Research Paper

Medicinal Plant Part Identification and Classification using Deep Learning based on Multi Label Categories

Misganaw Aguate[1,*], Abebe Tesfahun[2], Amlakie Aschale[1]

[1]*Department of Electrical and Computer Engineering, Debre Tabor University, P. O. Box 272, Debre Tabor, Ethiopia*

[2]*School of Electrical and Computer Engineering, Debre Markos University, P. O. Box 269, Debre Markos, Ethiopia*

Article Info	Abstract
	Plants have been used as direct medicinal sources since ancient times as well as today. However, researchers and pharmacists are facing difficulties to identify medicinal plant parts before starting ingredient extraction in the laboratory. This study was conducted to identify the medicinal plant part based on multi-label categories by employing a sigmoid classifier as the last layer of Convolutional Neural Network (CNN). The study employed supervised learning approach in which the true values were predefined initially for the classifier using data annotation phase. Hence, leaf images of the plants were taken as an identity for the rest of the plant parts. The system was designed based on transfer learning by adopting (fine tune) the pre-trained models that employ CNN and trained using Image Net. High-resolution cameras for data acquisition and google Colab for the experiment (training and testing) were used. Mobile Net performed best with an accuracy of 93% for training sets and 92% for testing sets. When the models were evaluated using F1_score, it performed 94%. Without batch normalization at fully connected layer, this model scored 84%. So, Mobile Net obtained higher performance, and suitable to classify the medicinal plant body part. It was also taken as the fastest model to train because Mobile Net used depth wise separable convolution method that reduces scalar multiplication through convolution. By observing the results obtained from the presence and absence of batch normalization, this study deduced that batch normalization is advantageous to obtain good classification performances of the models.

1. Introduction

Extensive research on medicinal plants identification has been done by various researchers (Dileep et al., 2019, Bandara, et al. 2019, Tan et al., 2020; etc.). However, the studies did not answer the questions like which part of the plants is used as a medicine. Accordingly, this study tried to identify the specific parts of medicinal plants. The identification of the medicinal plant parts can be done using image processing and chemical ingredient extraction. But using its chemical ingredient consumes time and needs high expenditure for laboratory equipment. So, to reduce this problem image processing is the preferred approach. This approach can be handled with the help of deep learning. It is obvious that, deep learning can be used in Speech Recognition, Natural Language Processing, Machine Translations, Audio Recognitions, bioinformatics, Drug design, Medical Image Identifications, and Medicinal plant identification and classification (Amuthalingeswaran et al., 2019). That excited us to apply image processing with deep learning. This study used a multi-label classification technique to

---

improve indigenous knowledge using transfer learning. Mobile Net, VGG16, and Inception_V3 are among the adopted deep learning techniques. The ability of plants to cure disease using its parts and the need of improvement of indigenous knowledge to modern medical science are the main points for the excitation to precede this study.

The digitization of useful species of plants and their information is necessary. Several researchers have tried to develop a more robust and efficient plant recognition system by exploiting pattern recognition and image processing techniques based on plant leaves, flowers, barks, and fruits. However, leaves play a very important role among all the other parts of a plant as they contains rich information as well as more reliability (Naresh et al., 2016). Hence, this study used the leaves of medicinal plants to extract unique features. Proving of overfitting is not only affected by the depth of convolutional layers and the number of datasets, addition of batch normalization at the fully connected layer of CNN to makes the model to learn fast and obtain good classification performances and taking of the leaf images from its back side to makes the CNN to extract feature of the leaves uniquely and accurately are the major contribution of this study.

In 2019, the extraction of shape, color and texture features from leaf images was employed and to train the Artificial Neural Network (ANN) to identify the exact leaf classes (R.janani, 2013). The plant species can be identified based on the input leaf sample (Sivaranjani et al., 2019). The medicinal plant data set was developed based on the extraction of texture and color features from plant leaf images (Pacifico et al., 2019). The author used machine learning for recognition of medicinal plants. But using deep learning is good to classify medicinal plant due to its ability to extract features automatically. Herbal species can also be identified using their flower images (Bandara et al., 2019). The authors have used SVM, Decision Trees and K-NN. Using only leaf features of shape, color, texture is not a distinct attribute of leaves. In Lee et al. (2017), the authors proved that vein and contour features are good to uniquely identify plant species. Raw plant leaf image is represented into deep features using knowledge transfer from object identification to plant species identification (Prasad et al., 2017). VGG-16 ConvNet architecture is used to train and classify with

combination of PCA which can reduce feature vector to optimize classification cost (Prasad et al., 2017). In (Tan et al. 2020) the leaf images were preprocessed and the features were extracted by pre-trained AlexNet, fine-tuned AlexNet and D-Leaf. These features were then classified by Support Vector Machine (SVM), Artificial Neural Network (ANN), k-Nearest Neighbour (KNN), NaïveBayes (NB) and CNN. AyurLeaf which is a Deep Learning based CNN model proposed in (Dileep et al., 2019) to classify medicinal plants.

2. Methods and Materials

This study used CNN which is the backbone of deep learning algorithm and can automatically extract multiple unique features of plant leaves. The study also employed transfer learning technique by adding batch normalization at the fully connected layer of CNN, and tune weights and training hyper parameters. In this study, we took the backside of leaves to make the model extract the features uniquely and identify medicinal plant parts accurately.

2.1. Proposed System Architecture

In this research six procedures were followed such as data collection, data annotation (labelling), image preprocessing, feature extraction and training, testing (model evaluation) and finally classification of medicinal plant parts. Figure 1 shows the high-level system architecture of the study. The input image phase includes data acquisition and annotation (data labelling). Model Building phase includes feature extraction and training tasks. The detail description is mentioned in the following consecutive sections.

2.2. Data Acquisition

The data was collected using a high-resolution camera (TECHNO SPARK K7 with 13MP and SAMSUNG A30S with 25MP). These sample data weree taken from Gojjam, Jimma, and Kefa.

These data collection procedure took two months. When taking images of the plant leaves, to avoid data augmentation, the different light intensity strength and direction of leaf positions were considered. Backsides of the leaves were captured to extract vein feature accurately. We mde the focal length different depending on the broadness of leaves to make the prediction consistent. The dataset contains 15,100 medicinal plant leaf images. Each plant has an equal number of leaf images (300 images) but the number of plants categorized in each
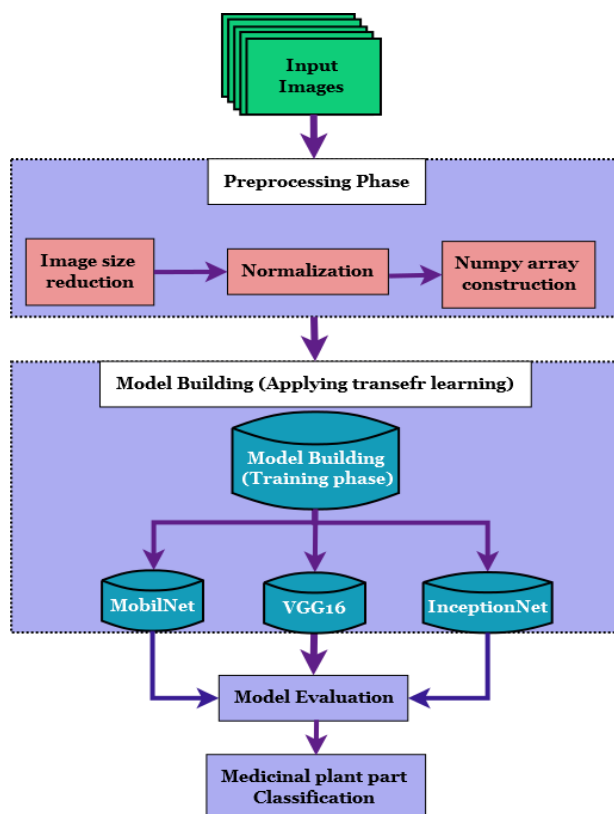
Figure 1: High level system architecture

label is different. Hence, the study has imbalanced data in each category. The following guidelines are taken for this data acquisition:

- Medium aged and young parts of the plant leaf are selected;

- The focal length of the camera (distance between leaves and camera) is taken depending on the broadness and narrowness of the plant leaves;
- The white background of the image was selected to avoid confusion and get a clear leaf image structure;
- Dried and diseased leaves excluded to make the prediction consistent, and
- The leaf images are captured immediately after cut out from the plants to reduce the loss of leaf features.
- The camera stands perpendicular to the leaf stand (i.e. the position taken 90° from the leaf stand).

2.3. Visualizing Intensity Variation of the Data

The frequency of pixels in RGB channels, brightness and, darkness of the images can be visualized by simply observing the vertical, left, and the right parts of the image histogram. Hence, Figure 2 shows images of one leaf but in different pixel intensity sample and Figure 3 and 4 (histogram plot) describe how it varies based on its three-channel pixel intensity and distribution throughout the different images.

Figure 3 shows the histogram of sample one (S1) image. It is different from sample three (S3). The leaf S1 and S3 are the same but this variation is due to light intensity while the images are captured. This variation of the datasets in a single leaf makes the model to classify consistently while it faces different images captured by different cameras and environments.

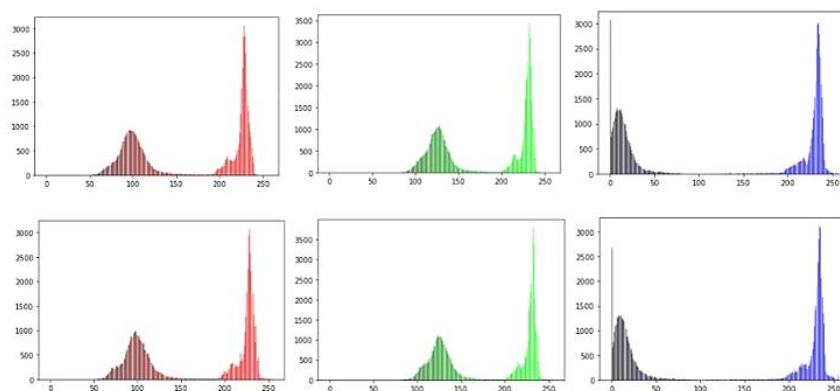**Figure 2**: Intensity variation of single leaf image

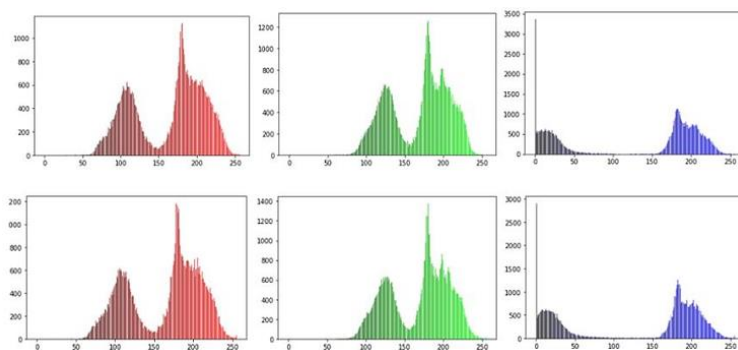**Figure 3**: Histogram of leaf image (S1)

98

Figure 4: Histogram of leaf image (S3)

2.4. Data Annotation

The labeling tasks were supported by five experts with 5, 10, 13, 28- and 30-year experiences who are currently working on traditional medicine preparation. Initially, 72 plants were selected. But finally, 52 were taken as the complete dataset. Twenty plants were avoided because there are not commonly labeled parts by the experts for those plants. Since the system is based on supervised learning approach, all parts of the plats were tagged (mapped) on the leaves image as indicated in Table 1, Img_1, img_2…. img_n indicates the name of the leaf images file.

Table 1: Sample of data annotation (labeling)

File name	Tags	root	bark	leaf	Seed	Un identified
img_1	['root']	1	0	0	0	0
img_2	['root','leaf','seed']	1	0	1	1	0
img_3	['bark','leaf']	0	1	1	0	0
img_4	['seed']	0	0	0	1	0

2.5. Pre-processing

To reduce computational time during training, the input image was resized to 128 x 128. In the system architecture, normalization means scaling of image pixels. Image pixels are integer values from 1 up to 255. In CNN, processing a large integer value can disrupt or slow down the learning process. Hence, the image pixels were scaled (normalized) between 0 and 1. A series of array values of images pixel saved as one variable and later reloaded to save processing time when using this data for the second time. After selecting the target label, it was changed into series of array elements and finally binarize the labels into 0's and 1's using the label binarize function to be understood by Sigmoid activation function (Zhao et al., 2014). Hence, the system used a binary classifier internally for each 5 target labels. Even though the color feature consumes computational time to process the image, RGB image used to avoid loss of some useful features.

2.6. Model Building (Transfer learning)

2.6.1. Feature Extraction

The fine-tuned Models of CNN such as Mobile Net, VGG16, and Inception_V3 are adopted to extract useful leave features and feed them to the classifier. These feature extractions are processed in the feature learning layer part of CNN as indicated in Figure 5. This is achieved with the help of filtering kernel that slides over the image pixels and compute dot product to produce different image features, Max pooling to reduce the spatial size of the convolved image features, and ReLU activation function to allow the models learn faster and better by overcoming the problem of vanishing gradient problem. In this feature extraction, the deep learning model extracts the necessary features of the plant leaf automatically.
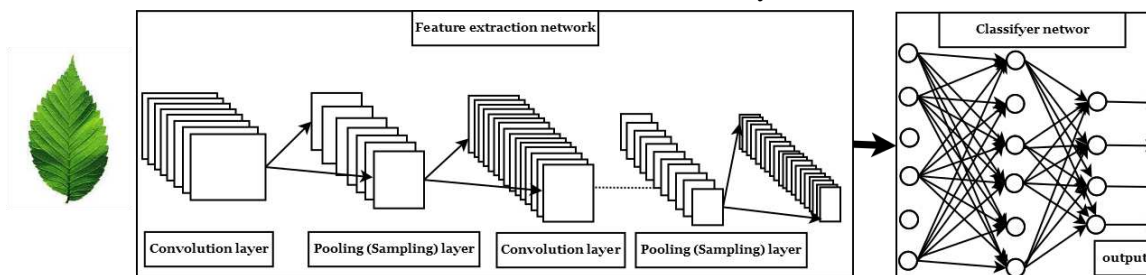
Figure 5: Structure of convolutional neural network

2.6.2. How CNN Internally Extract Leaf Features

Figure 6 shows how CNN extract leave features. At 1st, it detects simply identifiable patterns like horizontal and vertical lines present in leaves. At the 2nd layer, it can detect different corners on leaves as a result it tries to identify the shape of leaves. At the 3rd layer, it further computes more complex feature map identification like extracting differently structured veins on the surface of leaves. At the 4th layer, it becomes more power full to exploit each tiny vein structure and while going to a very deep layer it can identify the ubiquitous structure. So, by going through this execution CNN can identify leaves unique features for classification. In this study, feature map means the number of image features produced by convolution process. As the image size increases the number of features that are produced by convolution layer will be increases. If there is no padding, the convolution layer reduces the image sizes
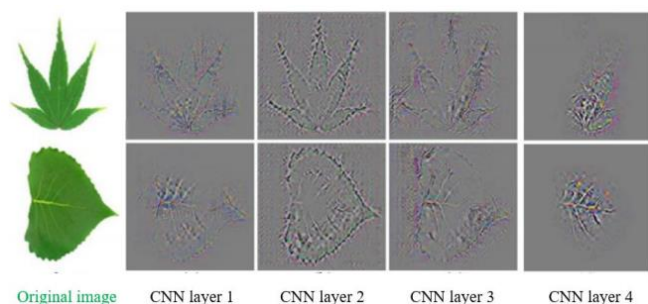

Figure 6: Feature extraction process in convolutional neural network

2.6.3. Classification

This part contains flatten, fully connected neural network, dropout, and sigmoid layer. Loss function of binary cross entropy is used because at the end layer, there is a sigmoid classifier shown in Equation 1. Sigmoid is used as classifier because this study performs five binary decisions (classification) internally to achieve multi label classification.

$$f(x) = \frac{1}{1+\exp^{-x}} \qquad (1)$$

Figure 7 shows the basic technical operations performed. In the figure, F_1, F_2, F_3, ... F_n represent features, D_1, D_2, D_3, ... D_n represents input image data which is feature vector as 2D array and C_1, C_2, C_3, ... C_n represents target classes.
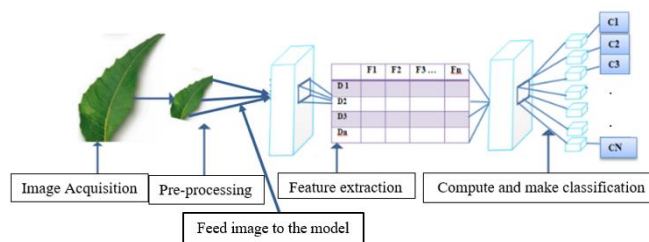

Figure 7: Basic technical operation performed in this study

2.7. Train the Model

To save computational resources and time, the transfer learning techniques are adopted on Mobile Net, VGG16, and Inceprion_V3. The adopted models are pre-trained with the image net. In this procedure some layers of the models are tuned with the new weights and the remaining layers was left to freeze. At the fully connected layer, the new dense layers drops out and batch normalization are added. The advantages obtained from this transfer learning are, the training is completed in a short period of time, no need of large number of training epochs, and also the data needed for training are small. So, to customize these pre-trained models in addition to layer tuning, the deep learning hyper parameters are tuned up on the models. On Mobile Net the last 4 layers of convolutional base trained with the new weights and the remaining layer was left to freeze. At the fully connected layer, two layers with a 1024 unit and ReLU activation functions are used. Due to this modification total number of layers in Mobile Net changed from 28 to 30. Another interesting thing is batch normalization added at the dense layers to facilitate training rates and obtain good performance with small epochs. On the VGG16, the same changes with Mobile Net are applied. Unlike Mobile Net, a dropout layer with a value of 0.5 added to overcome the problem of overfitting and only one dense layer is added. On the Inception_V3 also applied the same as Mobile Net and VGG16 except all base convolutional layers frozen and dropout of 0.9 is used at FC. The common hyper-parameters applied on all models are, batch size that is the number of training sample utilize in one iteration (epoch), epochs used as the number of passes to determine how many repetitions a machine learn from input data, learning rate to determine the step size in each iteration, dropout to make some neuron inactive temporarily to overcome the problems of over

fitting, and optimizers to update weight parameters and minimize error rate during back propagation.

2.8. Model Performance Evaluation

To evaluate the models Accuracy, Precision, Recall and F-Score, Equation 2, 3, 4 and 5 can be used (Bekkar et al., 2013). But in this study, we have imbalanced dataset among categories. Hence, F_1_Score is better than the other to measure for the performance of models (Jeni et al., 2013) as it is the harmonic mean of recall and precision.

$$Accuracy = \frac{TP+TN}{TP+TN+FP\ FN} \tag{2}$$

$$Precision = \frac{TP}{TP+FP} \tag{3}$$

$$Recall = \frac{TP}{TP+FN} \tag{4}$$

$$F1\_Score = 2 * \frac{Precision*Recall}{Precision+Recall} \tag{5}$$

where, TP, TN, FP and FN represent true positive, true negative, false positive and false negative respectively.

2.9. Multi Label Setup

The parts of the plant are taken as multi label for corresponding plant dataset to classify it as multi label classification. The target class labelled as L= [l_1, l_2, l_3, l_n] where L is the list of labels which can be taken as a target class for a given input dataset. l_n is an element of (0, 1). For a specific input row from element of target label array, l_n=0 means, the input leaf image has no the target label (the label is plant part) and l_n = 1 means, input leaf image have that target label.

2.10. Working Principle of the System

Figure 8 contains two main parts such as image data processing and target label processing. On the image processing side, after putting image data in a folder and reading the CSV file, these two files are concatenated to read a full image file. From a folder, the original image of leaves taken and from the CSV file, the corresponding filename of the image taken is labeled in the first column of the CSV file. The input image is resized to 128 x 128 and the pixel values are normalized between 0 and 1 to reduce computation time during training. The image pixels are changed into a series of a 2D array to be understandable by computer machines (models) while processing it. The array values of image pixel with the corresponding mapped (labeled) classes are saved as one variable and later reload to save computational time when these data are needed later. Two of the columns are dropped from the CSV file which holds image name and label tags, and leave the remaining column as the target label of corresponding image files. Figure 2-5 indicates the image data and corresponding target labels is split into training and testing set as X-train, X-test, and Y-train, Y-test to evaluate the models. Here 20% of the image data and target label are taken for the test set for one-time cross validation purpose. Training set is used to train the model
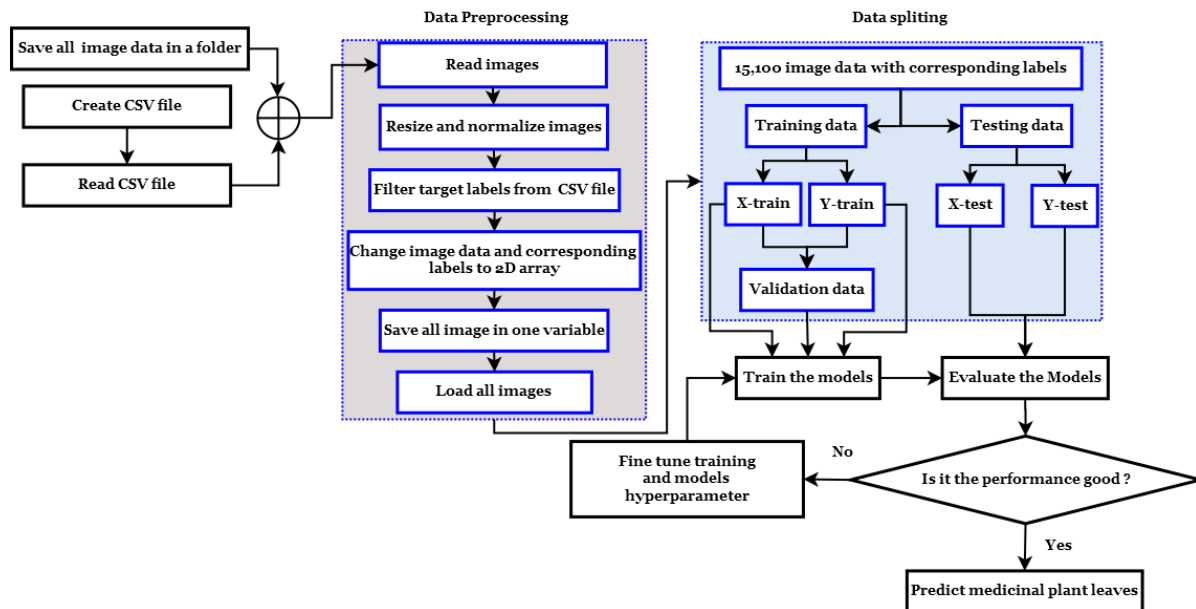

Figure 8: Over all working principle of the system

and the testing set is unseen data during training but used later to test the model. Further 20% from X-train and Y-train data are taken the validation set to cross check whether the model train well or not within each epoch during training. The next tasks are fitting X-train, Y-train, and validation data to the model and train it. Here fine-tuned Mobile Net, VGG16, and Inception-V3 are used on the train model section. Then we test the efficiency of the models using X-test Y-test. During this procedure the validity of the model are evaluated and then if the models are performed good, predictions of medicinal plant parts are takes place otherwise tuning of training and model hyper parameters are applied and train the model again until good result is obtained.

2.11. Experimental Setup

For the experimental analysis, first all image data with corresponding target labels are uploaded to google drive and the data further mounted to Google Colab. Then Google Colab with free 12 GB RAM and Tesla K80 GPRU0.0 is used for data preprocessing, model building (training), testing, evaluation and predication (classification). The data is split to 80% for training and 20% for test. 15,100 total medicinal plant images are prepared. From this data first 20% are taken for testing which are 3,020 images and the remaining 12,080 data taken for training. From 12,080 data, again 20% (2,416 images) is taken for validation. So, the data are split in to 9,664 for training, 3,020 for testing (unseen data during training), and 2,416 for validation. In all models, weights of image net and custom weights are used and the dense layer removed and replaced with new convolutional layers including batch normalization and dropout layer. Batch normalization to increase the speed of learning and acquire higher accuracy values. For the trained models in each experiment, the best optimizer is selected and mentioned in the table with the corresponding batch sizes. Different deep learning hyper parameters are tuned to come up with the best solution (Table 2).

3. Results and Discussion

3.1. Mobile Net

As indicated in Table 3, Adam gave a good result based on training and testing losses. Even though RMSprop score higher result in training and testing accuracy, it did not reduce the losses.

Figure 9 shows the best case of training and validation accuracy/loss of fine-tuned Mobile Net model using Adam and RMSprop optimizer, with batch size of 32 and 128. This Model gives a higher result with this specified batch size and optimizers.

Table 2: Hyper parameter tuning

Hyper parameter		MobileNet	VGG16	InceptionNet
Batch sizes	32	Applied	Applied	Applied
	64	>>	>>	>>
	128	>>	>>	>>
Learning rates	$1e^{-4}$	>>	>>	>>
	$1e^{-3}$	>>	>>	>>
Optimizers	Adam	>>	>>	>>
	Adamax	>>	>>	>>
	Adagrad	>>	>>	>>
	RMSprop	>>	>>	>>
Dropout		…………	……….	0.5	0.9
Weight modification		>>	>>	Not used

Table 3: Training and testing accuracy/loss of Mobile net

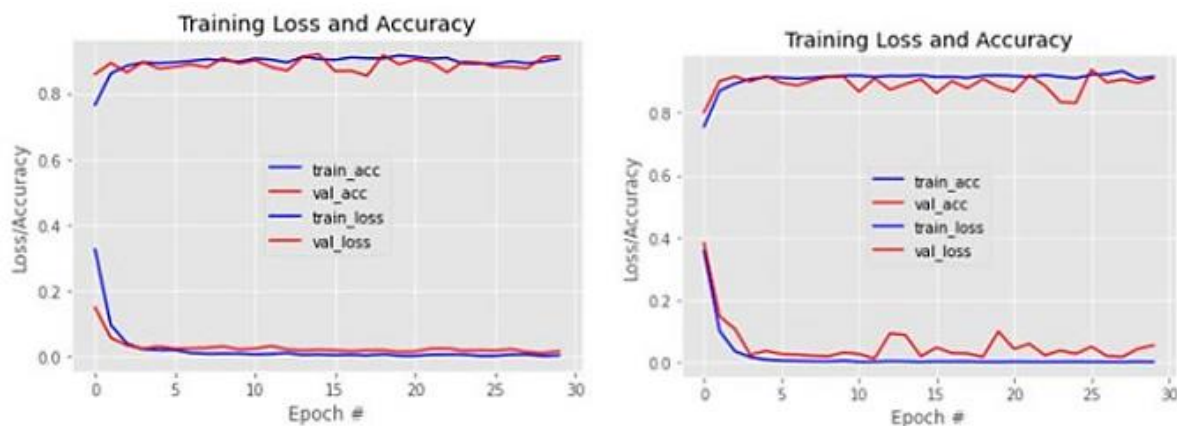| Batch size	Optimizers	Training accuracy	Testing accuracy	Training loss	Testing loss
32	Adam	92.59	92.15	0.34	1.29
	RMSprop	93.33	91.82	0.87	3.03
64	Adam	91.83	91.69	0.36	1.39
128	Adam	92.32	91.75	0.28	1.22
	RMSprop	92.63	92.12	1.63	3.99

Figure 9: Training and testing accuracy/loss using optimizer Adam and RMSprop respectively

3.2. VGG16

Training and testing accuracy/loss of VGG16 is included in Table 4. Using batch size 32 and Adamax gives preferable result but RMSprop using batch size 64 and Adam using batch size 128 gives valid accuracy. When compared among loss values Adam using batch size 128 gives an acceptable result. So, Adam is taken as a good optimizer for VGG16.

3.3. Inception_V3

As seen from Table 5, inception_V3 cannot give a satisfactory result compared to Mobile Net and VGG16. But when compared by itself RMSprop gives good results on the batch size of 32, 64, and 128.

Generally, when we look in to the result analysis, Mobile Net is a good and fastest classifier than VGG 16 and Inception Net. The reason behind is that Mobile Net constructed by depth wise separable convolution layers

that makes the model to learns fast and reduce the problem of overfitting. The advantage of using depth wise separable convolutional layer is to reduce the value of total scalar multiplications produced by convolution process.

3.4. Accuracy of Models with Batch Normalization

In this part, the study tried to show the empirical result of training accuracy, testing accuracy, training loss, and testing loss (Figure 10 and 11) with the presence of batch normalization at fully connected layer of CNN. Since our data is imbalanced, in this study accuracy is not used to measure the performance of the models. Because accuracy places more weight on the large classes than on small classes, which makes it difficult for a classifier to perform well on the rare classes, it become a misleading indicator (Bekkar, et al. 2013).

Table 4: Training and testing accuracy/loss of VGG16

Batch size	Optimizers	Training accuracy	Testing accuracy	Training loss	Testing loss
32	Adamax	88.86	86.95	0.39	1.69
64	RMSprop	91.71	90.83	0.64	1.6
128	Adam	91.78	90.93	0.4	1.33

Table 5: Training and testing accuracy/loss of Inception net

Batch size	Optimizers	Training accuracy	Testing accuracy	Training loss	Testing loss
32	RMSprop	74.30	70.93	7.67	11.23
64	RMSprop	74.77	71.32	6.58	10.27
128	RMSprop	75.2	72.25	6.31	9.83

Training accuracy with LR = 0.0001 and BS = 32

	Adam	Adamax	Adagrad	RMSprop
MobileNet	92.6	92.1	86.1	93.3
VGG16	85.5	88.9	82.0	84.1
InceptionNet_V3	71.4	68.9	62.6	74.3

Testing accuracy with LR = 0.0001 and BS = 32

	Adam	Adamax	Adagrad	RMSprop
MobileNet	92.2	90.8	81.7	91.8
VGG16	84.3	87.0	79.8	82.6
InceptionNet_V3	68.4	67.3	61.1	70.9
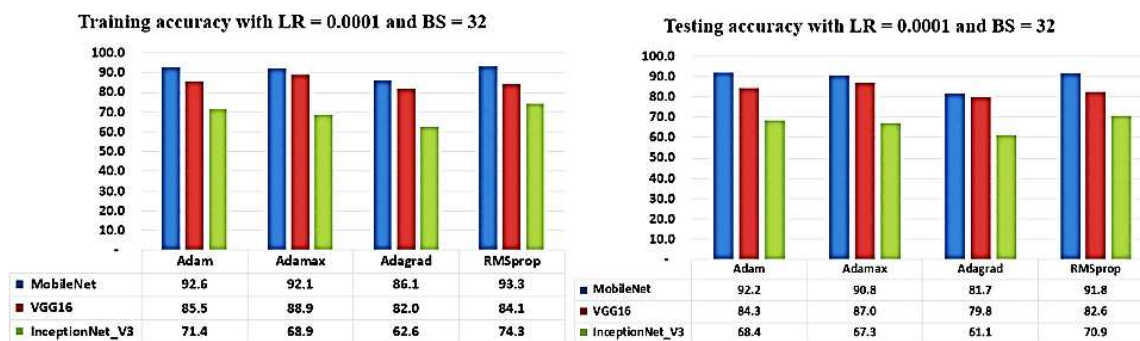
Figure 10: Training and testing accuracies of the models

Training loss with LR = 0.0001 and BS = 32

	Adam	Adamax	Adagrad	RMSprop
MobileNet	0.3	0.3	26.1	0.9
VGG16	0.4	0.4	28.6	1.1
InceptionNet_V3	10.6	14.4	52.4	0.7

Testing loss with LR = 0.0001 and BS = 32

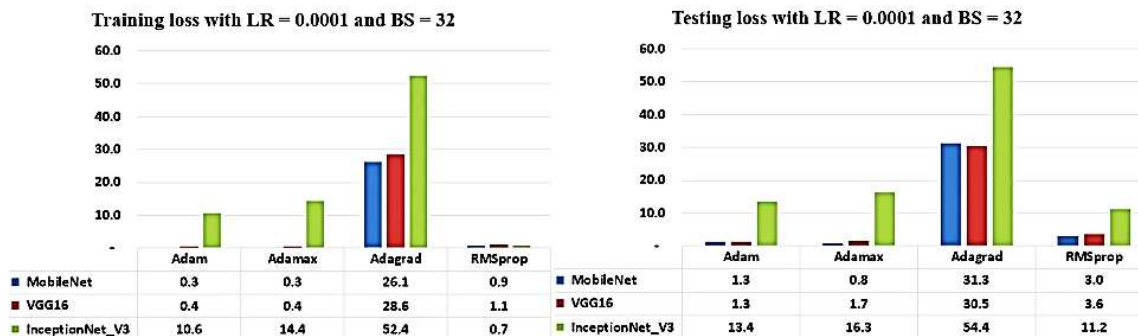|	Adam	Adamax	Adagrad	RMSprop
MobileNet	1.3	0.8	31.3	3.0
VGG16	1.3	1.7	30.5	3.6
InceptionNet_V3	13.4	16.3	54.4	11.2

Figure 11: Training and testing losses of the models

3.5. Accuracy of Models without Batch Normalization

Figure 12 and 13 shows an experimental result of all model accuracy and loss for both training and testing datasets respectively. This is taken as higher in case of training the models without batch normalization. But it is almost the same as the worst case of models with batch normalization.
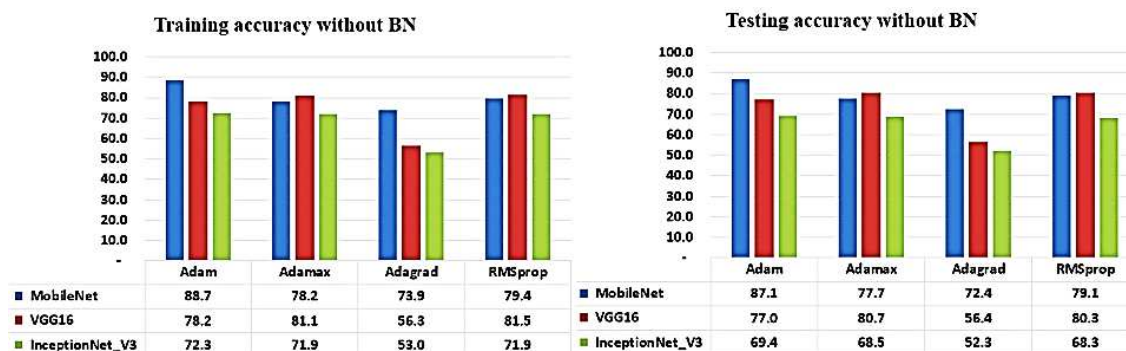
Training accuracy without BN

	Adam	Adamax	Adagrad	RMSprop
MobileNet	88.7	78.2	73.9	79.4
VGG16	78.2	81.1	56.3	81.5
InceptionNet_V3	72.3	71.9	53.0	71.9

Testing accuracy without BN

	Adam	Adamax	Adagrad	RMSprop
MobileNet	87.1	77.7	72.4	79.1
VGG16	77.0	80.7	56.4	80.3
InceptionNet_V3	69.4	68.5	52.3	68.3

Figure 12: Training and testing accuracies of the models without batch normalization

Training loss without BN

	Adam	Adamax	Adagrad	RMSprop
MobileNet	0.3	0.4	11.2	0.7
VGG16	0.5	0.5	28.2	2.9
InceptionNet_V3	6.1	10.2	39.0	7.4

Testing loss without BN

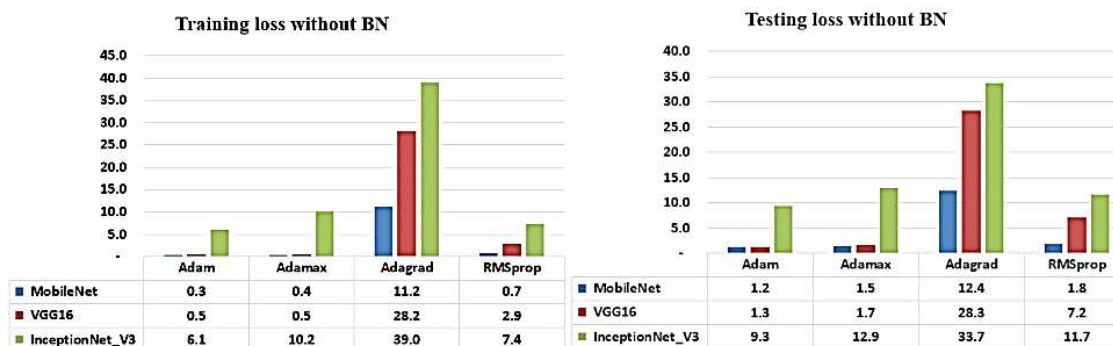|	Adam	Adamax	Adagrad	RMSprop
MobileNet	1.2	1.5	12.4	1.8
VGG16	1.3	1.7	28.3	7.2
InceptionNet_V3	9.3	12.9	33.7	11.7

Figure 13: Training and testing losses of the models without batch normalization

3.6. Performance Evaluation

To measure the performance of models, this study used the hyper parameters, the effect of batch normalization when it is added at fully connected layer of CNN and the type of convolutional layer such as depth wise separable and standard convolutional layer as evaluation benchmarks. The dataset in each category is not balanced (each category has different amount of data). Hence, when there are datasets that have unbalanced classes a better choice is F1_score, which can be interpreted as a weighted average of the precision and recall values (Jeni et al., 2013). As indicated in Figure 14, Mobile Net performs good using a batch size of 32 and optimizer of Adam and RMSprop. This indicates that Mobile Net is well trained on a new medicinal plant data than other models.

Micro_averaged for lr = 0.0001 and BS = 32

	Adam	Adamax	Adagrad	RMSprop
MobileNet	94.00	91.00	82.00	96.00
VGG16	86.00	88.00	80.00	87.00
Inception_V3	68.00	67.00	61.00	71.00

Figure 14: Model performance evaluation result of the models

MobileNet performance with and without BN

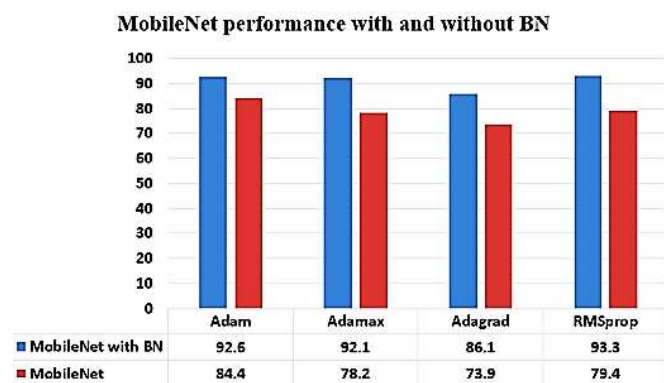|	Adam	Adamax	Adagrad	RMSprop
MobileNet with BN	92.6	92.1	86.1	93.3
MobileNet	84.4	78.2	73.9	79.4

Figure 15: Comparison of model (Mobile Net) accuracy with and without batch normalization

In Figure 15, the study tried to show performance comparison of with and without batch normalization by applying on Mobile Net. As indicated, for multi label classification Mobile Net without batch normalization decreases the accuracy by 8.2 % and 13.9 % using Adam and RMSprop respectively. This is a big difference and hence for image classification based on multi label technique application of batch normalization on the dense layers of convolutional neural network is highly recommended.

3.7. Classification Report

Figure 16 represents the confusion matrix on which the models are measured with the test data set by observing how many test samples of the data are classified correctly and how many are misclassified in each class (category). This classification report is taken as the best performance from the 72 experiments. The result is obtained using a learning rate of $1e^{-4}$, batch size of 32, and optimizer of Adam. The right side of the figure shows how many of the sample data are classified in True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN) in each of the five classes. The root of the plant is taken as an illustration for the confusion matrix. It has the true positive value of 732, false-positive value 4, true negative 2,223, and false negative 61. Hence, the root has a total test data of 793. From these data, 732 samples are classified into the root category, 2 samples are misclassified as leaf and seed, 2 samples are misclassified as unidentified. This shows the model misses only 0.5 % of the correct classification and a desirable result of the classification handled by Mobile Net.

Table 6 is taken from the classification report of Mobile Net to show the performances of the model in each category (class). The results of F1_score is taken for the description because this evaluation metric takes the advantages of both precision and recall by making the harmonic mean of Precision and Recall. The F1_score shows, the model obtained the adequate performances on both classes. It is also possible to make sure by referring from the above confusion matrix. The prediction results at the end of this section also another evidence for the performance of a model that is adopted in this study.
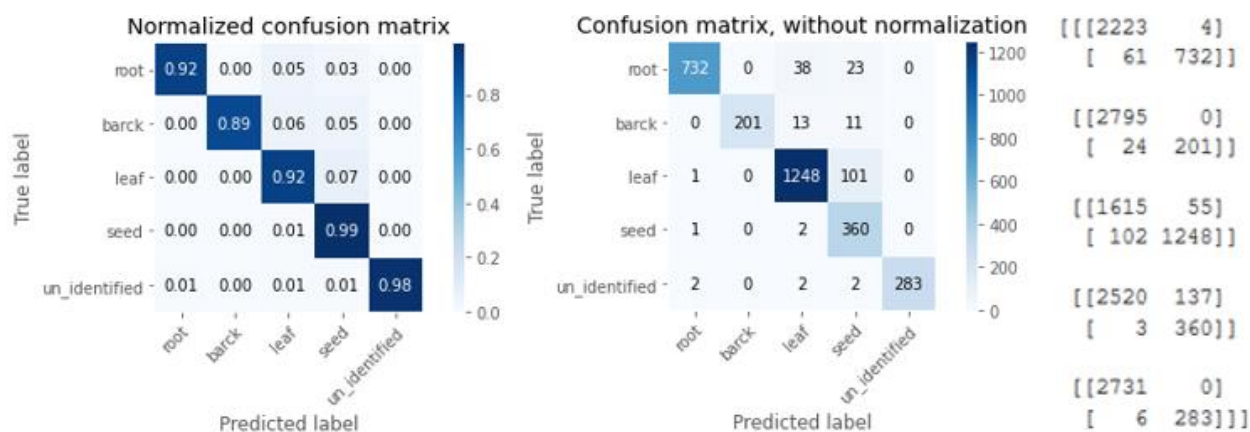
Figure 16: Confusion matrix of the best model (MobileNet)

Table 6: Performance measurement results of the best model (MobileNet)

Categories (classes)	Precision in%	Recall in %	F1_Scoer in %	No. of test data
Root	99	92	96	793
Bark	100	89	94	225
Leaf	96	92	94	1350
Seed	72	99	84	363
Un identified	100	98	88	289
Micro average	94	94	94	
Macro average	94	94	93	3020
Weighted average	95	94	94	

3.8. Prediction

The following screenshot is taken from the sample demonstration which indicates as the model can predict the medicinal plant parts effectively. The parts that have the percentage value approaches to 1 means the model classify accurately. No matter about the value 1 and 0.9,
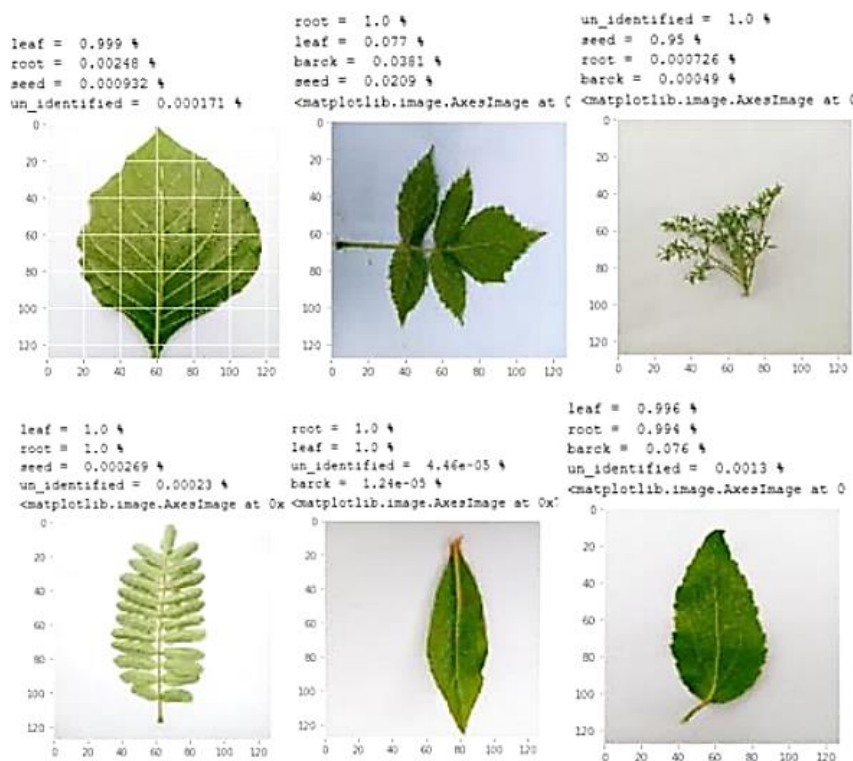

Figure 17: Samples of medicinal plant predicted by trained model

it is simply calculated as the decimal point. It is easy to make it whole number by multiplying by 100 and change to 100 % and 99%.

4. Conclusion and Recommendations

This research is the first to classify the specific parts of medicinal plants using multi-label classification. Since there is no published medicinal plants data based on their specific parts, the dataset is prepared based on the reference of pathobiological researches and consultation of experts who are currently working on preparations of traditional medicine. From the observation of data collections, leave part of the plants takes the majority role in curing diseases.

Based on the experimental results, we can conclude that over fitting of the models does not only depend on the number of a data set and the depth of hidden layers in CNN, it is also determined by the total number of scalar multiplication of filters (kernel) calculated by the convolution process. In this study, Mobile Net contains 30 convolutional layers and VGG16 has only 16 layers. But the Mobile Net does not over fit than VGG16, because it uses depth wise separable convolution technique. This technique reduces the scalar multiplication of the kernel by a factor of 1/20. So, the total number of scalar multiplications calculated in Mobile Net is very small compared to VGG16. That is why Mobile Net does not over fit even though it has the maximum number of convolutional layers than VGG16. In this study, three main pillar points are identified: 1) batch normalization at the fully connected layers of the CNN makes the model learn faster in a small number of training epochs, 2) the utilization of plant leaves image from its backside enables the models to extracts the vein features accurately. The combined effect of the two cases leads the model to score a good performance while it is tested by unseen (new) data, and 3) depth wise separable convolution highly determines the over fitting and under fitting conditions of the models.

Finally, this study deduced that using a learning rate of $1e^{-4}$, batch size of 32, and optimizer of Adam, the models achieved classification performances of 94%, 86%, and 68% for Mobile Net, VGG16, and Inception_V3, respectively. This result is based on F1_score evaluation metric. Generally, after conducting more than 72 experiments on 15,100 images of medicinal plants, the study concludes Mobile Net is the fastest model to train and test medicinal plant parts. This model is also selected as a suitable deep learning technique to come up with a good solution for the identified problems by achieving higher classification performance than VGG16 and Inception_V3.

This study introduces the idea and testing of the identification and classification of medicinal plant parts using deep learning technique based on multi label categories to answers the question of which part of medicinal parts are used to medicine preparation? But the study does not incorporate the identification of diseases cured by the identified parts of the plants, limited with small number of data sets, and the optimal learning rate is manually searched using grid search method. For the future work, we recommend researchers to incorporate the following concept for more relevant results:

- It is good using FastAI (a deep learning library running on top of Pytorch) to make the classification task easier and fast by finding the optimal learning rate using learning rate finder.
- Increasing the number of data set will leads the model to obtain accurate results.
- If researchers incorporate the corresponding diseases to be cured by specified parts, the output of a system will be more useful and relevant.
- If the future researcher works on the way that can increase the model performance, using multi label classification makes the research finding appreciable.

Reference

Amuthalingeswaran, C., Sivakumar, M., Renuga, P., Alexpandi, S., Elamathi, J., & Hari, S. S. (2019). Identification of medicinal plant's and their usage by using deep learning. *Proceedings of the International Conference on Trends in Electronics and Informatics, ICOEI 2019*, *Icoei*, 886–890. https://doi.org/10.1109/ICOEI.2019.8862765

Bandara, M., & Ranathunga, L. (2019). Texture Dominant Approach for Identifying Ayurveda Herbal Species using Flowers. *MERCon 2019 - Proceedings, 5th International Multidisciplinary Moratuwa Engineering Research Conference*, 117–122. https://doi.org/10.1109/MERCon.2019.8818944

Bekkar, M., Djemaa, H. K., & Alitouche, T. A. (2013). Evaluation Measures for Models Assessment over Imbalanced Data Sets.

Journal of Information Engineering and Applications, *3*(10): 27–38.

Dileep, M. R., and P. N. Pournami. 2019. "AyurLeaf: A Deep Learning Approach for Classification of Medicinal Plants." *IEEE Region 10 Annual International Conference, Proceedings/TENCON* 2019-Octob: 321–25.

Jeni, L. A., Cohn, J. F., & De La Torre, F. (2013). Facing imbalanced data - Recommendations for the use of performance metrics. *Proceedings - 2013 Humaine Association Conference on Affective Computing and Intelligent Interaction, ACII 2013*, 245–251. https://doi.org/10.1109/ACII.2013.47

Lee, S. H., Chan, C. S., Mayo, S. J., & Remagnino, P. (2017). How deep learning extracts and learns leaf features for plant classification. *Pattern Recognition*, *71:* 1–13. https://doi.org/10.1016/j.patcog.2017.05.015

Naresh, Y. G., and Nagendraswamy, H.S. (2016). "Classification of Medicinal Plants: An Approach Using Modified LBP with Symbolic Representation." *Neurocomputing* 173: 1789–97. http://dx.doi.org/10.1016/j.neucom.2015.08.090.

Pacifico, L. D. S., Britto, L. F. S., Oliveira, E. G., & Ludermir, T. (2019). Automatic classification of medicinal plant species based on color and texture features. *Proceedings - 2019 Brazilian Conference on Intelligent Systems, BRACIS 2019*, 741–746. https://doi.org/10.1109/BRACIS.2019.00133

Prasad, S., & Singh, P. P. (2017). Medicinal plant leaf information extraction using deep features. *IEEE Region 10 Annual International Conference, Proceedings/TENCON*, *2017-Decem*, 2722–2726. https://doi.org/10.1109/TENCON.2017.8228324

R.janani, A. G. (2013). Identification of selected medicinal plant leaves using image features and ANN. *International Conference on Advanced Electronics System, IEEE*, 99–117.

Sivaranjani, C., Kalinathan, L., Amutha, R., Kathavarayan, R. S., & Jegadish Kumar, K. J. (2019). Real-time identification of medicinal plants using machine learning techniques. *ICCIDS 2019 - 2nd International Conference on Computational Intelligence in Data Science, Proceedings*, 1–4. https://doi.org/10.1109/ICCIDS.2019.8862126

Tan, J. W., Chang, S. W., Abdul-Kareem, S., Yap, H. J., & Yong, K. T. (2020). Deep Learning for Plant Species Classification Using Leaf Vein Morphometric. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *17*(1): 82–90. https://doi.org/10.1109/TCBB.2018.2848653