Citrus Disease and Pest Recognition Algorithm Based on Migration Learning

Kangshun Li[1(✉)], Miaopeng Chen[1], Juchuang Lin[1], and Shanni Li[2]

[1] College of Mathematics and Informatics, South China Agricultural University, Guangzhou 510642, China
likangshun@sina.com
[2] Digital Grid Research Institute, China Southern Power Grid, Guangzhou 510660, China

Abstract. Citrus is the largest fruit production in the world. Owing to the damage by various pest diseases, the production of citrus is reduced and the quality is getting worse and worse every year. The recognition and control of the citrus diseases are very important. By now the main measures we take to control them is sowing pesticides, which is not good for the environment and do harm to the soil greatly. The technology of image identification can recognize what kind of citrus disease they have with high efficiency and low cost, which is also environmentally friendly and is not limited by time and space. It is our top priority to apply it to recognize and prevent the disease from citrus. In order to detect citrus pest disease and control them automatically, we studied the pests and traits of citrus leaves and their multi-fractal characteristics and methods for figuring pests and diseases, and created a model for detecting leaf images of citrus. We use Keras and Tensorflow to build the model. To reduce recognition loss and improve accuracy, we put the citrus photos into the model and train it persistently. After examining, the recognition accuracy of citrus greening disease of 120 images can reach 96%. The experimental result shows that the model can recognize citrus diseases with high accuracy and robustness.

Keywords: Image recognition · Machine learning · Deep learning · Convolutional neural network · Migration learning · VGG16 model

1 Introduction

There are many precedents of plant disease recognition at home and abroad. In 2017, Zhao et al. [1] adapted the Otus threshold segmentation algorithm to extract 4 kinds of diseased potato leaves images and extract underlying visual feature vectors. She used the SVMc classifier, and the recognition rate is 92%. In 2018, Shi JiHong [2] tried to combine traditional database with service of WeChat public platform, focused on agricultural disease and pest recognition and realize image database construction based on WeChat public account which provided the users a convenient query, identify and disease prediction platform. Sharada et al. [3] trained a deep convolutional network to detect 26 kinds of diseases of 14 kinds of plants, its classification accuracy reached 99.35% in 54306 training disease photos. And it highlights the importance of deep learning and convolutional network.

K. Li et al. (Eds.): ISICA 2019, CCIS 1205, pp. 3–20, 2020.
https://doi.org/10.1007/978-981-15-5577-0_1

Rastogi A et al. [4] proposed a universal system for leaf disease identification, the first stage is based on feature extracting and artificial neural network recognition, the second stage is based on Kemans segmentation and ANN disease classification.

All the above researches show the hot topic of plant disease recognition based on computer vision combined with the popular interconnecting devices. But the recognition with high accuracy and low response time is the guarantee for the promotion of the identification technology. This paper researches on the recognition algorithms based on migration learning, and finds its great advantage in the recognition of citrus disease and pest.

2 Deep Learning and Migration Algorithm

2.1 Machine Learning

Machine learning specializes in how computer simulates or realizes human learning behavior to acquire new knowledge or skills, and how to reorganize knowledge structures and continuously develop its performance. It is central to artificial intelligence and the base to make computers intelligent. Machine learning mainly refers to that computer acquires knowledge from experience(data) and we could deem it as figuring out patterns and then learned from it. And machine learning is also called pattern recognition [5].

Automatically learning from data instead of following certain rules is a data analysis method or technique, and experience-based learning is the focus of machine learning. Machine learning is not programmed to perform a task, but programmed to learn to perform a task [6]. Machine learning can be divided into supervised learning, semi-supervised learning, and unsupervised learning in the light of to what extend it is manually intervened. The division basis of supervised learning and unsupervised one is whether their input data needs labels. The algorithms of supervised learning mainly contain classification and regression, while unsupervised learning's is clustering. Artificial neural network abstracts the neural networks in human brain into certain models according to different linking mode from an informatic processing standpoint. And it is a computing paradigm which is composed of a large number of nodes, also known as artificial neural units, connecting to each other. The model may contain several layers, which could be grouped into input layer, hidden layer and output layer. Each layer could have lots of neural units in it. The neural unit is depicted in Fig. 1.

The regular present of a neural unit is shown in Fig. 1. Each neural unit in the network has its input and output. The unit gathers its input from other neural units in the preceding layer to form the weighted output.

Z is the weighed input of neural unit i (n is the total number of the unit's inputs; w_n represents the bias of the unit, and its weight is 1):

$$Z = \sum_{i=1}^{n} x_i \cdot w_i \tag{1}$$

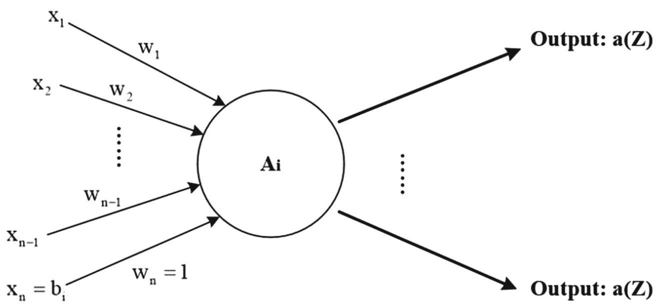

Fig. 1. Artificial neural unit

Then uses the weighted input Z as the input of its activation function, and finally outputs the result to other units.

A_i is the output of the neural unit i (a is the activation function of unit i):

$$A_i = a(Z) \tag{2}$$

2.2 Deep Learning

2.2.1 Deep Learning Network Structure

Machine learning could be divided into swallow learning and deep learning. Different from swallow learning which has only one hidden layer, deep learning has a lot of hidden layers. The overall structure of neural network is shown in Fig. 2.

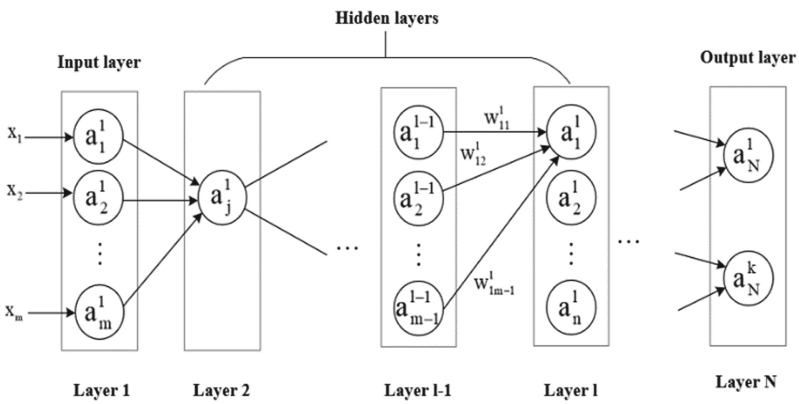

Fig. 2. Sequential neural network structure

The symbols definition of the sequential neural network is shown in Table 1.

Table 1. Sequential neural network symbols

Neural unit symbols	Symbol meanings
a_i^l	The output of unit i in layer l, if it represents the output of input layer, it can also be replaced by x_i
w_{ij}^l	The weight of the input from unit j in layer l-1 to unit i in layer l
b_i^l	The bias of the weighted input of unit i in layer l
z_i^l	The weighted input of unit i in layer l

2.2.2 The Forward Propagation Process

The units in the input layer get the inputs and use them as their output, the activation function of the units in the input layer can be described as a(x) = x, and the output of the first layer in the model(input layer, l = 1) can be depicted as, M is the total number of units in input layer:

$$a_i^1 = x_i(i = 1, 2, \ldots, M) \tag{3}$$

Each unit in the hidden layers receives other units' outputs in the preceding layer as its input if it's fully connected. And it uses the weighed input as the input of its activation function and gets the output.

The output of unit i in layer l, l > 1(m − 1 is the total number of units in layer l-1; a() is the activation function of the unit):

$$z_i^l = \left(\sum\nolimits_{j=1}^{m} a_j^{l-1} \cdot w_{ij}^l\right) + b_i^l \tag{4}$$

$$a_i^l = a\left(z_i^l\right) \tag{5}$$

Finally we get the outputs $a_1^N, a_2^N, \ldots, a_k^N$, assume there are k units in the output layer, we gather them as model's result. And the predict result is usually evaluated by the loss functions, some of which are listed below. $\widehat{y}$ represents the output of the model, y represents the input data's label, n is the total number of the input data. Common loss functions are shown as below:

squared error loss function

$$L(\widehat{y}, y) = \frac{1}{2}\sum\nolimits_{i=1}^{n} (y - \widehat{y})^2 \tag{6}$$

cross entropy loss function

$$L(\widehat{y}, y) = \sum\nolimits_{i=1}^{n} -y \cdot \log\widehat{y} - (1 - y) \cdot \log(1 - \widehat{y}) \tag{7}$$

Since the weighted input procedure presents the linear function and activation functions are usually nonlinear. Combined with linear and nonlinear function, the deep learning network could stimulate a large amount of transform in the world theoretically. The network itself is usually the approach of some kind of algorithm or function in nature, or some expression of strategy [7].

2.2.3 The Back Propagation Process

The model relies on the gradient decline algorithm, shown in formula (8) below, which is the mathematical basis of supervised learning model. The gradient descent method is a typical method which calculate the minimum values of the target function by slowly moving the point in the define domain to explore rather than finding the solution of the equation with partial derivatives equal to 0 [8]. w and b are weights and bias of the model, η is a small positive number which we call learning rate, Loss presents the loss function of the model, later we use L to represent it instead. w^* *and* b^* are the updated values of w and b.

$$(\Delta w, \Delta b) = -\eta\left(\frac{\partial Loss}{\partial w}, \frac{\partial Loss}{\partial b}\right)$$

$$(w^*, b^*) = (w + \Delta w, b + \Delta b) \tag{8}$$

We define neural unit error of unit i in layer l (l > 1) as below:

$$\delta_i^l = \frac{\partial L}{\partial z_i^l} \tag{9}$$

Neural unit error has relations with the gradient of weights and bias below. As long as we get the value of the neural unit errors of the layers from hidden layer to output layer, we could get the gradients of each parameter of the model easily.

$$\frac{\partial L}{\partial w_{ij}^l} = \delta_j^l \cdot a_i^{l-1}\left(\frac{\partial C}{\partial z_j^l} = \delta_j^l,\ \frac{\partial z_j^l}{\partial w_{ji}^l} = a_i^{l-1}\right) \tag{10}$$

$$\frac{\partial C}{\partial b_j^l} = \delta_j^l\left(\frac{\partial C}{\partial z_j^l} = \delta_j^l,\ \frac{\partial z_j^l}{\partial b_j^l} = 1\right) \tag{11}$$

The neural unit error in output layer, shown as below (N represents the output layer, a() is the activation function of unit i in the output layer, there are k units in the output layer):

$$\delta_i^N = \frac{\partial L}{\partial z_i^N} = \frac{\partial L}{\partial a_i^N} \cdot a^{'}\left(z_i^N\right)\,(i = 1, 2, \ldots, k) \tag{12}$$

And the neural unit errors in layer l and in layer l-1 has relations, shown as follows (m, n are the total number of units in layer l-1 and layer l; a() is the activation function of unit j in layer l-1):

$$\delta_j^{l-1} = \sum\nolimits_{i=1}^{n} \frac{\partial C}{\partial z_i^l} \cdot \frac{\partial z_i^l}{\partial a_j^{l-1}} \cdot \frac{\partial a_j^{l-1}}{\partial z_j^{l-1}} = \sum\nolimits_{i=1}^{n} \delta_i^l \cdot w_{ij}^l \cdot a^{'}\left(z_j^{l-1}\right)(j = 1, 2, \ldots, m) \quad (13)$$

The back propagation algorithm of sequential network model:

1) Get the output of the model, count the neural unit error in the output layer according to formula (12).
2) If the preceding layer is not the input layer, use formula (10), (11) to count the gradients of the weights and bias of this layer, turn to step 3; else turn to step 4.
3) Use formula (13) to count the neural unit errors in the preceding layer, turn to step 2.
4) Get all the gradients of the parameters in the model, then update the parameters according to formula (8).

2.3 Migration Learning

Migration learning is a sort of machine learning which refers to adapting a pretrained model to another recognition task. The migration learning refers to the migration from the original task and data to the target task and data, using the weight parameters in the original data domain to improve the predictive function of the target task [9]. It can efficiently reduce the over-fitting degree of the normal convolution neural network.

3 Citrus Pest and Diseases Identification Based on Deep Learning and Migration Learning

3.1 Problem Description

At present, the fruit plantation area reaches 1130 thousand hectares, among which citrus's occupy 266 thousand, in Guangdong province, China. And citrus is the main type of fruit in Guangdong. Due to the numerous citrus diseases, the planting area of citrus in Guangdong reduces by more than 30% and the diseases causes the direct economic loss of about 4 billion yuan each year. Therefore the recognition of citrus diseases is of great importance. In the past, people used the convolutional neural network (CNN) model to distinguish the disease citrus from healthy one. The convolution network can better solve the problem that it's hard to find an appropriate feature to train because of the citrus leaves' great similarity, and we don't have to choose features manually for the training, CNN is capable of learning from the original 2D photo. And It can extract new features of the input photos as well as renewing the features it learned persistently. But CNN also has drawbacks like overfitting.

This paper focus on the sick recognition of citrus, including citrus greening disease, Citrus canker disease and Citrus Anthracnose. We try to build a recognition model with the CNN and later we build the model based on migration learning methods.

3.2 The Structure of Convolutional Neural Network

Convolutional neural network is a kind of feed forward neural network. CNN's network structure which is local-sensitive and weight-sharing makes it more like a biological neural network, and it decreases the number of neural units' weights, and largely reduces the complexity of the model.

CNN can be divided into convolutional layer, sampling layer, flatten layer, and fully connected layer. The structure is shown in Fig. 6. The first three layers form the basic unit for CNN.

3.2.1 Convolutional Layer

Convolutional layer is used to extract the 2-dimension feature of the photo, using different kernels to detect different edge of the interest region. C is short for convolutional layer, which is used to extract features from the picture. Different from other neural network, CNN uses a matrix to store the values of the weights, which is defined as the convolutional kernel. When the layer gets the input photo data, it will use the kernel to scan the photo from the up-left side to the right-bottom side, and each step of the convolution will generate a new pixel in the output feature map according to the linear transformation between the kernel and the region of photo it cover. The schematic diagram of convolution is shown in Fig. 3.

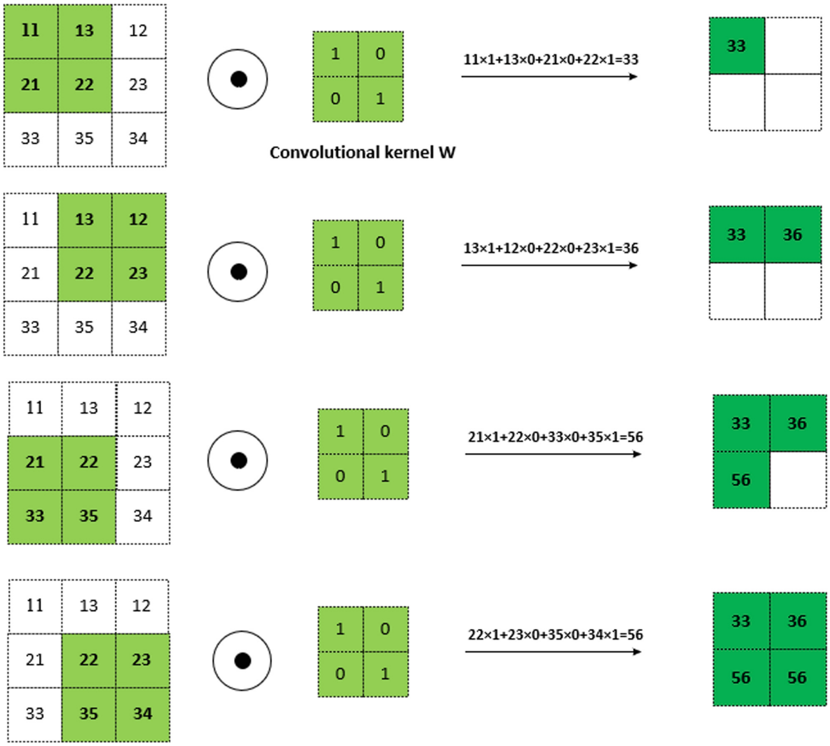

Fig. 3. Schematic diagram of convolution with kernel size (2 × 2), input size (3 × 3)

3.2.2 Sampling Layer

Sampling layer mainly aims at reducing the size of the feature map and extract the primary features. Through the process of sampling, the number of parameters drops. Sampling layer execute the pooling function which select the max, min, average value of the region it covers as its output. The sampling filter does not store any weights like the convolutional kernel. The schematic diagram of max-pooling is depicted in Fig. 4.

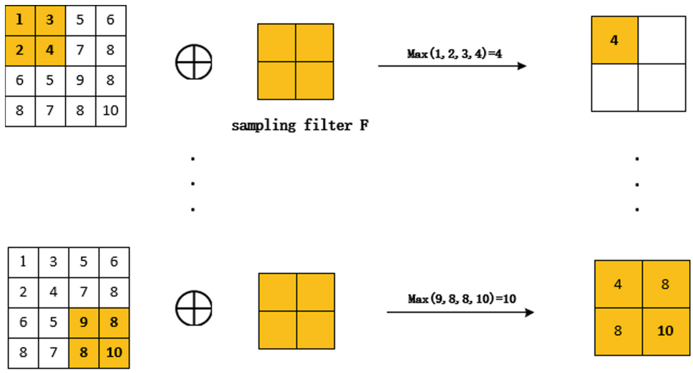

Fig. 4. Schematic diagram of max-pooling with filter size (2 × 2), input size (4 × 4)

3.2.3 Dropout Layer and Flatten Layer

The usage of dropout layer is to disable some neural units randomly, preventing the model from overfitting and gradient vanishing. CNN use the dropout function to pick neural units in the hidden layer randomly and disable them during the training process. Because the disabled units could not transmit the signal forward, it averts the overfitting problem effectively [10].

Flatten layer is designed to flatten the output result of sampling layer, it's usually connected between the parts of feature extraction and pattern recognition in the model.

3.2.4 Fully Connected Layer

Fully connected means each neural unit in the layer are fully connected to the preceding layers' units. Convolutional layer, activation layer and sampling layer are the layers to extract features, and the duty of fully connected layer is to integrate the features and ready for classification and identification procedure [11]. It is just like the neural network in Fig. 2.

3.3 The Forward Propagation of Convolution Neural Network

3.3.1 Symbols Definition of the Convolutional Neural Network

CNN's symbols definition is shown in Table 2.

Table 2. Convolutional neural network symbols

CNN layers	Layer's symbol	Symbol meaning
Input layer	x_{ij}	It represents the input in row i, column j
	a_{ij}^{I}	It represents the output of the input layer in row i, column j
Filter of CNN	w_{ij}^{Fk}	It represents the weight in row i, column j in the filter in CNN's sublayer k
Convolutional layer	z_{ij}^{Fk}	It represents the weighted input of the neural unit in row i, column j in CNN's sublayer k
	b^{Fk}	It represents the bias of the neural unit in CNN's sublayer k
	a_{ij}^{Fk}	It represents the output of the neural unit in row i, column j in CNN's sublayer k
Sampling layer	z_{ij}^{Pk}	It represents the input of the sampling layer in row i, column j in CNN's sublayer k
	a_{ij}^{Pk}	It represents the output of the sampling layer in row i, column j in CNN's sublayer k
Flatten layer	a_{i}^{F}	It represents the output of the flatten layer of unit i
Output layer	w_{ij}^{O}	It represents the weight in output layer which is from neural unit j in the flatten layer to unit i in the output layer
	z_{n}^{O}	It represents the weighted input of the nth neural unit in the output layer
	b_{n}^{O}	It represents the bias of the unit n in the output layer
	a_{n}^{O}	It represents the output of the unit n in the output layer

3.3.2 The Input Process of the CNN

The input process of CNN is shown in Fig. 5, and the relation of input layer is depicted below, in formula (15) (we use the input size of (5, 5) as an example):

$$a_{ij}^{I} = x_{ij} \tag{15}$$

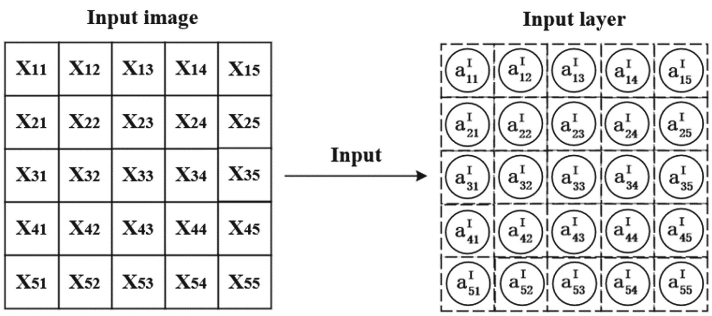

Fig. 5. Input process of the convolutional neural network

3.3.3 The Convolution Process of CNN

The convolution process of CNN is shown in Fig. 6, and the relations in convolutional layer are depicted below, assuming that the CNN has 3 sublayers (W_{in}, H_{in} are the input size of the convolutional layer, depicted as W_i, H_i below; W_{out}, H_{out} are the output size of the convolutional layer, depicted as W_o, H_o below; W_f, H_f are the kernel's size, P_w, P_h are the padding number of the output feature map, $stride_w, stride_h$ are the strides of the kernel in the horizontal direction and vertical direction; $a_f()$ represents the activation of the convolutional layer):

$$W_{out} = \frac{W_{in} - F_w + 2 * P_w}{Stride_w} + 1 \tag{16}$$

$$H_{out} = \frac{H_{in} - H_w + 2 * P_h}{Stride_h} + 1 \tag{17}$$

$$c_{ij}^{Fk} = \sum\nolimits_{k=1}^{W_f} \sum\nolimits_{l=1}^{H_f} w_{kl}^{Fk} * a_{i+k-1j+l-1}^{I} (i = 1, 2, \ldots, W_o; j = 1, 2, \ldots, H_o) \tag{18}$$

$$z_{ij}^{Fk} = c_{ij}^{Fk} + b^{Fk} (i = 1, 2, \ldots, W_o; j = 1, 2, \ldots, H_o) \tag{19}$$

$$a_{ij}^{Fk} = a_f\left(z_{ij}^{Fk}\right) \tag{20}$$

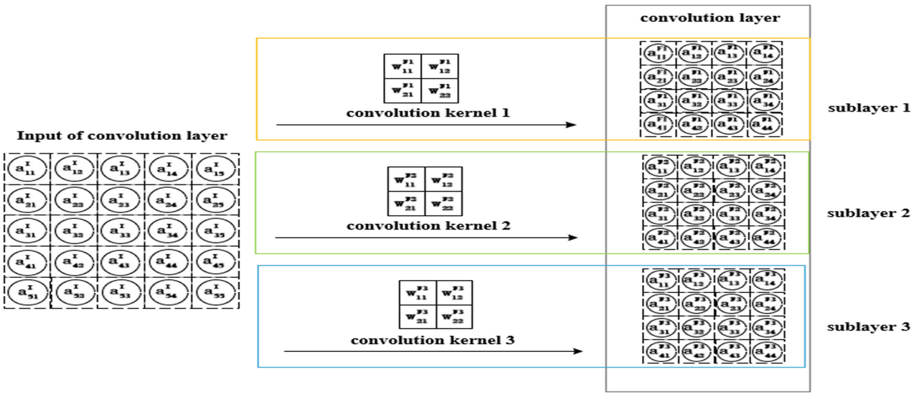

Fig. 6. Convolution process of the convolutional neural network

3.3.4 The Sampling Process of CNN

The max-pooling process of CNN is shown in Fig. 7, and the relations in sampling layer are depicted below (W_{in}, H_{in} are the input size of the sampling layer, depicted as W_i, H_i below; W_p, H_p are the size of the sampling filter; W_{out}, H_{out} are the output size of the sampling layer, depicted as W_o, H_o below):

$$W_{out} = \frac{W_{in}}{W_p}, H_{out} = \frac{W_{out}}{H_p} \tag{21}$$

$$z_{ij}^{Pk} = Max\left(a_{kl}^{Fk}\right)(i = 1, 2, \ldots W_o; j = 1, 2, \ldots, H_o; k = 1, 2, \ldots, W_p; l = 1, 2, \ldots, H_p) \tag{22}$$

$$a_{ij}^{Pk} = z_{ij}^{Pk} \tag{23}$$

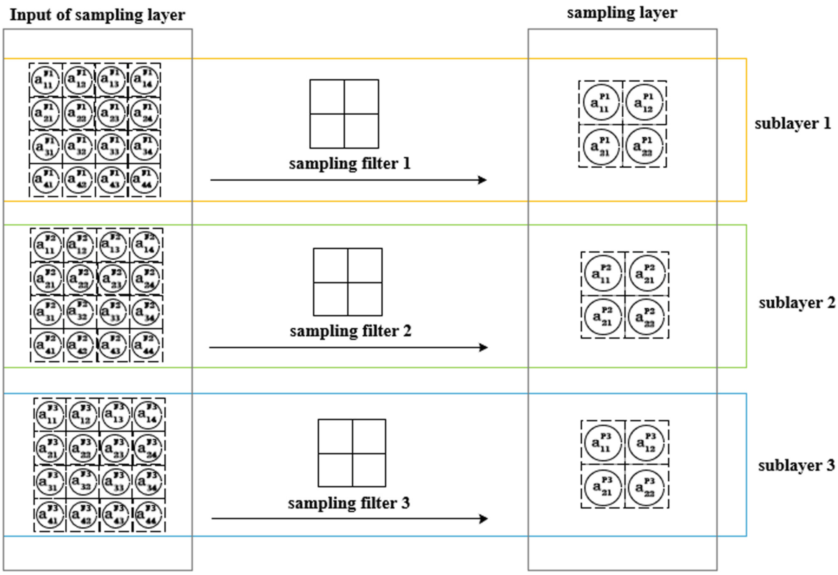

Fig. 7. Sampling process of the convolutional neural network

3.3.5 The Flatten Process and Output Process of CNN

The flatten and output process of CNN is shown in Fig. 8, and the relations in flatten layer and output layer are depicted below (W_{in}, H_{in} are the input size of the flatten layer, assume that M = $W_{in} \cdot H_{in}$ and there are K sublayers in the model, and there are N = M · K units in the flatten layer; there are O units in the output layer; a_o is the activation function of the output layer):

$$a_i^F = a_{jk}^{Pl}\left(i = 1, 2, \ldots, N; l = \frac{i}{M} + 1; j = \frac{i\%M}{W_{in}} + 1; k = i\%W_{in}\right) \tag{24}$$

$$z_i^O = \left(\sum\nolimits_{j=1}^{n} w_{ij}^O \cdot a_j^F\right) + b_i^O (i = 1, 2, \ldots, O) \tag{25}$$

$$a_i^O = a_o\left(z_i^O\right) \tag{26}$$

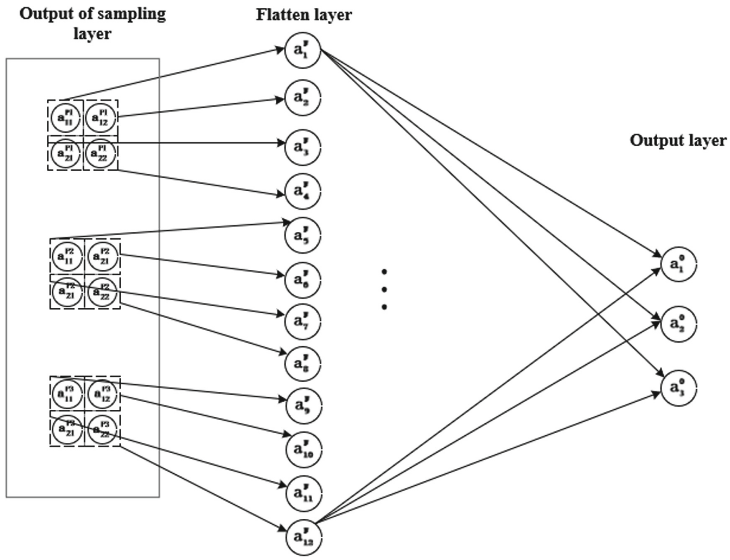

Fig. 8. Output process of the convolutional neural network

3.4 The Backward Propagation of Convolution Neural Network

The backward propagation algorithm of CNN is similar to other deep learning model, but the major difference is that convolutional layer's weights are in the convolutional kernels. The flatten layer and the subsequent layers backward propagation process is the same as deep learning model.

Define neural unit errors in the output layers and convolutional layers as below (δ_n^O means the neural unit error of output unit n; δ_{ij}^{Fk} means the neural unit error in row i, column j of convolutional layer in sublayer k):

$$\delta_n^O = \frac{\partial L}{\partial z_n^O} \tag{27}$$

$$\delta_{ij}^{Fk} = \frac{\partial L}{\partial z_{ij}^{Fk}} \tag{28}$$

Through the neural unit errors, it is easy to get the gradient of each parameter of the model. The relation between gradients and neural unit errors are shown below (W_f, H_f are the kernel's size; W_o, H_o are the output size of the convolutional layer):

$$\frac{\partial L}{\partial w_{ij}^O} = \delta_n^O \cdot a_j^F, \frac{\partial L}{\partial b_i^O} = \delta_n^O \tag{29}$$

$$\frac{\partial L}{\partial w_{ij}^{Fk}} = \sum\nolimits_{k=1}^{w_o} \sum\nolimits_{l=1}^{h_o} \delta_{kl}^{Fk} \cdot a_{i+k-1j+l-1}^{Fk} \left(i = 1, 2, \ldots, w_f; j = 1, 2, \ldots, h_f\right) \tag{30}$$

$$\frac{\partial C}{\partial b^{Fk}} = \sum\nolimits_{k=1}^{w_o} \sum\nolimits_{l=1}^{h_o} \delta_{kl}^{Fk} \tag{31}$$

Then we try to figure out the expressions of neural unit errors in output layer and convolutional layers:

$$\delta_n^O = \frac{\partial L}{\partial a_n^O} \cdot \frac{\partial a_n^O}{\partial z_n^O} = \frac{\partial L}{\partial a_n^O} \cdot a_o^{'}\left(z_n^O\right) \tag{32}$$

$$\delta_{ij}^{Fk} = \left\{\sum\nolimits_{i=1}^{O} \delta_i^O \cdot w_{ik}^O\right\} \cdot (v) \cdot a_{Fk}^{'}\left(z_{ij}^{Fk}\right) \tag{33}$$

v values 1 or 0 depend on whether a_{ij}^{Fk} is the largest number among the region of the filter of the convolutional layer F_k. Then we get all the neural unit errors and we could calculate all the gradients of the parameters in the convolutional network and update the parameters according to gradient decline algorithm.

3.5 Data Preprocessing

In order to reduce overfitting phenomenon, we extend the dataset by flipping the original photos horizontally and vertically and scaling randomly [12], the training photos are shown in Fig. 9. Then we set the plant images to (224, 224), and we divide the dataset according to the ratio of 5:1 into the training set and test set.

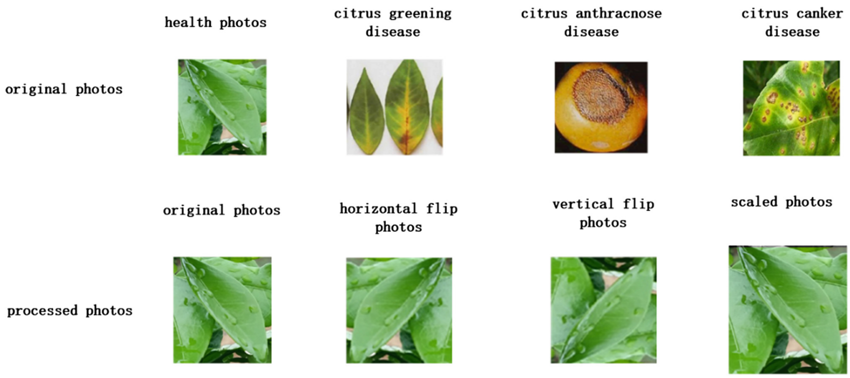

Fig. 9. Data preprocessing

3.6 Convolutional Network Model

The preliminary experiment is to build a convolutional network with 4 convolutional layers. Its structure is depicted in Table 3.

Table 3. Convolutional neural network structure

Layer (type)	Output shape	Parameter count
Convolution_1 (convolution)	(224, 224, 16)	448
Max_pooling_1 (Max pooling)	(112, 112, 16)	0
Convolution_2 (convolution)	(112, 112, 32)	4640
Max_pooling_2 (Max pooling)	(56, 56, 32)	0
Convolution_3 (convolution)	(56, 56, 64)	18496
Max_pooling_3 (Max pooling)	(28, 28, 64)	0
Convolution_4 (convolution)	(28, 28, 128)	73856
Max_pooling_4 (Max pooling)	(14, 14, 128)	0
Flatten_1 (Flatten)	(25088)	0
Dense_1 (Dense)	(512)	12845568
Dropout_1 (Dropout)	(512)	0
Dense_2 (Dense)	(11)	5643
Dropout_2 (Dropout)	(11)	0
Dense_3 (Dense)	(2)	24

We test the model in different situations, such as different convolutional kernel size and different learning rate. And the test results are shown in title 4.

3.7 Migration Learning Model

The preliminary experiment shows that the CNN has high accuracy rate on test dataset, but it can easily become overfitting. In order to solve this problem, we decide to alter our model based on the migration learning algorithm. We choose the VGG-16 model as the base of our recognition model. VGG model is a typical CNN with high classification and recognition rate. It increases the depth of the network steadily by adding more convolutional layers. And very small convolutional filters (3 * 3) make it work successfully [13].

3.7.1 The Structure of VGG16

The structure of VGG16 is shown in Fig. 10 below. There are totally 16 weighted layers in the model, 13 convolutional layers and the last 3 fully connected layers. Its basic unit is a convolutional layer followed by a sampling layer. The kernel size of convolutional layers are all 3×3, using relu as its activation function to train the model quickly. And the kernel of sampling layer is 2×2, adopting max polling.

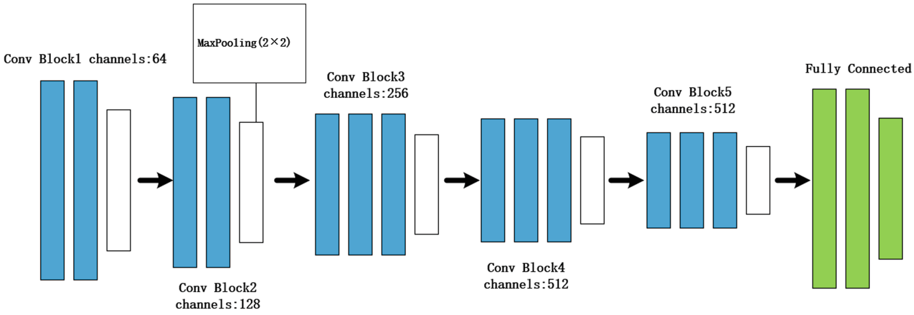

Fig. 10. Structure of VGG16

3.7.2 Fine-Tuning Method

We choose the fine-tuning method to construct our model, shown in Fig. 11. Fine-tuning means adjusting the model which trained by others to train our data. It can be seen as using the front layers of the original models to extract the features of the photos, and the newly add layers to classify [14]. The increase of network layers will not lead to the explosion of the parameters, because the parameters are mainly concentrated in the last three fully connected layers [15]. As in Fig. 16, the layers before the fully connected layers can not be trained, only the last three layers can be trained and its parameters could be updated. Since the parameters in convolutional layers are untrainable, the time for training the model is less and it's with high efficiency.

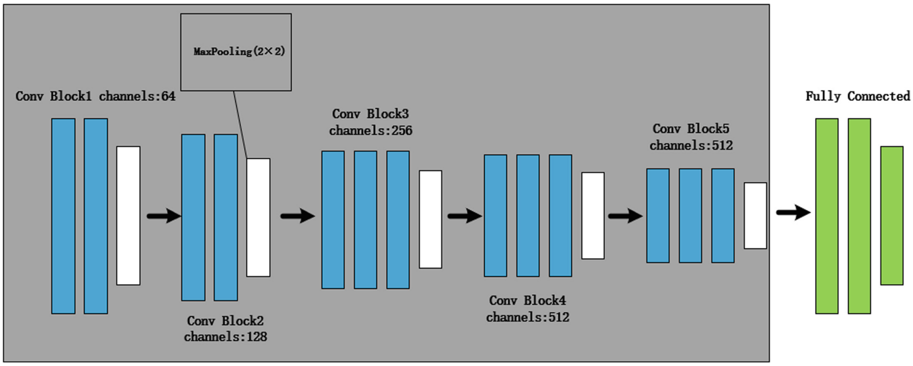

Fig. 11. Structure of VGG16 with fine-tuning

3.7.3 Migration Model Construction

Our migration model is shown in Table 4, we remove the top three FC layers and add a flatten, dropout and 2 dense layers for training.

Table 4. Migration model design

Layer (type)	Output shape	Parameter count
Vgg16 (Model)	(7, 7, 512)	14714688
Flatten_1 (Flatten)	(25088)	0
Dense_1 (Dense)	(1024)	25691136
Dropout_1 (Dropout)	(1024)	0
Dense_2 (Dense)	(2)	2050

4 Experimental Simulation and Analysis

The three citrus diseases are cgd (short for citrus greening disease), cad (short for citrus anthracnose disease) and ccd (short for citrus canker disease), and the experiment result are listed as below.

The test result of the simple convolutional network is shown in Table 5.

Table 5. Test result of simple convolutional network

Dataset	Train number	Train accuracy	Test number	Test accuracy
cgd	250	79%	50	67%
cad	200	78%	40	71%
ccd	180	82%	35	65%

It seems that a simple CNN could have train accuracy of 80% on average and test accuracy of 67%. But the problem of overfitting is still serious.

Then we change the kernel size to find out its influence on the accuracy, using dataset of citrus greening disease as training data. The result shows in Table 6.

Table 6. Test result of simple CNN with different kernel size

Dataset	Train number	Train accuracy	Test number	Test accuracy	Kernel size
cgd	250	83%	50	76%	(3, 3)
cgd	200	79%	40	75%	(5, 5)
cgd	180	78%	35	72%	(7, 7)

The test result shows that convolutional kernel size values (3, 3) gets optimize results, the main reason is that a smaller kernel can extract more local features, and develop the accuracy of citrus disease classification.

Then we alter the learning rate of the model to find out its influence on the accuracy, using dataset of citrus greening disease as dataset. The result is shown in Table 7.

Table 7. Test result of simple CNN with different learning rate

Dataset	Train number	Train accuracy	Test number	Test accuracy	Learning rate
cgd	250	76%	50	62%	0.01
cgd	200	77%	40	63%	0.001
cgd	180	83%	35	64%	0.0001

Through the result we find that the learning rate 0.0001 gets the best result, learning rate's value is important for the construction of the model. If it is too small, the convergence time would be long. If it is too big, it may cause oscillating around the minimal value.

At last we test the accuracy of these diseases on the migration model based on the fine-tuning VGG16. The result is shown in Table 3, it's clear that the recognition model based on migration learning with fine-tuning method has higher accuracy both on the test dataset and the validation dataset than the simple CNN. The test result of citrus disease recognition is shown in Table 8.

Table 8. Test result of fine-tuning convolutional network

Dataset	Train number	Train accuracy	Test number	Test accuracy
cgd	250	97%	50	93%
cad	200	98%	40	95%
ccd	180	98%	35	92%

5 Conclusion

Image recognition is a technology that dependents on deep learning. It can be done through extracting different level of features, from the insignificant characteristics next to the input end to the abstract features like the semantic of the photo near the output end. In our paper, in order to find the better disease recognition model of citrus, we study the algorithm of convolutional network and the model based on migration learning, the result of the experiment shows that a transfer model based on migration learning can ease overfitting, increase recognition accuracy and has a wider range of applications.

Acknowledgements. This work is supported by National Key R&D Program of China with the Grant No. 2018YFC0831100, the National Natural Science Foundation Youth Fund Project of China under Grant No. 61703170, the National Natural Science Foundation of China with the Grant No. 61773296, Foreign Science and Technology Cooperation Program of Guangzhou with the grant No. 201907010021, the Open Foundation of Key Lab of Data Analysis and Processing of Guangdong Province in Sun Yat-sen University with No. 201901, the Major Science and Technology Project in Dongguan with No. 2018215121005. Key R&D Program of Guangdong Province with No. 2019B020219003, Foreign Science and Technology Cooperation Program of Huangpu District of Guangzhou with No. 2018GH09 Technology Cooperation Program of Huangpu District of Guangzhou.

References

1. Zhao, J., Xue, X., Li, Q.: Potato disease recognition system based on machine vision. J. Jiangsu Agric. Sci. **45**(02), 198–202 (2017)
2. Shi, J.: Database design of agricultural pest and disease identification system based on WeChat public account. Hebei Agric. Mach. **11**, 15–16 (2018)
3. Mohanty, S.P., Hughes, D., Salath, M.: Using deep learning for image-based plant disease detection. (3), 42–48 (2015). Digital Epidemiology Lab, EPFL, Switzerland
4. Rastogi, A., Arora, R., Sharma, S.: Leaf disease detection and grading using computer vision technology & fuzzy logic. In: International Conference on Signal Processing and Integrated Networks, pp. 500–505. IEEE (2015)
5. Peng, L.: Deep Learning Practice Computer Vision. Tsinghua University Press, Beijing (2019)
6. Lin, H., Liu, T.T., Li, H., et al.: Prospects for machine learning research and its application in agriculture. Inf. Inst. Chin. Acad. Agric. Sci. **31**(10), 13 (2019)
7. Lin, H., Liu, T.T., Li, H., et al.: Prospects for machine learning research and its application in agriculture. Inf. Inst. Chin. Acad. Agric. Sci. **31**(10), 15 (2019)
8. Wakui, Y., Wakui, S.: Mathematics for Deep Learning. Yang Y.L., trans. Posts & Telecom Press (2019). 84
9. Liu, H.: Study on the automatic detection of tomato diseases and pests based on leaf images. Dalian JiaoTong University, Dalian (2017). 37
10. Qiu, N., Wang, X., Wang, P., et al.: Research on convolutional neural network algorithm combined with transfer learning model. Comput. Eng. Appl. **56**(5), 45 (2020)
11. Ran, D.: Research and implementation of pest detection algorithm and mobile client based on convolutional neural network. Anhui University, Anhui (2018). 10
12. Zuo, Yu., Tao Qian, W., Lian, W.Y.: Research of plant classification methods based on convolutional neural network. Internet of Things Technol. **10**(03), 72–75 (2020)
13. Zeng, G.: Fruit and vegetables classification system using image saliency and convolutional neural network. IEEE Beijing Section, Global Union Academy of Science and Technology, Chongqing Global Union Academy of Science and Technology, Chongqing Geeks Education Technology Co., Ltd. In: Proceedings of 2017 IEEE 3rd Information Technology and Mechatronics Engineering Conference (ITOEC 2017). IEEE Beijing Section, Global Union Academy of Science and Technology, Chongqing Global Union Academy of Science and Technology, Chongqing Geeks Education Technology Co., Ltd. IEEE BEIJING SECTION (2017). 615
14. Liu, H.: Study on the automatic detection of tomato diseases and pests based on leaf images. Dalian JiaoTong University, Dalian (2017). 39
15. Duan, C.: Image classification of fashion-MNIST data set based on VGG network. In: Proceedings of 2019 2nd International Conference on Information Science and Electronic Technology (ISET 2019). International Informatization and Engineering Associations: Computer Science and Electronic Technology International Society (2019). 19