Comparing Local Descriptors and Bags of Visual Words to Deep Convolutional Neural Networks for Plant Recognition

Pawara, Pornntiwa; Okafor, Emmanuel; Surinta, Olarik; Schomaker, Lambertus; Wiering, Marco

Published in:
6th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2017)

Publication date:
2017

Comparing Local Descriptors and Bags of Visual Words to Deep Convolutional Neural Networks for Plant Recognition

Pornntiwa Pawara[1], Emmanuel Okafor[1], Olarik Surinta[2], Lambert Schomaker[1] and Marco Wiering[1]
[1]*Institute of Artificial Intelligence and Cognitive Engineering (ALICE), Nijenborgh 9, University of Groningen, Groningen, The Netherlands*
[2]*Multi-Agent Intelligent Simulation Laboratory (MISL), Mahasarakham University, Mahasarakham, Thailand*
{*p.pawara, e.okafor, l.r.b.schomaker, m.a.wiering*}*@rug.nl, olarik.s@msu.ac.th*

Keywords: Convolutional Neural Network, Deep Learning, Bags of Visual Words, Local Descriptor, Plant Classification.

Abstract: The use of machine learning and computer vision methods for recognizing different plants from images has attracted lots of attention from the community. This paper aims at comparing local feature descriptors and bags of visual words with different classifiers to deep convolutional neural networks (CNNs) on three plant datasets; AgrilPlant, LeafSnap, and Folio. To achieve this, we study the use of both scratch and fine-tuned versions of the GoogleNet and the AlexNet architectures and compare them to a local feature descriptor with *k*-nearest neighbors and the bag of visual words with the histogram of oriented gradients combined with either support vector machines and multi-layer perceptrons. The results shows that the deep CNN methods outperform the hand-crafted features. The CNN techniques can also learn well on a relatively small dataset, Folio.

1 INTRODUCTION

The machine learning and computer vision community aims to construct novel algorithms for object recognition and classification. Recently, different works have studied the application of these algorithms on plant datasets. Plant classification is considered a challenging problem because of the variety and the similarity of plants in nature.

Early approaches to plant classification have considered the use of local descriptors. Nilsback and Zisserman (2008) used a joint learning approach of multiple kernels of local feature descriptors, including the histogram of oriented gradients (HOG) and the Scale-invariant feature transform (SIFT), a color histogram with a support vector machine (SVM) classifier for the classification of a 103 flower category dataset. The study showed that the classification performance can be improved by combining multiple features in a suitable kernel framework. An extension on the study of local feature descriptors with the use of the HOG-based approach (Xiao et al., 2010) for leaf classification showed a superior performance over inner-distance shape context (IDSC) features on the Swedish leaf and ICL datasets. Latte et al. (2015) worked on crop field recognition using the gray level co-occurrence matrix (GLCM) and various color features with artificial neural networks (ANNs). The performance was significantly increased when combining both types of features.

Other studies have focused on the use of segmentation and morphological based methods for recognizing plants using leaf datasets. For instance, Markov random field segmentation (Nilsback and Zisserman, 2010), which is optimized by using graph cut, has been used on the 13 classes of flowers. Munisami et al. (2015) combined several features of convex hull, morphological, distance map, and color histogram with *k*-nearest neighbors (KNN) to classify different kinds of leafs and provided comparable accuracies with less computational time. Wang et al. (2014) proposed the combination of texture feature (intersected cortical model), and shape features (center distance sequence) with an SVM for classification of leaf images. Furthermore, on the use of segmentation based methods, Zhao et al. (2015) showed that using learned shape patterns with independent inner-distance shape context (I-IDSC) features can be adopted for classification of both local and global information from leaves. The authors suggested that recognizing leaves by pattern counting approach is more effective than by matching their shape features.

Recently, attention has been shifted to the use of deep convolutional neural networks (CNNs) for plant classification. Lee et al. (2015) presented a leaf-based plant classification using CNNs to automatically learn

the discriminative features. Grinblat et al. (2016) employed a 3-layer CNN for assessing the classification performance on three different legume species and they emphasised the relevance of vein patterns. The works of Mohanty et al. (2016) and Sladojevic et al. (2016) used the deep CNN architectures to work on plant disease detection by focusing on leaf image classification. Mohanty et al. (2016) compared the performance of two CNN architectures: AlexNet and GoogleNet, with different sizes of training and test sets. The authors also worked on three choices of image type - color images, gray scale images, and leaf segmented images. The results showed that the GoogleNet architectures steadily outperform AlexNet. Additionally, with the train-test set distribution of 80%-20%, the learning methods obtained the best results.

In this study, we compare the performance of local descriptors and the bag of visual words with different classifiers to deep CNN approaches on three datasets: a novel plant dataset (AgrilPlant) and two already existing datasets.

Contributions: In this paper, we compare seven different techniques and assess their performance for recognizing plants from images using three plant datasets; AgrilPlant, LeafSnap, and Folio. We created a novel dataset, AgrilPlant, which consists of 10 classes of agriculture plants. For the comparison study, we make use of both scratch and fine-tuned versions of the GoogleNet and AlexNet architectures and compare them to a local descriptor (HOG) with k-nearest neighbors (KNN) and a bag of visual words with the histogram of oriented gradients (HOG-BOW) combined with either a support vector machine (SVM) and multilayer perceptrons (MLP). Using many experiments with the various techniques, we show that the CNN based methods outperform the local descriptor and the bag of visual words techniques. We also show that the reduction of the number of neurons in the AlexNet architecture outperforms the original AlexNet architecture and gives a remarkable improvement in the computing time.

Paper Outline: The remaining parts of the paper are organized in the following way. Section 2 explains the deep CNN architectures and the reduction of the number of neurons in details. Section 3 entails brief discussions on the hand-crafted local descriptors. In section 4, we describe the plant datasets and the experimental settings. Section 5 presents and discusses the performance of the various techniques. The last section concludes and recommends possible areas for future work.

2 DEEP CONVOLUTIONAL NEURAL NETWORKS

Deep convolutional neural networks (CNNs) were first introduced by LeCun et al. (1989) and have become the most influential machine learning approach in the computer vision field.

A deep CNN architecture consists of several layers of various types. Generally, it starts with one or several convolutional layers, followed by one or more pooling layers, activation layers, and ends with one or a few fully connected layers.

There are usually a certain number of kernels in each convolutional layer which can output the same number of feature maps by sliding the kernels with a specific receptive field over the feature map of the previous layer (or the input image in the case of the first convolutional layer). Each feature map that is computed is characterized by several hyper-parameters: the size and depth of the filters, the stride between filters and the amount of zero-padding around the input feature map (Castelluccio et al., 2015).

Pooling layers can be applied in order to cope with translational variances as well as to reduce the size of feature maps (Sladojevic et al., 2016). They proceed by sliding a filter along the feature maps and outputting the maximum or average value, depending on the choices of pooling, in every sub-region.

A nonlinear layer or activation layer is conventionally applied to a feature map after each convolutional layer to introduce nonlinearity to the network. The Rectified Linear Unit (ReLU) function is a notable choice (Glorot et al., 2011; Couchot et al., 2016) because of the computational efficiency and the alleviation of the vanishing gradient problem. The ReLU basically converts the input to its positive value or zero otherwise, i.e. $\int(x) = max(0,x)$.

The fully connected layers typically are the last few layers of the architecture. The drop out technique can be applied to prevent overfitting (Srivastava et al., 2014; Yoo, 2015). The final fully connected layer in the architecture contains the same amount of output neurons as the number of classes to be recognized.

2.1 AlexNet Architecture

The AlexNet architecture (Krizhevsky et al., 2012) follows the pattern of the LeNet-5 architecture (LeCun et al., 1989). The original AlexNet contains eight weight layers, which consists of five convolutional layers and three fully connected layers.

The first two convolutional layers (conv$\{1,2\}$) are followed by a normalization and a max pooling layer. The last convolutional layer (conv5) is followed by the

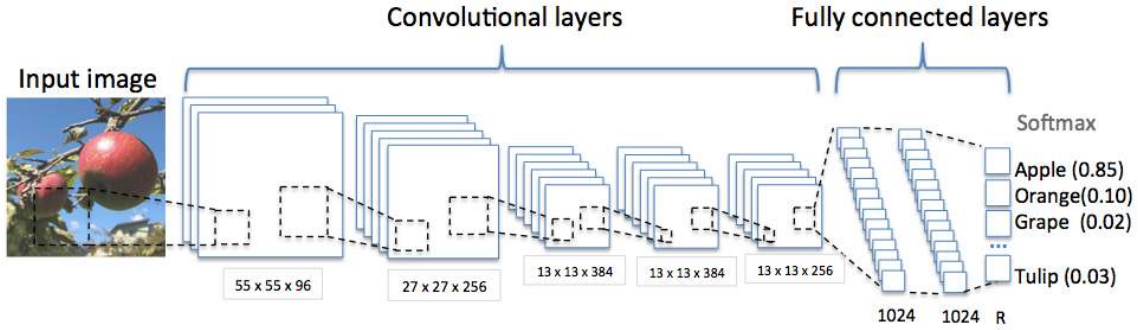

Figure 1: The AlexNet architecture used in our work. The number $w \times w \times d$ in each convolutional layer represents the size of the feature map for each layer. The fc6 and fc7 layers contain 1,024 neurons. R in the fc8 layer is the number of neurons, which represents the number of classes in each dataset, which are set to 10, 184, and 36 for the AgrilPlant, the LeafSnap, and the Folio dataset, respectively.

max pooling layer. Each of the sixth and seventh fully connected layers (fc$\{6,7\}$) contain 4,096 neurons. The final fully connected layer (fc8) contains 1,000 neurons because the ImageNet dataset has 1,000 classes to be classified. The ReLU activation function is applied to each of the first seven layers. A dropout ratio of 0.5 is applied to the fc6 and fc7 layers. The output from the fc8 layer is finally fed to a softmax function.

In our study, the original AlexNet architecture is adapted by reducing the number of neurons in the fc6 and fc7 layer from 4,096 neurons to either 256, 512, and 1,024 neurons in both layers. The idea behind this is to increase the computational performance and mitigate the risk of overfitting (Xing and Qiao, 2016). We performed preliminary experiments on the AgrilPlant dataset to choose the best number of neurons. The results of this experiment are shown in Table 1. It shows that 1,024 neurons are the most efficient in terms of accuracy and it provides 34% improvement in training time compared to 4,096 neurons. Consequently, we set the number of neurons in the fc6 and fc7 layers to 1,024 for all datasets. The AlexNet architecture used in our works is shown in Figure 1.

Table 1: Accuracy comparison among different numbers of neurons and time improvement compared against 4,096 neurons in the AlexNet architecture on the AgrilPlant dataset. The results are reported with test accuracies and standard deviations using five simulations.

Number of neurons	Accuracy	Time improvement (%)
4,096	88.30 ± 1.34	-
1,024	$\mathbf{89.53 \pm 0.61}$	34.06
512	89.13 ± 1.24	39.09
256	88.90 ± 1.35	41.08

2.2 GoogleNet Architecture

GoogleNet, presented in the work of Szegedy et al. (2015), is among the first architectures that introduced the inception module that greatly dropped off the large amount of trainable parameters in the network. The inception module uses a parallel combination of 1×1, 3×3, and 5×5 convolutions along with a pooling layer. Additionally, the 1×1 convolutional filter is added to the network before the 3×3, and 5×5 convolutions for dimensionality reduction as shown in Figure 2. This is called the "network in network" architecture (Lin et al., 2013).

The GoogleNet architecture uses 9 inception modules, containing 22 layers along with four max pooling layers, and one average pooling layer. The ReLU is used in all the convolutional layers, including those inside the inception modules. To deal with the problem of vanishing gradients in the network, inspired by the theoretical work by Arora et al. (2014), two auxiliary classifiers are added to the layers in the middle of the network during the training process (Yoo, 2015). A dropout ratio of 0.4 is applied to the softmax classifier. The illustration of the convolutional layers and the inception modules designed in GoogleNet is shown in Figure 2. A more detailed explaination along with all relevant parameters of the GoogleNet architecture can be found in the original paper (Szegedy et al., 2015).

3 CLASSICAL LOCAL DESCRIPTORS

3.1 Histogram of Oriented Gradients

The histogram of oriented gradients (HOG) was initially introduced for human detection (Dalal and Triggs, 2005). The HOG feature extractor represents objects by counting occurrences of gradient intensities and orientations in localized portions of an image. Based on the work of (Bertozzi et al., 2007; Surinta et al., 2015), the HOG descriptor computes feature vectors using the following steps:

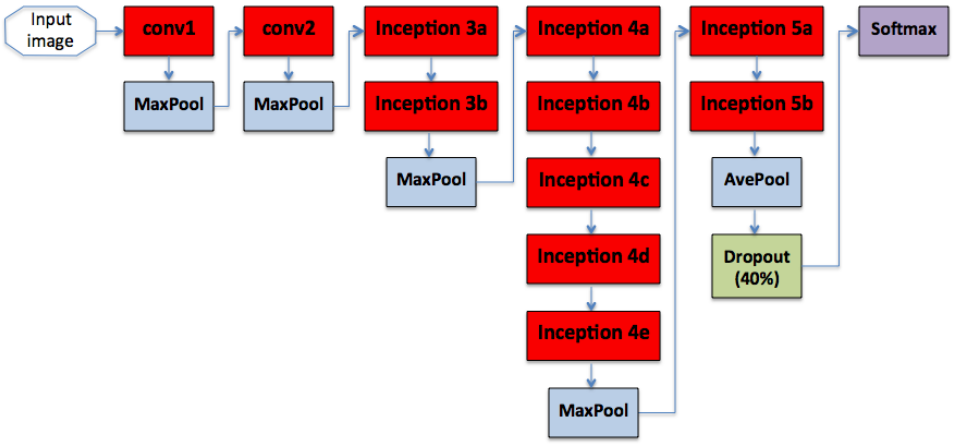

Figure 2: The illustration of the GoogleNet architecture (Szegedy et al., 2015). All convolutional layers and inception modules have a depth of two.

1) split the image into small blocks of $n \times n$ cells,

2) compute horizontal gradient H_x and vertical gradient H_y of the cells by applying the kernel [-1,0,1] as gradient detector,

3) compute the magnitude M and the orientation θ of the gradient as:

$$M_{(x,y)} = \sqrt{H_x^2 + H_y^2} \quad (1)$$

$$\theta_{(x,y)} = arctan\frac{H_y}{H_x} \quad (2)$$

4) form the histogram by weighing the gradient orientations of each cell into a specific orientation bin,

5) apply L2 normalization to the bins to reduce the illumination variability and obtain the final feature vectors.

In our preliminary experiments, we use 5×5 rectangular blocks and 8 orientation bins, thus yielding a 200-dimensional feature vector. We then feed the feature vector to the KNN classifier.

3.2 Bags of Visual Words with Histogram of Oriented Gradients

The idea of the bag of visual words (BOW) model (Csurka et al., 2004; Tsai, 2012) in computer vision is to consider an image consisting of different visual words. The image descriptor can be obtained by clustering features of local regions in the images, which contain rich local information of the images, such as color or texture. In the paper, we combine BOW with the HOG feature descriptor, resulting in HOG-BOW.

The construction of the HOG-BOW feature vectors involves the following steps:

1) To compute patches, the set of local region patches P is automatically extracted from the dataset of images, $P = \{p_1, p_2, ..., p_n, \}$, where n is the number of patches. The size of each patch is a square of $w \times w$ pixels. Each patch is computed by using local descriptors, and then used as an input to create a codebook.

2) The codebook C is obtained by applying the K-means clustering algorithm over the extracted feature vectors of each patch based on a number of centroids.

3) Construct the BOW feature by detecting the occurrences in the image of each cluster. Each image is split into four quadrants and we compute the feature activation using sum-pooling (Wang et al., 2013).

In our experiments, based on the work of Surinta et al. (2015), the HOG descriptor is employed as the local descriptor. The number of patches is set to 400,000, the size of each patch is 15×15 pixels, and the number of centroids is set to 600. As the image is split into four quadrants, the HOG-BOW generates 2,400 dimensional feature vectors.

The feature vectors are then fed to the classifiers, for which we use the L2-SVM (Suykens and Vandewalle, 1999) and a Multi-Layer Perceptron (MLP). The process of the HOG-BOW method used in our experiments is illustrated in Figure 3.

4 EXPERIMENTS

4.1 Plant Datasets

In our experiments, we performed experiments using three datasets; AgrilPlant, Leafsnap, and Folio.

AgriPlant Dataset: The AgriPlant dataset consists of 3,000 agriculture images that are collected from the website www.flickr.com. It consists of 10 classes with

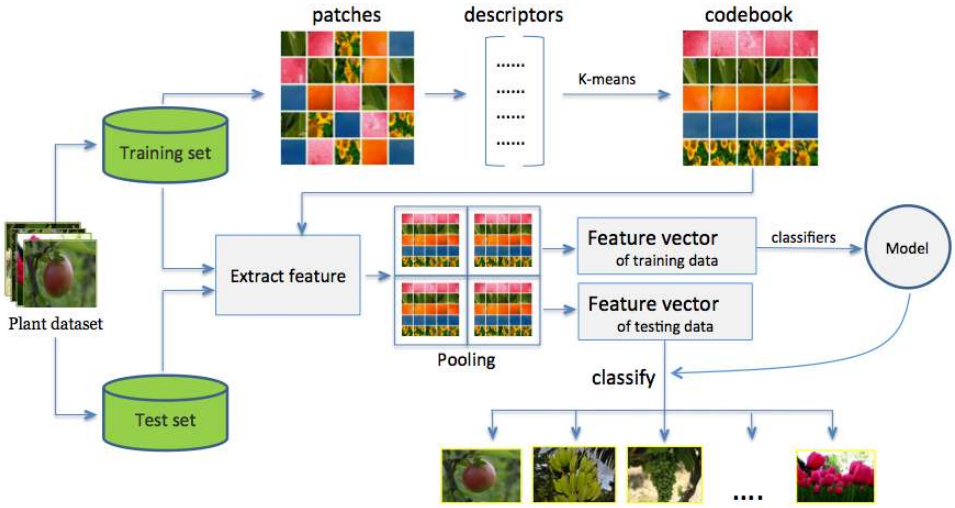

Figure 3: Illustration of generating the BOW feature vectors.

the following plants: apple, banana, grape, jackfruit, orange, papaya, persimmon, pineapple, sunflower, and tulip. Each class contains exactly 300 images. The images may have been taken from five different views, i.e. entire plant, branch, flower, fruit, and leaf. A sample of the AgrilPlant dataset is shown in Figure 4.

The challenges of classification on the AgriPlant dataset are (a) the similarity among some classes, i.e. apple, orange and persimmon have similar shapes and colors, (b) a diversity of plants within the same class, for example, there are green and red apples, or there are varieties of tulips, and (c) the existence of complex backgrounds or other objects such as human, car, and house on several images.

LeafSnap Dataset: The Leafsnap dataset (Kumar et al., 2012) originally contained 185 tree species and is used for leaf recognition research. The dataset consists of leaf images taken from two different sources; lab images and field images. In our experiments, we performed experiments with field images. This consists of 7,719 leaf images and has a coverage of 184 tree species (one class is missing for the field images) of the Northeastern United States. All the images were taken in outdoor environments with mobile devices and might contain some amounts of noise, blur, and shadows. The number of images in each class vary from 10 to 183 images. A sample of the LeafSnap dataset is shown in Figure 5(a).

Folio Dataset: The Folio dataset, introduced in the work of Munisami et al. (2015), consists of 32 different species of leaves which were collected from the farm at the University of Mauritius. It consists of approximately 20 images for each species. All images were taken under daylight on a white background. A sample of the Folio dataset is shown in Figure 5(b).

4.2 Experimental Settings

We evaluate the deep CNNs architectures and the hand-crafted local descriptors combined with KNN, SVM, and MLP for plant classification. In our study, the plant datasets are split into a training set and test set with the ratio of 80:20 and 5-fold cross validation is used to evaluate the performance of the studied methods. The resolution of plant images is set to 256×256 pixels.

Most parameters for the deep CNN architectures, for both AlexNet and GoogleNet, are set to the same values for scratch and fine-tuned versions, except for max iteration and step size that are set to different values. The parameters settings are shown in Table 2.

For the hand-crafted local descriptors, we combine the HOG with the KNN classifier and the HOG-BOW with MLP and SVM. We select the optimal k for the KNN classifier in the range of $k = \{3,5,7,9\}$.

On each dataset, a grid search is applied to tune the C parameters for the SVM in the range of $C = \{2^1, 2^2, ..., 2^8\}$ and choose the best C parameter that gives the highest accuracy result. We then perform the 5-fold cross validation using this C parameter.

For the MLP, we use the scaled conjugate gradient (Møller, 1993) as a training algorithm. The number of neurons and the learning rate are set to 512 and 0.001, respectively. These values resulted in the best performance using preliminary experiments.

5 RESULTS AND DISCUSSION

We now report the test accuracies using the deep CNN methods and hand-crafted local feature descriptors with different classifiers. The experiments are carried

Figure 4: Sample pictures from the AgrilPlant dataset. Note that, the images on each column represent one class. From left to right, the class is apple, banana, grape, jackfruit, orange, papaya, persimmon, pineapple, sunflower, and tulip.

(a) (b)

Figure 5: Sample pictures from two datasets (a) LeafSnap, and (b) Folio.

Table 2: Summary of experimental parameters for the AlexNet and GoogleNet architectures on the three datasets.

Parameters	AgrilPlant	LeafSnap	Folio
Learning rate	0.001	0.001	0.001
Weight decay	0.0005	0.0005	0.0005
Train batch size	20	20	20
Validation batch size	10	10	10
Max iteration (scratch)	50000	50000	50000
Step size (scratch)	25000	25000	25000
Max iteration (fine-tuned)	20000	20000	20000
Step size (fine-tuned)	10000	10000	10000
Test iterations of solver	30	77	6
Test iterations evaluation	60	154	12

out based on 5-fold cross validation and we report the top-1 accuracy. The results are shown in Table 3.

5.1 AgrilPlant Dataset Evaluation

Comparing the performance of the deep CNN methods and the hand-crafted local feature descriptors, the deep CNN methods consistently outperform the local descriptors. The fine-tuned approaches of both the GoogleNet and the AlexNet architectures obtain the best performance, reaching an accuracy of 98.33% and 96.37%, respectively. This is an improvement of approximately 5% and 6.8% over the scratch versions of each architecture. The GoogleNet fine-tuned version gives approximately 19% better performance than the HOG-BOW with SVM, which obtains the best performance among the local feature descriptors. The HOG-BOW with SVM outperforms the HOG-BOW with MLP with 4.8% difference. The HOG with KNN obtains the worst performance with an accuracy of 38.13%.

5.2 LeafSnap Dataset Evaluation

For the LeafSnap dataset, the GoogleNet fine-tuned and scratch versions obtain the best performance with an accuracy of 97.66%, and 89.62%, respectively. The AlexNet fine-tuned architecture follows up with an accuracy of 89.51%. The HOG-BOW with MLP, however, slightly outperforms the AlexNet scratch architecture with an accuracy of 79.27%. Comparing this to previous work on the LeafSnap dataset using curvature histograms, Kumar et al. (2012) reported a top-5 accuracy of 96.8%. We note that GoogleNet fine-tuned significantly outperforms that method with a top-1 accuracy of 97.66%. Comparing between the local feature descriptors, The HOG-BOW with MLP gives an accuracy of approximately 6.6% and 20.7% higher than the HOG-BOW with SVM and the HOG with KNN, respectively.

Table 3: Test Accuracy comparison among all techniques on three plant datasets.

Methods	AgrilPlant	LeafSnap	Folio
HOG with KNN	38.13 ± 0.53	58.51 ± 2.47	84.30 ± 1.62
HOG-BOW with MLP	74.63 ± 2.16	79.27 ± 3.36	92.37 ± 1.78
HOG-BOW with SVM	79.43 ± 1.68	72.63 ± 0.38	92.78 ± 2.17
AlexNet scratch	89.53 ± 0.61	76.67 ± 0.56	84.83 ± 2.85
AlexNet fine-tuned	96.37 ± 0.83	89.51 ± 0.75	$\mathbf{97.67 \pm 1.60}$
GoogleNet scratch	93.33 ± 1.24	89.62 ± 0.50	89.75 ± 1.74
GoogleNet fine-tuned	$\mathbf{98.33 \pm 0.51}$	$\mathbf{97.66 \pm 0.34}$	$\mathbf{97.63 \pm 1.84}$

5.3 Folio Dataset Evaluation

For the Folio dataset, Munisami et al. (2015) reported an accuracy of 87.3% by using shape features and a color histogram with KNN which outperforms the AlexNet scratch version on our study with an accuracy of 84.83%.

In our experiments, the AlexNet fine-tuned and the GoogleNet fine-tuned architectures obtain the best results with an accuracy of 97.67% and 97.63%, respectively. The next two techniques with the best performance are the HOG-BOW with SVM and the HOG-BOW with MLP classifiers, both of which yield an accuracy of 92.73% and 92.37%, respectively. The scratch version of GoogleNet still obtains acceptable results with an accuracy of 89.75%. Note that on this dataset, the HOG-BOW with either SVM and MLP classifiers gives roughly 8% better performance than the AlexNet scratch version. The HOG with KNN gives the worst result with an accuracy of 84.30%.

The evaluation on the Folio dataset shows that the deep CNN architectures also perform well on a small dataset as this dataset contain only 637 images in total for 32 classes.

6 CONCLUSIONS

In this paper, we have presented a comparative study of some classical feature descriptors to deep CNN approaches on three plant datasets. The HOG feature descriptor combined with KNN, and HOG-BOW combined with SVM and MLP classifiers are compared to AlexNet and GoogleNet, both trained from scratch and using the fine-tuned versions as deep CNN architectures.

We evaluated all the image recognition techniques on three plant datasets and achieved notable overall performances. The fine-tuned versions of the deep CNNs architectures persistently outperform the classical feature descriptors techniques on all datasets. The GoogleNet fine-tuned architecture obtains the best result with accuracies of 98.33% and 97.66% on the AgrilPlant dataset and the LeafSnap dataset, respectively. The AlexNet fine-tuned and the GoogleNet fine-tuned techniques also give the best result on a relatively small dataset, Folio, with an accuracy of approximately 97.6%.

Comparing between the HOG-BOW descriptors on each of the three dataset, on the AgrilPlant dataset, the HOG-BOW combined with SVM performs 4.8% better than the HOG-BOW combined with MLP. On the other hand, the HOG-BOW combined with MLP works 6.64% better than the HOG-BOW combined with SVM. On the Folio dataset, however, both HOG-BOW descriptors give insignificantly different results with an accuracy of approximately 92%. Among all studied techniques, the HOG with KNN always yields the worst accuracy on all datasets.

In further work, we want to study the deployment of deep learning in an unmanned aerial vehicle system targeted for precision identification of plant diseases.

REFERENCES

Arora, S., Bhaskara, A., Ge, R., and Ma, T. (2014). Provable bounds for learning some deep representations. In *Machine Learning (ICML'14), International Conference on*, pages 584–592.

Bertozzi, M., Broggi, A., Del Rose, M., Felisa, M., Rakotomamonjy, A., and Suard, F. (2007). A pedestrian detector using histograms of oriented gradients and a support vector machine classifier. In *IEEE Intelligent Transportation Systems Conference (ITSC'07)*, pages 143–148.

Castelluccio, M., Poggi, G., Sansone, C., and Verdoliva, L. (2015). Land use classification in remote sensing images by convolutional neural networks. *arXiv preprint arXiv:1508.00092*.

Couchot, J.-F., Couturier, R., Guyeux, C., and Salomon, M. (2016). Steganalysis via a convolutional neural network using large convolution filters. *arXiv preprint arXiv:1605.07946*.

Csurka, G., Dance, C., Fan, L., Willamowski, J., and Bray, C. (2004). Visual categorization with bags of keypoints. In *Computer Vision (ECCV'04), 8th European Conference on*, volume 1, pages 1–22.

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition (CVPR'05), IEEE Computer Society Conference on*, volume 1, pages 886–893.

Glorot, X., Bordes, A., and Bengio, Y. (2011). Deep sparse rectifier neural networks. *Journal of Machine Learning Research (JMLR)*, 15(106):275.

Grinblat, G. L., Uzal, L. C., Larese, M. G., and Granitto, P. M. (2016). Deep learning for plant identification using vein morphological patterns. *Computers and Electronics in Agriculture*, 127:418–424.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Kumar, N., Belhumeur, P. N., Biswas, A., Jacobs, D. W., Kress, W. J., Lopez, I. C., and Soares, J. V. (2012). Leafsnap: A computer vision system for automatic plant species identification. In *Computer Vision (ECCV'12), European Conference on*, pages 502–516. Springer.

Latte, M., Shidnal, S., Anami, B., and Kuligod, V. (2015). A combined color and texture features based methodology for recognition of crop field image. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, 8(2):287–302.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551.

Lee, S. H., Chan, C. S., Wilkin, P., and Remagnino, P. (2015). Deep-plant: Plant identification with convolutional neural networks. In *Image Processing (ICIP), 2015 IEEE International Conference on*, pages 452–456.

Lin, M., Chen, Q., and Yan, S. (2013). Network in network. *arXiv preprint arXiv:1312.4400*.

Mohanty, S. P., Hughes, D. P., and Salathé, M. (2016). Using deep learning for image-based plant disease detection. *CoRR*, abs/1604.03169.

Møller, M. F. (1993). A scaled conjugate gradient algorithm for fast supervised learning. *Neural networks*, 6(4):525–533.

Munisami, T., Ramsurn, M., Kishnah, S., and Pudaruth, S. (2015). Plant leaf recognition using shape features and colour histogram with K-nearest neighbour classifiers. *Computer Vision and the Internet (VisionNet'15), Second International Symposium on, Procedia Computer Science*, 58:740 – 747.

Nilsback, M.-E. and Zisserman, A. (2008). Automated flower classification over a large number of classes. In *Computer Vision, Graphics & Image Processing (ICVGIP'08), Sixth Indian Conference on*, pages 722–729. IEEE.

Nilsback, M.-E. and Zisserman, A. (2010). Delving deeper into the whorl of flower segmentation. *Image and Vision Computing*, 28(6):1049–1062.

Sladojevic, S., Arsenovic, M., Anderla, A., Culibrk, D., and Stefanovic, D. (2016). Deep neural networks based recognition of plant diseases by leaf image classification. *Computational Intelligence and Neuroscience*, 2016:1–11.

Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Surinta, O., Karaaba, M. F., Mishra, T. K., Schomaker, L. R., and Wiering, M. A. (2015). Recognizing handwritten characters with local descriptors and bags of visual words. In *Engineering Applications of Neural Networks*, pages 255–264. Springer.

Suykens, J. A. and Vandewalle, J. (1999). Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Computer Vision and Pattern Recognition (CVPR'15), the IEEE Conference on*.

Tsai, C.-F. (2012). Bag-of-words representation in image annotation: A review. *ISRN Artificial Intelligence*, 2012.

Wang, X., Wang, L., and Qiao, Y. (2013). A comparative study of encoding, pooling and normalization methods for action recognition. In Lee, K. M., Matsushita, Y., Rehg, J. M., and Hu, Z., editors, *Computer Vision (ACCV'12), 11th Asian Conference on*, pages 572–585. Springer Berlin Heidelberg.

Wang, Z., Sun, X., Ma, Y., Zhang, H., Ma, Y., Xie, W., and Zhang, Y. (2014). Plant recognition based on intersecting cortical model. In *Neural Networks (IJCNN'14), International Joint Conference on*, pages 975–980. IEEE.

Xiao, X.-Y., Hu, R., Zhang, S.-W., and Wang, X.-F. (2010). HOG-based approach for leaf classification. In *Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence*, pages 149–155. Springer.

Xing, L. and Qiao, Y. (2016). Deepwriter: A multi-stream deep CNN for text-independent writer identification. In *Frontiers in Handwriting Recognition (ICFHR'16), 15th International Conference on*, pages 1–6.

Yoo, H.-J. (2015). Deep convolution neural networks in computer vision. *IEIE Transactions on Smart Processing & Computing (IEIE SPC'15')*, 4(1):35–43.

Zhao, C., Chan, S. S., Cham, W.-K., and Chu, L. (2015). Plant identification using leaf shapes – A pattern counting approach. *Pattern Recognition*, 48(10):3203–3215.