Plant Leaf Recognition

Albert Liu, albertpl@stanford.edu Yangming Huang, yangming@standford

I. Introduction

Automatic plant species recognition with image processing has application in weeds identification, species discovery, plant taxonomy, natural reserve park management and so on [3]. It is considered a fine-grained image recognition problem and hard to solve because

1) The subtle differences between different species in the same class. Sometime such fine differences can be even challenging to human experts.
2) Typically this requires large training data but it is not feasible due to the number of species (over 220000 [2]).

In this report, we describe our exploration with this problem, using traditional handcrafted features and features extracted from pretrained deep convolution neural network (ConvNets). The input to our system is raw images from various datasets and the output is the label for each species.

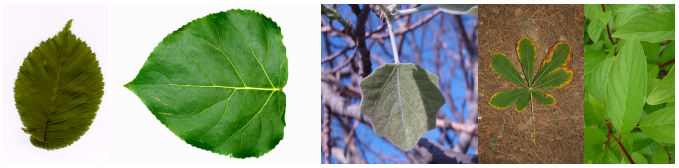
Fig. 1. Examples of species with fine differences

II. Related Work

Research on automatic leaf classification has been active since 2000. Lots of hand-crafted features have been proposed, ranging from shape based, to statistical texture and margin related [2] [3] [1]. Also generic computer vision object recognition features, such SIFT[32] and HOG[33], are studied for this problem. Most of such manually engineered features achieve excellent accuracy on clean images taken in controlled conditions, which consist of one single well aligned leave on contrasting background, such as those images in data set [15].

Recently, with the huge success of deep ConvNets, particularly from the winners of ILSVRC [11] [23], [24] and [22], researchers start to apply deep ConvNets to this problem. In [36], Sharif, et al. suggested that generic features can be extracted from large ConvNet and yield very good results on fine-grained classification problems even without fine-tuning the pre-trained model. [34] compares traditional approaches with ConvNet based approach, and discuss impacts on various conditions, including translation, rotation, scale and occlusion. However it only give results on Flavia[16] dataset which is relatively less challengingIII. [35] proposes a VGG[23] based architecture, and use multiple organ features. It produces above 70% mean average precision on ImageCLEF dataset[17].

III. DataSet



From left to right, Flavia, Swedish, natural, uniform, natural

We consider two types of dataset.

1) Swedish[15] and Flavia[16]. Both contain clean images only, which is characterized with well aligned leaf on contrasting background, with little or no variations of luminance or color.

2) ImageCLEF [17], which is collected through crowd sourced application, is a much more noisy datasets with considerable variations on lighting conditions, viewpoints, background clutters and even occlusions. The dataset can be further split into two subset: uniform, which is taken in a more controlled environment, and natural, which is taken in a natural environment.

Name	Species	samples
uniform	66	9607 (train) 1194 (test)
natural	57	2585 (train) 521 (test)
Swedish	15	75 / species
Flavia	33	$\sim$ 60 / species

IV. Approach

A. Overview

Here is the pipeline of our system.

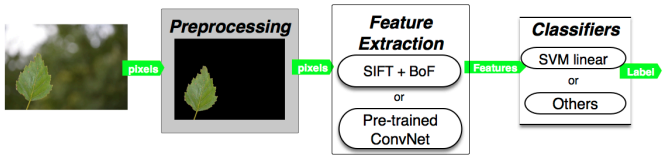


Fig. 2. Overview of the system

1) Firstly, during preprocessing, we apply CLAHE[30] to reduce lighting condition variation and then resize raw images to fit the next layer. We also attempt to remove background with the following techniques in a heuristic way
   - use K-means (K=2) and throw away background
   - find the convex hull containing the largest N contours and then use GrabCut[29] to segment leaf out of the background clutter.
2) Next we extract features
   - ConvNets

We take transfer learning approach, in order to make use of the power of deep ConvNets under the constrains of time and computations. Specifically, we take a couple of ConvNets that are pre-trained on ImageNet for ILSVRC object classification task, remove top FC layers and then treat the rest of the ConvNet as fixed feature extractor. The CNN codes are our features. To battle overfitting issue, we augment our input images with color channel shift, translation and rotation.

- Traditional SIFT + Bag of Features
  Key points are densely sampled and SIFT feature descriptors are retrieved at each key point. We fix size of the codebook (K) as 1000/300 for different data sets.

3) Finally, we train a simple classifier from the feature vectors and then predict labels for our test data. For linear SVM, we optimize as follows

$$\begin{aligned} min_{\gamma,w,b} \quad & \frac{1}{2}||w||^2 + C\sum_{i=1}^{m}\xi_i \\ s.t. \quad & y^{(i)}(w^\intercal x^{(i)} + b) \geq 1 - \xi_i \\ & \xi_i \geq 0, i = 1, \ldots, m \end{aligned}$$

B. ConvNets approach

Setup:

For ConvNets, we used Keras framework [18] with TensorFlow [19] backend of GPU support: NVIDIA GeForce GT 750M 2048 MB. As an alternative, we also run on CPU given the limited graphic memory of our GPU which is crucial for a deeper ConvNet architecture.

Exploration:

We started by training ConvNet classifier from scratch, following the guidelines below:

1) Convolutional layer learning features from general to specific, giving more layers helps with the transition. Study on deep convolutional nets suggests that deeper models are preferred under a parameter budget[27].
2) Dropout reduces overfitting [28] [11]
3) Use aggressive pooling to reduce the dimensionality.

We designed our ConvNet with the architecture below:

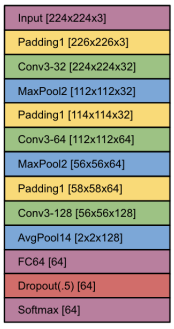


Fig. 3. Customized ConvNet Classifier Architecture

It has pretty good outcome for Swedish datasets, but unfortunately doesn't play well with other datasets. See the result section for more.

Compare the datasets, intuitively, there are a few things to consider:

- The lighting variation and background clutter. Swedish has very clean background, where for ImageCLEF uniform image quality varies in a large range.
- More species with less samples per classes.

Some of our data set such as ImageCLEF2013 is even more challenging in term of clutter and occlusion, which we didn't cover in the customized ConvNet classifier experiment.

A deeper ConvNet with more aggressive filters is needed to extract the features more efficiently and to deal with the noise and variation of datasets. But given the constrains of time and computations, we can't afford to train a deeper network with much more parameters.

As a common practice, we switched to use pre-trained weights of proven ConvNet architecture. Since the output is different for our specific problem, we cannot apply the architecture of the pre-trained weights directly. We used Transfer Learning which will be discussed in next subsection.

Transfer Learning:

The two well-known options for Transfer Learning are:

1) ConvNets as fixed feature extractor, and then classify with other Classifiers such as Logistic Regression/Softmax or SVM
2) Fine-tuning the ConvNets. For a N level structure, train the last H levels with the $N-H$ lower levels freeze (the higher the level, the less overfitting to the target-datasets) [21].

There are several options of architecture trained against the well-known ImageNet whose weights are available with Keras framework.

- VGGNet, runner-up in ILSVRC 2014 (GoogLeNet is the winner of that year).
- ResNet, winner of ILSVRC 2015. The available weights are for ResNet50 with 50 layers (including Fully Connected layers) in total.

The ideal situation to choose an architecture of pre-trained weights is that it has been trained against original datasets that is similar to the target datasets. ImageNet has $14,197,122$ images, 21841 synsets. Among them, there are 70 synsets that is related to leaves. Given the large number of syncsets, the weights trained against ImageNet is generalized enough that there will be relatively less bias. The weights are trained for the object image classification task of ImageNet, which is aligned with our task except that ours are more fine-grained leaf classification.

Further more, Jason et al. found that even features transferred from distant tasks are better than random weights [26]. Also note that ConvNets features are more generic in early layers and more original-dataset-specific in later layers[21].

After comparing preliminary results, we choose ResNet50 since ResNet50 gives better results and less overfitting. We believe this can be attributed to the fact that ResNet50 is deeper, but still having lower complexity[22]. It also generates lower dimension feature vector, which is likely due to the use of a more aggressive Average Pooling with a pool size of

7x7. This saves us from the effort to seek for reduction of dimensionality.

The ResNet is famous for it's deep layers[22], in our case, 50 layers, with 49 Convolution layers and one FC layer on top. Except for the first Convolution layer, the rest 48 composes 16 "residual" blocks in 4 stages. The block within each stage has similar architecture, i.e. same input & output shape.

The possible approaches as aforementioned, are either get the bottleneck features which is called CNN codes in the terms of transfer Learning, or freeze all the layers except for the last Residual Block and train with the target datasets. See Figure 4.

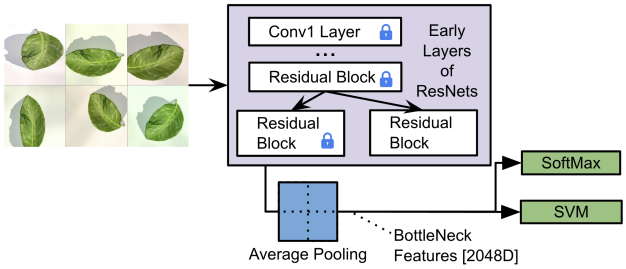


Fig. 4. Illustration of Transfer Learning with ResNet

Again, with the constrains of the time and computations, we choose to just extract the CNN codes.

We download the weights and load it into the preset model. And drop off the layer after the last Convolutional/Residual block. Feed the network with augmented data, the CNN codes (feature vector) we get from ResNet50 is 2048 dimensions.

With the help of visualization, intuitively we can see that each filter is seizing different features. The output from later layers is more abstract and global than the earlier layers. We can also see that some of the filters are completely dark, means for that particular filter, it doesn't response to certain feature of the image after rectification.

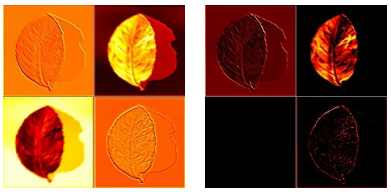


Fig. 5. Visualization of partial outputs for each stage of ResNet50. Left to right, top to bottom, they are from first convolutional layer, the activation layer right after respectively. Note that the 4 outputs of Convolution layer are corresponding to the 4 outputs of the activation layer

Thus, we can assume that the pre-trained weight is applicable to our problem, at least the early layers do generalize well to our feature space.

C. SIFT + Bag of Features (BoF)

Due to the simplicity and performance, this well established approach was taken at first. We prototyped the system with OpenCV libraries. Here is the illustration of the system 6. Key points are densely sampled with a step size of 20 from the grays caned copies of the training set. Then we extract SIFT descriptor for each key point, which are clustered to build visual words via K-Means. Several codebook size (500, 1000, 3000) are used and we pick the one with the best validation results for each data set. To reduce computation complexity, we randomly select 100 training images to build the codebook. Finally, each train and test sample is represented with histograms of visual words, i.e. term vectors, and this is used as feature vectors for classification.

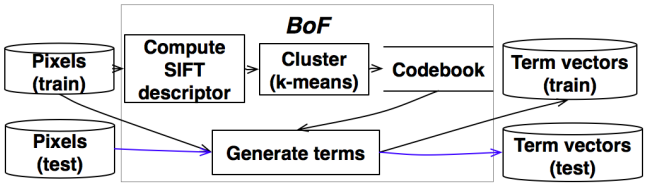


Fig. 6. Bag of Features

V. Experimental Results

First, let's take a look at the customized ConvNet architecture Using cross entropy to measure loss,

$$\sum_{i}^{n}\sum_{k}^{K} -y_{true}^{(k)} log(y_{predict}^{(k)})$$

we got the learning curve:

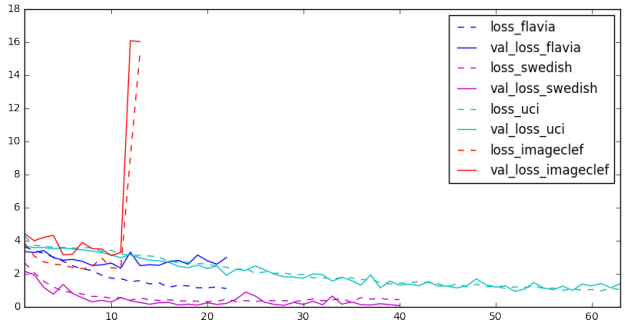


Fig. 7. Loss curve of ConvNet Classifier

The loss of Swedish dataset [15] keeps decreasing as expected. For Flavia dataset [16], the training loss keeps decreasing, but at some point, validation loss stop changing. For ImageCLEF uniform dataset [17], it actually stopped learning and the loss goes back up and stay there.

The accuracy is consistent from the loss, where accuracy of Swedish converges at 92%, with a test accuracy of 90.46%, but it doesn't work well with other datasets especially ImageCLEF dataset, which equals to random guessing.

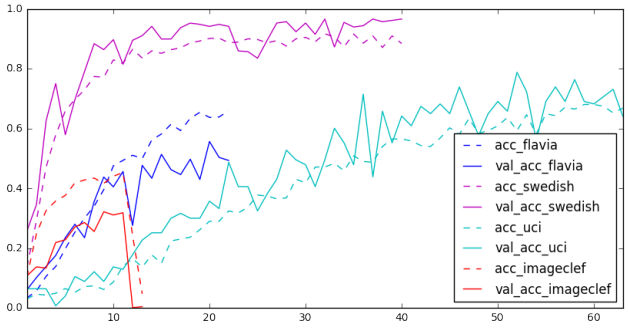


Fig. 8. Accuracy curve of ConvNet Classifier

We choose prediction rank-1 identification (i.e. accuracy) as our performance metric, which is defined as $Accurary = \frac{N_c}{N_t} \times 100\%$, where N_c represents the number of correct match and N_t is the total number of test samples.

We use 3-fold cross validation to select model and hyper parameters. All the best results are collected with Linear SVM and C=0.01.

And lastly, here is the test accuracy of the approaches we took.

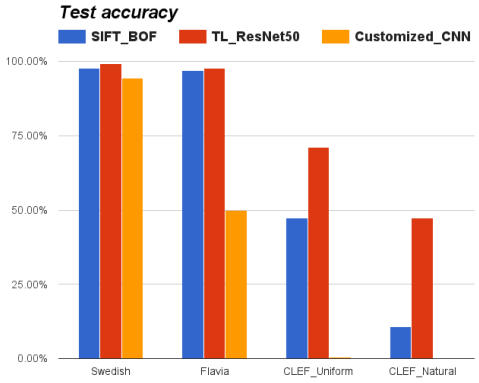
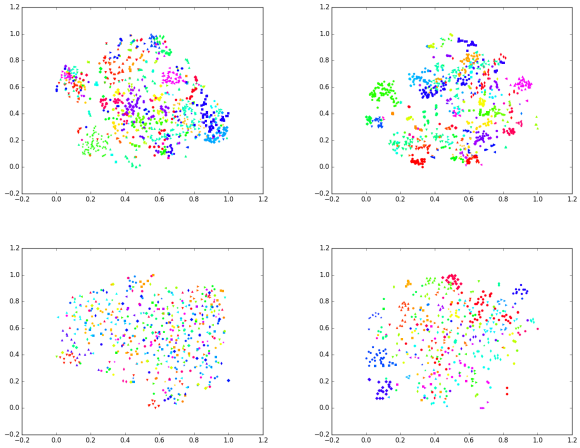


Fig. 9. test Accuracy

Dataset	SIFT_BOF	TL_ResNet50	Customized_CNN
Swedish	97.70%	99.33%	94.44%
Flavia	96.84%	97.81%	49.80%
CLEF_Uniform	47.40%	71.11%	0.50%
CLEF_Natural	10.75%	47.22%	

VI. Discussions

1) As expected, CNN codes with pretrained weights yields similar or better accuracy, compared to SIFT+BoF. Particularly traditional method suffers on noisy datasets. We use t-SNE [37] to visualize high dimension the feature vectors into 2D figure. It is clear that feature tend to spread and mix for ImageCLEF Natural dataset, which means it is harder to separate them in feature space. This is aligned with our results.



Top left: Uniform BoF. Top right: Uniform Transfer Learning. Bottom left: Natural BoF. Bottom right: Natural Transfer Learning.

2) To understand what stimulate neurons and how variations affect such stimulus, we visualize the feature map and deconv[31] of pool5 layer of ResNet for several pairs of data, using the visualization toolbox [38], in table I. Except for the last pair, all pairs come from the same species. And we can tell roughly that background clutter and color give quite different CNN codes.
   - Background clutter yields the biggest confusions for ConvNet. None of the top 5 ILSVRC labels match and the stimulated neurons are quite different.
   - Color and scale seem to activate slightly different sets of neurons.
3) As an exercise for error analysis, we manually inspect the results of two preprocessing methods and pick the best out of these two for ImageCLEF Uniform data set.

No preprocessing	K-means	Hand-picked
63.23%	58.79%	71.11%

This clearly tells that the K-means preprocessing is not solid and the background noise contributes to the misclassification results.

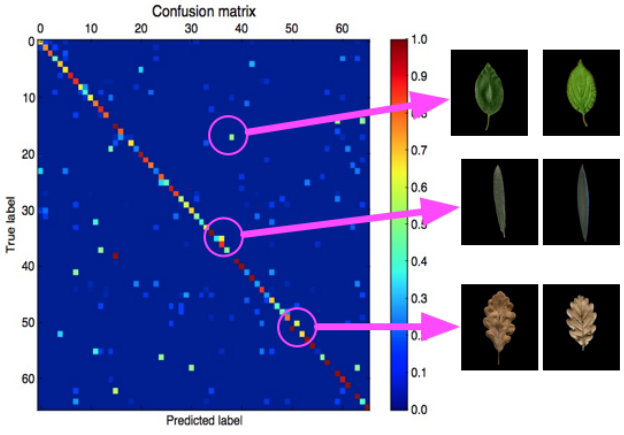


Fig. 10. Confusion Matrix

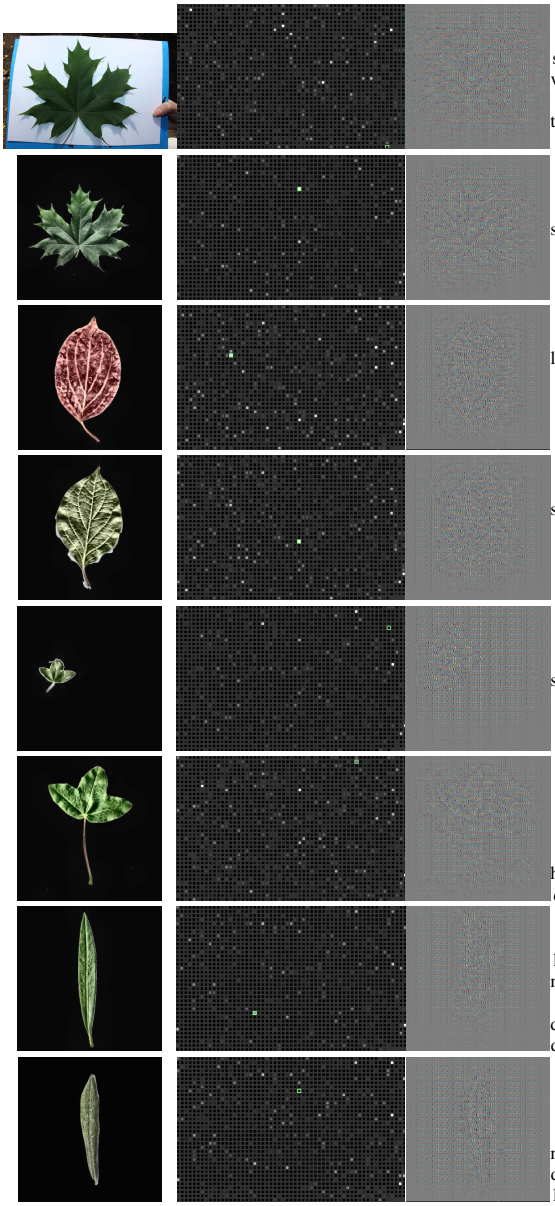


TABLE I. FEATURE MAP AND DECONV FOR POOL5 LAYER

4) Looking at the confusion matrix, we believe the main causes for misclassification
   - Very fine differences between species, which is hard even for human experts
   - Noisy and possibly non-representative train data lead to overfitting,

VII. CONCLUSION AND FUTURE WORK

We explore deep ConvNet with leaf classification problem. CNN codes and simple linear SVM gives the best results for both clean and noisy datasets. Further investigation gives some insights on impacts of background clutter, color/scale variations on CNN codes.

We think these are the area for improvement

1) Acquire more data and fine-tune ConvNets to fight overfitting problem
2) Engage advanced techniques for image augmentation
3) Explore state-of-art method to detect and locate leaf from background. This should allow ConvNet to focus on the discriminating features.

REFERENCES

[1] S. Cho, D. Lee, and J. Jeong. Automation and emerging technologies: Weedplant discrimination by machine vision and artificial neural network. Biosystems Engineering, 83(3):275280, 2002.

[2] Charles Mallah, James Cope, James Orwell. Plant Leaf Classification Using Probabilistic Integration of Shape, Texture and Margin Features. Signal Processing, Pattern Recognition and Applications, in press. 2013

[3] Pedro F. B. Silva, Andre R.S. Marcal, Rubim M. Almeida da Silva. Evaluation of Features for Leaf Discrimination. 2013. Springer Lecture Notes in Computer Science, Vol. 7950, 197-204.

[4] Itheri Yahiaoui, Nicolas Herve, and Nozha Boujemaa. Shape-based image retrieval in botanical collections, Lecture Notes in Computer Science including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinfor matics, vol. 4261 LNCS, pp 357-364, 2006.

[5] Jassmann TJ, Tashakkori R, Parry RM (2015) Leaf classification utilizing a convolutional neural network. In: SoutheastCon

[6] Neeraj Kumar, Peter N Belhumeur, Arijit Biswas, David W Jacobs, W John Kress, Ida C Lopez, and Joao VB Soares, Leafsnap: A computer vision system for automatic plant species identification, in ECCV, pp. 502516. Springer, 2012.

[7] DavidHall,ChrisMcCool,FerasDayoub,NikoSunderhauf, and Ben Upcroft, Evaluation of features for leaf classification in challenging conditions, 2015.

[8] Monica G Larese, Ariel E Baya, Roque M Craviotto, Miriam R Arango, Carina Gallo, and Pablo M Granitto, Multiscale recognition of legume varieties based on leaf venation images, Expert Systems with Applications, vol. 41, no. 10, pp. 46384647, 2014.

[9] Hasim A, Herdiyeni Y, Douady S (2016) Leaf shape recognition using centroid contour distance. In: IOP conference series: earth and environmental science, p 012002

[10] Hall, David, McCool, Chris, Dayoub, Feras, Sunderhauf, Niko, & Upcroft, Ben (2015), Evaluation of features for leaf classification in challenging conditions. In IEEE Winter Conference on Applications of Computer Vision (WACV 2015), 6-9 January 2015, Big Island, Hawaii, USA.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. Burges, L. Bottou, and K. Weinberger, editors, Advances in Neural Information Processing Systems 25, pages 10971105. Curran Associates, Inc., 2012.

[12] Y. Jia. Caffe: An open source convolutional architecture for fast feature embedding. http://caffe.berkeleyvision.org, 2013.

[13] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. arXiv preprint arXiv:1409.0575, 2014.

[14] https://archive.ics.uci.edu/ml/datasets/Leaf

[15] Oskar J. O. Sderkvist. Computer vision classifcation of leaves from swedish trees. Master's Thesis, Linkoping University, 2001.

[16] Stephen Gang Wu, Forrest Sheng Bao, Eric You Xu, Yu-Xuan Wang, Yi-Fan Chang and Chiao-Liang Shiang, A Leaf Recognition Algorithm for Plant classification Using Probabilistic Neural Network, IEEE 7th International Symposium on Signal Processing and Information Technology, Dec. 2007, Cario, Egypt

[17] http://www.imageclef.org/2013/plant

[18] Chollet, François, 2015, https://github.com/fchollet/keras,

[19] https://www.tensorflow.org/,

[20] Chollet, François, 2015, https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html

[21] Andrej Karpathy et al. http://cs231n.github.io/transfer-learning/

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. Deep Residual Learning for Image Recognition arXiv:1512.03385, Dec. 2015.

[23] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recogni- tion. arXiv preprint arXiv:1409.1556, 2014.

[24] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. arXiv preprint arXiv:1409.4842, 2014.

[25] Ali Sharif Razavian Hossein Azizpour Josephine Sullivan Stefan Carlsson. CNN Features off-the-shelf: an Astounding Baseline for Recognition, CVAP, KTH (Royal Institute of Technology) Stockholm, Sweden, May. 2014.

[26] Jason Yosinski,1 Jeff Clune,2 Yoshua Bengio,3 and Hod Lipson4. How transferable are features in deep neural networks? arXiv:1411.1792, Nov. 2014.

[27] David Eigen, Jason Rolfe, Rob Fergus, and Yann LeCun. Understanding deep architectures using a recursive convolutional network. arXiv preprint arXiv:1312.1847, 2013.

[28] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. J. Machine Learning Res. 15, 19291958 (2014)

[29] Rother, Carsten, Vladimir Kolmogorov, and Andrew Blake. "Grabcut: Interactive foreground extraction using iterated graph cuts." ACM transactions on graphics (TOG). Vol. 23. No. 3. ACM, 2004.

[30] Reza, Ali M. "Realization of the contrast limited adaptive histogram equalization (CLAHE) for real-time image enhancement." Journal of VLSI signal processing systems for signal, image and video technology 38.1 (2004): 35-44.

[31] Idier, Jrme, ed. Bayesian approach to inverse problems. John Wiley & Sons, 2013.

[32] Lindeberg, Tony. "Scale invariant feature transform." Scholarpedia 7.5 (2012): 10491.

[33] Dalal, Navneet, and Bill Triggs. "Histograms of oriented gradients for human detection." 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). Vol. 1. IEEE, 2005.

[34] Hall, David, et al. "Evaluation of features for leaf classification in challenging conditions." 2015 IEEE Winter Conference on Applications of Computer Vision. IEEE, 2015.

[35] Lee, Sue Han, et al. "Plant identification system based on a convolutional neural network for the lifeclef 2016 plant classification task." Working notes of CLEF 2016 conference. 2016.

[36] Sharif Razavian, Ali, et al. "CNN features off-the-shelf: an astounding baseline for recognition." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2014.

[37] Maaten, Laurens van der, and Geoffrey Hinton. "Visualizing data using t-SNE." Journal of Machine Learning Research 9.Nov (2008): 2579-2605.

[38] Yosinski, Jason, et al. "Understanding neural networks through deep visualization." arXiv preprint arXiv:1506.06579 (2015).