A MULTI-STAGE APPROACH FOR CLASSIFYING DRY MEDICINAL HERBS USING DEEP CNN

A THESIS

Submitted by

KIRAN B
(2023188035)

in partial fulfilment for the award of the degree of

MASTER OF ENGINEERING IN
COMPUTER SCIENCE & ENGINEERING (spl in BDA)

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING
COLLEGE OF ENGINEERING, GUINDY
ANNA UNIVERSITY, CHENNAI

DECEMBER 2024

ANNA UNIVERSITY, CHENNAI

BONAFIDE CERTIFICATE

Certified that this Report titled "**A MULTI-STAGE APPROACH FOR CLASSIFYING DRY MEDICINAL HERBS USING DEEP CNN**" is the bonafide work of **KIRAN B (2023188035)** who carried out the work under my supervision. Certified further that to the best of my knowledge the work reported here in does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

Project Coordinator

Dr. V. VERTRISELVI

Professor,
Department of Computer Science
and Engineering
College of Engineering, Guindy
Anna University,
Chennai-600025.

Supervisor

Dr. G.S. MAHALAKSHMI

Associate Professor,
Department of Computer Science
and Engineering
College of Engineering, Guindy
Anna University,
Chennai-600025

Professor and Head

Dr. V. MARY ANITA RAJAM

Professor & Head,
Department of Computer Science and Engineering
College of Engineering, Guindy
Anna University, Chennai-600025.

ABSTRACT

The convergence of advanced technology and traditional medicine offers transformative potential in healthcare, particularly through the integration of artificial intelligence and traditional medicinal practices. This project presents the development of a mobile application designed to identify dried medicinal herbs and provide personalized health recommendations based on traditional knowledge systems such as Ayurveda. Leveraging a robust machine learning framework, the application combines multiple feature extraction networks—W-LeafNet, S-LeafNet, and P-LeafNet—to classify herbs accurately and deliver detailed insights into their therapeutic properties.

The herb identification module employs advanced image processing techniques, including normalization, Sobel edge detection, and Gabor filters, followed by multimodal feature extraction optimized for local and global attributes. W-LeafNet, the most effective model, achieved a remarkable accuracy of 99.09%, underscoring its reliability and precision. S-LeafNet demonstrated solid performance with 94.53% accuracy, while P-LeafNet exhibited moderate success, achieving 36.30% accuracy but requiring further optimization. The combination of these models, supported by confidence validators and stacked generalization, ensures robust and accurate herb classification.

In addition to herb identification, the application integrates a comprehensive herb information database to provide users with detailed knowledge about the medicinal properties and uses of identified herbs. This functionality bridges the gap between traditional herbal knowledge and modern technological capabilities, fostering advancements in personalized medicine and traditional healthcare practices.

By combining state-of-the-art machine learning with traditional medicinal expertise, this project aims to enhance the accessibility and accuracy of herbal medicine, offering valuable tools for both practitioners and researchers in the field.

ABSTRACT – TAMIL

மேம்பட்ட தொழில்நுட்பம் மற்றும் பாரம்பரிய மருத்துவத்தின் ஒருங்கிணைப்பு, குறிப்பாக செயற்கை நுண்ணறிவு மற்றும் பாரம்பரிய மருத்துவ முறைகளின் ஒருங்கிணைப்பு மூலம், சுகாதாரப் பாதுகாப்பில் உருமாறும் திறனை வழங்குகிறது. உலர்ந்த மருத்துவ மூலிகைகளை அடையாளம் காணவும், ஆயுர்வேதம் போன்ற பாரம்பரிய அறிவு அமைப்புகளின் அடிப்படையில் தனிப்பயனாக்கப்பட்ட சுகாதார பரிந்துரைகளை வழங்கவும் வடிவமைக்கப்பட்ட மொபைல் பயன்பாட்டின் வளர்ச்சியை இந்த திட்டம் வழங்குகிறது. ஒரு வலுவான இயந்திர கற்றல் கட்டமைப்பை மேம்படுத்துவதன் மூலம், பயன்பாடு பல அம்சங்களை பிரித்தெடுக்கும் நெட்வொர்க்குகளை ஒருங்கிணைக்கிறது—W-LeafNet, S-LeafNet மற்றும் P-LeafNet—மூலிகைகளை துல்லியமாக வகைப்படுத்தி அவற்றின் சிகிச்சை பண்புகள் பற்றிய விரிவான நுண்ணறிவுகளை வழங்குகிறது.

மூலிகை அடையாளம் காணும் தொகுதியானது, இயல்பாக்கம், சோபல் எட்ஜ் கண்டறிதல் மற்றும் கேபர் வடிப்பான்கள் உள்ளிட்ட மேம்பட்ட பட செயலாக்க நுட்பங்களைப் பயன்படுத்துகிறது, அதைத் தொடர்ந்து உள்ளூர் மற்றும் உலகளாவிய பண்புகளுக்கு உகந்ததாக மல்டிமாடல் அம்சம் பிரித்தெடுக்கப்பட்டது. W-LeafNet, மிகவும் பயனுள்ள மாதிரி,

99.09% ஒரு குறிப்பிடத்தக்க துல்லியத்தை அடைந்தது, அதன் நம்பகத்தன்மை மற்றும் துல்லியத்தை அடிக்கோடிட்டுக் காட்டுகிறது. S-LeafNet 94.53% துல்லியத்துடன் திடமான செயல்திறனை வெளிப்படுத்தியது, அதே நேரத்தில் P-LeafNet மிதமான வெற்றியை வெளிப்படுத்தியது, 36.30% துல்லியத்தை அடைந்தது ஆனால் மேலும் மேம்படுத்தல் தேவைப்படுகிறது. இந்த மாதிரிகளின் சேர்க்கை, நம்பிக்கை சரிபார்ப்பாளர்களால் ஆதரிக்கப்படுகிறது மற்றும் அடுக்கப்பட்ட பொதுமைப்படுத்தல், வலுவான மற்றும் துல்லியமான மூலிகை வகைப்பாட்டை உறுதி செய்கிறது.

மூலிகை அடையாளத்துடன் கூடுதலாக, மருத்துவ குணங்கள் மற்றும் அடையாளம் காணப்பட்ட மூலிகைகளின் பயன்பாடுகள் பற்றிய விரிவான அறிவை பயனர்களுக்கு வழங்க, ஒரு விரிவான மூலிகை தகவல் தரவுத்தளத்தை பயன்பாடு ஒருங்கிணைக்கிறது. இந்த செயல்பாடு பாரம்பரிய மூலிகை அறிவு மற்றும் நவீன தொழில்நுட்ப திறன்களுக்கு இடையே உள்ள இடைவெளியைக் குறைக்கிறது, தனிப்பயனாக்கப்பட்ட மருத்துவம் மற்றும் பாரம்பரிய சுகாதார நடைமுறைகளில் முன்னேற்றங்களை ஊக்குவிக்கிறது.

பாரம்பரிய மருத்துவ நிபுணத்துவத்துடன் அதிநவீன இயந்திரக் கற்றலை இணைப்பதன் மூலம், இந்தத் திட்டம் மூலிகை மருத்துவத்தின் அணுகல் மற்றும் துல்லியத்தை மேம்படுத்துவதை நோக்கமாகக் கொண்டுள்ளது, இது துறையில் உள்ள பயிற்சியாளர்கள் மற்றும் ஆராய்ச்சியாளர்களுக்கு மதிப்புமிக்க கருவிகளை வழங்குகிறது.

ACKNOWLEDGEMENT

Over the past six months I have received immense support from a number of individuals and it gives me pleasure to thank all those who made this report possible. I would like to thank my Guide **Dr. G.S. Mahalakshmi**, Associate Professor, Department of Computer Science and Engineering who has been my mentor, and my well-wisher throughout the journey of my project. Her support and encouragement have always pushed me through when in doubts. I take this opportunity in extending my gratitude to our project review panel members, **Dr. C. BALAJI,** Assistant Professor, Department of Computer Science and Engineering and **Dr. R. BASKARAN**, Professor, Department of Computer Science and Engineering, **Dr. V. VERTRISELVI,** Professor, (Sr .Gr), Department of Computer Science and Engineering for their continuous support and encouragement in the completion of the project.

I would like to thank them for generously spending their time in analyzing the project work. Their valuable comments have guided me to take the right path in completion of the project. It is an honor to thank **Dr. V. MARY ANITA RAJAM** , Professor and Head of the Department, Department of Computer Science and Engineering, for her continuous encouragement for innovative ideas. I would also like to thank all research scholars, my friends and family for their encouragement and continued support.

KIRAN B

Table of Contents

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In recent years, the intersection of technology and traditional medicine has garnered significant attention due to its potential to enhance healthcare delivery and access. This project aims to explore this intersection by developing a comprehensive mobile application that integrates cutting-edge technologies with traditional medicinal practices. The focus of this project is to build an application that combines image recognition and recommendation systems to assist users in identifying medicinal plants and receiving personalized health recommendations based on traditional medicinal knowledge.

1.1 Identification of Medicinal Plants

Traditional medicine, particularly Ayurveda and other Indian medicinal practices, relies heavily on a diverse range of medicinal plants, each with specific therapeutic properties. Accurately identifying these plants is crucial for their effective use in treatments, but the manual identification process is often time-consuming and requires specialized knowledge. This challenge is further compounded by the variability in plant appearances and the vast number of plant species used in traditional medicine.

This project focuses on the development of a mobile application designed to accurately identify dried medicinal herbs using a combination of advanced machine learning models and image processing techniques. Leveraging a multimodal architecture, the system integrates multiple feature extraction networks, including

W-LeafNet, S-LeafNet, and P-LeafNet, to enhance the identification process. Each model is optimized for different aspects of herb recognition, from local features like texture and veins to global features such as shape and size. The goal is to provide precise herb classification and detailed information about each identified herb, supporting both research and practical applications in the field of herbal medicine.

The image processing workflow begins with standard techniques such as normalization, followed by Sobel edge detection and Gabor filters, which help to extract essential features from the images of dried herbs. The system then performs feature extraction using separate networks for local and global attributes, which are later combined to capture a comprehensive understanding of the herb's characteristics. The use of confidence validators in each model ensures that only predictions with high confidence are passed through, reducing the chances of misidentification. This layered approach, enhanced by a stacked generalization technique, ensures the robustness and accuracy of the final predictions.

The application not only serves as a tool for identifying herbs but also provides users with in-depth information about the medicinal properties of each identified herb. This is achieved through a connection to an herb information database, which the system consults after the herb is classified. By offering detailed insights and promoting accurate identification, the project aims to bridge the gap between traditional herbal knowledge and modern technology, contributing to advancements in personalized medicine, Ayurveda, and related fields.

1.2 Integration and Impact

The integration of medicinal dry herb identification into a mobile application aims to provide users with a comprehensive and accessible platform dedicated to traditional medicine. The application will allow users to identify various dry

medicinal herbs by analysing images, offering precise and detailed information about each herb, including its properties, traditional uses, and potential benefits.

Leveraging advanced machine learning techniques, the app ensures accurate identification and user-friendly interaction, making it a valuable resource for enthusiasts, researchers, and practitioners of traditional medicine. By seamlessly combining modern technology with the extensive knowledge of traditional medicinal practices, this project strives to enhance the accessibility, usability, and understanding of natural remedies. The resulting mobile application will serve as an innovative tool to bridge the gap between traditional health resources and contemporary technological advancements.

CHAPTER 2

LITERATURE SURVEY

The literature on plant identification has seen significant advancements through the application of machine learning and deep learning techniques, aiming to automate and improve the accuracy of plant species recognition. Various methodologies, from traditional machine learning approaches to modern deep learning models, have been explored to tackle this problem.

In recent years, machine learning has emerged as a powerful tool for plant leaf disease identification, allowing for effective decision-making and integration of expert knowledge. A comparative analysis of artificial intelligence techniques, including decision trees, logistic regression, and artificial neural networks (ANNs), highlights the flexibility of these methods in plant disease detection systems [1]. The use of support vector machines (SVMs) for classifying hyperspectral images and detecting diseases such as Huanglongbing in lemon trees further underscores the efficacy of these techniques, achieving significant accuracy [1]. Data augmentation techniques, such as image flipping and noise injection, have also been employed to enhance model performance, with one study achieving a classification accuracy of 96.46% [1].

In the context of mobile applications, ApLeaf represents an innovative Android-based system for automatic plant leaf identification. The application leverages a three-step process—leaf image segmentation, feature extraction, and species identification—to deliver real-time plant recognition capabilities, achieving

identification within 5 seconds on standard Android devices. ApLeaf's use of threshold segmentation and content-based image retrieval (CBIR) techniques allows it to match leaf features against a comprehensive database derived from the ImageCLEF competition, covering 126 tree species from the French Mediterranean area [paper 2]. While ApLeaf's offline functionality and user-friendly interface make it accessible for fieldwork, the requirement for high-quality images and a limited species database are noted drawbacks [2].

The classification of medicinal plant leaves has also been explored using image processing techniques, focusing on shape and texture feature extraction. A comparison of classifiers such as Back Propagation Neural Network (BP-NN), Probabilistic Neural Network (PNN), K-Nearest Neighbor (K-NN), and SVMs indicates that SVMs provide the most effective balance of recognition rate and computational efficiency [3]. The approach achieved a recognition rate of 93.3%, but its effectiveness depends heavily on the quality of input images and the selection of robust features, highlighting the need for extensive training data [3].

Another novel method, the D-Leaf approach, integrates Convolutional Neural Networks (CNN) for feature extraction with Artificial Neural Networks (ANN) for classification. Validated on datasets such as MalayaKew, Flavia, and Swedish Leaf Dataset, D-Leaf achieved notable accuracy rates, including 94.63% on the Flavia dataset and 98.09% on the Swedish dataset [4]. Despite these successes, the method's performance on the MalayaKew dataset was less favorable, attributed to the challenging background of the leaf images. This underlines the method's dependency on dataset quality and the significant computational resources required for CNN and ANN implementations [4].

Deep learning techniques have increasingly gained attention for their ability to handle complex feature extraction and classification tasks. A study utilizing a 50-layer deep residual network demonstrated a high accuracy rate of 93.09% on the

LeafSnap dataset, comprising 185 different tree species. The use of residual networks addresses common issues in deep learning, such as the vanishing gradient problem, and allows for the effective learning of features critical to plant identification [5]. However, the complexity of implementing such deep architectures and their dependence on high-quality, diverse datasets can be limiting factors [5].

Incorporating both handcrafted features and deep learning, another study achieved a state-of-the-art accuracy of 99.69% on the Flavia leaf dataset. This method combines shape, texture, and other leaf characteristics with neural network-based encoders for feature representation, ultimately using an SVM for classification [6]. The comprehensive feature analysis and combination of techniques highlight the potential for high accuracy, although the complexity and computational requirements of this approach may present challenges [6].

Finally, a smartphone application designed for the identification of Vietnamese Traditional Medicines demonstrates the practical application of deep learning in real-world settings. Utilizing a dataset of 10,000 images across 100 categories of Vietnamese herbs, the application employs CNNs to achieve an impressive accuracy rate of 99.275% [7]. The system's ability to process images in under 2 seconds makes it highly practical for real-time use, although its dependence on the dataset's specific conditions and the challenges of recognizing herbs in complex backgrounds remain areas for future improvement [7].

These studies collectively illustrate the breadth of approaches available for plant identification, from traditional machine learning methods to advanced deep learning techniques. Each approach offers unique advantages, such as high accuracy and real-time processing capabilities, but also faces challenges related to dataset quality, image requirements, and computational complexity. Future research directions could focus on enhancing model robustness, expanding species databases, and developing techniques that balance accuracy with computational efficiency, to better

support the diverse needs of ecological studies, traditional medicine, and agricultural practices.

CHAPTER 3

PROPOSED SYSTEM

3.1 BLOCK DIAGRAM

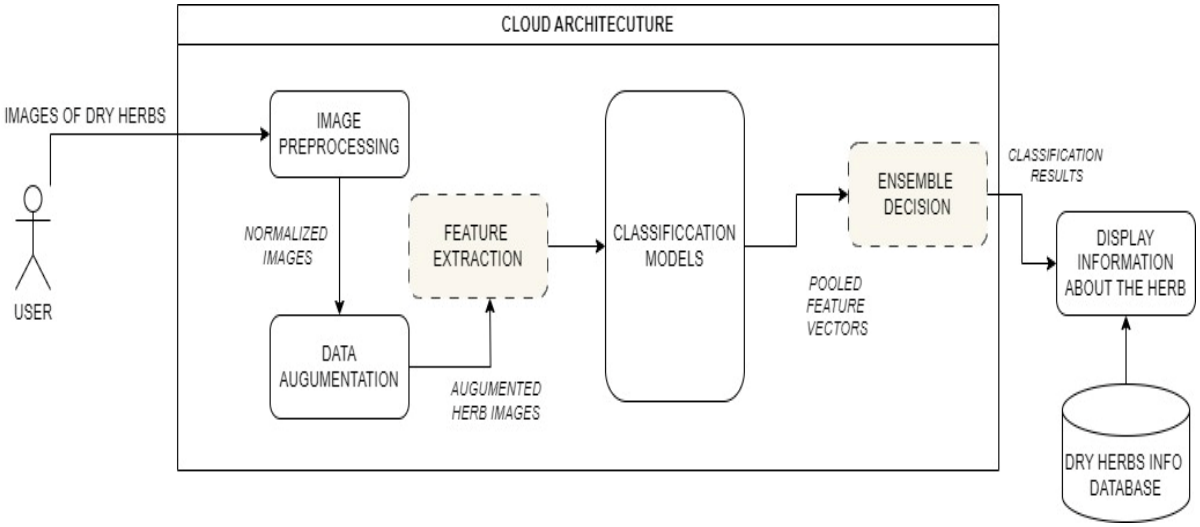


Figure 1: Block Diagram of the project.

High Level Diagram:

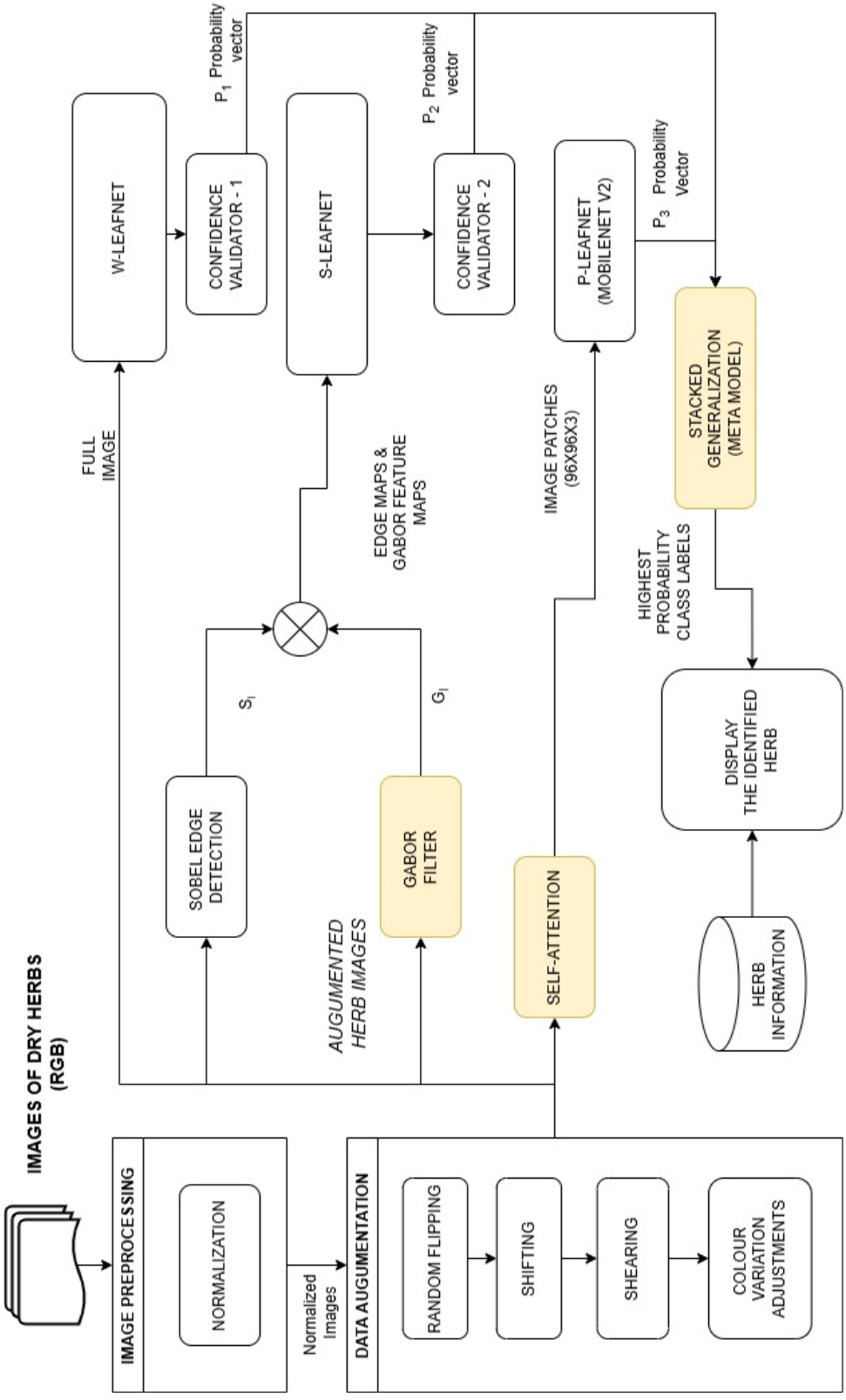


Figure 2 : Detailed architectutre diagram

3.2.1 IMAGE PREPROCESSING MODULE

Input: *Image (RGB image as a 3D tensor of size H x W x C)*

Output: *Normalized image (image with pixel values scaled between 0 and 1).*

Normalization generally scales pixel values from their original range (e.g., 0-255) to a target range (e.g., 0-1 or -1 to 1) using a formula such as:

$$Normalized\ Value = (pixelvalue - mean)/standard\ deviation$$

For basic normalization between 0 and 1, the formula would be:

$$Normalized\ Value\ =\ Pixel\ value\ /\ 255$$

Pseudo Code:

```
NormalizeImage(image)
   H, W, C = image.height, image.width, image.channels
   normalized_image = empty_tensor(H, W, C)

   for i = 1 to H do
      for j = 1 to W do
         for k = 1 to C do
            normalized_image[i][j][k] = image[i][j][k] / 255
         end for
      end for
   end for

   return normalized_image
End Algorithm
```

3.2.2 DATA AUGUMENTATION MODULE

Input: Normalized Image: An RGB image of a dry herb that has been normalized, typically represented as a 3D tensor with dimensions $H \times W \times C$ (Height, Width, Channels).

Output: Augmented Image Set: A collection of augmented images, which can include variations such as:

- *Flipped images (horizontal and/or vertical)*
- *Shifted images (translated in various directions)*
- *Sheared images (distorted along axes)*
- *Rotated images (if included as part of the augmentation)*
- *Adjusted images (with variations in brightness, contrast, saturation, etc.)*

Detailed Diagram:

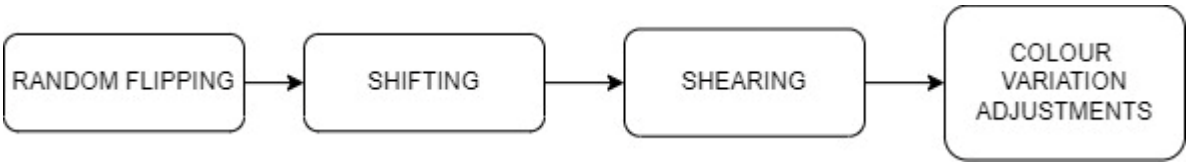


Figure 3: Flow Diagram of Data Augumentation Module

Algorithm:

Algorithm AugmentImage(normalized_image)

Input: normalized_image

Output: augmented_images

augmented_images = []

augmented_images.append(normalized_image)

flipped_image_h = FlipHorizontal(normalized_image)

augmented_images.append(flipped_image_h)

flipped_image_v = FlipVertical(normalized_image)
augmented_images.append(flipped_image_v)

shifted_image = Shift(normalized_image, x_shift=10, y_shift=0)
augmented_images.append(shifted_image)

rotated_image = Rotate(normalized_image, angle=30)
augmented_images.append(rotated_image)

brightness_adjusted_image = AdjustBrightness(normalized_image, factor=1.2)
augmented_images.append(brightness_adjusted_image)

contrast_adjusted_image = AdjustContrast(normalized_image, factor=1.5)
augmented_images.append(contrast_adjusted_image)

return augmented_images
End Algorithm

3.2.3 SOBEL EDGE DETECTION MODULE

Input: normalized_image
Output: edge_detected_image

The Sobel operator consists of two 3x3 convolution kernels, one for detecting edges in the x-direction and another for the y-direction.

Gradient Magnitude:

After convolving the image with these kernels, the gradients in the x and y directions (G_x and G_y),

$$G = \sqrt{G_x + G_y}$$

DETAILED DIAGRAM:

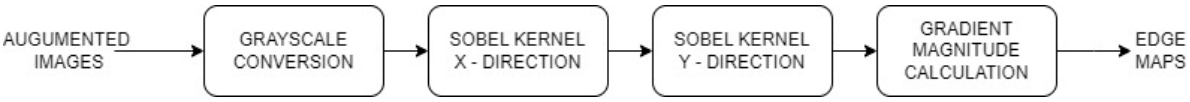


Figure 4 : Flow diagram of Sobel Edge Detection module

ALGORITHM:

Algorithm SobelEdgeDetection(normalized_image)

Input: normalized_image

Output: edge_detected_image

gray_image = ConvertToGrayscale(normalized_image)

sobel_x = ApplySobelKernel(gray_image, direction="x")

sobel_y = ApplySobelKernel(gray_image, direction="y")

edge_detected_image = CombineGradients(sobel_x, sobel_y)

return edge_detected_image

End Algorithm

3.2.4 Gabor Filters:

Input: image (grayscale or normalized), frequencies (list), orientations (list)

Output: feature_maps (list of filtered images)

A Gabor filter is a linear filter used for edge detection, texture analysis, and feature extraction. It is particularly effective for analyzing spatial frequencies in images. The Gabor filter is the product of a Gaussian function and a sinusoidal wave, which allows it to capture specific orientations and frequencies of texture patterns in an image.

Complex

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) \exp\left(i\left(2\pi\frac{x'}{\lambda} + \psi\right)\right)$$

Real

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) \cos\left(2\pi\frac{x'}{\lambda} + \psi\right)$$

Imaginary

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) \sin\left(2\pi\frac{x'}{\lambda} + \psi\right)$$

where

$$x' = x\cos\theta + y\sin\theta$$

and

$$y' = -x\sin\theta + y\cos\theta$$

ALGORITHM:

Algorithm GaborFilter(image, frequencies, orientations)

Input: image (grayscale or normalized), frequencies (list), orientations (list)

Output: feature_maps (list of filtered images)

Initialize feature_maps as an empty list

For each frequency in frequencies do

For each orientation in orientations do

Create Gabor kernel with the current frequency and orientation

Apply convolution of image with Gabor kernel

Store the result in feature_maps

End for

End for

Return feature_maps

End Algorithm

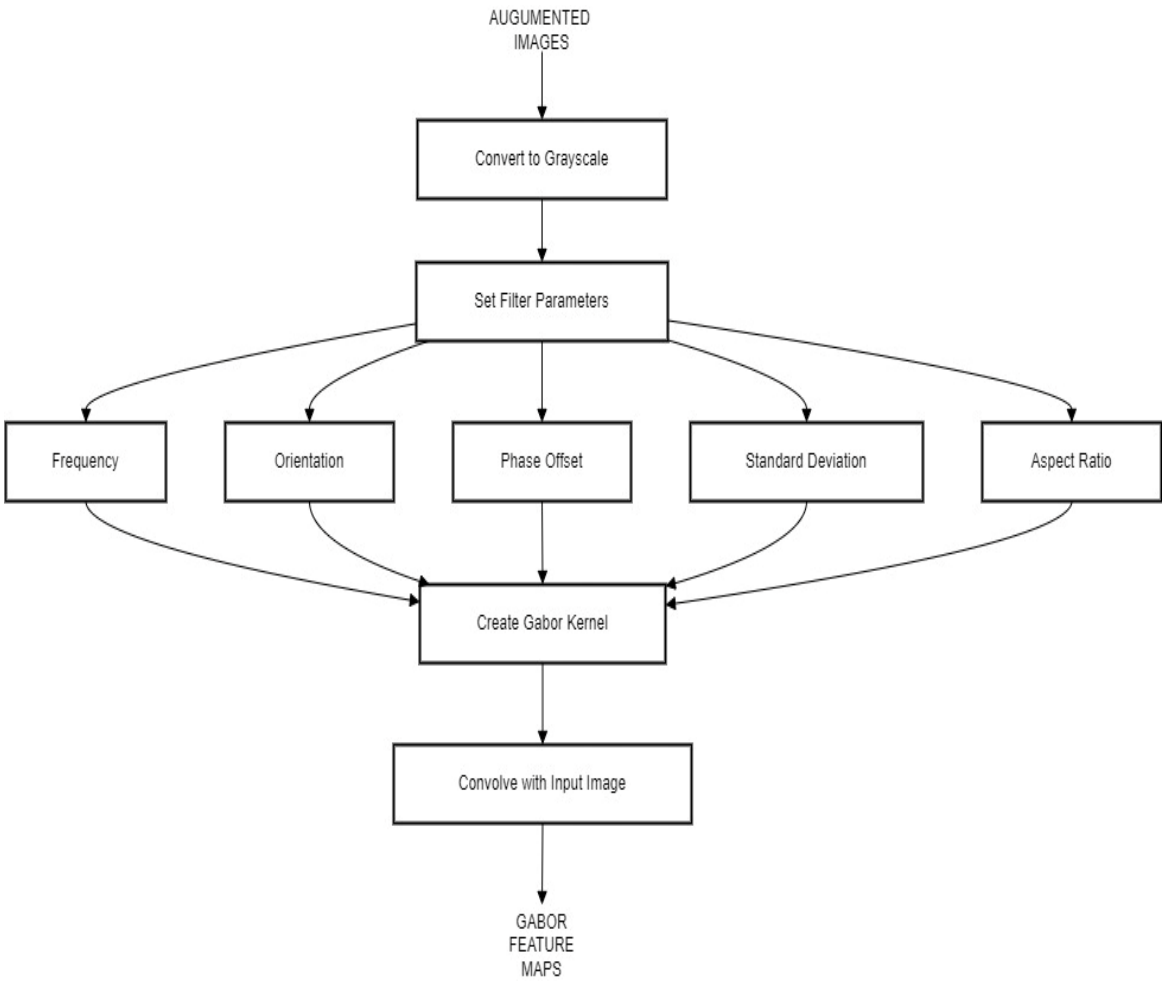


Figure 5 Gabor filter Flow diagram

3.2.5 S – leafNet

Input: Gabor feature maps and Sobel Maps

Output: Probability vector.

S-LeafNet is a specialized neural network architecture designed for leaf image classification and recognition. It leverages a combination of convolutional neural networks (CNNs) and feature extraction techniques to analyze leaf structures and characteristics effectively. The primary goals of S-LeafNet are to accurately identify different plant species based on their leaf morphology and to provide robust classification outcomes.

Algorithm:

```
function S_Leaf_Network(image):
    # Step 1: Preprocessing
    resized_image = resize(image, (224, 224))
    grayscale_image = convert_to_grayscale(resized_image)
    normalized_image = normalize(grayscale_image)

    # Step 2: Feature Extraction
    edges = edge_detection(normalized_image)
    shape_features = extract_shape_features(normalized_image)
    texture_features = extract_texture_features(normalized_image)

    # Step 3: S-Leaf Network Architecture
    model = Sequential()
    model.add(Conv2D(filters=32, kernel_size=(3, 3), activation='relu',
input_shape=(224, 224, 1)))
    model.add(MaxPooling2D(pool_size=(2, 2)))
    # Add more convolutional and pooling layers as needed
    model.add(Flatten())
    model.add(Dense(units=128, activation='relu'))
    model.add(Dense(units=num_classes, activation='softmax'))  # Probability
vector output

    # Step 4: Training
    model.compile(loss='categorical_crossentropy', optimizer='adam',
metrics=['accuracy'])
```

model.fit(training_data, training_labels, epochs=num_epochs, validation_split=0.2)

 # Step 5: Testing and Evaluation

predictions = model.predict(test_data)

probability_vectors = softmax(predictions) # Convert logits to probability vectors

 # Optional: Calculate accuracy

accuracy = calculate_accuracy(probability_vectors, test_labels)

return probability_vectors, accuracy

DETAILED ARCHITECTURE:

CBR : Convolutional, batch normalization and ReLU layers.

MP : Max Pooling.

AP : Average Pooling.

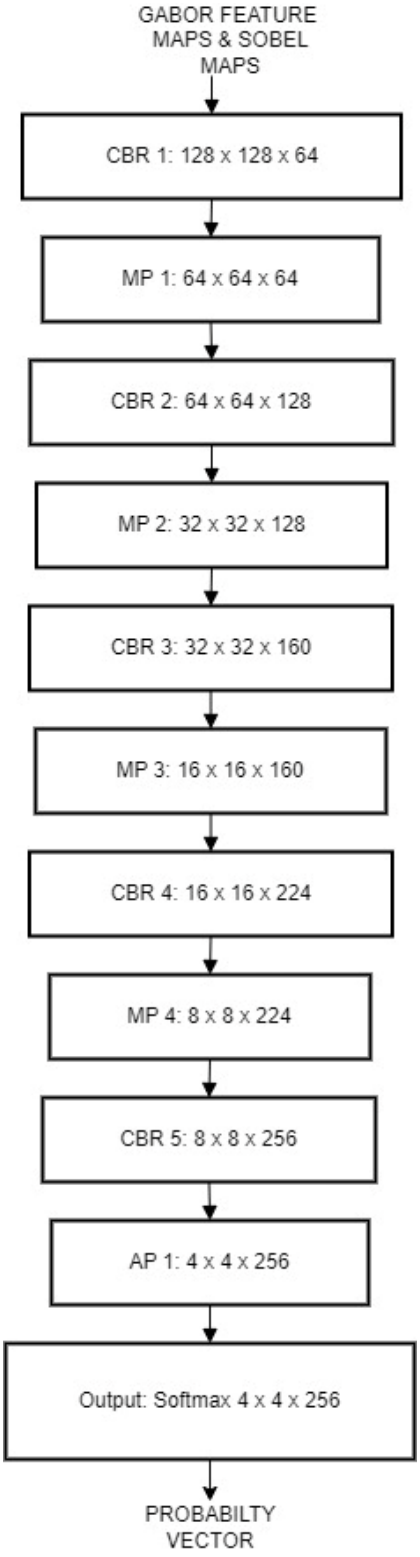


Figure 6 Flow Diagram of S-LeafNet

3.2.6 W-LEAFNET:

Input: Gabor feature maps and Sobel Maps.

Output: Probability vector.

W-LeafNet is an advanced neural network architecture specifically designed for the classification and recognition of leaf images. It combines deep learning techniques, particularly Convolutional Neural Networks (CNNs), to effectively analyze and categorize various plant species based on leaf morphology.

Key Features:

- Layered Structure: The architecture consists of multiple convolutional and pooling layers that extract hierarchical features from the input images, allowing the model to learn complex patterns and representations.
- Activation Functions: The ReLU activation function is employed to introduce non-linearity, enhancing the model's ability to capture intricate features in the data.
- Fully Connected Layers: After feature extraction, fully connected layers process the flattened features to make predictions about the class of the input image.
- Output Layer: A softmax layer at the end generates a probability vector, indicating the likelihood of the input image belonging to each of the predefined classes.

Algorithm:

Input:

- *Image: Input image of a leaf (H x W x C)*
- *Model Parameters: Trained weights and biases for each layer*

Output:

- Probability Vector: P(y|Image) for each class y

Begin

Initialize FeatureMap1 = Convolution(Image, Weights1, Biases1)

FeatureMap1 = Activation(ReLU, FeatureMap1)

FeatureMap1 = MaxPooling(FeatureMap1)

Initialize FeatureMap2 = Convolution(FeatureMap1, Weights2, Biases2)

FeatureMap2 = Activation(ReLU, FeatureMap2)

FeatureMap2 = MaxPooling(FeatureMap2)

Initialize FeatureMap3 = Convolution(FeatureMap2, Weights3, Biases3)

FeatureMap3 = Activation(ReLU, FeatureMap3)

FeatureMap3 = MaxPooling(FeatureMap3)

FlattenedFeatures = Flatten(FeatureMap3)

DenseLayer1 = FullyConnected(FlattenedFeatures, WeightsDense1, BiasesDense1)

DenseLayer1 = Activation(ReLU, DenseLayer1)

DenseLayer2 = FullyConnected(DenseLayer1, WeightsDense2, BiasesDense2)

DenseLayer2 = Activation(ReLU, DenseLayer2)

OutputLayer = FullyConnected(DenseLayer2, WeightsOutput, BiasesOutput)

ProbabilityVector = Softmax(OutputLayer)

Return ProbabilityVector

End

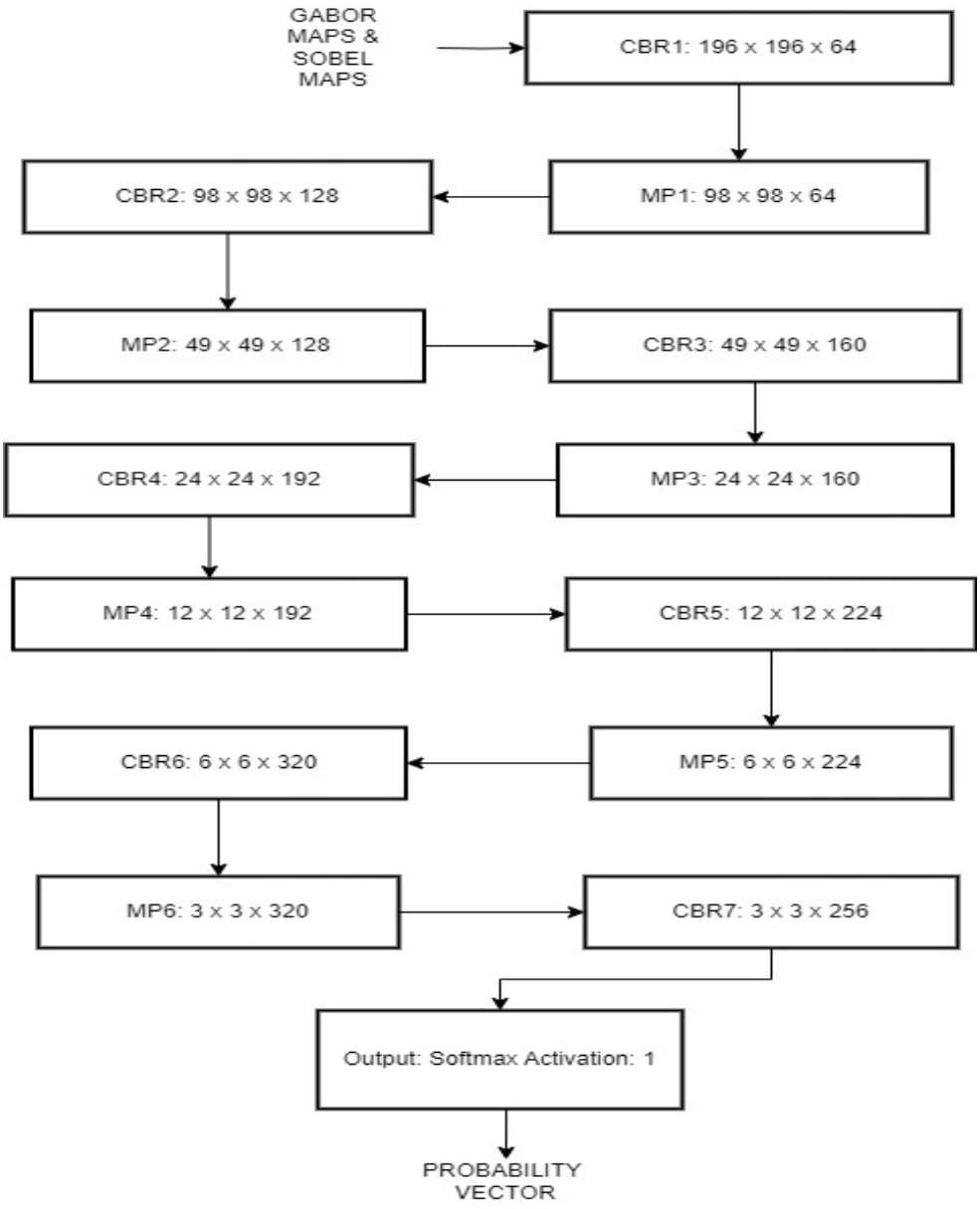


Figure 7 Flow Diagram of W-leafNet

3.2.7 Self-Attention Mechanism Module:

Input: augmented_images (Batch of images of size H x W x C)

Output: selected_patches (3 patches of size 96x96)

In this module, the input image is divided into a grid of patches, which are then transformed into a sequence of feature vectors. The self-attention mechanism computes attention scores between these patches to determine their interdependencies. Each patch attends to all other patches, allowing the model to focus on relevant areas while considering the overall context. This is particularly beneficial for tasks where the spatial relationship of objects within an image is critical, such as image classification, object detection, and segmentation.

Advantages:

The self-attention mechanism provides several advantages:

Global Context: It captures global relationships among patches, allowing for a deeper understanding of the image structure.

Dynamic Attention: The model can dynamically adjust its focus on different patches based on their relevance to the task, which enhances its ability to generalize.

Parallelization: Unlike recurrent neural networks, self-attention can process all patches simultaneously, leading to more efficient training and inference.

Algorithm:

Algorithm SelfAttentionModule(augmented_images)

Input: augmented_images (Batch of images of size H x W x C)

Output: selected_patches (3 patches of size 96x96)

for each image in augmented_images do

patches = extract_patches(image) // Extract patches from the image

attention_scores = compute_attention(patches) // Compute self-attention scores

important_patches = select_important_patches(patches, attention_scores) // Select patches based on scores

selected_patches.append(important_patches)

return selected_patches

End Algorithm

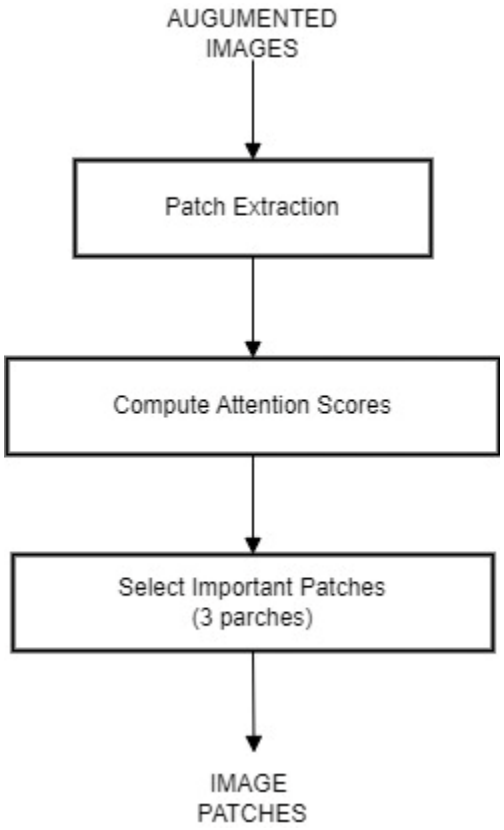
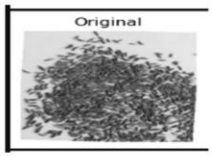
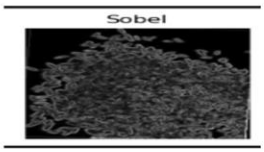
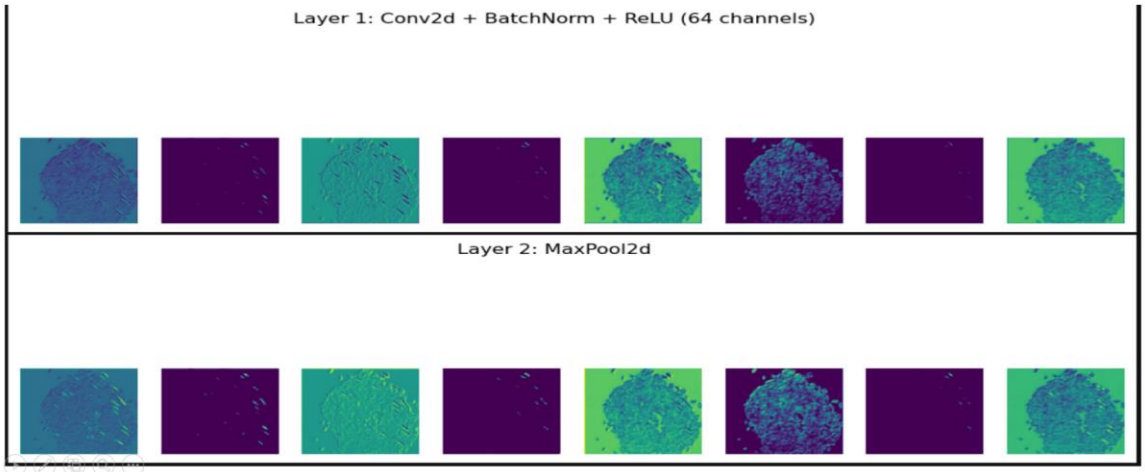


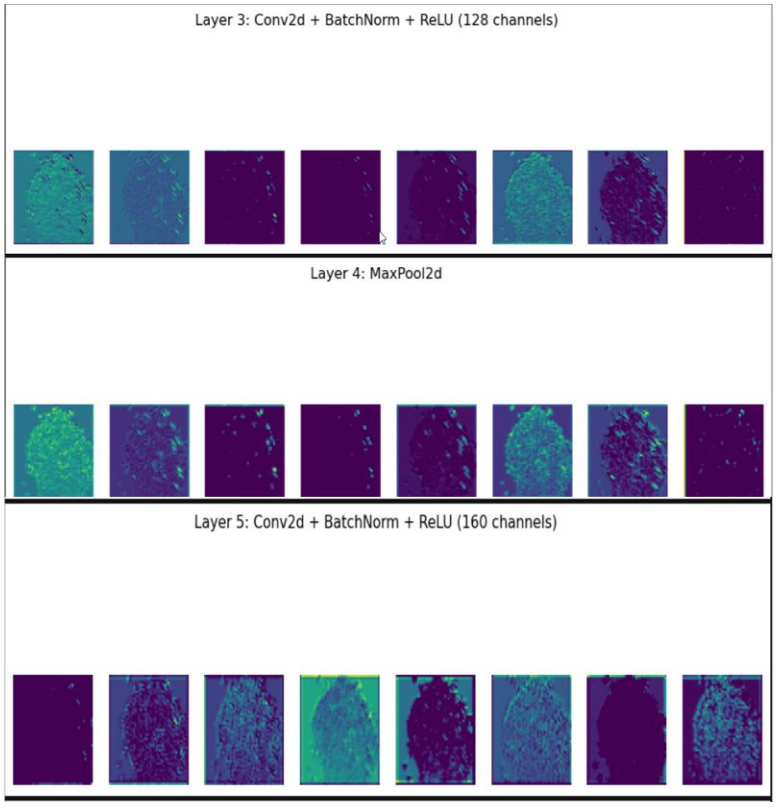
Figure 8 Flow Diagram of Self-Attention Mechanism Module

Intermediate outputs:









24

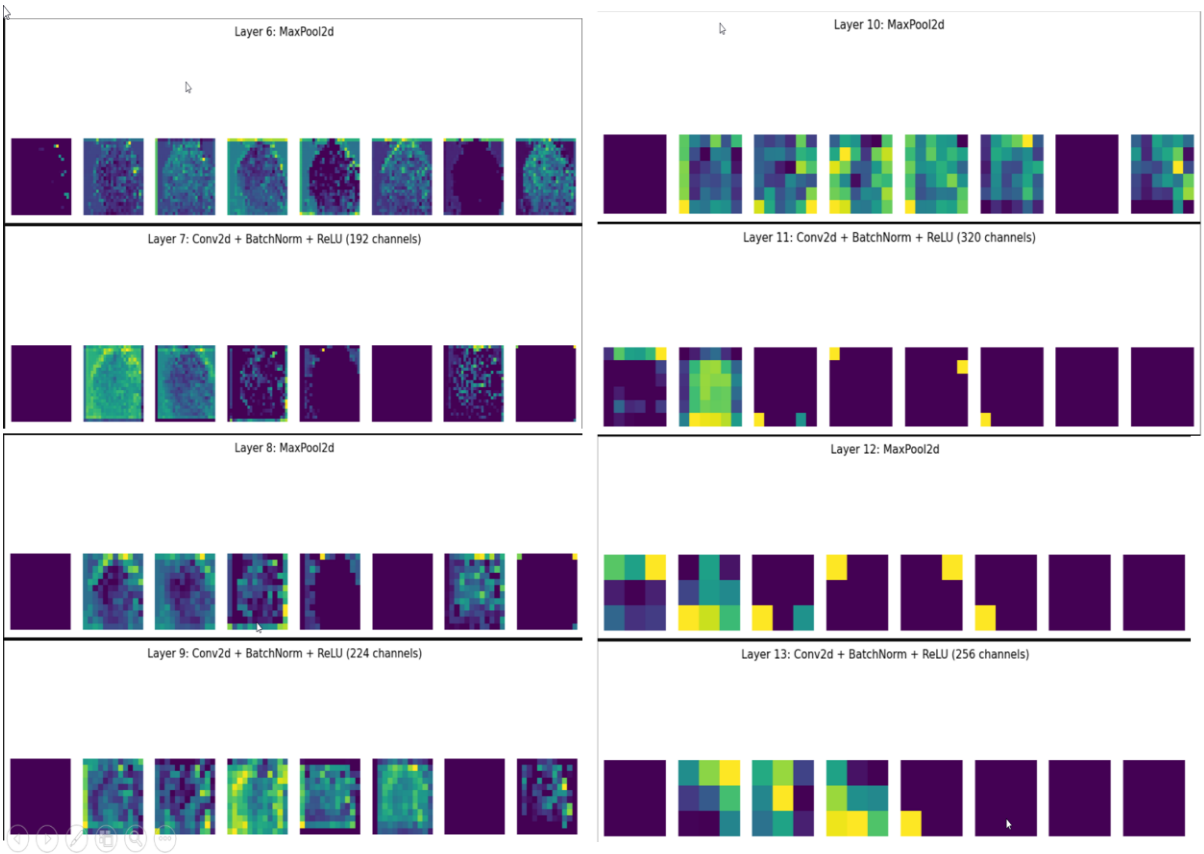


Figure 9 Output after each layer

3.2.8 P-LEAFNET:

P-LeafNet is a deep learning architecture designed for efficient and accurate image classification, particularly in plant leaf identification tasks. It is a compact model that leverages the efficiency of bottleneck layers and convolutional blocks to reduce computational complexity while maintaining high accuracy.

The architecture begins with an input image of size 96x96x3 (RGB), followed by a series of Convolutional (Conv2d) and Bottleneck blocks. The initial convolutional block captures low-level features from the image. Bottleneck layers, characterized by expansion ratios, depth multipliers, and multiple layers, progressively extract more abstract and detailed features. These bottleneck layers reduce the number of

parameters while preserving important information, making the network both efficient and effective.

Algorithm P-LeafNet:

Input: image (96x96x3)

Conv2dBlock(image, kernel_size=3x3, stride=2, padding=1) -> output1 (112x112x32)

BottleneckBlock(output1, expansion_ratio=1, depth_multiplier=1, num_layers=1, stride=1) -> output2 (112x112x16)

BottleneckBlock(output2, expansion_ratio=6, depth_multiplier=1, num_layers=2, stride=2) -> output3 (56x56x24)

BottleneckBlock(output3, expansion_ratio=6, depth_multiplier=1, num_layers=3, stride=1) -> output4 (28x28x32)

BottleneckBlock(output4, expansion_ratio=6, depth_multiplier=1, num_layers=4, stride=2) -> output5 (14x14x64)

BottleneckBlock(output5, expansion_ratio=6, depth_multiplier=1, num_layers=3, stride=1) -> output6 (14x14x96)

BottleneckBlock(output6, expansion_ratio=6, depth_multiplier=1, num_layers=3, stride=2) -> output7 (7x7x160)

BottleneckBlock(output7, expansion_ratio=6, depth_multiplier=1, num_layers=1, stride=1) -> output8 (7x7x320)

Conv2dBlock(output8, kernel_size=1x1, stride=1, padding=0) -> output9 (7x7x1280)

AveragePooling(output9, kernel_size=7x7, stride=1) -> output10 (1x1x1280)

Conv2dBlock(output10, kernel_size=1x1, stride=1, padding=0) -> output11 (1x1x1280)

FullyConnected(output11, num_neurons=1000) -> output12 (1000)

Softmax(output12) -> output13 (1000)

Return output13

End Algorithm

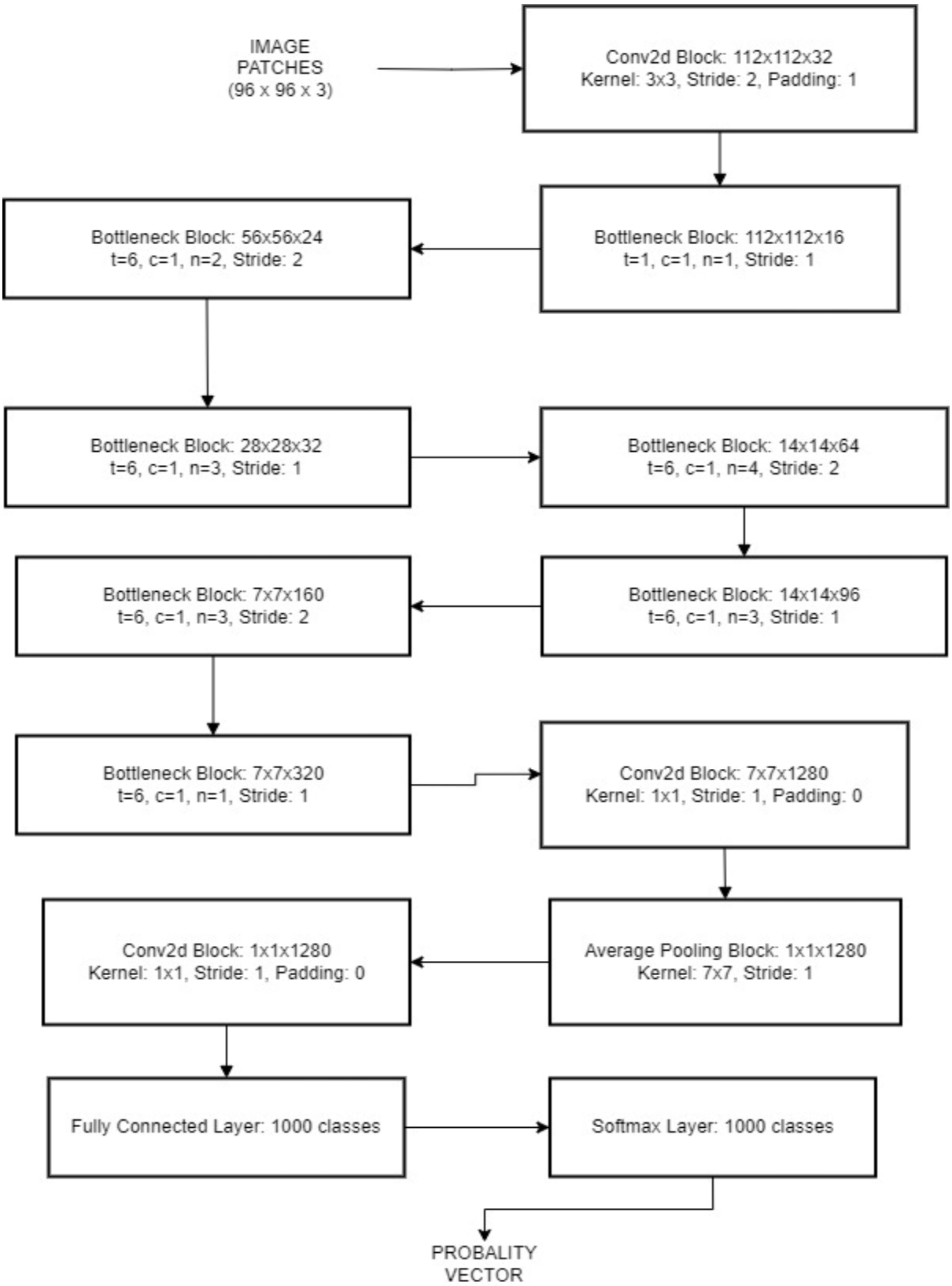


Figure 10 Flow diagram of P-leaf Net

3.2.9 Stacked Generalization:

Input: P1, P2, P3 (probability vectors)

Output: Predicted Class (final prediction after aggregation)

The Stacked Generalization Module is a meta-learning technique that combines the predictions from multiple base models—specifically, S-LeafNet, W-LeafNet, and P-LeafNet—to enhance overall predictive performance. This module operates by first generating predictions from each base model using the same input data. These predictions are then aggregated into a single feature set, which serves as the input for a meta model.

The meta model is trained on the combined predictions, enabling it to learn how to optimally weigh and integrate the outputs from the base models. Finally, the meta model produces a final prediction that ideally improves accuracy and robustness compared to individual models. This approach capitalizes on the diversity of the base models, reducing the likelihood of overfitting and increasing the generalization capability of the ensemble.

Algorithm StackedGeneralization(input_data)

Input: input_data (data for model predictions)

Output: final_output (final prediction after aggregation)

// Step 1: Get predictions from each base model

predictions_P1 = SLeafNet.predict(input_data)

predictions_P2 = WLeafNet.predict(input_data)

predictions_P3 = PLeafNet.predict(input_data)

// Step 2: Combine predictions into a single feature set

combined_predictions = combine(predictions_P1, predictions_P2, predictions_P3)

// Step 3: Train the meta model on the combined predictions

meta_model = train_meta_model(combined_predictions)

// Step 4: Get final output from the meta model

final_output = meta_model.predict(combined_predictions)

return final_output

End Algorithm

DETAILED BLOCK DIAGRAM:

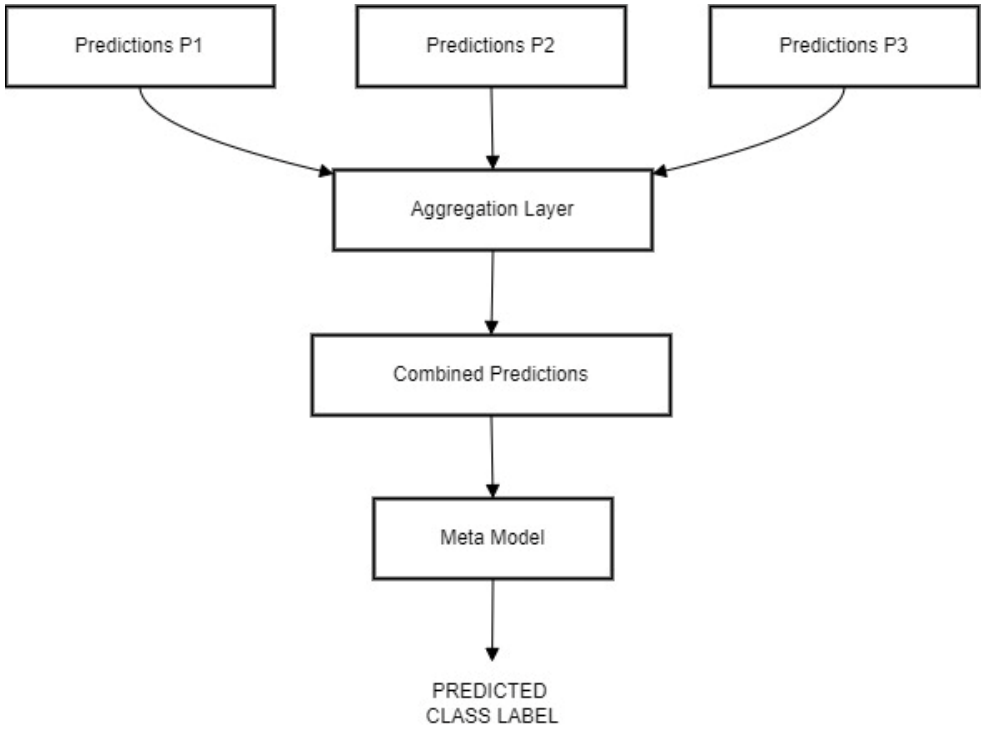


Figure 11 Flow Diagram Stacked Generalization

3.3 DATASET DESCRIPTION:

The Indian Spices Image Dataset consists of 10,991 high-quality images of 19 distinct Indian herbs. It supports research in machine learning, image recognition, and culinary studies. The dataset includes spices like black pepper, cumin seeds, dry ginger, Bay leaf etc.

List of Images:

1. Asafoetida
2. Cinnamon stick
3. Dry Ginger
4. Nutmeg
5. Bay Leaf
6. Cloves
7. Dry red Chilli
8. Poppy Seeds
9. Black Cardamom
10. Coriander Seeds
11. Fennel seeds
12. Star Anise
13. Black Pepper
14. Cubeb Pepper
15. Green Cardamom
16. Stone Flowers
17. Caraway seeds
18. Cumin seeds
19. Mace

Image Quality and Resolution: Each image was taken under controlled conditions with a white background to ensure consistency and minimize background noise. However, the images differ in resolution, reflecting the real-world conditions under which they were captured. This variability can potentially enhance the model's ability to generalize across different image qualities.

Lighting Conditions: The images were captured in a single lighting environment, with minor variations in lighting conditions that are noticeable across the images. These slight variations are intentional to simulate real-world scenarios where lighting may not be perfectly uniform. This aspect of the dataset aims to enhance the robustness of the model by training it to recognize items under different lighting conditions.

Class Labels: Each image in the dataset is associated with a specific class label corresponding to the item it represents. These labels are crucial for the supervised learning approach employed in this study, where the model learns to map input images to their respective classes.

CHAPTER 4

IMPLEMENTATION DETAILS

The implementation started with first desiging the UI of the mobile application,Then the initial preprocessing steps are done to the images present in the dataset to prepare the dataset to make it ready for training.

4.1 HARDWARE SPECIFICATIONS:

1. **Model Training**
    - **Hardware**:
        - **GPU**: NVIDIA T4 with 16GB VRAM
        - **Processor**: Intel Xeon with 4 cores
    - **Frameworks and Tools**:
        - Python (version 3.9.19)
        - PyTorch for model development and training
        - Image augmentation techniques applied during preprocessing
2. **Model Deployment**
    - **Hosting Platform**:
        - Google Cloud Platform (GCP)
    - **Deployment Tools**:
        - FastAPI/Flask for API development
        - GCP for hosting the trained model and handling inference requests
3. **Development Environment**
    - **Local Setup**:
        - Arch-based Linux system for code development and testing
        - Access to GPU-enabled college resources for training

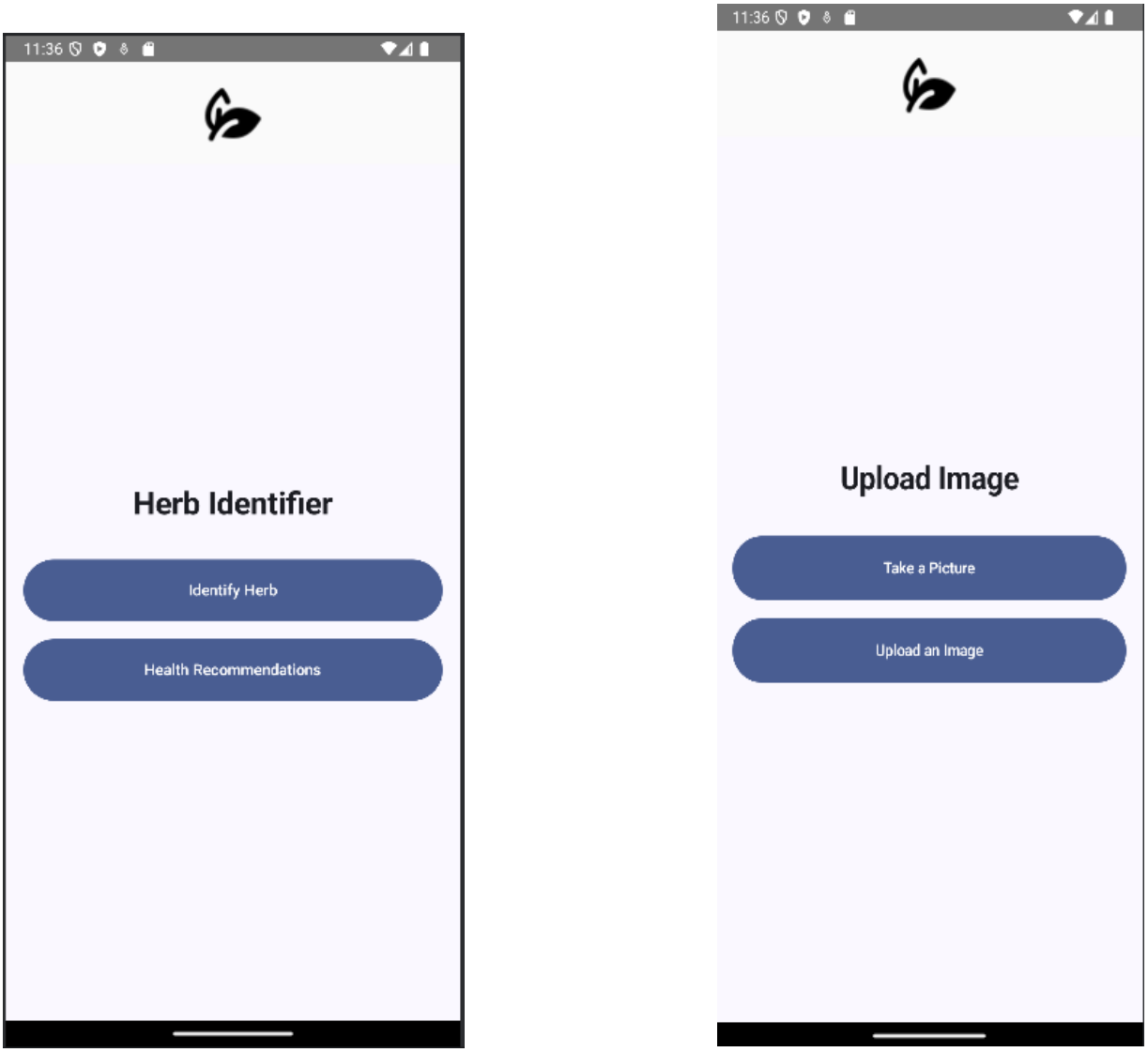


Figure 12 : UI of the mobile application

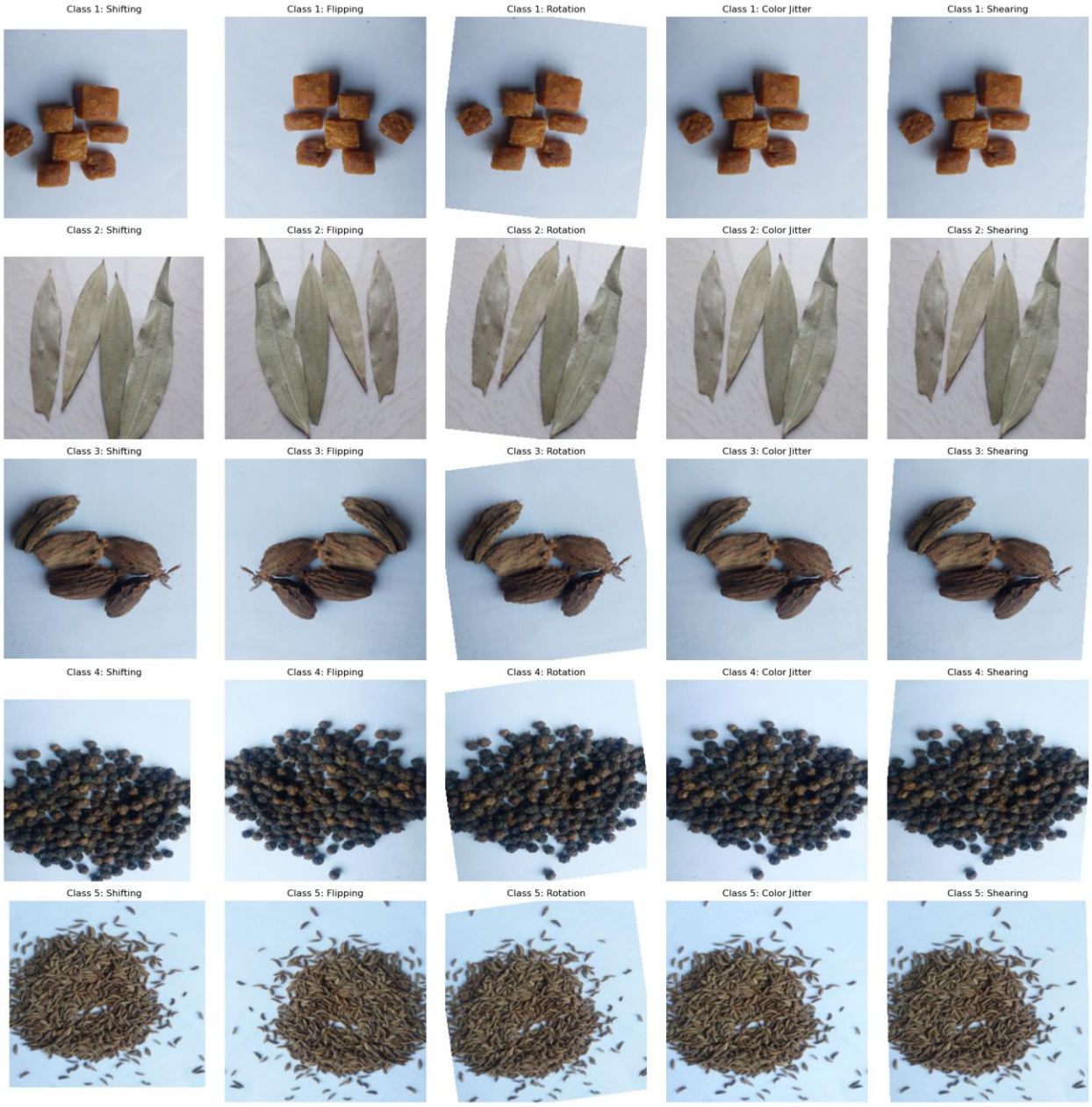


Figure 13 Images After Augumentaiton

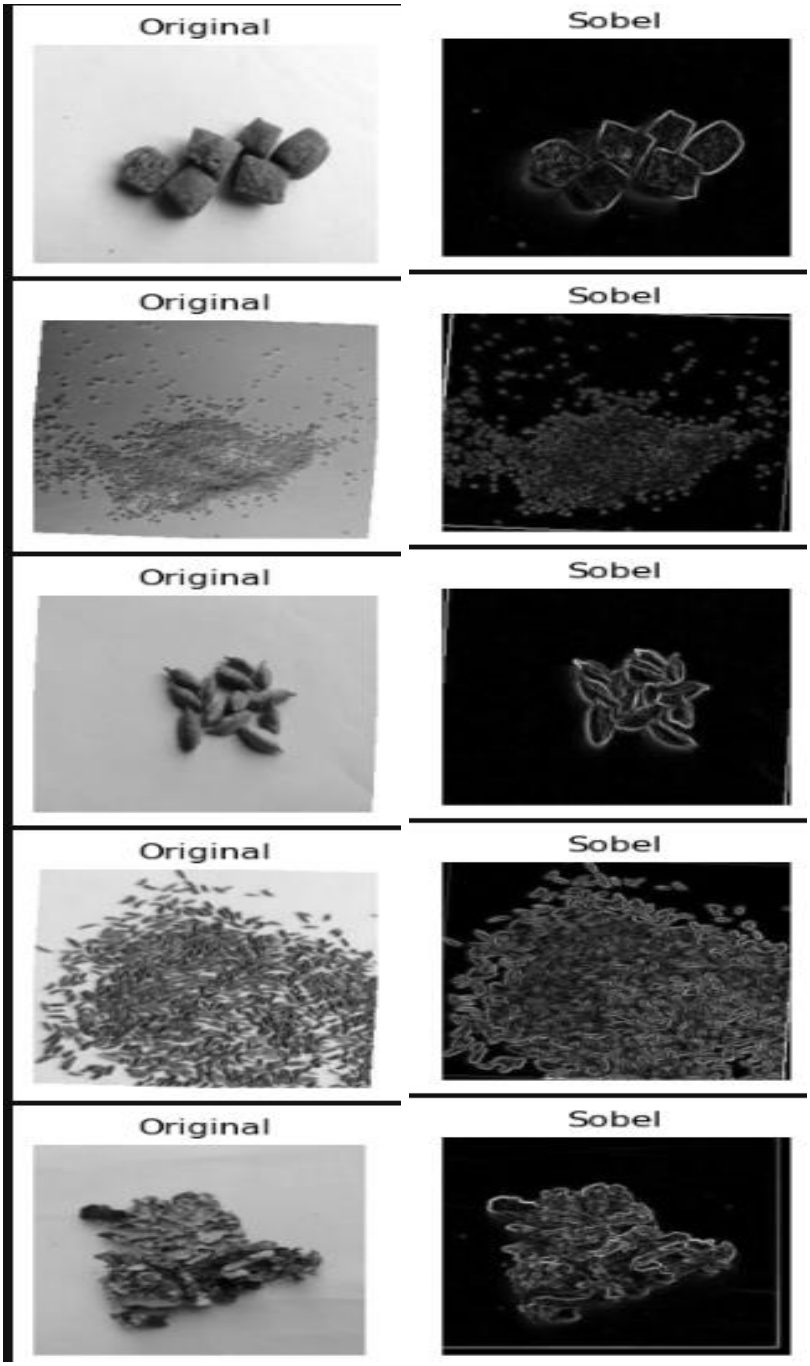


Figure 14 Results after sobel edge detection

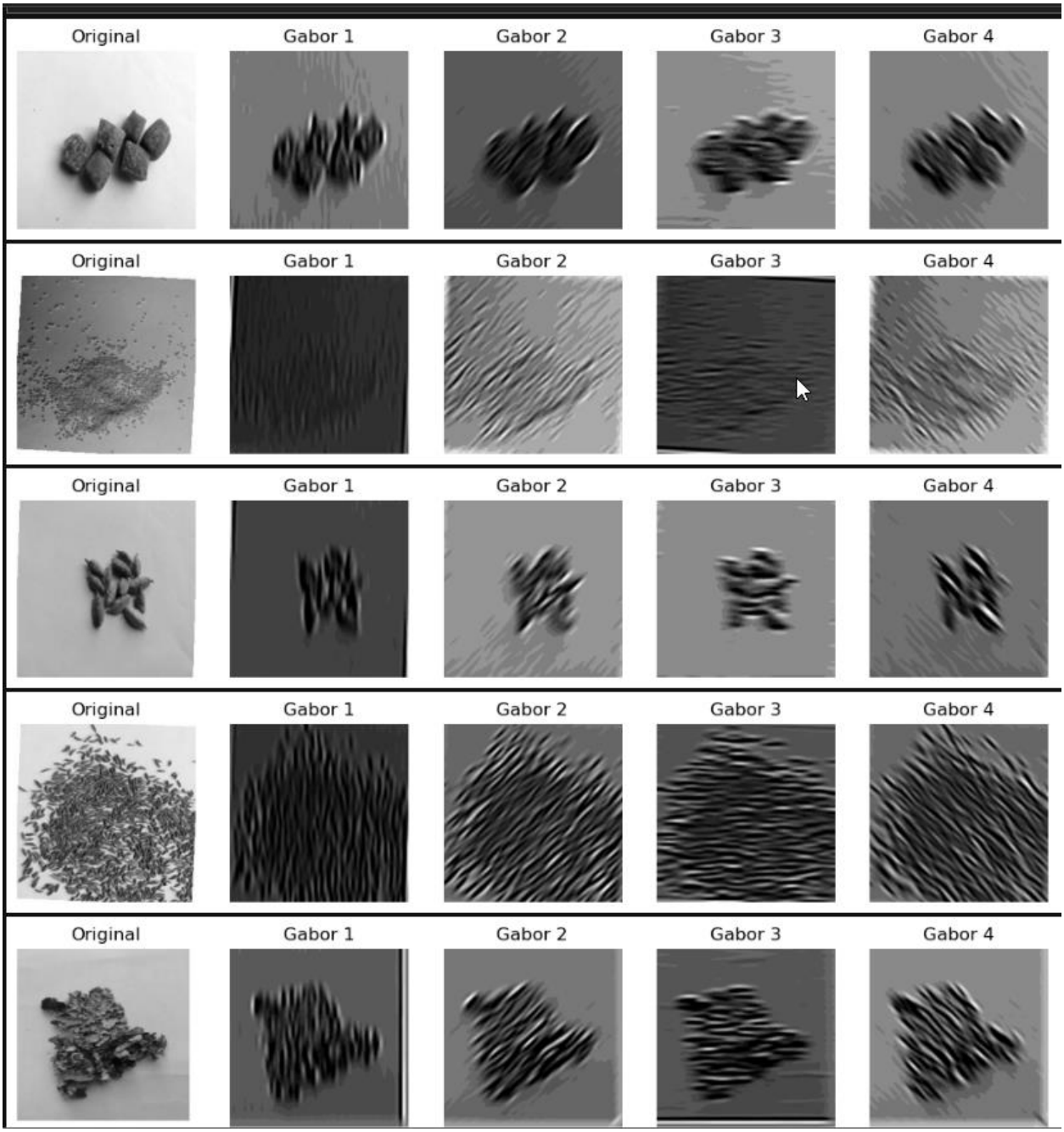


Figure 15 Gabor filter output

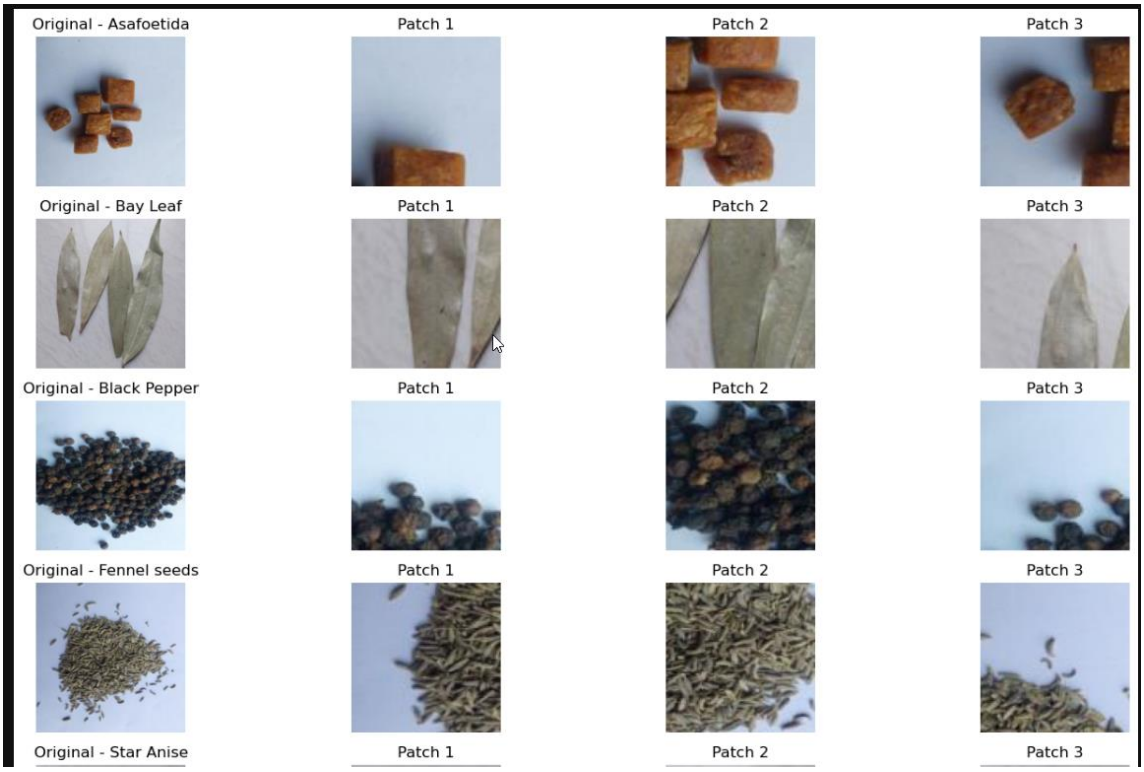


Figure 16 Self Attention Ouput

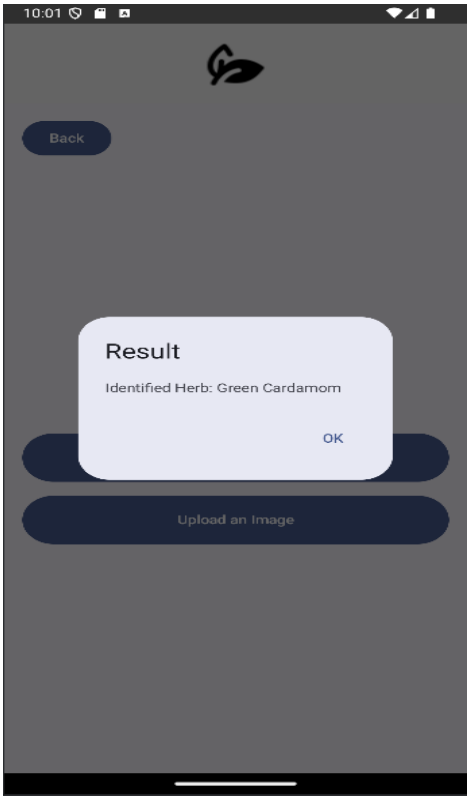


Figure 17 Final Prediction of Herb displayed on the application

CHAPTER 6

RESULTS AND ANALYSIS

The following results where obsereved when the models where tested on the performance metrics.

1. **Accuracy:**
   Measures the proportion of correct predictions out of all instances.
   Accuracy = (TP + TN) / (TP + TN + FP + FN)
2. **Precision:**
   Indicates the reliability of positive predictions by measuring the ratio of true positives to predicted positives.
   Precision = TP / (TP + FP)
3. **Recall:**
   Assesses the model's ability to identify all relevant instances by calculating the ratio of true positives to
   actual positives.
   Recall = TP / (TP + FN)
4. **F1 Score:**
   Provides a balance between precision and recall, useful for imbalanced class distributions.
   *F1 Score = 2 * (Precision * Recall) / (Precision + Recall)*
5. **Error Rate:**
   Quantifies the proportion of incorrect predictions.
   Error Rate = (FP + FN) / (TP + TN + FP + FN)
6. **Execution Time:**
   Measures the time taken for the model to process and classify images.
   Execution Time = End Time - Start Time
7. ***Top-K Accuracy:***
   Measures the percentage of samples where true label is among the model’s top k predicted labels.

Top-k accuracy = no. of correct labels in the top k pred. / total no of samples.

6.1 W-LeafNet

The results and graph highlight the outstanding performance of the W-LeafNet model in classifying a diverse range of spices. The model achieved an impressive test accuracy of 99.09%, showcasing its ability to correctly identify the majority of spice samples. Additionally, the precision, recall, and F1-score values were consistently around 99.1%, confirming the model's accuracy in predictions and its effectiveness in capturing true positive samples. With a low error rate of 0.91%, the W-LeafNet model proves to be a highly dependable tool for spice classification.

The class-wise breakdown in the graph demonstrates that the model performs exceptionally well for most spices, achieving near-perfect scores. While minor performance variations were noted for certain classes, these are likely due to inherent challenges in differentiating specific spices or limitations in the training data. Overall, the W-LeafNet model exhibits remarkable accuracy and reliability, making it a valuable asset for applications such as quality control, food safety, and spice authentication.

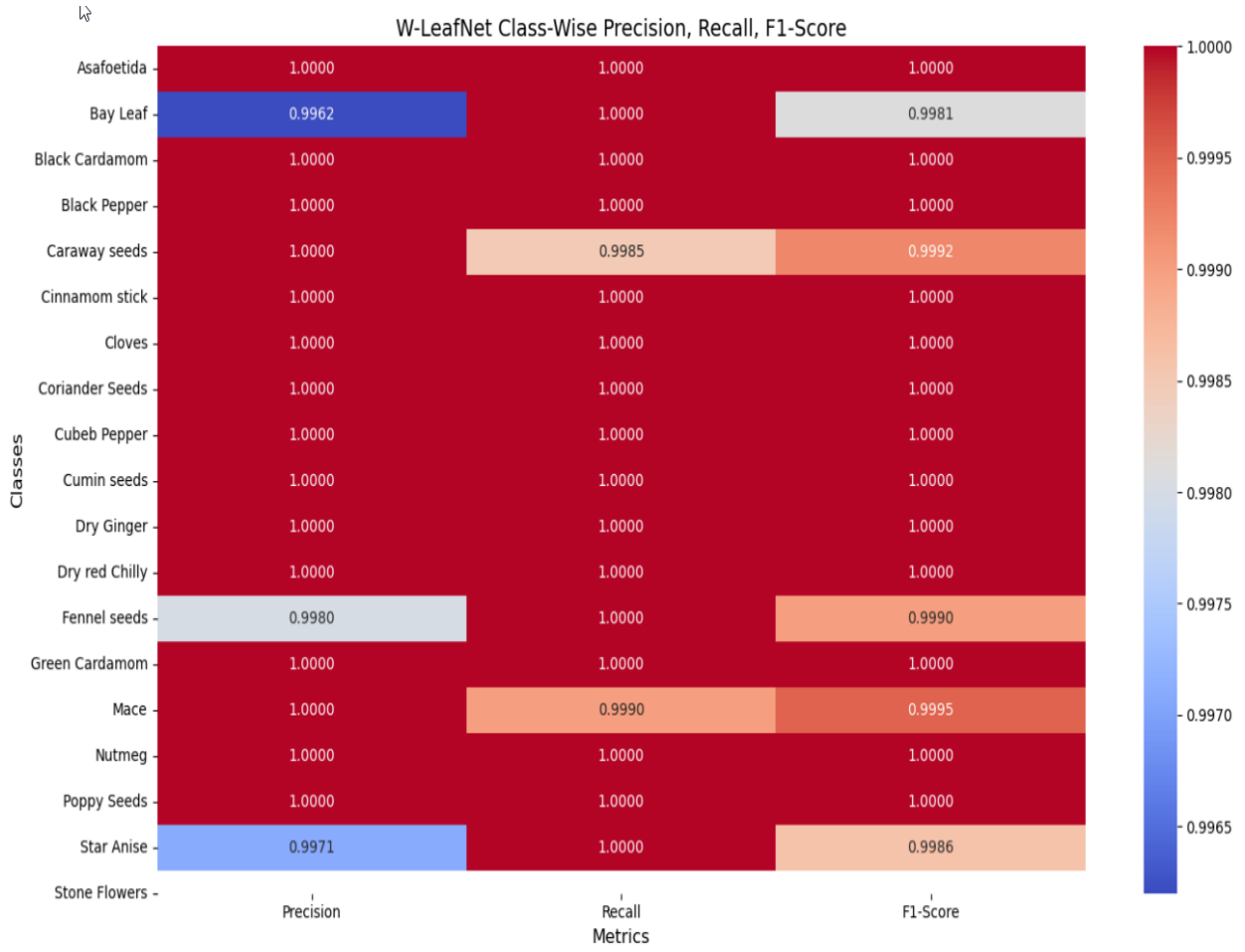


Figure 18 Class-wise metrics for W-LeafNet

6.2 S-LeafNet

The model achieved a test accuracy of 94.53%, demonstrating its ability to correctly classify the majority of dry herb samples. With precision, recall, and F1-score values hovering around 95%, the model shows a strong balance between accuracy and its capacity to identify true positive samples effectively. The error rate of 5.47% indicates a reasonable level of reliability in its classifications.

The graph provides a class-wise breakdown, revealing that most dry herb classes exhibit high precision, recall, and F1-scores, confirming the model's effectiveness in distinguishing individual herbs. Minor variations in performance across certain

classes are likely due to challenges in differentiating similar herbs or limitations in the training data.

Overall, the model demonstrates solid performance in dry herb classification. Although its accuracy is slightly lower than that of the W-LeafNet model, it still serves as a reliable tool for dry herb identification. Further analysis and optimization could enhance its performance, making it even more effective for practical applications.

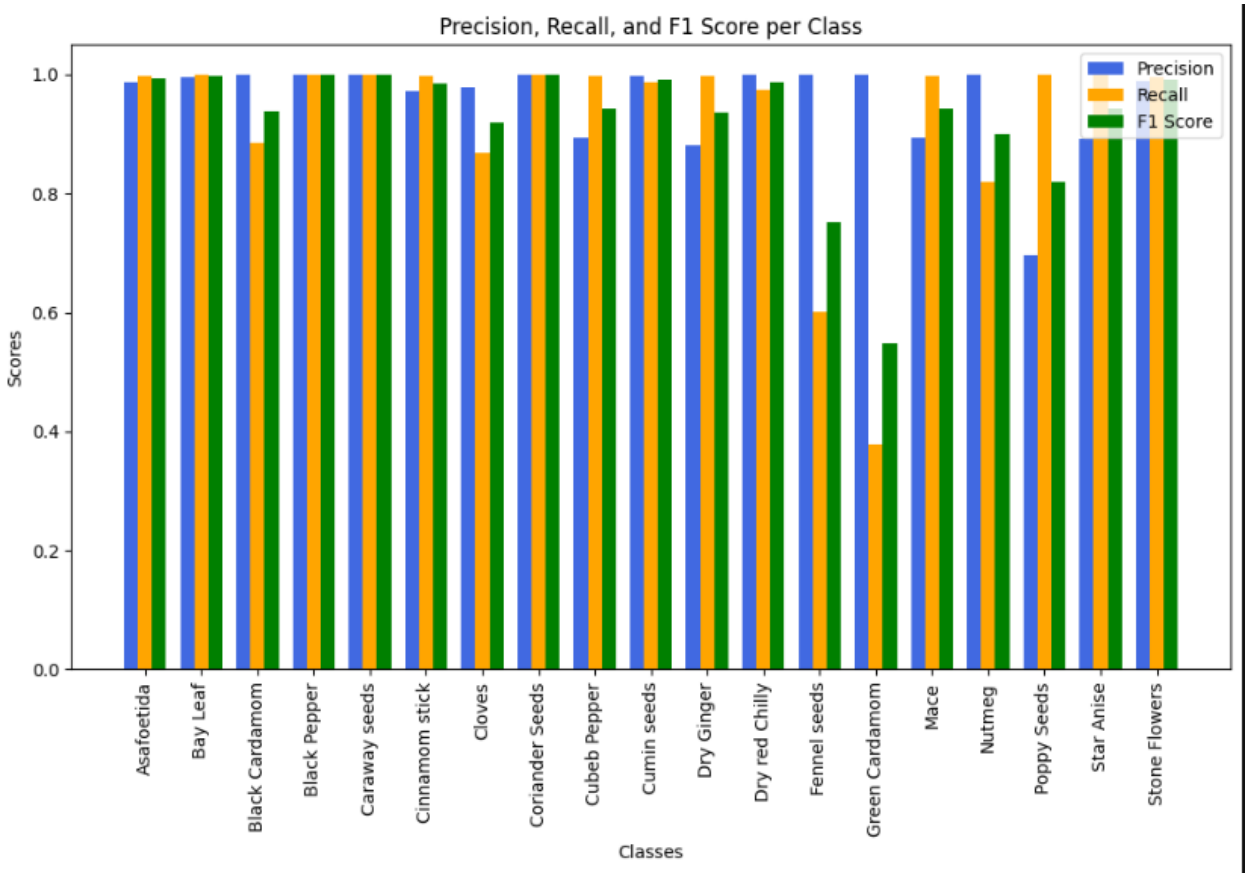


Figure 19 CLass wise metrics for S-leafnet

6.3 P-LeafNet

The model achieved an overall test accuracy of 36.30%, indicating moderate success in classifying dry herb samples. While the precision, recall, and F1-score

values vary significantly across different classes, certain herbs achieve relatively higher scores, reflecting the model's ability to distinguish them effectively. However, the low scores for many other classes suggest room for improvement in the model's ability to generalize across all herb categories.

The class-wise breakdown provided in the graph reveals notable inconsistencies, with some herb classes achieving better precision and recall, while others exhibit low performance. These variations may stem from challenges in differentiating visually similar herbs, imbalanced class distributions, or limitations in the training data.

Overall, the model demonstrates a basic level of capability in dry herb classification but requires further optimization to improve its accuracy and reliability. Strategies such as data augmentation, improved model architecture, or fine-tuning hyperparameters could significantly enhance its performance.

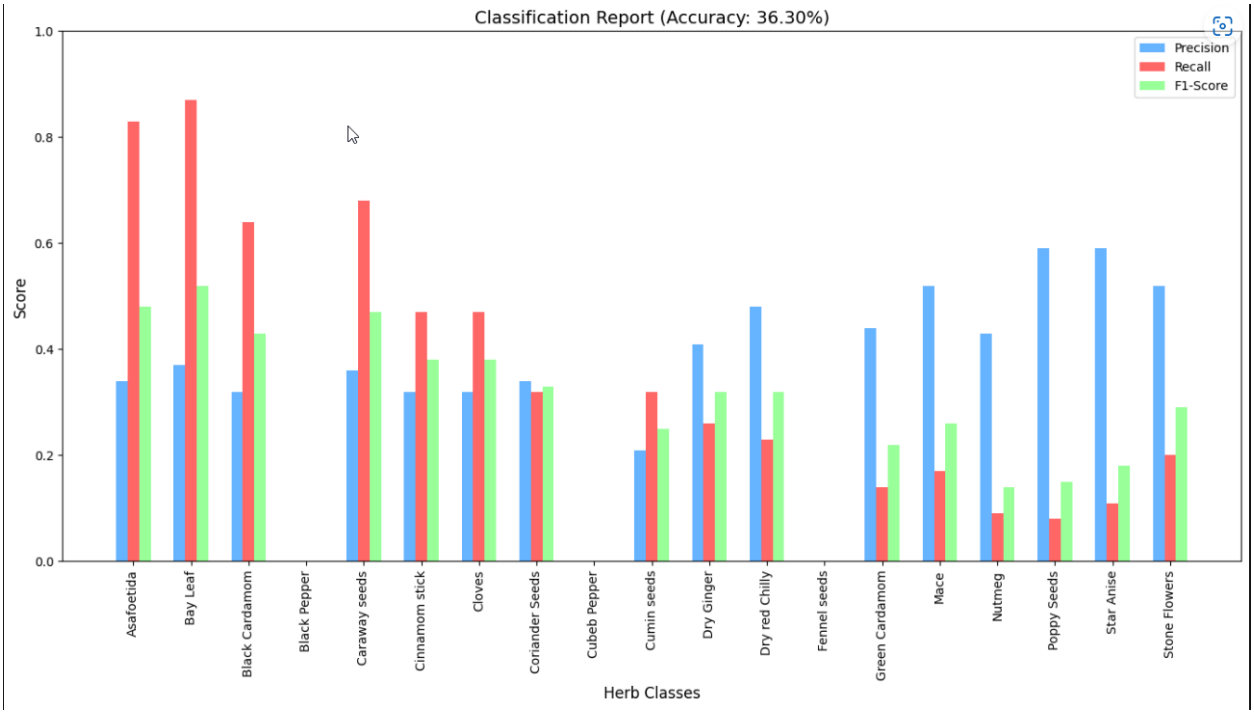


Figure 20 ClassWise metrics for P-LeafNet

Overall, the model demonstrates a basic level of capability in dry herb classification but requires further optimization to improve its accuracy and reliability. Strategies such as data augmentation, improved model architecture, or fine-tuning hyperparameters could significantly enhance its performance.

CHAPTER 7

CONCLUSION AND FUTURE WORK

The conclusion of this project highlights its significance in leveraging mobile usage data to understand behavioral patterns using a robust machine learning model. By implementing a Random Forest Classifier, the project successfully analyzes complex user data to provide accurate and insightful predictions. The seamless integr

ation of DevOps practices, including version control with Git, automated testing and deployment with Jenkins, and scalable infrastructure on AWS EC2, ensures a highly efficient and collaborative workflow. This synergy between machine learning and DevOps enhances the overall development process, reduces time-to-market, and ensures a reliable and maintainable system. The project serves as a powerful tool for behavior analysis, contributing to fields like digital well-being, personalized recommendations, and targeted interventions.

The future scope of this project lies in expanding its functionality and scalability. Enhancements can include incorporating real-time data streams for dynamic behavior analysis, integrating deep learning techniques to uncover complex patterns, and improving feature engineering for greater prediction accuracy. Additionally, extending the deployment to multi-cloud environments can provide better resilience and performance. Implementing advanced monitoring tools can further ensure system reliability and preempt potential failures. Moreover, the system can be tailored to support diverse datasets, extending its applicability to various domains such as education, healthcare, and marketing.

By continually adapting to emerging technologies and expanding its use cases, the project has the potential to become a versatile and indispensable tool for behavioral insights and digital transformation initiatives.

CHAPTER 8

REFERENCES

[1] Beikmohammadi, A., Faez, K., & Motallebi, A. (2022). SWP-LeafNET: A novel multistage approach for plant leaf identification based on deep CNN. Expert Systems with Applications, 202,117470.

[2] Diwedi, H.K., Misra, A. & Tiwari, A.K. CNN-based medicinal plant identification and classification using optimized SVM. *Multimed Tools Appl* 83, 33823–33853 (2024). https://doi.org/10.1007/s11042-023-16733-8

[3] Geetharamani, G., & Arun Pandian, J. (2019). Identification of plant leaf diseases using a nine-layer deep convolutional neural network. *Computers & Electrical Engineering, 76,* 323-338.

[4] Zhao, Z.-Q., Ma, L.-H., Cheung, Y.-M., Wu, X., Tang, Y., & Chen, C. L. P. (2015). ApLeaf: An efficient android-based plant leaf identification system. *Neurocomputing*, *151*(Part 3), 1112-1119. DOI: 10.1016/j.neucom.2014.02.077

[5] Kan, H.X., Jin, L. & Zhou, F.L. Classification of medicinal plant leaf image based on multi-feature extraction. *Pattern Recognit. Image Anal.* 27, 581–587 (2017).

[6] J. w. Tan, S. -W. Chang, S. Abdul-Kareem, H. J. Yap and K. -T. Yong, "Deep Learning for Plant Species Classification Using Leaf Vein Morphometric," in IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 17, no. 1, pp. 82-90, 1 Jan.-Feb. 2020, doi: 10.1109/TCBB.2018.2848653

[7] Quach, B.M., Dinh, V.C., Pham, N. et al. Leaf recognition using convolutional neural networks-based features. Multimed Tools Appl 82, 777–801 (2023).

[8] Kanda, Paul & Xia, Kewen & Sanusi, Olanrewaju. (2021). A Deep Learning-Based Recognition Technique for Plant Leaf Classification. IEEE Access. PP. 1-1. 10.1109/ACCESS.2021.3131726.
[9] T. N. Quoc and V. T. Hoang, "Traditional Vietnamese Herbal Medicine Image Recognition by CNN," 2023 15th International Conference on Knowledge and Smart Technology (KST), Phuket, Thailand, 2023, pp. 1-5, doi: 10.1109/KST57286.2023.10086725.
[10] Bodhwani, V., Acharjya, D. P., & Bodhwani, U. (2019). Deep Residual Networks for Plant Identification. Procedia Computer Science, 152, 186-194. DOI: 10.1016/j.procs.2019.05.042
[11] Pushpa BR, Lakshmi P (2022) Deep learning model for plant species classifcation using leaf vein features. In: 2022 international conference on augmented intelligence and sustainable systems (ICAISS). IEEE, pp 238–243
[12] Hu J, Chen Z, Yang M, Zhang R, Cui Y (2018) A multiscale fusion convolutional neural network for plant leaf recognition. IEEE Signal Process Lett 25(6):853–857
[13] Máthé Á, Khan IA (2022) Introduction to medicinal and aromatic plants in India. In: Medicinal and aromatic plants of India, vol 1. Springer International Publishing, Cham, pp 1–34
[14] Naeem S, Ali A, Chesneau C, Tahir MH, Jamal F, Sherwani RAK, Ul Hassan M (2021) The classifcation of medicinal plant leaves based on multispectral and texture feature using machine learning approach. Agronomy 11(2):263
[15] Geetharamani G, Pandian A (2019) Identifcation of plant leaf diseases using a nine-layer deep convolutional neural network. Comput Electr Eng 76:323–3