Proceedings of the 2023 International Conference on Machine Learning and Automation
DOI: 10.54254/2755-2721/39/20230582

Exploring multiple algorithms for leaf classification: A comparative study

Haoyang Pan

School of Mechano-Electronic Engineering, Xidian University, Xi'an, Shannxi, 710071, China

1372282248@qq.com

Abstract. In the world there are an estimated nearly half a million plant species, and the leaves of each plant have their own unique characteristics. However, leaf classification has historically been problematic, with some similar-looking leaves being repeatedly identified and errors may even occur. In the past, people used precision instruments, chemical analysis, and other methods to improve the success rate of resolution, but this often required a certain amount of professional knowledge, sometimes it was trouble to operate. And the development of computer in recent years makes the leaf classification problem can be solved better through the image processing and machine learning technology. Therefore, the topic of this paper is to classify leaves by different algorithms and compare the classification results, which include KNN, Random Forest, the neural network algorithm ANN, and CNN. It can be seen through experiments that the classification results obtained by ANN have a higher accuracy rate among these algorithms.

Keywords: Leaf Classification, Machine Learning, Multiple Algorithms, Validation Accuracy.

1. Introduction

For leaf classification, manual identification by a professional botanist is often required, which is not only inefficient, but also susceptible to subjective factors. With the development of computer vision and machine learning technologies, it can now use these technologies to automate and optimize the process of leaf classification. For example, the random forest algorithm can deal with many characteristic variables, which shows that the algorithms can perform the analysis objectively and classify some similar leaves. In addition, the computer has a large audience, it can attract different professional people to participate in leaf classification, which will dramatically lower the barriers to botanical research.

These paper mainly search for open-source data sets on the network, conduct relevant data preprocessing with python as the programming language, and then use four algorithms to classify and visualize the results in different ways. For the study of leaf classification through algorithms, the ability and efficiency of species identification can be improved, especially in botany. Secondly, by classifying leaves it is also possible to understand the distribution of different species in the ecosystem. Finally, the health of leaves can be determined by the texture and shape of the leaves, which is beneficial for the monitoring and research of plant disease.

DOI: 10.54254/2755-2721/39/20230582

2. Relevant works

2.1. KNN algorithm

K-Nearest Neighbor (KNN) is a relatively simple algorithm in the field of data mining. It should be noted that this algorithm has no obvious training process and belongs to the scope of lazy learning. The simplicity and low identification error of this rule makes it the reference tool to test classifiers and datasets [1-2]. Given a set of samples and their corresponding classes, the KNN algorithm identifies the K most similar samples when presented with an unknown class data. After analyzing which class most of these samples belong to, the input data can also be divided into this one [3]. The term "majority" refers to a measure of similarity, where the distance between samples is used to uncover their relationships. The three core elements of the algorithm are the distance measure, the choice of k-value and the classification decision rule.

2.1.1. Distance measurement. The selection of different Distance measurement tends to have different impacts on the results. The main methods are as follows:

(1) Euclidean Distance

It represents the actual distance between two or more points in space. Given that there are two points A and B, its formula in n-dimensional space is:

$$d(A,B) = \sqrt{\sum_{i=1}^{n}(xia - xib)^2} \quad (1)$$

Here x1, x2, x3..., it refers to n features on the sample data.

(2) Manhattan Distance

This distance can be seen as the sum of the distances of the projections produced by Euclidean distances and the formula is:

$$d(A,B) = |\sum_{i=1}^{n}(xia - xib)| \quad (2)$$

(3) Chebyshev Distance

In two-dimensional space, the analogy of chess is often used, for example, the minimum number of moves a king can move to any square can be regarded as the Chebyshev distance. In N dimension, the formula is:

$$d(A,B) = \max|(xai - xbi)| \quad (3)$$

2.1.2. Choice of k value. If the K value is too small, the range between samples will be narrowed, which may make the data more susceptible to some anomalies. If the value of k is too large, it will also bring some negative effects. As the scope of the sample is expanded, some distant points that do not belong to this category are also classified into one category, which will lead to classification errors in the test sample, so it is very important to choose the appropriate K value.

2.1.3. Classification decision rules. The classification decision rule adopts the idea of minority obeying the majority, which means that most species in the recent K samples belong to which category. For example, the IRIS model uses the length and width of the sepal and petal to predict which category of iris flowers belongs to Setosa, Versicolor and Virginica. Data points can also be classified by looking at the distance between the three regions, the closer they are, the more similar they are.

2.2. Random Forest

Random Forest is an extension of the bagging algorithm in ensemble learning. The name itself provides two key definitions: "Random" and "Forest". The former indicates that the selection of training samples and features for tree construction is random, while the latter signifies that the forest is composed of multiple decision trees. Instead of splitting each node using the best split among all variables, RF splits each node using the best among a subset of predictors randomly chosen at that node [4].

DOI: 10.54254/2755-2721/39/20230582

2.2.1. Ensemble learning. It cannot be defined as a specific machine learning algorithm. It involves training a series of individual learners and combining them using certain strategies to create a highly capable learner that can tackle more complex learning tasks. Weighing and aggregating several individual opinions will be better than choosing the opinion of one individual [5]. In other words, ensemble learning is a collection of advantages between various learners with higher accuracy [6].

2.2.2. Bagging. Given a data set containing m samples, first a random sample is selected from it and put into the sampling set, and then put the sample back into the original data set. After m iterations of random sampling, a bootstrap sample containing m samples is obtained. In this process, some samples in the training set may not appear in the bootstrap sample at all, while others may appear multiple times. For large m, this is about $1 - 1/e = 63.2\%$, which means that each bootstrap sample contains only about 63.2% unique instances from the training set [7]. From these bootstrap samples, we can further sample n samples and train an individual learner for each sample. Finally, these individual learners are combined using different strategies such as averaging, voting, or learning-based methods, depending on the specific circumstances.

2.3. ANN

Artificial Neural Network (ANN) is a powerful tool for processing machine learning problems. It can be widely used in regression classification, image processing and other fields [8]. By simulating the structure and function of biological brain network, ANN is composed of many nodes (or "neurons") connected to each other, which can be used to model the complex relationship between data.

A typical artificial neural network has the following three parts, architecture, activation rule and learning rule. Architecture indicated specifies the variables in the network and their topological relationships. Activation Rule means relu, Sigmoid or other functions. It can be regarded as a filter, receiving various signals from the outside world, and finally output the expected value. And, learning Rule specifies how the weights in the network adjust over time, such as the Delta rule or the Back-propagation rule.

2.4. CNN

Convolutional Neural Networks (CNN) is a multi-layer feed-forward neural network and is the popular deep learning model [9]. Its basic principle is to learn local input features automatically and adaptively through convolution operations, reduce the dimension of features through pooling operations, and finally classify or regression through the fully connected layer. It is divided into input layer, convolution layer, excitation layer, pooling layer, and fully connected layer.

The main task of input layer is to preprocess the original image data. De-mean is center all dimensions of the input data to 0. If the original sample center is not at the origin of the coordinate system, this method can pull the sample back to the center. Moreover, normalization is to change all the data into the same range 0-1, because sometimes the range of different eigenvalues is different. If we use the unprocessed data directly, the training time will increase, and the accuracy will be lost.

As Figure 1 shows, given a 4x4 input image and a 2x2 filter, the step length to 1. The 2x2 portion in the upper left corner of the input image is then summed by multiplying it with the data in the filter, and put the result into the output layer. Then move the box used to select the input one space to the right, and do the same operation as before, placing the results next to the previous set of data, thus completing the convolution in turn. After the calculation is complete, a 3x3 output is formed on the right side.

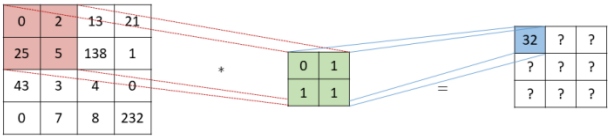

Figure 1. Convolution operation for 4x4 input image and 2x2 filter.

DOI: 10.54254/2755-2721/39/20230582

The activation function used by CNN is generally ReLU, which is a piecewise linear function, which can turn all parts less than 0 into 0 and make the parts greater than 0 still maintain the original shape. The sigmoid function can convert any value between 0 and 1, where the two sides are respectively in an approximate boundary state. Also, it can be used to solve the multi-label problem. Compared with sigmoid function, ReLU is characterized by fast convergence, simple gradient finding, and no problem of gradient disappearing. In addition, the SoftMax function is sometimes used for single label problem (Figure 2).

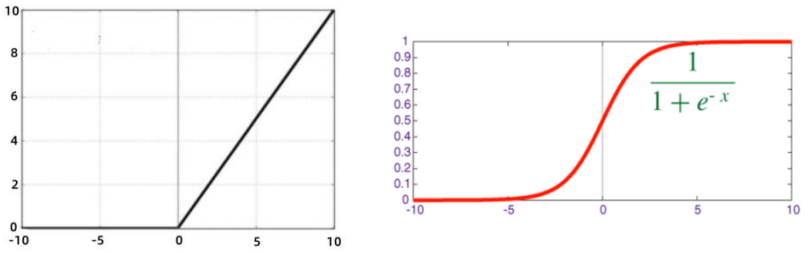

Figure 2. Graphical representation of ReLu (left) and Sigmoid (right) function [10].

After the convolution operation, pooling processing such as Max pooling and Average pooling can further reduce the amount of computation. In the Figure 3, the 4x4 data is divided into four regions through a 2x2 filter, and then the largest number in each of the four regions is found and a 2x2 result is generated.

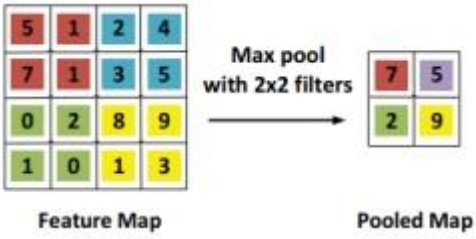

Figure 3. A max pooling operation [11].

FC plays the role of "classifier" in the whole convolutional neural network. If the previous operation is to extract features, then it is to map the convolution data to the sample label space, to make a classification. It should be noted that the hierarchical structure discussed in this paper is based on a single channel as an example, and in practical applications, multiple channels are often used, such as color images with RGB three color channels, then the image can be represented as 3*h*w.

3. Methodology

3.1. Data source

The data set used in this experiment was obtained from a competition project on the Kaggle platform and belongs to the public content. The dataset originally included 100 species and 16 samples each, but due to incomplete correlation data, the platform removed one species, leaving 99 species with approximately 1,584 leaf specimen images. These images are converted into binary black leaves on a white background, where each image also has three sets of features: a shape continuity descriptor, an internal texture histogram, and a fine-scale edge histogram. For each feature, each leaf sample is given a vector of 64 attributes.

3.2. Data Preprocessing

The selected dataset is calculated to know a total of 1584 images.

DOI: 10.54254/2755-2721/39/20230582

By using read_csv function to read the CSV files of the training and test sets, the leaf species information can be obtained after entering the train['species'] code. Next, this paper defined a function to encode the label "species" in the training set. After the transformation, the columns "species" and "id" in the training set are removed and tested to obtain the information. The rows indicate that there are 594 samples in the test set, while the columns indicate that each image has three sets of features with 64 vectors in each set. Then a new graph window is created, and the size of the graph is set to 20x15, 25 images are randomly selected and displayed as in Figure 4.

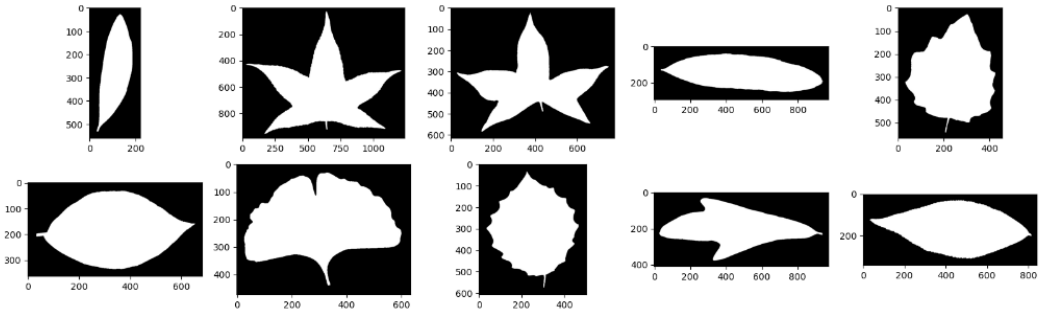

Figure 4. 25 random leaf images.

The training set is split into a training set and a validation set, via the train_test_split function, with the latter accounting for 20% of the total training set size and the former is the remaining 80%. In addition, a seed values is set to ensure that the data is split the same way every time the code is run. For the hierarchical strategy, it is set to labels, which means that in each partition, the proportion of each class is the same as in the original data set.

4. Experimental

4.1. KNN

4.1.1. Data preprocessing. First, a StandardScaler instance is fitted using the training data and the training data is normalized. Second, the validation data is normalized using the fitted StandardScaler instance. The authors did not normalize and weight the data at the beginning of the experiment, so the accuracy was only 87.8788%. After reopening the operation and normalizing the operation, the accuracy is greatly improved.

4.1.2. Model creation and training. After the pre-processing, a KNN classifier instance is created and got the best K value of 3 after cross-validation. After that, the KNN classifier is trained using standardized training data using distance weighting and uniform weighting methods, respectively.

4.1.3. Model prediction and evaluation. The trained KNN classifier is predicted using standardized validation data and accuracy, log loss and confusion matrix are calculated. When K=3, weight= 'distance', Accuracy is 97.9798% and Log loss is approximately equal to 0.2254. When K=3, weight= 'uniform', Accuracy is 96.9697% and Log loss is approximately equal to 0.2282. due to distance weighting method works better.

4.1.4. Visualization of confusion matrix. Because there are too many data, it is not easy to see the correctness of each group if it is displayed in a matrix, so this experiment sets a threshold to determine the text color. If the value of the cell is greater than the threshold, the text color is white, otherwise it is black. Since 99 different types of labels cannot fit in a single image, one label per ten groups is chosen to be displayed.

DOI: 10.54254/2755-2721/39/20230582

4.2. Random Forest

4.2.1. Model creation and training. A random forest classifier model is first created and seed value is set to 114514, which is consistent with the previous preprocessing. Then, the model is fitted using the training data.

4.2.2. Model prediction and evaluation. After the fitting was completed, the validation data set is predicted using the trained model, and the accuracy and logarithmic loss, etc. are calculated. In addition, the confusion matrix js introduced to further evaluate the performance of the model.

4.2.3. Visualization of confusion matrix. In the same way as KNN, a function to plot the confusion matrix is defined,and used to visualize the confusion matrix. Among them, the itertools library provided related functions for efficient handling of iterators and loops.

4.3. Classification using ANN

4.3.1. Preprocessing of the training and validation sets. The StandardScaler function is used to scale the features to a standard normal distribution. At the same time, the class labels are converted to integer encoding. After that, the to_categorical function is used to convert the integer encoded category labels into one-hot encoding, which is conducive to the training of the multi-category classification model.

4.3.2. Define the neural network model. The model built has three fully connected layers, where the first layer has 1024 nodes, the second layer has 512 nodes, and the final layer has 99 nodes (corresponding to 99 different categories of leaves in the database). Each of these layers is followed by a Dropout layer, here to prevent overfitting problems. In the last layer, a softmax activation function is used so that the output of the model could be interpreted as the probability of each category.

4.3.3. Compile and train the model. Set the loss function to classification cross entropy, the optimizer as RMSprop, and the evaluation index as accuracy. Then set the batch size to 128 and iterated 100 cycles.

4.3.4. Image rendering and data calculation. First, the loss and accuracy curves in the training process are plotted through the matplotlib library. Then the validation set is predicted, and the prediction results are converted from one-hot encoding back to integer encoding. After completing the previous operation, the classification report of the model on the verification set is computed and printed, which includes the accuracy rate, recall rate, and F1 score for each category.

4.4. Classification using CNN

4.4.1. Import the required libraries and modules. In addition to the regular library, here also added Sequential, Dense, Dropout, Flatten and other content, for the subsequent construction of convolutional neural networks.

4.4.2. Reshape the training set and validation set. Group the feature data by shape, texture, edge and store it in a 3D array for inclusion in the model. That is, the correlation between features can be utilized better by this operation to improve the performance of the model. After that, the to_categorical function is used to encode the category labels of the two sets of data separately in one-hot coding.

4.4.3. Define the convolutional neural network model. The convolutional neural network model is constructed, including adding convolutional layer, activation function, pooling layer, fully connected layer, etc., and setting loss function, optimizer and evaluation index.

DOI: 10.54254/2755-2721/39/20230582

4.4.4. Compile and train the model. The model is complied using the model. pile function and the loss function, optimizer, and evaluation metrics are set as defined in the ANN.

4.4.5. Image rendering and data calculation. After recording the loss and accuracy in the training process, the change trend can be more clearly seen by drawing the change curve of the loss and accuracy in the training process. After that, the validation set is predicted using the trained model and the classification report is printed.

5. Result

5.1. KNN algorithm

The accuracy on the validation set is 97.9798%, which shows that the model makes good predictions on unseen data. log loss is used to measure the performance of the classification model. The closer to 0, the better the performance of the model. In Figure 5, the white points are basically on the diagonal, which indicates that the trained model can identify the verification set well. However, there are still four white dots not on the diagonal, which means that the corresponding label of these points has been misclassified.

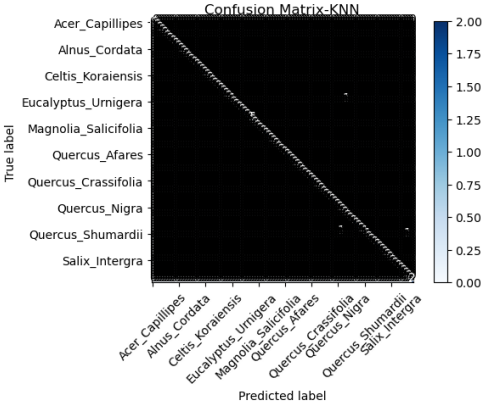

Figure5. Confusion matrix obtained by KNN.

5.2. Random Forest algorithm

The accuracy of the validation set is also 97.9798%, which shows that the algorithm can effectively classify the leaves. The value of log loss is 0.7514, which is very close to 0 overall. It can be seen from the Figure 6 that most of the leaves can be correctly classified, while a small number of points have certain deviations.

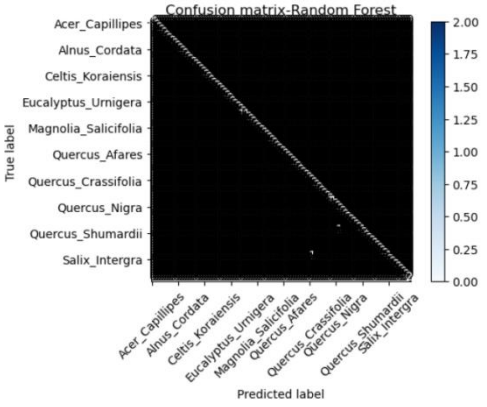

Figure 6. Confusion matrix obtained by Random Forest.

DOI: 10.54254/2755-2721/39/20230582

5.3. Classification result of ANN algorithm

In Figure 7, with the increase in the number of iterations, the value of the loss function continues to decrease, and finally approaches 0. The reduction of the loss function is a good sign, but it also needs to be careful to prevent overfitting and ensure that the model has good generalization ability.

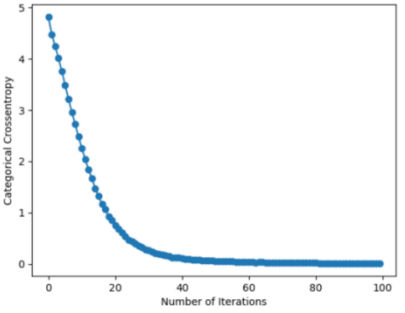

Figure 7. Train Error vs Number of Iterations (ANN).

Figure 8 claims that the ANN algorithm has a good effect on data fitting, the orange line and the blue line almost coincide. Although some overfitting occurs, this does not particularly affect the result.

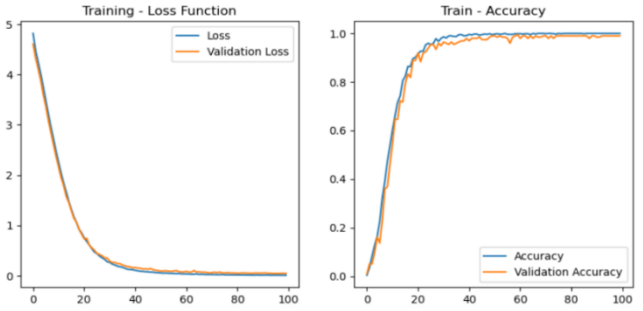

Figure 8. Graph of Loss Function and Accuracy (ANN).

The operation results show that the accuracy of ANN algorithm is as high as 0.9949, which indicates that the ANN algorithm can accurately assign the leaves to the 99 group. Of course, there are some categories that are not particularly accurate, and the current suspicion is that due to certain groups of leaves be so similar to other categories in terms of shape, color, or texture, the model may have a hard time distinguishing them.

5.4. Classification result of CNN algorithm

The author set the number of iterations to 30. If increased, the results are not significantly helpful and overfitting may occur. Figure 9 shows that the curve first rapidly declines and then slowly approaches 0. This indicates that error will gradually decrease with the increase of the number of iterations within a certain range.

DOI: 10.54254/2755-2721/39/20230582

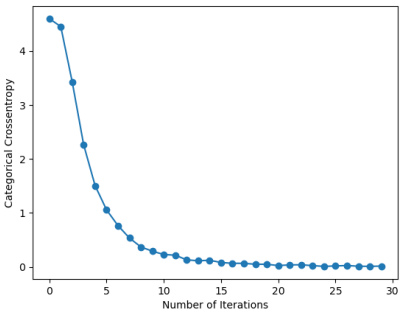

Figure 9. Train Error vs Number of Iterations (CNN).

Overall, based on the Figure 10, the curves are basically consistent, but from this graph, I find that the curve of the verification set is somewhat overfitted, which may be caused by less data.

Figure 10. Graph of Loss Function and Accuracy (CNN).

Furthermore, the test result regarding accuracy, recall and F1 scores show that the accuracy is 0.95.

6. Result analysis

6.1. Accuracy comparison

Based on the above results, it can be observed that the verification accuracy of the four algorithms exceeds 95%, indicating their effectiveness in the leaf classification tasks. However, there are differences among the algorithms.

The algorithms are divided into two categories for comparison analysis. The first category is KNN and Random Forest algorithms, both of them achieve the same accuracy, but their log loss values differ, with the former being lower than the latter. Additionally, in terms of computational time, KNN outperforms Random Forest. This may be because Random Forest adopts the concept of ensemble learning, which involves training a series of individual learners, potentially resulting in a more complex process. On the other hand, KNN is suitable for classification tasks with smaller datasets, provided that the dataset includes sample instances and their corresponding classes.

The second category of algorithms, namely ANN and CNN, has significant differences in accuracy. The ANN algorithm achieves a high accuracy of 0.9949, while the CNN algorithm only achieves 0.9495. By comparing and reviewing various curves, it was discovered that the CNN algorithm exhibited a high training accuracy but a low validation accuracy. This indicates that the algorithm suffered from overfitting during the classification process, resulting in a lower overall accuracy. In the definition of the model, the Dropout layer is used, which is a common strategy to prevent overfitting. The Dropout layer randomly "turns off" a subset of neurons during training so that the model cannot over-rely on any subset of specific input features, thus enhancing the model's generalization ability. After adjustment and

DOI: 10.54254/2755-2721/39/20230582

optimization, the results obtained by the CNN algorithm are still not as good as ANN. Too high number of iterations will more easily cause the overfitting of the CNN algorithm.

Overall, although the accuracy of the four algorithms may be the same, they do not identify the wrong kind of leaf. One of the algorithms may misclassify Quercus_Palustris, while another may misclassify Salix_Intergra. However, the results obtained by ANN algorithm training are the best, and the classification results obtained by Ann algorithm have the highest accuracy among the four algorithms.

6.2. Summary of the research problem

Firstly, the algorithm for leaf classification can obtain a high accuracy rate, which can make at least 95% of the leaves correctly classified. In a certain range, with the increase in training data, the accuracy of verification set will gradually improve.

Moreover, when classifying leaves, it is necessary to preprocess the data accordingly. Otherwise, the accuracy will decrease. Here we can remove some unhelpful data, such as the ID column in this experiment. When using various algorithms, we also need to preprocess data according to their respective classifiers, such as the data reshape operation in CNN.

Also, complex algorithms such as Random Forest and CNN may not be suitable for some lower dimensional data sets, or they may take a long time to compute. For higher-dimensional leaf datasets, such as adding color modules, these algorithms are better.

Then, simple algorithms such as KNN and ANN are also highly experimental. As long as we continue to find the most appropriate correlation values and optimize the algorithm in time, we can get better answers. For example, I used cross-validation in KNN to get the best K value and introduced the idea of weighting, which improved the accuracy by 10%.

Finally, the existence of overfitting will make accuracy drop, and we need to take measures to reduce its influence. Take CNN as an example, although it can reduce the error to near 0 after about 15 iterations, it can never approach 0. However, the increase in iterations will make the curve fluctuate more.

7. Conclusion

This article uses 4 different algorithms to classify leaves and compare the classification results. Through experiments, it is concluded that the classification results obtained by ANN have a higher accuracy among these algorithms. However, this article does not fully control the variables and the visualized results of the relevant algorithms. While effort is made to ensure the consistency of relevant inputs, the priority when researching on the problem is the accuracy of the results. Secondly, although the article optimized the code of each algorithm, the accuracy rate of CNN algorithm is always stuck at about 95%, which may be caused by the problem of convolutional neural network model or compilation model.

In the future, computers can be used to improve relevant algorithms to perform functional identification of leaves, such as analyzing the particulate matter content in leaves, to find more similar leaves that can monitor pollution.

References

[1] Gutiérrez P D, Lastra M, Bacardit J, et al. GPU-SME-kNN: Scalable and memory efficient kNN and lazy learning using GPUs[J]. Information Sciences, 2016, 373: 165-182.

[2] Duda R O, Hart P E, Stork D G, et al. Pattern classification, chapter nonparametric techniques[J]. 2000.

[3] Zhang Z. Introduction to machine learning: k-nearest neighbors[J]. Annals of translational medicine, 2016, 4(11).

[4] Akar Ö, Güngör O. Classification of multispectral images using Random Forest algorithm[J]. Journal of Geodesy and Geoinformation, 2012, 1(2): 105-112.

[5] Sagi O, Rokach L. Ensemble learning: A survey[J]. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2018, 8(4): e1249.

[6] Krawczyk B, Minku L L, Gama J, et al. Ensemble learning for data stream analysis: A survey[J]. Information Fusion, 2017, 37: 132-156.

DOI: 10.54254/2755-2721/39/20230582

[7] Bauer E, Kohavi R. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants[J]. Machine learning, 1999, 36: 105-139.
[8] Agatonovic-Kustrin S, Beresford R. Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research[J]. Journal of pharmaceutical and biomedical analysis, 2000, 22(5): 717-727.
[9] Sardogan M, Tuncer A, Ozen Y. Plant leaf disease detection and classification based on CNN with LVQ algorithm[C]//2018 3rd international conference on computer science and engineering (UBMK). IEEE, 2018: 382-385.
[10] Shankar V V, Kumar V, Devagade U, et al. Heart disease prediction using CNN algorithm[J]. SN Computer Science, 2020, 1(3): 170.
[11] Ahmadi M, Vakili S, Langlois J M P, et al. Power reduction in cnn pooling layers with a preliminary partial computation strategy[C]//2018 16th IEEE International New Circuits and Systems Conference (NEWCAS). IEEE, 2018: 125-129.