Final Report on Common Fruits Identification

Calvin Huang, Zhengbo Wang
calvinh1@vt.edu, zhengbo@vt.edu

December 9, 2020

1 Abstract

This project focuses on providing a deep learning solution for fruits identification using convolutional neural networks. The goal of this project is to be able to identify the correct fruit category of a given image and be able to provide its nutritional values. The significance of this project is to provide a quick and reliable way to find out the exact type of fruit people are looking for when grocery shopping. It can also have business values if it is applied to fruits classification in the market. To conduct the experiment, our group used the "Fruits 360" datasets from Kaggle. This dataset currently includes 90,380 images of 131 different types of fruits and vegetables. The dataset is then being divided into a training set and a test set. The training set contains 67,692 images and the test set contains 22,688 images. Our group uses the Keras library inside TensorFlow to apply the convolutional neural networks. The findings of this project vary as we try to improve our model. At first, we experienced the problem of overfitting where the training accuracy increases with the validation loss. After adjusting and improving our model, we are now able to achieve high accuracy and low validation loss for the test data. However, both models have less accuracy predicting real world fruit images.

2 Problem Statement

This problem we are focusing on for this project is identifying fruit types given an image of the fruit using machine learning algorithms and providing their nutrition facts. The reason we chose this topic is because we think that the ability to identify fruits can be really useful for both personal and business uses. In addition, another reason we chose this project is because there are many existing datasets about fruits that can be easily found on the internet. Some of the possible uses and impacts of this application include helping businesses categorizing fruits and identifying fruits' nutritional values quickly, which can be useful for people on diet. Another possible contribution is when buying fruits from grocery stores; Instead of scanning the barcode to check-out, one can simply show the fruit by the camera to check-out the product.

3 Results/Findings

3.1 Initial Result:

```
Model: "sequential_12"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_19 (Conv2D)           (None, 33, 33, 32)        896
_________________________________________________________________
max_pooling2d_11 (MaxPooling (None, 16, 16, 32)        0
_________________________________________________________________
conv2d_20 (Conv2D)           (None, 16, 16, 32)        9248
_________________________________________________________________
max_pooling2d_12 (MaxPooling (None, 8, 8, 32)          0
_________________________________________________________________
flatten_5 (Flatten)          (None, 2048)              0
_________________________________________________________________
dense_6 (Dense)              (None, 131)               268419
=================================================================
Total params: 278,563
Trainable params: 278,563
Non-trainable params: 0
_________________________________________________________________
None
```

Figure 1, Summary of the initial CNN model

3.2 Fitting the model with 10 epochs

Our results tend to achieve higher accuracy after each epoch. In addition, we have a low training loss. For validation accuracy, we see that it is mostly increasing after each epoch. However, we find that the validation loss is also increasing. After 10 epochs, we achieved an training accuracy of 0.9943, training loss of 0.1038, validation accuracy of 0.9379 and validation loss of 4.0207, which increased from initial validation loss of 1.5661.

3.3 Similar results are shown with 100 epochs

The loss and the accuracy of the training data look promising. However, validation loss is still increasing, reaching 38.7616 at the 67/100 epoch and the validation accuracy stays at around 0.94.

3.4 Prediction of initial model

```
Found 22688 images belonging to 131 classes.
Test loss: 46.66901397705078
Test accuracy: 0.9400564432144165
```

This model achieved high test accuracy and high loss on the test data.

We then tried to use the model we just trained to predict real fruit(not using images from the fruits 360). We took a picture of a watermelon bought from Kroger and the model predicted it as a cauliflower.

Then, we found an image of a watermelon on google, the model predicted it as a Melon Piel de Sapo. Then, we found an image of multiple apples on google, the model tells it is a kaki.

Lastly, we tried to make it easier for the model by selecting an image that looks similar to the training data of an apple. This time the model correctly tells it is an apple.

3.5 Initial Analysis

In summary, our initial model achieves high training and validation accuracy but also high validation loss. From the experiment, we can observe that even though our model has high training accuracy and relatively high validation accuracy, most of the predictions made by the model are incorrect. However, we can see that the answers made by the model are somewhat related to the original picture. For example, the model predicted watermelon as Melon Piel de Sapo. These two fruits are similar in their shapes and also have similar surface pattern. In addition, the model predicted apple as kaki. This is also reasonable because kaki has a similar shape as apple.

Moreover, by looking at the increasing validation loss and increasing training accuracy, we think that our model is overfitting the training data. We believe this is what mainly caused the wrong predictions. Next we will adjust the model to see whether we can improve it.

3.6 Result of the improved model

```
Model: "sequential_16"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_50 (Conv2D)           (None, 50, 50, 16)        448
_________________________________________________________________
max_pooling2d_50 (MaxPooling (None, 25, 25, 16)        0
_________________________________________________________________
conv2d_51 (Conv2D)           (None, 25, 25, 32)        4640
_________________________________________________________________
max_pooling2d_51 (MaxPooling (None, 12, 12, 32)        0
_________________________________________________________________
dropout_37 (Dropout)         (None, 12, 12, 32)        0
_________________________________________________________________
conv2d_52 (Conv2D)           (None, 12, 12, 64)        18496
_________________________________________________________________
max_pooling2d_52 (MaxPooling (None, 6, 6, 64)          0
_________________________________________________________________
dropout_38 (Dropout)         (None, 6, 6, 64)          0
_________________________________________________________________
flatten_16 (Flatten)         (None, 2304)              0
_________________________________________________________________
dropout_39 (Dropout)         (None, 2304)              0
_________________________________________________________________
dense_28 (Dense)             (None, 512)               1180160
_________________________________________________________________
dense_29 (Dense)             (None, 131)               67203
=================================================================
Total params: 1,270,947
Trainable params: 1,270,947
Non-trainable params: 0
_________________________________________________________________
None
```

In this improved model, we decided to add two more hidden layers with increasing filters and dropout layers in hope of solving the problem of overfitting. We have tried to experiment the model with other settings and this one comes out to have the best result overall. Therefore, as we trained the improved model, we achieved high accuracy and low validation loss. This time, the test loss and test accuracy all perform well with the test data from the dataset.

```
Test loss: 0.15518203377723694
Test accuracy: 0.9859396815299988
```

3.7 Predicting real world fruits

Even though the improved model achieved high accuracy and low loss in the test data on provided by “fruits 360”, the model still has low accuracy when predicting images of fruits taken by ourselves and found on the internet. However, the results are better than the initial model. This time the improved model correctly predicted watermelon and strawberry, and other predictions that are incorrect are mostly fruits in similar shapes or colors.

4 Prediction Format

Watermelon

Nutrition Facts	
Serving Size	100 g
Amount Per Serving	
Calories	30
	% Daily Value *
Total Fat 0.2g	0 %
Sodium 1mg	0 %
Total Carbohydrate 7.6g	3 %
Dietary Fiber 0.4g	1 %
Sugar 6.2g	
Protein 0.6g	1 %
Vitamin D 0.00mcg	0 %
Calcium 7.00mg	1 %
Iron 0.24mg	1 %
Potassium 112mg	2 %

* The % Daily Value (DV) tells you how much a nutrient in a serving of food contributes to a daily diet. 2,000 calories a day is used for general nutrition advice.

After predicting the fruit, the program will output the original image, the prediction, and the nutrition facts of that fruit. For the nutrition facts, since there are 131 different classes of fruits in this dataset, we are currently only supporting ten common fruits like apples, watermelon, orange, and etc. due to time constraint.

5 Analysis of low accuracy on real world predictions

We think that this difference in accuracy when testing with the test data from "fruits 360" and real fruit images is caused by how the dataset is collected. The author of this dataset took pictures of these fruits when they are rotating. We think that even though the dataset has around 100 images for a single type of fruits, but they are just the same fruit but from different angles. We think this might have caused our model to be less accurate when trying to predict fruits in real life because there are so many different variations while the test data from "fruits 360" are all pictures of fruits taken in similar fashion as the training set.

6 Possible improvements

Since the dataset is so large, in order to experiment with the model, we decided to shrink the original data images from (100, 100) pixels to (50, 50) pixels. This might have caused some variations in the results. In addition, due to the amount of time to train the model, we have limited the number of epochs to 80. If more computation power can be used, we could increase the number of epochs to see whether if it makes a difference to the result.

7 Conclusion

Overall, the final model we trained achieved high accuracy when predicting the test set from "fruits 360". However, it scores low when testing with real world images. We hypothesized that this difference in accuracy is caused by the training dataset as it only contains images of rotated fruits, all in similar fashion, whereas in real life fruits can be taken in various ways. As a result, we don't think this program is ready to be used in real life.

8 Reference

[1] Horea Muresan, Mihai Oltean, Fruit recognition from images using deep learning, Acta Univ. Sapientiae, Informatica Vol. 10, Issue 1, pp. 26-42, 2018.

[2] SuperDataScience Team, The Ultimate Guide to Convolutional Neural Networks (CNN), SuperDataScience Team, 2018.