Flower Species Recognition System using Convolution Neural Networks and Transfer Learning

I.Gogul, V.Sathiesh Kumar
Department of Electronics Engineering
Madras Institute of Technology, Anna University
Chennai-600044, India
gogulilangoswami@gmail.com

***Abstract*—Automatic identification and recognition of medicinal plant species in environments such as forests, mountains and dense regions is necessary to know about their existence. In recent years, plant species recognition is carried out based on the shape, geometry and texture of various plant parts such as leaves, stem, flowers etc. Flower based plant species identification systems are widely used. While modern search engines provide methods to visually search for a query image that contains a flower, it lacks in robustness because of the intra-class variation among millions of flower species around the world. Hence in this proposed research work, a Deep learning approach using Convolutional Neural Networks (CNN) is used to recognize flower species with high accuracy. Images of the plant species are acquired using the built-in camera module of a mobile phone. Feature extraction of flower images is performed using a Transfer Learning approach (i.e. extraction of complex features from a pre-trained network). A machine learning classifier such as Logistic Regression or Random Forest is used on top of it to yield a higher accuracy rate. This approach helps in minimizing the hardware requirement needed to perform the computationally intensive task of training a CNN. It is observed that, CNN combined with Transfer Learning approach as feature extractor outperforms all the handcrafted feature extraction methods such as Local Binary Pattern (LBP), Color Channel Statistics, Color Histograms, Haralick Texture, Hu Moments and Zernike Moments. CNN combined with Transfer Learning approach yields impressive Rank-1 accuracies of 73.05%, 93.41% and 90.60% using OverFeat, Inception-v3 and Xception architectures, respectively as Feature Extractors on FLOWERS102 dataset.**

Keywords—Deep Learning, Artificial Intelligence, Convolutional Neural Networks, Transfer Learning, Flower Recognition

I. INTRODUCTION

Plant species recognition based on flower identification remain a challenge in Image processing and Computer Vision community mainly because of their vast existence, complex structure and unpredictable variety of classes in nature. Because of these natural complexities, it is highly undesirable to perform normal segmentation or feature extraction or combining shape, texture and color features which results in moderate accuracy on benchmark datasets. Although some feature extraction techniques combining global and local feature descriptors reaches state of the art accuracy in classifying flowers, still there is a need for a robust and efficient system to automatically identify and recognize flower species at a larger scale in complex environment. Saitoh and Kaneko [1] proposed a method to recognize flowers, where two images are needed, one of the flower and other of the leaf. This method requires the user to place a black cloth behind the flower to recognize it. This is not feasible and is inconvenient for the user to use this method in real time scenario. Some of the modern plant recognition systems namely Leafsnap [2], Pl@ntNet [3], ReVes [4] and CLOVER [5] are all based on leaf identification which requires domain knowledge of flowers. A methodology that combines morphological features such as aspect ratio, eccentricity, rectangularity and Moving Median Center (MMC) hypersphere classifier was proposed by J.-X. Du et al [6]. A novel approach to recognize and identify plants using shape, color and texture features combined with Zernike moments with Radial Basis Probabilistic Neural Networks (RBPNN) was proposed by Kulkarni et al [7]. A flower classification approach based on vocabulary of texture, color and shape features was proposed by Zisserman and tested on 103 classes [8]-[9]. To accurately recognize flowers in images, Salahuddin et al. proposed a segmentation approach that uses color clustering and domain knowledge of flowers [10]. Although numerous algorithms and methodologies have been proposed and implemented to recognize flowers and plants, they still seem to be quite difficult to analyze due to their complex 3D structure and high intra-class variation.

II. GLOBAL FEATURE DESCRIPTORS

When it comes to quantify flower images, three most important attributes to be considered are Color, Texture and Shape.

2.1 Color

The first important feature to be considered to recognize flower species is “Color”. One of the most reliable and simple global feature descriptor is the Color Histogram which computes the frequency of pixel intensities occurring in an image. This enables the descriptor to learn about the distribution of each color in an image. The feature vector is taken by concatenating the count for each color. For example, if a histogram of 8-bins per channel is taken into consideration, then the resulting feature vector will be of 8x8x8 = 512-dfeature vector. In addition to it, simple color channel statistics such as mean and standard deviation could also be calculated to find the color distribution in an image.

978-1-5090-4740-6/17/$31.00

Color characteristics of an image alone is not sufficient to quantify flowers because in a multi-species environment, two or more species could be of same color. As an example, Sunflower and Daffodil will have similar color content.

2.2 Texture

Another important feature to be considered to recognize flower species is the "texture" or consistency of patterns and colors in an image. Haralick textures, which uses the concept of Gray Level Co-occurrence Matrix (GLCM) is commonly used as the texture descriptor. Haralick et al. [11] described about the 14 statistical features that could be computed to quantify the image based on texture. The resulting feature vector will be 13-d feature vector ignoring the 14^{th} dimension due to high computational time requirement.

2.3 Shape

When it comes to natural objects such as flowers, plants, trees etc. another important feature to quantify such objects in an image is the "shape". In computer vision research, Hu moments and Zernike moments are the two widely used global shape descriptors.

Moments rely on the statistical expectations of a random variable. There are seven such moments which are collectively called as Hu moments. First moment is the mean, followed by variance, standard deviation, skew, kurtosis and other statistical parameters. These seven moments are combined to form a feature vector of size 7-d.

Zernike moments are based on orthogonal functions and it was introduced by Teague as a shape descriptor. Similar to Hu moments, Zernike moments are also used to quantify the shape of an object in the image.

2.4 Local Binary Patterns (LBPs)

Similar to the Haralick textures, LBPs are also used to quantify the image based on "texture". Using LBPs, fine-grained details in the image are given importance. The main difference from Haralick textures is that LBPs processes pixels locally using the concept of neighborhood. The input grayscale image is divided into cells. For every pixel in a cell, based on its neighborhood (assumed to be of size "r"), an LBP value (decimal) is calculated by simple thresholding along the neighborhood. After calculating LBP values for all the pixels in a cell, a histogram with 256 bins is computed. This histogram is further normalized and concatenated for all the other cells in the image. If the number of points along the neighborhood is chosen as 9 and the radius is chosen as 3, then the resulting LBP feature vector size is of 11–d. The extended concept of uniform patterns in LBP was proposed to reduce the size of resulting feature vector for computational purposes. An LBP is said to be uniform, if the binary pattern computed for a pixel in a cell has at most two 0-1 and 1-0 transitions. For example, 1000000 has 2 transitions and so it is said to be uniform, whereas 01101001 has 5 transitions and is not uniform.

2.5 Histogram of Oriented Gradients (HOG)

Another widely used global descriptor in object detection community is the HOG descriptor. It uses the concept of counting the occurrences of gradient orientations in specific localized regions in an image. HOG descriptor computes these occurrences of gradient orientations on a dense grid having uniformly spaced cells with overlapping local contrast normalization.

2.6 Segmentation

Training images for global feature extraction are segmented initially using grabcut segmentation algorithm and the masks undergo bit-wise AND operation with the original image. This is to ensure that only the foreground flowers are considered for feature extraction instead of the cluttered background. Figure 1 shows one such image segmented using grabcut algorithm.

Fig1. Grabcut segmentation prior to feature extraction

2.7 Concatenating Global Features

Using only one global feature descriptor to quantify the entire dataset yields poor accuracy. Instead, different global feature vectors are concatenated for each image and then trained over a machine learning classifier. This type of concatenating feature vectors have some caveats because the feature vector sizes are different, and thus one feature vector might dominate other, thereby reducing the overall accuracy.

A comparative study on different types of global feature descriptors with the proposed system is performed to evaluate the impact of the latter. Local feature descriptors such as Scale Invariant Feature Transform (SIFT), Speeded Up Robust Features (SURF), Oriented FAST and Rotated BRIEF (ORB) etc. along with Bag of Visual Words is the popular choice among researchers in image classification challenges as these descriptors quantify the image locally and the resulting feature vector has multiple features that represents the entire image. Maria-Elena Nilsback and Andrew Zisserman [9] has investigated a feature combination technique that involves SIFT, Hue-Saturation-Value (HSV) and HOG features yielding an accuracy of 72.8% for FLOWER102 dataset. Although it yields a higher accuracy than other methods, much higher accuracies can be obtained using Deep Convolutional Neural Networks without feature engineering the entire dataset. Thus, the objective of this research work is to investigate the impact of transfer learning on OverFeat network to accurately recognize flower species using an image captured from the user's smartphone without using handcrafted features.

978-1-5090-4740-6/17/$31.00 ©2017 IEEE

III. Deep Learning Using CNN

3.1 Convolutional Neural Networks (CNN)

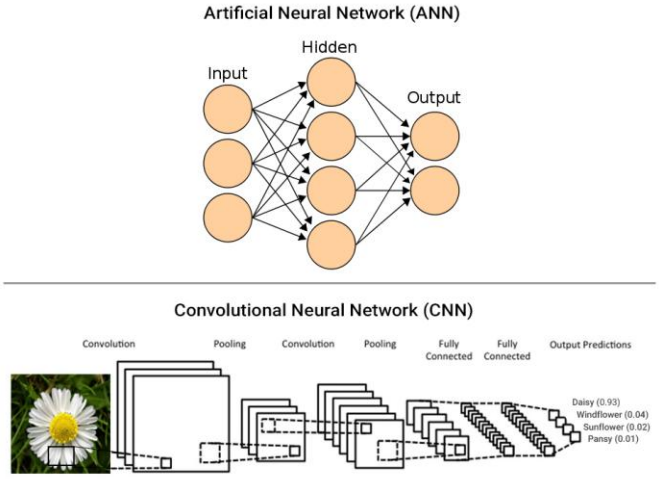

Fig.2 Artificial Neural Networks (ANN) and Convolutional Neural Networks (CNN)

In Deep Learning research, CNNs are specifically applied for Computer Vision applications that involves Image Classification and Object Recognition. Flower Species Recognition is a combination of both Object Recognition and Image Classification, as the system must detect a flower in the image as well as recognize which species it belongs to. To recognize the flower species, an intelligent system must be trained with larger set of images, so that it could predict the flower species from its learned patterns. This approach is termed as "Supervised Learning" which requires an external dataset of images with labels to predict the label of an unseen image. This research work uses Convolutional Neural Networks (CNN) along with Transfer Learning as the intelligent algorithm to efficiently recognize flower species in real-time. The major difference between a traditional Artificial Neural Network (ANN) and CNN is that only the last layer of a CNN is fully connected whereas in ANN, each neuron is connected to every other neurons as shown in Fig.2.

ANNs are not suitable for images because these networks leads to over-fitting easily due to the size of the images. Consider an image of size [32x32x3]. If this image is to be passed into an ANN, it must be flattened into a vector of 32x32x3 = 3072 rows. Thus, the ANN must have 3072 weights in its first layer to receive this input vector. For larger images, say [300x300x3], it results in a complex vector (270,000 weights), which requires a more powerful processor to process.

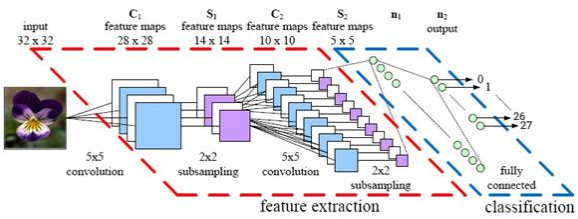

Fig 3. A Convolutional Neural Network (CNN) that shows three types of layers with different hyperparameters

CNNs consists of a stack of layers that takes in an input image, perform a mathematical operation (non-linear activation function such as ReLU, tanh) and predicts the class or label probabilities at the output. Instead of using standard handcrafted feature extraction methods, CNNs takes in the raw pixel intensity of the input image as a flattened vector. For example, a [30x30] color image will be passed as a 3-dimensional matrix to the input layer of CNN. CNN automatically learns complex features present in the image using the different layers which has "learnable" filters and combines the results of these filters to predict the class or label probabilities of the input image. Unlike an ANN, the neurons in a CNN layer are not connected to all other neurons, but connected only to a small region of neurons in the previous layer. The first layer might detect the lowest level features such as corners and edges in the image. The next subsequent layers might detect middle level features such as shapes and textures, and finally higher level features such as structure of the plant or flower will be detected by higher layers in the network. This unique technique of building up from lower level features to higher level features in an image is what makes CNNs most useful in many applications.

A Convolutional Neural Network (CNN) has three types of layers as follows –

1. Convolutional Layer (CONV)
2. Pooling Layer (POOL)
3. Fully-Connected Layer (FC)

3.1.1 Convolutional Layer (CONV)

This is the most important layer in any CNN architecture because this is the layer where CNN applies filters to learn features from the input image. This layer consists of filters and feature maps. The CONV layer contains M filters which are small in size (for example [3x3] or [5x5]). These filters are convolved with the input volume where it learns the features at a given spatial location, instead of going through the entire image. For every learnable filter, a 2-d feature map is formed as the filter slides through the width and height of the network, computing dot products with its entries and the input. Once all the M filters are applied to the input volume, all the corresponding 2-d feature maps are combined to produce the final output volume. This output volume has entries from the filters that looked at only a specific spatial region in the input image. For example, input image with a size and filter count of 64x64x3 and 12, respectively, is used to obtain an output volume of 64x64x12. These filters would have learnt the edges or corners present in the input image and would get activated only if they see those similar edges and corners again.

To get deeper into CONV Layer, each neuron in this layer will only connect to a smaller region in the input volume. This smaller region or the extent to which it gets connected is called as the Receptive Field or the filter size of the neuron. For example, if the input volume is of size [64x64x3], then using a Receptive Field of size [5x5], each neuron in the CONV layer will have 5x5x3 = 75 connections to specific spatial regions in the input volume. Apart from this

978-1-5090-4740-6/17/$31.00 ©2017 IEEE

connectivity of neurons, there are three hyperparameters that are required to tune the CONV layer. They are depth, stride and zero-padding.

1. Depth refers to the filter count (number of filters) of the network which is responsible to learn the lowest level features i.e. edges, corners etc.
2. Stride refers to how many jumps the neuron must take before it chooses the local region in the previous layer's volume.
3. Zero-padding is used to add zeros around the border of the input volume, if it is needed.

Thus, in a CONV layer, the input volume is represented as $[W_1 x\ H_1 x\ D_1]$ corresponding to the spatial dimensions of the input image, four hyperparameters are represented as [K, F, S, P] corresponding to the number of filters, the receptive field or the size of the filter, the stride and the amount of zero padding, and the output volume is represented as $[W_2 \times H_2 \times D_2]$ corresponding to –

$$W_2 = (W_1 - F + 2P) / S + 1 \quad (3.1.1.1)$$
$$H_2 = (H_1 - F + 2P) / S + 1 \quad (3.1.1.2)$$
$$D_2 = K \quad (3.1.1.3)$$

3.1.2 Pooling Layer (POOL)

This layer is used as an intermediate layer in the network where it downsamples or compresses the incoming volume along the spatial dimensions. For example, if the input volume is [64x64x12], its downsampled volume would be [32x32x12]. Thus, it downsamples the previous layer's feature maps obtained from different filters to reduce over-fitting and computations in the network. Thus, in a POOL layer, the input volume is represented as $[W_1 x\ H_1 x\ D_1]$ corresponding to the spatial dimensions of the input volume, two hyperparameters are represented as [F, S] corresponding to the receptive field or size of the filter and the stride, and the output volume is represented as $[W_2 \times H_2 \times D_2]$ corresponding to -

$$W_2 = (W_1 - F) / S + 1 \quad (3.1.2.1)$$
$$H_2 = (H_1 - F) / S + 1 \quad (3.1.2.2)$$
$$D_2 = D_1 \quad (3.1.2.3)$$

3.1.3 Fully Connected Layer (FC)

As in ANNs, the FC layer in a CNN has neurons that are fully connected to the neurons in the previous layer. This FC layer is often kept as the final layer of a CNN with "SOFTMAX" as its activation function for multi-class classification problems. The FC layer is responsible to predict the final class or label of the input image. Thus, it has an output dimension of [1x1xN] where N denotes the number of classes or labels considered for classification.

3.1.4 Popular CNN architectures

LeNet, AlexNet, GoogLeNet, VGGNet are the most popular CNN architectures used in current state-of-the-art Deep Learning research to solve various Computer Vision problems such as Image Classification, Object Recognition, Self-driving cars, Speech Recognition etc.

3.2 Dataset

On investigating the datasets available in the internet for flowers, FLOWERS17 dataset and FLOWERS102 dataset from the Visual Geometry group at University of Oxford are the challenging datasets available to benchmark algorithms. This is mainly because of the high variations in scale, pose and light conditions in the images of the dataset. The dataset also has high intra-class variation as well as inter-class variation. Thus, it is a harder Image Classification problem for a Computer Vision system that uses only handcrafted feature extraction techniques such as HOG, LBPs and Bag of Visual Words with Color Channel statistics, Color Histograms, Scale Invariant Feature Transform (SIFT), Speeded Up Robust Features (SURF) etc. In this research work, a few more flower species classes are added to the FLOWERS17 dataset and we named it as FLOWERS28 dataset (28 classes of flower species) as shown in Fig 4. The FLOWER28 dataset has 2240 images of flowers that belongs to 28 categories. Each category has 80 images. The FLOWERS102 dataset has 8189 images of flowers belonging to 102 categories. Each category has different number of images.

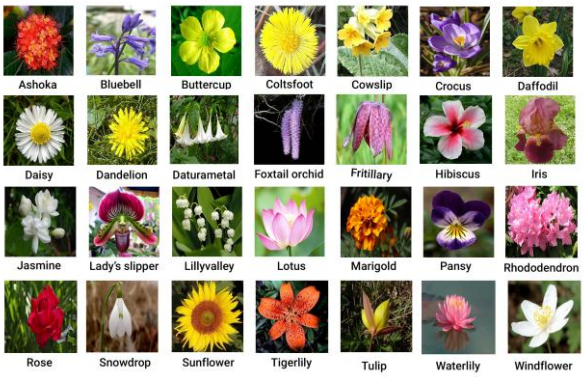

Fig 4. FLOWERS28 dataset used to train the flower recognition system

IV. Transfer Learning

4.1 CNN as a Feature Extractor

Deep Learning is used in situations where a large amount of data needs to be trained using GPUs. This is mainly because of the larger number of iterations or epochs required during training of a neural network as well as due to the fact that images, which are computationally intensive 3-d data (i.e. it has width, height and channels) are involved in processing.

TABLE 1
OVERFEAT NETWORK ARCHITECTURE

Layer	Stage	# filters	Filter size	Conv. stride	Pooling size	Pooling stride	Spatial input size
1	conv + max	96	11x11	4x4	2x2	2x2	231x231
2	conv + max	256	5x5	1x1	2x2	2x2	24x24
3	conv	512	3x3	1x1	-	-	12x12
4	conv	1024	3x3	1x1	-	-	12x12
5	conv + max	1024	3x3	1x1	2x2	2x2	12x12
6	full	3072	-	-	-	-	6x6
7	full	4096	-	-	-	-	1x1
8	full	1000	-	-	-	-	1x1

978-1-5090-4740-6/17/$31.00 ©2017 IEEE

So, instead of training a CNN from scratch with large number of images per class, a unique method called "Transfer Learning" is used where a pre-trained network on a very large dataset (ImageNet challenge) such as OverFeat, Inception-v3, Xception are taken as a feature extractor by keeping all the pre-trained layers except the last Fully Connected (FC) layer. One such pre-trained model namely the OverFeat network is considered. To compare the results of OverFeat network on other deep architectures, GoogLeNet Inception-v3architectureby Szegedy et al. [13] and François Chollet's Xception architecture [14] are also considered for evaluation.

4.2 OverFeat Network

The OverFeat network was introduced and trained by Sermanet et al. [12] on ImageNet 2012 training set that contains 1.2 million images over 1000 classes. The network architecture shown in Table 1 consists of 8 layers with ReLU non-linear activation applied after each Convolution Layer and Fully Connected layer, respectively. In this architecture, the filter size or the receptive field size is larger at the start of the network and gradually decreases along the subsequent layers. In addition, the number of filters starts off small and then gets increased in higher level layers of the network.

The input images from the FLOWERS28 dataset are resized to a fixed size of [231x231x3] and sent to the OverFeat network. The 1stlayer of neurons in the OverFeat network consists of CONV => RELU => POOL with M = 96 filters and receptive field of [11x11]. The 2ndlayer of neurons in the network consists of CONV => RELU => POOL with M = 256 filters and receptive field of [5x5]. The 3rd and 4thlayer in the network consists of CONV => RELU => CONV => RELU with M = 512 and 1024, respectively and receptive field of [3x3]. The 5thlayer consists of CONV => RELU => POOL with receptive field of [3x3] and M = 1024. The 6th and 7th layers are Fully Connected layers followed by a SOFTMAX classifier which gives the output predicted class probabilities.

V. PROPOSED SYSTEM

5.1 System Overview

Figure 5 shows the proposed system overview. The user captures a flower image using smartphone (assuming the flower as the only object in foreground with some random background).

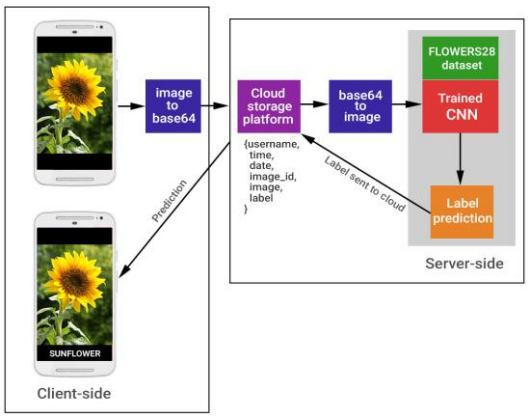

Fig 5. The Proposed System

The captured image is then converted to a base64 string format. It is then sent to a cloud storage platform called "Firebase" where it gets stored in a JSON format (username, time, date, image_id and image). At the server side, the trained CNN system on FLOWERS28 dataset receives the recent flower image in base64 format and converts it into a standard matrix form for processing. The converted image is sent to the CNN where its output class label is predicted. After prediction, the label name is sent to the same username with same image_id, from where the smartphone receives an automated response of the flower name from the cloud storage platform. This entire process of acquiring an image and getting the predicted label of the image takes around 1s tested with Moto G3 android smartphone running with a snapdragon quad-core processor and 2 GB of RAM.

5.2 Software requirement

Two programming languages are used to build the proposed system. Java is required to develop the Android app which captures the picture of the flower. Python 2.7is used to build the CNN subsystem. The transfer learning from OverFeat network is developed using sklearn-theano library provided under Open source, commercially usable-BSD license. Simple, modular and efficient deep learning python library called "Keras" is used to implement Inception-v3 and Xception architectures whose model and weights pre-trained on ImageNet are publicly available. Other python libraries required are NumPy, SciPy, matplotlib, cPickle and h5py. As there are2240 images in the dataset and each image is resized to [231x231] with 3 channels, the resulting feature matrix will be of size [2240x4096] which is stored locally in HDF5 file format using python's h5py library. The cloud storage platform used to store the base64 format of input image is called "Firebase" which is owned by Google.

VI. RESULTS AND DISCUSSION

6.1 Flower Species Recognition

The overall flower species recognition problem is divided into three parts. Firstly, features from the images in the training dataset are extracted using OverFeat CNN network (from FC-4096 i.e. 7th layer) and indexed into a HDF5 file format. Secondly, the network is trained using various machine learning classifiers such as Bagging trees, Linear Discriminant Analysis, Gaussian Naïve Bayes, K-Nearest Neighbor, Logistic Regression, Decision Trees, Random Forests and Stochastic Gradient Boosting. Finally, random test images are given to the network for label prediction to evaluate the accuracy of the system. It is observed that the system correctly identifies flower species with a Rank-1 accuracy of 82.32% and Rank-5 accuracy of 97.5% using Logistic Regression as the machine learning classifier on FLOWERS28 dataset.

The FLOWERS28 dataset is split into 1680 training images and 560 testing images. Figure 6 shows the accuracy obtained by training different machine learning classifiers on the CNN extracted features from the training images. It could be noted that bagging trees, logistic regression and random forests achieves a Rank-5 accuracy of 92.14%, 97.5% and 94.82%, respectively.

978-1-5090-4740-6/17/$31.00 ©2017 IEEE

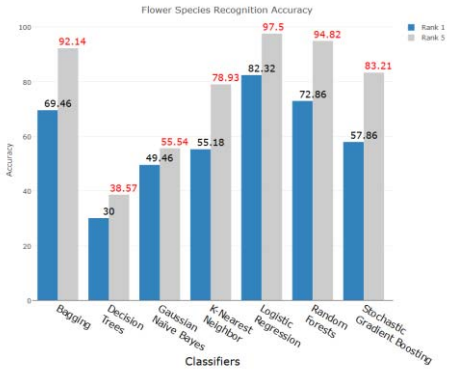

Fig 6. Accuracy of different machine learning classifiers on CNN extracted features (FLOWERS28 dataset)

Different global feature descriptors are applied on the training dataset and evaluated using random forests classifier which consistently outperformed all the other classifiers. Figure 7 shows the comparison between global feature descriptors and the proposed system applied on FLOWERS28 dataset.

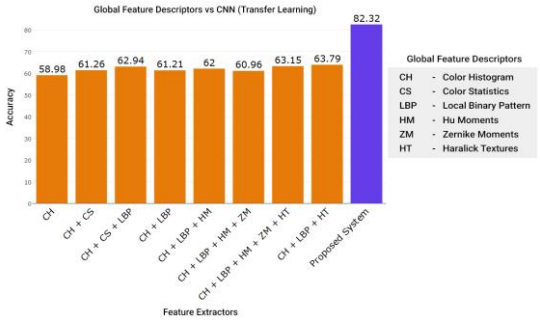

Fig 7. Global Feature Descriptors vs proposed system with CNN as Feature Extractor (FLOWERS28 dataset)

The proposed system with OverFeat network as the feature extractor is also tested with FLOWERS102 dataset, which is even more challenging to classify mainly because of the less number of training images provided. It is able to achieve a Rank-1 accuracy of 73.05% and Rank-5 accuracy of 90.58%. The recognition report of using deep architectures such as Inception-v3 and Xception as feature extractor on FLOWERS28 and FLOWERS102 dataset is shown in Table 2.

TABLE 2
RECOGNITION REPORT OF USING DEEP CNN ARCHITECTURES

Dataset	Model	Rank-1 accuracy	Rank-5 accuracy
FLOWERS28	Inception-v3	92.41%	98.66%
	Xception	90.18%	98.66%
	OverFeat	85.71%	99.11%
FLOWERS102	Inception-v3	93.41%	97.68%
	Xception	90.60%	96.58%
	OverFeat	73.05%	90.58%

From Table 2, it is clear that using very deep CNN architectures could increase the recognition accuracy by a greater amount.

V FUTURE WORK

The proposed work is a faster way to train a Convolutional Neural Network (CNN) with a smaller dataset and limited computational resource such as CPU. As there are millions of flower species around the world, this system could easily be adapted by training more number of flower species images to recognize different species around the world. Thus, the future work would be to construct a larger database with not only flower images, but also with leaves, fruits, bark etc., collected from different sources around different parts of the world. This system would also be useful to identify plants for medicinal purposes such as in case of first aid. The user can quickly take an image of the plant species and get information about it to decide whether or not it can be used for first aid. The crucial part in building such a system is the training dataset which needs to be prepared either by manually taking pictures of the plants around the city or by using public datasets.

REFERENCES

[1] Saitoh, T.; Kaneko, T., "Automatic recognition of wild Flowers", Pattern Recognition, Proceedings. 15th International Conference on, vol.2, no., pp.507-510 vol.2, 2000.

[2] Kumar, N., Belhumeur, P.N., Biswas, A., Jacobs, D.W., Kress, W.J., Lopez, I.C., Soares, J.V.B. "Leafsnap: A computer vision system for automatic plant species identification", European Conference on Computer Vision. pp. 502-516, 2012.

[3] D. Barthelemy. "The pl@ntnet project: A computational plant identification and collaborative information system", Technical report, XIII World Forestry Congress,2009.

[4] G. Cerutti, V. Antoine, L. Tougne, J. Mille, L. Valet, D. Coquin, and A. Vacavant, "Reves participation-tree species classification using random forests and botanical features", in Conference and Labs of the Evaluation Forum, 2012.

[5] Y. Nam, E. Hwang, and D. Kim, "Clover: A mobile content-based leaf image retrieval system", In Digital Libraries: Implementing Strategies and Sharing Experiences, Lecture Notes in Computer Science, pages 139-148, 2005.

[6] J.-X. Du, X.-F. Wang and G.-J. Zhang, "Leaf shape based plant species recognition", Applied Mathematics and Computation, vol. 185, 2007.

[7] A. H. Kulkarni, H. M. Rai, K. A. Jahagirdar and P. S. Upparamani, "A Leaf Recognition Technique for Plant Classification Using RBPNN and Zernike Moments", International Journal of Advanced Research in Computer and Communication Engineering, Vol. 2, Issue 1, pp. 984-988, 2013.

[8] Nilsback and Andrew Zisserman, "A Visual Vocabulary for Flower Classification", Computer Vision and Pattern Recognition, IEEE Computer Society Conference on. Vol.2, 2006.

[9] Nilsback, M.E., Zisserman, A,"Automated flower classification over a large number of classes", Indian Conference on Computer Vision, Graphics and Image Processing. pp. 722-729, 2008.

[10] Siraj, Fadzilah, Muhammad Ashraq Salahuddin, and Shahrul Azmi Mohd Yusof, "Digital Image Classification for Malaysian Blooming Flower", Computational Intelligence, Modelling and Simulation (CIMSiM), IEEE, 2010.

[11] Robert M. Haralick, K. Shanmugam, Its'hak Dinstein, "Textural Features for Image Classification", IEEE Transactions on Systems, Man and Cybernetics, Vol.SMC-3, No. 6, November 1973, pp.610-621, 1973.

[12] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks", ICLR, 2014.

[13] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going Deeper with Convolutions", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

[14] François Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions", arXiv:1610.02357 [cs.CV], 2016.

978-1-5090-4740-6/17/$31.00 ©2017 IEEE