School of Computing
FACULTY OF ENGINEERING

UNIVERSITY OF LEEDS

Images Classification and Difference measure by Deep Neural Networks

Yunpeng Chen

Submitted in accordance with the requirements for the degree of
MSc Advanced Computer Science

2017/2018

The candidate confirms that the following have been submitted*:*

<As an example>

Items	**Format**	**Recipient(s) and Date**
Deliverable 1	*Report*	*SSO (August/31/2018)*
Deliverable 2	*Software URL*	*Supervisor, assessor (August/31/2018)*
Deliverable 3	*Data files*	*Supervisor, assessor(August/31/2018)*

Type of Project: Exploratory Software

The candidate confirms that the work submitted is their own and the appropriate credit has been given where reference has been made to the work of others.

I understand that failure to attribute material which is obtained from another source may be considered as plagiarism.

(Signature of student)_____

Summary

The genetic code of certain plants has a profound effect on their growth. Researchers in plant sciences at the University are trying to discover the relationship between the genetic code of a plant to generate the effect of mutant flowers and normal flowers, and find the relation mutant flowers and the genetic code of plants. Since the Arabidopsis flowers are very small, if artificial identification is used, it will inevitably consume a lot of manpower and time. Moreover, the difference between flowers by artificial judgment requires the experience of botany and does not provide an objective judgment standard.

This project is designed to help the scientists through Machine Learning to distinguish between variegated flowers and normal flowers and automatically discover the characteristics of different variegated flowers. Furthermore, this project would find a way to correctly measure the difference between different flowers, which can provide objective criteria for scientifically judging the difference between variegated flowers and normal flowers.

Acknowledgement

First of all, I would like to express my deepest gratitude to my Director, Professor David Hogg, for his guidance and support of the project, as well as for the valuable meetings and discussions we have. Under his supervision, I have a good opportunity to improve my skills and study and his extensive experience has greatly helped me.

I am also very grateful to the Ph.D. student Richa Yeshvekar for providing me with valuable experimental data and opinions. I am also grateful to my project assessor Dr. Matteo Leonetti for his valuable feedback at certain stages of the project.

I would like to thank my parents, thank you for encouraging me to do my best.

List of Figures

List of Tables

Table of Content

Chapter 1 Introduction

1.1 Overview

Arabidopsis belongs to the Cruciferaceae family, which means that normal flowers of this plants have 4 sepals, 4 petals, 6 stamen and 1 carpel, with the petals arranged in a cross-like architecture. Flower development in Arabidopsis thaliana is specified by four gene activities, that is, the A, B, C and E function genes. A+E activity is needed for development of sepals, A+B+E activity for petals, B+C+E activity for stamens, and C+E activity for carpels (Figure 1.1).

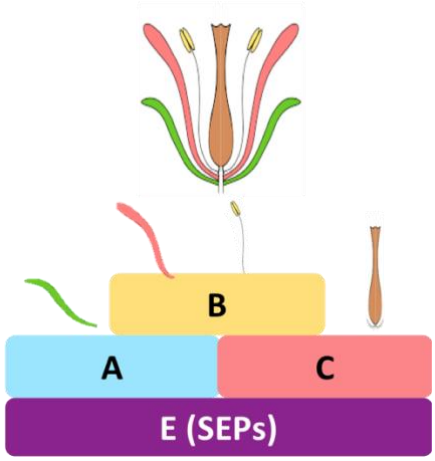

Figure 1.1 The ABC model of flower development

Data from the Davies Lab, University of Leeds, indicates that single and double mutants of SEP show significant phenotypic changes when grown under variable temperature conditions, as opposed to the ambient constant temperature conditions used in previous studies.

1.2 Aim and Objectives

The project is attempting to develop a high- throughput phenotyping system by employing artificial neural networks (ANNs). This process involves obtaining a high-resolution image of the flower (by using the Keyence microscope) and determining whether it is different from wild type flowers. Further, the project would develop a scoring system that would help in determining how different the flower is.

The solution is designed by re-training a pre-trained VGG16 model to extract the features of flower images, which be evaluated with some supervised learning algorithm. Next, based on images similarity principle, using Hamming distance and Euclidean distance to measure the

difference between two images, which be evaluated with clustering algorithm and Munkres algorithm. There are two objectives of this project:

1. The software should clearly classify normal flowers and eight types of mutant flowers.

2. For each sample within each mutant flower type, find the difference between it and normal flowers.

1.3 Motivation

It is evident from the figure 1.2 and 1.3 that although flowers of each SEP mutant look different from constant temperature when grown in fluctuating conditions, it is very difficult to objectively compose and describe the abnormalities. Apart from being laborious and time-consuming, techniques such as measuring the sizes of different floral organs do not give an obvious overview of the phenotype. The basic motivation of this project is to use the deep learning network to extract flower features, automatically classify a large number of normal flowers and eight variant types of flowers, which can effectively save manpower and speed up data processing.

There are differences between the eight variegated flowers and the normal flowers. If the difference is artificially analysed, there will be errors due to subjective intentions. The motivation of this project is to use the similarity algorithm to obtain the difference between flowers, and to provide objective criteria for judging the difference between variegated flowers and normal flowers.

Figure 1.2 Flowers in Variable Temperature **Figure 1.3** Flowers in Constant Temperature

1.4 Problem Statement

1. The size of the flowers for this project is very small, and there are background differences, light intensity difference and the complex flower shape. The project involves using the existing mutant flower data to obtain the characteristics of the mutant flowers.

2. This project aims to classify different types of flowers, it involves using the obtained flower characteristics to train the classifier to distinguish variegated flowers and normal flowers.

3. The second objectives of this project is to find difference between each type of mutant flower and normal flowers. The solution is to use the deep learning to get the features of flower images and using the extracted features to calculate the Euclidean distance and Hamming distance between each mutant flower and normal flowers, then judging the difference between two images.

1.5 Deliverables

1. Software for objectives 1 and 2.

2. Report including but not limited to 1) Background research 2) Project Design 3) Project experiment and results 4) Project Evaluation.

1.6 Project Planning

The initial plan (Fig. 1.4) was submitted to Minerva in April 2018.

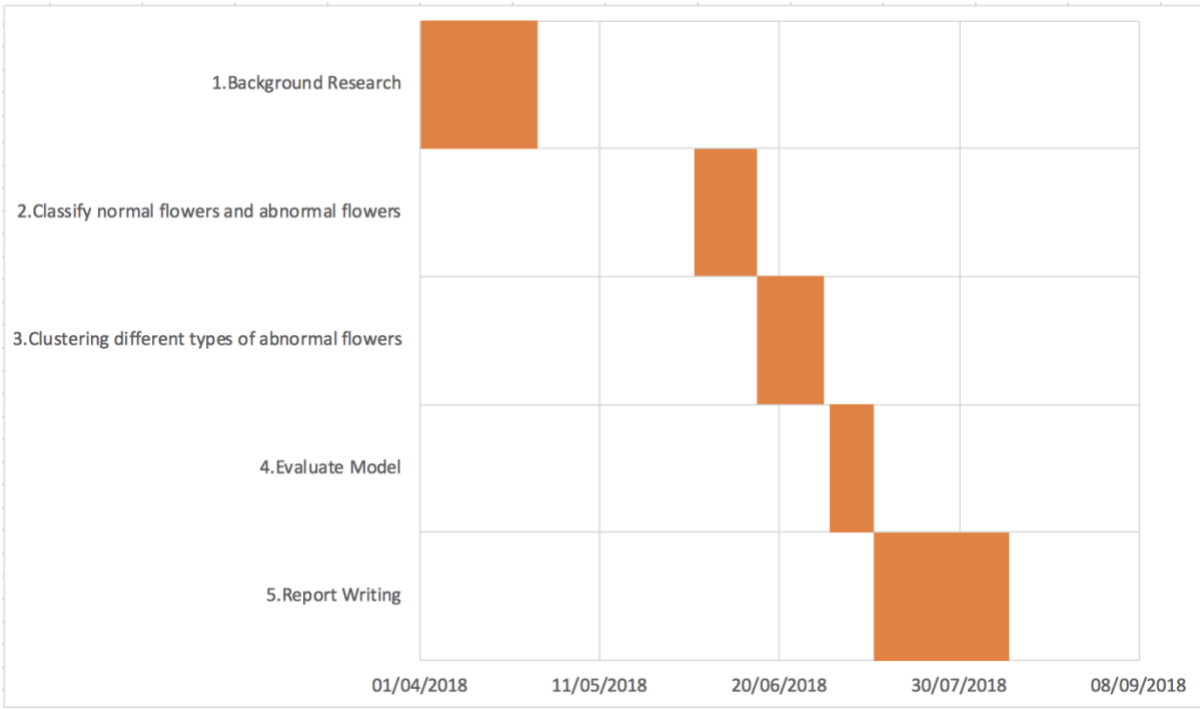

Figure 1.4 Gantt chart for initial project plan

The implementation of the initial plan was affected for the following reasons:

1. Examination review
2. Replaced the flower dataset
3. It took a long time to write code for finding the difference between images.

Figure 1.5 shows the real project plan and Table 1.1 shows the description of the real project plan:

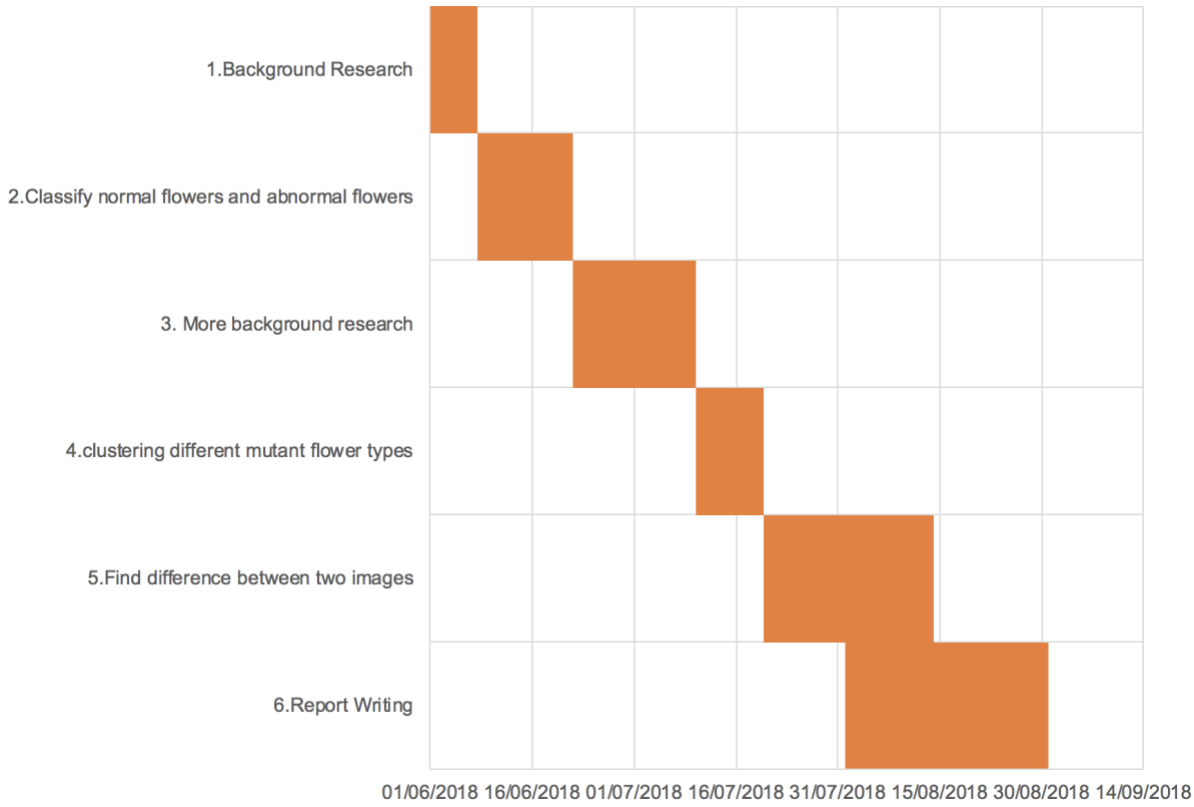

Figure 1.5 Gantt chart for real project plan

Table 1.1 Description for Project tasks

Task Number	Detailed Description	Duration Days
1	Read and analyse literature on artificial neural networks, convolutional neural networks, and image classification.	7
2	Using transfer learning to obtain the weight of the pre-trained VGG16 model, re-training the model with the flower data set, then classifying the normal flowers and the variegated flowers, and comparing with the ground truth to obtain the classification accuracy.	14
3	Read literature on unsupervised learning and images similarity analysis.	15
4	The VGG16 model is used to extract the features of the flower picture, then the flowers features are clustered by the clustering algorithm. Use the Monkres algorithm to obtain the optimal assignment for clustering results, which is the evaluation method for the performance of finding the	10

	difference between two images.	
5	Use the Euclidean distance and the Hamming distance to judge the difference between the flowers.	25
6	Report writing	30

1.7 Project Structure

The report is designed as follows:

Chapter 1 gives the background of the project and explains the problems to be solved by the project. It also explains the objectives and expected deliverables of the project. Finally, it shows the project time plan and overall structure of the project.

Chapter 2 provides background research, including machine learning, deep learning, and convolutional neural networks. This section mainly discusses related concepts, operating modes, model frameworks, and how to use them.

Chapter 3 refers to the project design, including code design and tasks design. This section discusses the technology needed to write the software. In addition, it explains in detail the design process and evaluation method of each task of the project.

Chapter 4 shows the experiments and results of the project. According to the project design, the project experiment is realized and the corresponding results are obtained, including classification results, image similarity results.

Chapter 5 presents the evaluation of the project. The project experiment and the corresponding results are evaluated and analysed, including classification accuracy, confusion matrix and clustering results.

Chapter 6 provides the conclusion and future work of this project. It discusses the general conclusion of the project, challenges, project limitations, personal recommendation.

Chapter 2 Background Research

This project aims to classify normal flower and eight types of mutation using machine learning and convolutional neural networks, and accurately determine the difference between different variegated flowers and normal flowers. Therefore, the project involves machine learning, artificial neural networks, and Images similarity algorithm.

2.1 Machine Learning

Tom M. Mitchell(1998, p2) gave a definition about Machine Learning: A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E. In simple terms, machine learning is a way to train a model by using data and then use model to predict.

The main reason why we need machine learning as followed:

1. the amount of data is huge, and people cannot get;
2. people's processing cannot meet the needs or need to define too many rules.

For example, one image, to determine whether this is a tree-containing picture, if you want to make the machine identification by definition, hundreds of definitions don't necessarily make the machine recognize correctly, so we need to use machine learning. Set an initial value and let the machine calculate the final value by calculation, the model can adapt to the data independently when it encounters new data, which can learn from previously generated reliable calculations, repeated decisions and results.

The two most widely adopted machine learning methods are supervised learning and unsupervised learning. Sometimes semi-supervised and reinforcement learning techniques are used.

2.1.1 Supervised Learning

For supervised learning, training samples with concept markers (classifications) are learned to classify and predict data outside the training sample set as much as possible. Because all the tags (classifications) are known, the training sample has low derogatory.

The more samples that are provided for supervised learning algorithm, the more accurate the ability to analyse the new data. And this is also the main challenge of supervised learning, that is, creating big data with tagged samples is time consuming and requires a lot of manpower.

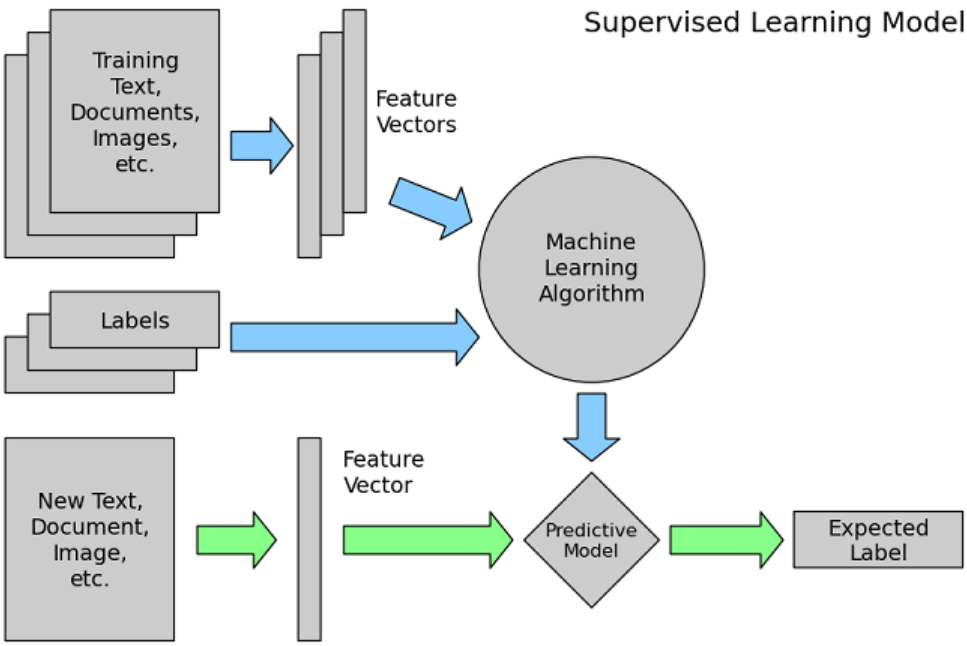

Figure 2.1 Supervised Learning Model(Kumar, 2018)

The process is as shown above. First, you need to prepare training data, which can be text, image, audio, etc., and then extract the required features to form feature vectors, and then send these feature vectors together with the corresponding markers/targets into the learning algorithm to train a Predictive Model. Next, applying the same feature extraction method to the new test data to obtain the feature vector for testing. Finally, using the prediction model to predict the feature vectors to be tested and obtain the results.

According to the different types of the target predictor, the supervised learning task is roughly divided into two categories: classification learning and regression prediction.

Table 2.1 Common Methods for Supervised Learning

Methods	**Classify Problem**	**Regression Problem**
Output type	Discrete data	Continuous data
Purpose	Looking for decision boundaries	Find the best fit
Evaluation method	Accuracy, Confusion Matrix	Sum of square errors

1. Classify Problem

Classified learning is the most common supervised learning problem. The most basic one is the binary problem, which is to judge the right and wrong and choose one of the two categories as the forecasting result. In addition, there are many types of classifications, for example, one of the multiple (≥3) categories is selected as the prediction result, and there is even a multi-label multi-classification problem, that is, whether a sample belongs to multiple different categories at the same time.

2. Regression Problem

The regression method is a supervised learning algorithm for predicting and modelling numerical continuous random variables, which is characterized by an annotated data set with numerical target variables, such as price, precipitation, and so on.

2.1.2 Unsupervised Learning

Unlike supervised learning, unsupervised learning algorithms focus on discovering the distribution characteristics of the data itself. When trying to identify patterns in the data, there is no need to use the expected results to mark the data set, that is, no meaningful mark is attached. In this way, while saving a lot of labour, the size of the data that can be used becomes unlimited.

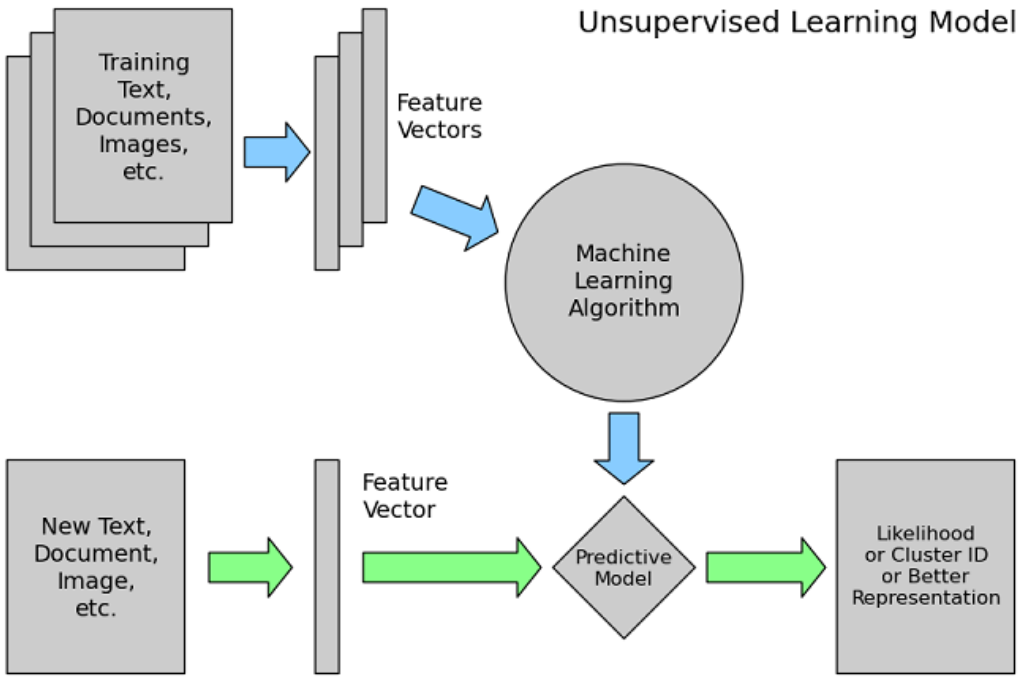

Figure 2.2 Unsupervised Learning Model

Some classic problems can be solved by unsupervised learning methods:

1. Clustering

Given a similarity criterion, find out which ones are more similar to the other. One area in which clustering is used is images search. For example, the returned search results contain many very similar images. Clustering can be used to group them, making it easier for users to identify images with large differences.

2. Association

Classify objects into different groups according to a relationship, such that the presence of object A in the group indicates that another object B also exists. For example, it is similar to the recommendation question that "people who bought A goods also bought B products." If you analyse a large number of shopping carts, you can see that the presence of product A in the shopping cart is likely to imply that product B also In the shopping cart, then you can immediately recommend product B to the person who puts product A into the shopping cart.

2.1.3 Semi-supervised Learning

This is the result of a mixture of supervised learning and unsupervised learning. This algorithm requires some labelled training data, but it is much less than supervised learning.

Usually there is a small amount of marked data and a large amount of unlabelled data, which is used when the required tagged data is too costly, for example, to identify a person's face on a webcam. This type of learning can be used in methods such as classification, regression and prediction.

2.2 Deep Learning

In recent years, due to the popularity of digital computers, human beings have entered the era of big data (Sivarajah, Kamal, Irani and Weerakkody, 2017). Every hour and every minute, the data on the Internet is massive and huge. The traditional machine learning algorithm has a general performance when the amount of data is large, and it is difficult to improve.

Therefore, deep learning has emerged. Deep learning is a new field in machine learning research. Its motivation is to build and simulate a neural network for human brain analysis and learning and mimics the mechanism of the human brain to interpret data such as images, sounds and texts. The deep learning model is very effective for the processing and analysis of big data due to the complexity of the network (Sohangir, Wang, Pomeranets and Khoshgoftaar, 2018).

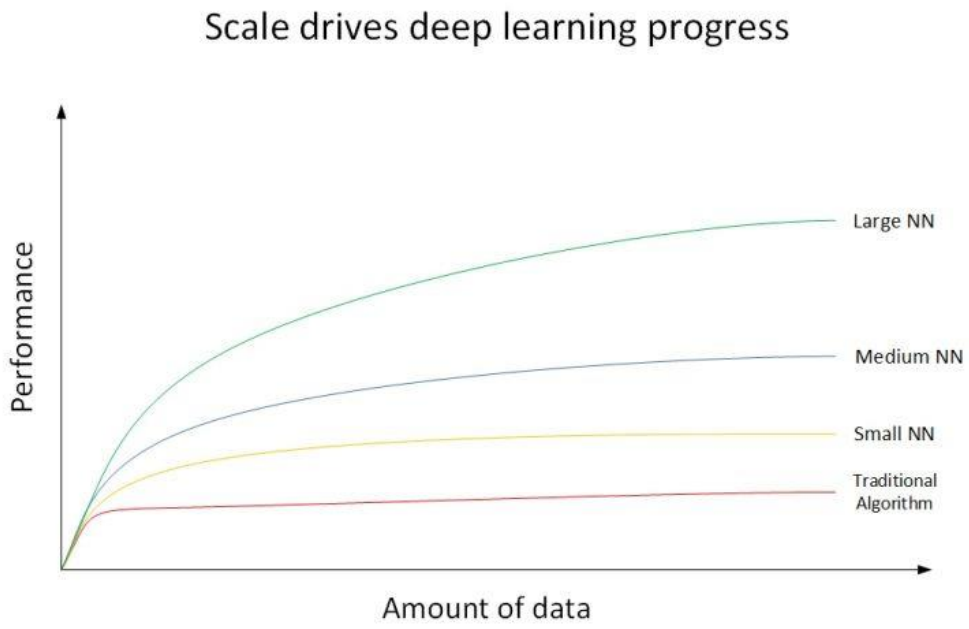

Figure 2.3 Performance for Deep Learning and Machine Learning

The above picture has 4 curves. Among them, the bottom red curve represents the performance of traditional machine learning algorithms, such as SVM, logistic regression, decision tree and so on. When the amount of data is relatively small, the performance of the traditional learning model is better. But when the amount of data is large, the trend of its performance is basically stable; The yellow curve above the red curve represents the smaller neural network model (Small NN). It outperforms traditional machine learning algorithms when the amount of data is large; The blue curve above the yellow curve represents a medium-sized neural network model (Media NN) that performs better than Small NN when the amount of data is larger; Finally, the top green curve represents a larger neural network (Large NN), the deep learning model, as can be seen from the figure, when the amount of data is large, its performance is still the best, and basically maintains a relatively fast rising trend.

2.3 Multi-layer Neural Network

Artificial neural network gives a general form of function, its calculation steps and methods are fixed, and its function is only determined by the value of the parameters (Chen and Wang, 1996). It can be regarded as a certain degree of simulation of the human brain, artificial nerve functions in the network are equivalent to defining the connection of a particular brain cell

(Joyce, Laurienti, Burdette and Hayasaka, 2010), and the parameters that are tunable in the function define the strength of these connections in this connection.

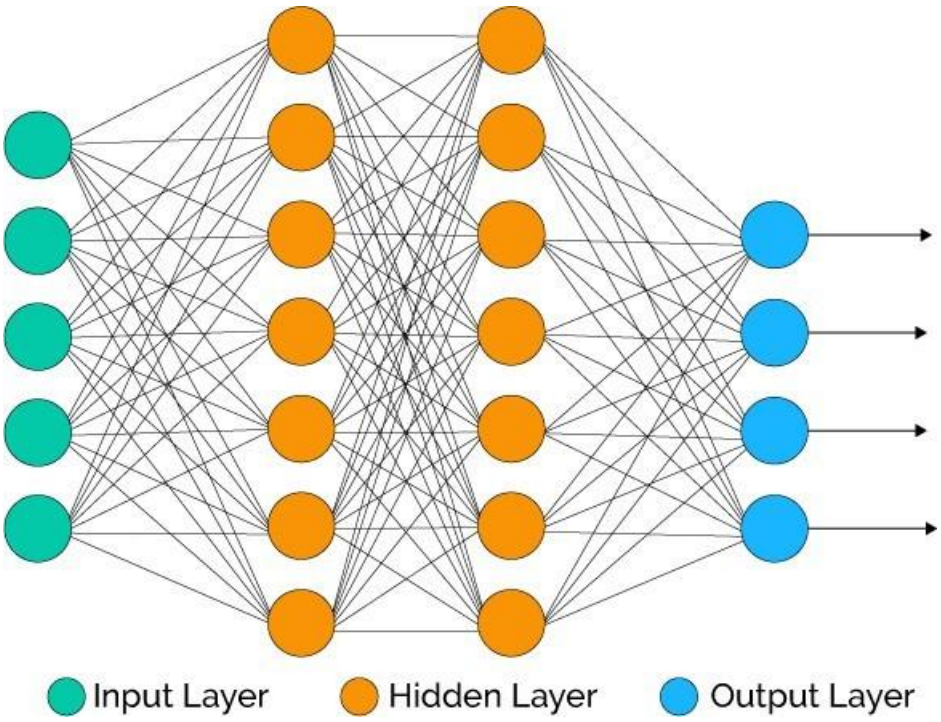

Figure 2.4 Multi-layer neural network

Each layer is composed of several neurons. The number of neurons in the input layer is determined by the dimension of the problem and the dimensions of the output layer are generally determined by the training objectives. For example, if the project want to do the K classification problem, then the number of neurons in the output layer is K. The number of layers in the middle hidden layer and the number of hidden layer neurons in each layer can be arbitrary, usually determined by the complexity of the problem. In general, the more complex the problem, the more layers of the hidden layer and the more neurons per layer, the better. However, the more layers, the more difficult it is to train neural networks.

However, multilayer neural networks have encountered bottlenecks in how to obtain the weight of the hidden layer, so the Back Propagation algorithm has emerged. The learning process of BP algorithm consists of two processes: forward propagation of signal and back propagation of error.

2.3.1 Forward Propagation

In the case of forward propagation, the input samples are passed in from the input layer, processed layer by layer through each hidden layer, and then transmitted to the output layer.

If the actual output of the output layer does not match the expected output, it goes to the backpropagation phase of the error.

The neurons in each layer are connected to the neurons in the next layer, which we call the weight of the network. In general, we use W_{ij} to represent the weights connecting the i-th neuron and the j-th neuron, so that the input of each layer of the network can be represented by the previous output and the weight W between the two layers. For example, if the output of the previous layer is represented by the vector x and the weight of the edge is represented by the matrix W, then the input of the current layer can be represented as x*W. After getting the input of the current layer, applying the activation function to it to get the output of the layer. The function of the activation function, for example, sigmoid function, is to make a nonlinear transformation of the input of each layer to get its output. If there is no nonlinear activation function, the whole network will degenerate into a linear mapping, and the learning ability of the neural network will be greatly reduced.

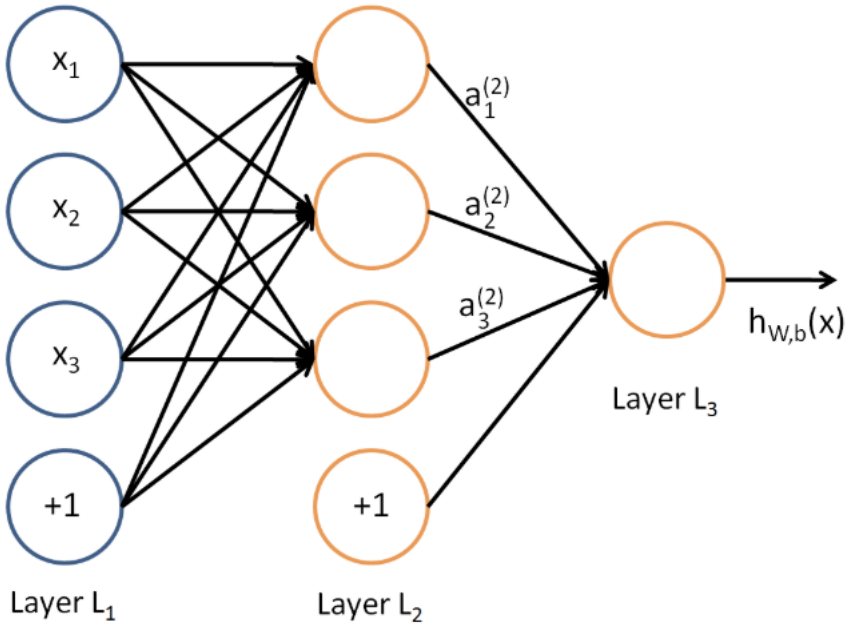

Figure 2.5 Forward Propagation Structure

$$
\begin{aligned}
a_1^{(2)} &= f(W_{11}^{(1)}x_1 + W_{12}^{(1)}x_2 + W_{13}^{(1)}x_3 + b_1^{(1)}) \\
a_2^{(2)} &= f(W_{21}^{(1)}x_1 + W_{22}^{(1)}x_2 + W_{23}^{(1)}x_3 + b_2^{(1)}) \\
a_3^{(2)} &= f(W_{31}^{(1)}x_1 + W_{32}^{(1)}x_2 + W_{33}^{(1)}x_3 + b_3^{(1)}) \\
h_{W,b}(x) &= a_1^{(3)} = f(W_{11}^{(2)}a_1^{(2)} + W_{12}^{(2)}a_2^{(2)} + W_{13}^{(2)}a_3^{(2)} + b_1^{(2)})
\end{aligned}
$$

Figure 2.6 Forward Propagation Formula

2.3.2 Back Propagation

In the case of back propagation, the output is back-transferred layer by layer to the input layer through the hidden layer in some form, and the error is distributed to all the cells of each layer, thereby obtaining the error signal of each layer unit, which is the basis for correcting the weight of each unit.

There is an error between the value of the output layer and the true value, we can use the mean square error to measure the error between the predicted value and the true value. The goal of back propagation is to make the value of the E function (see Figure 2.7) as small as possible, and the output value of each neuron is determined by the weight value of the point and the bias value corresponding to the layer. Therefore, we need to adjust the values of the weights (see Figure 2.8) and offsets to minimize the value of the error function.

$$\mathrm{E} = \frac{1}{2}\sum_{j=1}^{l}(y_j^k - a_j^l)^2$$

Figure 2.7 Mean Square Error Formula

$$\mathrm{w} = \mathrm{w} - \nabla_w$$
$$\mathrm{b} = \mathrm{b} - \nabla_b$$

Figure 2.8 Update Formula for Weights and Bias

2.4 Convolutional Neural Network

The effect of a multi-layer neural network is good, but it is a fully connected network and is not suitable for image recognition. There are three reasons:

1. Too many parameters

Consider a picture with 1000*1000 pixels input, and the input layer has 1000*1000=1 million nodes. Assuming that the first hidden layer has 100 nodes, then only this layer has (1000 * 1000 + 1) * 100 = 100 million parameters, which is really too much! The image only expands a bit, and the number of parameters is much larger, so its scalability is very poor.

2. Without usage of positional information between pixels

For image recognition tasks, the connection between each pixel and its surrounding pixels is relatively tight, and the connection to pixels far away may be small. If a neuron is connected to all neurons in the previous layer, it is equivalent to treating all the pixels of the image equally for one pixel, which is not in line with the previous assumptions. When we finish learning each connection weight, we would find that the value of a large number of weights is small, that is, these connections do not matter. Trying to learn a lot of weights that are not important, such learning will be very inefficient.

3. The limitation for the number of network layer

The more the number of network layers, the stronger the expression ability, but it is difficult to train the deep-connected neural network by the gradient descent method, because the gradient of the fully connected neural network is difficult to transmit more than 3 layers. Therefore, we are not able to get a deep fully connected neural network, which limits its ability.

Therefore, the convolutional neural network has been proposed (Lecun, 1998), which can solve the problems of the fully connected network from three aspects:

1. Local connections: Each neuron is no longer connected to all neurons in the upper layer, but only to a small number of neurons. This reduces many parameters.

2. Weight sharing: A set of connections can share the same weight, rather than having a different weight for each connection, which reduces many parameters.

3. Pooling: Reduce the number of samples per layer, further reducing the number of parameters, while also improving the robustness of the model.

2.4.1 Architecture

Convolutional neural networks have three layers: convolutional layer, pooling layer, and fully connected layer. Taking the Convolutional Neural Network of CIFAR-10 (Fig. 2.9) as an Example. The layers are described as follows:

1. Input layer: Input Images.

2. CONV layer: Calculate the local area of the image.

3. RELU function: It’s a activation function Max(0, x), which does not cause the gradient to disappear.

4. POOL layer: Sampling along the (width, height) of the image, reducing the dimension of the length and width.

5. FC layer: This layer is fully connected, and each unit is connected to each unit of the previous layer.

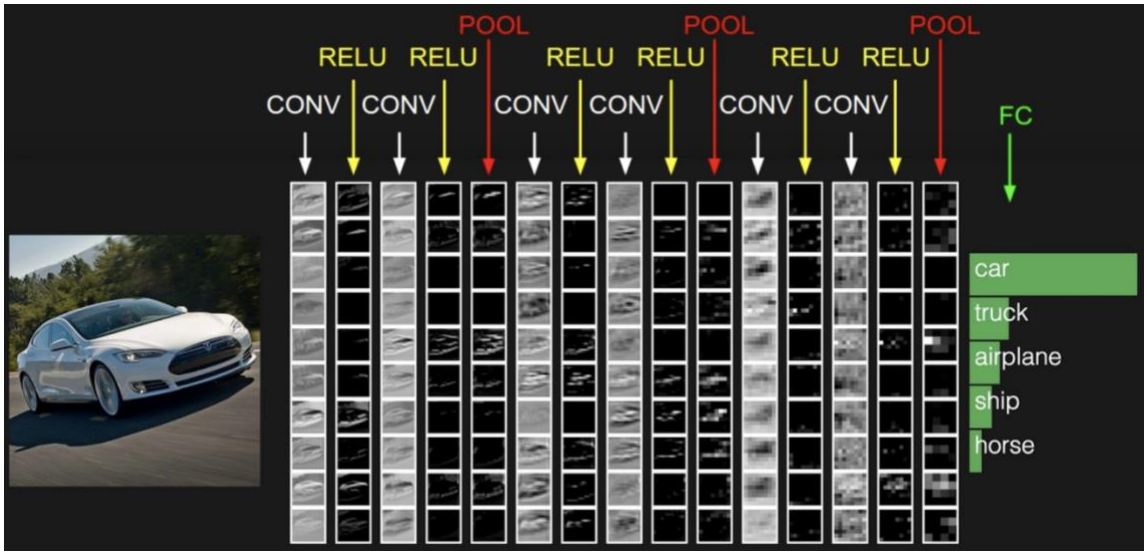

Figure 2.9 Example for Convolutional Neural Network (image credit Andrew Karpathy, Stanford class cs231n)

2.4.2 Basic layers Analysis

Convolutional neural networks have two basic layers:

1. Convolutional Layer

As mentioned above, the traditional Multi-layer neural network requires a large number of parameters because each neuron is connected to the neurons of the adjacent layer. The way of fully connecting the layers seems to be less friendly to the image data because the image itself has a "two-dimensional spatial feature", that is, a local feature. For example, if we look at an image of a cat, we may see that the cat's glasses or mouth know that it is a cat, and we don't need to say that after we look at every part of the image.

So if we can identify a typical feature of an image, then the category of the image is known. Therefore the concept of convolution was created. For example, now that there is a 4*4 image, we design two convolution kernels to see what the picture will look like after using the convolution kernel.

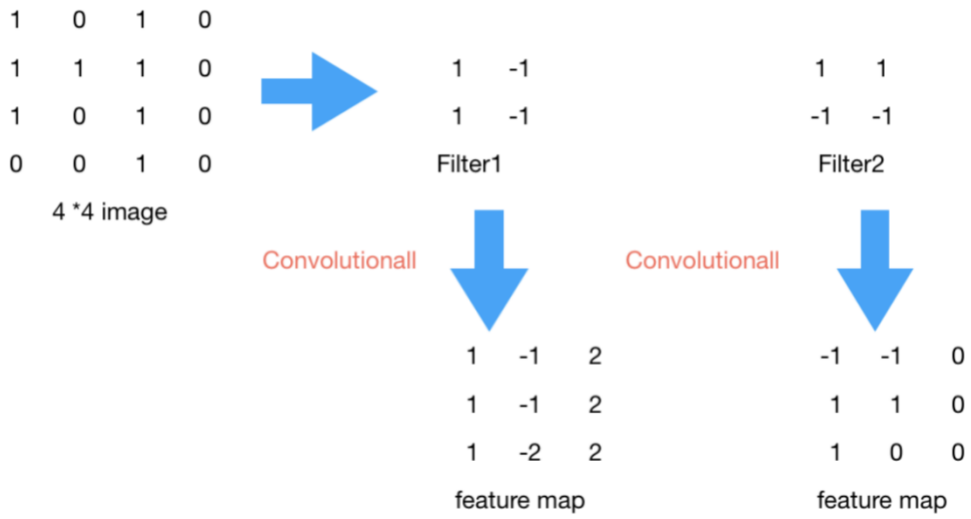

Figure 2.10 Process of Convolution

The convolution operation uses a convolution kernel to convolve with the corresponding region of the image to obtain a value, then by continuously moving the convolution kernel and finding the convolution, the convolution of the entire image can be completed.

In convolutional neural networks, the calculation of convolutional layers involves not only the general image convolution concept, but also the concept of depth and step size. Depth determines the number of neurons in the same region, that is, the number of convolution kernels that perform convolution operations on the same region; the step size is how many pixels the convolution kernel moves each time.

2. Pooling Layer

The convolution layer will follow a pooling layer. The direct purpose of the pooling layer is to reduce the amount of data to be processed in the next layer. For example, when the output size of the convolution layer is 32×32, if the size of the pooling layer filter is 2×2, then after the pooling layer processing, the size of the output data is 16×16. The pooling function has the following benefits for the network:

1). Greatly reduce the dimension of the feature and avoid overfitting.

2). Can remove some redundant information to get a low resolution feature map.

3). Make the model more concerned about whether there are certain features rather than the specific location of the features, and can tolerate some minor displacements of the features.

According to the calculated methods, pooling layer is divided into the mean pooling layer and the maximum pooling layer. Maximum pooling is even better in image tasks. Maximum pooling makes the network easier to capture changes in the image, gradient changes, and greater local information differences, better describing the semantic details of edges, textures, etc.

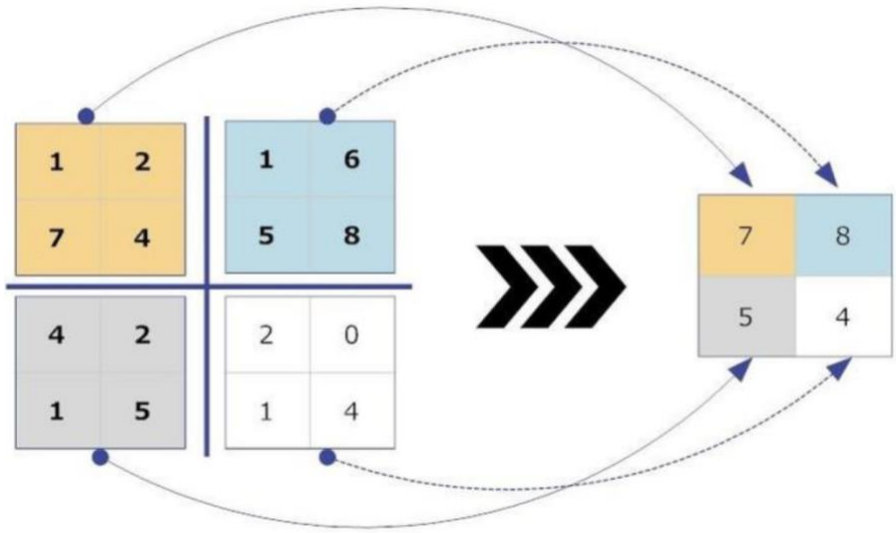

Figure 2.11 Maximum Pooling

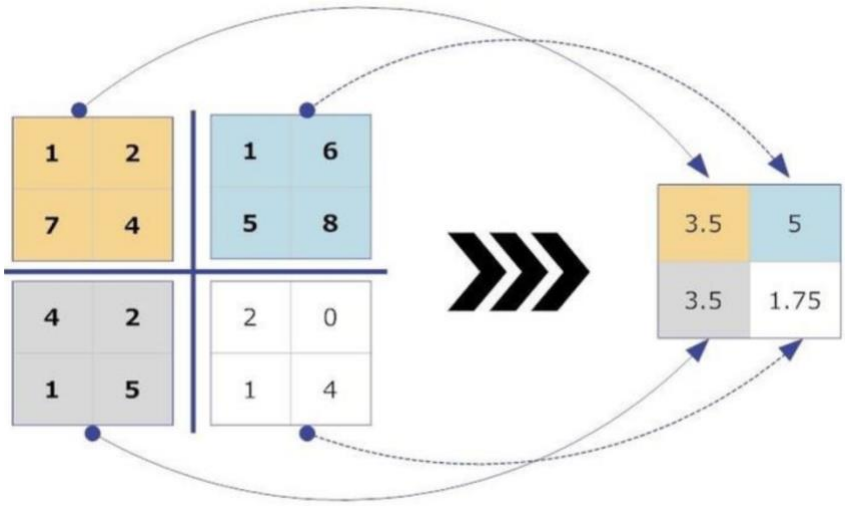

Figure 2.12 Average Pooling

2.5 Similarity between Images

In machine learning and data mining, it is often necessary to know the differences between individuals, and then to evaluate individual similarities and categories, such as clustering and computing similarity.

2.3.1 Euclidean Distance

The Euclidean Distance (Wikipedia) is a commonly used distance definition that refers to the true distance between two points in an m-dimensional space, or the natural length of a vector. The Euclidean distance in 2D and 3D space is the actual distance between two points.

$$d = \sqrt{\sum_{i=1}^{N}(x_{1i} - x_{2i})^2}$$

Figure 2.13 Formula for Euclidean Distance

In the process of calculating the similarity, the Euclidean distance is relatively intuitive, common similarity algorithm. In terms of its meaning, the smaller the Euclidean distance, the greater the similarity between the two users. The larger the Euclidean distance, the smaller the similarity between the two users.

2.3.2 Hamming Distance

In 1950, Richard W. Hamming had invented a code for correcting errors in communication and the Hamming code was born (Gedda, 2010). The Hamming distance between two equal-length strings is the number of different characters corresponding to the positions of the two strings.

In order to calculate the Hamming Distance between two strings, it need to be able to replace one string with another, so the two strings to be compared must be of equal length, otherwise the distance is not true.

Chapter 3 Design and Experiment

3.1 Coding Design

The software product of this project should implement two goals:

1. The users can accurately detect whether the input flower image belongs to normal flower or one of eight mutant flower types through software.

2. For four mutant flowers at constant temperature and natural temperature, the software needs to be able to accurately calculate the difference between each mutant flower and normal flowers.

Based on these requirements, things related to software development are listed below.

3.1.1 Programming Language

All my project experience is based on the Python language, so after analysing the Python language as follows, Python is chosen as the programming development language.

1. Python is cross-platform and open source.

Python can run across platforms and has been open source for more than 20 years. If you need code to run on Linux, Windows, and macOS at the same time, Python will suffice.

2. Python has one of the most mature package repositories(Dalcín, Paz and Storti, 2005).

Based on PyPI, Python is a repository of more than 85,000 Python modules and scripts that can be used to solve advanced data analysis such as database processing, computer vision implementation, and dimensional analysis.

3. Python is widely used in data science.

Data analysis skills are as important as coding skills(Team, 2016). For this project, the early processing and analysis of the results is important.

3.1.2 Pre-trained Model

The pre-trained model is a model created by predecessors to solve similar problems. When you solve the problem, you don't have to train a new model from scratch. You can start with a model that has been trained in similar problems. For example, if you want to make an image recognition algorithm, you can spend years building a new algorithm from scratch, or you can get the inception model (a pre-training model) that Google trained on the ImageNet

dataset. A pre-training model may not be 100% accurate for your application, but it can save you a lot of time and effort.

Imagenet dataset is a field that is widely used in the field of deep learning images. Research work on image classification, positioning, and detection is mostly based on this dataset. The Imagenet dataset has detailed documentation, is specially maintained by the team, and is very convenient to use. It is widely used in research papers in the field of computer vision, and has become a "standard" dataset for the performance testing of deep learning images in the field. ImageNet (Stanford Vision Lab, 2018) has more than 14 million images covering more than 20,000 categories; more than one million of them have clear category annotations and labeling of objects in the image, as follows:

1. Total number of non-empty synsets: 21841
2. Total number of images: 14,197,122
3. Number of images with bounding box annotations: 1,034,908
4. Number of synsets with SIFT features: 1000
5. Number of images with SIFT features: 1.2 million

We can use the models already trained on very large amounts of data such as ImageNet for difficult tasks with thousands of classes. Fortunately, there are many such pre-trained structures in the Keras library.

Keras library is a high-level neural network API, which is written in pure Python and based on TensorFlow, Theano and CNTK backends (Keras Documentation, 2018). The following requirements Keras can suffice:

1. To support rapid experimentation and can quickly convert your idea into results.
2. Easy and fast prototyping (keras are highly modular, minimalist, and expandable).
3. Support CNN and RNN, or a combination of the two.
4. Seamless switching between CPU and GPU.

Also, the state-of-the-art pre-trained networks included in the Keras core library represent some of the highest performing Convolutional Neural Networks, for example, VGGnet, on the ImageNet challenge over the past few years. We can use theses pre-trained model from Keras by import Keras library to the program.

3.1.2 VGG16

VGG network was proposed by Oxford's Visual Geometry Group, which is a related work at ILSVRC 2014(Department of Engineering Science, 2014). The main work is to prove that

increasing the depth of the neural network can affect the final performance of the network to some extent, exploring the relationship between the depth of the convolutional neural network and its performance by repeatedly stacking 3*3 small convolution kernels and 2*2 maximum pooling layers.

VGG has two structures: VGG16 and VGG19. There is no essential difference between the two, but the network depth is different.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure 3.1 Structure diagram for VGG network

As can be seen from the above figure 3.1, VGG has a total of five-segment convolution, and each segment is followed by a maximum pooling layer. The input of the network is an image of 224*224 size, and the output is an image classification result. The RELU activation function is not shown in the above structure in consideration of the reduced structure display of the entire network. Some of the above structures are explained:

1. Conv: The convolution layer.

2. Conv3: The convolutional layer uses 3x3 filters. Multiple small convolution stacks are better than single large convolution in classification accuracy. More activation functions, richer features, and greater discriminating power. The convolution is accompanied by an activation function and the use of more convolution kernels makes the decision function more discriminating. Furthermore, the parameters of the convolution layer are reduced. 3*3 significantly reduces the amount of parameters compared to the large convolution kernels of 5*5, 7*7 and 11*11.

3. Conv3-64: There is 64 of depth.

4. Maxpool layer: Maximum pooling. VGG network use kernel with size 2*2 in pooling layer. The smaller pool kernel brings more detailed information capture, which allows the maximum pooling to capture more detailed information.

5. FC: The fully connected layer.

VGG network deepens the number of network layers and in order to avoid too many parameters, 3x3 small convolution kernels are used in all layers, and the convolution layer step size is set to 1. The VGG fully connected layer has 3 layers, which can be from VGG11 to VGG19 according to the total number of convolutional layers plus fully connected layers. The least VGG11 has 8 convolutional layers and 3 fully connected layers, and the most VGG19 has 16 convolutional layers and 3 fully connected layers. Compared with Traditional Convolutional Neural Network, VGGNet has simplified structure of convolutional neural networks, but it need to train large number of features.

Table 3.1 The number of parameters for VGG network

The total number of parameters (Unit: Million)					
Network	A, A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

3.1.3 Clustering Algorithm

The process of clustering shows in the figure 3.2 below:

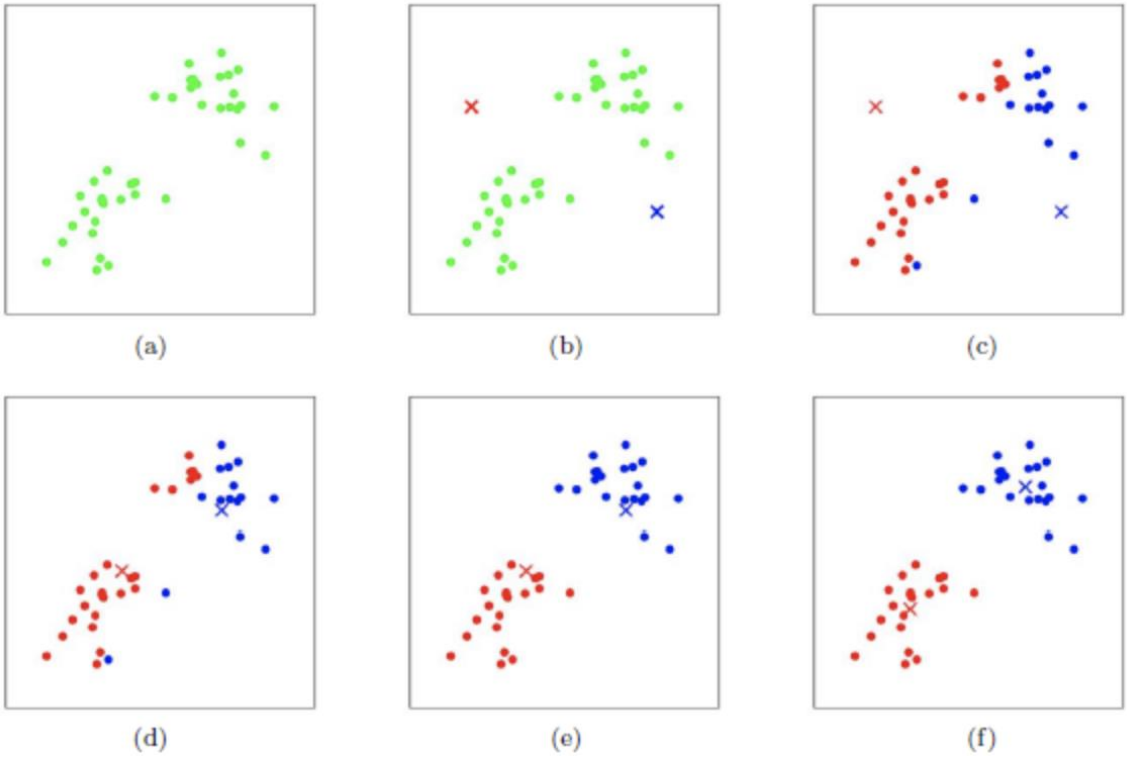

Figure 3.2 Process of Clustering

Figure (a) shows the initial data set, assuming k=2. In Figure (b), we randomly select the centroids of the categories corresponding to the two classes, namely the red centroid and the blue centroid in the graph, and then separately find the distances of all the points in the sample to the two centroids, and mark the category of each sample as the category of the centroid with the smallest distance from the sample, as shown in Figure (c).

After calculating the distance between the sample and the red centroid and the blue centroid, we obtained the category after the first iteration of all sample points, then mark as red and blue. As shown in Figure (d), the positions of the new red and blue centroids have changed. Figures (e) and (f) repeat the process in Figure (c) and Figure (d), marking the categories of all points as the nearest centroid category and seeking a new centroid. In the end we get two categories as shown in Figure (f).

3.1.4 Kuhn–Munkres Algorithm

Load data to clustering algorithm to get the predicted label, but this prediction is not accurate since we don’t know the assignment between predicted label and real category of image.

So we need The Hungarian algorithm, which is a combinatorial optimization algorithm for solving task assignment problems in polynomial time (O(n3)). It is called the Hungarian algorithm because a large part of the algorithm is based on the work of previous Hungarian mathematicians. The algorithm is then called the Kuhn–Munkres algorithm.

For example, suppose you are a provider of network services, and you can install Wi-Fi for free. Now, there are three users who want to install Wi-Fi. At the same time, you have three Wi-Fi workers nearby. Every worker who goes to the user's home to install Wi-Fi needs a certain cost, such as fuel costs. The cost is expressed as a matrix (Fig. 3.3). As a businessman, you will definitely consider how to distribute can save the most money. You will choose the most cost-effective one as the distribution plan by comparing the expenses of different allocation schemes. For each allocation scheme, we can calculate its total cost. In this question, there are a total of 6 allocation schemes. But this one-to-one comparison is too much trouble.

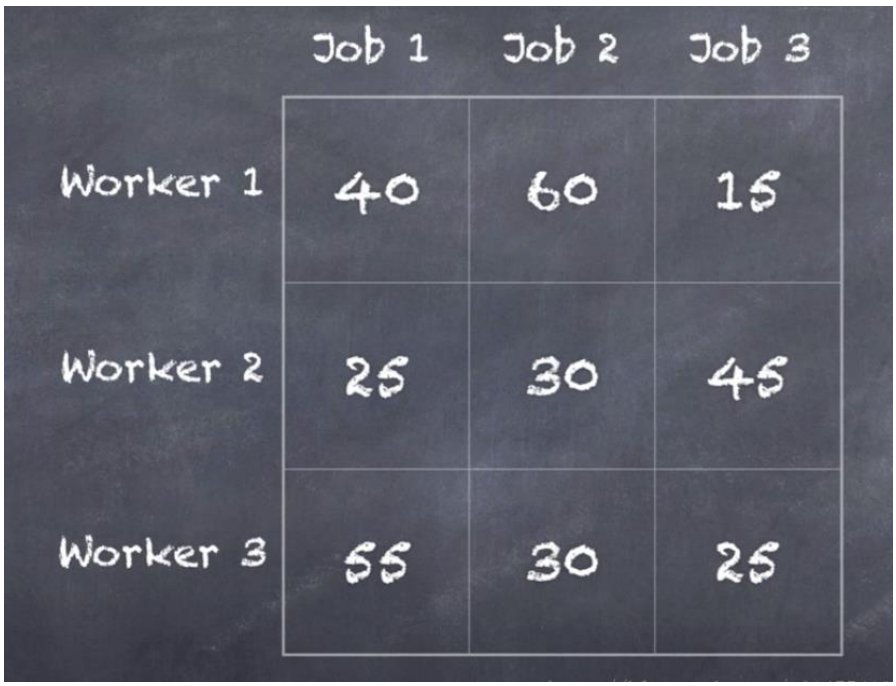

Figure 3.3 Cost Matrix

Then, the Kuhn–Munkres algorithm can solve this types of question effectively.

1) Matrix protocol

Traverse the rows of the matrix, find the minimum of each row, and subtract the minimum value for all elements on each row; traverse the columns of the matrix, find the minimum of each column, and subtract the minimum from all elements on each column.

2) Overwrite all 0s in the matrix with as few horizontal or vertical lines as possible(see Fig 3.4)

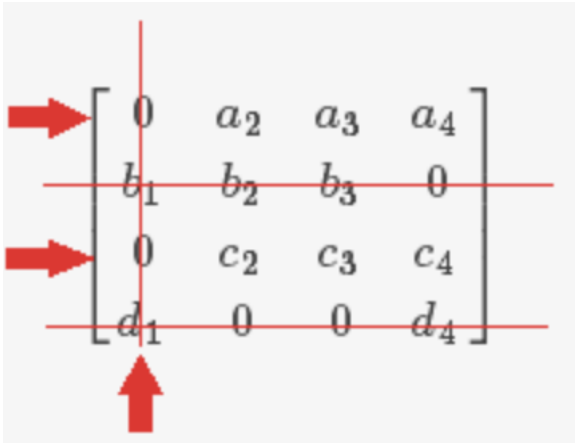

Figure 3.4 Matrix Process

Assign as many tasks as possible to workers. For example, the first line (representing the first manually accepted task) has a 0, so the first task is assigned to worker 1. Since the first task has been assigned, the 0 in the first column of the third row is no longer considered.

3) Adjust the matrix

Find the minimum value from the uncovered elements in the previous step, then subtract the minimum value from these elements, and add this minimum to the elements of the line intersection. The minimum value in the covered element is actually the inevitable overhead in completing all tasks. The role of this step is to increase the number of zeros in the overhead matrix, making the task easier to allocate.

4) Repeat steps 2 and 3 until all tasks are assigned.

3.2 Classification

The flowers this project choose are affected by four different genes at constant temperature and natural temperature, producing eight variants. The first objective of the project is to use the VGG16 neural network to make nine different forms of flowers, including one type of normal flower and eight types of mutant flowers, to make accurate classification.

3.2.1 Data Process

The raw data set includes one normal flower images set and eight variant flower images sets. However, the sample size of the image data sets is too small, the sample size of each mutant flower sets is less than 30.

Although a two-layer network can theoretically fit all the distributions, it is not easy to learn. Therefore, in practice, usually increase the depth and breadth of the neural network, so that

the learning ability of the neural network is enhanced, and it is easy to fit the distribution of training data. However, as the neural network deepens, the parameters that need to be learned increase, which makes it easier to over-fitting (when the data set is small, too many parameters will fit all the characteristics of the data set. Rather than the commonality between data).

Therefore, in order to prevent over-fitting, data augmentation (Tab. 3.2 and Fig. 3.5) has emerged, which is mainly used to prevent overfitting while the dataset is small. The project uses the following data augmentation methods:

Table 3.2 Data augmentation Methods

Rotate Image	Rotate the image at a fixed angle in a clockwise or counterclockwise direction
Flip Image	Randomly flip an image at a certain angle along a horizontal or vertical method
Shift Image	Pan the image a certain step along the horizontal or vertical method
Zoom Image	Zoom in or out of the image

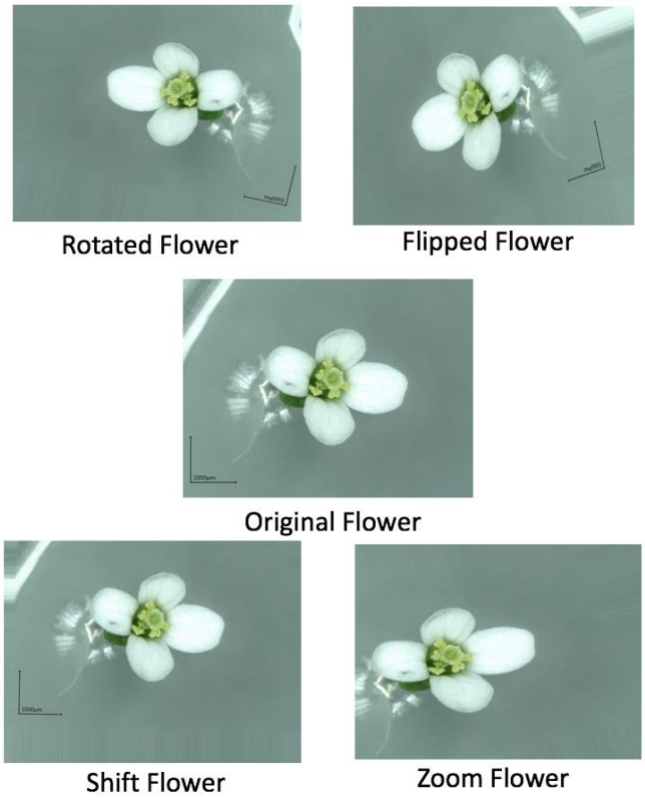

Figure 3.5 Data Augmentation

The amount of data increases dramatically by implementing data augmentation with the combination of methods above. However, Deep Networks have a large number of unknown parameters, the task of training a network is to find the optimum parameters using the training data. In order to find all the unknown parameters accurately, even there are lots of data, we still need large amount of time to train the network. In this case, the pre-trained model and transfer learning can provide great help.

3.2.2 Transfer Learning

A teacher usually has many years of experience in the field she/he teaches. On the basis of these accumulations, teachers can teach students the most concise content in the field and this process can be seen as a "information transfer" between the veteran and the novice.

This process is also applicable in neural networks. We know that neural networks need to be trained with data, which gets information from the data and converts them into corresponding weights. These weights can be extracted and migrated to other pre-trained neural networks, which is a model created by predecessors to solve similar problems. Then, we “migrate” these learned features and do not need to train a neural network from scratch.

This project uses a pre-trained VGG16 model on the ImageNet dataset, which can be found in the Keras library. The model framework can be viewed in Figure 3.1. There are two learning policies for using pre-trained model(Karpathy, 2014):

Table 3.3 Fine-Tuning Model

Feature Extraction	The pre-training model is used as a feature extraction device. The output layer is removed and the remaining network is then applied to a new data set as a fixed feature extractor.
Train specific layers and freeze other layers	To train the pre-training model partially. The specific approach is to keep the weights of some layers at the beginning of the model unchanged, retrain the latter layers, and get new weights.

How to use and train the model is determined by the similarity of the dataset size and the data between the pre-trained dataset and the dataset we are trying to solve (Karpathy, 2014). This project uses a small amount of dataset and is less similar to the ImageNet data participating in the pre-training, according to Table 3.4 and Figure 3.6, the first 13 layers of the VGG16 network will be frozen and then the next network will be retrained. This is done because the first few layers of the network capture the general characteristics of curves and edges, which is relevant to our problem. We want to ensure that these weights are the same,

so that the network will focus on the characteristics unique to this data set during the learning process, so as to adjust the network behind it.

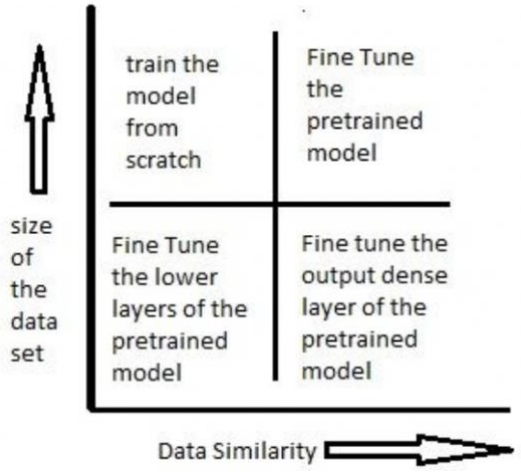

Figure 3.6 Choose pre-trained models in various situations

Table 3.4 Description for choosing pre-trained models in various situations

Small data set, high data similarity	The data is highly similar to the training data of the pre-trained model, we do not need to retrain the model. We only need to use pre-trained model as feature extractor and change the output layer into a structure that fits the problem situation.
Small data set, low data similarity	In this case, the weights in the first k layers in the pre-training model are frozen, and then the next n-k layers are retrained. Of course, the last layer also needs to be modified according to the corresponding output format.
Large data set, low data similarity	In this case, because we have a large data set, the neural network training process will be more efficient. However, because there is a big difference between the actual data and the training data of the pre-trained model, using the pre-training model will not be an efficient way.
Large data set, high data similarity	This is the ideal situation, and using a pre-trained model can be very efficient. The best way to use it is to keep the original structure and initial weight of the model unchanged, and then retrain on the basis of the new data set.

Furthermore, since the amount of training data is not enough, in order to prevent the over-fitting, a Dropout layer is connected to each fully connected layer. The probability of the

Dropout layer setting is **p**, indicating that each neuron is connected to the posterior neuron with a probability of **1-p** and the architecture randomly closes the activation function with probability **p**.

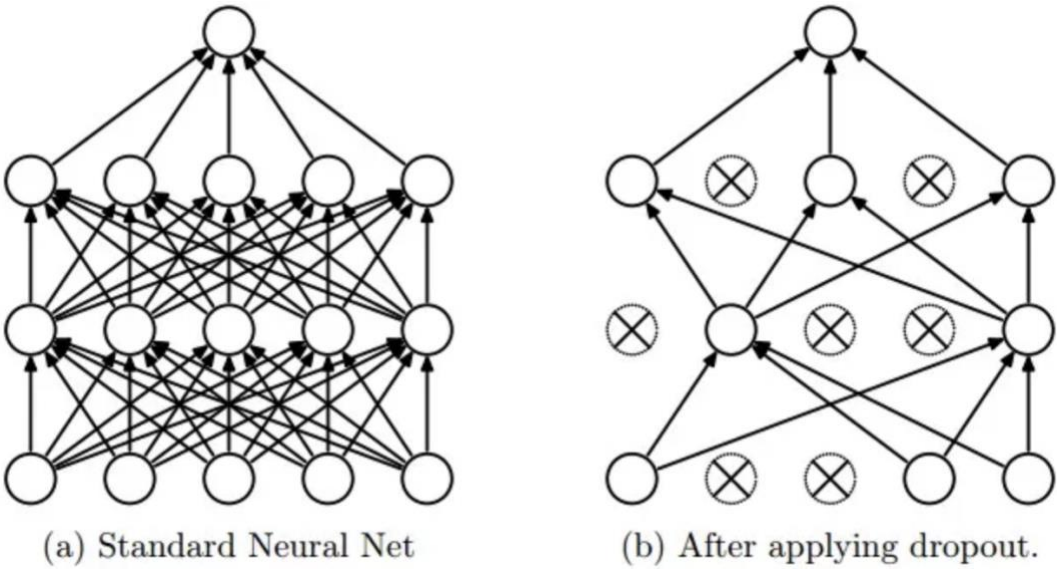

Figure 3.7 Layout of Dropout layer

The output layer of VGG16 model is a softmax layer with 1000 categories, it needs to remove this layer and replaced it with a softmax layer with only 9 categories because this project want to classify 9 different categories.

The softmax regression model is a generalization of the logistic regression model on the multi-classification problem. In the multi-classification problem, the number of categories to be classified is greater than 2, and the categories are mutually exclusive.

$$f(z_j) = \frac{e^{z_j}}{\sum_{i=1}^{n} e^{z_i}}$$

Figure 3.8 Formula of Softmax

From the figure 3.8 above, if one $\mathbf{z_j}$ is larger than the other **z**, then its value is closer to 1, the other is closer to 0, and all input data is normalized.

After the transfer learning is completed, the retrained model is tested by new data set. After each flower image is input into the VGG16 model, the output is an array of 9 values between 0 and 1, where the subscript of the each value is one of nine categories this project wants to classify and the subscript of the maximum value is the category to which the image belongs.

3.2.3 Evaluation Method

In the related research of pattern recognition and machine learning, the data set is often divided into a training set and a test set. The former is used to build a model, and the latter is used to evaluate the generalization ability of the model, that is, the accuracy when predicting unknown samples.

However, this simple method has two drawbacks:

1. The final model and parameter selection will depend greatly on your method of partitioning the training set and test set. If the training set and test set are not well defined, it is very likely that it will not be able to choose the best model and parameter.

2. This method only uses part of the data to train the model. When the amount of data used for model training is larger, the trained model usually works better. So the division of the training set and the test set means that we can't make full use of the data we have at hand, so the resulting model effect will also be affected.

There is another method named Simple Cross-Validation. First, the data set is divided into training set and validation set, and the training set is used to train the model to obtain the corresponding hypothesis function H_i; next, each hypothesis function is verified by the test set to obtain Hi which minimizes the training error. Normally, 25%-30% of the samples is used as a cross-validation set, and the rest is used as a training set.

However, this method wastes data from the validation set, and even if we bring the model back into the entire sample set, we still only use 70% of the sample to model. If the size of data set is very large, there is no problem with the simple cross-validation method; but if the sample is very scarce and the acquisition is difficult, then we need to consider a method that can make full use of the sample.

Hence, Stone (1974) proposed the K-fold Cross-Validation method. The k-fold cross-validation randomly divides the sample set into k parts, k-1 parts as the training set and one part as the validation set, rotating the training set and the verification set in turn, and the model with the smallest verification error is the optimal model. For example, if K=10, then the steps we take with ten-fold cross-validation are:

1. Divide all data sets into 10 parts

2. Take one of the 10 parts each time, use the other nine as the training set and validation set to train model, and then calculate the MSE_i (Mean Squared Error)of the model on the test set.

3. Average 10 times of MSE_i to get the final MSE(see Fig. 3.9).

$$\mathrm{CV}_{(k)} = \frac{1}{k}\sum_{i=1}^{k}\mathrm{MSE}_i.$$

Figure 3.9 Average MSE formula(Fushiki, 2011)

The K-fold cross-validation method, which is reserved for validation, is 1/k of the total sample size, so the amount of sample used for training is increased correspondingly, but K-fold cross-validation is needs to run k times, and its calculation cost is still high.

The project divides the normal flowers set and eight variegated flowers set in the data set into 10 parts, each time use one as test set, and the rest as the training set to train the model. There are 13 samples in the normal flower set and there are 3 samples in eight mutant flowers set in each test set. Test the trained model using the above data set, we can get the predicted categories of these images. Because it is supervised learning, the programme already know the real categories of these test images, comparing them with the predicted categories, then get the prediction accuracy.

The project uses 10-fold cross-validation to perform 10 tests on the processed data set, obtains 10 test prediction accuracy, and then goes to the average to get the final prediction accuracy.

3.3 Difference Measure for Images

The flower this project choose has eight mutant types and the samples in same mutant type are different. The second objective of the project is to find the difference between each mutant flower of each mutant type and normal flowers.

There are nine groups, including one normal flower group and eight other mutant flower groups, denoted as **N** and $\mathbf{M_i}$ with **i** = {1, 2, 3, 4, 5, 6, 7, 8}; and there exist more than one sample in each $\mathbf{M_i}$.

This project tested Four designs and the corresponding experiments for this task:

3.3.1 Raw Images

The flower image is processed when it is entered into the system, which has three colour channels and its pixel size is 224*224. Read the image file in order according to the directory, and convert it into an array format. Then using the array, calculate the Euclidean Distance between each sample of $\mathbf{M_i}$ and all samples in **N**.

For each variation flower set $\mathbf{M_i}$, each sample has **|N|** Euclidean distances, where **|N|** is the number of samples of normal flower set **N**, sum them and average it, that is, the Euclidean distance between each sample and normal flower set **N**. Since there are eight types of mutant flowers, the project can get eight arrays and the size of each array is the same as the number of samples in $\mathbf{M_i}$, denoted as $\mathbf{D_i}$, where **i** is one of {1, 2, 3, 4, 5, 6, 7, 8}.

Moreover, there exists difference between normal flowers, so it is necessary to calculate the Euclidean distance between each pair of samples in the normal flower set **N**, then sum them and average it, that is, the "**baseline**" value.

Because the difference between normal flowers is small, the value of baseline is relatively small; on the other hand, since the difference between the mutant flower and the normal flower is large, the value of $\mathbf{D_i}$ is larger. For Euclidean Distance, the higher the value, the lower the similarity between the objectives.

3.3.2 Hamming Distance

In the information field, the Hamming distance of two equal-length strings is the number of different characters in the same position, that is, the number of replacements required to replace one string with another.

If want to compare the similarity between two images, constructing a hash value for each image and calculate the number of different bits, that is, Hamming Distance. If this value is 0, it means that the two images are very similar.

There are two common methods to calculate the Hamming distance between two strings:

1. Average Hash

Average hash algorithm is a general term for a class of comparative hashing methods. The features contained in the image are used to generate a set of fingerprints that can be compared. The mainly steps of average hash algorithm shows below:

1) Reduce the size

The fastest way to remove high frequencies and details, leaving only the structure shading is to reduce the size. Reduce the image to 8x8 size for a total of 64 pixels. Discard image differences caused by different sizes and ratios.

2) Simplify the colour

Since the input image is three-color channels (RGB), firstly change the image to grayscale. Convert the reduced image to 64-level grayscale, that is, all pixels have a total of only colours.

3) Calculate the average value and compare the grayscale value of the pixels.

Calculate the grayscale average value of all 64 pixels, then compared with the grayscale value of each pixel. If greater than or equal to the average value, recorded as 1; less than the average value, recorded as 0.

4) Calculate the hash value

Combining the results of the previous comparison, it forms a 644-bit integer, which is the fingerprint of this picture. The order of the combinations is not important, as long as all the pictures are in the same order. If the image is enlarged or reduced, or the aspect ratio is changed, the result will not change. And increasing or decreasing the brightness or contrast value, or changing the colour, does not have much effect on the hash value.

According to the step above, the flower files firstly are read sequentially by normal flower set **N** and mutant flower sets $\mathbf{M_i}$. For each sample of the eight variegated flower groups M_i, limiting their sizes to 8*8 (Figure 3.10), and convert to the grayscale image (Figure 3.10), then calculate the grayscale average of the pixels of the entire image, comparing with each pixel in the image. If the value of pixel is greater than the average value, it is recorded as 1; if it is smaller than the average value, it is recorded as 0.

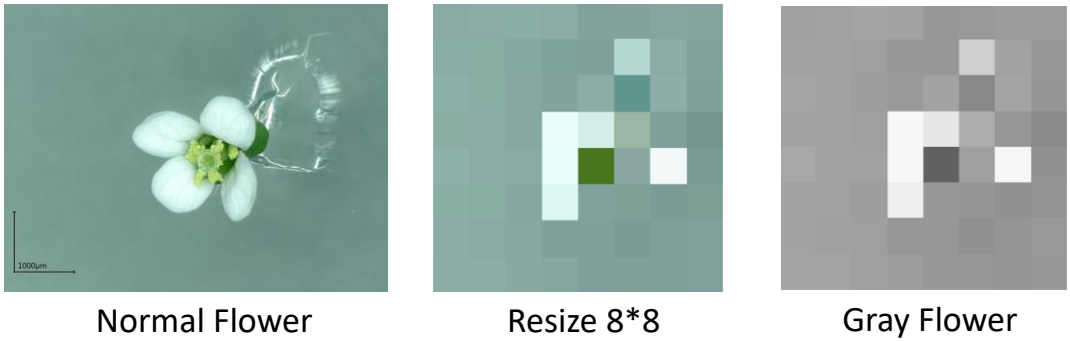

Figure 3.10 Image Process for Average hash

Next, for normal flower group, all samples of it are converted to same form as described above. Then compared with each sample of mutant flower groups, which can get the number of equal characters for the two images at the same position, that is, the similarity between two images.

Because each mutant flower image of $\mathbf{M_i}$ should compare with all samples of flower group **N**, there exists **|N|** number of similarity for each mutant flower, sum them and average it, that is, the similarity between a specific mutant flower and normal flower set **N**, called "$\mathbf{m_j}$", where **j** is one of {1, 2, 3, …, $|M_i|$}. Furthermore, there exists difference between normal flowers, it needs to calculate the similarity between each pair of samples of **N**, called

“**baseline**”. The larger the Hamming distance, the lower the image similarity. Because the normal flowers are more similar, the **baseline** value should be smaller than $\mathbf{m_j}$.

2) Difference Hash

Another method to calculate the similarity between two images is Difference Hash. The main steps is showed as below:

1. Zoom the image

The resolution of the original image is generally very high. A 200*200 image has a full 40,000 pixels. Each pixel holds an RGB value and 40,000 RGB. It is a huge amount of information, and a lot of details need to be processed. So we need to scale the image to very small, hiding the details of it. This project choose to scale the original image to 9*8

2. Convert to grayscale

Difference hash is calculated by calculating the difference in colour intensity between adjacent pixels. The details of the zoomed image have been hidden, and the amount of information has become less. But not enough, because it is coloured and consists of RGB values. White is represented as (255, 255, 255), and black is represented as (0, 0, 0). The larger the value, the brighter the colour, and the smaller the value, the darker.

Each colour consists of three values, which are red, green, and blue. If you use RGB values directly to compare colour intensity differences, it's quite complicated, so we convert to grayscale values - only a single integer from 0 to 255 represents grayscale. This will simplify the three-dimensional comparison for one-dimensional comparison.

3. Difference calculation

The difference value is obtained by calculating the intensity comparison of adjacent pixels in each row. Our picture is 9*8 resolution, then there are 8 lines with 9 pixels per line.

The difference value is calculated separately for each row, that is, the first pixel of the second row is not compared to any pixels of the first row. Each line has 9 pixels, then it will produce 8 difference values, which is why we choose 9 as the width, because 8bit can just form a byte, which is convenient to convert to hexadecimal value.

If the colour intensity of the previous pixel is greater than the second pixel, the difference value is set to True, that is, 1; and if it is not larger than the second pixel, it is set to False, that is, 0.

4. Convert to hash value

Treat each value in the difference value array as a bit, each 8 bits is composed of a hexadecimal value, and the hexadecimal value is concatenated and converted into a string, and the final dHash value is obtained.

5. Hamming Distance

Convert the dHash values of the two images into a binary difference and take the XOR operation. Calculate the number of digits of the "1" of the XOR result, that is, the number of digits that are not the same. This is the Hamming distance.

For normal flower group **N**, resize all samples of **N** to 9*8 and then convert them to grayscale (Fig. 3.11); calculating the hash value for each normal flower image as described above, then comparing with each other to get the Hamming distance. There exists **|N - 1|** Hamming distances for each normal flower image, then sum them and average it, so there are **|N|** Hamming distance for normal flower group **N**. Since we want to find the difference between mutation flower and normal flowers, the normal flower group **N** should be treated as one whole. Add **|N|** Hamming distances to sum and average it, that is, the difference among normal flower group **N**, called "**baseline**".

Next, for each mutant flower of eight mutant flower groups $\mathbf{M_i}$, calculating the corresponding hash value as described above, then compared with the hash value of all normal flowers, which can get the average Hamming distance between each mutant flower and normal flower set **N**, called "$\mathbf{m_j}$", where **j** is one of {1, 2, 3, …, $|M_i|$}.

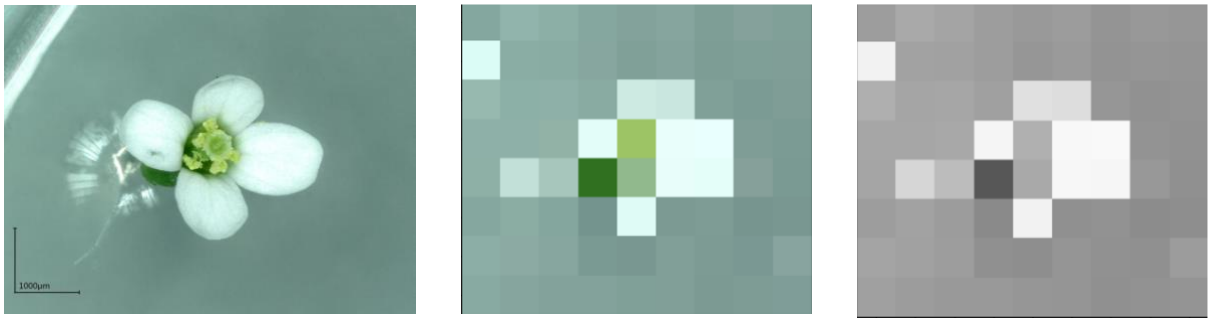

Figure 3.11 Image Process for Difference Hash

3.3.3 Feature Vectors

In addition to directly measuring the Euclidean distance using the raw images, it can also use the features of the image to measure the Euclidean distance between them in different dimensions.

This project attempts to use the pre-trained VGG16 model without the three full connected layer, using only the basic network to extract the basic features of the image, and measures the Euclidean distance based on the feature vector.

First, using the Keras library to obtain a pre-trained VGG16 model based on the ImageNet dataset (see Fig. 3.1), then load the flower data in order and convert it to an array format of (224, 224, 3). Next, add the array into the VGG16 model, the feature vectors of the images should be (1, 7, 7, 512) after processing (see Fig. 3.12).

The indexes of normal flowers and eight kinds of variegated flowers are placed in different lists. Since the features of these pictures have been extracted and stored by the VGG16 network, the index can be used to obtain the feature vectors corresponding to different images, and the Euclidean distance calculation is calculated based on these feature vectors.

input_1 (InputLayer)	(None, 224, 224, 3)	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0

Figure 3.12 Base model for VGG16

For each sample of the eight mutant flower sets $\mathbf{M_i}$, calculating the Euclidean distance between each mutant flower and all samples of normal flower set **N** based on the feature vectors, sum them and average it, that is, the average Euclidean distance between a specific mutant flower and all samples of normal flower set **N**, called "$\mathbf{d_j}$", where **j** is one of {1, 2, 3, …, $|M_i|$}.

For all samples of the normal flower set **N**, calculate the Euclidean distance between any pair of feature vectors, sum them and average it, that is, the average Euclidean distance between normal flowers, called "**baseline**". Since the normal flowers are more similar, the **baseline** value should be smaller than $\mathbf{d_j}$.

3.3.4 Classifier Vectors

The method used above is to use the basic network of the VGG16 model, and freeze the three full connected layers to extract the basic features of the images. However, it is also possible to use fully connected layer according to extract the global features of the images (Fig.3.13).

FC: [1x1x4096] memory: 4096 params: 7*7*512*4096 = **102,760,448**
FC: [1x1x4096] memory: 4096 params: 4096*4096 = 16,777,216
FC: [1x1x1000] memory: 1000 params: 4096*1000 = 4,096,000

Figure 3.13 Parameters of Fully connected layer

Due to the redundancy of the full connection layer parameters, this project uses Global Average Pooling instead of FC to fuse the deep features learned. Finally, the loss function such as softmax is used as the network objective function to guide the learning process.

Global Average Pooling (see Fig. 3.14)is mainly used to solve the problem of full connection. The main purpose is to make the feature map of the last layer into a mean value pool of the whole graph to form a feature point. These feature points are combined into the final feature vector for calculation of softmax.

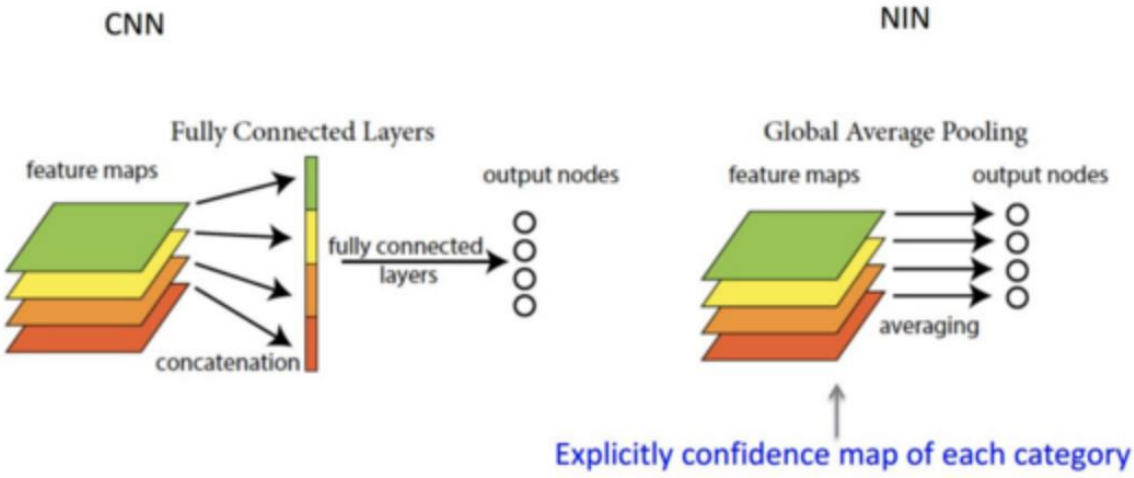

Figure 3.14 Process of Global Average Pooling

The first 13 layers of the pre-trained VGG16 model are frozen, and their weights are inherited. The flower data is used to train the last three layers of the VGG16 model to obtain the classifier, which can classify the flower images into different categories according to its features.

Next, using the trained classifier to extract the feature vectors of all images, including normal flowers and all mutant flowers, then put them into a list. Based on the feature vectors, calculating the Euclidean distance between flower images.

For each sample of the eight mutant flower sets $\mathbf{M_i}$, using the index to get the feature vector of each flower image, which can calculate the Euclidean distance between one mutant flower and all normal flowers, that is, the average Euclidean distance between a specific mutant flower of each mutant group $\mathbf{M_i}$ and all normal flowers of **N**, called "$\mathbf{d_j}$", where **j** is one of {1, 2, 3, …, $|M_i|$}.

For all samples of the normal flower set **N**, using the index to obtain their corresponding feature vectors, and then calculate the Euclidean distance between any pair of feature vectors, selecting the minimum value, that is, the minimum Euclidean distance between normal flowers, called "**baseline**". Because the difference between the mutant flower and the normal flower is greater than the difference between the normal flowers, the baseline value should be smaller than $\mathbf{d_j}$.

3.4.5 Evaluation Method

There are two ways to evaluate the experiments above:

1) Intra-class and Inter-class distance measure

Choose a distance measure as used in the previous study on intra-class and inter-class average distances. There are nine groups of flower images, including one normal type and eight mutant types. For each groups, calculating the average distance between each pair of samples, then sum them and average it, that is, the average distance for intra-class. For nine groups, calculating the average distance between each sample of each group and all samples of other eight groups, then sum them and average it, that is, the average distance for inter-class.

Due to the difference between the mutant flower and the normal flower is greater than the difference between the normal flowers, the average distance of intra-class should be far smaller than the average distance of inter-class.

2) Clustering

If the clustering algorithm is used to cluster nine types of flower images, the better the performance is, which indicates that the experimental method is more sensitive to the difference between the images, that is to say, the difference between images measured by using this experimental method is more accurate.

1. Raw Image

Load the raw images, then transform them into array format and flatten them into one array. Put the array into clustering algorithm to get the predicted labels, then using Munkres algorithm to get the optimal assignment for predicted labels, then compared with the real category of the images, which can get the clustering accuracy.

2. Feature Vectors and Classifier Vectors

The VGG16 model with classifier and the VGG16 model without classifier are used to extract the image features, and then expanded into the array format. The array is imported into the clustering algorithm for clustering, and the prediction results are imported into the Munkres algorithm to obtain the optimal assignment, then comparing with the real category of the images, which can get the clustering accuracy.

3. Average Hash and Difference Hash

The images would be processed by average hash and difference hash to be a matrix which consisting by 0 and 1. Then, flatten the matrix and put it into clustering algorithm, which can get the predicted labels and get the optimal assignment by Munkres algorithm. Finally, the clustering accuracy can be calculated by comparing the real category of the images and the optimized predicted labels.

Chapter 4 Results

4.1 Classification

The data is divided into a training data set, a validation data set and a test data set, wherein data enhancement is performed on the training data set and the validation data set to increase the data amount. After the training data set and validation data set have been processed, they are used as input to re-train the pre-trained VGG16 network by ImageNet, which can get a trained model with the optimal weights.

Then, the test data set is used to obtain the prediction accuracy of the model. The test data set is consisting of 13 normal flowers samples and 3 samples of each types of mutant flower. Because some mutant flower images exist shooting problems, the mutant flower data set only includes five kinds of variegated flowers: Sep1-Variable Temperature, Sep2-Constant Temperature, Sep2-Variable Temperature, Sep3-Variable Temperature and Sep4-Variable Temperature.

After each image is processed by the VGG16 model, the output is an array of 9 values and the subscript of each value represents one of the nine classes, where the subscript of the maximum value is the category to which the image belongs. Comparing with the true value of the flower category, the prediction accuracy of the VGG16 model is obtained.

By 10-fold cross-validation, there are ten experiments for different test data sets and therefore get ten results.

Table 4.1 Classification Accuracy for 10 experiments

1	2	3	4	5	6	7	8	9	10
62%	58%	57%	76%	68%	59%	65%	62%	59%	46%

Based on 10 results above, the average is obtained, which is the final model prediction accuracy: **61%**.

4.2 Difference Measure for Images

Because there is too much data, only one of the variegated flower collections sep1-Variable Temperature is displayed as figure 4.1, which is the sep1 variegated flower at normal temperature. There are several samples of normal flowers show as figure 4.2.

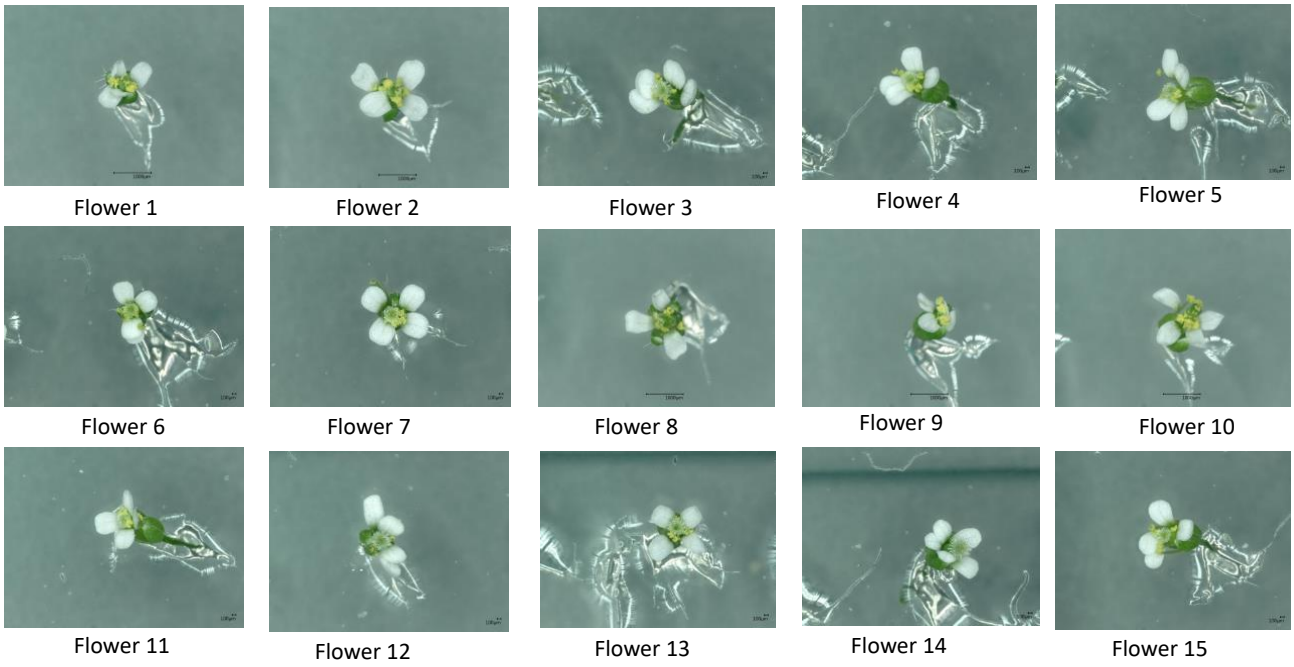

Figure 4.1 Sep1-Variable Temperature mutant flowers

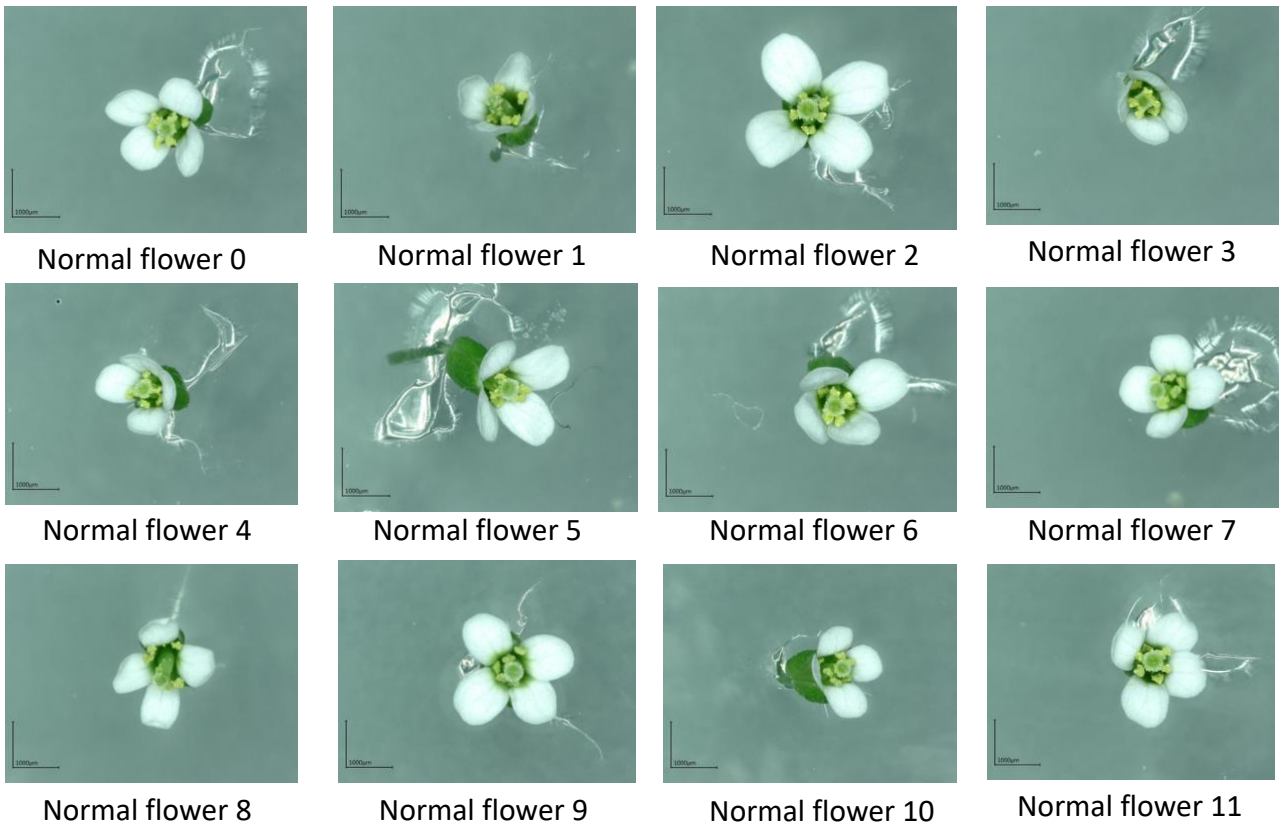

Figure 4.2 Normal flower samples

4.2.1 Raw Images

The table 4.2 shows the name of the flowers, the average distance between the mutant flower and normal flowers, the baseline distance between the normal flowers, and the difference between the mutant flower and the normal flowers.

Table 4.2 Difference Identification for Raw Images

Name	Average Distance	Baseline
Flower 1	10904.65	14275.57
Flower 2	11894.35	14275.57
Flower 3	14894.62	14275.57
Flower 4	14809.02	14275.57
Flower 5	15078.82	14275.57
Flower 6	15049.48	14275.57
Flower 7	13761.06	14275.57
Flower 8	11612.11	14275.57
Flower 9	11850.01	14275.57
Flower 10	11627.75	14275.57
Flower 11	12925.60	14275.57
Flower 12	12378.70	14275.57
Flower 13	14279.27	14275.57
Flower 14	16205.21	14275.57
Flower 15	14422.87	14275.57

4.2.2 Feature Vectors

The table 4.3 shows the name of the flowers, the average distance between the mutant flower and normal flowers, the baseline distance between the normal flowers, and the difference between the mutant flower and the normal flowers.

Table 4.3 Difference Identification for Feature Vectors

Name	Average Distance	Baseline
Flower 1	35.87	38.07
Flower 2	34.78	38.07
Flower 3	41.28	38.07
Flower 4	38.19	38.07
Flower 5	39.64	38.07
Flower 6	40.16	38.07
Flower 7	33.47	38.07
Flower 8	35.28	38.07
Flower 9	39.33	38.07
Flower 10	35.55	38.07
Flower 11	37.90	38.07
Flower 12	35.93	38.07
Flower 13	38.51	38.07
Flower 14	42.10	38.07
Flower 15	36.62	38.07

4.2.3 Classifier Vectors

The table 4.4 shows the name of the flowers, the average distance between the mutant flower and normal flowers, the baseline distance between the normal flowers, and the difference between the mutant flower and the normal flowers.

Table 4.4 Difference Identification for Classifier Vectors

Name	Average Distance	Baseline
Flower 1	0.38	0.34
Flower 2	0.71	0.34
Flower 3	0.97	0.34
Flower 4	0.98	0.34
Flower 5	1.02	0.34
Flower 6	1.00	0.34
Flower 7	0.80	0.34
Flower 8	0.47	0.34
Flower 9	0.44	0.34
Flower 10	0.50	0.34
Flower 11	0.91	0.34
Flower 12	0.83	0.34
Flower 13	1.01	0.34
Flower 14	0.86	0.34
Flower 15	0.84	0.34

4.2.4 Average Hash

The table 4.5 shows the name of the flowers, the average distance between the mutant flower and normal flowers, the baseline distance between the normal flowers, and the difference between the mutant flower and the normal flowers.

Table 4.5 Difference Identification for Average Hash

Name	Average Distance	Baseline
Flower 1	23.98	23.25
Flower 2	25.20	23.25
Flower 3	22.70	23.25
Flower 4	23.29	23.25
Flower 5	24.03	23.25
Flower 6	20.67	23.25
Flower 7	25.53	23.25
Flower 8	21.23	23.25
Flower 9	20.77	23.25
Flower 10	21.21	23.25
Flower 11	21.89	23.25
Flower 12	20.67	23.25
Flower 13	21.20	23.25
Flower 14	22.83	23.25
Flower 15	23.80	23.25

4.2.5 Difference Hash

The table 4.6 shows the name of the flowers, the average distance between the mutant flower and normal flowers, the baseline distance between the normal flowers, and the difference between the mutant flower and the normal flowers.

Table 4.6 Difference Identification for Difference Hash

Name	Average Distance	Baseline
Flower 1	27.04	28.84
Flower 2	28.52	28.84
Flower 3	30.02	28.84
Flower 4	30.57	28.84
Flower 5	31.90	28.84
Flower 6	33.60	28.84
Flower 7	29.01	28.84
Flower 8	29.60	28.84
Flower 9	31.08	28.84
Flower 10	32.51	28.84
Flower 11	28.93	28.84
Flower 12	29.23	28.84
Flower 13	31.77	28.84
Flower 14	32.75	28.84
Flower 15	30.98	28.84

Chapter 5 Evaluation

5.1 Classification

By 10-fold cross-validation, the experiment should select a non-repeating part of the whole flower data set as the test data set. For normal flowers, 13 images are selected as part of the test data set each time and each time is different. For the mutant flowers, each type of mutant flower data set is selected 3 images without repeating each time and put them into test data set. The confusion matrix of classification result shows below:

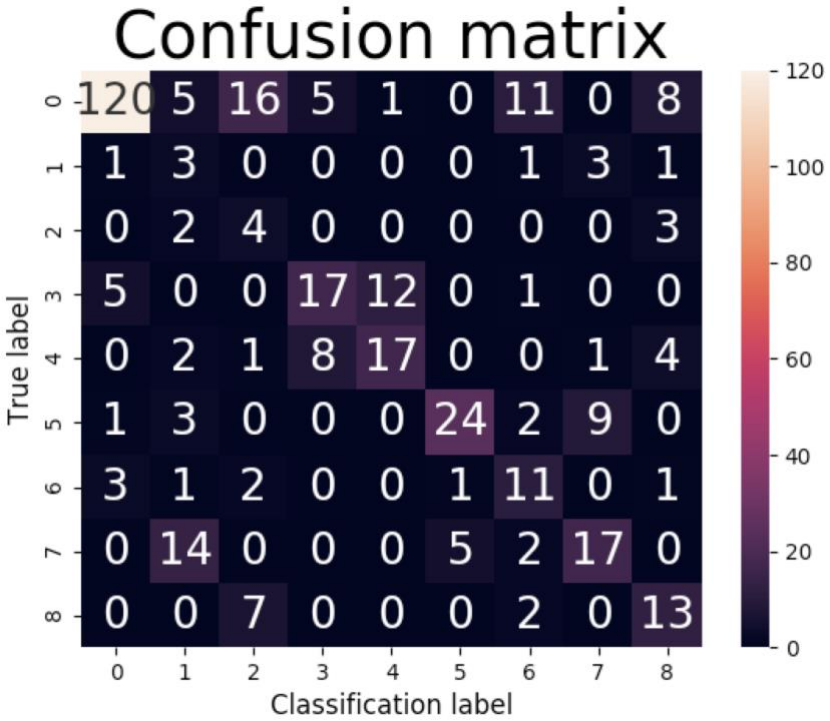

Figure 5.1 Confusion Matrix of Classification result

According to the confusion matrix above, it's easy to see that some variegated flower types are easily confused, and some variegated flowers are very similar to normal flowers, so they are easily confused, see Table 5.1:

Table 5.1 Analyse for Classification Confusion Matrix

Normal Flower	Most normal flowers are correctly classified, and the difference from sep1-V, sep2-V and sep4-C is very obvious.
SEP1-Constant	Sep1-C mutant flower is significant different from sep2-C, but is easily

Temperature	confused with sep4-C.
SEP1-Variable Temperature	Sep1-V is easily confused with normal flower, but is different from sep1-C, sep2-C, sep3-C and sep4-C.
SEP2-Constant Temperature	Sep2-C is easy confused with sep2-V, but most of sep2-C flower can be correctly classified.
SEP2-Variable Temperature	Sep2-V is easy confused with sep2-C, but most of sep2-V flower can be correctly classified.
SEP3-Constant Temperature	Most of sep3-C flower can be correctly classified.
SEP3-Variable Temperature	Sep3-V is easily confused with normal flower.
SEP4-Constant Temperature	Sep4-C is easily confused with sep3-C.
SEP4-Variable Temperature	Part of sep4-V flower is easily confused with normal flower.

5.2 Difference Measure for Images

Based on the evaluation method above, the higher the accuracy of clustering, the better the performance of the method. The evaluate result for each experiment shows below:

5.2.1 Raw Images

According to the table 4.2, for flower 1, 2, 7, 8, 9, 10, 11 and 12, the average distance between them and normal flowers are smaller than the baseline value, this is not accurate since these flowers are far different from normal flowers by observing the figure 4.1.

1) Intra-class and Inter-class distance

Table 5.2 shows the average distance of intra-class is smaller than the average distance of inter-class for Raw Images, but the gap is not obvious.

Table 5.2 Results for Raw Images

Intra-class average distance	**Inter-class average distance**
14122.17	14895.33

2) Clustering accuracy

The figure 5.2 shows the confusion matrix and the result shows the clustering accuracy is 52%:

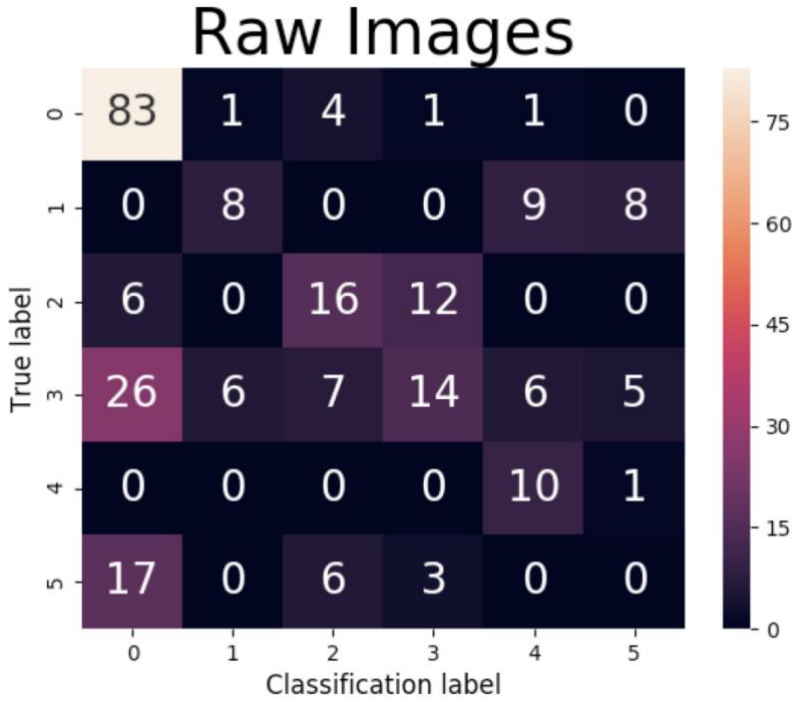

Figure 5.2 Confusion Matrix for Raw Images

5.2.2 Feature Vectors

According to the table 4.3, for flower 1, 2, 7, 8, 9, 10, 11, 12 and 15, the average distance between them and normal flowers are smaller than the baseline value, this is not accurate since these flowers are far different from normal flowers by observing the figure 4.1.

1) Intra-class and Inter-class distance

Table 5.3 shows the average distance of intra-class is smaller than the average distance of inter-class for Feature Vectors, but the gap is not obvious.

Table 5.3 Result for Feature Vectors

Intra-class average distance	Inter-class average distance
35.67	38.83

2) Clustering accuracy

The figure 5.3 shows the confusion matrix below and the result shows the clustering accuracy is 46%.

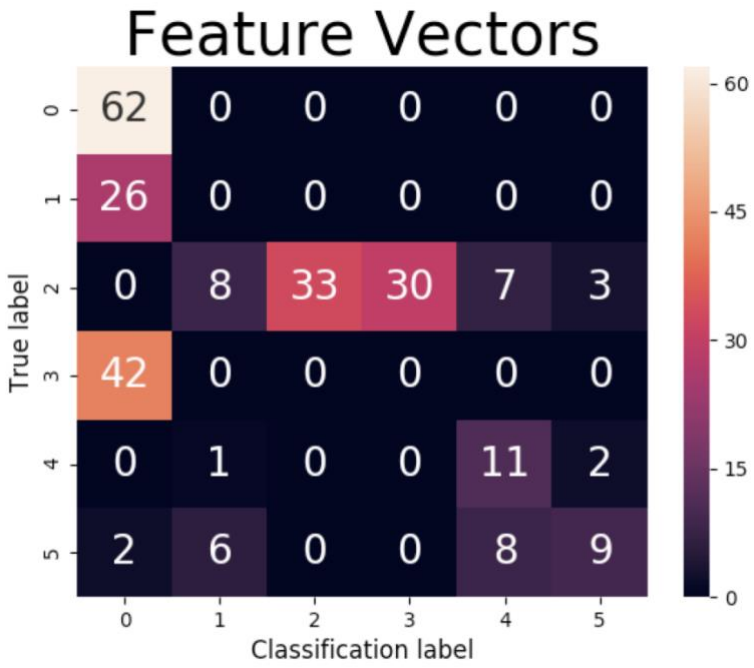

Figure 5.3 Confusion Matrix for Feature Vectors

5.2.3 Classifier Vectors

According to the table 4.4, for all flowers, the average distance between them and normal flowers are bigger than the baseline value, this is correct since these flowers are far different from normal flowers by observing the figure 4.1.

It's obviously that the difference of Flower 2 is smaller than the difference of Flower 1, but the table 4.4 shows the opposite.

1) Intra-class and Inter-class distance

Table 5.4 shows the average distance of intra-class is smaller than the average distance of inter-class for Classifier Vectors, and the gap is obvious.

Table 5.4 Result for Classifier Vectors

Intra-class average distance	Inter-class average distance
0.17	0.67

2) Clustering accuracy

The figure 5.4 shows the confusion matrix below and the result shows the clustering accuracy is 67%.

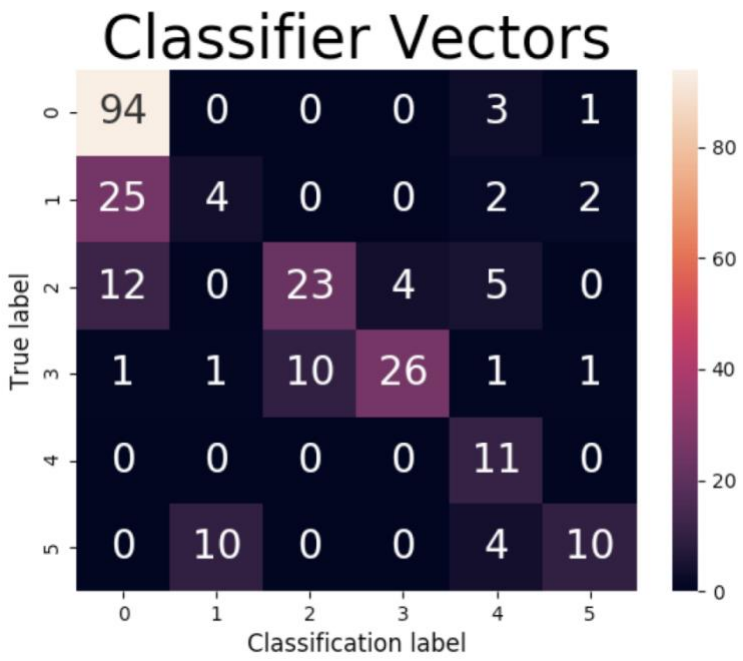

Figure 5.4 Confusion Matrix for Classifier Vectors

5.2.4 Average Hash

According to the table 4.5, for flower 3, 6, 8, 9, 10, 11, 12, 13 and 14, the average distance between them and normal flowers are smaller than the baseline value, this is not accurate since these flowers are far different from normal flowers by observing the figure 4.1.

1) Intra-class and Inter-class distance

Table 5.4 shows the average distance of intra-class is bigger than the average distance of inter-class for Average Hash, which indicates this method is not accurate.

Table 5.5 Result for Average Hash

Intra-class average distance	Inter-class average distance
22.83	22.33

2) Clustering accuracy

The figure 5.5 shows the confusion matrix below and the result show the clustering accuracy is 37%.

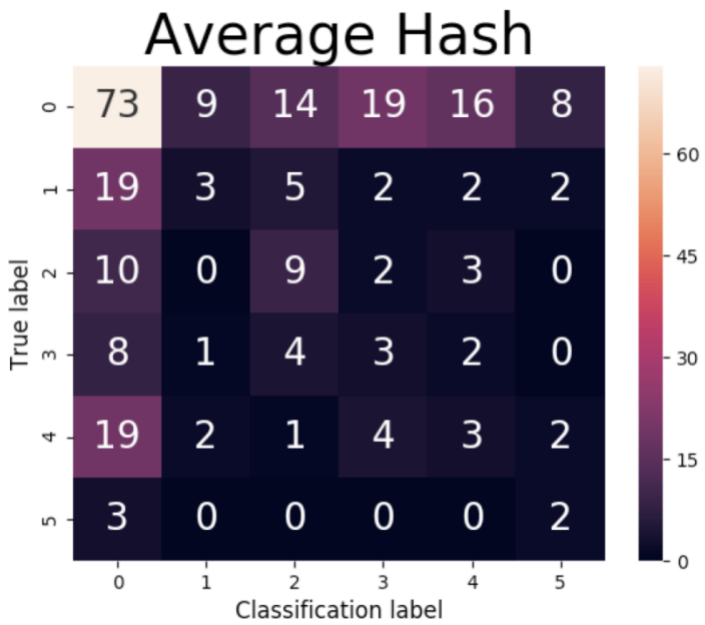

Figure 5.5 Confusion Matrix for Average Hash

5.2.5 Difference Hash

According to the table 4.6, for flower 1 and 2, the average distance between them and normal flowers are smaller than the baseline value, this is not accurate since these flowers are far different from normal flowers by observing the figure 4.1.

1) Intra-class and Inter-class distance

Table 5.4 shows the average distance of intra-class is smaller than the average distance of inter-class for Difference Hash, but the gap is not obvious.

Table 5.6 Result for Difference Hash

Intra-class average distance	Inter-class average distance
30.00	30.50

2) Clustering accuracy.

The figure 5.6 shows the confusion matrix below and the result shows the clustering accuracy is 24%.

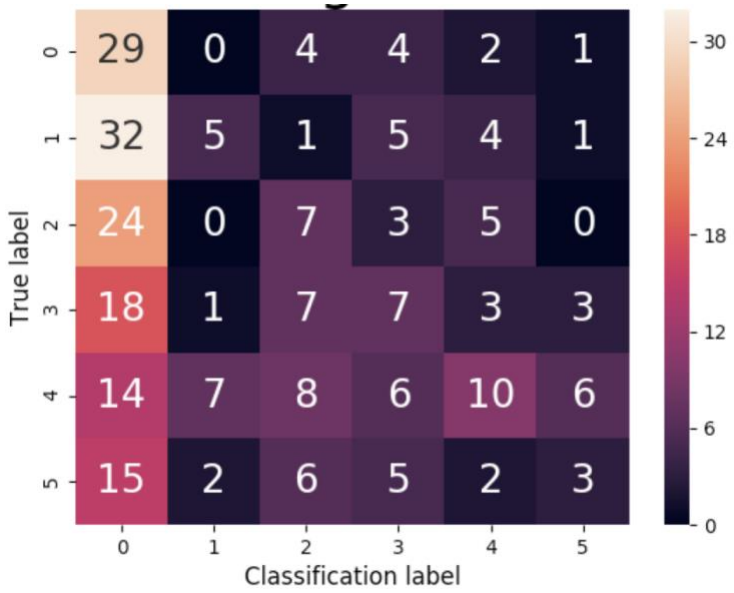

Figure 5.6 Confusion Matrix for Difference Hash

Chapter 6 Conclusion

6.1 Project Conclusion

The project uses a pre-trained deep learning network to extract flower features and automatically classify nine flower types, including one normal flower and eight variant types. The evaluation showed that the classification performed well, and the flowers could be classified, and the flower types with smaller differences were displayed by the confusion matrix.

Furthermore, the project uses the Euclidean distance and the Hamming distance to measure the similarity of flowers and obtain the difference between the flowers of the eight mutations and the normal flowers. There is the summarise for five evaluation results above, the table 6.1 shows below:

Table 6.1 Summarize for Evaluation

Methods	**Raw Images**	**Feature Vectors**	**Classifier Vectors**	**Average Hash**	**Difference Hash**
Accuracy	52%	46%	67%	37%	24%

The Evaluation shows that the performance is the best by using classifier vectors method. The pre-trained VGG16 network and the flower data are used to train the classifier, then the classifier is used to extract the features of the flower data. Based on the feature vectors of flower, calculating the Euclidean distance between two flowers, which can be used to calculate the difference between two flowers.

6.2 Challenges

1. The experimental flowers were influenced by four genes, and eight types of mutations were produced. Some of the differences between the types of variation and normal flowers were very small, which affected the classification effect.

2. Because the flowers are very small, special equipment is needed for photographing and sampling, which makes the experimental data more difficult to get and affects the training effect of the deep learning network.

3. Due to the light and angle of the images, the background of the flower picture is quite different, and some pictures include more than one flower, which greatly affects the judgment of the difference between flowers.

6.3 Project Limitation

This project uses the deep learning network to extract flower features, and then judges the difference between flowers. Therefore, the quality of flower pictures has a great influence on the experimental results. Therefore, when obtaining flower picture samples, it is necessary to pay attention to the consistency of sample photos.

6.4 Future Work

1. In case of sufficient time, sufficient picture data should be obtained to train the classifier of the VGG16 model, and K-fold Cross-Validation is used multiple times to obtain the most accurate classification.

2. This project only uses the VGG16 model, and does not use the remaining deep neural network models. It should try to use other deep learning networks to extract image features to determine difference between images. In addition, it should try more ways to get the difference between images, such as Histogram and Perceptron Hash.

List of References

Chen, C. and Chang, W. A feedforward neural network with function shape autotuning. *Neural Networks*. [Online]. **09**(04), pp. 627-641. [Accessed 10 August 2018]. Available from: www.sciencedirect.com

Dalcín, L., Paz, R. and Storti, M. MPI for Python. *Journal of Parallel and Distributed Computing*. [Online]. **65**(09), pp.1108-1115. [Accessed 10 August 2018]. Available from: https://www.sciencedirect.com

Department of Engineering Science. Visual Geometry Group. [Online]. [Accessed 10 August 2018]. Available from: http://www.robots.ox.ac.uk

Fushiki, T. Estimation of prediction error by using K-fold cross-validation. *Statistics and Computing*. [Online]. **21**(02), pp.137-146. [Accessed 10 August 2018]. Available from: https://link.springer.com

Gedda, R. CIO Blast from the Past: 60 years of Hamming codes. CIO. [Online]. [Accessed 10 August 2018]. Available from: www.cio.com.au

Jigsaw. 5 Essential Skills Every Big Data Analyst Should Have. 09 February. *JIGSAW ACADEMY*.[Online]. [Accessed 10 August 2018]. Available from: www.jigsawacademy.com

Joyce, K., Laurienti, P., Burdette, J. and Hayasaka, S. A New Measure of Centrality for Brain Networks. *PLoS ONE*. [Online]. **05**(08). [Accessed 10 August 2018]. Available from: http://journals.plos.org

Karpathy, A. CS231n Convolutional Neural Networks for Visual Recognition. [Online]. [Accessed 2 August 2018]. Available from: http://cs231n.github.io/transfer-learning

Keras Documentation. 2018. Keras: The Python Deep Learning library. [Online]. [Accessed 10 August 2018]. Available from: https://keras.io/

Kumar, A. Introduction to Machine Learning. All Programming Tutorials. [Online]. [Accessed 10 August 2018]. Available from: www.allprogrammingtutorials.com

Lecun, Y., Haffner, P., Bottou, L and Bengio, Y. Object Recognition with Gradient-Based Learning. [Online]. [Accessed 10 August 2018]. Available from: http://yann.lecun.com

Mitchell, T. 1997. *Machine Learning*. New York: McGraw-Hill Science. [Accessed 10 August 2018]. Available from: www.cs.ubbcluj.ro

Sivarajah, U., Kamal, M., Irani, Z. and Weerakkody, V. Critical analysis of Big Data challenges and analytical methods. *Journal of Business Research*. [Online]. 70, pp. 263-286. [Accessed 10 August 2018]. Available from: www.sciencedirect.com

Sohangir, S., Wang D., Pomeranets, A and Khoshgoftaar, T. Big Data: Deep Learning for financial sentiment analysis. *Journal of Big Data*. [Online]. [Accessed 2 August 2018]. Available from: https://link.springer.com

Stanford Vision Lab. IMAGENET. [Online]. [Accessed 15 August 2018]. Available from: http://www.image-net.org/

Stone M. Cross-validatory choice and assessment of statistical predictions. J. Royal Stat. Soc, 36(2), 111–147, 1974

Wikipedia. Euclidean Distance. [Online]. [Accessed 10 August 2018]. Available from: https://en.wikipedia.org

Appendix A External Material

1. Keras Library: This library is used to load the VGG16 model and flower images to get the features.

2. Sklearn Library: This library is used to get the confusion matrix of classification result and clustering result.

3. Matplotlib Library: This library is used to draw the plots.

4. Numpy Library: This library is used to implement the array operation.

5. Os Library: This library is used to read the files from the directory.

Appendix B Code

There are 14 programs for this project, including classification, difference measure and evaluation. The GitHub URL: https://github.com/Yunpneg/Master-Project.git

1. Classification Program

This part has two programs: (1.1)Re-train VGG16 model to get the optimal weights and (1.2)Classification for Images

2. Difference Measure for Images

This part has four programs: (2.1)Raw Images and Feature Vectors, (2.2)Classifier Vectors, (2.3)Average Hash and (2.4)Difference Hash.

3. Evaluation

This program has two parts: (3.1)Intra-class and Inter-class average distance and (3.2)Clustering accuracy.

For 3.1, there are five programs: (3.1.1)Raw Images, (3.1.2)Feature Vectors, (3.1.3)Classifier Vectors, (3.1.4)Average Hash and (3.1.5)Difference Hash.

For 3.2, there are three programs: (3.2.1)Raw Images and Feature Vectors, (3.2.2)Classifier Vectors and (3.2.3)Average Hash and Difference Hash.

Appendix C Ethical Issues Addressed

This exploratory software project is void of any ethical issue.