World Academy of Science, Engineering and Technology
International Journal of Computer and Systems Engineering
Vol:15, No:5, 2021

A Comparison of YOLO Family for Apple Detection and Counting in Orchards

Yuanqing Li, Changyi Lei, Zhaopeng Xue, Zhuo Zheng, Yanbo Long

***Abstract*—**In agricultural production and breeding, implementing automatic picking robot in orchard farming to reduce human labour and error is challenging. The core function of it is automatic identification based on machine vision. This paper focuses on apple detection and counting in orchards and implements several deep learning methods. Extensive datasets are used and a semi-automatic annotation method is proposed. The proposed deep learning models are in state-of-the-art YOLO family. In view of the essence of the models with various backbones, a multi-dimensional comparison in details is made in terms of counting accuracy, mAP and model memory, laying the foundation for realising automatic precision agriculture.

***Keywords*—**Agricultural object detection, Deep learning, machine vision, YOLO family.

I. Introduction

THE combination of information technology and agriculture is a new topic in the process of agricultural development. The application of machine vision in agriculture has laid a foundation for precision agriculture and agricultural production automation. It not only helps to liberate labour force but also helps to improve the quality and yield of crop products. Apple is one of the most widely consumed fruits in the world. Apple detection and counting and localisation in apple orchard are difficult problems in precision agriculture and agricultural automation. In this paper, through the different types of YOLO networks, the methodologies are applied to the topic of apple detection and counting. The main aim is to compare the accuracy and occupied memory of different YOLO models. It realises the exploration of the machine vision method and provides the feasibility for developing orchard specific crop automation system.

In Section II, related works and problem formulation on the topic of apple detection counting are presented. Data acquisition details together with the utilisation of and datasets are presented in Section III. Section IV demonstrates an overview of the approaches and the methodological details of the diverse approaches, including the traditional machine vision technic and deep learning method. Experiment process and implementation details are shown in Section V. Then the results of experiments along with evaluation and the strengths and weaknesses of each approach are correlated in Section VI. In Section VII, Specific challenges relevant to the topic of apple detection counting and localisation are revealed. Besides, the future work focus and potential methods of panorama study are introduced.

II. Related Works

Early in 1977, Parrish Jr. and Goksel presented a rudimentary experimental system for automated apple harvesting. The system used image pattern recognition and other artificial intelligence techniques to recognize apples [1]. Some solutions are given to solve the problem of overlapping and rough edges but only suitable for ideal conditions. Based on previous research, scientists summarized a method using the colour feature to separate the fruits from leaves called threshold segmentation. In most cases, this method is used with corresponding screening algorithm to eliminate the influence of shapes and textures. Zhou developed an apple recognition algorithm with colour difference R - B (red minus blue) and G - R (green minus red) for apple images after June drop, and with proper threshold segmentation, the apples are detected [2]. To further develop the recognition model, Qian added Hue, Saturation, Value (HSV) features and increased the identification success rate to over 90%. Because of the introduction of HSV, this method is more suitable for high-illumination cases [3]. For more advanced methods, Payne et al. proposed one calculating pixel properties from RGB and YCbCr colour spaces and applied a series of filters to classify the leaves and mature fruits as background and foreground, producing a binary mask and the correlation coefficient (R^2) of this method is 91.7% [4].

Besides the colour-based recognition, shape and texture features can also segmentation method. Si developed a random ring method to extract the shape features of fruits and with RGB colour feature. A matching algorithm based on epipolar geometry was used to locate apples. In addition, apples with similar shapes were matched according to the principle of ordering constraint. This method reduced the error of illumination and shadow, and the accuracy reached 89.5% [5]. In Li's project, the grey-scale difference statistical method is applied to get the texture feature vector of the image and the support vector machine is used to segment the image preliminarily, and then the shape and colour features are combined to achieve precise segmentation. Compared with colour segmentation, the method increased the accuracy from 72.4% to 88.25% [6]. A recent project of apple detection implemented by Tanco used morphology test to detect circular structures and got an accuracy of 91.5% [7].

The detection of fruits in orchard mainly depends on the

Yuanqing Li is with the University of Bristol, United Kingdom (e-mail: nj20193@bristol.ac.uk).

colour, shape and texture extraction methods, and the experiments provide references for detection under natural illumination. However, these methods still have several limitations because of uneven illumination, overlapping and environmental influences. The purpose of the project is to further study on these problems and find possible solutions.

In recent years, deep learning (DL) has been widely used, especially in the field of intelligent agriculture, such as pest detection, plant and fruit identification, crop and weed detection and classification. Object detection networks are proposed depend on region proposal algorithms to hypothesize object locations [8]. Bargoti used image-based fruit detection system for agriculture tasks such as yield mapping and robotic harvesting [9]. Chen presented a novel data-driven fruit counting pipeline based on DL [10]. Bruno et al. used segmentation based on U-Net DL as their methodology for corn plant counting [11].

The DL models mainly include CNN, R-CNN family, Sppnet and YOLO family. Early in 1995, CNN is applied in clinic for medical image pattern recognition. The positions of objects are determined by sliding window and region proposal and corrections are based on bounding box regression. In CNN, objects are classified using hog, dpm, haar, sift+svm, adaboost etc.

The R-CNN family includes R-CNN, Fast R-CNN and Faster R-CNN. In 2014, Girshick proposed an improved model based on CNN called R-CNN [12]. The position is determined by region of interesting which includes selective search and extraction candidate box and is modified by linear regression. Then it is classified using CNN feature extraction and SVM classification. In 2015, He introduced the Spatial Pyramid Pooling Networks to convert the features of different scales into fixed special-scale features and then the full connection layer can be connected [13]. One of the critical advantages of this network is the usage of shared convolution which significantly reduces the computation. In the same year, Girshick improved the model using SPP network and demonstrated the Fast R-CNN [14]. The determination of position is the same as R-CNN, but the classification and position modification are all based on CNN. CNN feature extraction and classification are applied in object classification and CNN regression is used for position modification. The key improvement of this model is the introduction of SPP layer, the classification is proceeded through SPP layer to the ROI pooling to accelerate the computation. In 2016, Ren introduced Region Proposal Network (RPN) and proposed Faster R-CNN and the difference to other models is the method of position determination [8]. This model applies CNN to extract candidate box and RPN is used to generate feature point corresponding to the positions of the original image to generate anchor boxes with different scales. Then the anchor boxes are dichotomized and regressed to generate the correct position.

Being inspired by RPN networks, Liu proposed a model called Single Shot MultiBox Detector (SSD) in 2016 [15]. The detection method is like the anchor box of RPN, different scale anchor boxes are generated at the original image receptive field frame corresponding to feature points, but anchor boxes are directly used for full classification and regression without using candidate boxes. The features are extracted and classified using CNN and output by multi-receptive field feature layer. In 2017, the Feature Pyramid Networks (FPN) was proposed [16]. The detection theory is similar to SSD, but the difference is the features are output after the up-sampling fusion.

In 2016, Redmon first demonstrated the method of YOLOv1 which learns very general representations of objects and outperforms other detection methods, including DPM and R-CNN, when generalizing from natural images to other domains like artwork [17]. The image is divided into 7*7 grids and two anchor boxes are generated for each grid. Different from R-CNN models, there is no candidate box in YOLO and the anchor box is directly classified and regression using the width, height and coordinates of the centre. In 2017, Redmon proposed YOLOv2, a real-time object detection system that can detect over 9000 object categories [18]. Compared with YOLOv1, more abundant anchor boxes are applied and K-means algorithm is used to count the anchor boxes in dataset. For the object classification, the classification prediction is decoupled from the space position cells, and the anchor boxes are used to predict the categories and coordinates. This improvement effectively solves the overlapping problem of objects. In 2018, the YOLO models are updated again and YOLOv3 was proposed. Different from YOLOv2, the backbone used in YOLOv3 is evolved from Darknet 19 to Darknet 53 and multiple scale is considered, the receptive field is more diverse [19]. In addition, more anchor boxes are used and several receptive fields output together. In 2020, impressive progress of the YOLO model was proposed and demonstrated in YOLOv4 [20]. Several advanced methods are used to process images include the Weighted-Residual-Connections (WRC), the Cross-Stage-Partial-connection (CSP), the Cross mini-Batch Normalization (CmBN), the Self-adversarial-training (SAT) and the Mish-activation. In addition, Mosaic is applied for data enhancement and when dealing with common problems, DropBlock is used for regularization and CIoU is used to calculate errors. The YOLO family has been widely used in object detection. In 2019, Tian improved YOLO detection model by incorporating the DenseNet method for detecting [21]. While data augmentation methods and the detection model need to be optimized to further improve the detection accuracy.

III. Datasets

A. Data Acquisition

Our project is based on the MinneApple dataset [22] collected at the University of Minnesota Horticultural Research Center (H.R.C.) in Eden Prairie, Minnesota June 2015 and September 2016. To reflect the diversity of our data and the ability of our network to steadily adapt to environmental changes, the types of apples include, firstly, green and red apples in terms of colour, large mature apples and underdeveloped apples in terms of maturity, apples dropped on the ground and apples in the distance in terms of location, apples in the shadow of leaves and apples in overcast in terms

of environments. The heaps of covered apples have been a great challenge to the training of the model. Example images are shown in Figs. 1 and 2. Considering the project's aim is to assist with automated fruit picking and yield estimation, local and global detections should be included, and the dataset is proper to use. Eventually, manual annotation needs to be added.

B. Semi-Automatic Image Annotation

Fig. 1 Apple trees in sunny environments

Fig. 2 Apple trees in cloudy days

Semantic segmentation and pixel level labelling leads to the disability of detecting scattered apples on the distant background and ground. A semi-automatic image annotation approach with TensorFlow and Keras object detection models is proposed in this section. The specific implementation steps are as follows:

1. A preliminary model is used to detect the small batch of data to be labelled. The preliminary model here can be trained by ourselves with a small batch of data sets.
2. The results were corrected by human intervention.
3. The corrected data are trained into a new model.
4. The model is used to detect the partial batch data which needs to be tested.
5. Through 1 ~ 5 steps of cyclic iteration, datasets will be refined step by step.

Though the method requires human participation as well, the workload can be significantly alleviated. Fig. 3 introduces the G.U.I. of Semi-Automatic Image Annotation Toolbox. Fig. 4 presents the process of utilising "Labelimg" Toolbox to modify the details of the dataset. In this section, we operated the feature engineering and counted the apples' size and location distribution, as shown in Fig. 5.

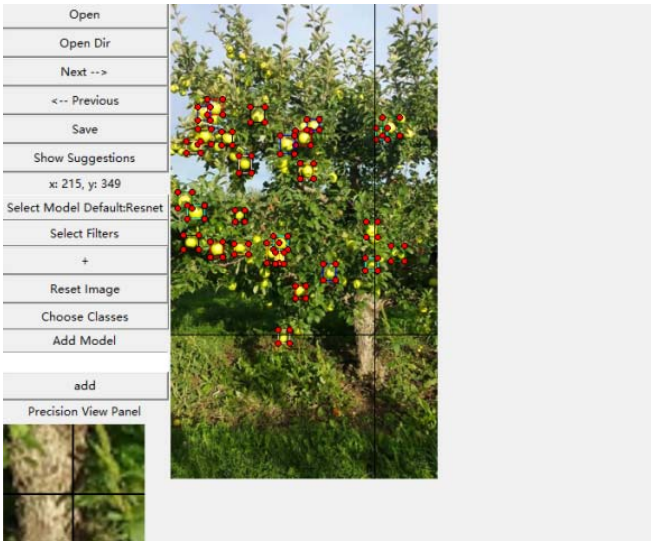

Fig. 3 Semi-Automatic Image Annotation

IV. Methodology

For a detection network, there are three main requirements:

1. Increasing the resolution of the network input to improve the performance of detecting small targets.
2. Using more network layers to expand the network's receptive field, so that the network can accept large resolution input.
3. Introduce multiple network parameters to better detect objects of different sizes in the image

To satisfy these requirements, YOLO family can be an appropriate choice.

YOLO family is a well-known series of neural network that have been widely used in object detection tasks, such as pedestrian detection, face detection, crowd counting, text detection, traffic sign and traffic light detection, etc. Compared with other mainstream object detection models including RCNN, Fast-RCNN, RetinaNet and CenterNet, YOLO family demonstrates outstanding performance on the training model's hardware performance requirements, accuracy, real-time, the complexity of the model, and the difficulty of implementation.

YOLO family is Convolutional Neural Network based one-stage detector, consisting of five different networks, i.e., YOLOv1, YOLOv2, YOLOv3, YOLOv4 and YOLOv5. The core idea of YOLO family is to transform the object detection problem into a regression problem. Taking the entire picture as

the input of the network, and the position of the bounding box and its category can be obtained through only one neural network, as the name of YOLO, "You Only Look Once". Among them, YOLOv3, YOLOv4 and YOLOv5 are more popular and more commonly used in real-world applications. In this paper, these three networks are chosen to be our methods to solve the apple detection and counting problem.

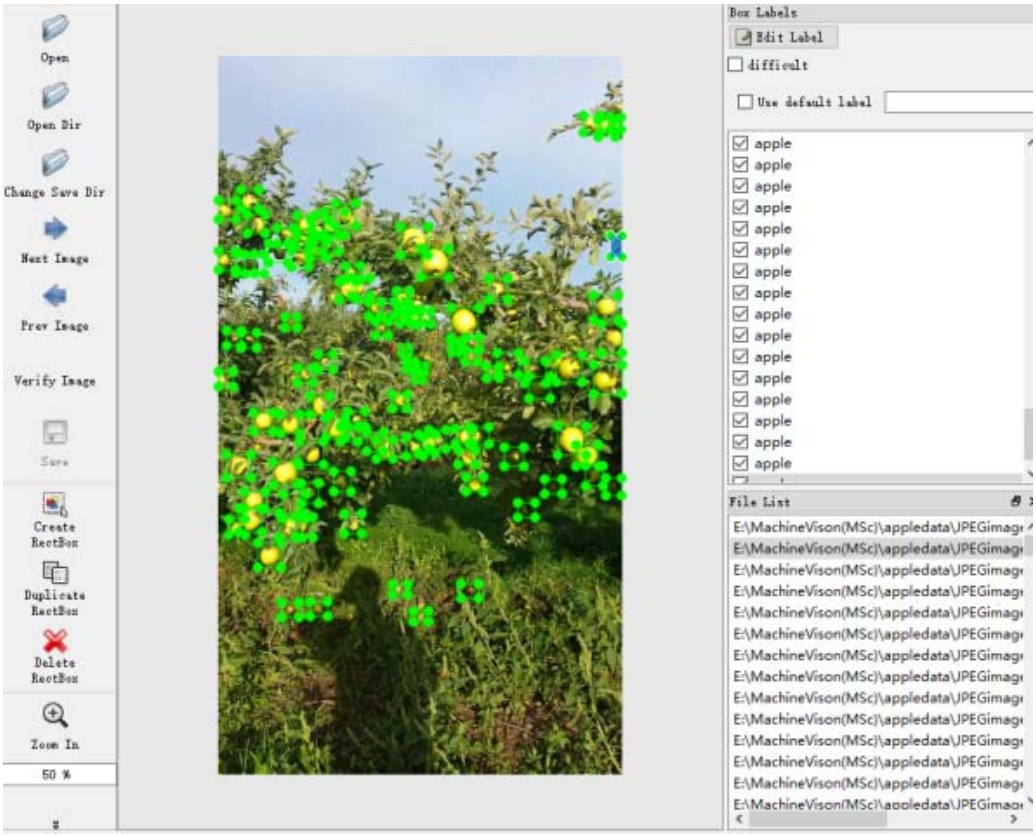

Fig. 4 "Labelimg" Toolbox

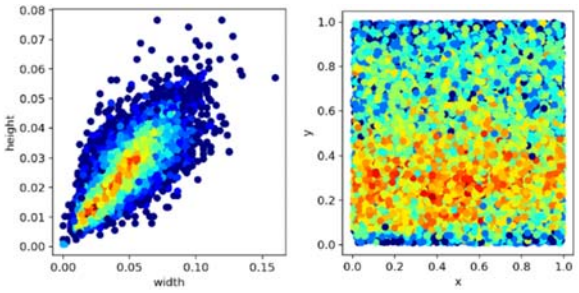

Fig. 5 Feature engineering

YOLOv3 is an end-to-end detection model based on YOLOv2, but it has made significant improvements. YOLOv3 replaces the backbone of Darknet-19 with Darknet-53, which introduces a residual structure that dramatically reduces the difficulty of training deep networks caused by the sequential structure of Darknet-19 (same as normal CNN), i.e., gradient vanishing problem induced by increasing depth of network. The Darknet-53 is a feature extractor that uses successive 3×3 and 1×1 convolutional layers to extract features from input images. It is also proved that Darknet-53 performs better than ResNet-101, ResNet-152 and Darknet-19 in the balance of classification accuracy and efficiency. Besides, YOLOv3 changed YOLOv2's single-label classification to multi-label classification in terms of category prediction and replaced the Softmax layer with a logical classifier in the network structure. The structure of YOLOv3 is shown in Fig. 6. In the implementation of YOLOv3, we also change the backbone of YOLOv3 to EfficientNet to see if it can improve the performance on fruit detection.

EfficientNet is a newly proposed neural network consisting of multiple MBConv blocks that has both fast processing speed and high accuracy on feature extraction, as shown in Figs. 7 and 8. The core structure of EfficientNet is the MBConv (Mobile Inverted Bottleneck Convolution) block [23] which introduces the idea of Squeeze-and-Excitation Network (SENet). Similar to depth-wise separable convolution, this MBConv block first performs a 1x1 point-by-point convolution on the input and changes the output channel dimension according to the expansion ratio e.g., when the expansion ratio is 3, the channel dimension will be increased by 3 times. However, if the expansion ratio is 1, then directly omit the 1x1 point-by-point convolution and subsequent batch normalization and activation functions. Then the output result is proceeded to the $k \times k$ depth-wise convolution. If the Squeeze-and-Excitation operations are needed, they will be performed after deep

convolution and the output of the $k \times k$ depth-wise convolution is restored the original channel dimension at the end of 1x1 point-by-point convolution. Finally, the drop connect and the skip connection of input are necessary, which allow the model to have a random depth and shorten the required time for model training. Drop connect is similar to dropout but the differences between these two operations are that the constant skip is added at the beginning and end of the drop connect. It is worth noting that in EfficientNets, only when the same MBConv block recurs, will the drop connect and skip connection of input be performed, and the stride of depth convolution will also be changed to 1. Additionally, after each convolution operation in this module, batch normalization is performed and the Swish activation function called Swish has been used.

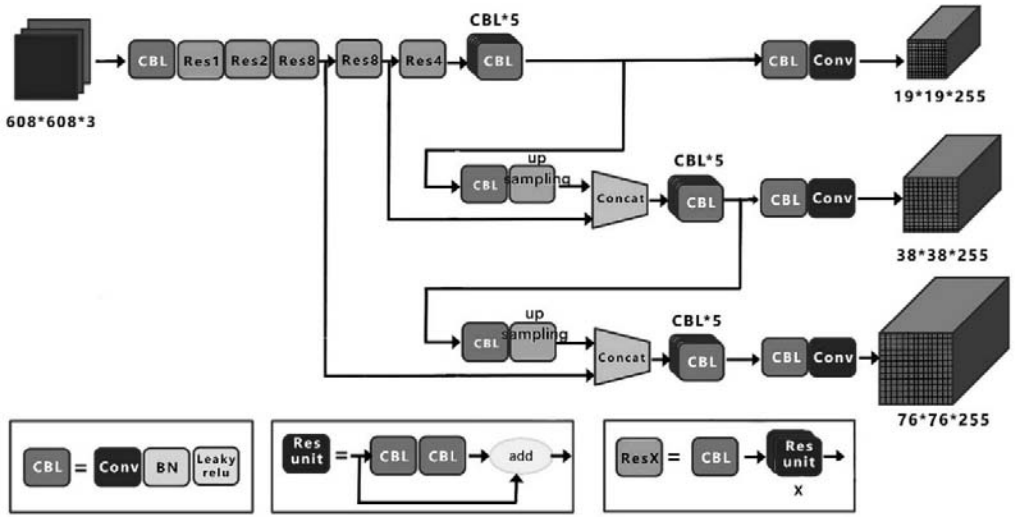

Fig. 6 The structure of YOLOv3 [26]

In Table I, the $n \times$ MBConv means the MBConv blocks has been repeatly used for n times and "6" means the expand ratio of the EfficientNet. Besides, $s1$ means the stride of MBConv equals to 1.

TABLE I
TRICKS IN YOLOV4

Type/(Expand ratio)/Stride	Filter Shape	Input Size
Conv/s1	3×3×3×32	224×224×3
MBConv/1	3×3×32×16	112×112×32
2×MBConv/6	3×3×16×24	112×112×16
2×MBConv/6	5×5×24×40	56×56×24
2×MBConv/6	3×3×40×80	28×28×40
2×MBConv/6	5×5×80×112	14×14×80
2×MBConv/6	5×5×112×160	14×14×112
2×MBConv/6	3×3×160×320	7×7×160
Conv/s1&AvgPool/s1&FC/s1	Pool 7×7	7×7×320

As shown in Fig. 7, the Se module in MBConv is an attention-based feature map operation. This module first compresses the feature map, performing global average pooling in the channel dimension direction, and obtains the global features of feature maps in the channel dimension direction. Then the global features are activated by using $C \times R \times 1 \times 1$ convolution kernels to convolve these global features, where R is the activation ratio and C is the number of global feature dimension. Through the excitation operation the relationship between each channel can be learned and the weights of different channels can be obtained by Sigmoid (an activation function). The final extracted features can be obtained by multiplying the relationships and weights by the original feature maps.

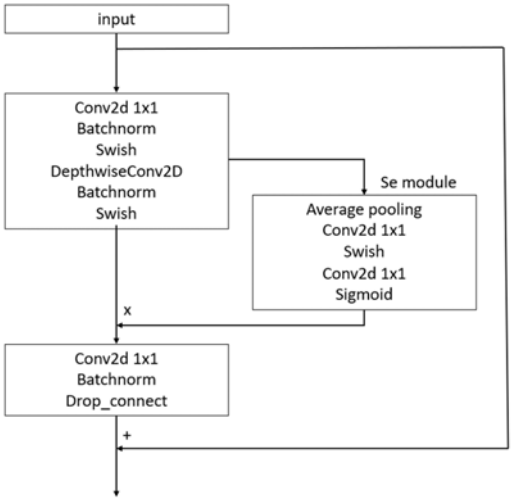

Fig. 7 The Structure of MBConv Block

Structurally, YOLOv4 uses CSPDarknet53 as the backbone, SPP and PANet as the neck. CSPDarknet53 is a combination of CSPNet (Cross Stage Partial Network) and Darknet-53. CSPNet has the advantages of richer gradient combinations and less computation. CSPDarknet53, which combines CSPNet and Darknet53, is proved to be the most suitable backbone for YOLOv4 because of its outstanding performance on detection. The primary considerations for choosing a neck are expanding the receptive field and integrating features better. After large numbers of experiments, the SPP and PANet path aggregation was finally selected as the neck of YOLOv4. The reason for

introducing SPP in the neck of YOLOv4 is that it significantly increases the receptive field and separates the most critical context features with reduction of running speed of YOLOv4. Besides, the introduction of PANet makes the extracted R.O.I. features richer, and the output can get more accurate segmentation results. In the head selection of YOLOv4, the head of YOLOv3 is proved to be the right choice to perform multi-scale prediction and improve the detection performance of targets of different sizes. The structure of YOLOv4 is shown in Fig. 8. Additionally, YOLOv4 used many tricks called 'Bag of freebies' and 'Bag of Specials'. 'Bag of freebies' are some methods which can obtain higher accuracy without increasing the reasoning cost, but only to change the training strategy or only increase the training cost. 'Bag of specials' are Plug-in modules and post-processing methods that only increase a small amount of reasoning cost but can significantly improve target detection accuracy. Table II shows the tricks used by YOLOv4.

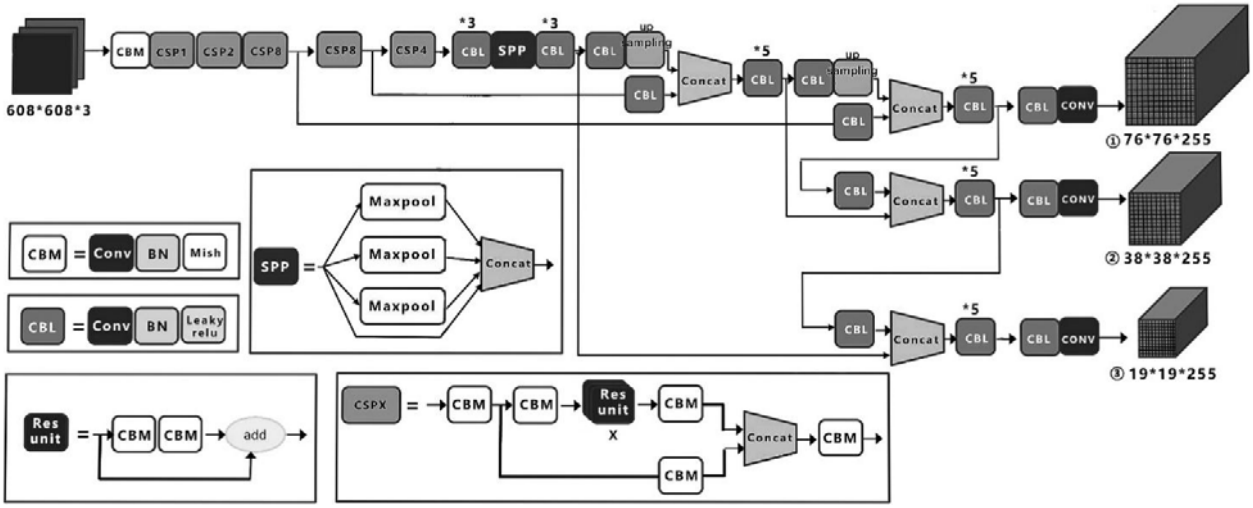

Fig. 8 YOLOv4 structure [26]

TABLE II
TRICKS IN YOLOV4

Bag of Freebies (BoF) for backbone	CuMix and Mosaic data augmentation, DropBlock regularisation, Class label smoothing
Bag of Specials (BoS) for backbone	Mish activation, Cross-stage partial connections (CSP), Multi-input weighted residual connections (MiWRC)
Bag of Freebies (BoF) for detector	CIoU-loss, CmBN, DropBlock regularisation, Mosaicdata augmentation, Self-Adversarial Training, eliminate grid sensitivity, Using multiple anchors for a single ground truth, Cosine annealing scheduler, Optimal hyper-parameters, Random training shapes
Bag of Specials (BoS) for detector	Mish activation, SPP-block, SAM-block, P.A.N. path–aggregation block, DIoU-NMS

In the implementation of YOLOv4, we also change the backbone to MobileNet which is a light-weight convolutional neural network consisting of depth-wise separable convolutions. The depth-wise convolutions aim to use factorized convolution operators to replace full convolution operators and split the convolution into two separate layers, i.e. depth-wise convolution layer and point-wise convolution layer. The structure of MobileNet is sequential (same as normal VGG) with no residual structure, as shown in Table III. MobileNet also introduces a new hyperparameter called Width Multiplier to adjust the number of channels of convolution output so as to balance the calculation speed and accuracy of the network more conveniently.

YOLOv5 includes four versions: YOLOv5s, YOLOv5m, YOLOv5l and YOLOv5x. Among them, YOLOv5s is the most popular and widely used network because the size of YOLOv5s is the smallest (usually less than 20M) and the processing speed is very fast, which is extremely suitable to embed in mobile devices. YOLOv5 uses the structure of FPN and PAN to splice the datasets by random scaling, random clipping and random arrangement. In the implementation of YOLOv5s, we focus on the letterbox function in modified datasets subfile to add the least black edges to the original image adaptively.

TABLE III
YOLOV4 MOBILENET

Type/Stride	Filter Shape	Input Size
Conv/s2	3×3×3×32	224×224×3
Conv dw/s1	3×3×32dw	112×112×32
Conv/s1	1×1×32×64	112×112×32
Conv dw/s2	3×3×64dw	112×112×64
Conv/s1	1×1×64×128	56×56×64
Conv dw/s1	3×3×128dw	56×56×128
Conv/s1	1×1×128×128	56×56×128
Conv dw/s2	3×3×128dw	14×14×112
Conv/s1	1×1×128×256	28×28×128
Conv dw/s1	3×3×256dw	28×28×256
Conv/s1	1×1×256×256	28×28×256
Conv dw/s1	3×325dw	28×28×256
Conv/s1	1×1×256×512	14×14×256
5×Conv dw/s1	3×3×325dw	14×14×512
Conv/s1	1×1×512×512	14×14×512
Conv dw/s2	3×3×512	14×14×512
Conv/s1	1×1×512×1024	7×7×512
Conv dw/s2	3×3×1024dw	7×7×1024
Conv/s1	1×1×1024×1024	7×7×1024
AvgPool/s1	Pool 7×7	7×7×1024
FC/s1	1024×1000	1×1×1024
Softmax/s1	Classifier	1×1×1000

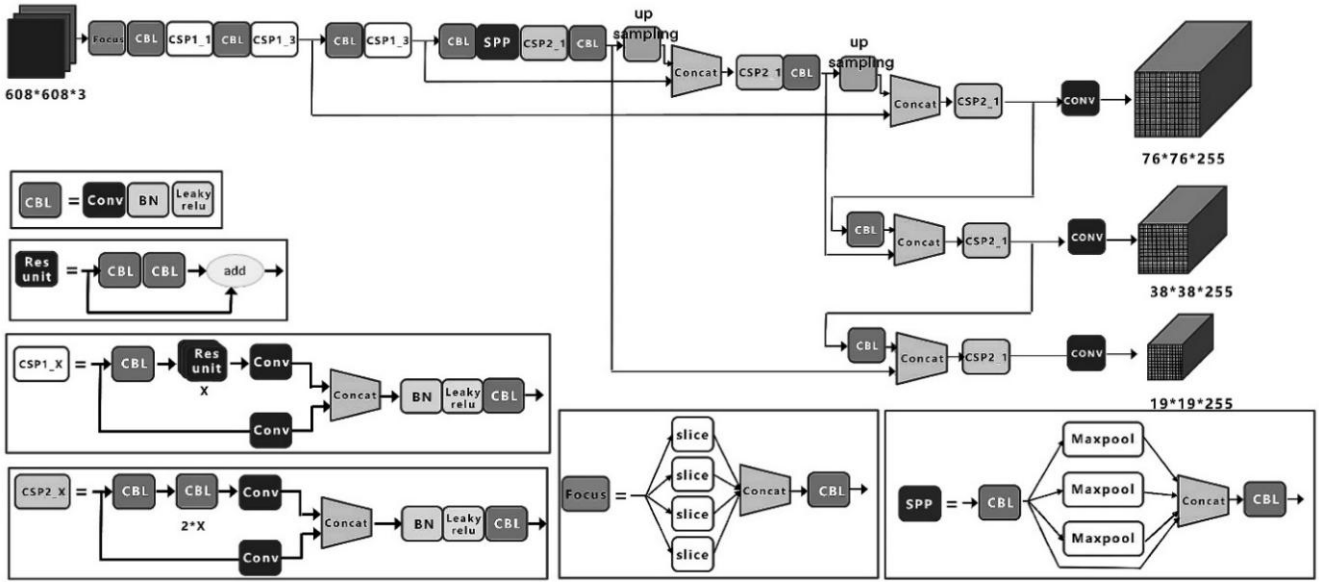

Fig. 9 YOLOv5 structure [26]

V. Implementation

A. Training Overview

We use Pycharm as our python IDE to program python and the core python packages used are pytorch, numpy, os, cv2, pandas, P.I.L., matplotlib, sys, etc.

Firstly, we use the hold-out method to split the entire dataset (550 images in total) into 3 parts, 85%, 10%, 5% for training, validation and testing, respectively. Using most of the dataset for training can better fit the model to the required weights while retaining enough test set to ensure that the model's generalisation performance is not too bad. During training, the input data are augmented through Mosaic augmentation, rotation, resize and flipping to improve variety of data. We use BCE-Loss for category metric and IoU series for bounding box regression. Different optimizers are selected for different networks. Adam optimizer with the learning rate of 0.0001 is assigned to YOLOv3 and YOLOv4. YOLOv5 uses SGD with learning rate of 0.01. For the reason that *num-classes* merely include apple and background, we adapt lower value for batch-size. Simultaneously, to increase the operation speed, batch-size is set to be 4 under this circumstance.

B. Data Preparation (Bag of Freebies)

To improve data variety and prevent overfitting, we implied pixel level data augmentation including rotate, resize, flip, random location to obtain better accuracy. Strategy of mixed augmentation using multiple graphs is abandoned in consideration of the narrow size of apples. This process helps elevate training performance without reasoning cost.

C. Learning Rate Decay

In YOLOv4, we implement Cosine Annealing Algorithm [24]. When we use gradient descent algorithm to optimise the objective function, the learning rate should be smaller to make the model as close as possible when it is closer to the global minimum of loss value, and Cosine Annealing Algorithm can reduce the learning rate through cosine function. In the cosine formula (1), with the increase of x, the cosine value first decreases slowly, then decreases rapidly, and then decreases slowly again. This decline mode can cooperate with the learning rate and produce good total_loss results in a very effective way of calculation.

$$\eta_t = \eta_{min}^i + \frac{1}{2}(A - B)\left(1 + cos\left(\frac{Tcur}{T_i}\pi\right)\right) \quad (1)$$

For YOLOv3 and YOLOv5, we implement a simple learning rate decay of 0.0005.

D. Bounding-Box Regression

Selection of loss function matters in the Bounding-Box regression task. Traditionally, Mean Square Error (MSE) and L1-loss are used as loss metrics, and M.S.E. is used for bounding box regression of YOLOv3. However, these losses change with the size of the target, which means the lack of scale invariance. Therefore, the loss of IoU type with scale invariance are also considered in this paper (IoU, GIoU, DIoU, CIoU). The disadvantage of vanilla IoU loss function is that it remains unchanged if two regions do not intersect, as is shown in (2). Also, when the centre points coincide but the aspect ratio is different, the loss remains unchanged. CIOU is introduced to overcome those problems and is implemented in YOLOv4. As shown in (2)-(5), the penalty term of CIoU is based on the penalty term of DIoU by adding an influence factor' a', 'v', which considers the aspect ratio of the prediction the aspect ratio of fitting target frame where 'a' is the parameter used to do trade-off, and 'v' is the parameter used to measure aspect ratio consistency. For YOLOv5, GIOU is utilized. GIOU is insensitive to scale, and it ranges from -1 to 1. It pays attention to both overlapped and noncoincident areas, therefore it better reflects accuracy of prediction.

$$\mathcal{L}_{IoU} = 1 - \frac{|B \cap B^{gt}|}{|B \cup B^{gt}|} \quad (2)$$

$$\mathcal{L}_{CIoU} = 1 - I_o U + \frac{\rho^2(b, b^{gt})}{c^2} + \alpha v \quad (3)$$

$$\alpha = \frac{v}{(1 - I_0 U) + v} \quad (4)$$

$$v = \frac{4}{\pi^2}\left(arctan\frac{w^{gt}}{h^{gt}} - a\frac{w}{h}\right)^2 \quad (5)$$

E. NMS (Non-Maximum Suppression)

In the post-processing process of target detection, NMS operation is usually needed to filter many target boxes. For the detection task, NMS is a necessary component, which is a post-processing algorithm to remove the redundancy of the detection results. The standard NMS is designed by hand, and greedy accepting local maximum and discarding their neighbours is based on a fixed distance threshold, that is, greedily selecting the detection results with high scores and deleting the adjacent results that exceed the threshold, so as to achieve a trade-off between recall and precision.

F. Pretrained and Transfer Learning

Before training, we take a measure called pretraining. To some extent, using pretraining enables the model to obtain more reasonable weights initially. Simultaneously, it improves the model's training speed and makes the model fit faster according to the correct gradient descent method. VOC dataset is a widely used dataset for most object detection network. Although the VOC dataset does not include apples, the settings of many parameters are relatively mature. Moreover, because the number of network layers is large and the deep layers' features are relatively close, fine-tuning transfer learning can be used to adjust the details of the network more reasonably.

By retraining based on the trained model using the VOC. dataset, the model successfully obtains better initial value, which significantly increases the converging speed of the model. Besides, through retraining with a more mature model, it can be sure to improve the robustness of the model, and better reduce the loss, improve the accuracy of the model, map and other functions.

G. Parameter Adjustment Tuning Details

Fig. 10 shows the principle we adjust the anchors. Parameter adjustment is mainly aimed at batch-size, overlap, epoch total, training of CIoU and adjustment of confidence after training, which results in the improvement of training results.

In point of object detection loss, researchers have addressed the advantages of CIoU [25]. Compared with IoU and GIoU, CIoU has better bounding box fitting effect and can improve the model's prediction and recognition ability.

Because many stacks appear while counting apples, the stack is set to be 3 to minimise image overlap and avoid double counting. Besides, through comparative experiments, it is found that overlap has better counting effect at 0.2.

The freezing training method is used to freeze every 50 epochs, and then thaw for training, increasing the operation speed and better adjusting the network to a certain extent. Then the method of dropout is used to prevent the model from overfitting. After training for about 180 times, we conclude that 150-times-epoch model is optimal because after training for 150 times, loss cannot be reduced, so the model with least training loss and less training times is chosen.

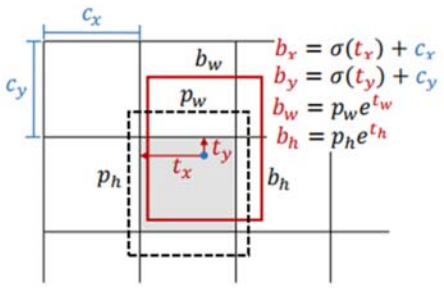

Fig. 10 Principle of anchor adjustment

VI. Results and Evaluation

A. Results Generation

For a fair comparison, a strict data partitioning of the dataset is performed for results evaluation, in which 90% is for training set and 10% for test set. Meanwhile, the dropout method is applied to generalize the model in order to prevent overfitting and falling into a local optimal solution.

Taking the size of the models into consideration, the models with the fewer parameters are eventually deployed in the series models. Through comparing the results by accuracy, the best confidence is obtained by experimental adjustment as the counting accuracy reaches highest. Then we trained the models with best confidence according to the tuning rules in the previous chapter, resulting in the following results. The comparison of the different models is shown in Table IV. The model trained on the VOC dataset is applied in each network. Meanwhile, the fine-tuning method of transfer learning is implemented on this basis to better improve the detection and technical capabilities of the models.

TABLE IV
Comparation of Different Models

	Counting best confidence	Counting accuracy	Map (confidence 0.5)	Model memory
YOLOv3 DarkNet-53	0.45	60.21%	48.51%	240693Kb
YOLOv3 EfficientNet-B0	0.55	90.02%	85.79%	61330Kb
YOLOv4 CspdarkNet53	0.64	93.44%	90.53%	250159Kb
YOLOv4 MobileNet-v1	0.32	27.7%	20.63%	52254Kb
YOLOv5s	0.4	85.60%	80.11%	57045Kb

B. Analysis

In this section, the results of all models are analysed and compared in terms of the essence of the model network.

Firstly, YOLOv3 (Darknet-53) generates mediocre results and has still relatively deep layers compared to other models, thus generating a large amount of memory shown in Table III.

Fig. 11 Detection results

By replacing the YOLOv3 backbone with an EfficientNet and using MBConv block, the number of parameters for network training is significantly reduced. Meanwhile, depth-wise convolutional is applied in YOLOv3 using 16 MBConv, which results in great improvement in the large colour variations produced by red and green apples and improves the network's ability to extract apples features as well.

For YOLOv4 family, the YOLOv4 CSPDarkNet and DarkNet53 models have the same number of layers, while the memory of YOLOv4 (CSPDarkNet) is not much larger than the YOLOv3 (DarkNet53) model. In detail, residual edges are introduced in the CSPDarknet's convolutional block in YOLOv4, which results in that more of the influence passed from the previous layer on the input quantity is retained. Meanwhile, the Cosine Annealing Algorithm and Mosaic method greatly improve the recognition filter on obscured apples due to the better learning rate. Meanwhile the implementation of the Cosine Annealing Algorithm results in better approximation for the global optimum.

CIoU algorithm is applied in YOLOv4 models to obtain more accurate bounding boxes. From the perspective of apple detection, depth-wise convolutional is applied in the MobileNet-v1 backbone. As a result, the MobileNet-v1 model has relatively minimal memory though the performance is not as appreciable as EfficientNet-Bo.

YOLOv5s is the smallest model in the YOLOv5 family and deployed in our detection project. To a certain extent, YOLOv5s is not suitable for small object detection in our applied dataset, but the structure of the model does get greatly simplified, while at the same time obtaining a respectable accuracy. Thanks to the GIoU algorithm, the model has great bounding box regression capabilities. The downside of the model is that it needs better confidence to perform the matching process. The detection results with bounding boxes examples are shown in Fig. 11.

VII. Conclusion and Future Work

A comparison of various YOLO models with different backbones is conducted on extensive apples datasets in terms of multi-conditions. This paper is a research attempt studying comparative YOLO models including YOLOv5s for apple detection and counting application. Based on the experimental results, proposed state-of-the-art YOLO models have excellent performance in counting accuracy, mAP with acceptable memory. The future work is to collaborate with agricultural workers and researchers to deploy the models in apple orchards and evaluate the performance in multi-scenarios. Meanwhile, we will focus on the models' ability of dealing with extreme cases for further improvement on yield estimation. The proposed models provide a foundation for research on the integration of DL methods, especially the improvement of YOLO family and its practical application in precision agriculture.

References

[1] Parrish, E.A. and Goksel, A.K., 1977. Pictorial pattern recognition applied to fruit harvesting. *Transactions of the ASAE*, *20*(5), pp.822-0827.

[2] Zhou, R., Damerow, L., Sun, Y. and Blanke, M.M., 2012. Using colour features of cv. 'Gala'apple fruits in an orchard in image processing to predict yield. *Precision Agriculture*, *13*(5), pp.568-580.

[3] Qian, J., Yang, X., Wu, X., Chen, M. and Wu, B., 2012. Mature apple recognition based on hybrid color space in natural scene. *Transactions of the Chinese Society of Agricultural Engineering*, *28*(17), pp.137-142

[4] Payne, A.B., Walsh, K.B., Subedi, P.P. and Jarvis, D., 2013. Estimation of mango crop yield using image analysis–segmentation method. *Computers and electronics in agriculture*, *91*, pp.57-64.

[5] Si, Y., Liu, G. and Feng, J., 2015. Location of apples in trees using stereoscopic vision. *Computers and Electronics in Agriculture*, *112*, pp.68-74.

[6] Li, D., Shen, M., Li, D. and Yu, X., 2017, August. Green apple recognition method based on the combination of texture and shape features. In *2017 IEEE International Conference on Mechatronics and Automation (ICMA)* (pp. 264-269). IEEE.

[7] Tanco, M.M., Tejera, G. and Di Martino, M., 2018. Computer Vision based System for Apple Detection in Crops. In *VISIGRAPP (4: VISAPP)* (pp. 239-249).

[8] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *arXiv preprint arXiv:1506.01497*.

[9] Bargoti, S. and Underwood, J., 2017, May. Deep fruit detection in orchards. In *2017 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 3626-3633). IEEE.

[10] Chen, S.W., Shivakumar, S.S., Dcunha, S., Das, J., Okon, E., Qu, C., Taylor, C.J. and Kumar, V., 2017. Counting apples and oranges with deep learning: A data-driven approach. *IEEE Robotics and Automation Letters*, *2*(2), pp.781-788.

[11] Kitano, B.T., Mendes, C.C., Geus, A.R., Oliveira, H.C. and Souza, J.R., 2019. Corn plant counting using deep learning and UAV images. *IEEE Geoscience and Remote Sensing Letters*.

[12] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 580-587).

[13] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on*

pattern analysis and machine intelligence, *37*(9), 1904-1916.
[14] Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision* (pp. 1440-1448).
[15] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016, October). Ssd: Single shot multibox detector. In *European conference on computer vision* (pp. 21-37). Springer, Cham.
[16] Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2117-2125).
[17] [17] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779-788).
[18] Redmon, J., & Farhadi, A. (2017). YOLO9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 7263-7271).
[19] Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.
[20] Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*.
[21] Tian, Y., Yang, G., Wang, Z., Wang, H., Li, E. and Liang, Z., 2019. Apple detection during different growth stages in orchards using the improved YOLO-V3 model. *Computers and electronics in agriculture*, *157*, pp.417-426.
[22] Häni, N., Roy, P., & Isler, V. (2020). MinneApple: a benchmark dataset for apple detection and segmentation. *IEEE Robotics and Automation Letters*, *5*(2), 852-858.
[23] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4510-4520).
[24] Loshchilov, I. and Hutter, F., 2016. Sgdr: Stochastic gradient descent with warm restarts. arXiv preprint arXiv:1608.03983.
[25] Zheng, Z., Wang, P., Liu, W., Li, J., Ye, R., & Ren, D. (2020, April). Distance-IoU loss: Faster and better learning for bounding box regression. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 34, No. 07, pp. 12993-13000).
[26] Nan Yang. Introduction to YOLO series. *https://blog.csdn.net/nan355655600/article/details/107852353*