Machine Vision: Apple Counting in Orchards

Shuhao Kang, Yi Li, Yupeng Wang, Haoming Huang and Zhifei Yang

Department of Engineering Mathematics, University of Bristol

No.	Name of group members	Contribution to project	Contribution to report
1	Shuhao Kang	Data pre-processing of YOLOv3 and Faster R-CNN (100%) Implementation of YOLOv3 and Faster R-CNN (100%) Parameter adjustment (80%)	Section 5 (50%) Section 6 (50%)
2	Yi Li	A literature review of apple testing A literature review of YOLO and Faster R-CNN	Section 1 (80%) Section 2 (80%) Section 3 (80%) Section 7 (100%) Structure and formatting of the report (100%)
3	Zhifei Yang	Collection of relevant literature on orchard detection, summary of fruit detection Methods. YOLOV3 method description	Section 1 (20%) Section 2 (20%) Section 4.2 (100%)
4	Yupeng Wang	Search relevant works for Faster-RCNN （50%） Parameter adjustment (20%)	Section 5 (50%) Section 6 (50%)
5	Haoming Huang	Search relevant works of Faster R-CNN (50%)	Section 4.1(100%)

CONTENTS

1 Introduction

1.1 Background

The technology of automated agriculture is developing rapidly, taking advantage of existing machinery, computers, artificial intelligence, and other technologies such as monitoring crop fields (Gamaya, 2016), sowing (PlantTape, 2019), harvesting (CNH, 2019), and packing, however the current automation of agriculture is mainly used for the main crops. As a result, there is a lack of sufficient research in the field of fruit identification such as apples and cherries. In order to reduce the considerable labour costs of fruit picking and the hidden risks that come with it (Gongal, et al., 2015). For machine vision-based production prediction and automation it is of great significance.

1.2 Aim&Objectives

The aim of this experiment is to accurately detect and locate each apple in an orchard. With the development of object detection technology, deep learning has become the mainstream method (Zou, et al., 2019). Therefore, our goal is to use two different deep learning methods to count apples in photos of apple trees in orchards based on MinneApple, a benchmark dataset for apple detection and segmentation, (Häni, et al., 2019) , for yield mapping of apples in orchards.

1.3 Challenges

There are currently four challenges to this apple counting task, which are assumed to be differences between apples under different lighting conditions, variations in the appearance of different apples, the effect of fragmentation of apples due to obstacles such as leaves and variable camera views.

2 Related works

In the field of object detection, the deep learning method is a milestone. Before deep learning, object detection was done by manually extracting features. With the birth of RCNN (Zou, et al., 2019), object detection has entered the aera of deep learning. Deep learning methods are divided into two categories, RCNN (two-stage detection) and YOLO (one-stage detection).

The following briefly introduces the evolution path of these two types of deep learning and three steps to apple counting.

2.1 Fruit detection

The first step in apple counting task was the precise detection of all apples in the orchard. In the early stages of research, key point extraction and fruit detection were often applied to vineyards and orchards. For example, Nuske, et al. (2014) used circular Hough transforms to detect citrus fruits. In addition, Liu, et al. (2015) A has also used simple colour classifiers to classify feature points, such as grape bunches and peppers, and for fruit detection, image spots are extracted around each fruit to classify the image as either fruit or non-fruit. Research in recent years has focused on a generic approach to image segmentation, which consists of two feature learning algorithms: multiscale multilayered perceptrons (MLP) and convolutional neural networks (CNN). In research at the University of Sydney, Bargoti & Underwood (2017) has used watershed segmentation (WS) to detect the output of pixelated apples. This study improved the efficiency of segmentation between fruit individuals by adding metadata to the previous MLP benchmark. The final F1-score[1] for apple detection generated by the WS algorithm reached 0.861.

In contrast to these algorithms, two object detection algorithms have been implemented in this task.

2.1.1 Two-stage detector (R-CNN Family)

One of them is the Faster R-CNN algorithm, which is faster than earlier region convolutional neural networks (R-CNN) object detection algorithms. Faster R-CNN is a further optimized object detection algorithm based on R-CNN. In contrast to region convolutional neural networks, Faster R-CNN not only improves the detection speed but also the detection accuracy. In contrast, the Faster R-CNN convolves the whole image instead of each region proposal. Both classification and regression are trained together in the network (Ren, et al., 2016).

2.1.2 One-stage detector (YOLO Family)

The YOLO v3 visual frameworks were used in this Apple Count task. YOLOV3 (Redmon and Farhadi, 2017) uses multi-scale box prediction to detect large, medium, and small objects, respectively.

Since YOLO's detection pipeline is a single network, it allows for end-to-end optimization of detection performance. In theory, YOLO would outperform other detection algorithms, including DPM and Faster-CNN. (Redmon, et al., 2016)

2.1.3 Deep learning methods in fruit detection

In the field of fruit detection, the earliest article appeared in 2016. Sa, et al., (2016) used multi-mode Faster R-CNN model fusing data of color (RGB) and near infrared (NIR). Compared with the F1 score of the previous model[n+1], the score of this model improved from 0.807 to 0.838 in the detection of sweet pepper.

From 2016 to the end of 2020, deep learning has been widely used in fruit detection. The table1 below has listed the a few typical examples.

[1] The F1-score is a measure of the classification issue. It is the summed average of precision and recall, with a maximum value of 1 and a minimum value of 0. The greater the value of F1, the more precise it is. $F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$ (Chinchor, 1992).

Table 1: Deep learning methods for fruit detection between 2016 and 2020.

Author	Type of Fruit	Methodology	Metrics
Sa et al (2016)	Sweet pepper	Faster R-CNN	F1 (0.838)
Fuentes et al (2017)	Tomato	Faster R-CNN, R-FCN, SSD	mAP > 0.8
Chen et al (2017)	Apples and Oranges	Blob detector based on FCN	Ratio counted (0.968)
Dias et al (2018)	Apple flower	Pre-trained CNN sensitive for flowers	Precision and Recall (both>0.9)
Selvaraj et al (2019)	Banana	ResNet50 and InceptionV2 based models, SSD	Precision (>0.9)
Jiang et al (2019)	Apple leaf	INAR-SSD	mAP (78.80%) FPS (23.13)
Liu et al (2020)	Kiwi fruit	Faster R-CNN with data fusion	AP (90.5%) with fusion model

2.2 Fruit counting

There are many of the previous orchard counting approaches, including the classical circular Hough transform and watershed segmentation methods. However, each of these approaches has their own drawbacks and strengths. For example, Hough transform and Watershed alogrithm are two basci method for detecting geometric shapes which can be seen in the table below.

Table 2: The defect and strength of CHT and WA.

Detecting geometric shapes	**Defect**	Strength
Circular Hough Transform (Duda & Hart, 1972)	Insensitive to vestiges, noise, and other co-existing non-linear structures in the image.	excellent resistance to interference
Watershed Alogrithm (Beucher & Meyer, 1993)	positive response to weak pixels, noise in the image and small changes in the grey scale of the surface of the image	positive response to weak pixels and is a guarantee of closed continuous pixels.

2.3 Fruit tracking

The task of counting apples in an orchard requires not only the detection of each apple itself by the object detection algorithm, but also the tracking across frames to avoid incorrect double counting by the object detection system. In previous research, Wang, et al. (2013) to avoid double counting the number of apples, a threshold of 0.16m (approximately twice the diameter of the apple) is used, and the software calculates the distance from the apple detected on one side to the apple detected on the other. If the tolerance is less than this, the apple is recognized as an apple. However, there are some potential issues with this approach for the present apple counting task. For instance, if there are apples at a distance, and if there are multiple apples at a distance, the total diameter of the apples may also be less than the 0.16m threshold.

In a recent research, Roy, et al. (2018) merges the number of fruits recognized on both sides of an apple by means of semantic information. Since apples are usually identified on both sides of the tree, it is essential to merge the number of fruits on both sides.

Without this step of the analysis, it is crucial to trace the fruit trees. Even if the apples are detected accurately, this can result in errors in apple counts.

3 Data acquisition and datasets

3.1 Data acquisition

Bargoti & Underwood (2017) has used ground vehicles with monocular vision to capture high-resolution orchard image data, and the images for this task were obtained from a dataset provided by the University of Minnesota (Häni, et al., 2019). However, an orchard data set based on a ground vehicle with a monocular system is assumed in this paper. While the ground vehicle is moving slowly through the orchard, the monocular vision system will take pictures of the orchard trees in different orientations to obtain data about the orchard.

In this apple counting mission, the task team did not actually collect data on apples in the orchard. The dataset used in this experiment is a. PNG image from the University of Minnesota's dataset folder "train" in the "detection" folder. The MASK was used to produce the dataset in this application. The first step is to obtain the coordinates of the diagonal corners of each apple frame in each image through the mask and count the number of frames in each image, then convert the information about the classification (object) of each frame in each image - apples, the number of frames and the diagonal coordinates (xmin, xmax, ymin, ymax) of each frame - into an XML file according to the XML file format. The final information such as the image paths, coordinates of each frame and the classification of the frames are converted into a TXT file to complete the dataset for the training of the apple counting model.

3.2 Dataset

In the "train" dataset, there are multiple categories of apples and environments, for illustration, including red apples, green apples, and wide range of distant and close views of apples, and variations in light and shadow in the environment, which complicates and makes the detection much more challenging. However, in the mask, the distinction between green apples and the environment is satisfactory, with most of the apples being identified in most images, but a small number of clustered apples or smaller, more distant, and blurred apples are not well identified in the mask, resulting in some images with fewer than 10 apples identified by the mask. (Estimated by the human eye to be around about hundred), As a result the inaccuracy of training learning is increased.

4 Methodology

4.1 Faster R-CNN

4.1.1 The structure of Faster R-CNN

In general, Faster R-CNN which can been seen in the figure 4.1 below (Ren, et al., 2016). will resize a picture and transfer it to the backbone feature extraction network to obtain the feature maps, a collection of features. Use the Region Proposal Networks (RPN) to get the proposals. They are candidate prediction boxes. Then it puts the proposals and the feature maps into the ROI pooling layer for feature interception to obtain proposal feature maps of different sizes and resize them again to get the same size proposal feature maps. After that, bounding box regression and classification are performed on the proposal feature maps, thereby adjusting the proposals to obtain the final prediction boxes. These boxes are the results.

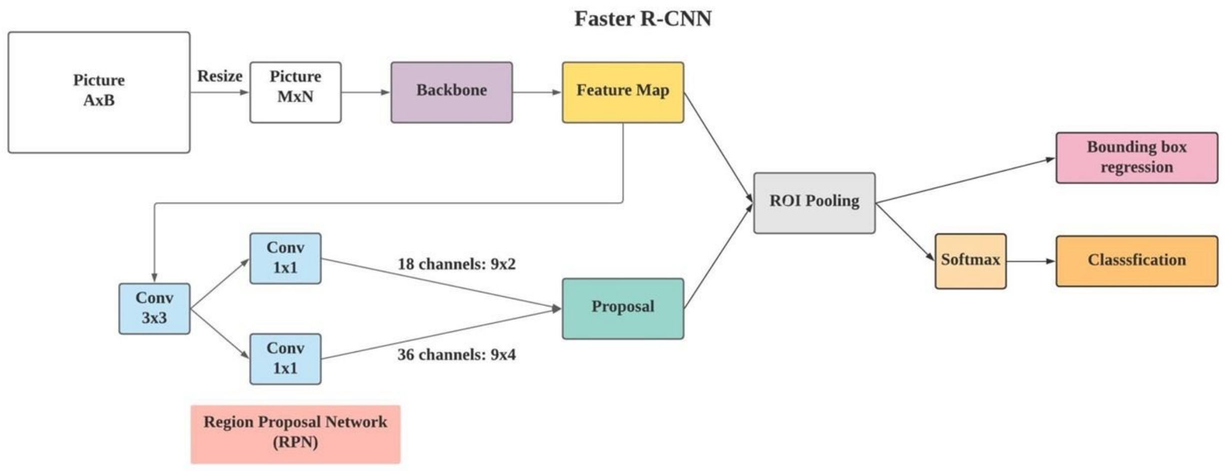

Figure 4.1: The framework of the Faster R-CNN

4.1.2 The Backbone of Faster R-CNN

Faster R-CNN can use a variety of backbone feature extraction networks. The figure 4.2 shows the backbone of Faster R-CNN. The commonly used ones are VGG and Resnet. Given that Resnet 50 is very common and Resnet solves the degradation problem, which represents that as the depth of the network increases, the performance of the network first gradually increases to saturation, and then rapidly decreases, of deep networks through residual learning, the effect of training is better than VGG network training (He, et al., 2016). Thus, this experiment uses Resnet 50.

The Resnet 50 has two basic blocks, Conv Block, and Identity Block, which can be seen in the figrue 4.2 below.

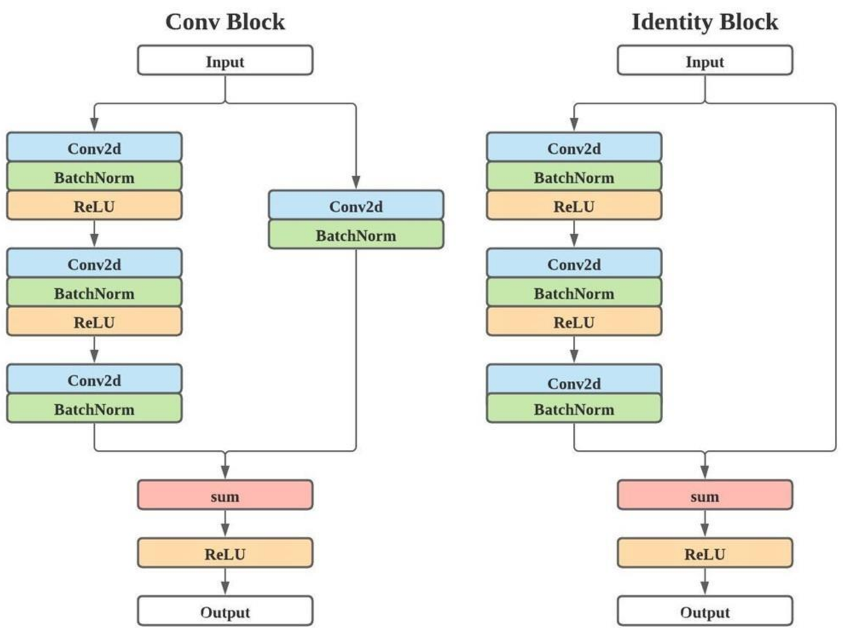

Figure 4.2: The backbone of Faster R-CNN

Figure 4.3 shows a sample of resnet 50. The Resnet 50 can be regarded as a synthesis of several Conv Blocks and Identity Blocks. Its structure is shown in the figure 4.3 below.

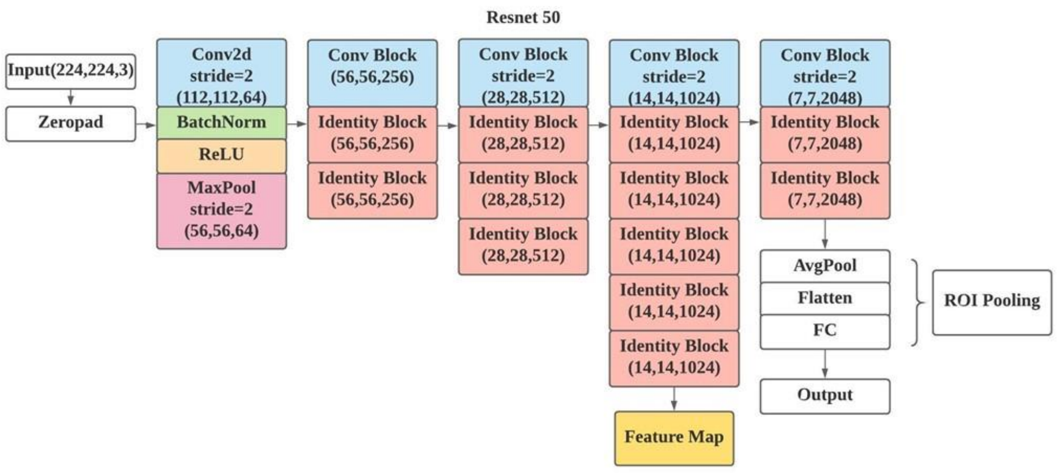
Resnet 50
Input(224,224,3)
Zeropad
Conv2d stride=2 (112,112,64)
BatchNorm
ReLU
MaxPool stride=2 (56,56,64)
Conv Block (56,56,256)
Identity Block (56,56,256)
Identity Block (56,56,256)
Conv Block stride=2 (28,28,512)
Identity Block (28,28,512)
Identity Block (28,28,512)
Identity Block (28,28,512)
Conv Block stride=2 (14,14,1024)
Identity Block (14,14,1024)
Identity Block (14,14,1024)
Identity Block (14,14,1024)
Identity Block (14,14,1024)
Identity Block (14,14,1024)
Feature Map
Conv Block stride=2 (7,7,2048)
Identity Block (7,7,2048)
Identity Block (7,7,2048)
AvgPool
Flatten
FC
Output
ROI Pooling

Figure 4.3: Resnet 50

4.1.3 Region Proposal Networks

RPN is the most important improvement of Faster R-CNN. There are nine anchors in RPN for detection. The anchors are pre-set boxes on the picture. The RPN will judge whether these anchors contain objects, and use offset obtained from bounding box regression to adjust anchors' position. Therefore, the proposals' function is to make a rough selection of the target in the picture and then judge the objects in the proposals through a classifier to make further adjustments to make the proposals more accurate.

RPN is a remarkable improvement made by Faster R-CNN and it is also the reason for the significant performance improvement. The traditional Selective Search method takes much time to generate the detection boxes. In contrast, the RPN has an advantage that it greatly improves the speed of generating the detection boxes because it is based on anchors (Ren, et al., 2016).

4.1.4 Region of Interest Pooling

The next ROI pooling layer is an improvement made by Fast R-CNN compared to R-CNN and Faster R-CNN also uses it.

The function of ROI layer is to use the roughly filtered proposals to intercept the feature maps for subsequent classification and bounding box regression operation (Girshick, 2015). Putting all the proposals on the feature maps for interception means that there are targets in these regions of the picture. Due to the different shapes of the proposals, the same number of proposal feature maps with different shapes and sizes will be obtained. Resnet's original fifth compression is performed on each proposal, and average pooling is performed after compression. After pooling is performed by region, proposal feature maps with the same shape can be obtained. Similar to generating proposals from anchors, it performs bounding box regression and classification as well. Anchors judge whether there is an object while proposals judge the type of object.

4.1.5 Faster R-CNN in apple detection

R-CNN is a classic target detection algorithm. As an outstanding representative of the R-CNN family, Faster R-CNN has been improved. In terms of speed and accuracy, it far exceeds R-CNN. RPN, which is the most important improvement of Faster R-CNN, profoundly influences other algorithms. Faster-RCNN uses the proposals for rough selection at first and then fine-tunes the proposals through classification and bounding box regression. Therefore Faster-RCNN is called the two-stage algorithm, which achieves a more accurate target detection effect. Due to many leaves and apples' small size, an algorithm that can accurately identify apples is required in apple detection. As an excellent two-stage algorithm, Faster R-CNN can first detect objects from the background and then select apples from the recognized objects to accurately detect apples in a complex background.

4.2 YOLO_V3

4.2.1 The structure of YOLOV3

YOLOV2 is one of the fastest detectors, but it struggled in the detection of small objects. To improve the accuracy of YOLO, the author traded off part of the speed and used a deeper Darknet to improve the detection precision. YOLOV3 uses a fully convolutional layer, which means that resizing of the feature map is done through convolution. The following figure shows the variant backbone network Darknet used in YOLOV3, which originally had 53 layers. YOLOV3 uses 106 fully convolutional layers for detection. The structure of YOLO_V3 can be seen in the picture 4.4 (Redmon and Farhadi, 2018).

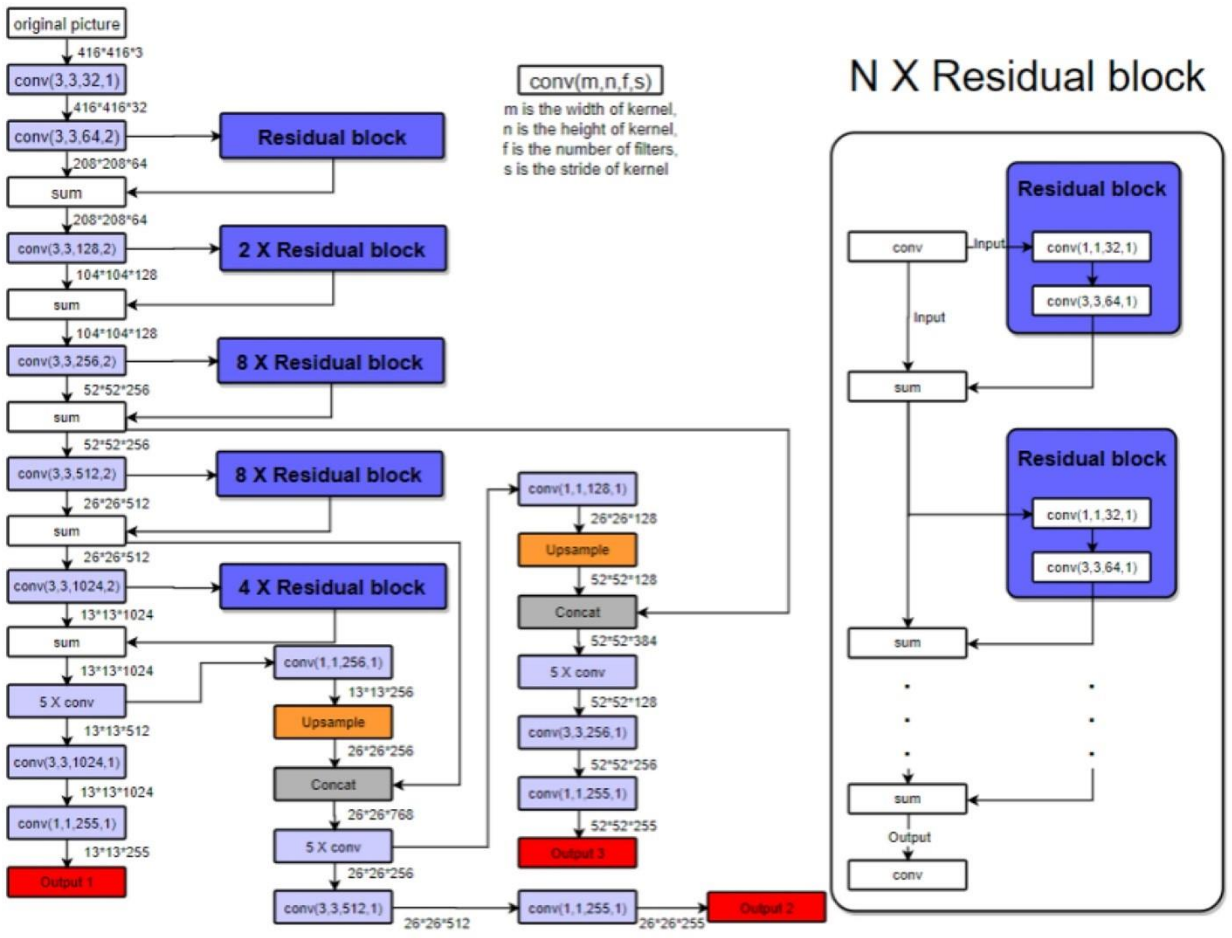

Figure 4.4: The structure of YOLO_V3

In the backbone feature extraction network, YOLOV3 continuously perform the down-sampling. As a result, the picture's height and width are continuously compressed, and the number of channels is continuously expanded. Many feature maps can be obtained through this downsampling process, representing the features of the input picture.

4.2.2 Multiple feature map for detection

Feature map detection of different scales helps to solve the problem of detecting small objects, which is a problem that YOLOV2 often encounters. The up-sampling layer connected to the upper layer helps to retain fine-grained features and helps detect small objects.

The 13×13 feature map[1] is responsible for detecting large objects, while the 52×52 feature map detects smaller objects, and the 26×26 feature map detects medium objects. This is a comprehensive comparison of different information collected from different layers in the same object.

4.2.3 Residual network

YOLOV2 lack an important element, the residual network, which results in the network depth may not be deepened because of the network degradation problem. However, YOLOV3 can use a deeper network by using a residual network to improve precision.

[1] YOLOV3 uses three feature maps to divide the picture into grids of 13×13, 26×26, and 52×52.

Experiments show that the residual network solves the degradation problem of deep neural networks well, and the residual network also converges faster under the premise of the same number of layers (He, et al., 2016). Besides, removing individual neural network layers, the performance of the residual network will not be significantly affected (Veit, et al., 2016), which is very different from the traditional feedforward neural network.

The residual unit can solve the reasons of network degradation from two perspectives. One is that in the chain propagation process of the residual, the input signal can be arbitrarily propagated from the low layer to the high layer (He, et al., 2016).

From another perspective, the residual network can be regarded as an ensemble model assembled from a series of path sets. The relatively short paths in the residual network mainly contribute to the gradient in training (Veit, et al., 2016).

4.2.4 Loss function

YOLOV2 uses a square loss function, while in YOLOV3, they have been replaced by a cross-entropy loss function. Object detection is a discrete classification problem rather than a continuous regression problem, so it is more appropriate to use cross-entropy as the loss function. Because the square loss function not only considers making the predicted value of correct classification as large as possible, but also making the predicted value of incorrect classification as small as possible, but we do not care about the latter part. For logistic regression, that is, using cross-entropy as the loss function, it only cares about the result of the correct classification, which can train the network more precise.

4.2.5 YOLOV3 in apple detection

Several improvements of YOLOV3 mentioned above are beneficial for apple counting. YOLOV3 has improved small object detection and network depth. Therefore, although YOLOV3 has a slight decrease in speed, its accuracy should be greatly improved. YOLOV2 is considered to be close to faster R-CNN in precision, so we can speculate that YOLOV3 should be more precise than faster R-CNN. Meanwhile, YOLOV3 detects every gird of pictures so there are fewer mistaking background errors as an object. It is very beneficial to Apple detection, which can effectively avoid the interference of complex background factors such as leaves and shadows, and accurately identify objects.

5 Experiment and Implementation

5.1 The preparation of implementation

The version of Python used is 3.7, and the IDE of Python is PyCharm. Additionally, the main packages used to construct the experimental framework is Opencv, Pytorch (torch 1.6.0), numpy, os, matplotlib, tensorboard etc., and cuda10.2 is used for GPU acceleration. Computing platform is shown in the table3 below:

Table 3: The parameters of the calculation platform

Hardware	Model/Storage
CPU	E5-2673 v3 @ 2.40GHz
RAM	64G
Graphics Card	NVIDIA GeForce GTX 1660

The images (666 images totally) in the folder named train in the MinneApple dataset (Isler, 2019) are utilized to make this experiment's dataset. 10% of the dataset was assigned randomly to test set; 10% of the rest was used as validation set, and meanwhile 90% of the rest was used as train set. For Faster R-CNN and YOLOv3, a pre-trained weight file of VOC07 and VOC12 datasets and a pre-trained weight file of COCO datasets are loaded separately.

5.2 Tricks in YOLOv3 and Faster-RCNN

(1) IOU:

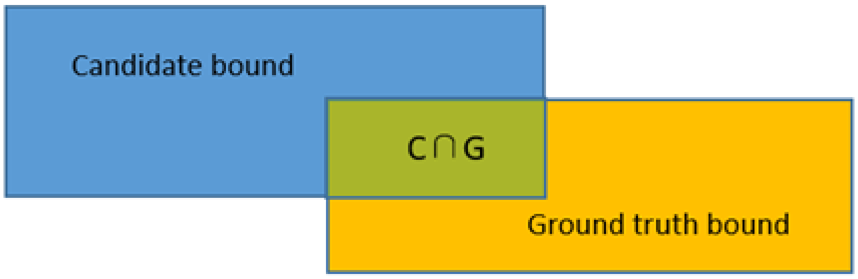

Figure 5.1: The structure of the IOU

$$\text{IOU}=\frac{area(C)\cap area(G)}{area(C)\cup area(G)}$$

The Intersection-Over-Union (IOU), which can be seen in the figure 5.1 above, represents the overlap rate between the candidate bounding box and ground truth bounding box. In generally, IOU is used in NMS and the process of calculating mAP.

(2) NMS:

This trick can be used to delete redundant prediction boxes in the results of prediction. The confidence scores of all candidate boxes are sorted in descending order, and the box which has the highest score is selected (Neubeck, 2006). Next, via comparing IOU between the rest of boxes and the one selected before in sequence, the boxes whose IOU exceeds the threshold will be deleted. The box which has highest score from the rest of unprocessed boxes is selected to repeat the NMS process until all the bounding boxes are filtered.

(3) Adam:

Adaptive Moment Estimation (Adam) algorithm is an algorithm with adaptive learning rate, it is a combination of Momentum algorithm and RMSProp algorithm. Empirical results show that Adam algorithm has excellent performance in practice and has great advantages over other types of stochastic optimized algorithms (Ba, 2015).

(4) lr_scheduler:

The lr_scheduler method is used to adjust learning rate of Adam optimizer. Every time the model runs one epoch, the learning rate drops at the ratio which is set in Adam optimizer.

5.3 Parameter adjustment

5.3.1 YOLOv3

(1) Batch size:

The batch size determines the number of iterations in every epoch. The training speed of using a big batch size is faster than which one using small batch size, but the model is easy to fall into the local optimal solution (Jastrzębski, 2017). Therefore, batch size is set as 8 in previous 50 epochs and 4 in the later 50 epochs to obtain a converged and efficient decline of loss.

(2) Epoch and learning rate(lr):

One epoch is one procedure to train all samples belonging to tarin test, and learning rate affects the training speed. If the value of lr is high, loss decreases fast. However, the model is difficult to converge and tend to vibrate. If the value of lr is low, the model will overfitting and waste much time to train. After experimenting several times, lr is determined as 1e-3 in previous 50 epochs, and it is determined as 1e-4 in later 50 epochs. The Fig.5.2 is shown below. Under this condition, loss declines steadily and converges gradually as the epoch increases. Furthermore, the data of lr and train loss are written into log files during the training process, and these data are visible via Tensorboard after training. The Fig.5.3 below demonstrates the relation of lr and epoch and train loss and epoch, and the value of train loss declines smoothly and converges in the end. Furthermore, the trend of loss in the validation set is quite similar to that in the training set, which means that there are no overfitting and underfitting in the process of training.

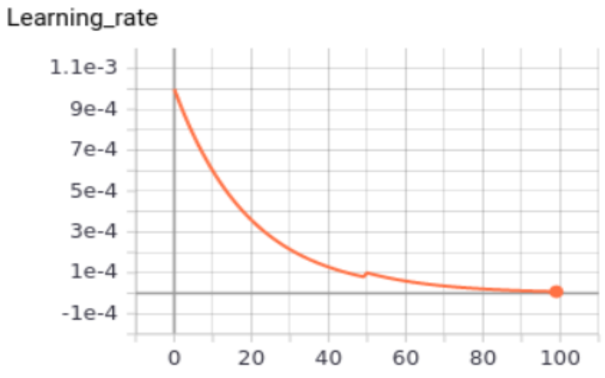

Figure 5.2: The changes in learning rate

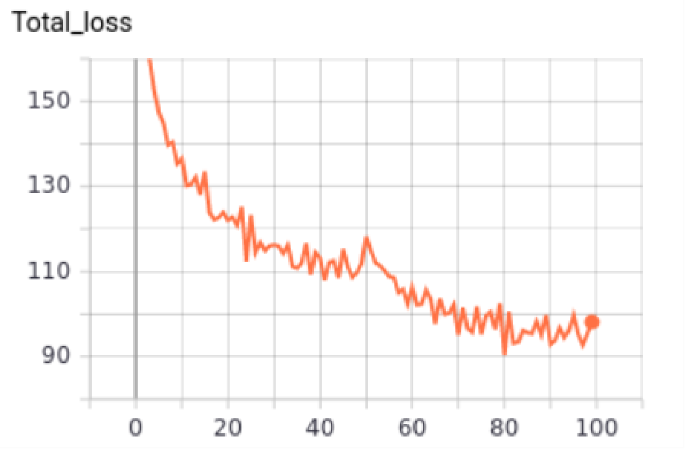

Figure 5.3: Total loss

5.3.2 Faster-RCNN

(1) Data loader:

Data loader includes some initialize settings. Firstly, batch size is set as one, which means the number of samples in one dataset and the number of iterations were equal in every epoch. And the data will be reordered at the beginning of running each epoch to avoid invalid training. Secondly, Shuffle is set True, and it means that the dataset is out of order, which is beneficial to train the dataset sufficiently.

(2) RPN:

The candidate bounding boxes whose IOU is between 0.3 and 0.6 will be deleted. If IOU >0.6, the samples will be recognized as positive samples, and if IOU<0.3, based on the parameter of adjustment. To compare with initial parameter, the number of positive samples will increase, and more samples will be used to train than before.

(3) Epoch and learning rate(lr):

At first, 50 epochs are set, but the loss cannot converge. Comparing with YOLOv3, learning rate is set as 1e-3, loss vibrates heavily. Therefore, 150 epochs are set, and 50 epochs are in the first part with initial learning rate of 10-4, and 100 epochs in the second part with initial learning rate of 1e-5. The relation of lr and epoch is shown in Fig.5.4.

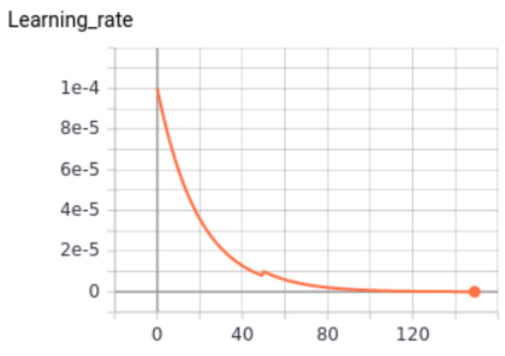

Figure 5.4: Learning rate

As shown in Fig.5.5, train loss declines as the number of epochs increases and converges in the end. Furthermore, the trend of loss in the validation set is quite similar to that in the training set. This means there are no overfitting and underfitting.

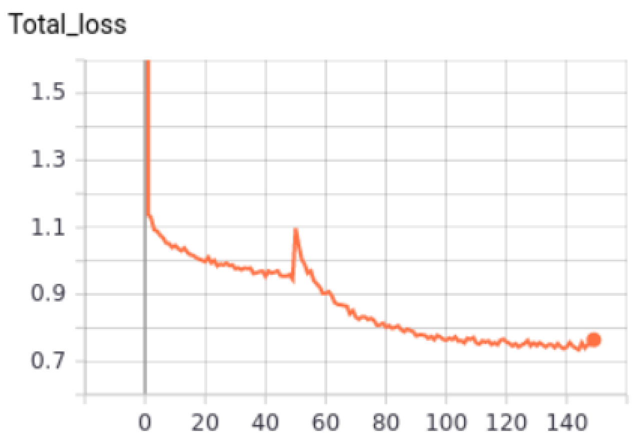

Figure 5.5: Val_loss

6 Results and Evaluation

6.1 Evaluation metrics

After training, a comparative optimal weight file is selected from all the wight files. The standard of selection is that the train loss is as low as possible and the model is not overfitting. After analysing the result of prediction, the weight file of 140th epoch is selected for Faster R-CNN, and the weight file of 25th epoch is selected for YOLOv3 because there are some signs of overfitting in the latter epochs (Elad Hoffer, 2017).

6.1.1 TP, TN, FP and FN

One sample is regarded as a positive sample when the bounding box includes apple, and if the box does not include apples, the sample is negative.

6.1.2 Precision(P) and Recall(R)

Precision represents the ability of model to find the correct sample in all sample have been found.

$$P = \frac{TP}{TP + FP}$$

Recall represents the ability of model to find all positive sample. If the final recall is closed to one, the model is supposed to have almost found all positive samples.

$$P = \frac{TP}{TP + FN} = \frac{TP}{All\ ground\ truths}$$

6.1.3 Average Precision (AP)and mean Average-Precision (mAP)

When the threshold of confidence is definite, every value of recall has a maximum of precision correspondingly, and as recall increases from 0 to maximum, the sum of all the values of maximal precision is AP, which is the main assessment indicator of object detection algorithm (Henderson, 2017). The mAP refers to the area enclosed by the precision-recall (p-r) curve. A higher value of mAP represents better effect of the detection model. The calculation formular of mean Average-Precision (mAP) is (c is the number of classes):

$$mAP = \frac{\sum_{i=1}^{c} AP_i}{C}$$

6.2 Result

6.2.1 Precision-recall diagram

After calculating the precision and recall, these data can be used to generate a two-dimensional image. For every recall value, there is a largest value of precision correspondingly. Two P-R pictures 6.1 and 6.2 about YOLOv3 and Faster-RCNN are shown below.

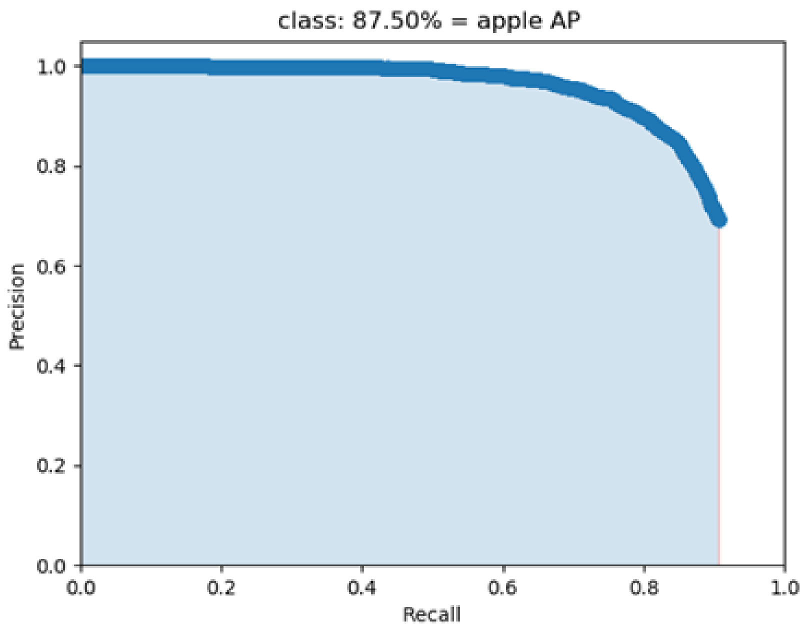

Figure 6.1: YOLOv3 P-R picture

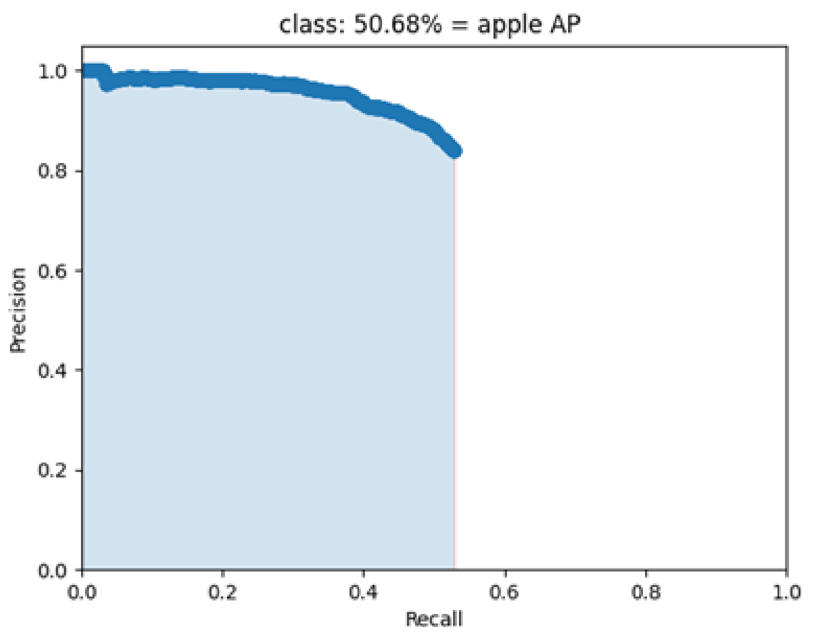

Figure 6.2: Faster-RCNN

6.2.2 mAP

These values are obtained under the condition that confidence threshold is 0.5, and the table 3 of all the metrics introduced is presented below, and results will be evaluated in detail in 6.3:

Table 4: Evaluation data for YOLOv3 and Faster-RCNN

Method	**TP**	**FP**	**Precision**	**Recall**	**mAP**	**Running time (Hour)**
YOLOv3	2805	1248	69.2%	90.7%	87.5%	3
Faster-RCNN	1636	318	83.7%	52.9%	50.68%	6

6.2.3 The detection results of two pictures by two methods

One red apple picture and one green apple picture are selected from test set for prediction by two methods. Figures 6.3 and 6.4 show the test images of YOLOv3 and Faster-RCNN respectively.

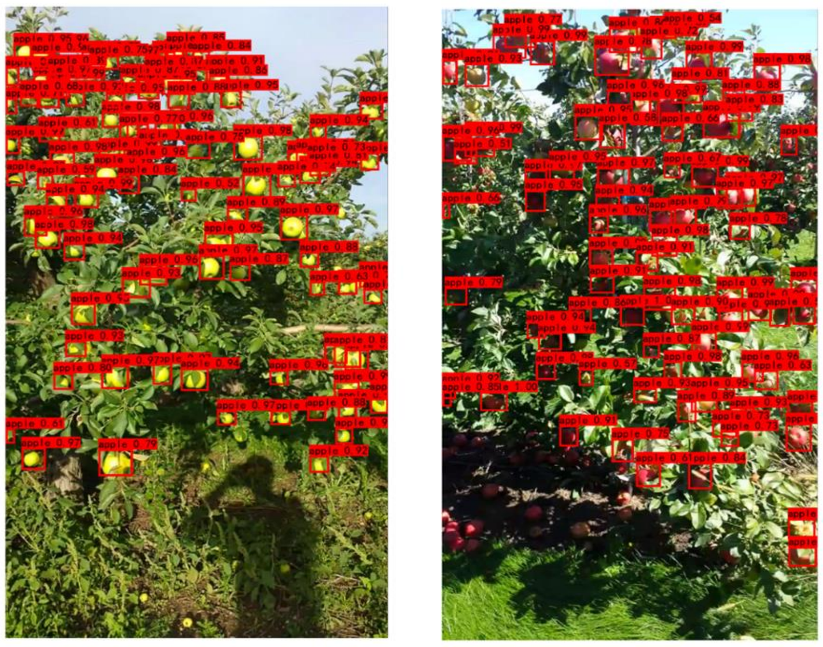

Figure 6.3: test set by YOLOv3

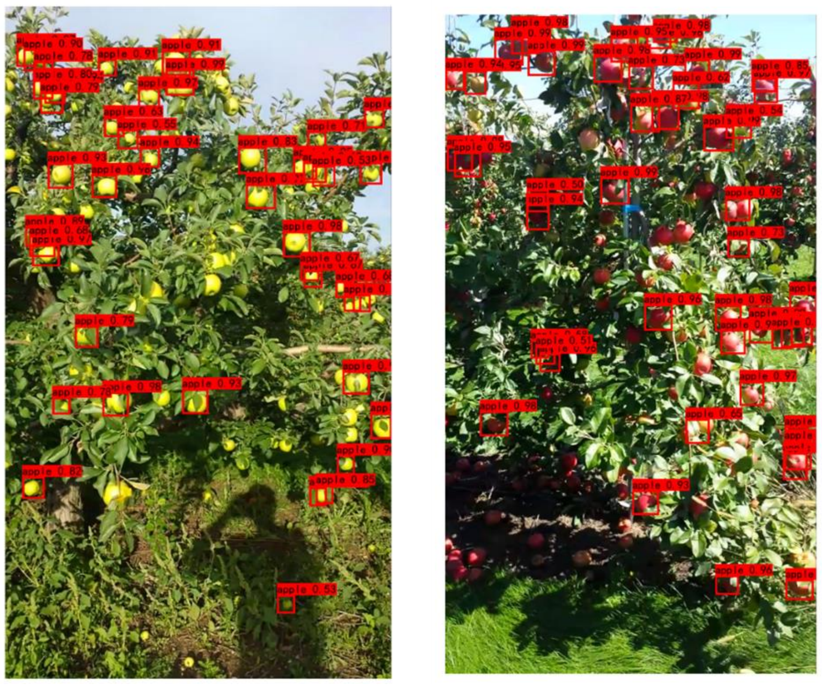

Figure 6.4: test set by Faster-RCNN

6.3 Evaluation

As shown in Table 3, Faster-RCNN model spends about 6 hours to reach the convergence, and the YOLOv3 model spends about 3 hours. Although these two models have different initial learning rates, YOLOv3 is more efficient than Faster R-CNN to some extent.

Additionally, mAP of Faster R-CNN is far below YOLOv3's, and YOLOv3 shows a stronger ability of apple detection than Faster R-CNN. In detail, recall of Faster-RCNN is very low because the quantity of TP about Faster-RCNN is small, which leads to low value of mAP. Furthermore, these results are anastomotic basically with the result of single-picture test. As these pictures presented, Faster-RCNN cannot accurately identify apples stack together and covered by leaves, and apples with medium sizes are easy to be detected, but some with small sizes are omitted. There are three reasons causing this result:

1. The initial parameter of receptive field in Faster R-CNN's backbone (Resnet 50) is not suitable for apple detection, especially it is weak in detecting apples stacking together or covered by leaves.
2. The proposal bounding boxes' size of RPN or some other mechanisms of RPN ignore some essential information of samples so that the model cannot be trained sufficiently and accurately.
3. Apples on the ground or far away in these pictures of the train dataset are not labeled as ground truth, which causes certain negative influence on training, and pictures of green apple are much more than red apple's, which causes the accuracy of red apple's detection is not very ideal.

At the same time, the number of FP about YOLOv3 is appreciably large. Although YOLOv3 is not perfect, its accuracy is much better than Faster-RCNN's. It is worth mentioning that, in the process of YOLOv3's training, the result is overfitting in the second 50 epochs. However, there are not some unusual signs on the validation loss. It is because that the validation set are lack of generalization, in other words, it is too similar to the train set, which results in the invalidation of validation set.

In conclusion, the result of two approaches is basically consistent with the advantages and disadvantages of two methods.

7 Conclusions and Future works

7.1 Conclusions

After completing the apple counting task in both methods, although the code is in part referenced by an author on GitHub (Anon., 2020). The challenges of the apple counting task were not well tackled. The mAP value of Faster-RCNN was surprisingly low, although the YOLOv3 algorithm had a high mAP value in this experiment. However, this is because YOLOv3 mis-tests many apples during its performance, which means that YOLOv3 has a high FP in the apple counting task. This may be related to the fact that YOLOv3 has poor detection of small object clusters. In general, both methods do not overcome the challenges posed by different lighting conditions, differences in apple appearance, leaf shading and other environmental issues in the apple counting task.

7.2 Future works

7.2.1 3D Object Detection

Although 3D object detection is currently used mainly in the field of autonomous driving, it is undeniable that 3D vision detection has an additional dimension of position and size compared to 2D vision detection, and that it is insensitive to environmental light and has advantages that 2D object detection does not have for the task of counting fruit in orchards.

7.2.2 Three tricks fine-tuning measures

a) Individual detection thresholds for each branch

Different layers use different confidence levels to improve TP (Ture positive).

b) GIOU loss and focal loss

c) Warm up and cosine learning rate

(Bormann, et al., 2020)

7.2.3 Visual Transformer

Transformer is a new algorithm for target detection, and has competitive accuracy compared to common vision algorithms (Han, et al., 2020). Therefore, it is expected to perform well in apple counting.

References

Ba, D., (2015). *Adam: A Method for Stochastic Optimization.* s.l., s.n.

Bargoti, S. and Underwood, J., 2017. Image Segmentation for Fruit Detection and Yield. *Jouranl of Field Robotics,* 34(6), pp. 1039-1060.

Beucher, S. and Meyer, F., (1993). The morphological approach to segmentation: the watershed transformation. In: E.R.Douherty, ed. *In mathematical Morphology in Image Processing.* s.l.:s.n., pp. 433-481.

Bormann, R., Wang, X. and Xu, J., (2020). Bormann R, Wang X, Xu J, et al. DirtNet: Visual Dirt Detection for Autonomous Cleaning Robots. In: *2020 IEEE International Conference on Robotics and Automation (ICRA).* s.l.:IEEE, pp. 1977-1983.

Chen, S., Shivakumar , S. and Dcunha , S., (2017). Counting apples and oranges with deep learning: A data-driven approach. *IEEE Robotics and Automation Letters,* 2(2), pp. 781-788.

Chinchor, N., (1992). *MUC-4 Evaluation Metrics.* s.l., in Proc. of the Fourth Message Understanding Conference.

CNH, (2019). *CNH.* Available from: https://www.cnhindustrial.com/en-US/media/thedaythefarmchanged/pages/default.aspx
[Accessed 16 December 2020].

Dias, P., Tabb , A. and Medeiros , H., (2018). Apple flower detection using deep convolutional networks. *Computers in Industry,* Issue 99, pp. 17-28.

Duda, R. and Hart, P., (1972). Use of the Hough Transformation to Detect Lines and Curves in Pictures. *Communications of the ACM,* 15(1), pp. 11-15.

Elad Hoffer, I., (2017). *Train longer, generalize better: closing the generalization gap in large batch training of neural networks.* s.l., NIPS 2017.

Fuentes , A., Yoon , S. and Kim , S., (2017). A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition. *Sensors,* 17(9), p. 2022.

Gamaya, (2016). *Gamaya.* [Online] Available from: https://www.gamaya.com/ [Accessed 16 December 2020].

Girshick, R., (2015). Fast r-cnn. In: *Proceedings of the IEEE international conference on computer vision.* s.l.:s.n., pp. 1440-1448.

Gongal, A. et al., (2015). Sensors and Systems for Fruit Detection and localization: A review. *Computers and Electronics in Agriculture,* Volume 116, pp. 8-19.

Anon.,(2020). Github.
Available from: https://github.com/bubbliiiing?tab=overview&from=2020-03-01&to=2020-03-31
[Accessed December 2020].

Häni, N., Roy, P. and Isler, V., (2019). A comparative study of fruit detection and counting methods for yield mapping in apple orchards. *Field Robotics*, pp. 1-20.

Häni, N., Roy, P. and Isler, V., (2019). *MinneApple: A Benchmark Dataset for Apple Detection and Segmentation.* Available from: https://conservancy.umn.edu/handle/11299/206575 [Accessed 10 December 2020].

Han, K., Wang, Y. and Chen, H., (2020). A Survey on Visual Transformer.

He, K., Zhang, X., Ren, S. and Sun, J., (2016). Deep Residual Learning for Image Recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770-778.

Henderson, P. and F. V., (2017). End-to-End Training of Object Class Detectors. *Lecture Notes in Computer Science,* p. 198–213.

Isler, N., (2019). MinneApple: A Benchmark Dataset for Apple Detection and Segmentation.

Jastrzębski, S.,(2017). *Three Factors Influencing Minima in SGD.* s.l., ICLR.

Jiang, P., Chen, Y. and Liu, B., (2019). Real-time detection of apple leaf diseases using deep learning approach based on improved convolutional neural networks. *IEEE Access,* Volume 7, pp. 59069-59080.

Kapach, K. et al., (2012). Computer vision for fruit harvesting robots-state of the art and challenges ahead.. *International Journal of Computational Vision and Robotics,* 3(1-2), pp. 4-34.

Liu, S., Whitty, M. and Cossell, S., (2015). Automatic grape bunch detection in vineyards for precise yield estimation. In: *14th IAPR International Conference on Machine Vision Applications (MVA).* Tokyo: Curran Associates, pp. 238-241.

Liu, Z., Wu, J. & Fu, L., (2019). Improved kiwifruit detection using pre-trained VGG16 with RGB and NIR information fusion. *IEEE Access,* Volume 8, pp. 2327-2336.

Neubeck, A. a. V. G. L., (2006). *Efficient non-maximum suppression..* s.l., In 18th International Conference on Pattern Recognition (ICPR'06) (Vol. 3, pp. 850-855). IEEE..

Nuske, S. et al., (2014). Automated Visual Vield Estimation in Vineyards. *Journal of Field Robotics,* 31(5), pp. 837-860.

Padilla, R., (2018). *Github.* Available from: https://github.com/rafaelpadilla/Object-Detection-Metrics [Accessed 10 January 2021].

PlantTape, (2019). *planttape..* Available from: https://www.planttape.com/ [Accessed 16 December 2020].

Redmon, J., Divvala, S., Girshick, R. and Farhadi, A., (2016). You Only Look Once: Unified, Real-Time Object Detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 779-788.

Redmon, J. and Farhadi, A., (2018). *Yolov3: An incremental improvement,* s.l.: s.n.

Ren, S., He, K., Gireshick, R. and Sun, J., (2016). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE transactions on pattern analysis and machine intelligence,* Volume 39, pp. 1137-1149.

Roy, O., Dong, W. and Isler, V., (2018). *Registering reconstructions of the two sides of fruit tree rows.* International Conference on Intelligent Robots and Systems (IROS), IEEE.

Redmon, J. and Farhadi, A., (2017). YOLO9000: better, faster, stronger. In: *Proceedings of the IEEE conference on computer vision and pattern recognition.* pp.7263-7271.

Sa, I. et al., (2016). Deepfriots: A fruit detection system using deep neural networks. *Sensors,* 16(8), p. 1222.

Selvaraj , M. G., Vergara , A. and Ruiz , H., 2019. AI-powered banana diseases and pest detection. *Plant Methods,* 15(1), p. 92.

Veit, A., Wilber, M. and Belongie, S., (2016). Residual networks behave like ensembles of relatively shallow networks. *Advances in neural information processing systems,* Volume 29, pp. 550-558.

Wang, Q., Nuske, S., Bergerman, M. and Singh, S., (2013). Automated Crop Yield Estimation for Apple. In: J. Desai, G. Dudek, O. Khatib & V. Kumar, eds. *Experimental robotics.* s.l.:Heidelberg: Springer International, pp. 745-758.

Zou, Z., Shi, Z., Guo, Y. and Ye, J., (2019). Object Detection in 20 Years: A Survey. *IEEE*, 16 May.