Flower Classification with Traditional and Deep Learning Methods

FENG Xuening
SJTU-UM JI
019370910019

HU Zongyang
SJTU-UM JI
019370910024

XIAO Yuhang
SJTU-UM JI
119370910010

LI Zehuan
SJTU-UM JI
119370910018

Abstract

Image classification has been a major task in the domain of computer vision. In this project, we work with scraped flower images and classify each of these images into one of five types of flowers. To perform the classification, we apply both traditional (Support Vector Machine, SVM) and deep learning methods (Convolutional Neural Network, CNN and transfer learning) in this task. Experimental results show that the transfer learning method can achieve a better performance than SVM and pure CNN model, since the dataset is relatively small with limited number of training images for each class. To further improve the performance, more suitable features for SVM and more ways to augment the dataset could be investigated.

1. Introduction

Image classification can be defined as the task of categorizing images into one of several predefined classes. It is a fundamental problem in computer vision. And it forms the basis of other computer vision tasks such as localization, detection, and segmentation. Although the task can be considered second nature for humans, it is much more challenging for an automated system.

Traditional approaches for image classification tasks have a dual-stage routine. Firstly, handcrafted features are extracted from raw pixels using feature descriptors, and then these features served as input to a trainable classifier. With well-selected features, traditional methods, such as Support Vector Machine (SVM), can be extremely effective and achieve good performance. On the other hand, this means the accuracy of these approaches is highly dependent on the design of the feature extraction stage, and this usually proved to be a formidable task[6].

Thanks to the development of deep learning (DL) techniques, DL models have exploited multiple layers of nonlinear information processing for feature extraction automatically. This have been shown to overcome traditional challenges. Among them, convolutional neural networks (CNNs)[6] have become the leading architecture for most image recognition, classification, and detection tasks[5]. However, CNNs still work like a black-box which humans can not explicitly understand. Besides, such neural networks usually require large amount of data, which may be not always easy to obtain, to train a good model. In such cases, knowledge transfer or transfer learning[7], which means utilizing knowledge from a mature domain to another data-starved domain, would be highly beneficial.

In this project, we conduct analysis on the classification of flower images. Specifically, we are applying both traditional classification method and deep learning method to classify a given flower image as one of the five types: daisy, tulip, rose, sunflower and dandelion.

2. Methods

2.1. Support Vector Machine (SVM)

2.1.1 Brief Introduction of SVM

Support Vector Machine (SVM) is a traditional supervised learning model used for classification[1]. Developed at AT&T Bell Laboratories by Vapnik with colleagues, it presents one of the most robust prediction methods, based on the statistical learning framework or VC theory proposed by Vapnik and Chervonekis and Vapnik. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier (although methods such as Platt scaling exist to use SVM in a probabilistic classification setting). An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on the side of the gap on which they fall.

In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

On top of binary classification, SVMs can also do multiclass classification, where the labels are drawn from a finite

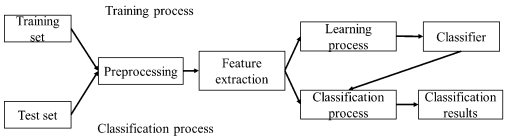

Figure 1. SVM classification process.

set of several elements. The dominant approach for doing so is to reduce the single multiclass problem into multiple binary classification problems[2] [4]. Building binary classifiers that distinguish between one of the labels and the rest (one-versus-all) or between every pair of classes (one-versus-one). Classification of new instances for the one-versus-all case is done by a winner-takes-all strategy, in which the classifier with the highest-output function assigns the class (it is important that the output functions be calibrated to produce comparable scores). For the one-versus-one approach, classification is done by a max-wins voting strategy, in which every classifier assigns the instance to one of the two classes, then the vote for the assigned class is increased by one vote, and finally the class with the most votes determines the instance classification.

2.1.2 SVM in Our Project

For the flowers classfication problem, we use one-versus-all non-linear SVM to do our experiments. By extracting HOG and GLCM features from training images, the initial feature spaces can be set. Then RBF kernel is chosen for mapping the features to high-dimensional fea- ture spaces. By contrasting the scores gotten by 10 binary classifiers, we chose the highest score to assign samples to corresponding classes.

The detailed steps of applying SVM for classification are as follows: first collect the training set and test set of each class, next select the appropriate image features for classification, extract features from the training set, and then train with the SVM classifier to obtain the classification template, finally pass the template Classify the images to be classified. The whole process is shown in Figure. 1.

The feature selection process is divided into two steps:

- Visual inspection. The color, texture and shape of the two kinds of images are analyzed, and the features that may separate the two types are selected.

- Experiment. The feature is extracted and trained by SVM. The test result is good or bad to decide whether to optimize (such as block) or replace other features.

In this experiment, two features are selected: directional gradient histogram and gray level co-occurrence matrix.

- Histogram of Oriented Gradients (HOG)

HOG feature descriptor, which is widely used in image processing for object detection, constructs features by calculating and counting the histogram of gradient direction of local area of image. Extracting HOG features includes the following steps:

Normalized image. Firstly, the input color image is transformed into gray image, and then the image is compressed by square root gamma to achieve the normalization effect. This compression process can effectively reduce the local shadow and illumination changes of the image, so as to improve the robustness of hog features to illumination changes.

Calculate the image gradient. Firstly, the normalized image is convoluted with one-dimensional discrete differential template $[-1, 0, 1]$ and its transposition, and the gradient components in horizontal direction and vertical direction are obtained. Then, according to the horizontal gradient and vertical gradient of the current pixel, the gradient amplitude and gradient direction of the current pixel are obtained. The formula is as follows:

$$\begin{aligned} G_x(x,y) &= H(x+1,y) - H(x-1,y) \\ G_y(x,y) &= H(x,y+1) - H(x,y-1) \\ G(x,y) &= \sqrt{G_x(x,y)^2 + G_y(x,y)^2} \\ \alpha(x,y) &= \tan^{-1}\frac{G_y(x,y)}{G_x(x,y)} \end{aligned} \tag{1}$$

The histogram of gradient direction was constructed for each cell unit. Firstly, the 64×128 image is divided into 8×16 cells, that is, each cell is 8×8 pixels. Then the gradient direction is limited to $[0, \pi]$, and the gradient direction is divided into 9 intervals (bin), each of which is 20 degrees. Finally, each pixel in the cell is weighted in the histogram with the gradient direction. That is to say, each pixel in the cell votes for bin in a certain direction according to the gradient amplitude of the pixel. In this way, the gradient direction histogram of the cell can be obtained, which is the 9-dimensional feature vector corresponding to the cell.

The cell units are combined into large blocks, and the normalized gradient histogram within the block. The adjacent 2×2 cells are formed into a block, so each block corresponds to a 36-dimensional eigenvector. The variation range of gradient intensity is very large due to the change of local illumination and

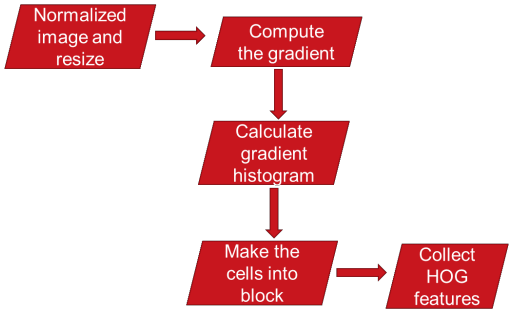

Figure 2. HOG features extraction.

the contrast of foreground and background. In order to further eliminate the influence of illumination, the 36-dimensional feature vectors in the block are normalized.

Collect HOG characteristics. As shown in Fig. 1, the method of sliding window is used to scan the sample image with block. The scanning step is a cell, so there is overlap between blocks. Finally, all normalized block features are concatenated to get 3780-dimensional feature vectors.

The whole process of extracting HOG features is shown in Fig. 2.

- Gray Level Co-occurrence Matrix (GLCM)

The second feature we choose is the GLCM. The gray level co-occurrence matrix of an image can reflect the comprehensive information about the direction, adjacent interval and change range of the image gray level. It is the basis of analyzing the local patterns of the image and their arrangement rules. In 1973, Haralick studied the spatial dependence of gray levels in image texture from the perspective of pure mathematics, and proposed a texture description method of gray level co-occurrence matrix[3]. The essence of this method is to count the number of times $p(i, j, d, \theta)$ of pixels with gray level of $(x + Dx, y + Dy)$ whose distance is D and whose gray level is $J(x + Dx, y + Dy)$.

$$\begin{aligned} p(i, j, d, \theta) = [(x, y), (x + Dx, y + Dy) \\ |f(x, y) = i, \\ f(x + Dx, y + Dy) = j] \end{aligned} \quad (2)$$

Many texture features can be derived from GLCM, and 14 kinds of GLCM feature statistics can be calculated. The gray level co-occurrence matrix of a certain neighborhood is obtained for each pixel in the image, and then the statistics are obtained from the gray level co-occurrence matrix to obtain the statistics of the corresponding texture image. Some statistics can be used to form the feature vector of image classification.

These four features are uncorrelated, which can effectively describe the texture features of optical or remote sensing images, which is easy to calculate and has good discrimination ability. Due to the large gray level of the original image to be processed, the gray level is compressed to 9 levels from the calculation time and texture separability. Considering the rotation invariance of the parameters, the mean values in four directions are selected as the texture feature parameters, and the step size $D = 1$. According to the definition of the gray level co-occurrence matrix, the gray level co-occurrence matrix of the original image is obtained, and the four texture features under the gray level co-occurrence matrix are calculated as the input of the classifier.

2.2. Convolutional Neural Network (CNN)

2.2.1 Brief Introduction of CNN

Convolutional Neural Network is a famous deep learning method architecture derived from natural visual perception mechanism. And it was commonly applied to image classification task.

Most notably, Krizhevesky proposed their classical CNN architecture and had a huge improvement on image classification task. Their method, AlexNet[8], has a deeper structure and more accurate result. With the success of krizhevesky, many other similar methods proposed, such as ZFNet[11], VGGNet[9], and GoogleNet[10]. The typical evolutional trend of CNN is deeper and more complex, but those more complex model may cause overfitting problem.

CNN can mainly be divided into three basic components, convolutional, pooling, and fully connected layers. The convolutional layer aims to extract features from the input image. The activation function introduces nonlinearities to CNN, which is desirable to detect nonlinear features. The pooling layer aims to achieve shift-invariance by reducing the resolution of the feature maps. The fully connected layer performs the high-level reasoning[9] [11], and connects the perceptions obtained from previous layers to generate global semantic information, such as classification species.

2.2.2 CNN in Our Project

Our simple model is based on the previous theory. Firstly, we applied four convolutional layers to our model. The configuration of these layers is shown in Table. 1 The convolutional layer is a generalized linear model for the underlying local image patch. And We use ReLU function as our activation function. The ReLU is Rectified linear unit activation

Layer	Filter	Kernel Size	Activation
Conv2D	32	5×5	ReLU
Conv2D	64	3×3	ReLU
Conv2D	96	3×3	ReLU
Conv2D	96	3×3	ReLU

Table 1. Convolutional layers configuration of the CNN model.

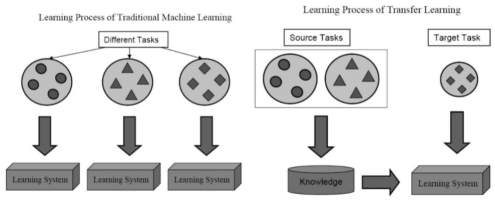

Figure 3. Different learning processes between (left) traditional machine learning and (right) transfer learning.

function. It is a piecewise linear function which prunes the negative part to zero and returns the positive part. It works better than most other activation function.

After each convolutional layer, we add a 2×2 pixel size max pooling layer to reduce the resolution of the previous feature map. Our pooling layer is the max pooling layer.

Then, we applied the two fully-connected layers to the top. Firstly, we connects previous result to 512 parts, and then connected them to our five classification species. After the first fully-connected layer, we add a ReLU activation layer, then the final fully-connected layer we add softmax function as its loss function.

2.3. Transfer learning

2.3.1 Brief Introduction of Transfer Learning

Transfer learning[7] has emerged as a new learning framework for data mining and machine learning tasks in recent years. It basically means to apply the knowledge for one task in a domain with sufficient training data to a similar enough task in another domain, where expensive data-labeling efforts are needed to collect enough training data. And Fig. .3 shows the different learning processes between traditional machine learning and transfer learning.

According to which part of knowledge can be transferred across domains or tasks, transfer learning can be divided into four categories, that is, instance-transfer, feature-representation-transfer, parameter-transfer and relation-knowledge-transfer. Among these four ways of transfer learning, parameter-transfer is mostly used in neural network to tackle the problem of lacking training data. This is also what we mainly used in this project.

In computer vision area, the image dataset ImageNet is the most widely used large-scale one, with more than 10 million images and objects of over 1000 categories. It

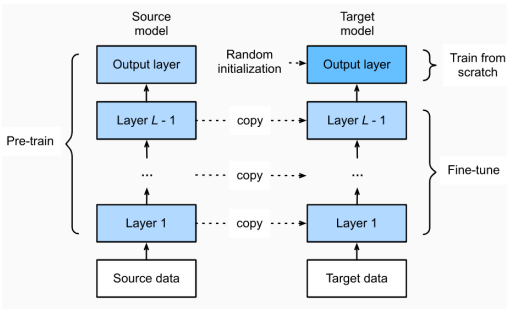

Figure 4. Fine tuning.

is obvious that a deep neural network classification model trained on ImageNet can have pretty good performance. Therefore, when only having limited amount of data for a classification task, we can apply transfer learning to migrate the knowledge learned from ImageNet to the our much smaller dataset. One most common technique to do so is fine tuning, which means making use of the network model that have already been trained and tailoring it to our own task.

As shown in Fig. .4, fine tuning consists of four steps.

First, pre-train a neural network model, i.e., the source model, on a source dataset (e.g., the ImageNet dataset).

Second, create a new neural network model, i.e., the target model. This replicates all model designs and their parameters on the source model, except the output layer, which is closely related to the labels of the source dataset and is therefore not used in the target model.

Then, add an output layer whose output size is the number of target dataset categories to the target model, and randomly initialize the model parameters of this layer.

Finally, train the target model on a target dataset. We will train the output layer from scratch, while the parameters of all remaining layers are fine-tuned based on the parameters of the source model.

2.3.2 Transfer Learning in Our Project

For this flower classification task, since the dataset is too small to train a satisfying neural network model from scratch, we then build our model based on the well-known VGG16[9] model using transfer learning techniques.

VGG16 is a convolutional neural network model proposed by K. Simonyan and A. Zisserman. The model achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes.

The architecture of VGG16 model is shown in Fig. .5. It takes a fixed size 224×224 RGB image as input. The image is passed through a stack of blocks, which consist of two or three convolutional layers and one max-pooling layer. Three fully-connected (FC) layers just follow these

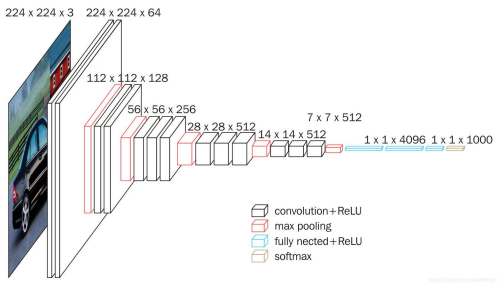

Figure 5. VGG16 architecture.

Layer (type)	Output Shape	Number of Parameter
vgg16 (Model)	(None, 512)	14714688
dense7 (Dense)	(None, 256)	131328
dense8 (Dense)	(None, 5)	1285

Table 2. Network configurations of the first model.

blocks. The first two have 4096 channels each, the third performs 1000-way classification based on its dataset and thus contains 1000 channels (one for each class). The final layer is the soft-max layer.

According to our flower classification task, since there are only five kinds of flowers, we build the first model using VGG16 but replace the FC layers of VGG16 with our own. In this model, we reuse the weights of VGG16 trained on ImageNet. VGG16 without FC layers works as a pretrained feature extractor here, so that we can just train our task-specific classifier based on the features. Network configurations of the **first model** are shown in Table. 2.

However, although there is similarity between ImageNet and our dataset, the results may be less satisfying by directly training a classifier on a pretrained model. This is because in deep neural networks, higher layers always learn more task-specific features, which are not suitable to transfer, and lower layers always learn more generalized features.

Therefore, we then build two more models by fine tuning, which refers to successively unfreezing some top block(s) of the base VGG16 model. That is, in the second model, the last block of the base model is unfreezed and will be trained. Similarly, in the third model, the last two blocks of the base model is unfreezed and will be trained. And for these two models, the network architecture and configurations are all same with the first one.

3. Experiments

3.1. Dataset

The dataset for this project contains 4323 images of flowers. Each of the image belongs to either of five types, which are daisy, tulip, rose, sunflower and dandelion. And some examples are shown in Fig. 6.

Figure 6. Dataset visualization.

Class	Daisy	Dandelion	Rose	Sunflower	Tulip
Daisy	131	29	14	12	14
Dandelion	34	120	13	15	18
Rose	16	14	142	12	16
Sunflower	10	11	13	154	12
Tulip	17	16	16	10	141

Table 3. Results of SVM. Confusion matrix is used, where the row represents the actual class and the column represents the predicted class.

These images are not high resolution and are not reduced to a single size. Besides, the data collection is based on scraped data from flickr, google images, and yandex images.

3.2. Results

3.2.1 SVM

Confusion matrix is a common expression in the field of pattern recognition. It describes the relationship between the real attributes of sample data and the type of recognition results, which is a common method to evaluate the performance of classifier. The row of confusion matrix stands for real class of the samples. In our experiments, columns represent the predict class. The samples on the diagonal line of the matrix means their predicted classes coincide with their actual class. We have five kinds of flowers such that 5×5 confusion matrix is gotten shown in Table. 3. For each kind of flower, 200 images are chosen to the test set. No color information is included in the input of SVM. Only HOG and GLCM features are extracted for the input.

In general, the accuracy of the SVM method is about 0.69.

3.3. CNN and Transfer Learning

For the experiments of CNN and transfer learning models, we split the dataset into training and validation sets by $3:1$ and watch the accuracy on the validation set after 50 epochs.

About the experimental results, the CNN model can achieve about 0.87 accuracy on the training set, and 0.80 accuracy on the validation set after 50 epochs. The accuracy curve on both training set and validation set is illustrated in Fig. 7.

For the transfer learning models, the accuracy curves on both training set and validation set are shown in Fig. 8. And

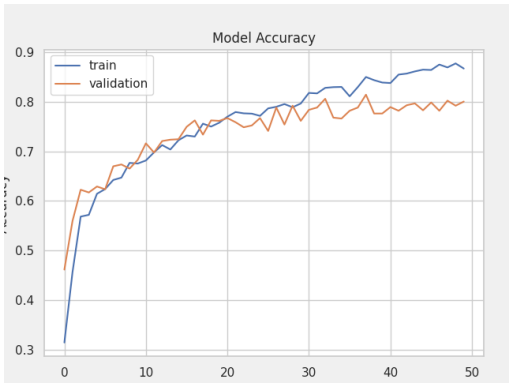

Figure 7. Accuracy curve of CNN model.

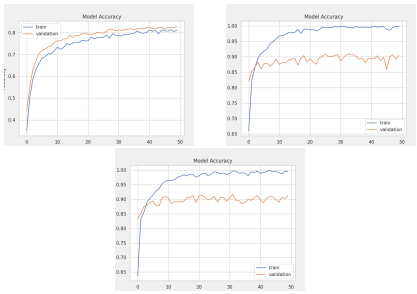

Figure 8. Accuracy curve of transfer learning models.

Model	Train Acc	Val Acc
Vgg16 no-top + our FC layer	0.82	0.84
Unfreeze last one block	0.99	0.91
Unfreeze last two blocks	0.99	0.92

Table 4. Accuracy of transfer learning models on training and validation sets.

the final accuracy on both training set and validation set after 50 epochs of transfer learning models are listed in Table. 4.

From the results above we can see that, only replace the top layer of VGG16 with our own and keep the remaining weights fixed can lead to an improvement on validation accuracy by about 0.04. Unfreezing last one or two blocks has no much difference on accuracy. However, unfreezing the last block(s) means that we train the model to learn the task-specific features for our problem. And this eventually improves the accuracy on validation set by almost 0.1 compared with the CNN model.

4. Conclusion

In this project, we apply both traditional and deep neural network method on the flower classification task.

For the traditional SVM method, the classification accuracy is not satisfying. For the low accuracy, there are serval possible reasons. First, there are no color information applied as classification input in the training process. Second, from the result, daisy and dandelion have Large classification errors. We should have more samples of those two kinds of flowers for training. Finally, in the training process, we just combine the two feature matrixs as input. Better classification results can be obtained by normalizing the feature matrix.

However, the CNN model can have a relatively high accuracy. What's more, with deep transfer learning, we achieve even much better performance than CNN model. This proves that transfer learning can really contribute a lot if used properly when lacking data.

References

[1] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[2] Kai-Bo Duan and S Sathiya Keerthi. Which is the best multiclass svm method? an empirical study. In *International workshop on multiple classifier systems*, pages 278–285. Springer, 2005.

[3] Robert M Haralick, Karthikeyan Shanmugam, and Its' Hak Dinstein. Textural features for image classification. *IEEE Transactions on systems, man, and cybernetics*, (6):610–621, 1973.

[4] Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. *IEEE transactions on Neural Networks*, 13(2):415–425, 2002.

[5] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[6] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[7] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[8] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[9] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[10] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[11] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

A. Workload Assignment

Task	Member(s)
Image data analysis	All
Feature extraction for SVM	HU Zongyang
SVM	XIAO Yuhang
CNN	LI Zehuan
Transfer learning using pretrained model	FENG Xuening
Result analysis	All
Report integration	FENG Xuening