电子科技大学
格拉斯哥学院
Glasgow College, UESTC

Final Year Project Report

Bachelor of Engineering

Deep person detection in video

Student: Zilai Wei

GUID: 2429587W

1st Supervisor: ******

2nd Supervisor: ******

2021-22

Coursework Declaration and Feedback Form

The Student should complete and sign this part

Student Name: Zilai Wei	Student GUID:2429587W
Course Code: UESTC4006P	Course Name: INDIVIDUAL PROJECT 4
Name of 1st Supervisor:	Name of 2nd Supervisor:
Title of Project: Deep person detection in video	
Declaration of Originality and Submission Information	
I affirm that this submission is all my own work in accordance with the University of Glasgow Regulations and the School of Engineering requirements Signed (Student) : Zilai Wei	
Date of Submission : 2022.04.28.	
Feedback from Lecturer to Student – to be completed by Lecturer or Demonstrator	

Grade Awarded: Feedback (as appropriate to the coursework which was assessed):	
Lecturer/Demonstrator:	Date returned to the Teaching Office:

Abstract

In recent years, the detection technology is no longer limited to static image detection, but upgraded to video that can achieve real-time detection, and it has already widely used in many areas, such as monitoring system, human-computer interaction. Besides, in order to improve the precision and increase the detection speed, the deep learning algorithms are gradually replacing the tradition detection algorithms. YOLOv5, as a general object detection algorithm, has a great performance in real-time video detection. However, for person detection, due to the imbalanced face samples with different sizes and a lot of small faces in the dataset, YOLOv5 algorithm is often limited to achieve the expected effect.

This paper will discuss how to apply YOLOv5 network to data pre-processing, model design and training, and finally realize the effect of real-time tracking the person in the video. Besides, two main improvement methods are introduced, including data enhancement to balance the samples in the dataset, and adding a detect layer in order to ensure the good detection effect for small objects. The research results shows that the mAP of the improved YOLOv5 model increases from about 94% of the original model to 98%, while the precision and recall improved about 3% to 96% and about 7% to 97% respectively. Compared with the original YOLOv5 model, the improved model has better object detection precision, and enhances the capacity of small object detection, while maintains the speed to meet the real-time detection requirement.

Keywords: Person detection, Real-time detection, Deep learning, YOLOv5

Acknowledgements

On the completion of my paper, I would like to express my deepest gratitude to my first and second supervisors. During my research on the final projects and papers, they gave me careful guidance, full trust and a wide space for development, and it was one of the fortunate gains of my university stage. Their rigorous and realistic academic style, meticulous working attitude, optimistic and positive spirit will inspire me to make progress in my future study and work. Besides, I would like to express my sincere appreciation for all of you, my teachers, relatives, classmates, lovely girlfriend and all my friends. It is your sincere love, encouragement and support, which will enable me to complete my studies and graduate successfully. I will put all of my efforts and bring the best of myself to make further progress in the future. Once again, I would like to thank all those who have helped me and supported me.

Contents

1 Introduction

The project is going to achieve real-time person detection in video based on deep learning methods. This section will introduce the background of object detection, traditional detection methods and deep learning detection methods.

1.1 Object Detection Background and Significance

Object detection aims to find out all the objects we are interested in from an image or video, and determine their categories and positions. It is a part of artificial intelligence, more specifically, it has become a challengeable problem and the core of computer vision.

Objection detection can be divided into two main types: instance detection and category detection. The former aimes to recognize the instances of a specific target, such as a specfic person, the unique Big Ben and so on, while the latter aims to detect different instances of predefined target categories, such as person, car, dog. Compared with the instance detecion, the category detection is more complex and more widely used.

Many years ago, researches about object detection main focused on individual class, like person, cat, dog and other specific categories. In recent years, scientists were devoted to building the universal object detection system so that the scope of object detection can be comparable to that of human beings.

Almost every object detection project needs to solve four problems:

- **Classification problem** – The detection system needs to recognize which category of the target in the picture (or an area) belongs to. This is the core of object detection.
- **Localization problem** – The detection system should have the ability to locate the coordinate of the objects in the image for they may appear anywhere.
- **Size problem** – The system has to detect the objects with various sizes.
- **Shape problem** – The system has to detect the objects with various shapes.

In fact, object detection is a combination of classification and regression problems. The paper will introduce some algorithm to achieve these tasks later.

In recent years, the technology of object detection has widely used in many areas of artificial intelligence and information technology, including robot vision, security, automatic driving, human-computer interaction, intelligent video surveillance [1] and so on. In other words, it is closely related to people's lives.

1.2 Traditional Object Detection Methods

Many years ago, scientists tried to using some traditional algorithms to extract the features of the object in order to achieve object detection. These traditional algorithms have three main similar operation steps, which can be summarized as candidate window extraction, feature extraction and classifier determination. The three steps are introduced in detail as followed.

- **Candidate window extraction** – Sliding window method is commonly used. The system will use some classical algorithms in computer vision pattern recognition to extract the local information in each window, including the methods based on colour, texture and shape, and some medium or high-level semantic features methods as well.
- **Feature extraction** – There are three levels of feature in computer vision according to the dimension of feature extraction.

  1) **Low-level features**: The most basic features of the images, such as colour and texture.
  2) **Middle-level features**: These features are gotten based on the low-level features, and are extracted after the feature learning process through machine learning methods, including PCA features or the features after LDA learning.
  3) **High-level features**: These features are gotten through further excavate the low-level and middle-level features. For example, in person detection, high-level features can be defined as the semantic features like whether wear a hat or glasses.

  Object detection algorithms are commonly concentrated in low-level and middle-level, in other words, the features are extracted by manual.
- **Classifier determination** – The features extracted from the above step in the candidate region will be classified and determined, where the classifier needs to be learned and trained in advance. For single category object detection, the classifier only needs to distinguish whether the target contained in the current window or not in this process, while for multi-classification problems, it is necessary to further distinguish the classes of the objects.

There are some classical object detection algorithms like Viola-Jones, HOG+SVM and DPM. Although some of the algorithms can achieve a relatively high precision, there are some disadvantages in common, for example, the region selection method based on sliding window has no pertinence and will easily results in computation complex and window redundancy [2].

1.3 Deep Learning Object Detection Methods

In the past decade, researchers gradually tried to using deep learning methods to replace the traditional algorithms in object detection [3]. The concept of deep learning proposed from the research of artificial neural network, but it is different to traditional neural network [4]. However, many deep learning algorithms will include the word "neural network", such as Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN). Therefore, deep learning can be seen as an upgrade on the basis of traditional neural network [5].

Deep learning is the most important branch of machine learning. Although they are very similar in data preparation and pre-processing, compared with the traditional machine learning, deep learning requires designing models and training instead of extracting the features and selecting the classifier in traditional methods. In other word, the feature extraction is extracted automatically by machine in deep learning instead of manual extraction.

In the development of object detection, the year 2012 can be seen as a watershed, for the famous AlexNet [6], which is a deep convolution neural network (DCNN) proposed by krizhevsky, achieved a record in image classification accuracy in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC). Since then, many computer vision applications have focused on the deep learning methods, for it has several advantages:

- **Great learning ability** – From many project results, the deep learning algorithms usually have strong learning ability in classification and performed well.
- **Good adaptability** – Deep learning neural network has many layers and wide width. Therefore, it can be mapped to any function in theory to solve many complex problems.
- **High upper limit** – Deep learning is highly dependent on data. Its performance can be improved through increasing the amount of data, and has even exceeded human in some projects such as image recognition, person detection.
- **Good portability** – There are many frameworks that can be used for deep learning model design, such as TensorFlow and Pytorch, and they can be compatible with many platforms.

These years, there are two main methods for object detection based on deep learning: Two-stage and One-stage [7].

Two-stage detection networks are proposed earlier. The detection process is mainly completed through a convolution neural network. It trains the Region Proposal Network (RPN) and the detection network in target area [8], therefore it has two detection steps.

In 2014, Ross Girshick proposed the R-CNN (Regions with CNN features) [9], which is the first object detection algorithm based on CNN. It replaces the manual feature extraction part of the traditional algorithm with CNN extraction to improve detection precision. Due to the slow exhaustion of candidate regions in R-CNN, Girshick et al. then proposed Fast R-CNN [10] and Faster R-CNN [11], in which a RPN is used instead of the original selective search algorithm, and the generated RPNs are then used to generate a new region. Although the two-stage method has high precision, the detection speed is relatively slow. Taking Faster R-CNN as an example, although it can achieve 70.4% average accuracy on Pascal VOC 2012, its inference speed on GPU is only 5 FPS.

One-stage algorithms directly regress the category probability and position coordinate value of the object (without region proposal). In one-stage method, these values are given directly through the backbone network instead of the RPN network. This algorithm has faster speed and slightly lower precision compared to the two-stage detection network. The typical examples of these methods are YOLO, SSD, Retina-Net and so on.

In 2016, the YOLO (You Only Look Once) [12] algorithm proposed by Redmon was the first one stage algorithm. The main advantage is that it only uses one network to complete object detection, and the operation speed is very fast while ensuring the detection precision. For instance, in Pascal VOC 2012 detection task, the FPS of the YOLO network reaches 45 FPS, and the average precision reaches 63.4%, which is only slightly lower than that of Faster R-CNN. The main idea of YOLO is to convert the object detection problem into a regression problem by dividing the image into multiple grids, with each grid predicting a vector containing information on the coordinates of the bounding box, the confidence level and the conditional probability of each class. In 2017, Redmon proposed YOLOv2 [13] network, which adding the priori anchor frame mechanism in Faster R-CNN and global level pooling from Network in Network (NIN) to YOLOv1 network, so that it no longer directly predicts the object coordinates, but predicts the deviation between the object and the anchor frame, which reduces the difficulty of training and improves the recall rate of the network. The next generation YOLOv3 [14] was proposed in 2018, for it adds the feature pyramid and residual block structure to increase the detection effect of small objects. After the release of YOLOv3, Redmon, the original author of the YOLO series from one to three generations, announced his decision to retire from the field of computer vision because he could not ignore the negative impact of his research. Fortunately, in 2020, Alexey Bochkovskiy, a participant in the YOLO community, took over the YOLO project and published YOLOv4 [15]. In YOLOv4, the author introduces a

lot of training skills and structural improvements, such as CSP structure, CIoU loss, Mish activation function and so on, which greatly improves the performance of the network. Shortly after the birth of YOLOv4, Ultralytics released its own version of YOLO network and named it YOLOv5. Compared with YOLOv4, YOLOv5 has little improved, for it adds mosaic data enhancement, focus module, automatic anchor box calculation, etc. In summary, the operation speed of YOLOv5 is faster while ensures the precision of the model. Besides, the weight of the model is smaller than that of YOLOv4, which is benefit for calculation and detection speed.

1.4 Content and Structure

As mentioned above, for the person detection project, the precision of traditional methods is not very well, and the speed of the R-CNN series algorithms cannot meet the requirements although they have high precision. Therefore, this paper uses YOLOv5 algorithm to train, get the weight for person, and achieve the real-time person detection in video. Besides, for the dataset and model analysis, some improvement methods are proposed for the YOLOv5 model, and the mean average precision, precision, recall, loss value and confidence interval are used as the evaluation metrics for the model.

The main content for each chapter is presented:

Chapter 1 Introduction to the background and importance of person detection. Some traditional methods and deep learning methods are briefly introduced.

Chapter 2 Introduction to the principles of the YOLO algorithm. It introduces the basics of convolutional neural networks, the structure and algorithmic flow of YOLOv5 networks, and the evaluation metrics of the algorithm.

Chapter 3: Analysis and improvement of the YOLOv5 model. It analyses the person detection dataset and the original YOLOv5 model. The improvement methods to overcome the shortcomings of the original YOLOv5 model are also proposed.

Chapter 4: Experimental analysis. This chapter presents the results of the experiments and a comparison between the original and improved YOLOv5 detection results.

Chapter 5: Summary and expectation. The strengths and weaknesses of the improved YOLOv5 model are summarised, and future directions for improvement are also introduced.

2 YOLO Network Architecture and Principle

YOLO is a kind of CNN, which is able to extract features much better and faster than traditional neural networks and can also reduce model parameters. This chapter first introduces the components and principles of CNN, then it analyses the network structure of YOLOv5 and introduces its detection process. Finally, the evaluation index of the model is introduced.

2.1 CNN Introduction

Although the traditional fully connected neural network has good performance in some simple tasks, its fully connected layer will lead to excessive parameters. This drawback is more evident in image processing. For example, the size of the YOLOv5 model will be scaled to 640×640×3, and the weight of only one neuron in the input layer will be 640×640×3=1228800 if using a fully connected network, which is about 5MB in the form of float and it is obviously a huge amount of work. However, let the input channel of a VGG network is 3, and the output channel is 96, the size of convolution kernel is 3×3 as an example, the number of weights is 3×3×3×96=2592, which is significantly less than the fully connected network. Due to the weight sharing of convolution calculation and the characteristics of local receptive field, the number of parameters of CNN is far less than that of traditional network.

CNN usually consist of an input layer, convolutional layers, activation function, pooling layers, fully connected layers, and an output layer [16]. These parts will be explained in the following subsections.

2.1.1 Convolution Layer

The “convolution layer” comes from the convolution operation in digital signal processing. The convolution formula of two-dimensional is shown in Equation 2.1:

$$y(m,n) = x(m,n) * h(m,n) = \sum_i^{\infty} \sum_j^{\infty} x(i,j)h(m-i,n-j) \tag{2.1}$$

Convolution layer is the basic module of CNN, and its main function is to extract the input features. The convolutional layer is composed of multiple feature maps, unlike fully connected, each neuron in the feature map is locally connected through the previous layer of the convolutional kernel. The convolution kernel is a weight matrix with the size of N, and its weight is learned by itself during the training process. In order to ensure that the convolution kernel has a central point and facilitate the filling of the feature map, N is usually an odd number. The operation of convolution is to use a sliding window to move on the image with a certain step size, and weighted sum the features in the window to obtain a new output. It is

generally believed that the closer the convolution layer in the CNN structure is to the output, the more abstract and higher-level features can be extracted than artificially designed filters.

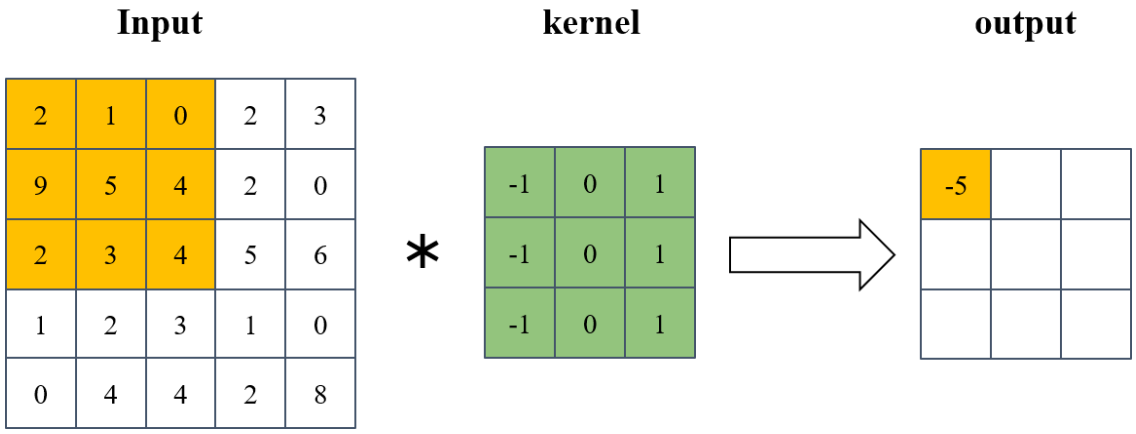


Figure 1. The process of convolution in feature map

The Fig. 1 shows the calculation steps of a convolution kernel. *Input* shows an input feature map, and the number represents the value of pixel points, where the yellow part indicates the region that the convolution operation will do. *Kernel* shows a convolution kernel with the size 3×3, where the green part represents the weight. The *output* on the right side of the figure represents the output after convolution operation, and the yellow area represents the operation structure of convolution. The convolution process is shown in Equation 2.2.

$$2 \times (-1) + 1 \times 0 + 0 \times 1 + 9 \times (-1) + 5 \times 0 + 4 \times 1 + 2 \times (-1) + 3 \times 0 + 4 \times 1 = -5 \tag{2.2}$$

The size of the output feature map may does not matched the input after convolution, so it is necessary to fill the zeroes around the input feature map. The relationship between the length and width of the input and output after convolution operation is presented in Equation 2.3.

$$w' = \frac{w-n+2p}{s} + 1 \tag{2.3}$$

In Equation 2.3, w' and w is the width of the output and input receptively. n is the convolution kernel size, p is the filling padding, that is, the size of the zero filling of the input, and s is the sliding step of the convolution kernel.

There are several advantages of using convolutional layers in neural networks:

- **Reduce parameters** – The parameters within the convolutional kernel are fixed when the convolutional operation is performed on different regions of an input. Therefore, it can be seen as sharing weights and biases, thus reducing the number of parameters of the network.

- **Stronger abstract ability** – Compared with the artificial designed convolutional filters, the convolutional filters based on deep learning can learn the weights by themselves during training and can extract the high-dimensional features.

2.1.2 Activation Function

The convolutional, pooling, and fully connected layers can all be considered as linear weighted summations of the inputs. When multiple convolutional, pooling, and fully connected layers are connected, they can all be considered as linear combinations of the inputs, so that no matter how deep the network is, it is no different from a single linear combination. Therefore, some non-linear activation functions need to be added to the network to avoid the network being a mere linear combination. The general activation functions are sigmoid, ReLU, Mish, etc.

1) Sigmoid

Sigmoid function is a common S-shaped function, and it has been widely used as activation functions for artificial neurons. The formula of this function is shown in Equation 2.4.

$$f(x) = \frac{1}{1+e^{-x}} \tag{2.4}$$

The advantage of sigmoid is that it converts the output to a real number between 0 and 1. However, the mean of the output is not 0, which makes the network converge slowly. In certain cases, some gradient disappearance and gradient explosion problems may happen.

2) Tanh

Tanh is one of the hyperbolic functions and tanh() is the hyperbolic tangent. The formula of this function is shown in Equation 2.5.

$$f(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \tag{2.5}$$

The tanh can solve the problem of mean shift of the output of sigmoid, however, there is still a problem of gradient disappearance.

3) ReLU [17]

The ReLU activation function is used for the output of neurons in the hidden layer. The formula of this function is shown in Equation 2.6.

$$f(x) = max(0, x) \tag{2.6}$$

The ReLU can solve the gradient disappearance problem in the positive interval. In addition, the ReLU activation function has a dead zone, for instance, when the input is in the negative interval, the neuron output is 0, resulting in the failure to update the neuron

weight parameters of the previous layer during backpropagation. In general, ReLU function used in many deep networks.

4) Leaky ReLU [18]

The formula of this function is shown in Equation 2.7.

$$f(x) = max(ax, x) \tag{2.7}$$

In Equation 2.7, 'a' can be adjusted in different situations. Generally, 'a' is taken as 0.01. The advantage of Leaky ReLU is that there is still a small amount of output in the negative interval, so that the neuron weights of the previous layer can still be updated when the input is in the negative interval.

5) Mish

Mish is the newly proposed activation function. The formula of this function is shown in Equation 2.8.

$$f(x) = x \times \tanh(\ln(1 + e^x)) \tag{2.8}$$

The advantage of Mish activation function is smoother with better generalization and summarization ability, but it has more complicated calculation compared to ReLU.

2.1.3 Pooling Layer

The pooling layer, also called the downsampling layer, usually used after one or several convolutional layers. The function of the pooling layer is to compress the data and parameters and can reduce overfitting. For an image, the pooling layer is used to remove redundant information and compress the image.

The adjacent pixels in the feature map of convolution output are almost the same, and the pixels change significantly only at the edge of the target. If these redundant parts can be removed, the amount of network parameters can be significantly reduced.

Pooling can be broadly classified into maximum pooling and mean pooling according to the calculation method. The calculation method of pooling is similar to that of convolution. In maximum pooling, for example, the pooling layer uses a sliding window to slide over the feature map in certain steps, and finds the maximum value of the pixels in each window as the output. However, unlike convolution, the pooling layer does not contain any parameters and therefore does not affect the model size.

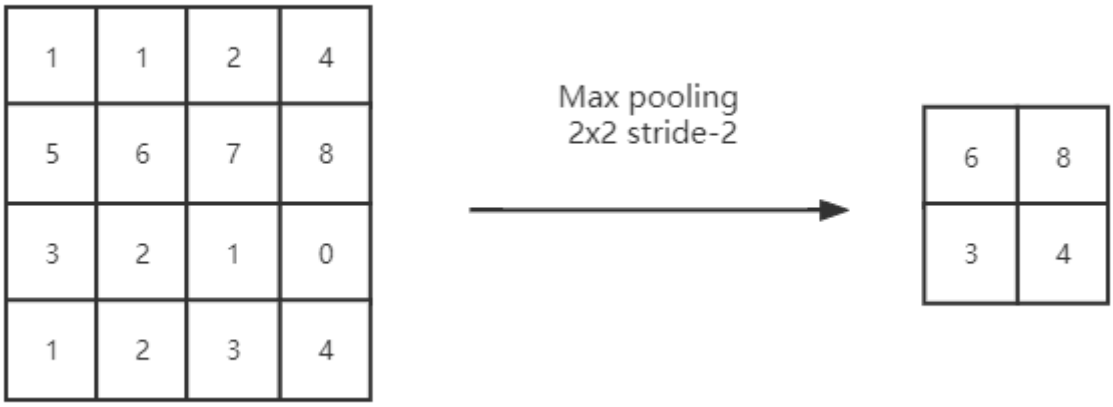


Figure 2. The process of a maximum pooling in feature map

The Fig. 2 shows the operation of a maximum pooling process. A maximum pooling layer is usually with a size of 2×2 and 2 steps. The input of the maximum pooling layer is defined on the left side, and it is divided into four parts with the size of 2x2, which are top left, top right, bottom left and bottom right, and each part represents the area covered by the pooled window. The output of the maximum pooling layer is shown on the right side.

The advantages of using pooling layers in neural networks are it can not only remove the redundant information in the feature map to reduce the parameters and computational effort, but also increase the robustness of the network.

2.1.4 Fully Connected Layer

At the end of the convolutional neural network structure, one or more fully connected layers are usually used to achieve a "classifier". The structure of it is quite similar to a traditional multilayer perceptron, where each neuron is connected with every neuron in the last layer, and thus can also be seen as an integration of the local information extracted from the convolutional layer. To enhance the fitting ability of the fully connected layer, a nonlinear activation function is usually added after each neuron to avoid the network being simply fitted linearly.

Although fully connected layers have a good fitting effect, they are prone to overfitting problems during the training period of neural networks. To solve this problem, it is usually necessary to add a dropout layer during the training period so that some neuron nodes are randomly hidden during training to prevent overfitting. In addition, the fully connected layer requires more parameters because each neuron is fully connected to other neurons in the preceding and following layers, which causes larger size of model. Therefore, a global average pooling layer is proposed to take place of the fully connected layer, which meanings that each

output of the previous convolutional layer is averaged. The pooling layer has no parameters to be optimized, so it not only greatly reduces the parameters of the network, but also avoids overfitting of the model.

2.2 YOLO Network Introduction

The two-stage algorithm, represented by the R-CNN family of algorithms, implements target detection in multiple steps. It first finding the candidate frames that may contain objects, and then performing coordinate regression and target classification on the candidate frames. The step of finding candidate frames is time consuming due to the large number of searches required. Therefore, although the R-CNN series algorithm has high accuracy, its detection speed is slow. As a representative of the one-stage algorithm, YOLO series algorithm adopts an end-to-end idea, directly combining the candidate frame prediction and classification in one step, and converting the object detection problem to regression problem. Finally, the mean average accuracy of detection results slightly decreased, but the detection speed is greatly improved.

2.2.1 YOLOv5 network structure

The network structure of the YOLOv5s model is presented in Fig. 3, where the three squares in the upper of the backbone part represent the three input channels for an image, with a fixed size of 640×640×3 in YOLOv5. The column on the left is the backbone of the YOLOv5 network, which main focus on the feature extraction from the input. The column on the right is the output part of YOLO, also known as the head of the YOLO algorithm, its main role is to compute the output of the network. The column in the lower middle is the neck part between the network skeleton and the head, which has been added to the object detection algorithm in recent years, and its role is to fuse features at different scales, usually consisting of a feature pyramid or a Path Aggregation Network (PAN). The column at the top includes the specific structures for different modules.

In YOLOv5, the PAN is used, that is, a layer of top-down feature fusion path is added to the original bottom-up feature pyramid. In the backbone part, a 640×640×3 input means the input image is 640×640 RGB colour image in length and width. In the four-parameter modules such as Focus, CBL, and Detect, each parameter refers to the number of input and output channels, kernel size and step size respectively. For two-parameter modules, such as C3, the two parameters refer to the number of input and output channels, respectively. Single-parameter modules such as SPP, Concat, Upsample, the parameter represent the number of output channels.

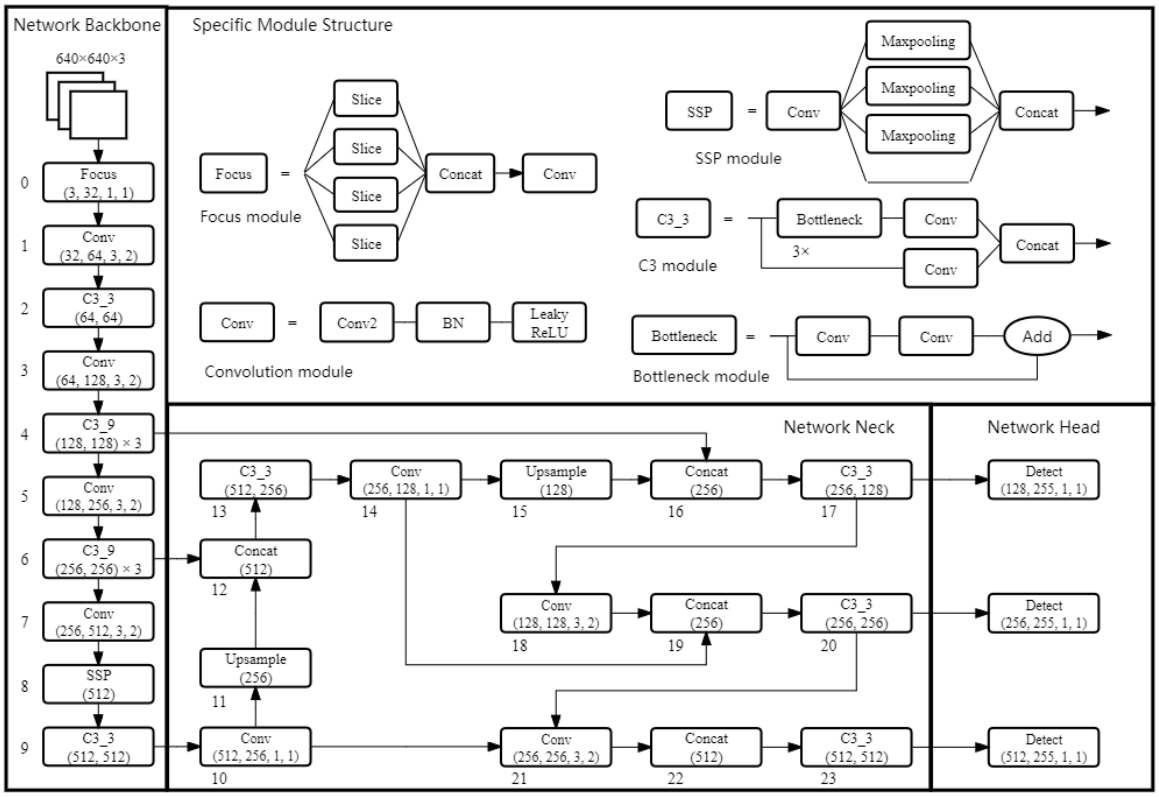


Figure 3. YOLOv5s network structure

The 1st, 3rd, 5th, 7th, 10th, 14th, 18th, 21st layers in Fig. 3 is the basic convolutional unit CBL module in YOLOv5. In CBL, C refers to the convolutional layer, which is computed as described in the previous section. B stands for Batch Normalization (BN), which serves to normalize the output of the convolutional layers within a batch to a range of zero mean and one variance, to regularize the network. It can not only allow the network to avoid overfitting, but also helps the network to reach the best point faster to reduce the training time. L refers to the Leaky ReLU activation function, which serves to introduce a nonlinear function to enhance the representational ability of the network. Using CBL module alone in YOLOv5 network is aiming to complete the downsampling operation of the network.

The first module in the backbone part in Fig. 3 is the focus module. Its method of is to slice the input into quarters by taking a value for every pixel, and then stitching the four images together in the channel dimension. This is equivalent to downsample the image without any information loss, while reducing the parameters in the network.

The 2nd, 4th, 6th, 9th, 17th, 20th, 23rd layers in Fig. 3 is the C3 module. The parameter (128, 128) × 3 in C3 means that the number of input and output channels are both 128, and the C3 module contains 3 residual units, while (64, 64) means that the number of channels in both input and

output modules is 64, and only 1 residual unit exists. The residual block structure is proposed in the residual network, and its structure is to add another way to connect the input and output of the network directly, which solves the problem that the traditional CNN degrades with the increase of network depth. Besides, using the residual structure also helps to solve the gradient disappearance and gradient explosion problems in deep neural networks. C3 module is an improvement of the cross stage partial connections (CSP) structure of YOLOv5. CSP module separates the feature map to two parts for calculation so that the gradient can propagate through different paths to obtain more gradient combinations. In addition, because only half of the feature maps in the CSP structure enter the residual unit for computation, the computation amount of the network is significantly reduced. Compared with YOLOv4, C3 module in YOLOv5 has one less convolution layer, which makes its calculation easier.

The 8^{th} layer at the backbone part in Fig. 3 shows the Spatial Pyramid Pooling (SPP) layer. SPP layer uses layers with different sizes and steps of maximum pooling to compress the images and stitch the number of channels, which improves the receptive field of the network.

The 11^{th}, 15^{th} layers at the neck part in Fig. 3 is the upsample module, and its main function is to increase the dimension. In YOLOv5, when the feature map is transmitted in the network, upsampling applies the method of filling adjacent values to double its dimension, therefore it can match the dimension of that in the 4^{th} and 6^{th} layer and can be concatenated.

The 12^{th}, 16^{th}, 19^{th}, 22^{nd} layers in Fig. 3 is the concat layer. In YOLOv5, its function is to splice the feature map according to the channels, and therefore requires that of input to be of the same dimension.

The module at the head part in Fig. 3 is the detect layer, which is responsible for the prediction of the input feature maps. In YOLOv5, three feature maps are responsible for outputting the prediction results. The large-sized feature map is 80×80, medium-sized feature map is 40×40, and small-sized feature map is 20×20, which is mainly used for predicting small objects, medium objects, and large objects, respectively.

2.2.2 YOLO Testing Process

The YOLO algorithm performs k-means clustering on all the bounding boxes in the dataset labels in advance and obtains the centre of mass of the clusters as the anchor. The core idea is to divide the image into S×S grids, which is reflected in the output feature map. It is their responsibility for each grid in predicting objects whose centres fall in it. Each grid predicts B bounding boxes, and each bounding box contains the deviation of the target relative to the grid

centre coordinates as well as the deviation between the length and width of the bounding box and the confidence of the bounding box.

The confidence is calculated as shown in Equation 2.9.

$$confidence = P_r(Object) \times IoU_{pred}^{truth} \tag{2.9}$$

It reflects the probability that the bounding box contains an object and the degree of overlap between it and the true value. In addition, each bounding box is accompanied by the conditional probability that the object belongs to category C, encoded in one-hot form. Thus, the final output of YOLO is shaped as shown in Equation 2.10.

$$filters = S \times S \times (B \times 5 + C) \tag{2.10}$$

In order to improve the poor performance of the small-sized targets by YOLOv5 model, the feature pyramid structure and path aggregation network structure are used for reference to fuse the features at different scales. Compared with previous versions of YOLO that used only one feature map for prediction, YOLOv5 uses three feature maps to predict small, medium and large sized targets, which greatly improves the performance of small target detection.

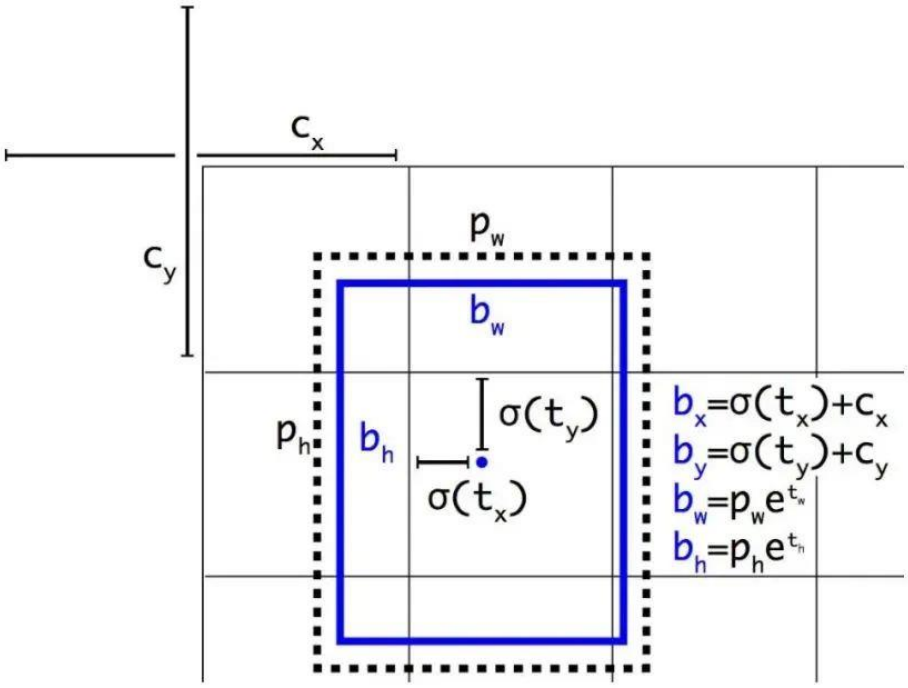


Figure 4. Anchor frame decoding

Since the coordinate part of the YOLOv5 output is about the offset of the anchor box, this part of the data needs to be decoded. In Fig. 4, c_x and c_y refer to the centre coordinates of the grid, p_w and p_h refer to the length and width of the anchor frame, t_x and t_y denote the deviation of the prediction relative to the centre of the grid, t_w and t_h denote the deviation of the prediction

relative to the anchor frame. σ refers to the sigmoid function, and b_x, b_y, b_w, b_h are centre coordinates, length, width of the actual prediction frame.

The prediction result of YOLO algorithm will have multiple redundant detection frames. Therefore, YOLOv5 uses Non-Maximum Suppression (NMS) to select the edge-joined box that is closest to the actual target and remove the redundant boxes.

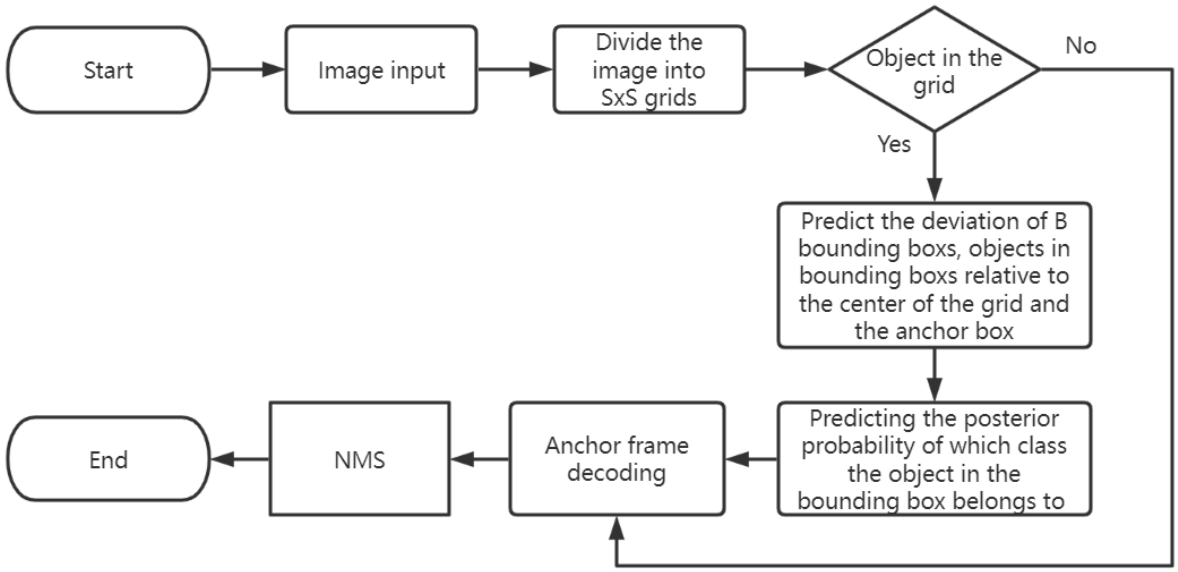


Figure 5. YOLO algorithm detection process

The calculation process is as follows.

1) Sort the bounding boxes according to the confidence level, and set an IoU threshold and an NMS threshold.
2) Remove all boxes with confidence less than the NMS threshold, the lower the confidence means the higher the probability of detection error.
3) Take the prediction box with the highest confidence level and delete the prediction boxes with IoU greater than the threshold [19]. The larger the IoU is, the more overlapping parts, and the prediction boxes with higher overlap in the image may predict the same object, so the redundant boxes should be deleted.
4) Repeat the above operation for the remaining unprocessed boxes until all boxes are processed.

The overall detection process of the YOLO algorithm is represented in Fig. 5. The deviation is first predicted, then the anchor frame is decoded, and finally NMS processing is performed.

2.3 YOLO Evaluation Index

The same evaluation index is often used to evaluate the detection effect of a model [20]. In object detection algorithm, the following indicators are usually used:

- **Intersection over Union (IoU)** – It is defined in Equation 2.11. The predicted and actual bounding boxes has an intersection and a merged area, an IoU is the rate of the two parameters. Besides, it is an indicator to measure the coincidence degree between the prediction and label boxes. The closer it is to 1, the better prediction result it achieves.

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union} \tag{2.11}$$

- **Confusion Matrix [21]** – The definition of some parameters in confusion matrix are presented in table 1. It is an important indicator for assessing the performance of the model and is the basis for many subsequent derived indicators.

Table 1. Neural network confusion matrix

Actual / Predict	0 (Negative)	1 (Positive)
0 (Negative)	TN (True Negative)	FN (False Negative)
1 (Positive)	FP (False Positive)	TP (True Positive)

- **Precision** – The definition of precision is shown in Equation 2.12, and the parameters are come from table 1.

$$Precision = \frac{TP}{TP+FP} \tag{2.12}$$

A confidence threshold is usually used to determine TP and FP. If the confidence level is greater than the threshold, it is TP, otherwise it is FP. The precision represents the proportion of all positive samples that are correctly predicted.

- **Recall** – The definition of recall is shown in Equation 2.13, and the parameters are come from table 1.

$$Recall = \frac{TP}{TP+FN} \tag{2.13}$$

Different from the precision, recall is for our original sample. It aims for the positive sample, and indicates the rate that the correct sample are predicted.

- **Average Precision (AP)** – Generally, the higher precision and recall, the better the detection results. However, the two indexes usually changed in opposite directions. Therefore, the PASCAL VOC contestant proposes a new metric AP, aims to describe the detection algorithm effect better. The different precision and recall rates can be obtained through setting different confidence thresholds, and finally, it can be plotted as a P-R curve, the horizontal coordinate is the recall and the vertical coordinate is the precision. The area between the curve and the coordinate axis is the AP. The formula for calculating AP is shown in Equation 2.14, where $p(r)$ is the precision corresponding to the recall r.

$$AP = \int_0^1 p(r)\, dr \tag{2.14}$$

  AP is widely used in object detection for its perfect description of the detection effect.

- **Mean Average Precision (mAP)** [22] – The AP is calculated for a particular category, and mAP is the average summed APs of each category. Formula for calculating mAP is shown in Equation 2.15, where N is the number of detection categories. For one category detection, mAP has the same meaning as the AP.

$$mAP = \frac{1}{N}\sum_i^N AP_i \tag{2.15}$$

  mAP can also be calculated based on the individual IoU. Generally, IoU is chosen to be 50% or 75%, which is recorded as mAP_{50} or mAP_{75}. For instance, mAP_{50} refers to the mAP of considering only the samples with IoU greater than 0.5. In this project, the detection category is only one which is the person, so that the mAP and AP are same in this situation.

3 YOLO Algorithm Analysis and Improvement

Although YOLO is a general algorithm with good detection results, it requires some adjustments for specific datasets and object tasks. In this chapter, we analyse the dataset and the YOLOv5 model, and then propose several improvements.

3.1 Dataset Analysis

Before starting to train the model, a suitable dataset is required to collect for person detection. In order to better match the project for person detection and achieve better training effect, we decide to establish our own dataset. The images in the dataset are main come from our daily lives, including our teachers, classmates, friends from primary school to university, our family members and some other people. The people in the sample cover all age groups, gender, different sizes, angles and situations, and the dataset contains more than 1500 images.

After collecting the images and establishing our own dataset, we need to build the labels that contain the coordinate information of people in the image. Therefore, a tool called Labelimg is needed to assist us to finish the work. We need to mark all the faces in these 1500 images manually at first, then the Labelimg can automatically save all coordinates of the faces we marked as a label, and each picture has its corresponding label. The Fig. 6 shows the face marking process for one image.

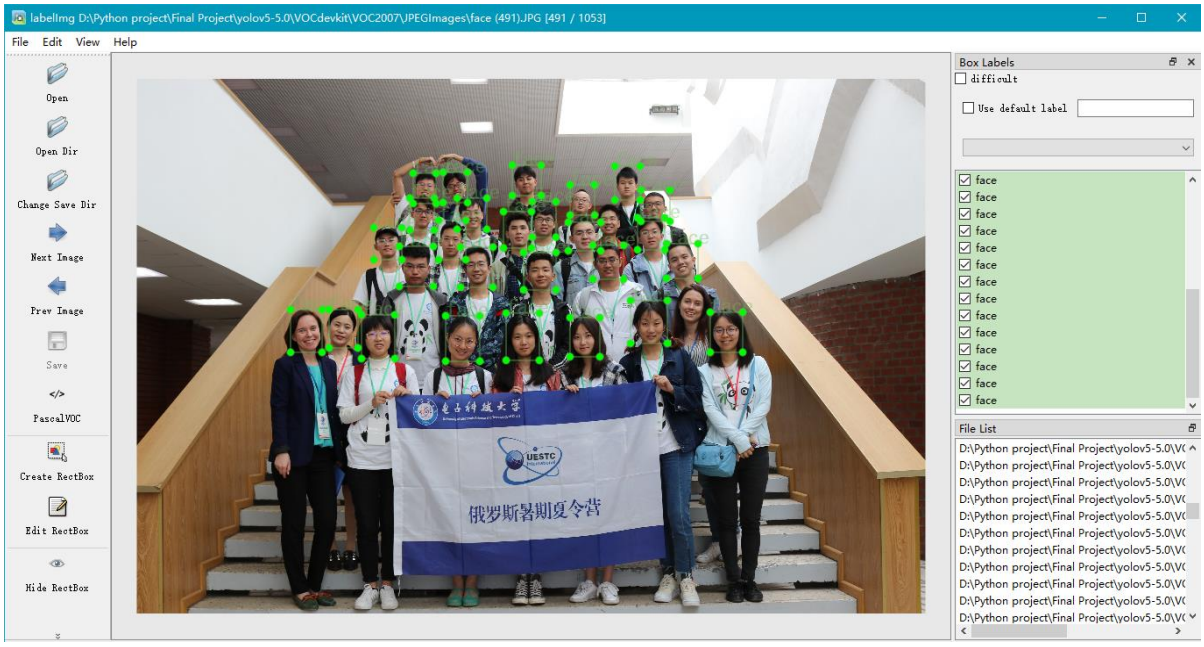


Figure 6. Face marking in Labelimg

```
▼<annotation>
    <folder>JPEGImages</folder>
    <filename>face (10).JPG</filename>
    <path>D:\Study\Final Project\yolov5-5.0\VOCdevkit\VOC2007\JPEGImages\face (10).JPG</path>
  ▼<source>
      <database>Unknown</database>
    </source>
  ▼<size>
      <width>5760</width>
      <height>3840</height>
      <depth>3</depth>
    </size>
    <segmented>0</segmented>
  ▼<object>
      <name>face</name>
      <pose>Unspecified</pose>
      <truncated>0</truncated>
      <difficult>0</difficult>
    ▼<bndbox>
        <xmin>1377</xmin>
        <ymin>1232</ymin>
        <xmax>1908</xmax>
        <ymax>1769</ymax>
      </bndbox>
    </object>
  ▼<object>
      <name>face</name>
      <pose>Unspecified</pose>
      <truncated>0</truncated>
      <difficult>0</difficult>
    ▼<bndbox>
        <xmin>4220</xmin>
        <ymin>826</ymin>
        <xmax>4827</xmax>
        <ymax>1432</ymax>
      </bndbox>
    </object>
</annotation>
```

Figure 7. An example of a label of an image in the dataset

The Fig. 7 is an example of a label of an image, it records the information of two objects, including the file path, object size, object location etc., the coordinates are the four vertex coordinates, and it is worth noting that the label should be saved in txt. file, which will normalize the coordinates of the objects in the image.

When finishing the dataset and label collecting, a train set and a test set are required for the following training work. For this project, we only have one class which is person, and we disrupt the previous dataset in random and set the train and test set according to the rate 8:2.

We have already finished the preparation for the training, but before that, an exploratory analysis of the dataset is needed. In order to make our dataset have higher generalization to avoid overfitting during training, there are usually two ways to improve, one is to add some advanced loss functions, such as Focal loss to the training loss function, and the other is to use

data enhancement on the training data during the training period. Because it is complex to modify the loss function, data enhancement is selected as a method to improve the structure of the dataset in this project. Data enhancement refers to the pre-processing stage before training. Though some small operation to the input image, such as random erasure, adjusting saturation, etc., so that the neural network thinks it is a brand-new image to achieve the effect of increasing the amount of data. This paper uses mosaic data enhancement [23], which is based on the principle of randomly scaling, flipping, adjusting the colour gamut, randomly distributing and then stitching together four images. This can not only increase the amount of detection data, but also enrich the background of the detection object in the dataset and reduce the probability that the background is misclassified as the detected object.

3.2 YOLOv5 Model Analysis and Improvement

In the previous chapter, we discussed which network model would be chosen for this person detection project, and YOLO series has many advantages in video detection. Besides, compared with YOLOv1 to YOLOv4, the latest network YOLOv5 has been greatly improved. However, there are still several different types of the YOLOv5 series model, and they have different performance. The specific parameters and performance for different models are shown in Fig. 8. From YOLOv5n to YOLOv5x, the width of the network gradually deepens and the AP accuracy continues to improve, but at the same time, the speed consumption is also increasing. Therefore, in different projects, the precision and speed requirement of the project should be comprehensively considered. However, although different types of YOLOv5 models have some differences in depth and width, the basic network architecture of the five models are same, only the two parameters, the depth and width of the model are different.

For the projects which need to detect small objects, or classify several categories of the objects, a deeper and more precision model will be better. For person detection, it only needs to detect one class that is person, and requires high detection speed since it needs to achieve real-time detection, so YOLOv5s is the first choice for its smaller model size and faster detection speed, with relative high accuracy. In this project, we choose the latest version YOLOv5s5 until November 2021 as the person detection model.

Model	size (pixels)	mAP^{val} 0.5:0.95	mAP^{val} 0.5	Speed CPU b1 (ms)	Speed V100 b1 (ms)	Speed V100 b32 (ms)	params (M)	FLOPs @640 (B)
YOLOv5n	640	28.0	45.7	**45**	**6.3**	**0.6**	**1.9**	**4.5**
YOLOv5s	640	37.4	56.8	98	6.4	0.9	7.2	16.5
YOLOv5m	640	45.4	64.1	224	8.2	1.7	21.2	49.0
YOLOv5l	640	49.0	67.3	430	10.1	2.7	46.5	109.1
YOLOv5x	640	50.7	68.9	766	12.1	4.8	86.7	205.7

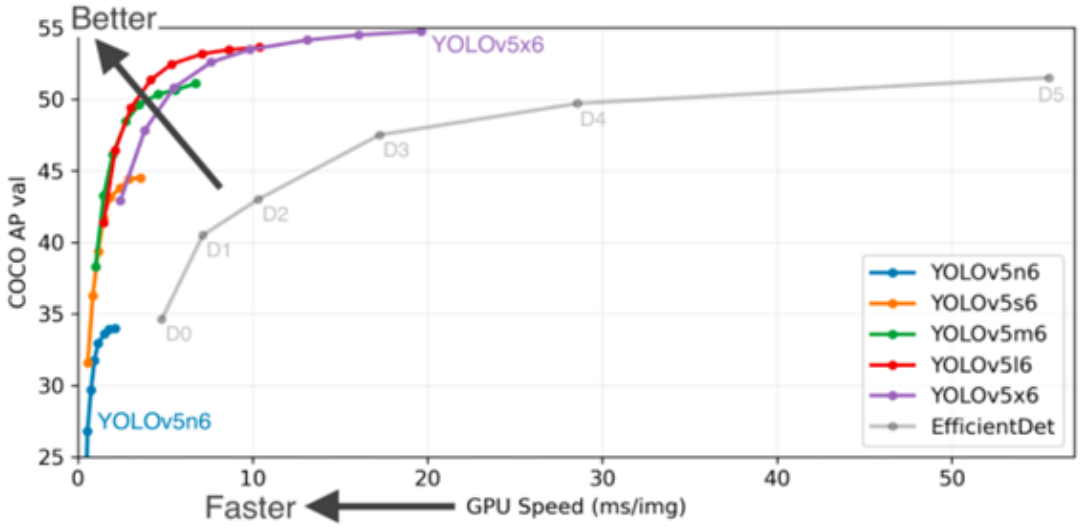


Figure 8. Different models of YOLOv5 and their performance

However, after analysing the parameters in Fig. 8, the performance of YOLOv5 for detecting the public dataset COCO is not so well. The main reason may be that the object feature extracted by the original YOLOv5 model is not enough. By analysing the source code of YOLOv5, the original model has only three detection layers and they are corresponding to three anchors. When the input size is 640×640, P3, P4, P5 correspond to the detection layer size of 80×80, 40×40, 20×20, which can be used to detect objects with the size 8×8, 16×16, 32×32 or larger respectively. Although the three output layers can predict the objects from small to large, it cannot detect the samples of objects smaller than 8×8. In addition, the localization performance of YOLOv5 for dense small objects is not very satisfactory. Therefore, it is necessary to make some improvement in order to achieve higher precision for person detection, especially for small-sized person.

An improvement method is to improve the backbone of the model. For the feature fusion part, a layer of upsampling is added to the original 8x downsampled output to fuse with the 4x downsampled features to extract more features of the input images, so that it can improve the

detection precision of small targets. The specific steps are adding a group of anchors at first. The parameters should be setting small enough in order to detect the tiny objects. Then several operation layers are added in the neck part of the model. After upsampling for two times, the feature map still keeping to be unsampled in the 19th layer, so that it can still be increased. Meanwhile, the feature map with the size of 160×160 is concatenated with that in the second layer, to consist a larger feature map in order to realize the target for detecting the tiny object in the 20th layer. Besides, a new layer is added for detecting the tiny object in the head part of the network in 31st layer. Therefore, the 21st, 24th, 27th and 30th layers are used for detection. The Fig. 9 shows the architecture of the improved YOLOv5 model. The specific code for the improved YOLOv5 model is shown in the Appendix.

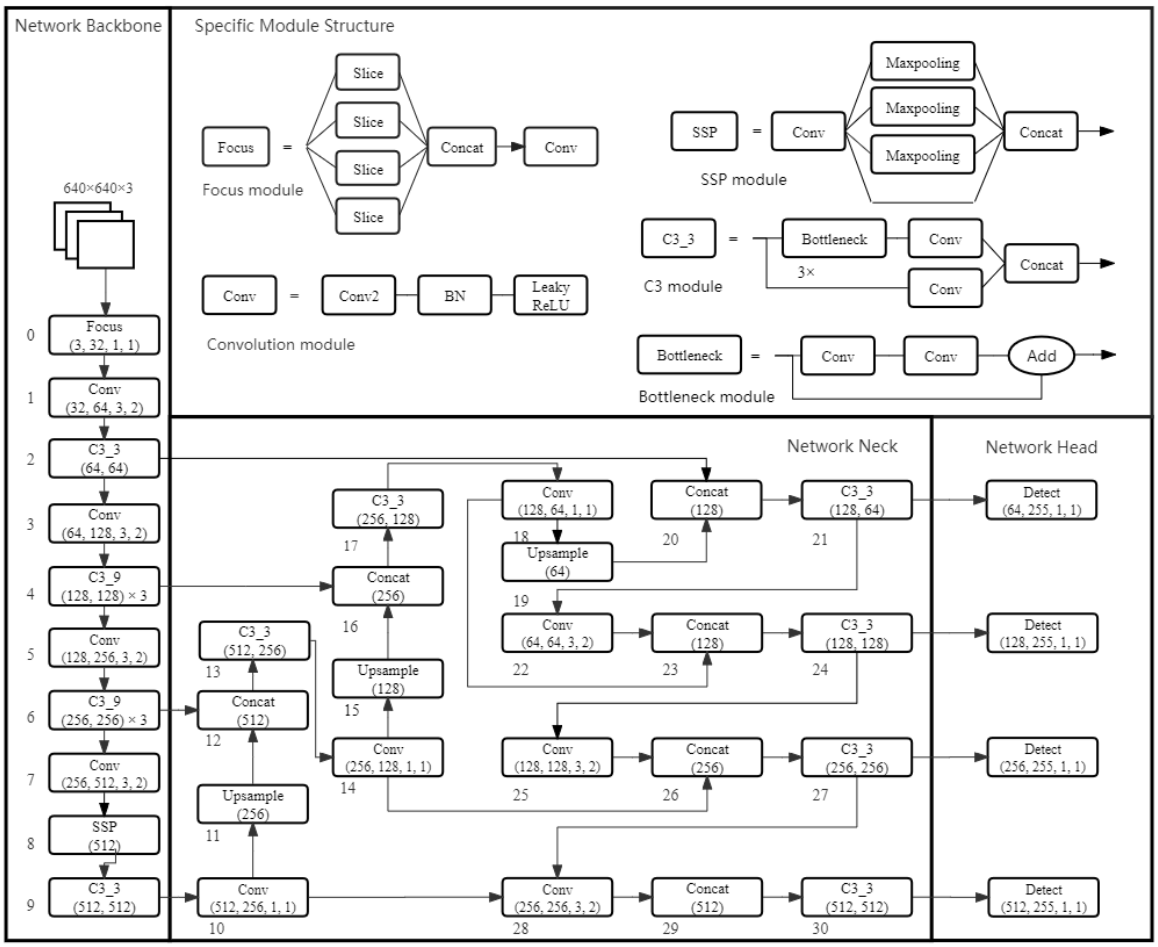


Figure 9. Improved YOLOv5s network structure

Based on these improvements, the feature map responsible for prediction output is expanded from the original three to four. This improvement will bring two main advantages for the training.

- **Better fitting features** – In general, the convolution layers, pooling layers and activation layers make up the main structure of a network based on deep learning algorithms. This module is a kind of nonlinear transformation module as mentioned in chapter 2. If the depth of the network increase, more complexed feature can be extracted and learned by the training model for the high-level feature are in the form of nonlinear expression, which can achieve a higher precision. Meanwhile, it can fit more complex inputs.
- **Simplify the burden of each layer** – For a simple network as an example, the first, second and third layers learn the edge, the simple shape and the target shape respectively, and some high-level features of the objects can be learned by the deeper network layers. However, the transformation to be learned is very complex for only one layer, which influences the effect of the training.

On the other hand, the deepening of network structure will increase the amount of calculation, and results in the disadvantage that the network become more complex and the operation speed decreases. Therefore, it is necessary to achieve a balance between the precision and the speed.

4 Detection Results and Analysis

In this chapter, we will introduce the software and hardware platform for this project, followed by a comparative analysis of the performance of the original and improved model. Besides, the effects of practical person detecting will also be shown.

4.1 Detection Platform Introduction

The deep learning environment and framework building is one of the necessary parts of this project. For object detection, Pytorch and TensorFlow are the two frameworks which used most commonly, and the main difference is that Pytorch is a dynamic framework, while TensorFlow is a static framework. However, compared with TensorFlow, Pytorch is an open-source framework developed which have some advantages:

1) Similar to python programming, easy to get started and high readability.
2) Using dynamic computational graphs which is more convenient to write and debug.
3) Easy to use GPU for accelerated computing.

Therefore, we will use the Pytorch framework to implement the YOLOv5 model in this paper. Besides, we choose Python as the programming language, and PyCharm as the editor. Therefore, the whole information, including the parameters and version of the software and hardware for this project is shown below.

- **Language** – Python 3.7.11
- **IDE** – PyCharm 2.3
- **Framework** – Pytorch 1.10.2
- **CPU** – AMD Ryzen 7 5800H
- **Cores / Threads** – 8 cores / 16 threads
- **RAM** – 16GB (8GB×2)
- **GPU** – NVIDIA GeForce RTX 3070, 8GB video memory
- **Operation System (OS)** – Windows 11, 64bit

All the results of this projects we will show in the next part are obtained through this platform.

4.2 Training Results Analysis

The training results of the original and improved YOLOv5 model for this project are shown in this part. Each model is used to training for 200 epochs, and the mAP, precision, recall is used as the evaluation index of the model. The introduction for these indexes is shown in chapter 2. The line diagrams are gotten from the tensorboard, which is a callable function recording the results for each epoch, and make the whole results visualization. In every line diagram in this part, the blue curve represents the performance of the original model, and the red curve refers to it of the improved model.

The Fig. 10 is the line diagram of the mAP with IoU threshold of 50% during the training period. The vertical coordinate is the mean average accuracy considering only the predicted values of IoU greater than 50% of the bounding box, and the horizontal coordinate is the number of training rounds epoch, where one epoch refers to the completion of training all the data in a dataset.

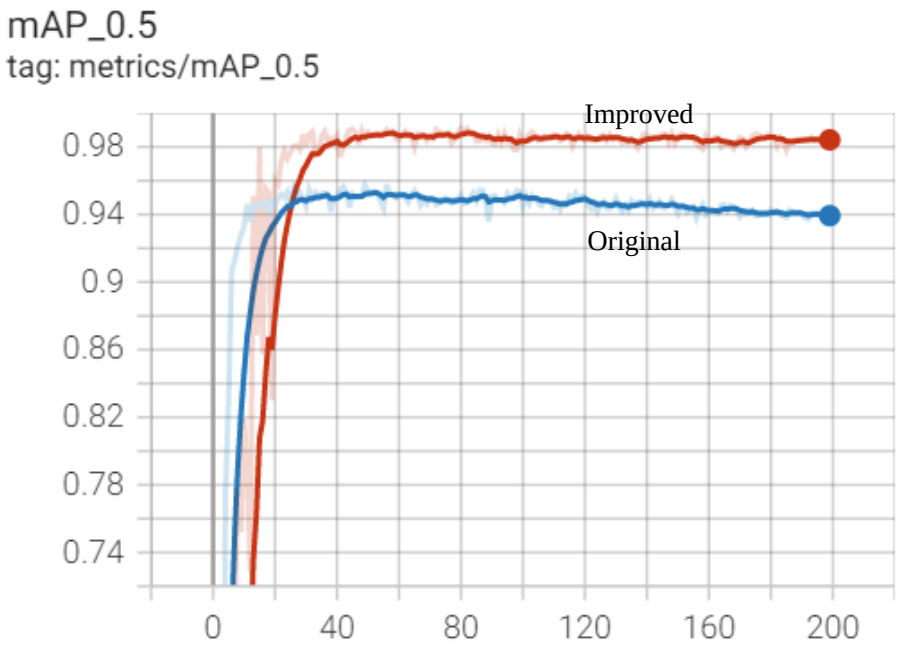


Figure 10. The mAP of the original and improved YOLOv5s model in training

The original YOLOv5s network model has a better result than the improved model before the first 30th epochs, however, when the mAP of the model still increases and tends to be stable, the value of the improved model is more than 98%, which is higher than the 95% of the results gotten by the original model. In general, a higher map means that the model can locate the object more accurately, which proved that the improved model has better ability in localization.

The Fig. 11 and Fig. 12 are the line diagrams of the precision and recall value of the two models during the 200 epochs training. Similar to the Fig. 10, the vertical coordinate is the precision and recall value, and the horizontal coordinate refer to the training epochs.

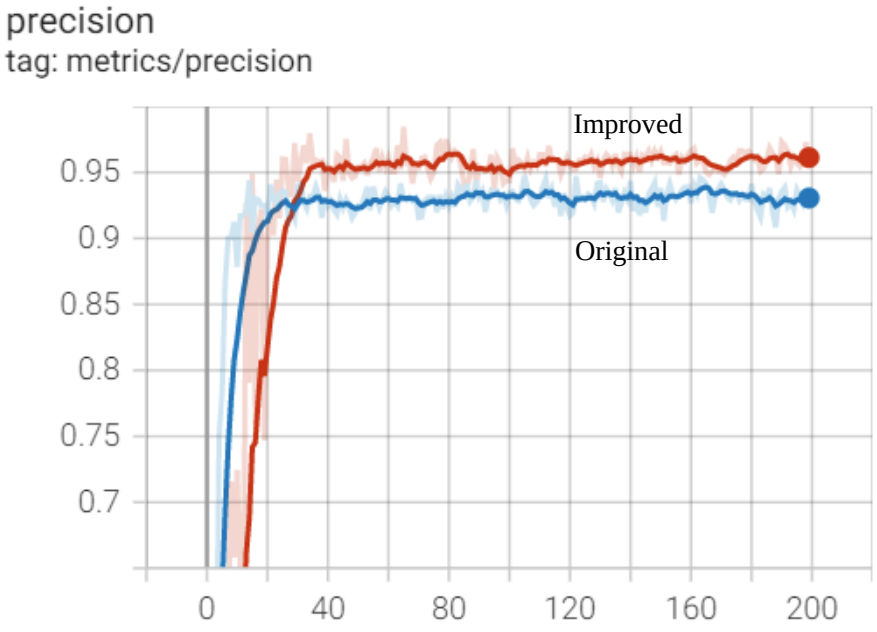


Figure 11. The precision of the original and improved YOLOv5s model in training

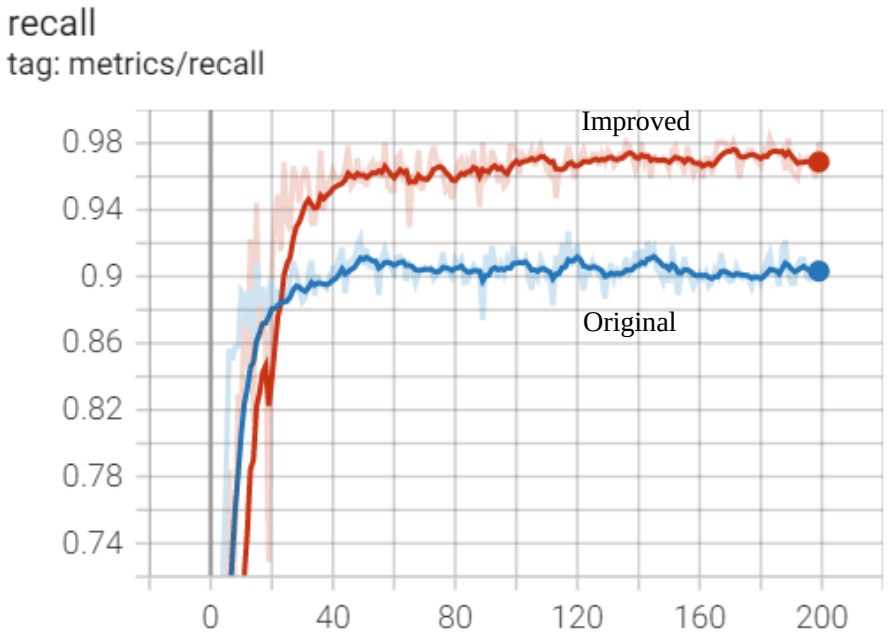


Figure 12. The recall of the original and improved YOLOv5s model in training

The results in Fig. 11 and Fig. 12 present that the improved YOLOv5s model has better performance for it achieves 96% in precision and 97% in recall when it tends to be stable, which is higher than the original model with 93% in precision and 90% in recall.

The Fig. 13 and Fig. 14 are the diagrams of the loss results of the two models detecting the training set and test set for 200 epochs receptively. For each loss result, there are three diagrams, which refers to the box_loss, cls_loss and obj_loss.

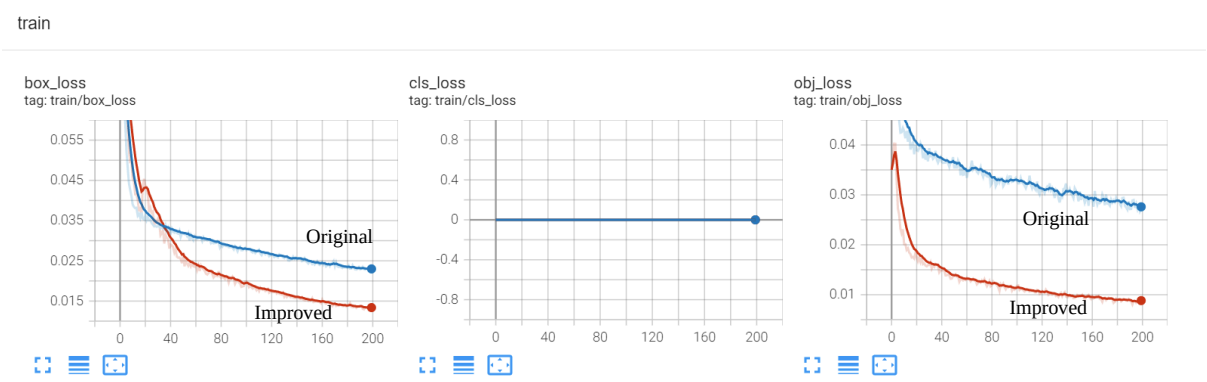


Figure 13. The loss of the original and improved YOLOv5s model for training set

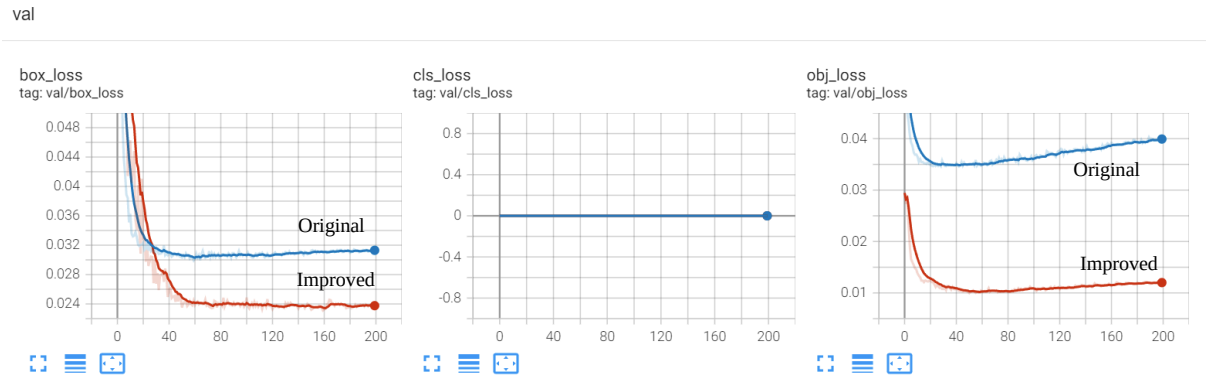


Figure 14. The loss of the original and improved YOLOv5s model for test set

The curves on the left of the two diagrams are box_loss, which represents the error of the model's bounding box coordinates positioning, usually the lower it is, the more accurate the positioning of the boundary box of the model is. Besides, a new loss function C-IoU loss is used in YOLOv5 to calculate box_loss, which considers the distance between the centre point and the aspect ratio of the bounding box more than the original loss function, and can better reflect the proximity between the prediction box and the real box. From the results, the improved model has better performance than the original model, for it has a lower box_loss in both the training set and the test set, which indicates that the improved model is more accurate in localization.

For person detection, there is only one class need to be detected, so that the cls_loss for the two model is both 0 and it does not need to be taken into consideration.

In addition, the curves on the right of the two diagrams refer to the obj_loss, which indicates the error in confidence that the bounding box is an object. The lower it is, the stronger the

ability of the model to judge whether an object is present in the bounding box. From the results, the obj_loss of the improved model is lower when they tend to be stable, which indicates that the improved model is more capable of judging the presence or absence of objects and can reduce the probability of the background being misclassified. Besides, both the blue and red curves tend to decrease at first and then increase. The main reason causes this phenomenon is overfitting. Therefore, the model weight corresponding to the lowest point in the loss curve should be selected as the best model weight in order to select the model weights with the strongest generalization ability.

The table. 2 shows the training results for two models in the final training epoch. All of the data are gotten from the tensorboard.

Table 2. Results for two models after training 200 epochs

Evaluation index	Original YOLOv5 model (%)	Improved YOLOv5 model (%)	Improvement (%)
mAP	93.89	98.41	4.52
precision	93.06	96.13	3.07
recall	90.30	96.87	6.57
box_loss(train)	2.30	1.33	-0.97
obj_loss(train)	2.70	0.88	-1.82
box_loss(val)	3.16	2.36	-0.80
obj_loss(val)	4.00	1.20	-2.80

From the results above, when the training model gradually trend to be stable after plenty epochs of training, the improved YOLOv5 model has a better performance in every evaluation indexes.

4.3 Detecting Results Presentation

After finishing the training and getting the model weight for person detection, the model should to be used for detecting person. We use an image from the dataset to let the training model detect at first, in order to test whether it can achieve person detection. The visualized results are represented from Fig. 15 and Fig. 16. The Fig. 15 is detected by the original model while Fig. 16 is detected by the improved model.

Figure 15. Detection effect for an image by the original YOLOv5s model

Figure 16. Detection effect for an image by the improved YOLOv5s model

It is obviously that there are eight people in this image, and both of the models can detect all of the persons and marked their faces with a confidence. The confidence means the probability

that the model considers the detected objects as the correct objects. The table 3 shows the specific confidence and the improvement percentage.

Table 3. Confidence results for two models

Number (from left to right)	Original YOLOv5 model	Improved YOLOv5 model	Improvement (%)
1	0.80	0.89	11.25
2	0.83	0.88	6.02
3	0.84	0.90	7.14
4	0.84	0.90	7.14
5	0.77	0.88	14.29
6	0.81	0.87	7.4
7	0.83	0.89	7.23
Ave			8.69

It is easy for both models to detect because the objects are clear with suitable size. However, compared with the two visualized results, for each face in the image, the improved model always gives a higher confidence than the original model. The higher the confidence, the better effect of the model detection. After analysing the table 3, the improved model has about 8.69% confidence higher than the original model.

In order to verify that our trained model can achieve person detection in video, we use a new video for verification. The video is a piece of music MV, and it contains several scenes where many people with small and fuzzy faces get together. We detected this video with the original model and the improved one respectively, the results are that both of the two models can achieve the real-time detection. Besides, we extracted the same frame of the image scene from it, and visualized the detection results of the two models for comparison.

The Fig. 18 and Fig. 19 is the visualized detect results of a frame image from the video by the original model and improved model respectively. The frame we extracted contains plenty of people with fuzzy and small faces, some people even wear a hat or with a face in different directions. This will result in the loss of human facial features. Therefore, compared with the clear test image above, it is quite difficult for the model to detect all human faces in this scene.

Figure 17. Detection effect for the video by the original YOLOv5s model

Figure 18. Detection effect for the video by the improved YOLOv5s model

Comparing the two detected results, the original YOLOv5s model only detected five faces, which is worse than the improved model, for it detected almost all faces in the scene. Besides, for the face confidence, the improved model still gives a higher value than the original model.

Therefore, the improved model performs better in both image and video, the improved methods is very successful that it can achieve a precision-speed balance.

5 Conclusions and Expectation

5.1 Conclusions

The paper describes the research background of object detection and its significance, and briefly introduces the status of research on deep learning algorithms in the first chapter. Then in the second chapter, the principle and structure of the convolutional neural network are introduced, and the advantages and disadvantages of them are analyzed. Also, the YOLOv5 network model is the chosen for this project, and the third chapter main focus on the introduction for the architecture and algorithms of the YOLOv5 network.

To achieve a higher precision while ensure the speed can meets the requirement of real-time detection, this paper propose two improvement methods.

- **Data enhancement** – Using mosaic data enhancement to ensure the samples of large, middle and small sized faces in the dataset are more balanced, and increase the generalization degree of data set to avoid the problem of overfitting in model training.
- **Network deepening** – This paper proposes to adjust the feature fusion part of the YOLOv5 model, adding a tiny object detection layer, to avoid the problems for the original model in small objects detection and localization.

In the chapter 4, the improved methods are verified by experiments. The results show that the improved model has a better performance than the original model in person detection, and the localization effect and confidence of the improved model for small faces are also better than the original one. For the training results, the mAP increases from about 94% of the original model to 98% of the improved model. The precision and recall also increases from about 93% to 96% and about 90% to 97% through the improvement, which improved nearly 3% and 7% respectively. From the training results of the image and video detection tests, although the size of the model has become larger and the amount of computation increases, the improved model still maintains the effect of real-time detection which meets the requirement.

5.2 Expectation and Further Work

Although this paper proposes some improvement methods for the original YOLOv5 model, it still has the advanced potential.

Firstly, the improved YOLOv5 model makes improvement in the mean average precision, precision and recall, but the number of parameters has increased due to the addition of detection layer and the fusion of features with large feature maps, which is not conducive to deployment

in resource-constrained devices such as embedded devices. Therefore, it is expected that the size of the model can be reduced while maintaining the existing precision. Secondly, through the detection effect for some test videos, the improved YOLOv5 model will still make mistakes in recognize a large number of objects or distant small objects, although the overall detection effect is pretty good. This problem can be improved through adjusting the output layer of the network model or making some changes for the dataset.

The further work for improvement has forth directions.

1) Making some changes for the person detection dataset, like upsampling or artificially increasing the number of faces of relatively small sizes, so that the distribution of different sized faces is more balanced.
2) Referring to the deep separable convolution in MobileNet [24] and try to apply the deep separable convolution to the network structure of improved YOLOv5 to lighten the model. Therefore, the amount of computation will be reduced to accelerate the detection speed.
3) Trying to apply the popular Visual Transformer [25] to the YOLOv5 network structure to improve the detection effect of the network.
4) Designing a system for human-computer interaction. After finishing the model training and improvement, it is necessary to apply it into practical applications. Therefore, a more convenient, user-friendly system will make the detection model wider used.

The improvement and progress I made during this research project cannot leave the people who helped me. Besides, I will continue to optimize my detection network model in my next study stage, to achieve a higher precision, faster speed and humanized person detection system.

References

[1] Z. Wu and Z. Hu, “Object detection based on self feature distillation,” Journal of Physics: Conference Series, vol. 1982, no. 1, p. 012081, Jul. 2021.

[2] H. Li, J. Yuan, H. Liu, L. Cao, J. Chen, and Z. Zhang, “Incremental Learning of Infrared Vehicle Detection Method Based on SSD,” Oct. 2020.

[3] He X, Gao X, Zhang Y, et al. Intelligence Science and Big Data Engineering. Big Data and Machine Learning Techniques[M]. Springer International Publishing, 2015.

[4] X. Chu, “Speech recognition method based on deep learning and its application,” Aug. 2021.

[5] U. Arshad, “Object detection in last decade - A survey,” Scientific Journal of Informatics, vol. 8, no. 1, pp. 60–70, May 2021.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” Communications of the ACM, vol. 60, no. 6, pp. 84–90, May 2017.

[7] S. Kim and Y. Hwang, “A Survey on Deep Learning Based Methods and Datasets for Monocular 3D Object Detection,” Electronics, vol. 10, no. 4, p. 517, Feb. 2021.

[8] W. Liang, P. Xu, L. Guo, H. Bai, Y. Zhou, and F. Chen, “A survey of 3D object detection,” Multimedia Tools and Applications, vol. 80, no. 19, pp. 29617–29641, Jul. 2021.

[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation,” Jun. 2014.

[10] R. Girshick, “Fast R-CNN,” Dec. 2015. Accessed: Apr. 10, 2022.

[11] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[12] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” Jun. 2016.

[13] J. Redmon and A. Farhadi, “YOLO9000: Better, Faster, Stronger,” Jul. 2017.

[14] J. Redmon and A. Farhadi, “YOLOv3: An Incremental Improvement,” Jul. 2018.

[15] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “YOLOv4: Optimal speed and accuracy of object detection,” arXiv.org, Apr. 23, 2020.

[16] K. Fukushima and S. Miyake, “Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Visual Pattern Recognition,” in Competition and Cooperation in Neural Nets, Berlin, Heidelberg: Springer Berlin Heidelberg, 1982, pp. 267–285.

[17] “Artificial Neural Networks and Machine Learning – ICANN 2017 Volume 10614,” Springer Science and Business Media LLC, 2017.

[18] Y. Tian, “Artificial Intelligence Image Recognition Method Based on Convolutional Neural Network Algorithm,” IEEE Access, vol. 8, pp. 125731–125744, 2020.

[19] J. Zhao, T. Chen, and B. Cai, “A computer-aided diagnostic system for mammograms based on YOLOv3,” Multimedia Tools and Applications, Feb. 2021.

[20] Z. Chen et al., “Deep Learning for the Detection and Recognition of Rail Defects in Ultrasound B-Scan Images,” Transportation Research Record: Journal of the Transportation Research Board, vol. 2675, no. 11, pp. 888–901, Jul. 2021.

[21] F. A. Batarseh and R. Yang, Data Democracy: At the Nexus of Artificial Intelligence, Software Development, and Knowledge Engineering. Academic Press, 2020.

[22] R. Padilla, S. L. Netto, and E. A. B. da Silva, “A Survey on Performance Metrics for Object-Detection Algorithms,” Jul. 2020.

[23] G. Zeng, W. Yu, R. Wang, and A. Lin, “Research on Mosaic Image Data Enhancement for Overlapping Ship Targets,” arXiv.org, May 11, 2021.

[24] A. G. Howard et al., “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” arXiv.org, Apr. 17, 2017.

[25] Y. Liu et al., “A Survey of Visual Transformers,” arXiv.org, Nov. 11, 2021.

A Improved model code

This is the python code of the improved YOLOv5s model.

For the improved model, there is only one class for detection, so the number of class is set to 1, and the depth and width are the two parameters of the network.

```yaml
# parameters
nc: 1  # number of classes
depth_multiple: 0.33  # model depth multiple
width_multiple: 0.50  # layer channel multiple

```

For the anchors, the improved model adds a group p2, and the parameters are in small value.

```yaml
# anchors
anchors:
  - [5,6, 8,14, 15,11]  # P2/4
  - [10,13, 16,30, 33,23]  # P3/8
  - [30,61, 62,45, 59,119]  # P4/16
  - [116,90, 156,198, 373,326]  # P5/32
```

For the backbone part of the network, maintian the original structure of the YOLOv5 model. The 1st, 3rd, 5th, 7th layers are the convolution layers, and 2nd, 4th, 6th, 9th layers are the C3 layers. Besides, the 8th layer is the SPP layer.

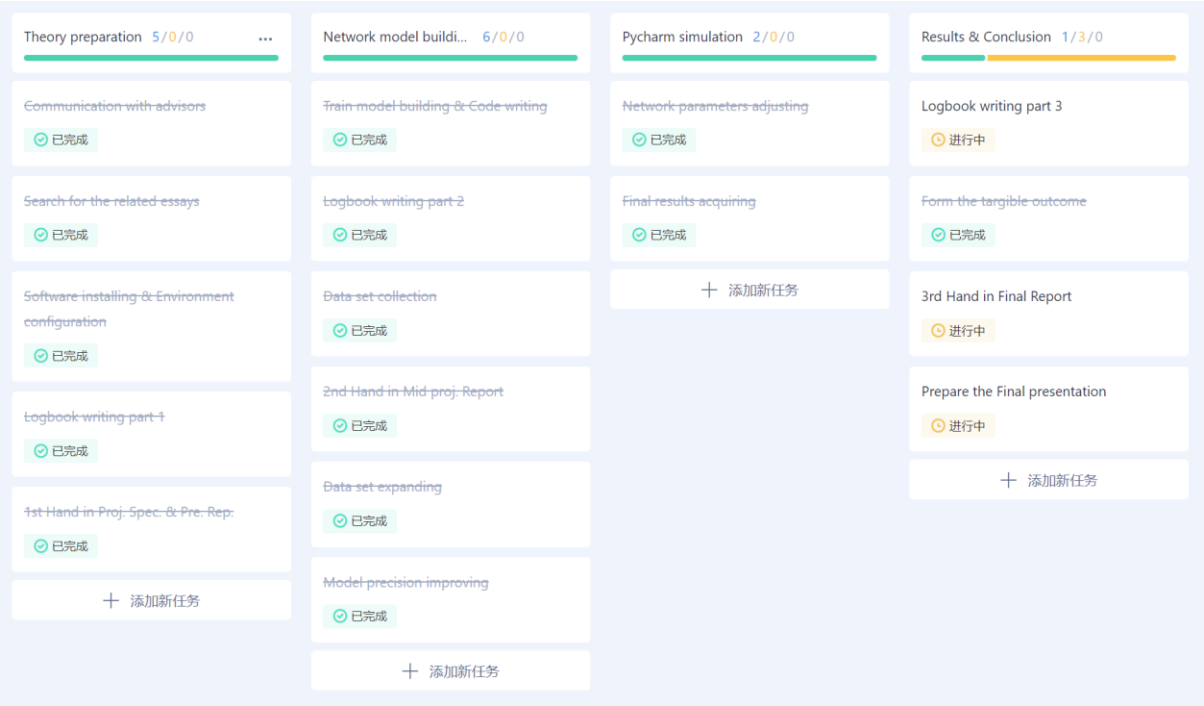
```yaml
# YOLOv5 backbone
backbone:
  # [from, number, module, args]
  [[-1, 1, Focus, [64, 3]],  # 0-P1/2:320
   [-1, 1, Conv, [128, 3, 2]],  # 1-P2/4:160
   [-1, 3, BottleneckCSP, [128]],
   [-1, 1, Conv, [256, 3, 2]],  # 3-P3/8:80
   [-1, 9, BottleneckCSP, [256]],
   [-1, 1, Conv, [512, 3, 2]],  # 5-P4/16:40
   [-1, 9, BottleneckCSP, [512]],
   [-1, 1, Conv, [1024, 3, 2]],  # 7-P5/32:20
   [-1, 1, SPP, [1024, [5, 9, 13]]],
   [-1, 3, BottleneckCSP, [1024, False]],  # 9:20
  ]
```

For the head part of the network, adding an upsampling layer, and concatenated the feature map of 160×160 with that of the second layer at the 20th layer. The 31st layer is a new detection layer for tiny object detection.

```
# YOLOv5 head
head:
  [[-1, 1, Conv, [512, 1, 1]],  #20*20
   [-1, 1, nn.Upsample, [None, 2, 'nearest']], #40*40
   [[-1, 6], 1, Concat, [1]],  # cat backbone P4  40*40
   [-1, 3, BottleneckCSP, [512, False]],  # 13     40*40

   [-1, 1, Conv, [512, 1, 1]], #40*40
   [-1, 1, nn.Upsample, [None, 2, 'nearest']],
   [[-1, 4], 1, Concat, [1]],  # cat backbone P3   80*80
   [-1, 3, BottleneckCSP, [512, False]],  # 17 (P3/8-small)  80*80

   [-1, 1, Conv, [256, 1, 1]], #18  80*80
   [-1, 1, nn.Upsample, [None, 2, 'nearest']], #19  160*160
   [[-1, 2], 1, Concat, [1]], #20 cat backbone p2  160*160
   [-1, 3, BottleneckCSP, [256, False]], #21 (P2/4-tiny) 160*160

   [-1, 1, Conv, [256, 3, 2]],  #22   80*80
   [[-1, 18], 1, Concat, [1]], #23 80*80
   [-1, 3, BottleneckCSP, [256, False]], #24 80*80

   [-1, 1, Conv, [256, 3, 2]], #25  40*40
   [[-1, 14], 1, Concat, [1]],  # 26  cat head P4  40*40
   [-1, 3, BottleneckCSP, [512, False]],  # 27 (P4/16-medium) 40*40

   [-1, 1, Conv, [512, 3, 2]],  #28  20*20
   [[-1, 10], 1, Concat, [1]],  #29 cat head P5  #20*20
   [-1, 3, BottleneckCSP, [1024, False]],  # 30 (P5/32-large)  20*20

   [[21, 24, 27, 30], 1, Detect, [nc, anchors]],  # Detect(p2, P3, P4, P5)
  ]
```

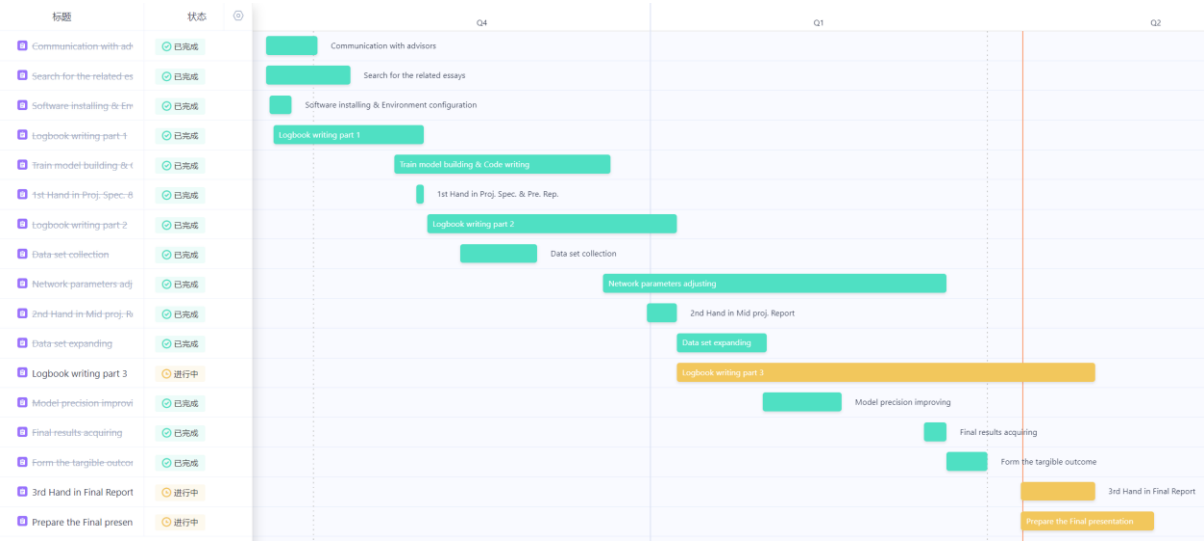
B Task Review and Gannt Chart

The Appendix Fig. 1 presents the work plans I made before I started this project, and the task completion. There are only two tasks are still under working, while other tasks are all finished.



Appendix Figure 1. Work plan listing and review

The Appendix Fig. 2 presents the timeline of the tasks.



Appendix Figure 2. Gannt Chart