Plant Leaf Recognition

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

I. Introduction

Automatic plant species recognition with image processing is a fine-grained image recognition problem and generally considered a hard problem due to the subtle differences between different species in the same class. Sometime such fine differences can be even challenging to humans. Typically this requires large training data but it is not feasible due to the number of planets (over 220000 [2]).

1) The subtle differences between different species in the same class. Sometime such fine differences can be even challenging to humans.
2) Large number of categories, over 220000 [2].

The main application are weeds identification, species discovery, plant taxonomy, natural reserve park management and so on [3].

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 rest of the report is organized as follows. In section III, we describe the data set. Section.

II. Related Work

Research on automatic leaf classification from image has been active since 2000. Lots of hand-crafted features have since proposed, ranging from shape based, to statistical texture and margin [2] [3] [1]. Also generic computer vision object recognition/detection features, such SIFT and HOG are studied for this problem. [TODO] 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 [TODO]. Recently, with the huge success of ConvNets, particularly in annual ImageNet Large Scale Visual Recognition Challenge [11], researchers start to apply ConvNets to this problem [TODO]. [TODO] have 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 pretrained model.

III. DataSet

We found two types of data set

1) Clean images, which consists of well aligned leaf on single contrasting background, with little or no variations of luminance or color.

Name	Species	Samples Per Species
Swedish [15]	15	75
Flavia [16]	33	$\sim$ 60
UCI [14]:	40	5 16
Kaggle [3]:	99	16

2) ImageCLEF [17], which is collected through crowd sourced application, has 250 species and 26077 images. This is a much more challenging datasets with variations on lighting conditions, viewpoints, background clutters and even occlusions. The data further split to different subset: uniform, which is relatively with less clutter, and natrual-background, which is taken in a total natural enviroment. To effectively evaluate our approaches, we use subsets of this dataset.

Name	Species	train samples	test samples
uniform	66	[TODO:]	1194
natural	57	2585	521

IV. Approach

A. Overview

Images captured in the field, such as those from ImageCLEF [17], are more challenging as there are much more intra class variations due to viewpoint, lighting condition and occlusion. We break down the problem into two pieces. Firstly, we want to locate the leaf in the image which is an single object localization problem. This step becomes necessary if the image have multiple objects and possibly leaf is not the most salient object. Then, we predict the species of the leaf, which is a fine-grained classification problem. This is the main task and for most data set, we only need to perform this task. Here we explore both traditional method, such as Bag of Features, and ConvNets based method.

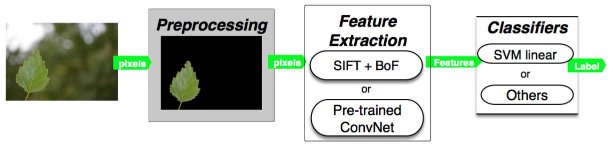

Fig. 1. Overview of the system

Firstly, during preprocessing, we apply Contrast Limited Adaptive Histogram Equalization to reduce lighting condition variation and raw images are resized to fit to the next layer. In traditional approach, we also use K-means to remove background color heuristically. For challenging dataset, we find the convex hull containing the largest N contours and then use GrabCut to segment leaf out of the background clutter. Next we extract features and we explore two options,

1) ConvNets. We take transfer learning approach, in order to make use of the power of ConvNets with the constrains of time and computations. Specifically, we take a couple of ConvNets that are pretrained with ImageNet for ILSVRC object image classification task, remove one or two top

layers and then treat the rest of the ConvNet as fixed feature extractor for the our data set.

2) SIFT + Bag of Features Key points are densely sampled and SIFT feature descriptors are retrieved from each raw images. Then at last, we train a classifier, using the feature vectors from the above step and predict labels for our test data.

B. *ConvNets approach*

1) Setup: For ConvNets, we used Keras framework [18] with Tensorlow 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 ConvNets architecture.

2) Exploration: We started by training ConvNets 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[24].
2) Dropout reduces overfitting [25] [11]
3) Use aggressive pooling to reduce the dimensionality.

We designed our ConvNets with the architecture below:

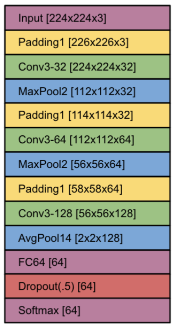

Fig. 2. Customized ConvNets Classifier Architecture

Using cross entropy to measure loss, we got the learning curve:

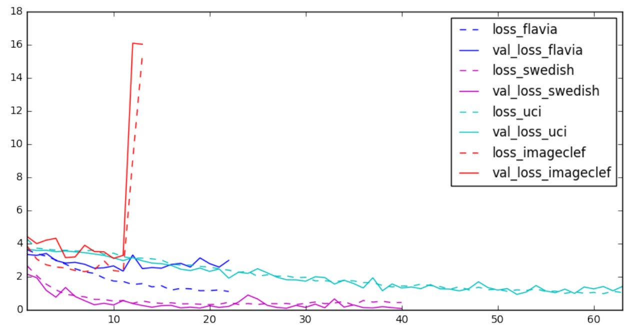

Fig. 3. Loss curve of ConvNets Classifier

The loss of swedish dataset [15] keeps decreasing as expected. But for flavia dataset [16], the training loss is keep 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 accaracy of 90.46%, but it doesn't work well with other datasets especially imageclef dataset.

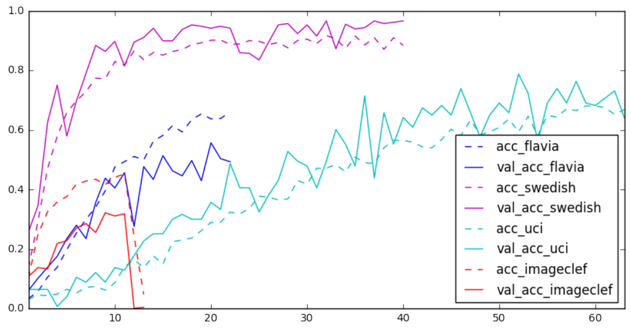

Fig. 4. Accuracy curve of ConvNets Classifier

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 ConvNets classifier experiment.

A deeper ConvNets 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 ConvNets architecture.

3) Data Augmentation: Before we start with the pre-trained weights, application of data augmentation is helpful. Research shows number of samples need to be big enough to avoid excessive overfitting. See B section of the supplementary material of [23]. Note that the research is proven for AlexNet. While we don't have enough data to seek for the threshold which suffices to avoid overfitting for our choice of architecture, we assume for the ImageCLEF data sets [17], 20 per samples is not enough.

Data augmentation is widely used to reduce overfitting on image data [11]. This technique makes up the lower amount of samples and with some transformation increases the variation of the image. In our case, we've applied rotation, flipping, space shifting and channel shifting.

For our problemset, we experimented both to apply or to not apply data augmentation for the ImageCLEF dataset.

4) Transfer Learning: Since the output is different for our specific problem, we cannot apply the architecture of the pre-trained weights directly. At minimum we have to drop the softmax layer at the very end to retrain with our own dataset. The technique is known as Transfer Learning. The two well-known options for Transfer Learning are:

1) ConvNet as fixed feature extractor, and then classify with other Classifiers such as Logistic Regression/Softmax or SVM
2) Fine-tuning the ConvNet. 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) [20].

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 ideality 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 [23]. Also note that ConvNet features are more generic in early layers and more original-dataset-specific in later layers[20].

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[21]. 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[21], in our case, 50 layers, with 49 Conv layers and one FC layer on top. Except for the first Conv 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 forementioned, 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 5.

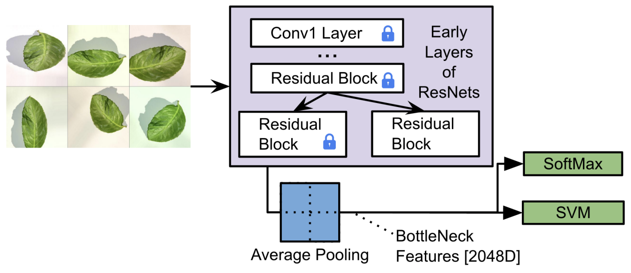

Fig. 5. Illustration of Transfer Learning with ResNet

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, roughly the outlines, the texture or the contour. 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.

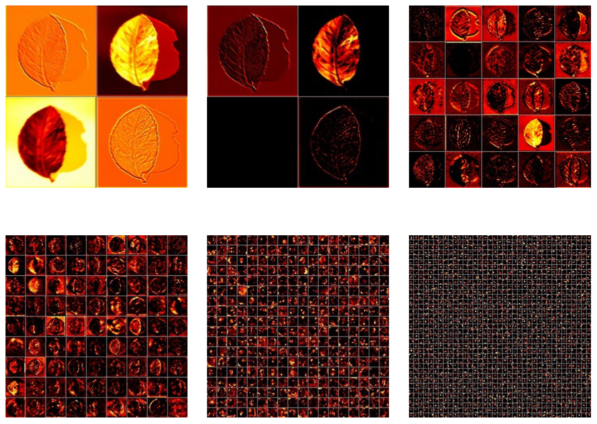

Fig. 6. Visualization of partial outputs for each stage of ResNet50. Left to right, top to bottom, they are from First Convolutional layer, the activation layer right after, Stage 2, Stage 3, Stage 4 and Stage 5 respectively. Note that the 4 outputs of Conv layer are corresponding to the 4 outputs of 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.

5) Classification: The feature vector is normalized with unit variance and we learn a linear SVM classifier (or others) from the features.

We applied grid search for the hyperparameters. For SVM, we searched for the C parameters with logarithmic scale .01, .1 and 1.

C. Discussion

How do you decide what type of transfer learning you should perform on a new dataset? This is a function of several factors, but the two most important ones are the size of the new dataset (small or big), and its similarity to the original dataset (e.g. ImageNet-like in terms of the content of images and the classes, or very different, such as microscope images). Keeping in mind that ConvNet features are more generic in early layers and more original-dataset-specific in later layers, here are some common rules of thumb for navigating the 4 major scenarios: [20]

1) similarity (transfer learning is good)
2) size for each class(fine tuning is not a good idea)

D. Bag of Features (BoF) + local feature descriptors

Due to the simplicity and performance, this well established approach was taken at first. Interest points are detected from the raw images and then local invariant feature descriptors are collected, which are clustered to form the visual vocabulary/codebook. Afterwards, each raw image can be represented with histograms of visual words, i.e. term vectors. We have prototyped a BoF system, based on the OpenCV package. Here is the illustration of the system 7.

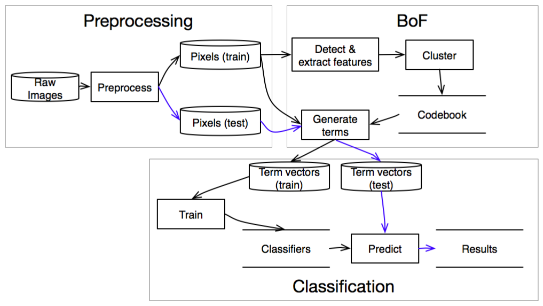

Fig. 7. System Design for BoF

During preprocessing, raw images are converted to gray-scaled images and resized to reduce computation complexity. We extract SIFT descriptors from the pixels after detecting the key points. Limiting the width of image to 128 pixels, we have roughly 200 SIFT descriptors per image. Then all the descriptors are clustered to build visual words via K-Means. Due to computation complexity, we pick randomly 100 training images to build the visual vocabulary for the initial run.

V. Experimental Results

[Explain cross validation approach we used]

We use 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.

[Test results as table.]

VI. Discussions

VII. Conclusion and Future work

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

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

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

[22] 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.

[23] 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.

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

[25] 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)