Improve Unsupervised Machine Learning Model on Fruits by Using VGG-16

Zhui Geng *

Faculty of Math, University of Waterloo, Waterloo, Canada

* Corresponding author: z8gengzhui@uwaterloo.ca

Abstract. The demand for efficient fruit classification has risen with supply chain complexities. For the sake of saving the labor force and improving efficiency, applying machine learning algorithms on fruit categorization to multiple stages in the factory is a feasible solution. The paper briefly delivers the rationale of algorithm VGG-16 and validates its advantage in accuracy by comparing it with other convolutional neural network models. Traditional convolutional network reaches its peak accuracy which is around 40% after 45 training epochs. VGG-16 results with a 97% percent accuracy on the classification of more than 20 kinds of fruits only after 6 epochs and can be improved further by enriching and augmenting the dataset. However, although VGG-16 only requires a small number of epochs, the weighted parameter for each layer increases dramatically and increases the running time. Future researchers should focus on optimizing the algorithm to make it more feasible in the industry. Also, remote computing might be a solution for large computational requirements.

Keywords: Machine Learning, Classification, Neural Networks, Computer Science, Statistics.

1. Introduction

With the increasing demand for fruit and the supply chain, manual classification and fruit inspection by eye are inefficient and prone to errors. Therefore, Automatic fruit classification is important. By using new technology, the industry can not only free more labor but also improve overall efficiency. Machine learning has been successfully used to solve a wide range of issues in a variety of application fields over the past few years. Examples of these include robotics, businesses, cybersecurity, virtual assistants, image recognition, healthcare, and many more [1]. Among these, the most effective approach for classification problems at the moment is the Deep Learning (DL) model, which has made numerous image-based object detection and recognition-based solutions possible in a variety of sectors. It would therefore be interesting to use this method on other problems [2].

A very large number of models are now used for graph classification, including Support Vector Machine, Naive Bayes Classifier, Logistic Regression, Random Forest Classifier, and so on [3, 4]. To implement the classification, after comparing multiple models, the Visual Geometry Group 16 (VGG16) structure is proven to be one of the best-performed models. It is recognizable as an improvement algorithm from the Convolutional Neural Network (CNN) [5]. In comparison to other models used in the same area, during training the model can achieve an accuracy close to.97 when belonging to the 36 recognized classes and an accuracy close to.96 belonging to the 46 recognized classes with acceptable computing power.

In that way, constructing a classification machine learning model for fruit identification is meaningful in lots of areas, including supply chain, quality control, efficiency improvement, and cost savings. Large datasets of a variety of fruits are used to train machine learning algorithms to identify complex patterns and features that are frequently invisible to the human eye. These models enable more precise sorting and grading because they can identify fruits not only by kind but also by ripeness, size, and flaws. The objective of this paper is to find a way to categorize fruits efficiently by comparing several models and datasets. In this paper, the VGG-16 algorithm is mainly used to recognize several types of fruit datasets and is proven to be one of the best models for classification problems. The research in this paper can provide a reference for other scholars in this area of research

2. Dataset and Method

2.1. Dataset and Methodology

2.1.1. A file preprocessing function

Several different training sets and test sets are used to improve the performance of the model in this study. The original (very first) dataset was from Kaggle called Grocery Store Dataset which contains a series of photos taken by hand in grocery stores. It takes advantage of a large number of classes which is considered crucial to the broadening of usage. The dataset contains 81 finely segmented categories, including fruits, vegetables, and packages. However, the downside of this dataset is also significant. There are 5125 images, which means only around 60 images are included in each category. Such a small number of images can't support such large numbers of classes (large models). Besides data in the original dataset includes lots of background noise. After several rounds of tests and optimization, this study decided to move the larger datasets.

The new dataset includes 36 classes, including fruits and vegetables, with 100 data for each category. There are 3600 images with pure color background, so that background noise will not be a problem which might influence the performance. Furthermore, every image in the new dataset has a high resolution with 2218 * 2216 pixels, which is a factor to improve the model.

The new dataset contains way fewer images than we want. The size of the training dataset is a crucial factor in the final performance, so emerging a new dataset with other related datasets is the most trivial way to do it. This paper uses the famous datasets, Fruits 360, and Fruits 262. After picking the same classes as our data, the files are renamed by a Python script.

2.1.2. A file preprocessing function

By using OS.py, the script will read the names of files in a folder and use a for loop to rename all files in that folder. After the harmonization of the names of the data, the study simply copies the chosen data to our new dataset. By doing so, more than 10,000 images are added to the dataset, which significantly increases the performance of the model.

2.2. The model of VGG-16

2.2.1. Structure of VGG-16

The full name of VGG-16 is Visual Geometry Group 16 layers,13 convolutional layers, and 3 fully connected layers. Besides 5 pool layers with no weight coefficient are also deployed. VGG-16 is one of the best performance CNN models up to date. The creator of this model used a very small convolution layer and pushed the depth to 16-19 weight layers. Such a model takes in 224 * 224 with 3 RGB channel images as input.

2.2.2. Features of VGG-16

There are several unique features of VGG-16. For example, VGG-16 does not use large amounts of hyperparameters. Instead, they focus on having convolutional layers of 3*3 filter with stride 1 and maxpool layer of 2*2 filter with stride 2. In this case, there are 13 convolutional layers, five maxpool layers, and three Dense layers (fully connected layers) [6]. Compared to previous classification algorithms such as AlexNet which have relatively large convolutional Kernal, in VGG-16, by replacing large convolutional kernel with multiple layers of small convolutional kernels, multiple nonlinear layers can increase the depth of the network to ensure the learning of more complex patterns, with still relatively small cost.

2.2.3. Number of parameters

Due to the multiple layers of small convolutional kernels, VGG is expected to have large numbers of weighting parameters. For example, for the first convolutional layer, the input image has 3 channels, the convolutional layer has a size of 3*3 filter, and each layer has 64 such convolutional kernel, the number of parameters for this layer for one image would be 3*3*3*64 = 1728. The second convolutional layer will have a study size of 3*64 and result a total number of 3*3*64*64 = 36864 parameters. For the following blocks, the number of kernels is doubled from 64 to 128, then to 256, until 512 is kept constant without further doubling. At the same time, the size of images is halved from 224 to 112, to 56, to 28, to 14, to 7. The detailed number of parameters for convolutional layers is shown in Table 1.

Table 1. The detailed number of parameters for convolutional layers

Block	CONV index	Study Size	Kernal Size	Parameters
1	1	3	64	$3 \times 3 \times 3 \times 64 = 1{,}728$
1	2	64	64	$3 \times 3 \times 64 \times 64 = 36{,}864$
2	1	64	128	$3 \times 3 \times 64 \times 128 = 73{,}728$
2	2	128	128	$3 \times 3 \times 128 \times 128 = 147{,}456$
3	1	128	256	$3 \times 3 \times 128 \times 256 = 294{,}912$
3	2	256	256	$3 \times 3 \times 256 \times 256 = 589{,}824$
3	3	256	256	$3 \times 3 \times 256 \times 256 = 589{,}824$
4	1	256	512	$3 \times 3 \times 256 \times 512 = 1{,}179{,}648$
4	2	512	512	$3 \times 3 \times 512 \times 512 = 2{,}359{,}296$
4	3	512	512	$3 \times 3 \times 512 \times 512 = 2{,}359{,}296$
5	1	512	512	$3 \times 3 \times 512 \times 512 = 2{,}359{,}296$
5	2	512	512	$3 \times 3 \times 512 \times 512 = 2{,}359{,}296$
5	3	512	512	$3 \times 3 \times 512 \times 512 = 2{,}359{,}296$

Furthermore, for three fully connected layers, parameters are given by:

$$\text{FC1:}\ \ 7 \times 7 \times 512 \times 4096 = 102{,}760{,}448 \tag{1}$$

$$\text{FC2:}\ \ 4096 \times 4096 = 16{,}777{,}216 \tag{2}$$

$$\text{FC3:}\ \ 4096 \times 1000 = 4{,}096{,}000 \tag{3}$$

From the table above, a total number of 138,357,544 weighted parameters could be observed, including in both convolutional layers and pooling layers. Because of that, it can be expected that the model has a high fitting ability which can crucially extract the features of images. Hence, it can handle the job of image classification well.

3. Results and Visualization

The Matplotlib module is used for visualization purposes. Build-in function “plot” is used to generate a line graph to present the relationship between the number of times the model was trained and the corresponding loss, as well as the currency.

Firstly, the traditional convolutional network is used to train the data. Three convolutional layers and three pooling layers are included in the model, with kernel sizes of 3*3 and 2*2 correspondingly. After training 6 times, test accuracy is only slightly above 0.6. The detailed accuracy is shown in Fig.1 and Fig.2. This unacceptable result inspires the study to move to another model

```
Epoch 36/45
83/83 [==============================] - 191s 2s/step - loss: 0.5866 - accuracy: 0.8030
Epoch 37/45
83/83 [==============================] - 180s 2s/step - loss: 0.5354 - accuracy: 0.8231
Epoch 38/45
83/83 [==============================] - 174s 2s/step - loss: 0.5285 - accuracy: 0.8136
Epoch 39/45
83/83 [==============================] - 171s 2s/step - loss: 0.4916 - accuracy: 0.8345
Epoch 40/45
83/83 [==============================] - 195s 2s/step - loss: 0.4946 - accuracy: 0.8307
Epoch 41/45
83/83 [==============================] - 171s 2s/step - loss: 0.5109 - accuracy: 0.8288
Epoch 42/45
83/83 [==============================] - 169s 2s/step - loss: 0.5573 - accuracy: 0.8129
Epoch 43/45
83/83 [==============================] - 169s 2s/step - loss: 0.4489 - accuracy: 0.8417
Epoch 44/45
83/83 [==============================] - 193s 2s/step - loss: 0.4759 - accuracy: 0.8356
Epoch 45/45
83/83 [==============================] - 171s 2s/step - loss: 0.4690 - accuracy: 0.8432
```

Figure 1. Training Outcomes

```
78/78 [==============================] - 37s 475ms/step - loss: 1.9250 - accuracy: 0.6008
```

Figure 2. Test Outcomes

Fig. 3 states the loss of the model concerning the training time. There was a spike in losses when the training got to about 5,000 training time. This corresponds to the low accuracy that occurs around 5000 training time in Fig.4. Indeed, the model is shown to have a significant reduction of loss while increasing accuracy up to 97 as opposed to the previous test CNN model with an accuracy of approximately 50% would be unable to achieve.

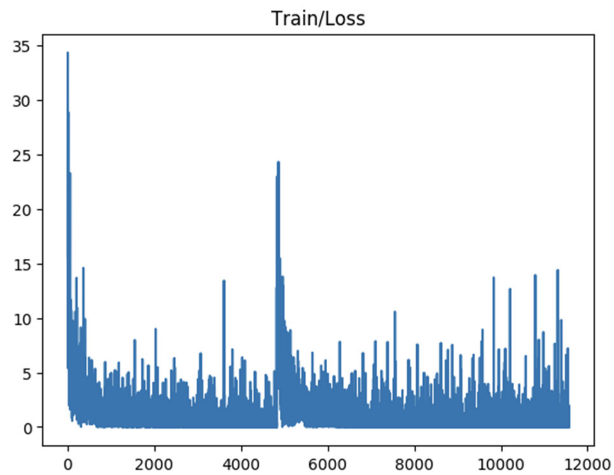

Figure 3. Train/Loss

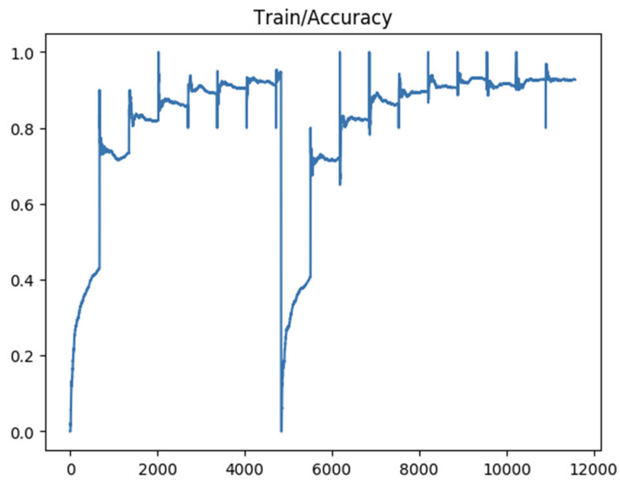

Figure 4. Train/Acc

After validating the feasibility of the model VGG-16, the last step to improve the accuracy will be to enrich the training dataset. By emerging more images from Fruits 360 and Fruits 262 with the original dataset, we get a dataset with more information. However, because the previous model is already accurate enough with an accuracy of 0.97, small improvements are expected for the new model with a larger training dataset. The result of the new model is provided in Fig.5 and Fig.6. An improvement of around 10 in loss and 0.01 in accuracy is observed.

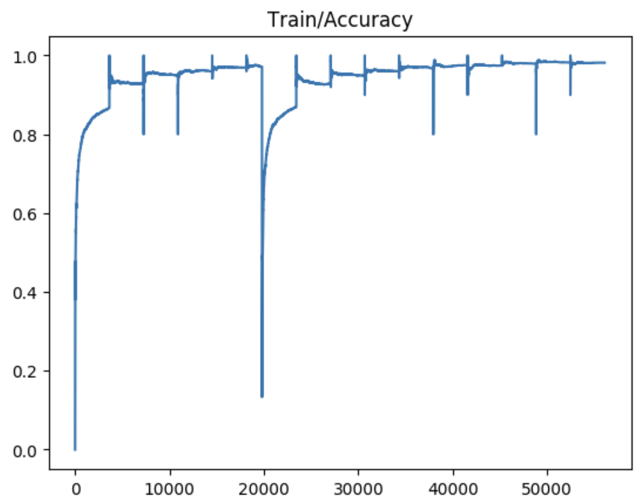

Figure 5. Train/Loss

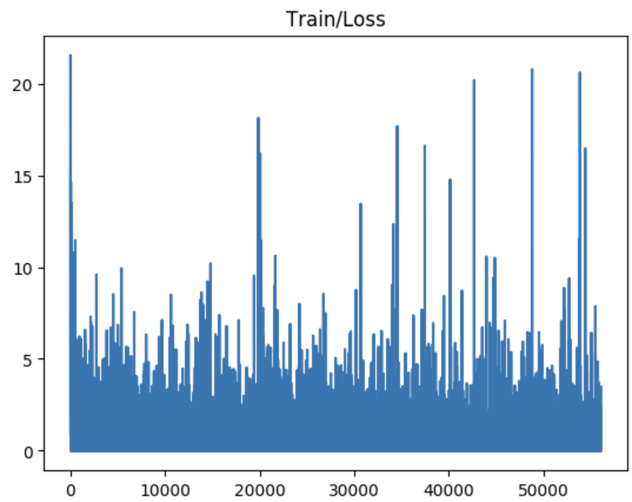

Figure 6. Train/Acc

4. Limitation and Improvement

As mentioned in the 2.2.3 "Number of Parameters", there exist large numbers of weighted parameters in the model, which directly cause large consumption of memory and slow training speed. In specific, as mentioned in 2.2.3, 138, 357, 544 weighted parameters are created in the model. Also, an image size of 224*224 with an RGB channel is used for training, the memory consumed is shown in Table2.

Table 2. Layer Size/Image

Layers	Size	Layers	Size
Input	$224 \times 224 \times 3 = 150K$	CONV4-2	$28 \times 28 \times 512 = 400K$
CONV1-1	$224 \times 224 \times 64 = 3.2M$	CONV4-3	$28 \times 28 \times 512 = 400K$
CONV1-1	$224 \times 224 \times 64 = 3.2M$	POOL4	$14 \times 14 \times 512 = 100K$
POOL1	$112 \times 112 \times 64 = 800K$	CONV5-1	$14 \times 14 \times 512 = 100K$
CONV2-1	$112 \times 112 \times 128 = 1.6M$	CONV5-2	$14 \times 14 \times 512 = 100K$
CONV2-2	$112 \times 112 \times 128 = 1.6M$	CONV5-3	$14 \times 14 \times 512 = 100K$
POOL2	$56 \times 56 \times 128 = 400K$	POOL5	$7 \times 7 \times 512 = 25K$
CONV3-1	$56 \times 56 \times 256 = 800K$	FC1	4096
CONV3-2	$56 \times 56 \times 256 = 800K$	FC2	4096
CONV3-3	$56 \times 56 \times 256 = 800K$	FC3	1000
POOL3	$28 \times 28 \times 256 = 200K$	TOTAL MEM	96MB/Image
CONV4-1	$28 \times 28 \times 512 = 400K$		

Each image consumes 96 MB of memory only for forward propagation, which means a total number of 196 MB of memory is going to be used for both forward and backward propagation. Due to the large number of weighted parameters and huge memory consumption, slow learning speed is expected. For example, by using a 2-core CPU, 8GB RAM, and no GPU boosting, training the model for this paper with 10 epochs costs 2 hours in total. However, that is still a tolerable speed. Although the model discussed in this paper is trained under extremely limited computing power, this problem can be easily solved by using stronger computing power.

5. Conclusion

The topic of this paper is justifying the performance of VGG-16 by comparing it to traditional convolutional neural networks and enhancing its accuracy by expanding and augmenting the dataset. After validating two models (VGG-16 and traditional CNN) with the same dataset, VGG-16 has significantly higher accuracy than traditional CNN. After training by traditional convolutional neural network forty-five times, accuracy could only reach 0.6 maximum. However, training in VGG-16 by 8 times reaches an accuracy of 0.96 in the test set, 0.3 higher than traditional CNN. Moreover, although the VGG-16 already had high accuracy, after augmenting and optimizing the dataset, the accuracy of VGG-16 increased by point one percent. The drawback of VGG-16 is still significant, including a large number of weighted parameters and a large size of convolutional layers.

The VGG-16 architecture is famous for its proficiency in image recognition. The high accuracy of the model has already demonstrated that. Such implementation in the agriculture industry can significantly reduce labor costs and improve efficiency not only on the supply side but also in delivery. Future research should aim to enrich the training dataset with more fruit types and images that contain a variety of features. Moreover, a more challenging improvement will be the requirements of large computational power. This is a crucial problem for deploying such algorithms into industry since it is impossible to deploy strong computers everywhere. One solution is to transfer data from the user to the computing center and then transfer the result back to the user. However, a large amount of data is usually required in industry. It is costly and difficult to transfer large amounts of data back and forth. So, the industry desperately demands a way to optimize the algorithm from the root cause. As a result, VGG-16 is a reliable algorithm for fruit classification algorithm and optimization is required for actual deployment in industries.

References

[1] Mohammad, M. T. Understanding of machine learning with deep learning: Architectures, workflow, applications and future directions. Computers, 2023, 12 (5), 91.

[2] Krug, S., & Hutschenreuther, T. A case study toward apple cultivar classification using deep learning. Agri Engineering, 2023, 5 (2), 814.

[3] Shankar, D. D., & Azhakath, A. S. Random embedded calibrated statistical blind steganalysis using cross validated support vector machine and support vector machine with particle swarm optimization. Scientific Reports (Nature Publisher Group), 2023, 13 (1).

[4] Awad, F. H., Hamad, M. M., & Alzubaidi, L. Robust classification and detection of big medical data using advanced parallel K-means clustering, YOLOv4, and logistic regression. Life, 2023, 13 (3).

[5] Simonyan, K., & Zisserman, A. Very deep convolutional networks for large-scale image recognition. Ithaca: Cornell University Library, arXiv.org. 2015, Retrieved from deep-convolutio nal-networ k s - l arge-scale/d ocview/2081521649/se-2.

[6] Syed, R. S., Qadri, S., Bibi, H., Syed Muhammad, W. S., Sharif, M. I., & Marinello, F. Comparing inception V3, VGG 16, VGG 19, CNN, and ResNet 50: A case study on early detection of a rice disease. Agronomy, 2023, 13 (6), 1633.