**(12) INNOVATION PATENT**
**(19) AUSTRALIAN PATENT OFFICE**

(11) Application No. **AU 2019100352 A4**

(54) Title
**A Flower-Image Search System Based on Deep Learning**

(51) International Patent Classification(s)
***G06K 9/62*** (2006.01) ***G06N 3/08*** (2006.01)
***G06F 16/953*** (2019.01) ***G06N 20/00*** (2019.01)
***G06N 3/04*** (2006.01)

(21) Application No: **2019100352** (22) Date of Filing: **2019.04.04**

(45) Publication Date: **2019.05.09**
(45) Publication Journal Date: **2019.05.09**
(45) Granted Journal Date: **2019.05.09**

(71) Applicant(s)
**YUFAN TAN;YUANBO YANG;ZICHENG DUAN;JINGWEN WU;JIE LIANG;YIXUAN YU**

(72) Inventor(s)
**TAN, YUFAN;WU, JINGWEN;YANG, YUANBO;DUAN, ZICHENG;LIANG, JIE;YU, YIXUAN**

(74) Agent / Attorney
**Gloria Li, 65b Hattaway Ave, Buckland Beach, Auckland, 2012, NZ**

## ABSTRACT

The invention patent specifically designs a flower image recognition system based on deep learning and neural network learning. First, images can be acquired by web crawler from the Internet as many as possible. Second, the searched images are divided into 2 parts: the train section and the test section. The train section is used for training deep learning architecture while the test section is used for test performance. Third, the train section will be input into the system which contains several layers of convolutional neural networks. By adjusting the parameters from the neural learning network such as learning rate, drop-out rate, and decay rate, the system will finally reach the optimal performance. In this case, users simply upload an image and the system will input it into corresponding structures based on the features of the image. Finally, the system will automatically return some similar images to the users. This invention can be applied to other image recognition areas such as mood-detecting and vehicle recognition.

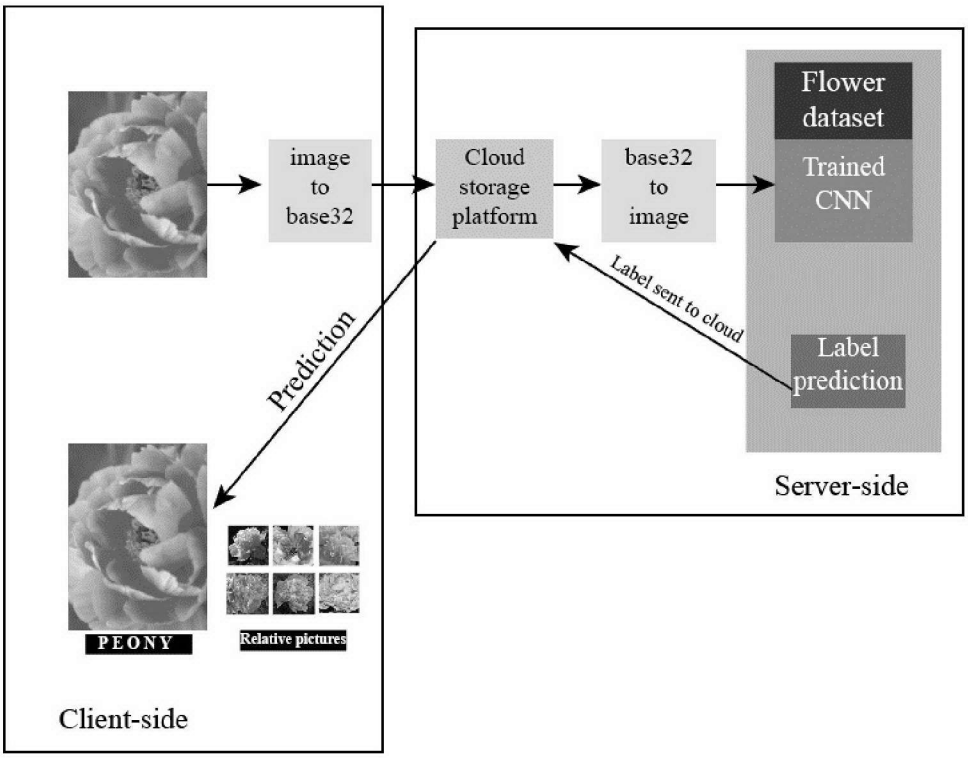



Figure 12

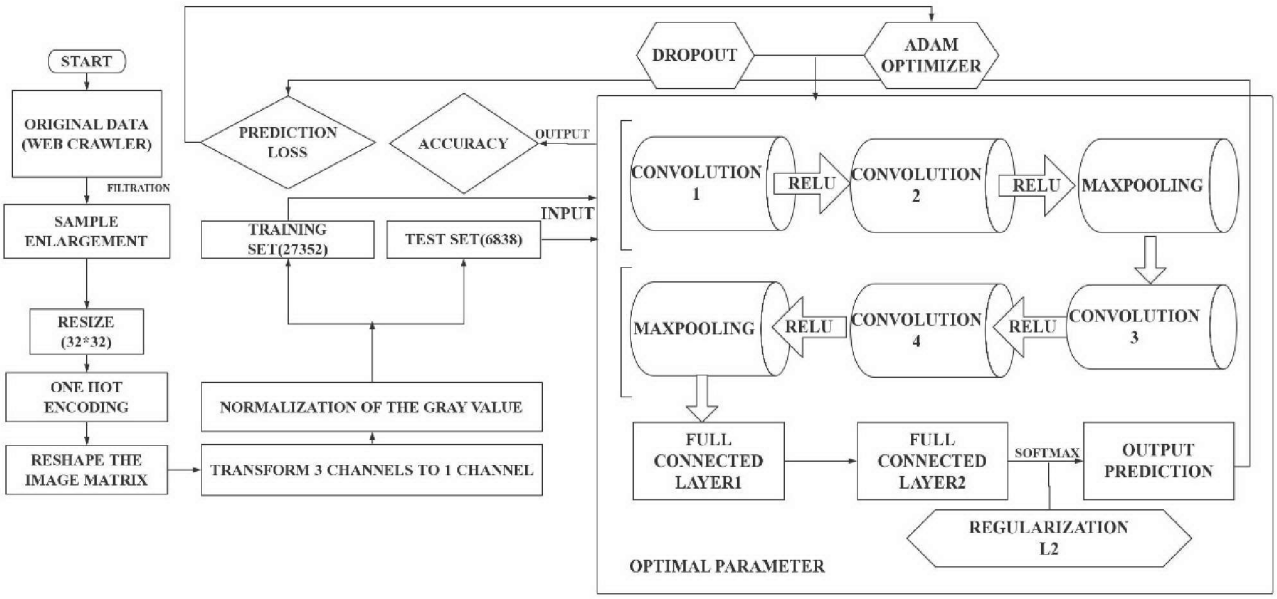



Figure 13

# DESCRIPTION

## Title

A Flower-Image Search System Based on Deep Learning

## Field of The Invention

This invention lies in the field of image processing field, which specifically involves flower-image recognition system based on deep learning and neural network learning.

## Background

The first method that used to search for qualified training sample is to download relevant data sets which have already been processed by former users from the Internet. The second method is to crawl images from the Internet, which means acquiring images with the use of crawling programs. The working process of the program has been divided into several steps. The crawling program will decode the URL of the goal first, then return the list of websites which contain the required images. Next, JSON is used to store the images' URL from the search engine, for loop is used to go through the JSON file and

resolve the acquired image URL, the images would be downloaded by the program and finally stored in local disk. The images of Peony, Lotus, White Rose, Narcissus, Daisy and Morning Glory of more than 2,000 each. However, not all of them are qualified for the research in terms of accuracy. Images that are not qualified (e.g. a woman called Rose) should be eliminated. After the elimination, each type of the training samples is nearly 2,000 images.

## 1. Solutions of Sample Imbalance

Several simple methods are introduced with respect to dealing with over-fitting and under-fitting of images.

1) Images are rotated or symmetrized for various steps. Firstly rotate 90 degrees and save the images in the same folder. Secondly, symmetrize all and save. Thirdly rotate 37 degrees and save. The purpose for multiple rotation and symmetry operations is to enlarge and to standardize the data set. The final size of one type of plants is 16000.

2) Adding noise to a specific percent of images represents the simulation of the unpredictable interference happens in picture taken in real life. Randomly selected images which take 25% of all pictures have added salt-and-pepper noise (i.e. 3% salt, 3% pepper

per image), otherwise, if the amount of negative sample is superior to the positives', the amount of them cannot be more than positive sample images. The negative sample images can play a comparative role, making classification boundaries more precise.

3) Despite the images acquired from the Internet, pictures taken by cameras also own 10%, which are used to satisfy the requirements and additionally ensure the practicality of the data set.

## 2. Resize of Images

First, python script is used to resize the origin images. Recognizing origin images requires a large bunch of time, therefore images are resized to $32 \times 32$, making machine learning efficient. For each type of flowers, assuming $P$ training samples are contained in the training set and $T$ testing samples are contained in the testing set, the distribution ratio of the two sets is set to

$$T=\frac{1}{4}P$$

(i.e. $P$=12800 per type, $T$=3200 per type). The training set is used to train the model, and testing set is to test the accuracy of the output model.

## 3. Design of Model Structure

Our model is based on the tandem of a 4-layer Convolution Neural

Network and a 2-layer Fully Connected Layer.

Some basic parameters are needed to be declared before constructing the Neural Network:

1) The Height or Width of the input image (the input should be a square image, e.g. $32 \times 32$), denoted as $W$.

2) In the convolutional neural network, the matrix of the input layer corresponding one-to-one to an element in the output result of a certain layer and the matrix is called Receptive Field, the width or height of which is denoted as $F$.

3) The variable used to determine the size of steps of the receptive field window sliding when processing convolution, named as Stride, denoted as $S$.

4) The number of the zero-padding, denoted as $P$.

5) Filters, which represent convolution kernels in single convolution network.

6) The number of filters, also known as the depth of the output of one convolution layer, denoted as $K$.

The number of Filter($K$) is determined by $W, F, S, P$ using the formula (2).

***Convolutional Layer 1***

As introduced before, the input image size is $32 \times 32$, meaning that $W = 32$, and manually, $F$ is set to 3, $P$ is set to 1, therefore the number of filters($K$) and the depth is 32, i.e., the shape of the input image data is $[32 \times 32 \times 32]$.

***Convolutional Layer 2***

The output data from the first layer is set as the input for the second layer, which is a set of pixel data with the shape $[32 \times 32 \times 32]$, using the same steps as the first convolutional layer, convoluted by $[3 \times 3 \times 32]$ convolution kernel, the output remains to be $[32 \times 32 \times 32]$. After the process of convolution and ReLU, the output data will then be put into a Max-pooling layer.

***Max-Pooling Layer 1***

Max-pooling is a sample-based discretization process. The objective is to down-sample an input representation (image, hidden-layer output matrix, etc.), reducing its dimensionality and allowing for assumptions to be made about features contained in the sub-regions binned.

This is used to in part to help reduce over-fitting by providing an abstracted form of the representation. As well, it reduces the computational cost by reducing the number of parameters to learn and provides basic translation invariance to the internal representation.

In the first max-pooling layer, the $[32\times 32\times 32]$ data is transformed into $[16\times 16\times 32]$.

***Convolutional Layer 3***

The output of the max-pooling layer 1 is the input of the convolutional layer， more details and features are gathered in this layer, the shape of the data remains $[16\times 32\times 32]$, ReLU is applied to this convolutional layer.

***Convolutional Layer 4***

The output of the third convolutional layer is the input of this layer, which is the final convolutional layer of the CNN, providing the final output of the convolution to the Max-pooling Layer 2, ReLU is applied to this convolutional layer.

***Max-pooling Layer 2***

This is the final layer of the convolutional module, the data size is further reduced, the input shape is $[16\times 16\times 32]$. After max-pooling, the output is $[8\times 8\times 32]$. After this step of max-pooling, the output will be transferred to the Fully-Connected layer, used as the training data.

## 4. Illustration of Training Process

2019100352 04 Apr 2019

After four convolutional layers with two max-pooling, it then enters a two-fully-connected-layer model. The image matrix is reshaped from $[batch \times 8 \times 8 \times 32]$ to $[batch \times 2048]$.

The first fully-connected layer's weights $W_1$ are set to be a variable of the normal distribution of matrix $32 \times out\_num\_nodes$ (i.e. the value of $out\_num\_nodes$ could be 32, 64, 128…). The biases $b_1$ of it is variables between the number of a matrix of 0.1 to $out\_num\_nodes \times 1$. The output fully-connected layer's weights $W_2$ and biases $b_2$ could also be calculated this way. By using the formula below, the inputs of this process could be calculated and tested with the origin data sets. (i.e. $y$ represents the output of the nodes, $x$ represents the input of the nodes. $w$ and $b$ represent the weight and bias, and l represents the layer of the nodes.) The formula is formula (5) and (6).

The average loss is calculated through python. The accuracy rate and recall rate is calculated and output to show the performance of the model, listed as formula (14) and (15).

(i.e. P represents the accuracy rate, $R$ represents recall rate. $TP$ represents situations when positive samples are predicted as positive samples, $FP$ represents situations when negative samples are predicted as positive samples and $FN$ represents when positive samples are predicted as negative samples.)

The output is the high-level feature of the image. SoftMax function is used to classify the input images according to the probability of labels. The relevant formula is formula (7).

$V_i$ represents the $i$-th elements of a matrix V, then the SoftMax value of this element is $S_i$.

## 5. Optimization of Model Training

1) Loss function

Regularization is adopted aiming at optimizing the weights and biases for the fully-connected layer. for loop is applied, add every weight and bias together and get regularization. Then multiply regularization with lambda (i.e. here lambda is defined as 5e-4). Then the outcome is added to loss make the optimization solution stable and fast. Listed as formula (8).

2) Drop out

To optimize the model, drop-out throws away some nodes in the fully-connected layer at a given rate during the training proce

The left part of figure 10 represents a model without drop out, and the right picture represents a model adding the drop out. Let $dropout\_rate$ be 0.99 (i.e. this element could be changed to

optimize the learning model), the activation value of a neuron stops working with this probability. During the training process, it does not update the weights, nor does it participate in the calculation of the neural network. Nevertheless its weight has to be retained (won't be updated for the time being) because it may have to work again next time the sample is entered.

3) Parameter Update

The parameters are updated through methods, such as gradient, momentum, Adam. Their decay rate and step are defined by the given learning rate $\eta$. Learning rate $\eta$ is optimized automatically by MATLAB's API tf.train.exponential_decay. Adam is set as default optimize method to update weights and biases. Adam is based on gradient descent method. The following formula shows gradient descent method's way to update the parameters.

$$\theta = \theta - \eta \times \nabla(\theta).J(\theta) \tag{14}$$

$\eta$ represents learning rate, $\nabla(\theta).J(\theta)$ represents the gradient of loss function.

Adam estimation uses independent adaptive learning rates to calculate the first moment estimation and second moment estimation of the gradient. It dynamically adjusts the learning rate

for each parameter based on the first-order moment estimate and the second-order moment estimate of the gradient of each parameter based on the loss function. The formula of it is listed as formula (9).

(m represents first moment, s represents second moment, $\nabla_\theta J(\theta)$ represents the gradient of the loss function)

4) ReLU function

ReLU (Rectified Linear Unit) is selected as the activation function in this convolutional layer, ReLU refers to:

ReLU can alleviate gradient disappear problems and reduce the training time, greatly speeding up the rate of convergence of the model.

In convolutional layer 1, the data after ReLU processing owns the size of $32 \times 32 \times 32$, data shape will not be modified by ReLU.

## 6. Graphical User Interface Design

An GUI is designed for clients, shown as figure 7.7.1.

Clients could load a local *.png file or a *.jpg file into the software, by pressing "search", the software will transform the image into $32 \times 32 \times 32$ matrix, and send it to remote server, after transferring and

processing, the image will be sent to the trained CNN, which will finally give back the prediction and some related pictures from its remote database.

## Summary of The Invention

In order to improve the accuracy and the convenience of searching flower images, making the flowers image searching system smart, the system designed in this application proposes an image classification method for flowers that is based on deep learning. In this section, we will introduce all the key steps of the designing process about this system, including the pivotal details of preparing the flower images dataset or sample, the way of data processing, the model structure, the technical details, and the optimization.

## Description Of The Drawings

Fig. 1 illustrates the structure of data preparing.

Fig. 2 is the illustration of peony image preprocessing.

Fig. 3 illustrates the convolutional neural network.

Fig. 4 illustrates the structure of cnn.

Fig. 5 illustrates an example of convolution, in this example, the result '2'

in the second output volume is acquired from the sum of the sum of point multiplication of the input volume and its corresponding position in filters and biases, i.e., -1 + (-1) + 2 +(-2) + 2 + 2 + 2 + (-2) = 2.

Fig. 6 illustrates the ReLU.

Fig. 7 illustrates an example of Max-Pooling Layer, 20, 30, 112, 37 is selected to form the output.

Fig. 8 illustrates the structure of fully connected layer and output layer.

Fig. 9 illustrates the structure of optimization.

Fig. 10 illustrates an example of drop out.

Fig. 11 illustrates the gui for flower recognition software.

Fig. 12 illustrates the flow of the proposed system.

Fig. 13 illustrates the procedure of the project.

Fig. 14 illustrates the curve of accuracy and loss for the model.

Fig. 15 illustrates an example diagram of flower species recognition accuracy.

Fig .16 illustrates the performance of the final model.

## Description Of The Preferred Embodiment

### 1. Searching Methods of Training Sample

The methods are as follows:

1) Downloading relevant data sets from the Internet. These data sets that have already been processed by former users contain high-quality images, which are beneficial for our model training.

2) Crawling images from the Internet is the main method that used for preparing the training samples. A web crawler based on Python has been put into use.

3) Shooting a sufficient number of flowers in the field and deleting useless photos to make sure the samples are suited for the requirement of training.

**2. Solutions of Sample Imbalance**

In order to reduce the over-fitting or under-fitting during training, the original data can be processed by the following methods:

1) Making rotated, symmetrical, vertical images based on original images.

2) Adding noise to the copy of the original images.

3) Adding negative sample images with false or extra objectives.

**3. Resize the Images**

Each image should be resized as $m \times n$, and divide the dataset into

two groups, one is training sample set which contains $P$ samples and another is testing sample set which contains $T$ samples. The sample ratio is as:

$$T=\frac{1}{h}P \tag{1}$$

where $h$ represents the scale factor ($h>1$).

## 4. Design of Model Structure

Multi-layer Convolutional Neural Networks and Fully Connected Neural Networks in tandem are used as the network structure to train the model. This method significantly tackles the problems of feature extraction of flower images and automatic image recognition. This neural network consists of a sequence of layers, including an input layer, followed by several convolutional layers and fully connected layers, and finally is the output layer. The model is structured as

$$input \to [conv \times Q + mp] \times P + FC \times N \to output \tag{2}$$

where $input$ represents the input layer, $[conv \times Q + mp]$ represents the convolution operation, $conv$ denotes the convolutional layers and the number of them is $Q$, $mp$ represents the max pooling layer, and the network has the number of this structure of $P$. $FC$ represents the

fully connected layers and the number of them is $N$ $(1 \leq N \leq 2)$. *output* represents the output layer, the final result will be acquired from it.

1) Convolutional layer

The convolutional layer consists of several convolutional units, each of which is optimized by a backpropagation algorithm. Convolutional operation is similar to the template operation, using a filter as a set of weights with the size of $M \times M \times channel$ to slide through the surface of the former layer for computing the output for the next operation. The function of convolution layer is to extract different features from the input image. After the computation, one dimension in an outputted image size is calculated by the formula as:

$$G = \frac{W - F + 2P}{S} + 1 \quad (3)$$

where $W$ is the size of the input image, $F$ is the size of filter, $P$ represents the number of zero-padding and $S$ represents the

stride.

2) ReLU layer

ReLU layer is chosen (Rectified Linear Units Layer) as the nonlinear activation function of the network. The formula is listed below:

$$\max(0, x) = \begin{cases} x, x>0 \\ 0, x \leq 0 \end{cases} \tag{4}$$

3) Max Pooling layer

Max pooling performs a function of reducing feature dimensions and retaining significant features and hence reduce the complex computation in the network.

4) Fully connected layer

The fully connected layer mainly involves two functions of forward-propagation and back-propagation. The forward-propagation calculates for the output value of neurons, where its formula is

$$y = w^T x + b \tag{5}$$

where $y$ represents the output value, $x$ is the input of one neuron, $w$ represents the weight of one neuron, $b$ means bias. The back-propagation is used for updating the parameters of each layer after classification, where its formula is

$$\frac{\partial v}{\partial x} = w \cdot \frac{\partial v}{\partial y}, \frac{\partial v}{\partial w} = x \cdot \left( \frac{\partial v}{\partial y} \right)^T \tag{6}$$

and $v$ represents the layer of neurons. Fully connected layers are used for classification.

5) SoftMax layer

SoftMax function is applied to map the output of each neuron to the interval of (0,1), calculating and outputting the probability of which categories the input flower image belongs to. The formula of SoftMax is

$$Softmax(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}} \tag{7}$$

where $x_i$ is the output of one neuron.

## 5. Optimization of Model Training

To improve the accuracy of recognition and reduce the over-fitting while training, methods of optimization are taken into the design of

the network.

1) Regularization

We use $L2$-loss function to eliminate the over-fitting. The formula of $L2$-loss function is

$$S=\sum_{i=0}^{n}(y_i-h(x_i))^2 \tag{8}$$

The $L2$-loss function can minimize the sum of the squares (S) of the difference between the target value ($y_i$) and the estimated value ($h(x_i)$).

2) Dropout parameter

Aiming to reduce over-fitting, the dropout parameter is introduced in the fully connected layer while training. Dropout parameter will randomly discard several nodes with a given probability, which can prevent units from co-adapting too much and reduce over-fitting significantly.

3) Adam optimization

Adam (Adaptive Moment estimation) is also an algorithm based on one step to optimize random objective function. Adam is chosen as the optimization method of the network. The formula of it is

$$
\begin{aligned}
&q \leftarrow \beta_1 \cdot q + (1-\beta_1) \cdot \nabla_\theta J(\theta) \\
&s \leftarrow \beta_2 \cdot s + (1-\beta_2) \cdot \nabla_\theta J(\theta) \square \nabla_\theta J(\theta) \\
&q \leftarrow \frac{q}{1-\beta_1} \\
&s \leftarrow \frac{s}{1-\beta_2} \\
&\theta \leftarrow \theta - \frac{\eta}{\sqrt{s+\varepsilon}} \square q
\end{aligned}
\tag{9}
$$

and $q$ is the first moment, $s$ means the second moment and $\nabla_\theta J(\theta)$ is the gradient of the loss function, $\eta$ represents the learning rate.

4) Learning rate optimization

The learning rate is a hyperparameter that guides the system through how to adjust the network weights through the gradient of the loss function and control the speed of adjusting neural network weights based on loss gradient. Many optimization algorithms involve it. To obtain a high-quality result, exponential decay is used to optimize the learning rate. The formula of exponential decay is:

$$\eta = t \times d^{\frac{\hat{\partial}}{\lambda}} \tag{10}$$

where $\eta$ is the current learning rate, $t$ is the start rate, $d$ represents the decay rate of every round of learning ($0 < d < 1$), $\hat{\partial}$ represents the global steps and $\lambda$ means decay steps, $\lambda$ can be

expressed as:

$$\lambda = \frac{sample\_size}{batch\_size} \tag{11}$$

## 6. Illustration of training process

Training samples are used to train the model, and then use the test samples to acquire the accuracy of every batch of the model and calculate the average accuracy to judge the effect till reaching a set of optimal parameters and model. Accuracy rate and recall rate are used to evaluate the quality of the result, where accuracy rate illustrates how many predicted positive samples are real positive samples and recall rate illustrates how many positive samples in the original data are correctly predicted. The model of them is expressed as

$$P = \frac{TP}{TP + FP} \tag{12}$$

$$R = \frac{TP}{TP + FN} \tag{13}$$

where $P$ is accuracy rate and $R$ is recall rate, $TP$ represents the true positive rate, $FP$ represents the false positive rate, $FN$ represents the false negative rate.

## 7. Design of user system

An application system is necessary for better use of this application.

The details of the user system will be introduced in the description part.

## Claim

1. A flower-image search system based on deep learning, comprising:

first, images are acquired by web crawler from the Internet as many as possible;

second, the searched images are divided into 2 parts: the train section and the test section, wherein the train section is used for training deep learning architecture while the test section is used for test performance;

third, the train section will be input into the system which contains several layers of convolutional neural networks;

by adjusting the parameters from the neural learning network such as learning rate, drop-out rate, and decay rate, the system will finally reach the optimal performance;

wherein users simply upload an image and the system will input it into corresponding structures based on the features of the image, finally the system will automatically return some similar images to the users.

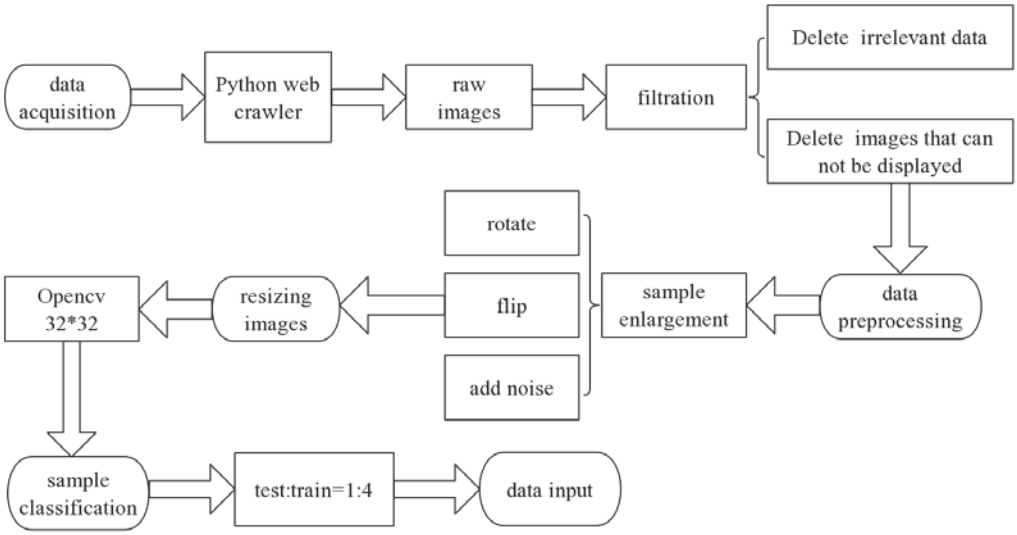
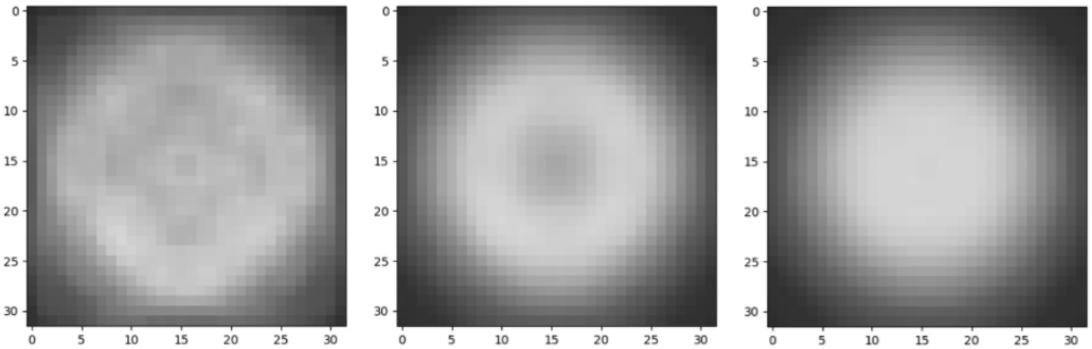



Figure 1

\



Note:Scale in Fig. means pitel.

Figure 2

2019100352 04 Apr 2019

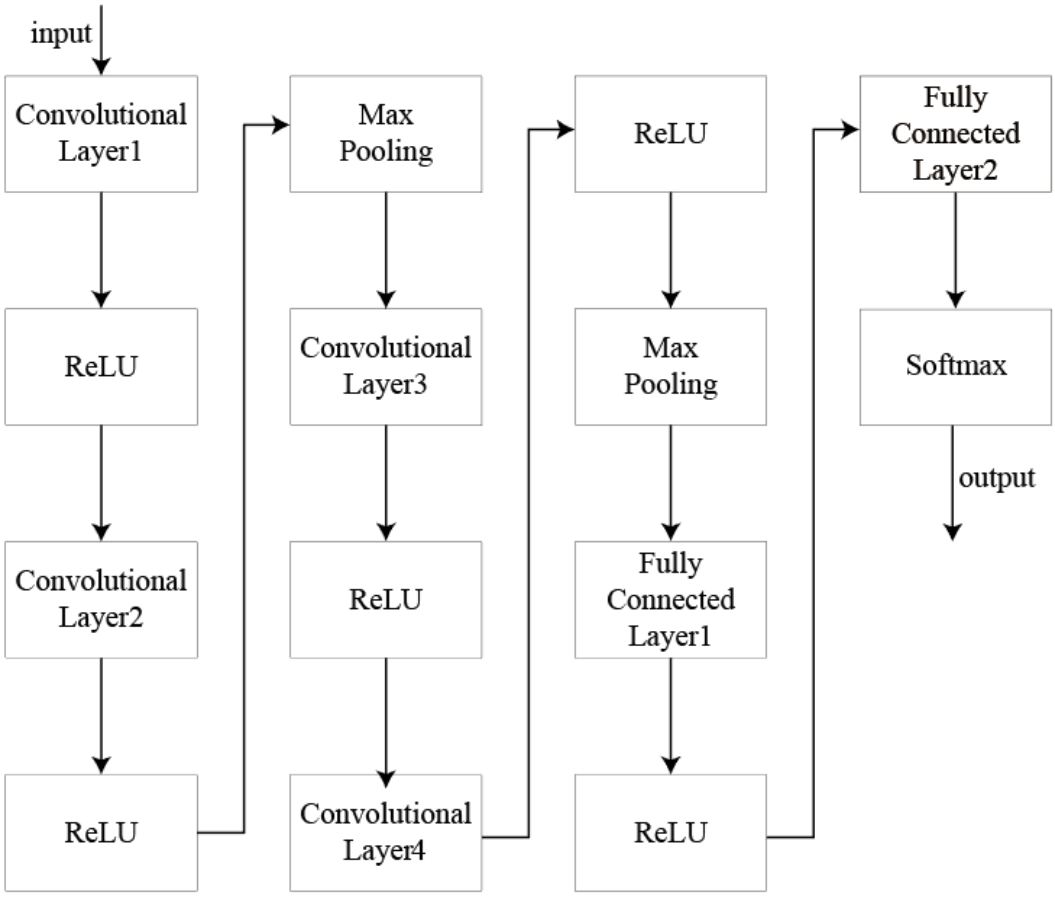



Figure 3

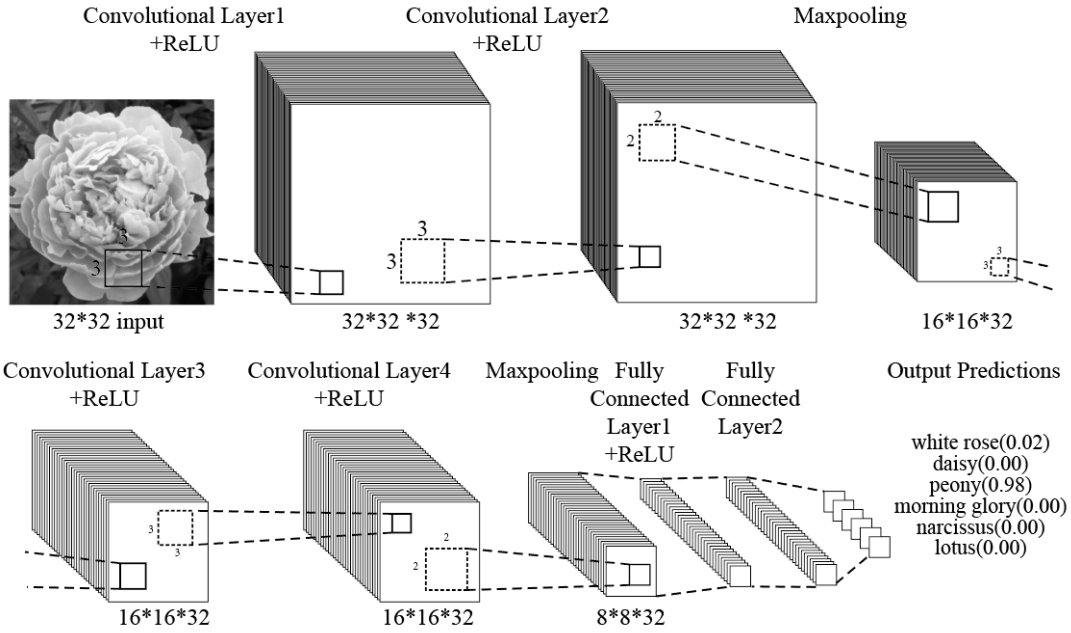



Figure 4

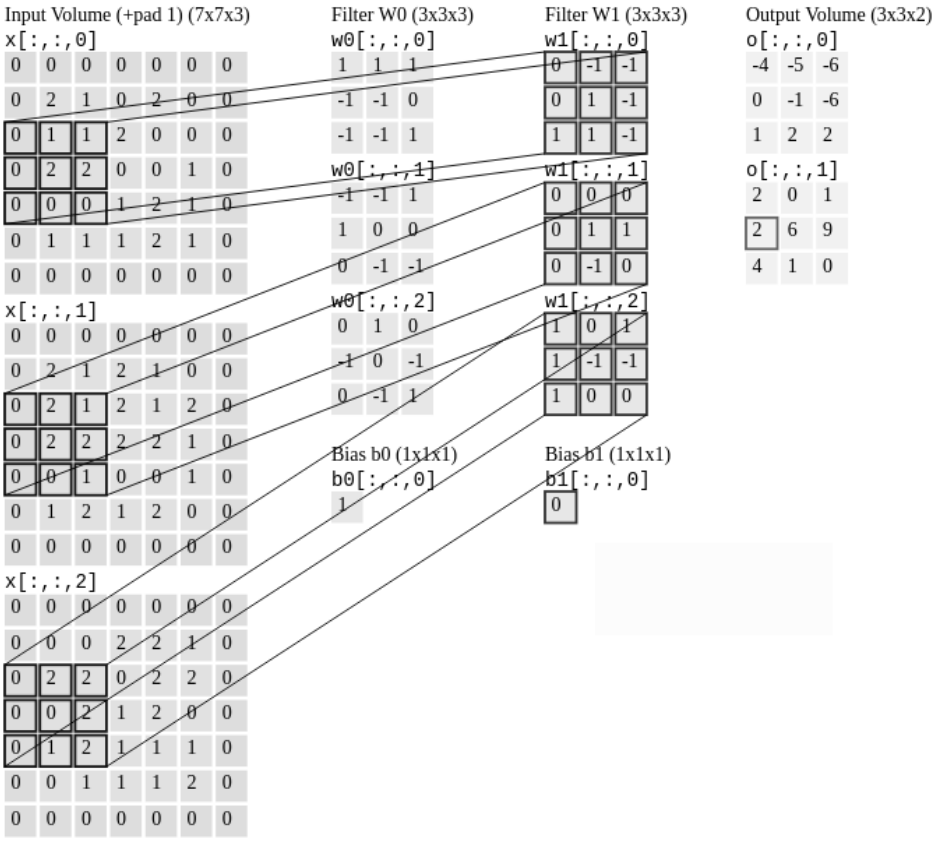



Figure 5

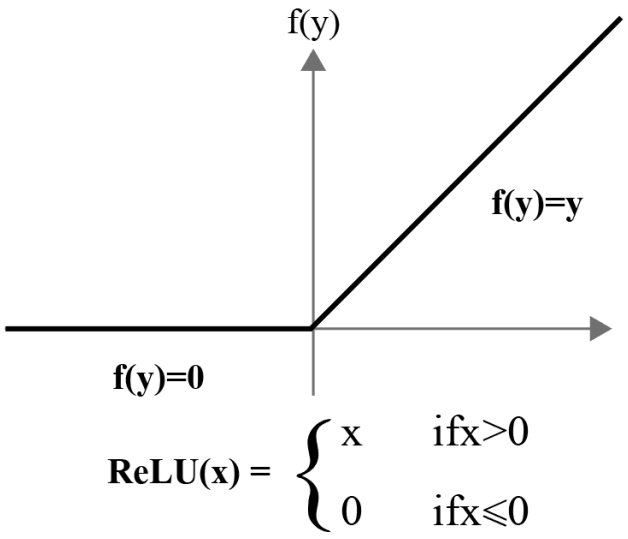



Figure 6

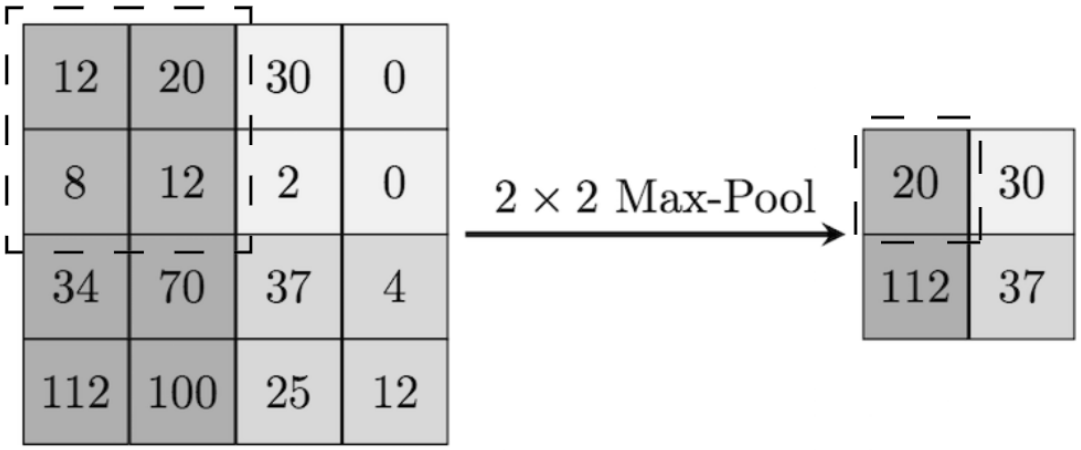



Figure 7

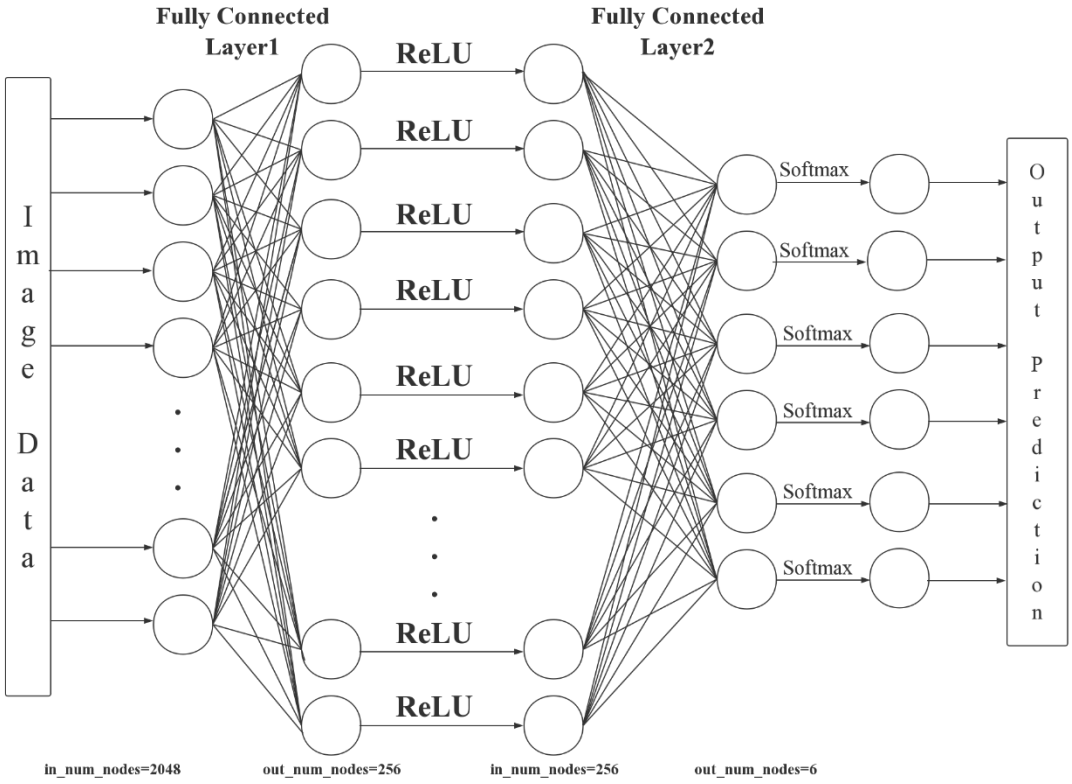
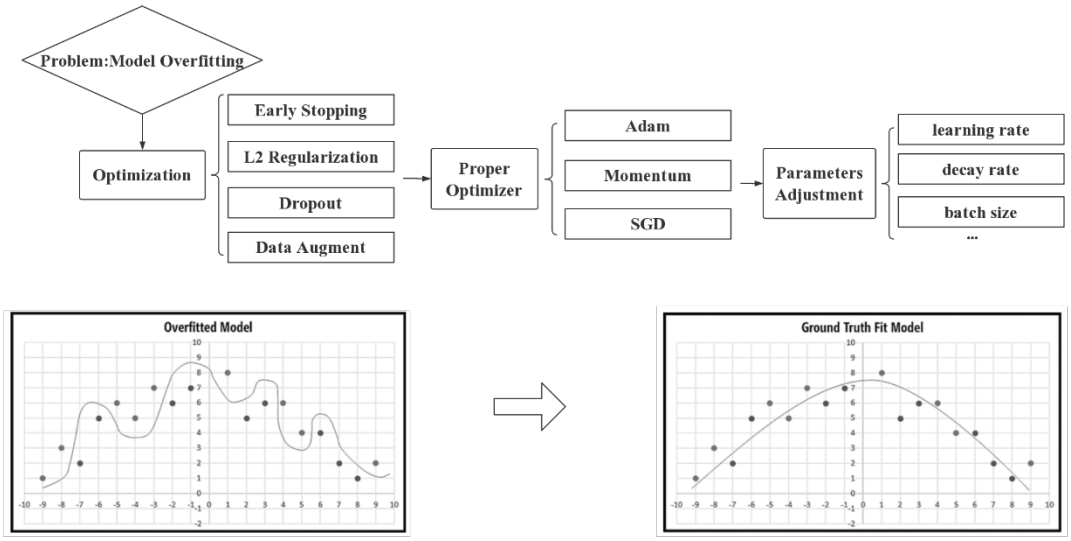



Figure 8



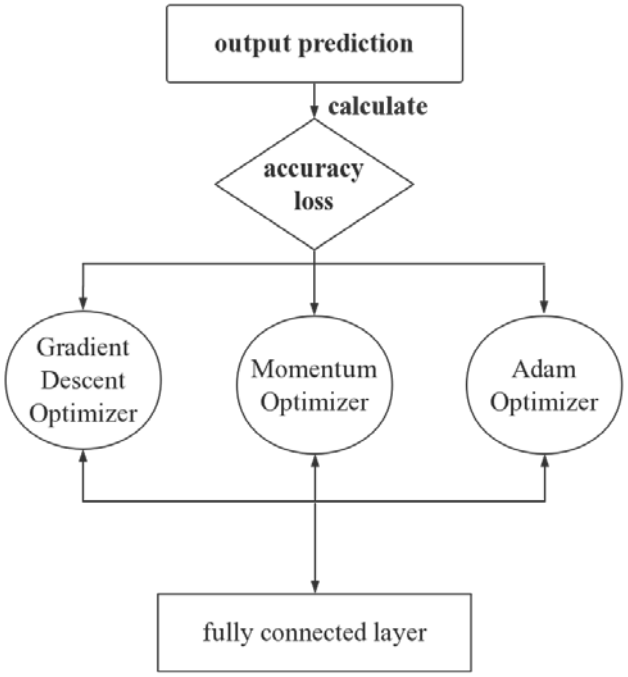




Figure 9

2019100352 04 Apr 2019

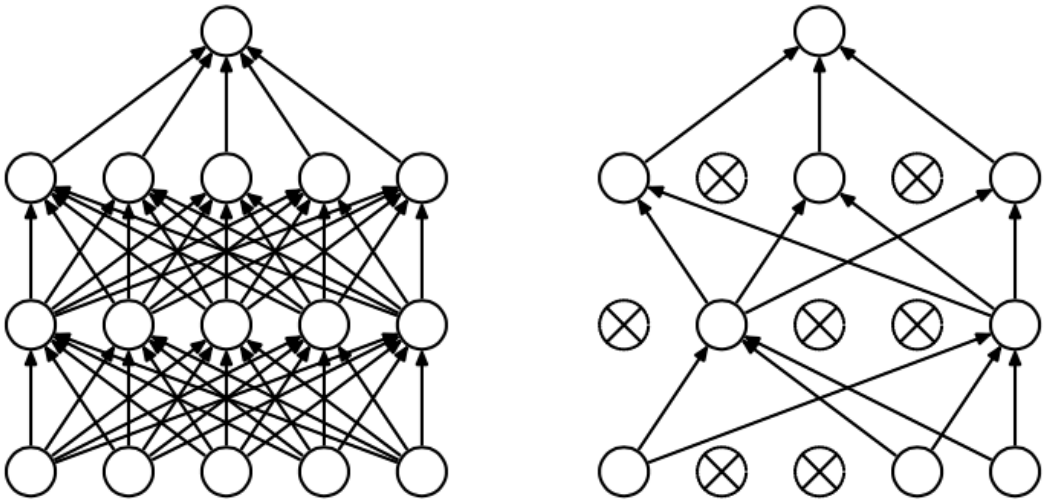


Figure 10

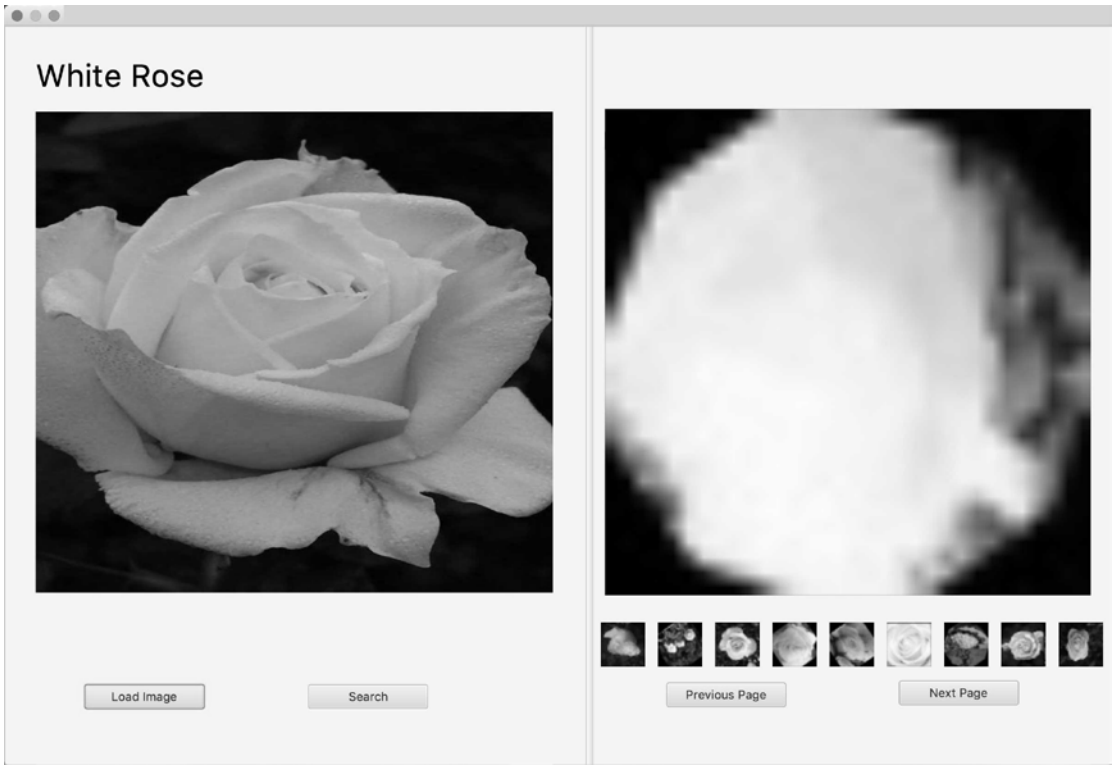



Figure 11

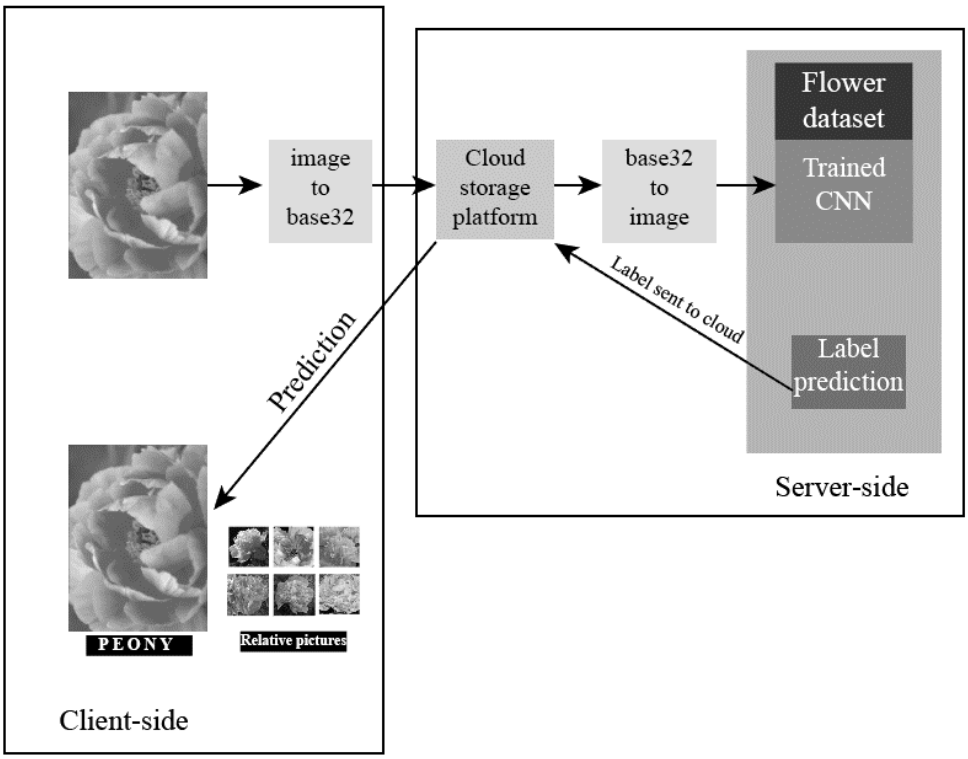



Figure 12

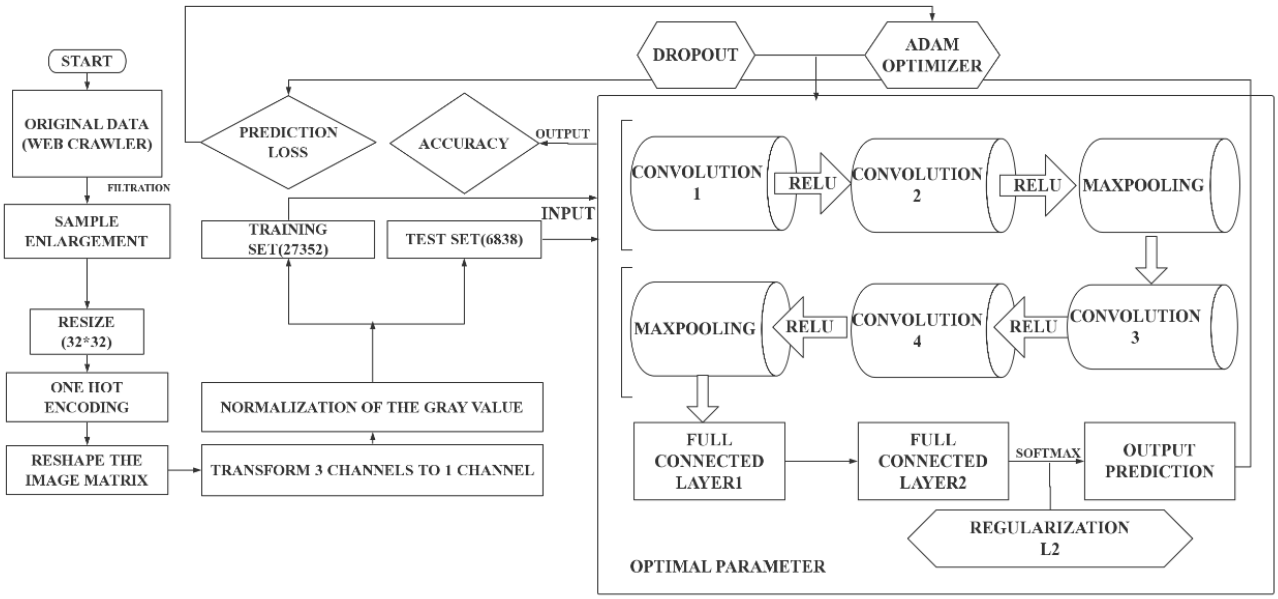



Figure 13

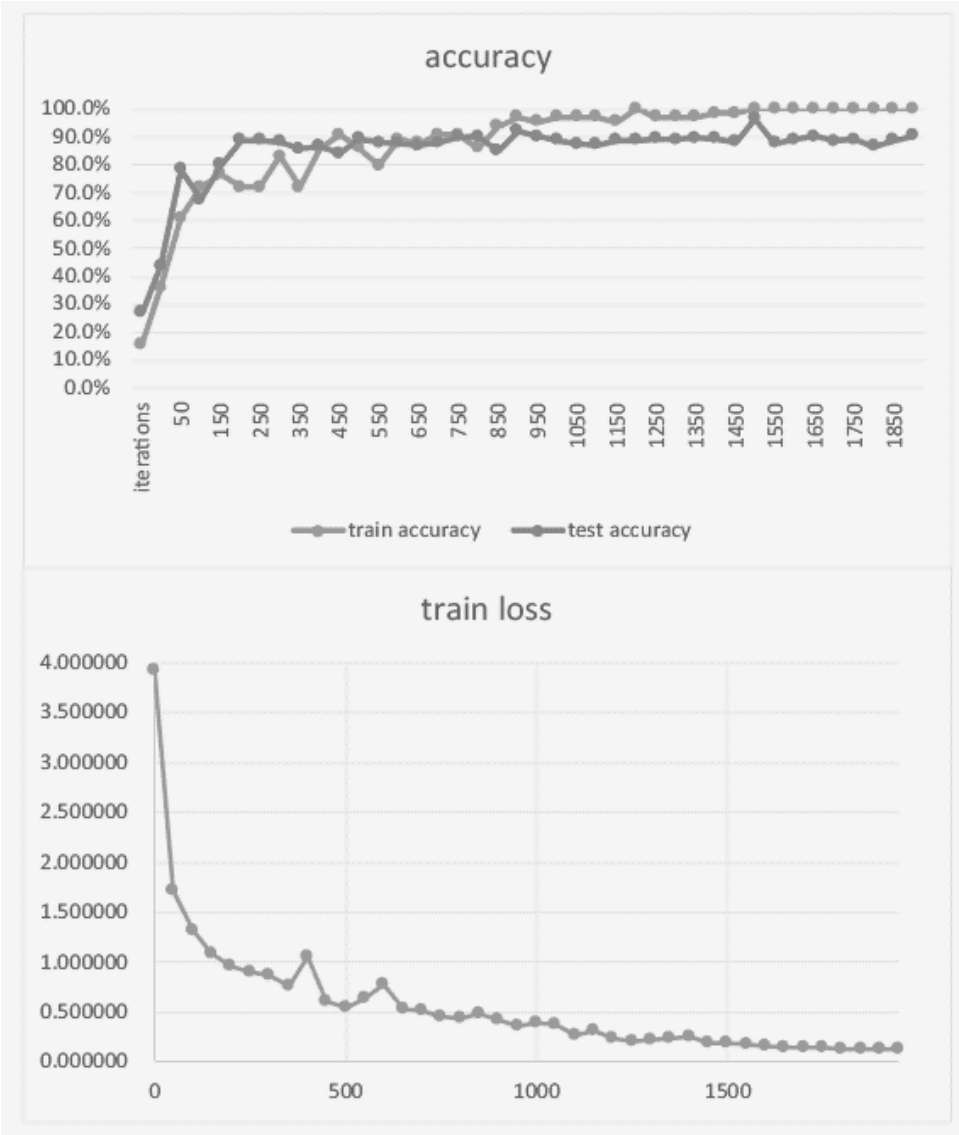

accuracy
100.0%
90.0%
80.0%
70.0%
60.0%
50.0%
40.0%
30.0%
20.0%
10.0%
0.0%
iterations
50
150
250
350
450
550
650
750
850
950
1050
1150
1250
1350
1450
1550
1650
1750
1850
train accuracy
test accuracy
train loss
4.000000
3.500000
3.000000
2.500000
2.000000
1.500000
1.000000
0.500000
0.000000
0
500
1000
1500

Figure 14

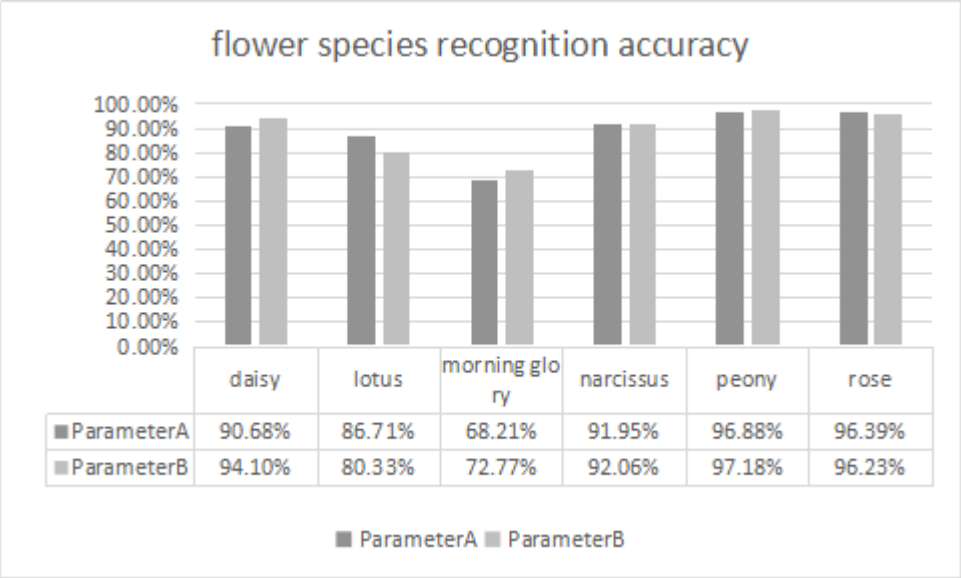



Figure 15

Recognition Result

| Train batch size | Dropout rate | Base learning rate | decay rate | Iteration steps | Test acceracy(%) |
|---|---|---|---|---|---|
| 64 | 0.96 | 0.0009 | 0.97 | 1700 | 94.90 |
| 64 | 0.96 | 0.0007 | 0.97 | 1700 | 95.00 |
| 64 | 0.96 | 0.0060 | 0.97 | 2000 | 93.32 |
| 64 | 0.96 | 0.0010 | 0.99 | 2500 | 94.53 |
| 64 | 0.96 | 0.0060 | 0.97 | 2000 | 93.32 |
| 64 | 0.98 | 0.0010 | 0.99 | 2000 | 95.53 |
| 256 | 0.99 | 0.0010 | 0.99 | 2500 | 70.55 |
| 64 | 0.43 | 0.0011 | 0.99 | 1700 | 94.96 |
| 64 | 0.43 | 0.0014 | 0.99 | 1700 | 95.66 |
| 64 | 0.43 | 0.0016 | 0.99 | 1700 | 93.37 |
| 64 | 0.42 | 0.0010 | 0.99 | 1400 | 95.53 |

Figure 16