Plant Leaf Classification

EECS E6690 Final Project Report

Shuyi Tan (st3092)

Tingyu Guan (tg2663)

Yusang Mao (ym2694)

Introduction

Plant species classification is of great importance since it has been widely used for many industries such as medicine and farming. Clear identification of plants benefits the environment as well, as chemical fertilizers or pesticides can be correctly applied so that in a way reduce the chemical wastage. According to a report by the Royal Botanic Gardens, Kew, there are about 391,000 species of plants and 369,000 of which are flowering plants. [2] The numerous species do make the classification work more challengeable, and a variety of approaches are researched for plant species classification. A method of identifying leaves to classify the plant species is proposed according to plant taxonomy. The leaves are relatively easily found and collected.[2]

In this project, we are going to reproduce the work from the paper *Plant Leaf Classification Using Probabilistic Integration of Shape, Texture, and Margin Features.* Classification, using K-nearest-neighbour to estimate posterior probability and make predictions, is based on the three features extracted from leaves including a shape descriptor, a texture histogram and a fine-scale

margine, which are studied independently. From the results, the method is believed to be effective and accurate in conditions of small training set size.

Except the technique proposed in the paper, another three techniques are conducted for comparison. Decision trees, support vector machines and neural networks are designed for the same objective. In this report, results of the three methods together with the K-NN are discussed and compared.

The report starts from introduction of the datasets used and paper content, followed by the description of the reproduction work and other techniques we applied and ends with the discussion of different methods and conclusions.

Datasets and Paper

Dataset

The paper mainly uses a dataset created by the authors which contains 100 species of leaf and 16 samples for each species. However, in our reproduction work, because there is a missing value in the species “Acer Campestre”, the whole species is deleted and we only use 99 species for training as well testing. Leaf samples are primarily recorded as color images and placed on a white background. They were then transferred to binary images using global thresholding [5] on gray scale images. [2] Afterwards, features are extracted from the binary images. Figure 1 shows some examples of binary images in the dataset.

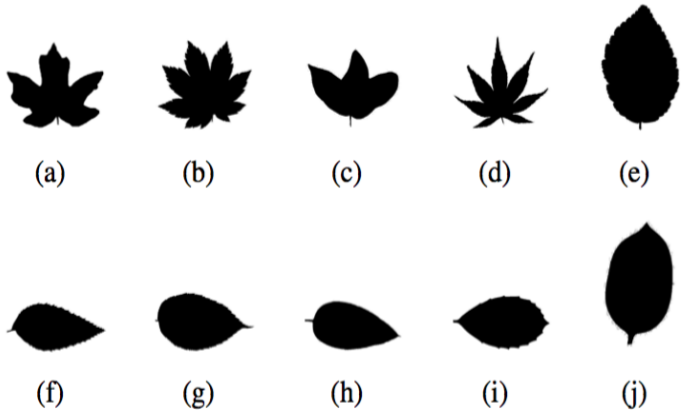

Figure 1. Examples of leaf binary images

There are three features used in this paper: shape, texture and margin. Leaf shape uses centroid contour distance curve (CCDC) approach to build a 1D function that represents a 2D area or boundary[2]. The feature texture is generated from the filters. Features are produced by applying different filters to 1024 randomly selected windows. Each window contributes to 20 features respectively. As for leaf margin, it is computed by the combination of a median filter, calculations of distance between points and overlapping windows. The three features are all quantized to feature vectors which contain 64 elements.

As a matter of fact, we do not begin from processing the raw binary images. The feature data is already prepared and can be downloaded from the internet. The following work, which mainly focuses on leaf classification, is based on the feature data.

Methodology of the paper

Density Estimation using K-NN

The paper aimed at producing the posterior distribution of a leaf's class, and K-nearest-neighbour with k=3 is used to estimate the probability density. Two methods are introduced, proportional and weighted proportional density estimation. In the first method, the probability of the leaf belonging to class i is simply the proportion of neighbours which belong to class i among all k neighbours. In the second method, different weights are proposed on each classes so that their votes also have different weights. The weights are determined by maximizing the log likelihood of the set of weights.[1][4]

Also, one of the basic assumptions proposed on the data set is that three features are independent, so they can be modeled separately. The posterior probability is the product of probabilities gained by three features, i.e.

$$Pr(x \in C_i) = Pr(x \in C_i | feature_1) \times Pr(x \in C_i | feature_2) \times Pr(x \in C_i | feature_3)$$

Besides, to prevent potential problems when calculating expected log likelihood, a small adjustment term is added to each probability to ensure they are all greater than zero [4], i.e.

$$P_i' = P_i(1 - t) + \frac{t}{c}$$

where $t = 0.01$and c is the total number of classes.

To standardize the probabilities for comparison, they are scaled to sum to one.

Evaluation

Performance evaluation of various classifiers are based on validation which is similar to leave-one-out cross validation, that is, one leaf example is extracted as testing data for each of the 99 species, and the remaining 15 leaf examples act as training data[3]. Therefore, 16 different training/testing sets are obtained, and the following two quantities are calculated to assess the model.

The first one is ACC, i.e. the mean accuracy of the $n = 99 \times 16 = 1585$ trials. The other one is expected log likelihood (ELL), i.e.

$$ELL = \frac{1}{n}\sum_{j=1}^{n} \log(W_j)$$

where $W_j$is the estimated density of the correct class for trial j. ELL indicates how confident we are to make the predictions, and ideally $ELL = 0$.

Reproduction

R function kknn in package “kknn” is used, and arguments *kernel=”rectangular”* is set for proportional density estimation and *kernel=”optimal”* for weighted one. Final results are shown in the following table.

Method	MAR	SHA	TEX	ACC	ELL
Prop	√			0.776	-1.164
		√		0.612	-2.261
			√	0.79	-1.094
	√	√		0.855	-0.762
	√		√	0.939	0.267
		√	√	0.885	-0.489
	√	√	√	0.968	-0.16
W.Prop	√			0.758	-1.189
		√		0.645	-2.258
			√	0.815	-1.076
	√	√		0.85	-0.75
	√		√	0.944	-0.268
		√	√	0.893	-0.484
	√	√	√	0.965	-0.159

Table 1: Reproduction results

It can be seen that two methods perform quite similarly, and the highest accuracy, which equals 96.8%, is achieved by proportional density estimation using all three combinations. Comparing the accuracy of classifiers using different combinations of features, it seems that “margin” and “texture” are more strong features, which means they contribute more, than “shape”, and it is against intuition. Perhaps it is because among all 99 species, many tree leaves share really similar shapes. For example, leaves (f) and (h), (e) and (j) in Figure 1 are almost the same in shape. As a result, “shape” alone cannot accurately makes the classification. However, a single feature can achieve over 80% accuracy already, which means the feature extraction is quite satisfactory.

Also, the impact of different training size is examined by decreasing the number of leaves for training, in order to see whether K-NN is effective for even a smaller sample size. We still pick 1

leaf of each species for testing, and pick 1 to 15 leaves for training. The relationship is shown by the following two graphs.

Figure 2: ACC vs Training size

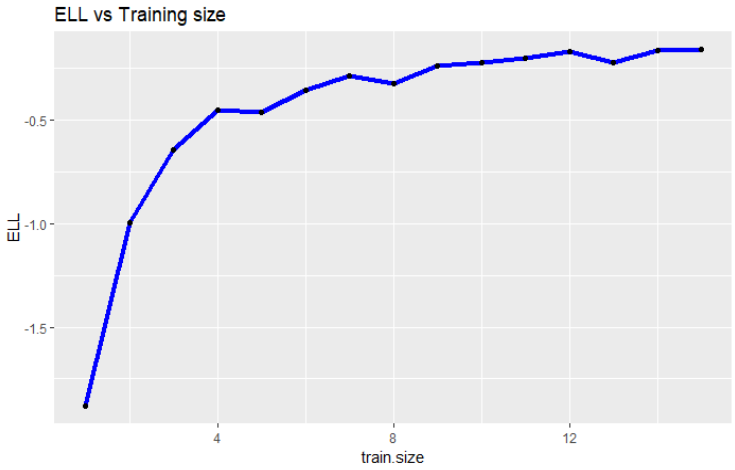

Figure 3: ELL vs Training size

It is amazing to find when the training size is only 4, the prediction accuracy can reach 90%, which means K-NN deals with the issue of small sample size quite well, and large samples of leaves are not needed to collect for classifier building.

Different Techniques

SVM

Support vector machine, unlike K-NN, is a two-class classifier. Simply using SVM can only split the data points to 2 classes. However, our plant leaf classification is a 99-class classification problem. There are three approaches to use SVM to do multi-classification problem.

The first approach is one-versus-rest. We can define one of the class as positive class and define the rest of the classes as negative class. For k-class problem, we should build *k* SVM classifiers to fulfill the classification. These *k* classifiers give the classification result in order, until we put the testing sample into a class. The complexity is $O(k)$. This algorithm is fast. But the samples are unbalanced, which decreases the performance of the model.

The second approach is one-versus-one. For every two classes, we build a SVM classifier. Set one of the class as positive class and another one as the negative class so we have to build totally $\frac{k(k-1)}{2}$ classifier. These classifiers all give a vote for classification. The class with the most vote will be assumed as the final class. The complexity of this model is $O(k^2)$, so this is costly when the *k* is large

The third approach is to use a binary tree for the classification. For *k*-class problem, we only have to build $k-1$ classes. It is very efficient. But if the classification in one layer is wrong, it will definitely lead to the mistake of the following layers.

We use the svm function in R package "e1071" to build the multi-classification model and the performance is quite well. We concatenated 3 different features together, so that we have training

and testing data with 192 elements. We train the SVM model using linear kernel and evaluate the model using 16-fold evaluation. The average accuracy we got is 98.9%.

The table shows the accuracy of the model when using different combinations of the 3 features. We can see that with only one feature, the accuracy is comparably lower. They are only 85.2% for margin, 71.0% for shape, 87.1% for texture. However, when combining the features, the accuracy raised a lot. All of the combinations have their accuracy above 90%. In particular, with margin features and texture features combined together, the model accuracy achieves 98.7%. We can see that margin and texture features contribute much in SVM model.

Method	MAR	SHA	TEX	ACC
SVM	√			0.852
		√		0.71
			√	0.871
	√	√		0.934
	√		√	0.987
		√	√	0.953
	√	√	√	0.989

Table 2: Results of SVM

Tree

We tried to apply classification trees on the data set, so only predicted labels can be achieved and predicted probabilities are not available. As a result, the three features are concatenated together, forming a vector of 192 elements. Since there are 99 classes, considering one-verses-rest method, 99 binary classifiers are needed. Therefore, the data are relabeled, and the *i*-th tree aimed at determining whether the leaf belongs to class *i* or not. If the *i*-th tree classifies the leaf as class *i*, then class *i* will get one vote; otherwise, the other 98 classes will get $\frac{1}{98}$ vote equally. The votes

from 99 trees will be added, and the class which has the most votes will be the final predicted class.

R function “tree” in the package “tree” is used. Following table shows the results. The mean prediction accuracy is around 90%, which is much lower than K-NN. To find the possible reason, some of the trees are examined, and it is interesting that there are only about five nodes in each tree, which means the information contained in the data is not fully utilized by the trees.

Method	MAR	SHA	TEX	ACC
Tree	√			0.825
		√		0.827
			√	0.831
	√	√		0.875
	√		√	0.875
		√	√	0.888
	√	√	√	0.900

Table 3: Results of Tree

Neural Networks

We build a simple neural network to model the plant leaf data. Also, three features are concatenated. Figure 4 displays the structure of the neural networks and Table 4 shows the parameters. The input dimension of the network is 3 multiply 64. We build 2 hidden layers in the model, and both of the two hidden layer have 180 neurons. The dimension of output layer is 99, which refers to the 99 classes in our classification problem.

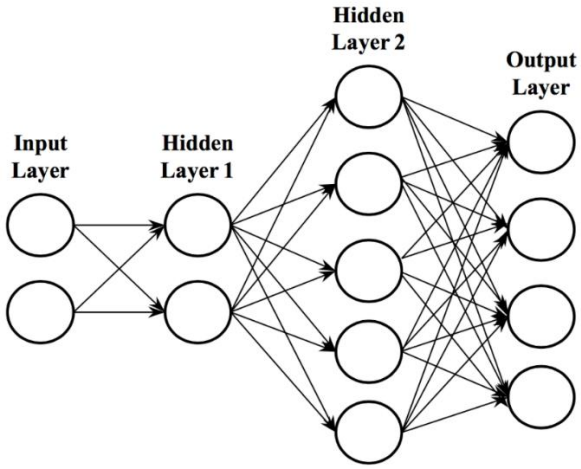

Figure 4: Structure of the NN

input dim	64 * 3
1st hidden layer dim	180
2nd hidden layer dim	180
output dim	99
transfer function	Relu
optimizer	Adam
batch size	200
number of epochs	1000

Table 4: Parameters of the NN

We have tried to build the third hidden layer, but this does not perform better than two-layer network. Maybe it is because of the small size of our data set, so it is not very suitable to build too complicated model.

We choose Relu as the transfer function and Adam as the optimizer in the neural network. The batch size is set to 200. and since our data set is not very big, we run 1000 epochs to get fine performance. We also tried to run more epochs. When we run for example 1200, 1500 epochs to

train the model, we find that the accuracy decreases. So, the training with 1000 epochs is an appropriate setting for this model and this dataset to avoid overfitting.

We still used 16-fold evaluation to validate our model. The following table shows the accuracy of the 16 models. And the average accuracy of neural network model is 96.6%.

Index	1	2	3	4	5	6	7	8
ACC	96.97	97.98	95.96	96.97	96.97	97.98	97.98	97.98
Index	9	10	11	12	13	14	15	16
ACC	95.96	92.93	96.97	94.95	97.98	95.96	97.98	94.95

Table 5: Results of NN

Other Attempts

We also tried logistic regression to solve the problem, but the number of attributes is relatively large for each feature, it cannot converge to find the estimated parameters. Naturally, LASSO is considered to be able to reach sparsity effect, which can reduce the number of covariates included. However, it does not work either.

Then, we tried to perform dimension reduction before model building. Principal component analysis is used and the results are desirable. The first 20 components of each feature are able to explain more than 90% of the variation, which is much smaller than the original dimension 64. Then, logistic regression is built on the principle components, but with so few covariates, it still fails to converge. Considering the meaning of the values of margin and texture(they are from histograms), perhaps there is no hidden linear relationship and logistic regression is not suitable for the dataset. However, the results of PCA is quite inspiring, and other classifiers can be tried to perform based on the principal components in future work.

Discussions and Conclusion

There are two special features of the dataset. One is that the number of classes, which is 99, is pretty large. On the other hand, there are only a small number samples, which is 16, in each class. As a result, two potential problems may appear when training a classification model. First is that data for training is inadequate, which can severely influence the performance of the model. Second is that since there are 99 classes, it is inefficient to build one-versus-rest classifies for binary classifiers such as decision tree and SVM.

Table 4 summarizes the results obtained by the four techniques. Except the decision tree, the other three techniques have relatively good performance. It seems that K-NN are able to achieve good classification result although the presence of a large number of classes and small size of samples. Also, notice that even though the neural network can reach a desirable outcome finally, it requires to run over 1000 epochs until it reaches a high accuracy, which is at a cost of much money and time in practice. In addition, SVM is excellent for this dataset, even though it is not efficient for training as mentioned above, it is really effective for classification.

Evaluation\Classifier	K-NN	Tree	NN	SVM
ACC	0.968	0.9	0.967	0.989

Tabel 6. Results of four techniques

Overall, the K-NN method implemented in this paper is useful and applicable, it is efficient and effective. Some other methods which can implement this function or even obtain great accuracy in classification, may flaw in its efficiency of computation. Thus, we can say that K-NN is a good way to cope with small training samples and a relatively large species.

As for the three features, generally, texture is the strongest one. That is to say, if only one feature can be used, or there is no such condition to measure all three features, texture is recommended to be measured so that we can predict accordingly. Besides, the combination of texture and margin almost has the same power as the combination of all three features. To conclude, considering the high accuracy of some simple models such as K-NN and trees, the three features are really representative and suitable for the classification.

Reference

[1] Atiya, Amir. “Estimating the Posterior Probabilities Using the K -Nearest Neighbor Rule.” (2004)

[2] Dasgupta, Shreya. “How Many Plant Species Are There in the World? Scientists Now Have an Answer.” (2016) https://news.mongabay.com/2016/05/many-plants-world-scientists-may-now-answer/

[3] Mallah, Charles. “Plant Leaf Classification Using Probabilistic Integration of Shape , Texture and Margin Features.” (2012).

[4] Mallah, Charles. “Probabilistic Classification from a K-Nearest-Neighbour Classifier.” (2013)

[5] Otsu, Nobuyuki. “A threshold selection method from graylevel histograms.” (1975)