Automatic Detection of Oranges Peel Based on the YOLOv5 Model

Yukun Ma *

RDFZ Chaoyang Branch School, Beijing, China

* Corresponding Author Email: mayukun@rdfzcygj.cn

Abstract. In recent years, the deep learning algorithms have been widely used in target detection with the development of artificial intelligence algorithms. This experiment tends to detect whether the oranges are totally peeled or not by using the YOLOv5 automatic detection system. By using this automatic detection, the factory can classify the peeled and unpeeled orange easily and successfully, hence improving the efficiency and quality of the production of orange products. To achieve this goal, this experiment uses the YOLOv5s.pt model for training the detection; for collecting the images, considering the real situation, this experiment collect the whole orange with peel, partly peeled, the whole orange without peel, and some cloves of peeled oranges; overall 1100 images were collected, 310 came from the real-life photos and other found online; the ratio of the number of training images and the validation images is 8:2; set up two classes which are “orange with peel” and “peeled orange”; Using the online tool to label the images for training; Using the PyTorch (1.12.1) as the deep learning library. The experimental result shows that the proposed method achieved 88.4% for the Mean Average Precision (mAP) value and 88.7% for the average recall rate. Besides, in the testing part, this result can successfully detect the “orange with peel” and the “peeled orange”.

Keywords: YOLOv5, Object Detection, Machine Learning, Orange.

1. Introduction

With the gradual maturity of convolutional neural network techniques, object detection has become a huge aspect of machine learning [1-2]. The YOLOv5 is one of the most popular programs in object detection. it provides several training models and an environment that is easy to set up.

Some studies relate the YOLOv5 model to fruits. Like Automatic Fruit Recognition Based on Attention YOLOv5 Model [3], Strawberry Maturity Recognition Algorithm Combining Dark Channel Enhancement and YOLOv5 [4] and Apple stem/calyx real-time recognition using YOLO-v5 algorithm for fruit automatic loading system [5]. These are all very good researches that combined fruit and YOLOv5 detection. However, in the previous literature, no research focused on fruit with peel, like oranges. So, this time, this experiment will focus on this area.

Orange is a fruit of various citrus species in the family Rutaceae (see list of plants known as orange); it primarily refers to Citrus sinensis [6], which is also known as sweet orange. The average weights of the oranges are 140g. Like all citrus fruits, oranges are protected externally by a thick crust, which makes them quite resistant to transport [7]. On the other hand, when eating it, people have to peel the orange peel first. In the Large-scale industrial mass production of orange products, like orange juice, Orange canned, the factory also needs to peel them. Although the production technology is improving, it is hard to make sure that every orange can be peeled totally, if the unpeeled orange is produced juice by machines, the quality of the juice and cans will decrease. Identifying whether the orange is peeled or not is very important for factories.

Actually, in real production, the selection of the unpeeled orange is done by humans. But this is not effective because it is hard to know how many oranges are unpeeled, which means a worker may be faced with many unpeeled oranges, so the worker needs to peel all of them. At the same time, another worker may be received a batch of oranges that are all totally peeled. So, this worker does not need so much time to finish his work, so less orange is available for the next step of processing since the worker that faced more unpeeled oranges and cannot give enough orange to the next process. That led to the problem that the factory cannot use the workers with high efficiency. To solve this

problem, the YOLOv5 model can be used to detect whether the orange is peeled completely or not, then divided the unpeeled orange among the workers on average.

2. The Method of the Detection

2.1. Image Acquisition & Management

This experiment needs to set up the data set for training the model. In this experiment, the first 310 photos of oranges are got from real-life photos and then got some photos on the internet by searching the keywords “peeled orange” and “orange” on the websites. In the end, this experiment collects 1100 RGB orange photos for training in total. Which is more than enough to get a proper model. Figure 1 shows some orange pictures in the collected dataset.

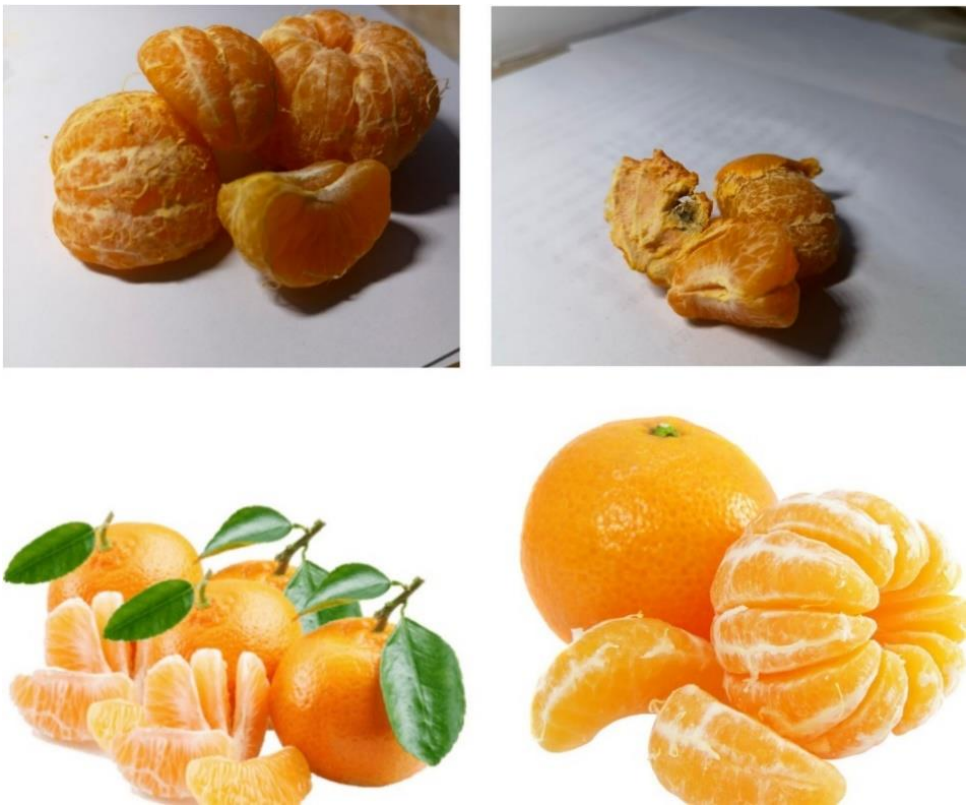

Figure 1. Some orange pictures in the collected dataset

After that, this experiment divided them into two categories which are “orange with peel” and “peeled orange”. This experiment used the website “https://www.makesense.ai/” to label the images. Since some pictures contain both “orange with peel” and “peeled orange”, this experiment has not divided them into two groups and then label them separately, instead, this experiment marks them together. In the labelling part, 0 represents “peeled orange”; 1 represents “orange with peel” in the output .txt file. On that website, you should set up the names and colors of the squares that represent different categories. Then label the object by drawing a rectangle to represent the part of the object that you want to detect and the corresponding categories of that rectangle. As the data sets output, the website itself will normalize all the files into the “.txt” format, then output all the files in a zip package.

2.2. YOLOv5 Model

This experiment uses a single-stage target detection technique called YOLOv5 algorithm, which is an improvement over previous versions of the You Only Look Once (YOLO) series [4]. Based on YOLOv4, YOLOv5 inherits the benefits and adds features to speed up detection without sacrificing accuracy. It is typically separated into four general modules, namely Input, Backbone, Neck, and Prediction. Figure 2 depicts the YOLOv5 network structure.

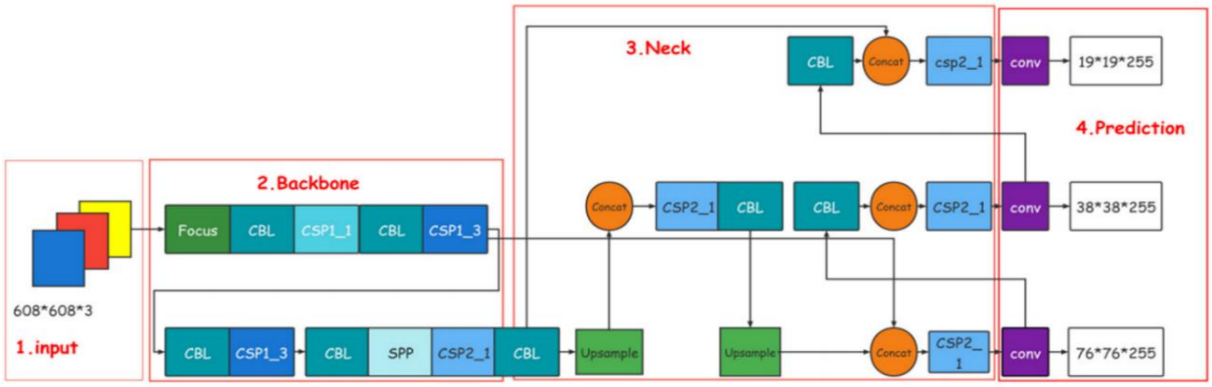

Figure 2. The network structure of YOLOv5.

YOLOv5 is subdivided into five versions, according to the network model from small to large, YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x. As the model become larger, the Mean Average Precision(mAP) value increases, which means the detection is more precise, however, the time is taken become longer and more parameters are used. The difference between the five different YOLOv5 network structures is shown in Table 1.

Table 1. Differences between five different YOLOv5 model

Model	Size (Pixels)	mAPval 0.5:0.95	mAPval 0.5	Speed CPU b1 (ms)	Speed V100 b1 (ms)	Speed V100 b32 (ms)	Params (M)
YOLOv5n	640	28.0	45.7	45	6.3	0.6	1.9
YOLOv5s	640	37.4	56.8	98	6.4	0.9	7.2
YOLOv5m	640	45.4	64.1	224	8.2	1.7	21.2
YOLOv5l	640	49.0	67.3	430	10.1	2.7	46.5
YOLOv5x	640	50.7	68.9	766	12.1	4.8	86.7

2.3. Setting Up the Environments

Anaconda 3 is used to set up the environments. Firstly, use the command “conda create -n YOLO” to create a new environment called YOLO, then “conda activate YOLO” to activate the environment. Secondly, the command on PyTorch official website can be found, and then add the PyTorch package into the environment. Apart from that, as the program is running, some new libraries still need to be added into the environment.

2.4. Image Resizing

In YOLOv5, there is another system that can resize the images. That is because the image is too big to identify.

Step 1: Calculate the resize ratio. The two ratios are formula (1) and formula (2). Then the smaller one is the resize ratio. E.g., resize an 800*700 picture, 640/800 = 0.8; 640/700 = 0.91, 0.8<0.91 so the resize ratio is 0.8

Step 2: Calculate the image size after resizing.

Step 3: Calculate the filling area. For the bigger one, need not fill black areas. But for the smaller side, first, find the difference between the two sides, then mod 32, divide 2, the result is the height (width) that is added to the image. (1) the formula is used to calculate the ratio of the lengths of the images before and after the resize process. (2) the formula is used to calculate the ratio of the widths of the images before and after the resize process.

$$\frac{the\ length\ of\ the\ image}{the\ length\ of\ the\ image\ after\ resize} \tag{1}$$

$$\frac{the\ width\ of\ the\ image}{the\ width\ of\ the\ image\ \ after\ resize} \tag{2}$$

2.5. Implementation details:

The implementation details of the experiment are shown in Table 2 and Table 3. These settings are used in the latter experiment.

Table 2. The settings of PyTorch

PyTorch Build	Stable (1.12.1)
Operating System	Windows 10
Package	Conda
Language	Python
Compute Platform	CPU

Table 3. The settings of the experiment

Name of the variable	Value of the variable
Batch size	1
epochs	100
Initial learning rate(lr0)	0.01
Final OneCycleLR learning rate(lrf)	0.2
CPU	11th Gen Inter(R) Core(TM) i7-11370H
CPU Clock Speed	3.3 GHz
Weights	YOLOv5s.pt
Deep learning library	PyTorch
Operating System	Windows 10

3. Results and Discussion

3.1. Describe the Result

By adjusting the values of batch size, epochs, Initial learning rate(lr0), Final OneCycleLR learning rate (lrf), CPU, and Weights, as Table 2 showed, and under the settings shown in Table 3, after 31.894 hours running by the CPU (11th Gen Inter(R) Core (TM) i7-11370H 3.3 GHz), the experiment got the following result, shown in the figure below.

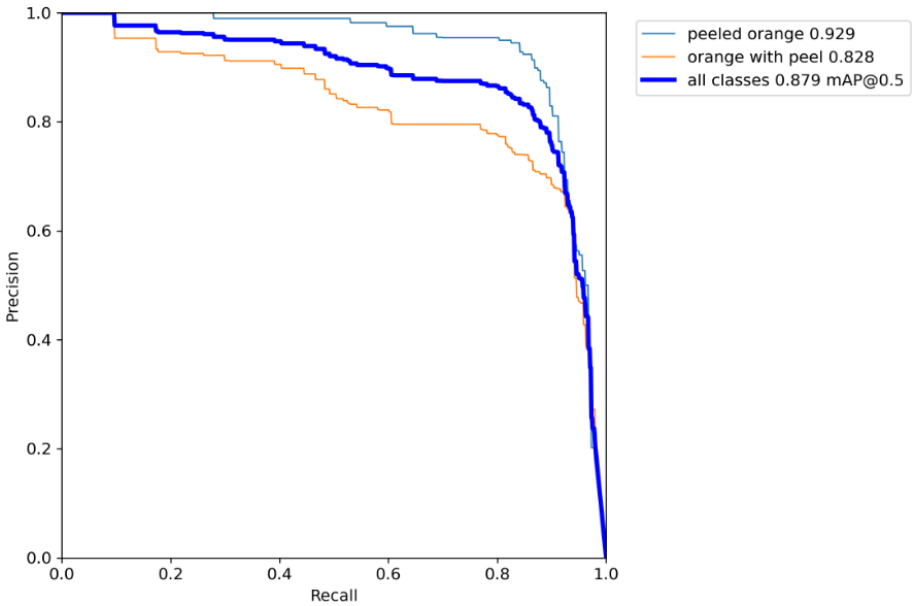

Figure 3. The Precision-Recall(P-R) curve of this experiment

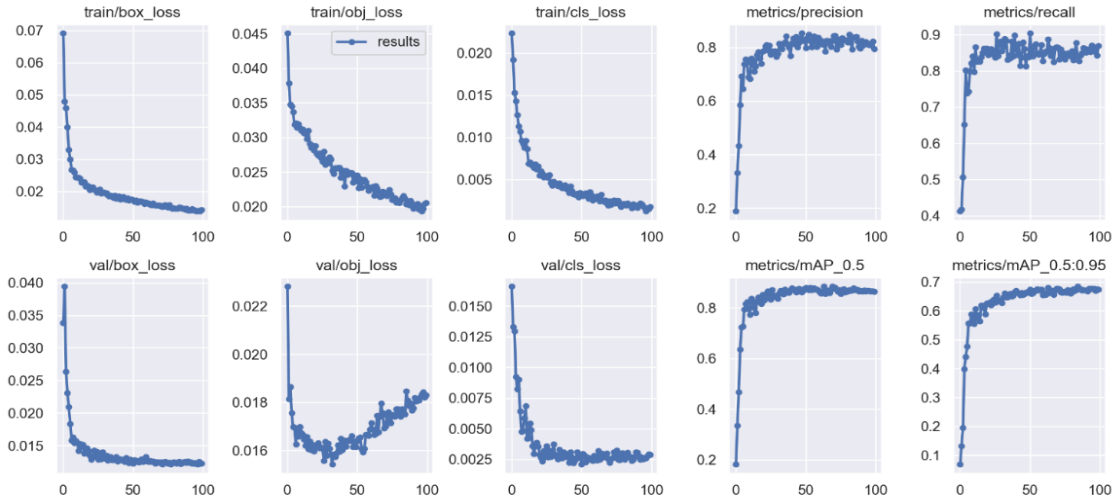

Figure 4. The data for 100 epochs

The Precision-Recall curve shows the relationship between the precision and recall. The area under the graph is called the Average Precision (AP). As the AP value is higher, the accuracy of the model is higher. So, in this graph, the peeled orange's P-R curve enclosed more area than the orange with peel's curve. That means the accuracy of this model to detect the peeled orange is higher than the detection of the unpeeled orange.

For Figure 4, there are ten graphs. For the first three graphs, these are the three loss functions' values for the training set. In these three graphs, the values decrease as more epochs are completed. That means more accurate results are trained. For the metrics/precision-epoch graph, the precision is higher as more epochs are completed. After 40 epochs, the value of precision varied around 0.825. The fifth graph is the metrics/ recall-epoch graph, with the same occasion of precision, a higher value with more epochs, however, the fluctuation of the recall graph is bigger than the precision one.

For the three graphs in the left bottom corner, there are three graphs that show the three loss functions' values of the validation set. For the box_loss and cls_loss, both of them decrease as the number of epochs finished increases. However, for the obj_loss, the lowest value is on the 32nd epoch, after that, the obj_loss value increase. That tends to avoid overfitting the model.

The line graph of metrics/mAP_0.5 shows that from the 1st epoch to the 12th epoch, the mAP value increases rapidly; from the 13th epoch to the 32nd epoch fluctuation with a little increase; after the 33rd epoch, the mAP value becomes more stable and varied between 86% and 88.4%. The best epoch is the 63rd epoch, with the mAP value of 88.4% and a precision of 81.26%. The details of the best epoch are shown in the table below.

Table 4. The details of the 63rd epoch (the best epoch)

Name of the variable	value
epoch	63
train/box_loss	0.01581
train/obj_loss	0.02302
train/cls_loss	0.00242
metrics/precision	81.26%
metrics/recall	88.70%
metrics/mAP_0.5	88.44%
metrics/mAP_0.5:0.95	68.00%
val/box_loss	0.01252
val/obj_loss	0.01670
val/cls_loss	0.00249
x/lr0	0.00386
x/lr1	0.00386
x/lr2	0.00386

3.2. Analysis of Experimental Results

In this experiment, there was an output model document called “best. pt”, this was the weight of the best epoch. Then, by setting the variables conf-thres as 0.5,iou-thres as 0.45, and max-det as 100, testing the output model by using the detect.py file. The detection result is shown in the figure below.

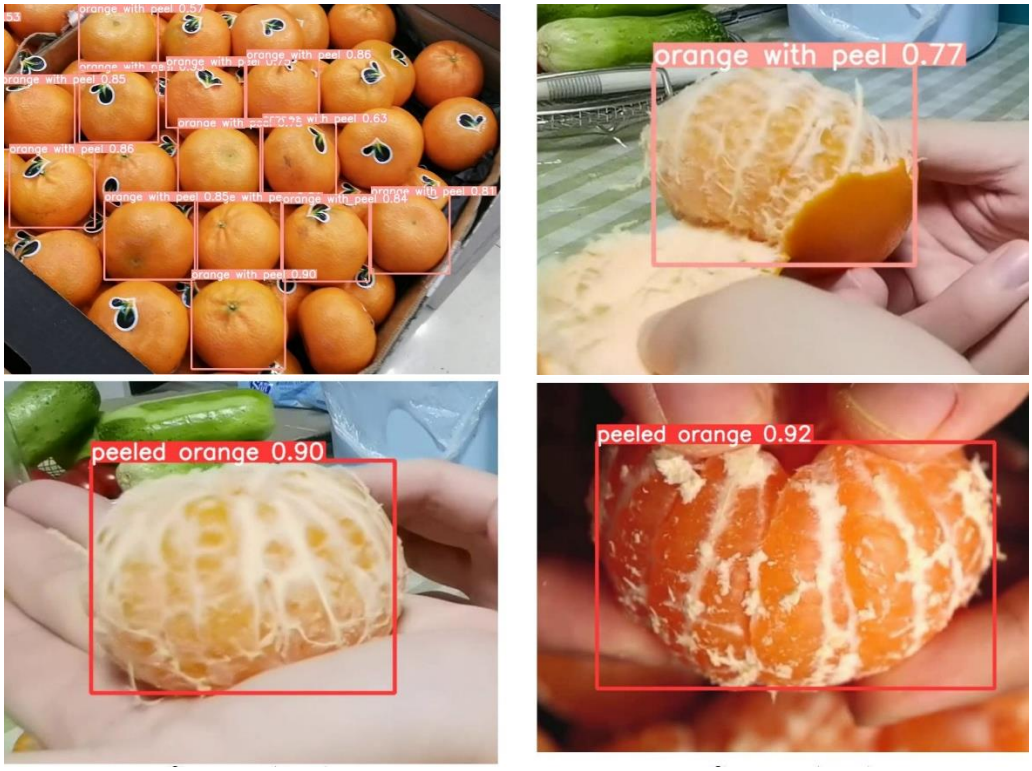

Figure 5. The detection results

In these detection results, the whole unpeeled orange can be detected successfully (Figure 5-1), and so do the “part peeled” orange (Figure 5-2). For the totally peeled oranges, no matter it is whole orange (Figure 5-3) or a clove of orange (Figure 5-4), both of them can be detected successfully. In terms of the confidence of the detection, for Figure 5-3 and Figure 5-4, both of them have confidence higher than 0.90, in Figure 5-1, most of them have confidence higher than 0.85.

The results demonstrated the excellent performance of the employed YOLOv5 model in this experiment. The main reason for it is due to the advanced architecture of the version of YOLOv5 and the convolutional neural network. In addition, the detection speed of YOLOv5 is also an outstanding point compared to the previous various series of YOLO. In this experiment, the detection of the orange has a promising speed. But sometimes, the accuracy may be degraded when there are many objects in one image as shown in Figure 5-1. More advanced models e.g., U-Net may be considered in the future to further improve the performance and carry out the semantic segmentation task [8-10].

4. Conclusions

In this work, the purpose was to detect the unpeeled orange and the totally peeled orange by using YOLOv5s.pt model. This experiment uses PyTorch as the deep learning library and uses Python 3.10 language. The experiment’s result shows that under 1100 images and the YOLOv5s.pt model can successfully train a new model that has the ability to detect whether this is an unpeeled orange or a totally peeled orange. By validation, the precision of the model is up to 88.4% which is very accurate. For the limitation and future plan part, firstly, after the detection of the unpeeled orange, how to use the AI or other system to allocate them averagely to the workers to speed up the producing process and make sure don’t redetect any orange. Apart from that, orange is a kind of fruit that has a big difference in “unpeeled” and “peeled” situations, so this becomes easier to detect. However, in future

studies, fruit like yellow peach which has a similar appearance to “unpeeled” and “peeled” can be tried.

References

[1] Zou Zhengxia, et al. Object detection in 20 years: A survey [R]. arXiv preprint arXiv:1905.05055 (2019).

[2] Amit Yali, Pedro Felzenszwalb, and Ross Girshick. Object detection [J]. Computer Vision: A Reference Guide (2020): 1-9.

[3] Cao Qiuyang, Shao Yeqin, Yin He. Automatic fruit recognition based on the Attention YOLOv5 model [J] (in Chinese). Computer Systems and Applications, 222,31(7):333-340

[4] Fan Youchen, et al. Strawberry maturity recognition algorithm combining dark channel enhancement and YOLOv5 [J]. Sensors 22, no. 2 (2022): 419.

[5] Wang Zhipeng, et al. Apple stem/calyx real-time recognition using YOLO-v5 algorithm for fruit automatic loading system [J]. Postharvest Biology and Technology 185 (2022): 111808.

[6] USDA. Plants Profile. 2011, https://web.archive.org/web/20110512023634/http://plants.usda.gov/java/profile?symbol=CISI3

[7] Wikifarmer, Orange Fruit Information.https://wikifarmer.com/orange-fruit-information/, 2022

[8] Qiu Yuhang, et al. Semantic segmentation of intracranial hemorrhages in head CT scans [C]. 2019 IEEE 10th International Conference on Software Engineering and Service Science (ICSESS). IEEE, 2019.

[9] Du Getao, et al. Medical image segmentation based on u-net: A review [J]. Journal of Imaging Science and Technology 64 (2020): 1-12.

[10] Alom Md Zahangir, et al. Recurrent residual U-Net for medical image segmentation [J]. Journal of Medical Imaging 6.1 (2019)