Fruits identification

based on deep learning for smart shopping

Group Number:

Names:

***Abstract*—Self-check-out is commonly used in the supermarkets and stores in order to save time and manpower. However, for most vegetables and fruits without barcode, people must remember the item’s responding identifier when they conduct self-check-out. Deep learning models, especially convolutional neural networks (CNNs), are often used in image classification and computer vision problems. In this paper, we propose some deep learning models based on CNNs to automatically identify the different types of fruits. A classic CNN model built by us and two popular CNN models used for transfer learning are utilized in the proposed system. In addition, the 3 CNN models are combined with an ensemble learning method, averaging, to further improve the accuracy, and support vector machine (SVM) serves as a comparison method. The experimental results of using the proposed averaging method exhibits approximately 1% - 2% accuracy enhancement over the other considered machine learning algorithms. Since the accuracy of the proposed method is up to 98.8%, it is possible to apply it to real-life scenarios.**

***Keywords—Deep learning, SVM, CNN, image classification, Xception, smart shopping***

# Introduction

With the rapid development and wide application of artificial intelligence and machine learning models, much convenience has been brought to daily lives over the past decade in many aspects including object identification, intelligent transportation systems, entertainment and consumption recommendation. This paper focuses on the fruits identification which is defined as outputting the most probable type name of the fruit, given the input of the fruit image, using machine learning algorithms.

Self-checkout is ubiquitous in Canadian supermarkets, where people can scan the code bar of items by themselves. However, for most fruits without code bar, people have to record the digital code and input the code in the self-checkout area, which is inconvenient when people forget to record. Although some fruits can be found in the self-checkout system by searching their names, there are still a wide range of names with which people are not familiar. Besides, there are subcategories of some fruits that people do not know the exact name. If the fruits could be identified by the self-checkout system itself, this fine grained classification problem would be addressed, which motivates this study.

In this paper, the problem is addressed via three deep learning algorithms, specifically, CNN [1] models, including , a classical CNN built by us, VGG16 (Visual Geometry Group) [2], and Xception [3]. SVM [4] (Support Vector Machine) is used as a comparison method. For SVM, gradient features are extracted using HOG [5] (Histogram of Oriented Gradients) and PCA [6] (Principle of Components Analysis) is used to reduce the dimension. Besides, the multi-class classification problem is addressed by one-versus-all SVM. In terms of deep learning algorithms, a classical CNN model built by us was trained and evaluated. When it comes to VGG16 and Xception (two transfer learning models), the pre-trained models on ImageNet database are transferred to the fruit datasets. The accuracy of four methods is compared after obtaining the optimal models respectively. Besides, the three deep learning algorithms are combined using the averaging method in order to further improve the classification accuracy.

This paper is organized as follows: Section II presents the background including the utilized algorithms and accuracy measures. Section III reviews a few related works. Section IV and V introduces the methodologies of using SVM (Support Vector Machine) and deep learning algorithms in detail respectively. Section VI provides the results of using all methods and discusses their performance comparison. Section VII summarizes the achievement of this study.

# Background

## HOG, PCA and SVM

The HOG is a feature descriptor of image based on gradient direction, which was firstly proposed in [5]. The basic idea is to characterize the shape of objects as a directional distribution of edges by calculating the orientations of image gradients and their histograms [7]. Generally, this algorithm consists of two steps. First, the color image is converted into grayscale image and normalized by using Gamma correction. Secondly, the gradient of image in horizontal and vertical direction are calculated by using the edge operator [8].

|  |  |
| --- | --- |
|  | (1) |

where is the pixel value, and and present gradients at the vertical and horizontal directions of pixel .

PCA is a multivariate statistical analysis algorithm [6]. In this paper, it is used to reduce the dimensions of the HOG feature set with a compact feature representation. Suppose there are training samples and the dimensions of image is . Transforming the image into one-dimensional column vector, the training sample set are . The mean and covariance matrix  of can be calculated as

|  |  |
| --- | --- |
|  | (2) |

The main extract direction of PCA is the eigenvectors of (The size of is ), while the number of principle components of PCA is corresponding to the largest eigenvectors.

Support Vector Machine [4] [9] is a supervised learning model with the aim of determining the optimal hyperplane for binary classes. The principle of SVM is to map the points from low-dimensional into high-dimensional space, making them linearly separable so that the classification boundary can be judged by linear division.

The classification plane is expressed as. Note that is a multidimensional vector and represents the calculation of the dot product of two vectors. The reciprocal of the sorting interval is , so the optimization problem is expressed as:

(3)

where is the absolute value of a vector. The constraint in (3) is that the distance between each data point (*,*) and the classification plane is greater than or equal to one. Among them, is the classification of data.

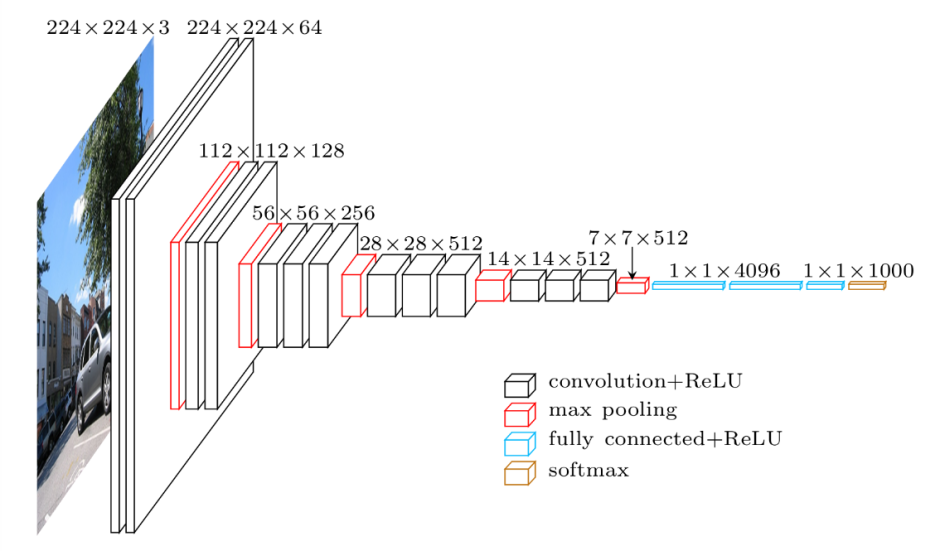
For multi-class classification problem, one-versus-all is one of the solution. The basic principle is to establish a few SVMs, each of them separates one class from all the other classes. The one with largest confidence level is selected as the final classification result.

## CNN, transfer learning, VGG and Xception

CNN is a kind of neural network structure that is widely used in object recognition because the images can be directly input to the network, avoiding feature extraction and data reconstruction processes in traditional recognition algorithm [1]. It generally works by two stages, image extraction using image convolution and image classification using neural networks [10]. The typical structure of CNN layers includes convolutional layers, pooling layers and fully-connected layers. In convolutional layers, the original signal features can be enhanced by convolution operation, while in pooling layers, the amount of data processing can be reduced without losing useful information by subsampling image to do local correlation. At last, the fully-connected layers can connect all the features and send the output values to the classifier.

Transfer learning is a technology that extracts useful information from data in a related domain and transferring them for being used in target tasks [11]. When images data (with width, height, and color channels) are involved in processing, large number of iterations or epochs are required during the training process of a neural network, which is computationally consuming. To address this problem, transfer learning is used, in which a pre-trained network on a very large dataset, such as Resnet, VGG16, Xception, and Inception-v3 can be used as a feature extractor by keeping all the pre-trained layers except the last fully connected layers [12]. In this paper, two pre-trained model called VGG16 and Xception are considered.

VGG16 is a deep CNN with 16 weighted layers, in which 224\*224 RGB images are passed through 5 blocks of convolutional layers with increasing number of 3\*3 filters. Besides, adjacent blocks are connected by a max-pooling layer. In addition, the 5 blocks of convolutional layers are followed by three fully-connected layers. Finally, the last layer is a softmax layer [2]. The structure is shown in Fig. 1.

Figure 1. VGG16 structure

Xception is a CNN architecture based on depthwise separable convolution layers with residual connections, which is under the hypothesis that the mapping of cross-channels correlations and spatial correlations in the feature maps of convolutional neural networks can be entirely decoupled [3]. It has 36 convolutional layers as the feature extraction base, which are structured into 14 models, with linear residual connections around them except the first and the last modules.

## Accuracy measures

In this study, hold-out validation is used to evaluate the models instead of cross-validation. The reason is that it would result in massive training time if cross-validation is performed on the considerable data set that we have used. For hold-out validation, 10% data is split from original data as test data, which would only be used after all optimal models of various algorithms determined using the training data.

The accuracy is measured by the ratio of the number of correctly classified sample images over the total number of test sample images, which is between 0 and 1, it can be denoted by,

|  |  |
| --- | --- |
|  | (4) |

# Related work

In [13], the authors used some machine learning algorithms to develop a fruit classification approach, they used scale invariant feature transform (SIFT) to extract the features from the images and SVM to classify the images, but its accuracy was lower than using random forests (RF). In [1], the proposed CNN method showed high accuracy (over 99%) when processing their image sets, and the authors proposed a voting method to select the best result of all the regions in an image, which could slightly increase the accuracy. and in [2], VGG was used to classify the images and got good results. In [14] and [15], the authors also used deep transfer learning, including VGG, Inception-V3 and Xception to identify the CT lung images and flowers, which showed good accuracy.

In our approach, first, we combined HOG, PCA and SVM to obtain a comparison method. Next, we built our own CNN model to suit the proposed problem, and compared it with popular CNN models, VGG16 and Xception. The main difference is that we propose an averaging method which combines the 3 DL models and develop a model which has a higher accuracy than the other approaches.

# Methodology of using SVM

## Data pre-processing

The data set consists of three public data sets. The main images of our data set is from fruit-360 [16], which consists of approximately 90% of the whole images. On the other hand, fruit-360 only has simple images which have pure white background, as shown in Fig. 2. Therefore, another two data sets, the whole data from [17] and part of the data in [18], were added into our data set to increase the accuracy to recognize fruit images in real-life senarios. After merging all the images from 3 data sets, there are about 55,000 images in total. For the machine learning algorithm, we divided the image set into training set and tested set with the ratio of 10 to 1, which consist of 50,000 images and 5,000 images, respectively. The test images were automatically randomly selected from the original image set by the computer.

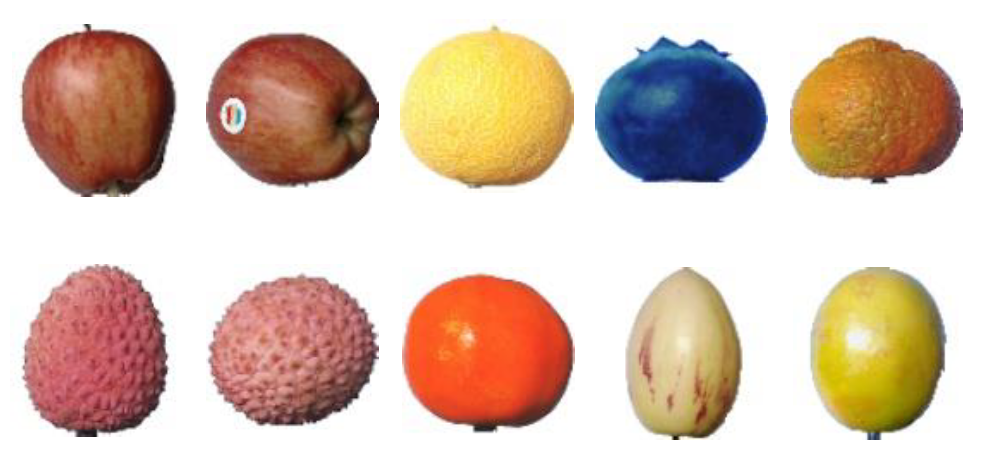


Figure 2. Sample images in the fruit-360 database

For the data pre-processing required for HOG algorithm, all the color images are then converted into grayscale images. In a color image, the color of each pixel is determined by 3 color channels, also called color components. They are R, G, B, which correspond to red, green and blue color channels, and the value of each color channel is in the range of 0-255. The grayscale image is a special image consisting of pixels whose R, G, B components are converted to one grayscale values by:

|  |  |
| --- | --- |
|  | (5) |

This equation is proposed based on the brightness of each color component in a pixel, and the brightness value is used to represent the grayscale value. After that, the variation range of each pixel reduces from about 256\*256\*256 to 256, which improves the computation efficiency to a large degree.

Next, all the images were resized to 224\*224 images, which is required for HOG feature extraction. After that, the images could have a same size of features. Since the output of an image would be a vector which indicates the probabilities of belonging to each category, normalization was done by dividing the pixel values of each image by 255, so that the range of all the pixel values is between 0 and 1, which can also accelerate the convergence speed of training.

Since the proposed program is a multi-class classification problem and each image has a label that indicates its category, then one-hot encoding is used for the output labels. By this normalization, it would be easier for the classifiers to process the data and give an output.

## Feature engineering

After the procedures in Section IV-A, the images were converted to gray scale images and normalized. In the next step, we used HOG algorithm for feature extraction. The block size of HOG model is set to contain 2\*2 cells and each cell consists of 20\*20 pixels. The bin size, which represents the number of gradient directions, is set to 9. Afterwards, features were extracted for each image, which is a large amount. In order to reduce computing time and improve the generalization capability of the model, PCA was used for dimensionality reduction. The number of principal components is set to 200, so the final image set consists of 200 features.

## Apply SVM

In the next step, the training set consisting of 50,000 images was trained by SVM algorithm. The input data are the image data with 200 features and the output is a label which indicates the classification result of an image. Linear kernel and Radial Basis Function (RBF) kernel functions were tested in the model and RBF shows better performance since the data in the proposed data set are not linearly separable. The training time was about 133 seconds using a 2.5 GHz Core i7 processor with 8GB RAM. A multi-classifier is generated after training.

## Evaluation

To evaluate the performance of the SVM classifier, we use hold-out validation since the test images were randomly generated from the original data set and cross-validation would have a similar accuracy and increase training time. All the 5,000 images from the untouched test set were tested by the classifier and the accuracy were printed to show its performance.

# Methodology of deep learning

The procedures of deep learning are different from the steps of machine learning, the main one is that deep learning can automatically extract features, so it does not need feature extraction or dimension reduction. This is because in CNN models, the convolutional layers serve as the filer of features.

## Data-preprocessing

The data-preprocessing process of deep learning algorithm is also different from the process described in section IV-A. In the training process, the model needs to calculate and compare the training loss and test loss and both the training set and the test set need to be used, another set which consists of randomly selected 5,000 images, called the validation set, was separated from the training set and would not be processed before the final model evaluation. Thus, there is a training set of about 45,000 images, a test set of 5,000 images and a validation set of 5,000 images.

In the next step, unlike using machine learning algorithms which need transformation to grayscale images and feature extraction, normalization is only needed, and the image set would be ready for training. In the normalization procedure, all the images were resized to 224 by224 pixels, which are commonly used for low or medium resolution images. This is because in deep learning models, especially fully-connected layers, the input size should be a fixed value, otherwise the parameters in these layers would dynamically change and the training would fail. Similar to the normalization presented in section IV-A, all the images were also normalized to be in the range of 0 to1.

## Apply a self-built CNN

Training CNN is a process of reducing the differences between output estimation results and ground truth labels by finding better kernels in convolutional layers and better weights in fully connected layers [19]. Based on the basic CNN models, we developed a simple CNN model, its structure is shown in Fig. 3. The first layer is the input layer, and the second part of the model, layers 2 to 10, are combinations of convolutional layers and max-pooling layers. The 11th layer is a global average pooling layer which can is to calculate the mean value of the whole feature map, and it aims to reduce the chance of over-fitting. After that, there is fully-connected layer to integrate the features and a drop-out layer to avoid over-fitting. A fully connected layer serves as a final layer to get outputs.

Regarding the parameters in this CNN model, the activation function of all the convolutional and pooling layers was selected to be ReLU (Rectified Linear Unit) which has a higher speed and is beneficial to avoid over-fitting, and the activation function of the last fully-connected layer was selected to be softmax to let the model give a output of a probability list for each category. The loss function was chosen to be categorical cross-entropy which is mostly suitable for the multi-class classification problems, and the optimizer was selected to be Adam. Besides, to save training time, early stopping was conducted and the parameter “patience”, indicating how many unchanged epochs should be wait for, were tuned in our program.

## Apply VGG16 and Xception

To apply transfer learning, we mainly tested two models in our project, VGG16 and Xception. These two models were imported from the Keras library and the weights were chosen to be the weights in pre-trained models of processing ImageNet which is a large visual database designed for object recognition research. There are 19 pre-trained layers in VGG16 model and 132 layers in Xception model in total. To apply the models in the proposed problem, the first 10 layers in VGG16 and first 100 layers in Xception were frozen, which means the weights of these layers were entirely transferred to our model. The other layers were set to be trainable. Similar to the simple CNN model proposed in Section V-B, the same 4 layers, a global average pooling layer, a fully connected layer, a drop-out layer and another fully connected layer were added as the last 4 layers to avoid overfitting and produce outputs.

## Combination and Validation process

After training and saving the models, there are 3 trained deep learning models in our system. To optimize the results, an averaging method is proposed to combine the 3 models, and its structure is shown in Fig. 4. Since the output of each DL model after processing an image is a vector which is a list of the probabilities of each category, the output vectors were added to calculate the average value of the probabilities for each category. The category with the largest final probability value is selected to be the final classification result.

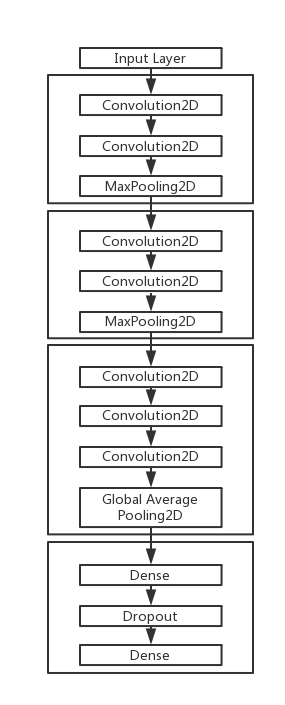


Figure 3. Classical CNN structure

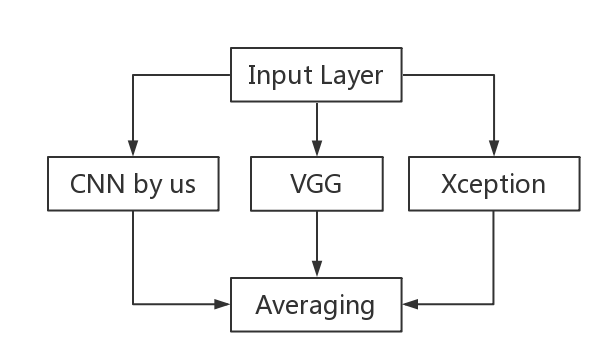


Figure 4. The structure of the combination method

For the validation process, as mentioned in Section V-A, the data set was split into a training set (about 45,000 images), a test set (5000 images) and a validation set (5000 images). The reason why there is a separate validation set is that during the training procedure, the test set would be processed many times in order to update better weights, we need an untouched data set to evaluate the performance of our models. Hold-out validation was selected because the images in the test and validation set were randomly selected from the original data set and hold-out validation is better to save training time than cross-validation.

# Results

To evaluate the proposed algorithms and models, many experiments were carried out. Since the proposed problem is a multi-class classification problem, the classification accuracy would be the most important accuracy measure. Moreover, the used data set is a relatively simple one, so top-1 accuracy is enough for evaluation. For a complex data set such as ImageNet database, top-5 accuracy would also be needed to evaluate the models. On the other hand, the training time of each model was also calculated to compare their efficiency.

The programming language and tools used to perform the experiments were Python and the Python libraries. To build the model, we used Sklearn [20] and Keras [21]. In addition, Numpy [22] was used to program all algorithms and Matplotlib [23] was used to plot the graphs. For training the deep learning models, Kaggle cloud was used because its GPU could save time.

All the models were evaluated by calculating their top-1 accuracy and training time, and several parameter were tuned in our program. Controlled experiments were conducted, which means when we tuned a parameter, all the other parameters were keep the same values - their default values. However, not all the parameters were tuned and not many values were selected for tuning due to time constraints, especially for DL algorithms which need a massive training time.

The results of using SVM algorithm is shown in Table I. It should be noted that the features of all the images were extracted by HOG algorithm and were selected by PCA algorithm, otherwise the accuracy would be much lower if we only use SVM. As mentioned in Section IV-B, the total number of features extracted by HOG of each image is 3,600 and PCA reduced this number by selected a number of features from them. Therefore, the number of selected features of PCA, also called principal components (PC) is the first parameter chosen to be tuned and the test values were selected to be 200 and 400. For the SVM algorithm, the parameter “kernel” was also tuned by us. Its value is set to ‘rbf’ or ‘linear’. The training time was calculated using a 2.5 GHz Core i7 CPU with 8GB RAM. The tuning results of each value can be seen in Table I. From the results, it can be seen that setting PC to be 200 would get higher accuracy and less training time than 400. For the kernel type, it seems that using linear kernel has better training accuracy than RBF kernel, but the testing accuracy is lower, which means that using linear kernel will result in overfitting to some degree, so RBF kernel is better. Finally, the values of parameters PC and kernel type were chosen to be ‘200’ and ‘rbf’, respectively, and the accuracy of the better model is 96.7%.

1. Results Accuracy of Using SVM.

| Tuning parameters | Training accuracy (%) | Testing accuracy (%) | Training time (s) |
| --- | --- | --- | --- |
| PC = ‘400’, kernel = ‘linear’ | 98.9 | 95.9 | 212 |
| PC = ‘200’, kernel = ‘linear’ | 98.7 | 96.4 | 89 |
| PC = ‘400’, kernel = ‘rbf’ | 95.9 | 96.0 | 592 |
| PC = ‘200’, kernel = ‘rbf’ | 96.7 | 96.7 | 192 |
| **Best. Accuracy or time** | **98.9** | **96.7** | **89** |

For deep learning models, the accuracy would be higher than SVM in the proposed problem, but the training time would increase to a large extent. Thus, DL models were running on Kaggle cloud and its NVIDIA Tesla K80 GPU was used. We also measured their top-1 accuracy and training time, but there is another type of accuracy for the validation data set and the unit of training time is set to minute.

For the simple CNN model, the main tuned parameters are batch size (32, 64 and 128), drop-out rate (0.5 and 0.25), optimizer type (‘Adam’ and ‘SGD’) and early stop patience (2 and 3). Since controlled experiments were conducted, when we tuned each parameters, all the other parameters were set to be their default values. The selected default parameter values are: batch size = 64, drop-out rate = 0.5, optimizer type = ‘Adam’, early stop patience = 2. The results of tuning the simple CNN model is shown in Table II. From the results in Table II, we can see that the importance of batch size is not much, and the result for Drop-out rate = 0.25 is similar to 0.5. Using SGD to be the optimizer is much harder to be convergent, it needs much more training time and has lower accuracy than using Adam. Besides, setting stop patience to be 3 has a larger change to show higher accuracy but the training time would increase much. To conclude, when batch size = 64, drop-out rate = 0.5, optimizer type = ‘Adam’, early stop patience = 3, the validation accuracy is the highest among the results, which is 97.9%. The graph of the model with the best accuracy is shown in Fig. 5, which shows the changing process of loss and accuracy.

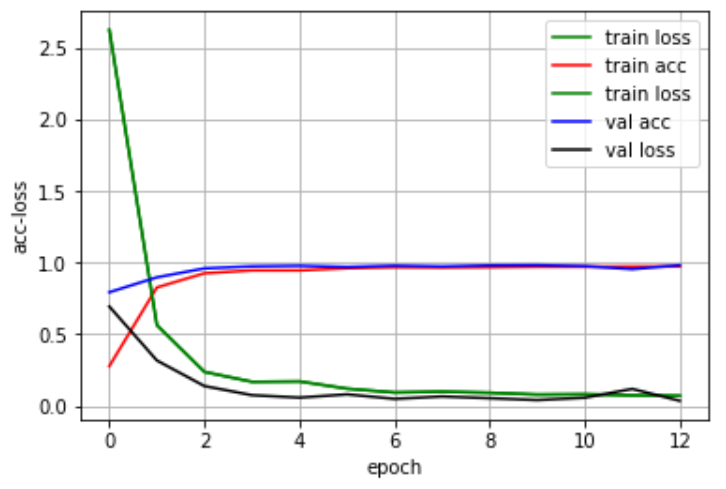


Figure 5. The changing process of CNN training

1. Results Accuracy of Using A Simple CNN.

| Tuning parameters | Training accuracy (%) | Testing accuracy (%) | Validation accuracy (%) | Training time (m) |
| --- | --- | --- | --- | --- |
| Default values | 95.7 | 97.7 | 97.3 | 112 |
| Batch size = 32 | 96.8 | 97.6 | 97.0 | 155 |
| Batch size = 128 | 95.8 | 96.7 | 96.1 | 100 |
| Drop-out rate = 0.25 | 96.9 | 97.4 | 97.2 | 140 |
| Optimizer type = ‘SGD’ | 91.4 | 94.9 | 95.9 | 239 |
| Early stop patience = 3 | 97.4 | 98.1 | 97.9 | 212 |
| **Best** | **97.4** | **98.1** | **97.9** | **100** |

For VGG16 and Xception models, similar to the simple CNN model, the batch size and early stop patience were also tuned by training them. In addition, the number of frozen layers were also tuned, which is important in transfer learning methods because it determines what percentage of weights will be transferred and what percentage need to be re-trained. The number of frozen layers of VGG16 is set to 5 or 10, and it set to 50 or 100 for xception. The results of the tuning process are shown in Table III and IV.  The default batch size, early stop patience, and learning rate of both models were set to 64, 2, 0.001, respectively, while the number of frozen layers of VGG16 and Xception was set to be 10 and 100, respectively.

1. Results Accuracy of Using VGG16

| Tuning parameters | Training accuracy (%) | Testing accuracy (%) | Validation accuracy (%) | Training time (m) |
| --- | --- | --- | --- | --- |
| Default values | 95.9 | 97.5 | 97.6 | 86 |
| Batch size = 32 | 95.0 | 96.7 | 96.4 | 87 |
| Batch size = 128 | 94.6 | 96.3 | 96.2 | 54 |
| Early stop patience = 3 | 96.9 | 97.4 | 97.5 | 193 |
| Frozen layers = 5 | 96.3 | 97.0 | 96.8 | 225 |
| **Best** | **96.9** | **97.5** | **97.6** | **54** |

1. Results Accuracy of Using Xception.

| Tuning parameters | Training accuracy (%) | Testing accuracy (%) | Validation accuracy (%) | Training time (m) |
| --- | --- | --- | --- | --- |
| Default values | 97.8 | 98.5 | 98.3 | 112 |
| Batch size = 32 | 97.5 | 98.2 | 98.2 | 128 |
| Batch size = 128 | 97.0 | 97.7 | 97.9 | 88 |
| Early stop patience = 3 | 98.0 | 98.6 | 98.2 | 216 |
| Frozen layers = 50 | 97.3 | 97.3 | 97.5 | 162 |
| **Best** | **97.8** | **98.6.** | **98.3** | **88** |

From the results in Table III and IV, the early stop patience makes very little differences and when the batch size was set to 64, the validation accuracy is the highest. Besides, when the number of frozen layers of VGG16 and Xception models was set to 10 and 100, their accuracy is better. The learning rate was also tuned from 0.001 to be 0.01, but the accuracy is near 0, which shows that the model could not be convergent if the learning rate is set to 0.01. The best validation accuracy of VGG16 and Xception among the experiments is 97.6% and 98.3%. The graph showing the changing process of the model with the best accuracy is shown in Fig. 6 and Fig. 7.

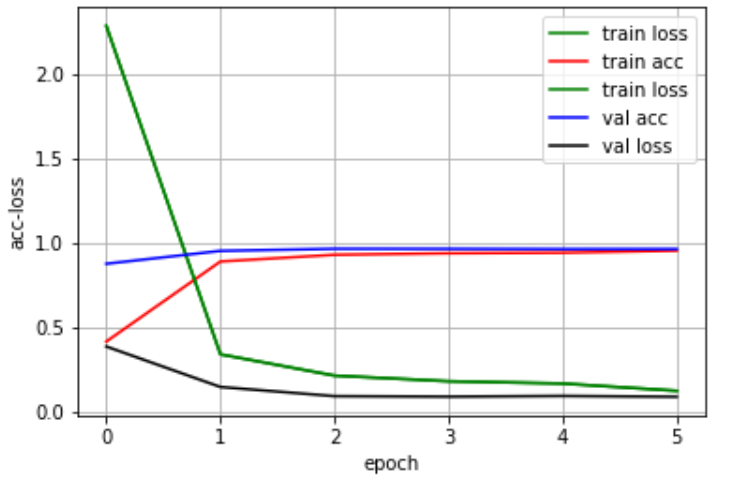


Figure 6. The changing process of VGG 16 training

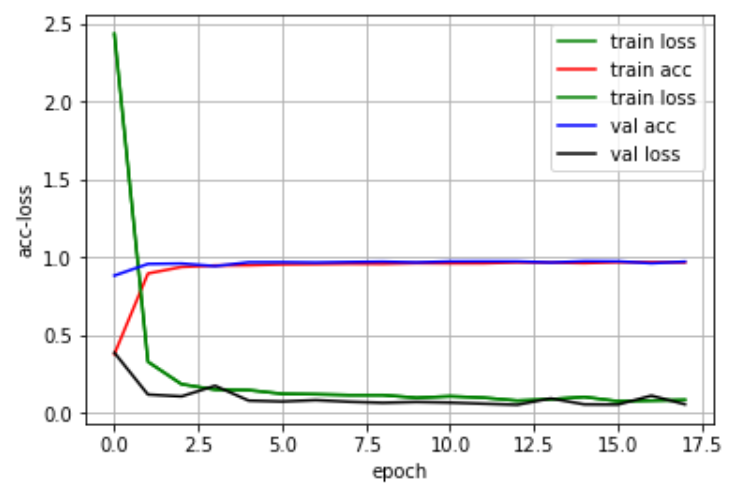


Figure 6. The changing process of Xception training

Finally, the accuracy of the combination method, averaging, which combines the results of testing the trained simple CNN, VGG16 and Xception models, is obtained to be 98.8%. The final comparison of all the results are shown in Table V. The averaging method has the highest accuracy among the proposed models because it combines the three algorithms and a few falsely classified results could be avoided and the accuracy would increase slightly.

1. Results Accuracy of All the proposed models

| Methods | Validation Accuracy |
| --- | --- |
| HOG+SVM | 96.7% |
| CNN by us | 97.9% |
| VGG16 | 97.6% |
| Xception | 98.3% |
| Averaging | 98.8% |

# Conclusion

This work presented a few models for smart shopping with fruit identification as its purpose. For self-check-out in supermarkets or stores, it still needs barcode to identify different types of goods, automatic fruit recognition enables the machine to identify self-weighing fruits. First, the proposed system utilizes SVM to classify the images with HOG to extract features and PCA to reduce dimension. After that, three deep learning models including a CNN model created by our own, VGG16 and Xception are trained to implement classification. An ensemble learning algorithm, averaging method is utilized to combine the three DL models and further improve the accuracy. The experimental results indicated that the combination approach, averaging, has a high accuracy of up to 98.8%, which is higher than the individual accuracy of the SVM and the other three CNN models. In our future work, we will explore and validate if the proposed model can still show good performance in a more complex datasets or other image classification problems.

##### References

1. L. Hou, Q. Wu, Q. Sun, H. Yang and P. Li, "Fruit recognition based on convolution neural network," 2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), Changsha, 2016, pp. 18-22.
2. M. S. Hossain, M. H. Al-Hammadi and G. Muhammad, "Automatic Fruits Classification Using Deep Learning for Industrial Applications," in IEEE Transactions on Industrial Informatics.
3. F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 1800-1807.
4. L. Yang, R. Muresan, A. Al-Dweik and L. Hadjileontiadis, "Image-Based Visibility Estimation Algorithm for Intelligent Transportation Systems," in IEEE Access.
5. N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 2005, pp. 886-893 vol. 1.
6. J. Jiang and H. Xiong, "Fast Pedestrian Detection Based on HOG-PCA and Gentle AdaBoost," 2012 International Conference on Computer Science and Service System, Nanjing, 2012, pp. 1819-1822.
7. S. Lee, M. Bang, K. Jung and K. Yi, "An efficient selection of HOG feature for SVM classification of vehicle," 2015 International Symposium on Consumer Electronics (ISCE), Madrid, 2015, pp. 1-2.
8. C. Yao, F. Wu, H. Chen, X. Hao and Y. Shen, "Traffic sign recognition using HOG-SVM and grid search," 2014 12th International Conference on Signal Processing (ICSP), Hangzhou, 2014, pp. 962-965.
9. C. Cortes and V. Vapnik, "Support-vector networks," in Machine Learning, vol. 20, no. 3, pp. 273-297, 1995.
10. A. Khumaidi, E. M. Yuniarno and M. H. Purnomo, "Welding defect classification based on convolution neural network (CNN) and Gaussian kernel," 2017 International Seminar on Intelligent Technology and Its Applications (ISITIA), Surabaya, 2017, pp. 261-265.
11. L. Shao, F. Zhu and X. Li, "Transfer Learning for Visual Categorization: A Survey," in IEEE Transactions on Neural Networks and Learning Systems, vol. 26, no. 5, pp. 1019-1034, May 2015.
12. I. Gogul and V. S. Kumar, "Flower species recognition system using convolution neural networks and transfer learning," 2017 Fourth International Conference on Signal Processing, Communication and Networking (ICSCN), Chennai, 2017, pp. 1-6.
13. H. M. Zawbaa, M. Hazman, M. Abbass and A. E. Hassanien, "Automatic fruit classification using random forest algorithm," 2014 14th International Conference on Hybrid Intelligent Systems, Kuwait, 2014, pp. 164-168.
14. I. Gogul and V. S. Kumar, "Flower species recognition system using convolution neural networks and transfer learning," 2017 Fourth International Conference on Signal Processing, Communication and Networking (ICSCN), Chennai, 2017, pp. 1-6.
15. R. V. M. d. Nóbrega, S. A. Peixoto, S. P. P. d. Silva and P. P. R. Filho, "Lung Nodule Classification via Deep Transfer Learning in CT Lung Images," 2018 IEEE 31st International Symposium on Computer-Based Medical Systems (CBMS), Karlstad, 2018, pp. 244-249.
16. M. Oltean, Fruits 360 dataset. Accessed: Aug. 19, 2018. [Online]. Available: <https://www.kaggle.com/moltean/fruits>
17. M. Buyukkinaci, Fruit Images for Object Detection. Accessed: Jul. 2018. [Online]. Available: <https://www.kaggle.com/mbkinaci/fruit-images-for-object-detection>.
18. I. Hussain,; Q. He; Z. Chen; W. Xie, Fruit Recognition dataset. Accessed: Jul. 12, 2018. [Online]. Available: <https://zenodo.org/record/1310165#.XAICOWhKg2x>.
19. R. Li et al., "DeepUNet: A Deep Fully Convolutional Network for Pixel-Level Sea-Land Segmentation," in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 11, no. 11, pp. 3954-3962, Nov. 2018.
20. F. Pedregosa et al., “Scikit-learn: Machine learning in python,” J. Mach. Learn. Res., vol. 12, pp. 2825–2830, Feb. 2012.
21. F. Chollet, Keras, 2015. [Online]. Available: https://github.com/fchollet/  
    keras.
22. NumPy—NumPy. Accessed: Jan. 9, 2018. [Online]. Available:  
    <http://www.numpy.org/>
23. J.D. Hunter, “Matplotlib: A 2D Graphics Environment,” Computing in Science & Eng., vol. 9, no. 3, 2007, pp. 90–95.