© Universiti Tun Hussein Onn Malaysia Publisher's Office

IJIE

Journal homepage: http://penerbit.uthm.edu.my/ojs/index.php/ijie
ISSN : 2229-838X e-ISSN : 2600-7916

The International Journal of Integrated Engineering

Plant Classification based on Leaves using Artificial Neural Network

Vishwad Desai[1], Vijay Savani[2*], Rutul Patel[3]

[1]Alumani EC Dept, Institute of Technology,
Nirma University, Ahmedabad, 382481, India

[2,3]Department of Electronics and Communication Engg, Institute of Technology,
Nirma University, Ahmedabad, 382481, India

*Corresponding author

DOI: https://doi.org/10.30880/ijie.2021.13.06.003
Received 14 March 2020; Revised 29 January 2021; Accepted 04 Feb 2021; Available online 31 August 2021

Abstract: Manual methods to examine leaf for plant classification can be tedious, therefore, automation is desired. Existing methods try distinctive approaches to accomplish this task. Nowadays, Convolution Neural Networks (CNN) are widely used for such application which achieves higher accuracy. However, CNN's are computationally expensive and require extensive dataset for training. Other existing methods are far less resource expensive but they also have their shortcomings for example, some features cannot be processed accurately with automation, some necessary differentiators are left out. To overcome this, we have proposed a simple Artificial Neural Network (ANN) for automatic classification of plants based on their leaf features. Experimental results show that the proposed algorithm able to achieve an accuracy of 96% by incorporating only a single hidden layer of ANN. Hence, our approach is computationally efficient compared to existing CNN based methods.

Keywords: Artificial neural network, feature extraction, dimensionality reduction, classification, plant

1. Introduction

Classifying plants based on their leaves with the utilization of automation and machine vision can be of great help. For instance, a botanist can quickly identify a plant using a leaf's picture, make an entire record of the plant species present in that environment. Another example is just a hiker or a traveler can obtain information about a plant perhaps also getting alerted to whether is it poisonous or not. The manual method of classification might require a large logbook, searching and sifting through perhaps a mountain of information, the application of automation in such a case can expedite the process. Due to the development of sophisticated machine learning algorithms, one can classify plants based on their leaves accurately and automatically using Artificial Neural Networks (ANN). For automatic classification, firstly, Neural Network is trained based upon carefully chosen unique features. Next, for the query plant leaf image, those features are extracted for classification. For the classifier, feature selection is critical. Features that can individually distinguish samples quickly and a massive number incorporated in the classifier will make the classifier perform better. For example, two leaves can be distinguished quickly when they have very different colors and a colour feature is used by the classifier. A feature capturing the same information in fewer computations, less intricate computations will be preferable too. The selection of the classification method is either K-Nearest Neighbors (KNN) or Support Vector Machines (SVM) [1] . Each of the mentioned classifiers has inherent pros, cons, nuances, distinctions, conveniences, and inconveniences. In general, the SVM classifier has a very complex decision boundary but should not be used for classification involving many computationally expensive classes. In such cases, the SVM classifier would be very slow and hence not preferable. On the other hand, the KNN classifier is simple, but may not achieve the required optima in the training/adjusting period as initialization begins with randomly placed centroids [2]. Due to this, in most cases, KNN

Corresponding author: vijay.savani@nirmauni.ac.in does not translate to the optimal location that results in the least error in classification. A lot of work has been done in the field of plant classification using leaves previously, therefore, it is important to know about the performance of the existing classifiers and their capabilities. Hence, a combination of a good classification algorithm and distinguishing features can make an efficient Artificial Neural Network which has been proposed in this paper. The next section describes the existing literature which utilizes feature-based plant classification.

2. Related Works

Previously, a lot of work has been done in the field of plant classification using leaf images. Beginning from the pioneers, *Wu et al.* in 2007 proposed a PNN (Probabilistic Neural Network) based classifier [3]. The classifier is comprised of features of shape like length, width, diameter, and morphological features like vein. The proposed method achieved an accuracy greater than 90%. Next, *Singh et al.* in 2010 combined the concept of Binary decision Tree and SVM [1]. In the proposed classifier, the architecture was an SVM at every node of the binary tree for classification. The method achieved very high accuracy and claimed to have better generalization capabilities than the one proposed by *Wu et al*. [3]. Using similar shape features *priya et al*. in 2012, once again used SVMs in their methodology [4]. The classifier achieved good results with respect to time of execution on samples from the Flavia dataset. Following this, *Gwo & Wei* in 2013 proposed a method to extract leaf features from key points on leaf contours [5]. A feature involved calculating the centroid of a leaf contour and the distances to individual points of the leaf contour from the centroid. The distances are then used to form a length histogram, which is then normalized. A fuzzy score algorithm was then applied to the normalized histogram to compensate for the differences in length histogram of the same species. The fuzzy score matrix was then given as an input to a Bayes classifier for classification. The authors stated that their methodology could not fail when rotation, scale changes were introduced. The method managed to achieve a very high accuracy of 92.7%. Succeeding them, *satti et. al* in 2013 applied an ANN for the purpose of classification [6]. This method uses shape, digital morphological, and a color feature to extract color information from the leaf. Uniquely, a tooth feature was incorporated to count tooth-like features, jagged edges which some leaves have. This method achieved a high accuracy of 93.33% on the ANN.

Distinctively, *kadir, et. al,* in 2013 used texture features [7]. This method used many color moments, for example, mean, skewness, kurtosis, etc. All the features were then inputted to a PNN. The classifier achieved an accuracy of 93.75%. The work done by authors extended ahead by *Kadir, Abdul* in 2014 added grey-level co-occurrence matrix-based feature calculation, and shen features added to the existing ones from his work previously [8]. This time though, a Bayesian classifier was used and an accuracy of 97.19% was reported. *Aakif & Khan* in 2015 used an ANN with a unique shape feature, morphological features, and Fourier descriptors and achieved the highest accuracy of 96% [9]. The Shape Defining Feature (SDF) of calculating the shape feature involved calculating gradients on the edge of the leaf [10].

Image feature extraction techniques like SIFT have also been applied to make a classifier. *Lavania & Matey* in 2014 used SIFT and a contour-based corner detection method to make a classifier and achieved a very high accuracy of 87.55% [11]. The proposed classifier was also claimed to reduce false positives and false negatives in contrast to Curvature Scale Space (CSS) based methods. Most Recently in 2018, *Saleem et. al.* proposed a KNN classifier that achieved 98.75% accuracy on the Flavia dataset [12]. This method incorporated many statistical features like average intensity, average contrast, skewness, kurtosis, and entropy and was claimed to outstrip CNN's when the training set was small. CNN's are very powerful learning tools capable of learning features in-depth [13]. Author *Krizhevsky et al.,* in 2012 [14] have given the AlexNet architecture which can achieve up to 99.48% accuracy as reported by *saleem et. al.* [12]. The performance of CNN's is convincing, but they need a massive amount of training set which both, is temporally, computationally expensive. Similar fields are flower classification, plant classification using flowers. A swift look at some of the work in those fields also allows us to know more about features, their extraction methods which may be applicable in this field too. This completes the part of enlightening on the history of this field, the novel method is discussed next.

3. Proposed Method & Work

This section discusses this work, beginning from the typical method, the features selected, processing related to features to lastly, Dimensionality Reduction.

3.1 General Methodology

Fig. 1 illustrates the proposed workflow for the plant classification. First, the acquired image is preprocessed to extract a required portion of the image and convert it into forms that conveniently allows feature extraction. In this stage, the boundary of plant leave is extracted and represented with white pixels. This stage is illustrated in Fig. 2 where the plant leave image boundary is extracted. The next step, feature extraction, counting the number of white pixels in Fig. 2 (c). This represents the leaf area and it is considered as a feature describing that plant. Following to feature extraction stage of Fig. 1, features are normalized as a requirement for Dimensionality Reduction. Dimensionality Reduction allows the removal of features that contain redundant information as it conveys the same information. Next, in the training phase, the classifier is provided with the training samples along with labels. In contrast, during the testing, we evaluate the performance of the classifier. Further details about each block of Fig. 1 are discussed in subsequent sections.

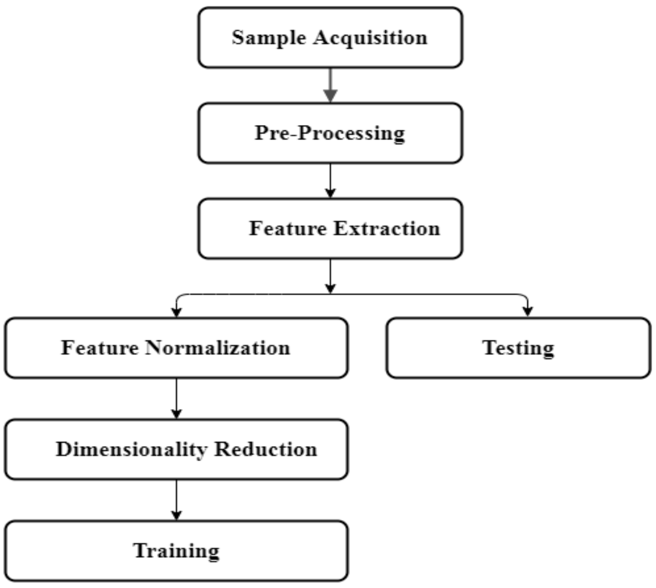


Fig. 1 - Methodology

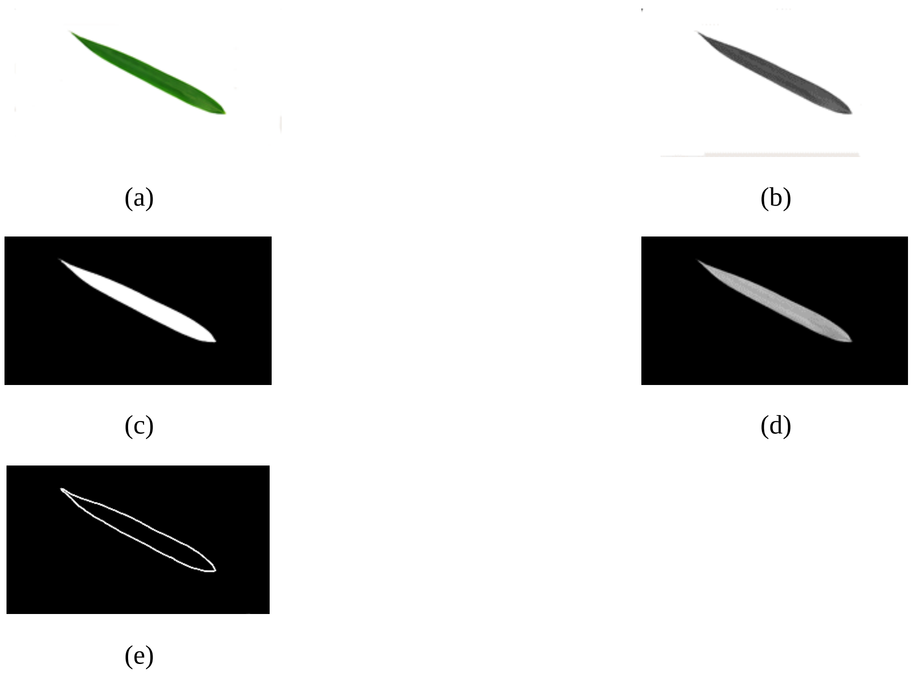


Fig. 2 - Preprocessing of the leaf image

3.2 Dataset

The dataset that should be used depends on (not just) which classifier has been used. CNN's would require a large dataset, but then an extremely large dataset will also be computationally and temporally expensive when training. The dataset should also have enough variance in samples per class so that the classifier is capable of generalization. Among all, one of the most suitable datasets is the Flavia dataset [15]. It is a widely used dataset that has 32 species, with 1907 samples. Each of the samples has a white background with the leaf on top of it. The leaves are aligned in arbitrary directions which may or may not be convenient. It is prudent to use this dataset since a lot of work has been performed on it, and that also gives a lot of work to compare the proposed method. Comparing performance using different datasets should not be done since each dataset brings different conditions. The background in the Flavia dataset is plain white, but if the background is other than plain in a different dataset, background removal techniques would have to be applied. Even then, the background may not fully be removed from the image. In such a case anything other than the region of interest is a noise that will decrease the accuracy of the classifier. Hence, the classifier with images that contain noise will be judged as inferior to the classifier where the samples were not contaminated with noise.

3.3 Preprocessing

In general, image processing is performed to remove noise, interference, and undesired information. If an image is noisy, the noise should be reduced otherwise the noise can lead to negative consequences. For example, gloss on the leaf due to lighting conditions can lead to wrongly calculated color features. Here, image Preprocessing is done primarily to extract features from the image. The preprocessing in Fig. 2 has been achieved as explained ahead. From Step 1 to step 2, convert to grayscale, the following formula

$$C = 0.299R + 0.587G + 0.114B \qquad (1)$$

Where C is the grayscale, value, R is the value of the red channel at that pixel, G is the value of the green channel at that pixel, B is the value of the blue channel at that pixel. Step 2 to step 3 requires the use of color inversion by applying bitwise not operation on each pixel of the image. This is convenient since from a white background we have a black background that will not interfere when doing operations that involve pixel intensity addition. Next, step 3 to 4 binarizes the image by thresholding:

$$V = \begin{pmatrix} 255 : p > 17 \\ 0 : otherwise \end{pmatrix} \qquad (2)$$

Here, threshold value 17 was chosen empirically for better accuracy. Finally, moving from step 4 to step 5, the edges are extracted. While existing algorithms generally utilize Laplacian filters, we have used Canny Edge detection. Canny edge detection seems more appropriate in the general sense to detect edges as the canny detector will only output the edges and seems more relevant when there is abundant information in the image. The Canny edge detector by Canny in 1986 finds the difference between points to locate edges [16]. The difference must be with a manually specified range. Sometimes the difference may not satisfy the criteria but if it is near the threshold and attached to what has been classified as an edge by the canny detector the detector will classify that as an edge.

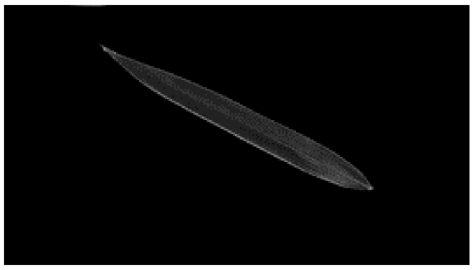
3.4 Features

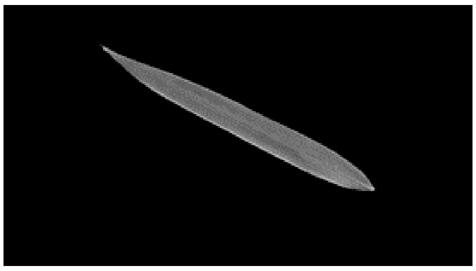
Features were selected to try and capture as many distinctive features as possible. By plain observation of the images, one can determine various features that can be used to set leaves apart. For example, its size, length, width, shape, jagged edges, tooth features, vein structure, etc. The following elaborates on the features used and how they are calculated. The categories of features are used in this methodology are Color Features, Geometric Features, Morphological Features, and Statistical Features. The feature definition and extraction part is discussed below.

A. Color Features:

1) RGB channel means: The method is proposed by Kekre in 2011 describes a method of calculating color features by calculating the average means [17]. In the proposed algorithm, first, the three color channels namely Red, Green, and Blue are separated. Then for each channel, row mean and column mean of colors are calculated. The average of all row means and all columns means is calculated for each plane. The features of all 3 planes are combined to form a feature vector.

2) RGB channel deviation: Calculated as the standard deviation of each color channel, this can help capture more color features from the leaf. Some leaves might have highly varying color content due to distinctive visual patterns.


(a)


(b)

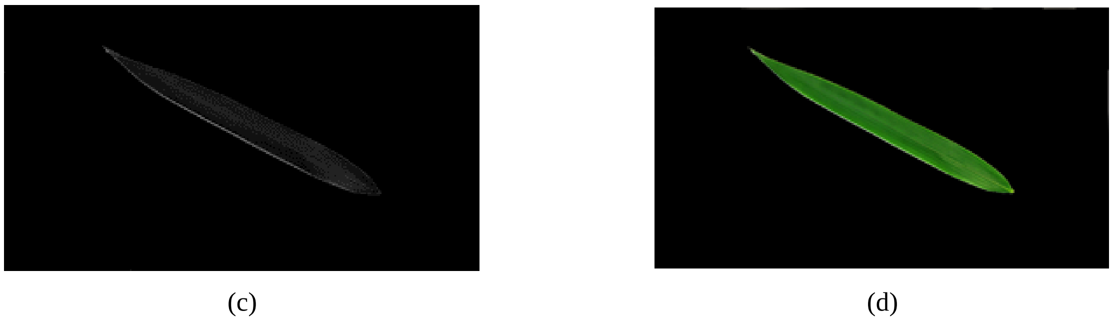


Fig. 3 - Additional Preprocessing of color separation

The extraction of color features requires additional preprocessing which has been shown in Fig. 3.

B. Geometric Features:

1) Diameter (D): The diameter of the leaf is the longest distance between any two points on the closed contour of the leaf or leaf with only the edges highlighted. (Refer Fig. 2 (e)).
2) Physiological Length (L): It is the length of the line connecting the two terminal points of the main vein in the leaf. (Refer Fig. 2 (e)).
3) Physiological Width (W): It refers to the distance between the two endpoints of the longest line segment perpendicular to the physiological length. (Refer Fig. 2 (e)).
4) Leaf Area (A): It is the number of pixels of binary value 1 on smoothed leaf image. (Refer Fig. 2 (d)).
5) Leaf Perimeter (P): It is the number of pixels along the closed contour of the leaf or leaf with only the edges highlighted. (Refer Fig. 2 (e)).

C. Morphological Features:

1) Smooth Factor: This is defined as the ratio between the area of leaf image smoothed by a 5x5 rectangular averaging filter and the one smoothed by a 2x2 rectangular averaging filter.
2) Aspect Ratio: This is defined as the ratio of physiological length and physiological width. It is calculated as in (3)

$$AR = \frac{L}{W} \tag{3}$$

3) Form Factor: It is defined as the ratio between the area of leaf and the area of a circle. It is calculated as in (4)

$$FF = \frac{4\pi A}{P^2} \tag{4}$$

4) Perimeter Ratio of Diameter: It is defined as the ratio of the perimeter of the leaf to the diameter of the leaf, calculated as in (5)

$$PRD = \frac{P}{D} \tag{5}$$

5) Perimeter Ratio of Physiological Length and Physiological Width: It is defined as the ratio of the perimeter of the leaf to the sum of its physiological length and physiological width, calculated as

$$PRLW = \frac{P}{L+W} \tag{6}$$

6) Vein Features: Different species have different leaf vein patterns which can be used in distinguishing the leaves that have a similar shape. Their formula is given in (7) and (8). The standard procedure for computing the vein features is to

   1. Perform a morphological opening operation on the grayscale image. A flat, disk-shaped structuring element of radius 1, 2, 3, 4 is used.

2. The resultant image is then subtracted from the contour of the leaf. The output resembles the vein structure of the leaf based on which the following features are obtained

$$V_i = \frac{A_i}{A} \tag{7}$$

$$V_5 = \frac{A_4}{A_1} \tag{8}$$

In (7), ‘i’ refers to the ith radius or features

7) Shape variance: Calculated by taking the standard deviation of all distances which originate from the centroid of the image with outer edges of the leaf to each point on the leaf edge. It can help to capture irregularly shaped leaves.

D. Statistical Features:

1) Average Intensity: It is the mean of all the values of pixels in the grayscale image. It is calculated as in (9)

$$\mu = \frac{\sum_{p=1}^{N} G_p}{N} \tag{9}$$

The formula above is for computing the mean of a grayscale image, N is the number of pixels, G refers to the value of the pixel in the grayscale image.

2) Average Contrast: It is the variance of all pixel values in the grayscale image. It is calculated as in (10)

$$\sigma^2 = \frac{\sum_{p=1}^{N} (G_p - \mu)^2}{N} \tag{10}$$

3) Skewness: It is the measure of the third moment of the pixel values in the grayscale image. It represents the asymmetry in the histogram of pixel values. If it is positive, it is skewed to the right, else if negative it is skewed to the left. It is calculated as in (11)

$$\gamma = E\left(\frac{L}{\sigma}\right)^2 \tag{11}$$

Where E is the expectation operator

4) Kurtosis: It is the measure of the fourth moment of all the pixel values in the grayscale image. It indicates how much is the histogram “tailed”. If it is a normal distribution we obtain kurtosis 0, negative if is uniformly distributed, positive if peaked more than the normal distribution, it is calculated as in (12)

$$K = \left(\frac{\sum_{p=1}^{N} (G_p - \mu)^4}{\left(\sum_{p=1}^{N} (G_p - \mu)^2 \right)^2} \right) \tag{12}$$

5) Entropy: Measures randomness in the values of the image pixels an is calculated as in (13)

$$E = \sum h \times log_2 h \tag{13}$$

6) Smoothness: This is a measure of the relative intensities in the segmented region of the image. It can be calculated as in (14)

$$R = 1 - \frac{1}{(1+\sigma^2)} \qquad (14)$$

7) Uniformity: It measures the equality of the grayscale values of the image, it is calculated as in (15)

$$U = \sum_{n=0}^{N-1} P^2(z_n) \qquad (15)$$

The information given above completes the part about features and their implementation, but just inputting features to the model is not the end of the discussion about features. Even features need to be preprocessed for the succeeding part of dimensionality reduction which is a part of this methodology. The preprocessing is done employing mean normalization and then scaling the data point to lie between zero and unit variance. Mathematically, the preprocessing operation can be represented as in (16)

$$d_p = \frac{d - m_s}{r_s} \qquad (16)$$

Where d_p is the data point processed, d is the actual data point, m_s is the mean of the set of values in which the data point is located (in different terms the mean of all values from all samples for a particular feature) and r_s is the range of the set of values in the feature set. The range is the difference between the maximum and minimum value of that set. Finally, after discussing features, their processing, we move on to Dimensionality Reduction.

3.5 Dimensionality Reduction

Sometimes many features represent redundant information and do not help in improving the classifier's output. Such features take up space and utilize computation to no avail. Hence, it is necessary to reduce the feature set to only those features that are unique and convey the most information. To accomplish this task, the Principal Component Analysis (PCA) has been used. PCA helps determine which features contribute most and which are redundant. Mathematically, PCA would do the following

$$R^x \rightarrow R^y \qquad (17)$$

Where x is greater than y indicating Rx is a set larger than Ry.

This completes the discussion related to features. An important constituent of the classifier is the classification method (classifier/classification algorithm). Here, ANN has been used the details of which follow.

4. Artificial Neural Network

ANNs are an imitation of the network that exists in the brains of living things. As the name suggests they are a massive network that consists of individual units called neurons. When a neuron receives enough inputs such that their aggregate crosses the threshold of the neuron, the neuron "fires", i.e. generates its signal which it transmits to whatever the neuron is succeeded by. Fig. 4 shows the mathematical model of a neuron. A huge network containing millions of such neurons exists within our brains. As the number of neurons and intermediate (hidden) layers increases the network is capable of learning deeper features or more complex features.

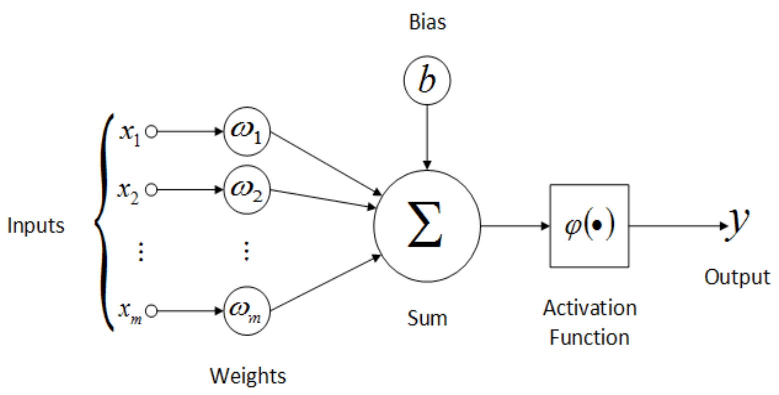


Fig. 4 - Mathematical Model of Neuron

The previous statement can be understood by the following example, the first hidden layer identifies eyes, the second hidden layer identifies nose and mouth, the third hidden layer identifies the entire face. (This does not represent how a neural network might function). The architecture of the ANN proposed here is very simple. It only has 1 hidden layer. Most applications of ANNs can be accomplished by one or two hidden layers. The consensus about ANNs is that the hidden layers have fewer nodes than the input layer and it holds here too. The hidden layer has fewer nodes than the input layer, their number counts to 20. The activation function mentioned in Fig. 4 is the sigmoid function for this neural network architecture. The input and output layers use the sigmoid function too. The training of a neural network is similar to other machine learning approaches in the part that it involves minimizing the error of the neural network output. The error is computed as per the function of choice which here is Mean Square Error. The mean square error function mathematically is given as

$$E = \frac{\sum_{i=1}^{N}\left(\hat{Y}_{(i)} - Y_{(i)}\right)^2}{N} \tag{18}$$

Where E is the error, N is the number of samples, $\hat{Y}_{(i)}$ is the predicted output for sample ‘i’, $Y_{(i)}$ is the true/actual output for that sample. Note that this is a general representation. The differences should not be overlooked when the values are vectors of length greater than 1. In that case, the subtraction will yield a vector that must be summed too to obtain just an absolute number, not a collection like a vector or a matrix. This completes information related to the structure of the classifier. In the next section, the training and testing of the classifier will be discussed.

5. Training and Testing

This section elaborates on the hardware, software required for training the classifier, the training phase, and finally the testing phase. A machine learning approach requires the machine to learn/adjust. The same was done over here.

1.5 Hardware

The work, data size are simple enough to not require specific, specialized hardware. Machine Learning, Deep Learning tasks that involve hundreds or more features, data set, training set, in the millions will require very powerful hardware like the highest end GPUs, CPUs, and a large quantity of RAM. Still, care should be taken for tasks that require storage space, processing power, otherwise, it can lead to system crashes. The following are the system specification that was used to do this work, in a nutshell, it is just a simple Laptop computer. Note that while it is enough, feature extraction of all 1907 images can take up to 20 hours with such a system. The CPU onboard was Intel i7-5500U, the GPU onboard was Nvidia Geforce GT 940M which has 2 GB Memory, RAM had a capacity of 8 GB, OS was Ubuntu 18.04.2 LTS.

1.6 Software

The programming language selected was Python (v3.6.7) since it offers many libraries like Pandas (Data Handling, Manipulation), Numpy (Data Manipulation, Array Handling), Scikit-Learn (Algorithms, Preprocessing), Keras (Neural Networks, with a backend of Tensorflow), and OpenCV (Helps to speed up computation). Note that the entire implementation can be performed without major involvement of OpenCV, but the use of OpenCV to the greatest amount will speed up computation by magnitudes to the user’s convenience.

1.7 Training and Testing

An important fact to know when working with Artificial Neural Networks is that they need a significant amount of data when training, even more, when there are a large number of features and classes for classification. Hence, it was decided to train and test the Neural Network with 5 classes. The Flavia dataset is quite large but still does not have enough samples per class to train a large network for classification involving all 32 classes in the dataset. When the neural network is trained, the cost function (often referred to as loss) must decrease with iterations, that trend was observed and concurrently the accuracy increased indicating the training process is proceeding correctly. The training is succinctly presented as graphs ahead. An important note before discussing the training results and proceedings, the training was done for 2000 epochs (‘iterations’ in Fig. 5 and Fig. 6). Each epoch means a forward propagation where the output is computed for an input/set of inputs. During backpropagation, the weights related to neurons’ connections are adjusted to minimize error. Fig. 5 shows the accuracy of the proposed model over the iterations. Over the initial 600 iterations, the model accuracy increases substantially from 0 to 0.9. This indicates that the neural network is learning and updating the model variables. It also indicates that training data contains samples enabled learning and hyperparameters of the model are suitable. The model converges at approximately 1200 iterations and accuracy does not improve further. Fig. 6 shows the loss versus iterations to observe the effectiveness of chosen loss function. The loss is the error metric for the output of a neural network which is calculated in this case as given by (19). The loss is decreasing with the number of iterations which indicates the neural network is learning and correcting the error in its output by adjusting the input weights and biases of the neurons. As shown in Fig. 6, during the initial 100 iterations, the model variable learns in a way to reduce the loss. However, the model takes about a further 1100 iterations to converge to a minimum possible loss. Therefore, from Fig. 5 and Fig.6, we can consider that the proposed model is converging at approximately 1200 iterations to achieve maximum possible accuracy. As mentioned previously, the testing was done on samples from 5 classes from the Flavia dataset. The accuracy and other metrics for evaluation were calculated based on the results of the testing phase. The results are discussed next.

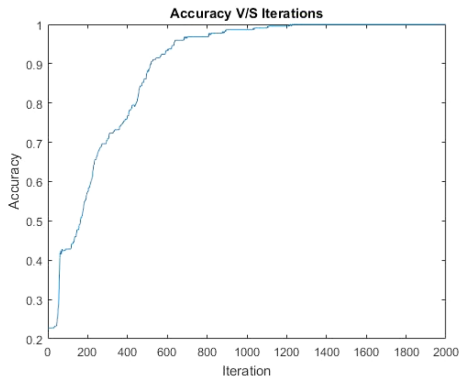


Fig. 5 - Accuracy v/s. Iterations

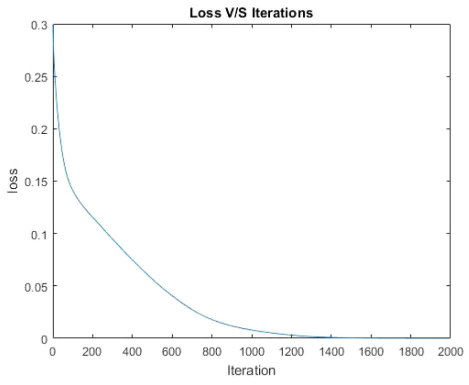


Fig. 6 - Loss v/s. Iterations

Experimental Results

To evaluate and express the model's performance, various metrics are used. These metrics conveniently convert the model's output into numbers which can be used to compare different models. Accuracy is one of the general metrics of performance. It expresses the overall performance of the model which is can be used to compare with other models. Other metrics allow us to evaluate specific parameters of the models which include false positives, false negatives. They are precision and recall.

Table 1 - Comparison of proposed technique

Methodology	Accuracy (%)
Satti, Satya, & Sharma [5]	93.33
Aakif & Khan [8]	96
Saleem, Akhtar, Ahmed & Quereshi [10]	98.75
Proposed Methodology	96

Table 2 Precision and recall

Classes	**Metrics**	
	Precision (%)	***Recall (%)***
Big-Fruited Holly	100	100
Anhui Barberry	100	100
Chinese Toon	87.5	93.33
Beale's Barberry	91.66	84.62
Canadian Poplar	100	100
Average	95.83	95.59

1.8 Evaluation Metrics

A. Accuracy, as defined in (19)
B. Precision, as defined in (20)
C. Recall, as defined in (21)

$$Accuracy(\%) = \left(\frac{N_c}{N_t}\right) \times 100 \tag{19}$$

$$precision = \left(\frac{t_p}{t_p + f_p}\right) \times 100 \tag{20}$$

$$recall = \left(\frac{t_p}{t_p + f_n}\right) \times 100 \tag{21}$$

Where t_p is the number of true positives, f_p is the number of false positives, f_n is the number of false negatives

1.9 Statistics

The classifier has achieved an accuracy of 96% which is very high. This accuracy is compared to other methods in Table 1 given below. These methods either employ neural networks or are the latest state of the art models. The other metrics Precision and Recall are discussed herein in

Table 2. The precision and recall are very high indicating that this method manages to perform on metrics other than accuracy. The metrics might be slightly lower than the ones stated by Saleem, Akhtar, Ahmed & Quereshi but the testing set used by the authors was larger which means one sample is a smaller piece of the pie, i.e. one sample contributes lesser to overall [12]. The discussion related to the performance of the classifier ends here. Next, comes the part of conclusions reached from applying this methodology.

Conclusion

The classifier manages to achieve very high performance. The feature selection is such that they convey unique information, the features extraction has been tried to do as simply as possible. As previously stated, one of the objectives was to be as comprehensive as possible in the feature selection which was accomplished here by incorporating all categories of features, shape, morphological, color, and statistical. The classification method, ANNs, is a really good tool when comes to the learning part, they can establish relationships between features, and hence they were used here. Just by controlling some parameters the size of the ANN can be augmented to make the classifier larger more deep (concurrently increasing training set size). This methodology attempts to overcome shortcomings from previous methods.

The method used by *Aakif & Khan* [9] required to normalize images by rotation. The rotation involves aligning the leaf with its tip upward which cannot be done in a simple way when leaves have an irregular or unconventional shape. That process is not needed in this case. There might be training needed to even recognize the tip of leaves which can be of many shapes. *Saleem, Akhtar, Ahmed & Quereshi* did not include color features [12]. *Satti, Satya, & Sharma,* incorporated geometric morphological tooth which is difficult to calculate since it can give varying results due to its reliance on a manually set parameter [6]. The method here of using shape variance tries to capture irregular shapes by using standard deviation which does not require parameter adjustment in contrast to the tooth feature used by them. This concludes the argument for this method posited as superior. This method relied on a simple, single hidden layer neural network with a typical activation function, changes in the neural network may be done by checking where, which activation function is best suited. Extensive hyperparameters tuning was not done here so one can devote time to it. Accuracy may increase if Fourier based features can be incorporated or a different way of capturing shape be used here. The testing set here comprised of samples from the Flavia dataset, but while applying the classifier practically, there would be additional information in the background which is both noise and interference necessitating additional preprocessing.

2. References

[1] K. Singh, I. Gupta and S. Gupta, "SVM-BDT PNN and fourier moment technique for classification of leaf shape," *International Journal of Signal Processing, Image Processing and Pattern Recognition,* vol. 3, pp. 67-78, 01 2010

[2] M. Nilsback and A. Zisserman, "Automated Flower Classification over a Large Number of Classes," in *2008 Sixth Indian Conference on Computer Vision, Graphics Image Processing*, 2008

[3] S. G. Wu, F. S. Bao, E. Y. Xu, Y. Wang, Y. Chang and Q. Xiang, "A Leaf Recognition Algorithm for Plant Classification Using Probabilistic Neural Network," in *2007 IEEE International Symposium on Signal Processing and Information Technology*, 2007

[4] C. A. Priya, T. Balasaravanan and A. S. Thanamani, "An efficient leaf recognition algorithm for plant classification using support vector machine," in *International Conference on Pattern Recognition, Informatics and Medical Engineering (PRIME-2012),* 2012

[5] C.-Y. Gwo and C.-H. Wei, "Plant Identification Through Images: Using Feature Extraction of Key Points on Leaf Contours," *Applications in plant sciences,* vol. 1, 11 2013

[6] V. Satti, A. Satya and S. Sharma, "An automatic leaf recognition system for plant identification using machine vision technology," *International Journal of Engineering Science and Technology (IJEST),* vol. 5, pp. 874-879, 04 2013

[7] A. Kadir, L. Nugroho, A. Susanto and P. Santosa, "Leaf Classification Using Shape, Color, and Texture Features," *International Journal of Computer Trends and Technology*, vol. 1, pp. 225-230, 07 2013

[8] A. Kadir, "A Model of Plant Identification System Using GLCM, Lacunarity and Shen Features," *Research Journal of Pharmaceutical, Biological and Chemical Sciences,* vol. 5, pp. 1-10, 01 2014

[9] A. Aakif and M. Faisal Khan, "Automatic classification of plants based on their leaves," *Biosystems Engineering,* vol. 139, pp. 66-75, 11 2015

[10] S. Dharwadkar, G. Bhat, N. V. S. Reddy and P. K. Aithal, "Floriculture classification using simple neural network and deep learning," in *2017 2nd IEEE International Conference on Recent Trends in Electronics, Information Communication Technology (RTEICT),* 2017

[11] S. Lavania and P. S. Matey, "Leaf recognition using contour based edge detection and SIFT algorithm," in *2014 IEEE International Conference on Computational Intelligence and Computing Research*, 2014

[12] G. Saleem, M. Akhtar, N. Ahmed and W. S Qureshi, "Automated analysis of visual leaf shape features for plant classification," *Computers and Electronics in Agriculture,* vol. 157, pp. 270-280, 12 2018

[13] S. H. Lee, C. S. Chan, P. Wilkin and P. Remagnino, "Deep-plant: Plant identification with convolutional neural networks," in *2015 IEEE International Conference on Image Processing (ICIP)*, 2015

[14] A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, Lake, 2012

[15] Flavia, "Flavia," 24 Dec 2009 . [Online]. Available: http://flavia.sourceforge.net/. [Accessed 24 Dec 2019].

[16] J. Canny, "A Computational Approach to Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vols. PAMI-8, pp. 679-698, Nov 1986

[17] H. Kekre, D. Mishra, M. Stuti Narula and M. Vidhi Shah, "Color Feature Extraction for CBIR," *International Journal of Engineering Science and Technology,* vol. 3, no. 12, p. 8357–8365, 12 2011

[18] H. Yalcin and S. Razavi, "Plant classification using convolutional neural networks," in *2016 Fifth International Conference on Agro-Geoinformatics (Agro-Geoinformatics)*, 2016