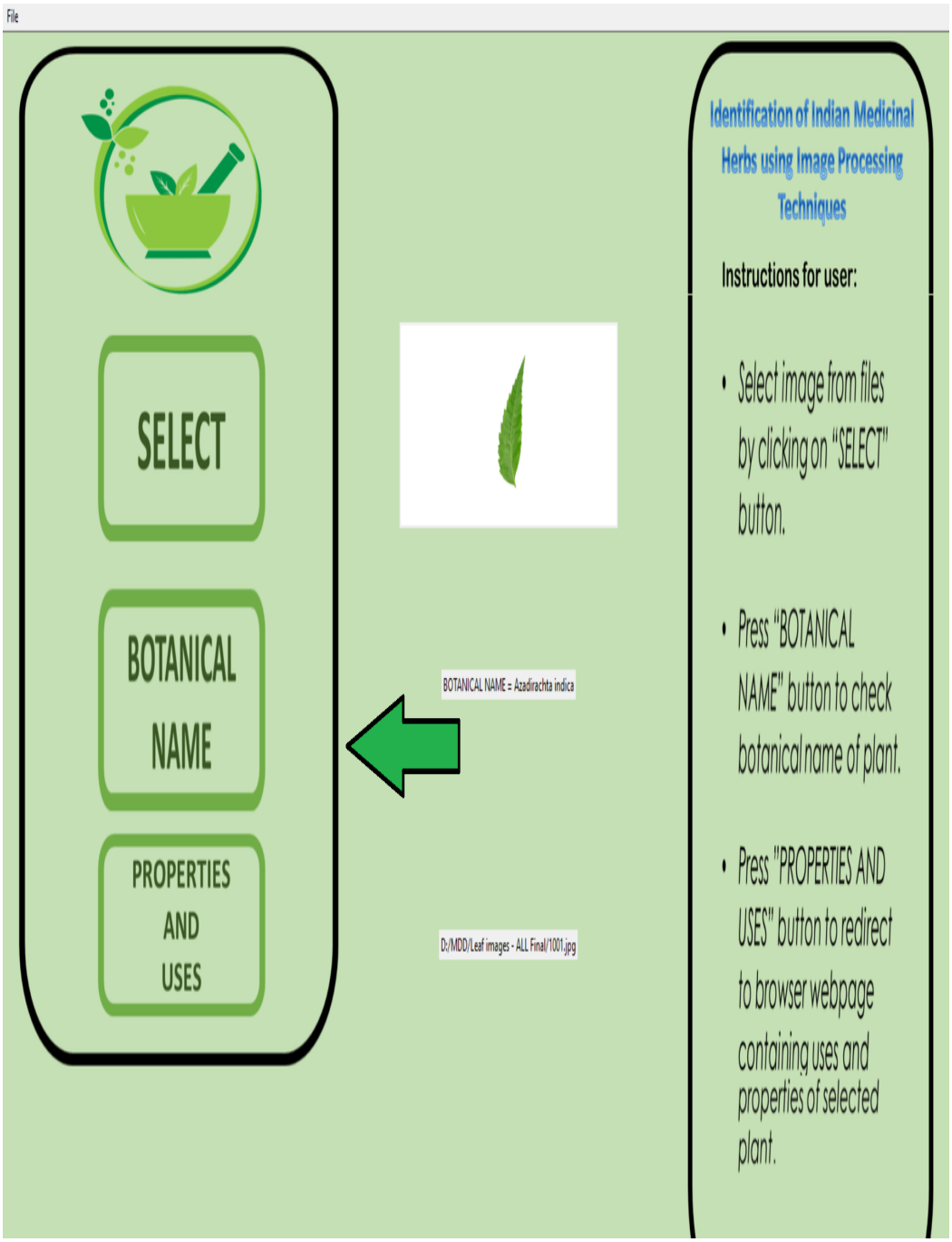

Figure 6.4: BOTANICAL NAME button to display botanical name

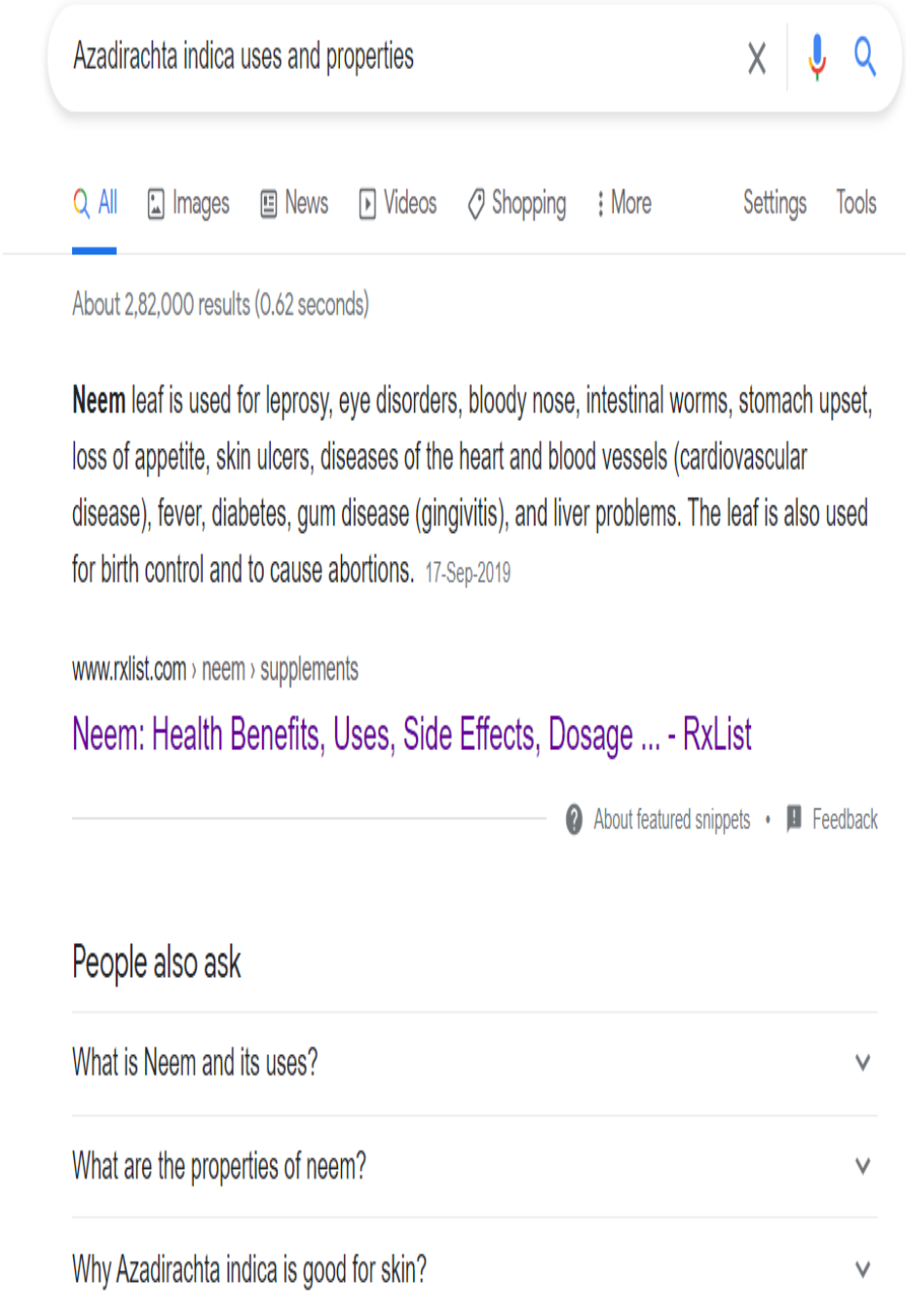

Figure 6.5: Uses and properties of identified species on web page

CHAPTER 7

CONCLUSION

Study of medicines plays an important role in our lives. It is a never ending process and needs improvement and enhancement in techniques as and when needed. Ayurveda is an ancient practice which used to boost immunity and cure plethora of ailments. Ayurveda is the study of different medicinal plants and their uses. In this model different algorithms like contouring, feature extraction are performed on leave samples and finally by the use of SVM classifier, system displays the botanical name of the plant .This botanical name is fed to browser control feature which directs user to web page with uses and properties of the sample leaf . This incorporation reduces the on-system memory usage.

CHAPTER 8

FUTURE ENHANCEMENT

- Future scope of the study includes augmentation of new features in feature vector like venation patterns of leaf.Venation patterns will benefit the study in ways like scale-space analysis which could be a promising research direction to pursue.
- In addition to this, more number of samples per species can enhance the accuracy of the model.Having a huge number of species in a training set can lead to a better understanding of the model, thereby, giving us more precise results.
- Moreover, drafting of a standard database of Ayurvedic plants can be supportive in doing the analysis of the training model and would give more insightful results with better understanding of species.
- A standalone dedicated hardware implementing the algorithms used in our model can be constructed using digital circuitry like microprocessors, microcontrollers by feeding them with the codes.
- Software applications based on this methodology can be developed for portable devices. A simple example would be an app based on the methodologies used in this project.
- A cloud based storage for dataset will further reduce the on-system memory requirements. This will enhance our memory storage and will definitely speed up the process.

APPENDIX A

MACHINE LEARNING FOR CLASSIFICATION

A.1 Support vector machine

Support vector machine or SVM is a machine learning algorithm based on statistical learning theory by Vapnik et al[16] which is useful for training sets containing limited number of samples [15] .SVM is mainly used in classification and database technology.It is based on the principle of structural risk minimization which is different from conventional empirical risk minimization and thus allows learning on small number of samples.Basic concept of SVM consists of a decision boundary or Hyperplane which separates datapoints from different classes.This Hyperplane is a N dimensional plane where N is directly connected to number of features used for classification.

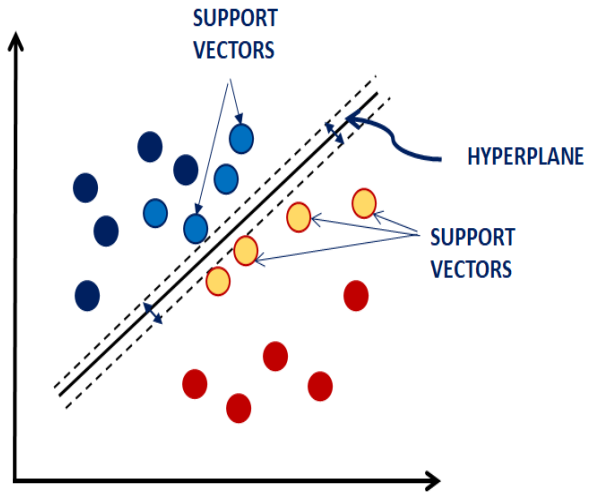

Figure A.1: Hyperplane separating two classes

APPENDIX B

B.1 Contours and Edge detection

OpenCV package for Python programming language allows user to find and draw contours from binary image .The function "cv.findContours()" is based on the algorithm of Suzuki Be (1985) [17].Contours are usually a curve joining continuous points having same color or intensity near the boundaries.These contours encloses every single closed shape in a test image providing ability to separate required object from its background.This contour enclosing the object, then supplies with morphological or shape based features such as area, perimeter etc.

Figure B.1: Contour enclosing the ball on grass

REFERENCES

[1] Dahigaonkar, Tejas D., and R. Kalyane. "Identification of ayurvedic medicinal plants by image processing of leaf samples." International Research Journal of Engineering and Technology (IRJET) 5, no. 05 (2018).

[2] Kalyoncu, Cem, and Önsen Toygar. "Geometric leaf classification." Computer Vision and Image Understanding 133 (2015): 102-109.

[3] Beghin, Thibaut, James S. Cope, Paolo Remagnino, and Sarah Barman. "Shape and texture based plant leaf classification." In International conference on advanced concepts for intelligent vision systems, pp. 345-353. Springer, Berlin, Heidelberg, 2010.

[4] Nourhan Zayed, Heba A. Elnemr, "Statistical Analysis of Haralick Texture Features to Discriminate Lung Abnormalities", International Journal of Biomedical Imaging, vol. 2015, Article ID 267807, 7 pages, 2015. https://doi.org/10.1155/2015/267807

[5] R. C. Gonzalez and R. E. Woods, “Digital Image Processing”, Prentice hall, 2008

[6] Mathew. M. and Gopi V. P., “Transform based bleeding detection technique for endoscopic images”, 2nd International Conference on Electronics and Communication Systems (ICECS), pp. 1730-1734, Feb. 2015

[7] Kumar, P. and Surya, C. and Gopi, Varun. (2017). Identification of ayurvedic medicinal plants by image processing of leaf samples. 231-238. 10.1109/ICRCICN.2017.8234512.

[8] van der Walt, Stéfan and Schönberger, Johannes and Nunez-Iglesias, Juan and Boulogne, François and Warner, Joshua and Yager, Neil and Gouillart, Emmanuelle and Yu, Tony and contributors, the. (2014). scikit-image: Image processing in Python. PeerJ. 2. 10.7717/peerj.453.

[9] Elhariri E., El-Bendary N., Fouad M.M.M., Platos J., Hassanien A.E., Hussein A.M.M. (2014) Multi-class SVM Based Classification Approach for Tomato Ripeness. In: Abraham A., Kromer P., Snasel V. (eds) Innovations in Bio-inspired Computing and Applications. Advances in Intelligent Systems and Computing, vol 237. Springer, Cham.

[10] Sulc M., Matas J. (2015) Texture-Based Leaf Identification. In: Agapito L., Bronstein M., Rother C. (eds) Computer Vision - ECCV 2014 Workshops. ECCV 2014. Lecture Notes in Computer Science, vol 8928. Springer, Cham.

[11] Kan, H. and Jin, L. and Zhou, F.. (2017). Classification of medicinal plant leaf image based on multi-feature extraction. Pattern Recognition and Image Analysis. 27. 581-587. 10.1134/S105466181703018X.

[12] D. Venkataraman and N. Mangayarkarasi, "Support vector machine based classification of medicinal plants using leaf features," 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2017, pp. 793-798, doi: 10.1109/ICACCI.2017.8125939.

[13] Maheswari, A. and Bharathi, N. and Neelamegam, Periasamy and Gayathridevi, T.. (2014). Classification and recognition of herbal leaf using SVM algorithm. Research Journal of Pharmaceutical, Biological and Chemical Sciences. 5. 415-423.

[14] R. Janani and A. Gopal, "Identification of selected medicinal plant leaves using image features and ANN," 2013 International Conference on Advanced Electronic Systems (ICAES), 2013, pp. 238-242, doi: 10.1109/ICAES.2013.6659400.

[15] Zhang Y. (2012) Support Vector Machine Classification Algorithm and Its Application. In: Liu C., Wang L., Yang A. (eds) Information Computing and Applications. ICICA 2012. Communications in Computer and Information Science, vol 308. Springer, Berlin, Heidelberg.

[16] Vapnik, V.N.: Statistical learning theory. Springer, New York (1995)

[17] Suzuki, S., and Be, K. (1985). Topological structural analysis of digitized binary images by border following. Computer Vision, Graphics, and Image Processing 30, 32–46. doi:10.1016/0734-189X(85)90016-7.