

Final Report

School of Computing

Faculty of Engineering AND PHYSICAL SCIENCES

<Object Detection Website Based on Deep Learning>

<Yunjia Feng>

Submitted in accordance with the requirements for the degree of

<BSc Computer Science>

**<2021/22>**

The candidate confirms that the following have been submitted*:*

*<As an example>*

|  |  |  |
| --- | --- | --- |
| **Items** | **Format** | **Recipient(s) and Date** |
| *Final Report* | *Hard copy and PDF file* | *Hard copy handed to SSO (DD/MM/YY); PDF uploaded to Minerva (DD/MM/YY)* |
| *Participant consent forms* | *PDF file / file archive* | *Uploaded to Minerva (DD/MM/YY)* |
| *Link to online code repository* | *URL* | *Sent to supervisor and assessor (DD/MM/YY)* |
| *User manuals* | *PDF and/or hard copy* | *Sent to client and supervisor (DD/MM/YY)* |

The candidate confirms that the work submitted is their own and the appropriate credit has been given where reference has been made to the work of others.

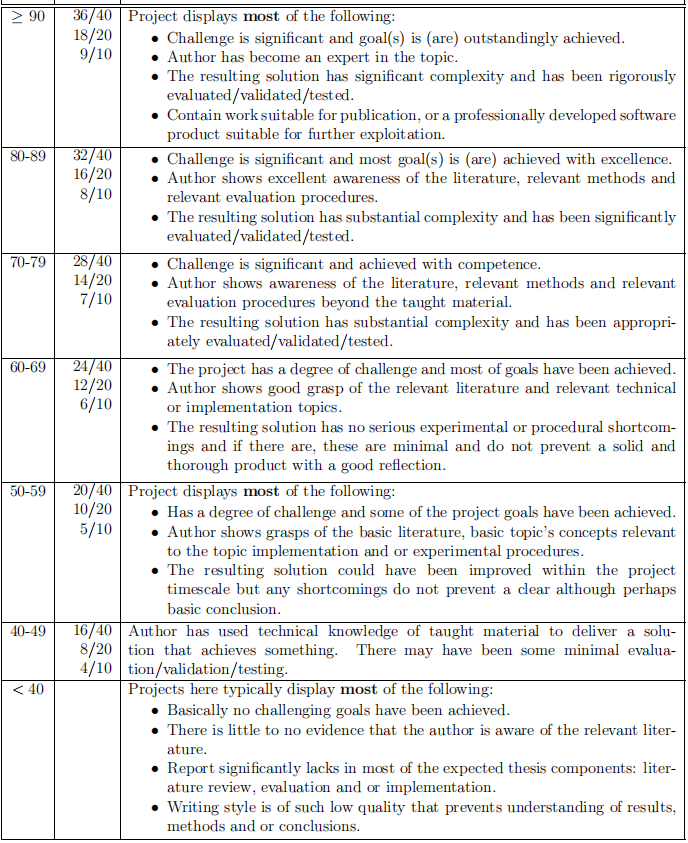
I understand that failure to attribute material which is obtained from another source may be considered as plagiarism.

(Signature of student) Yunjia Feng

© <2022> The University of Leeds and <Yunjia Feng>

# Summary

*<Concise statement of the problem you intended to solve and main achievements (no more than one A4 page)>*



# Acknowledgements

*<This page should contain any acknowledgements to those who have assisted with your work. Where you have worked as part of a team, you should, where appropriate, reference to any contribution made by others to the project.>*

*Note that it is not acceptable to solicit assistance on ‘proof reading’ which is defined as “the systematic checking and identification of errors in spelling, punctuation, grammar and sentence construction, formatting and layout in the text”; see*

https:://www.leeds.ac.uk/secretariat/documents/proof\_reading\_policy.pdf

# Table of Contents (example of how to format)

Summary iii

Acknowledgements iv

Table of Contents v

Chapter 1 Level 1 Heading with ‘heading 1’ Style 1

1.1 Level 2 Heading with ‘heading 2’ Style 1

1.1.1 Level 3 Heading with ‘heading 3’ Style 1

1.1.1.1 Level 4 Heading with ‘heading 4’ Style 1

Chapter 2 Tables and Figures 2

2.1 Tables using the ‘table caption’ and ‘table description’ Styles 2

2.2 Figures using the ‘figure caption’ and ‘figure description’ Styles 2

List of References 3

Appendix A Self-appraisal 4

A.1 Level 2 Heading with ‘heading 2’ Style 4

A.1.1 Level 3 Heading with ‘heading 3’ Style 4

A.1.1.1 Level 4 Heading with ‘heading 4’ Style 4

Appendix B External materials 4

B.1 Level 2 Heading with ‘heading 2’ Style 4

## Chapter 1

## Introduction and Background Research

## 1.1 Introduction

Computer vision analysis of target motion can be roughly divided into three levels: motion segmentation and object detection, target tracking, action recognition and behavior description[2].Object detection has always been of great significance in the field of computer science, with the main objective to enable the computer to accurately classify the objects in a given picture and find the position of each object. Object detection is not only one of the basic tasks to be solved in the field of computer vision but also the basis of video surveillance technology among the other tasks. However, object detection is still a challenging task with great potential and has great space for improvement, since the objects in the video have different poses and often appear to overlap, especially when their movements are irregular. Meanwhile, the resolution, weather, illumination, and other conditions of the surveillance video or images as well as the diversity of scenes should also be taken into consideration, which makes this task more challenging.

In recent years, many computer vision researchers both at home and abroad had developed a large number of excellent object detection algorithms in a neural network, including Faster R-CNN, SSD, YOLO.

From the perspective of research, the significance of object detection dues to it is one of the fundamental tasks in the field of computer vision, since it is the basis of many other high-level tasks, including image classification, face recognition, target tracking, pedestrian re-recognition. Meanwhile, there is a large number of well-known national and international research teams had been focused on the field of object detection: MIT Computer Science and Artificial Intelligence Laboratory, Stanford Computer Vision Lab, National Laboratory of Pattern Recognition of Chinese Academy of Sciences, LAMDA Institute of Nanjing University.

While from the perspective of the application, object detection has shown a wide range of practical usages: face detection technology, pedestrian detection technology applied in video surveillance, entrance and exit statistics, traffic sign detection technology, vehicle detection technology applied in aided driving, automatic driving. At the same time, major technology companies, for example, Microsoft, Google, Ali, and Baidu, have also spent a lot of manpower and material resources to explore the object detection field, which also indicates the significance and prospect of object detection.

The goal of this project is to select appropriate object detection algorithms and data sets to train a deep learning model, then develop a website for users which allows them to complete object detection tasks easily. Moreover, the website should provide additional functions that exclude basic object detection functionality to offer users a better experience, for instance, allowing them to change weights to suit different tasks, or mark recognition results after logging in.

## 1.2 Literature Review

## 1.2.1 Significance of Object Detection

Object detection had long been one of the most important and challenging task in computer vision, which detects instances of objects in a given image. It is also basis of other high-level significant computer vision tasks, instance segmentation[1-4], object tracking[8], etc.

Over twenty years, object detection had been developed rapidly especially due to the achievement of deep learning recently[9], tremendous improvements pushed many applications into usage, for instance, face detection technology, autonomous driving, pedestrian detection technology applied in video surveillance, entrance and exit statistics.

## 1.2.2 Methods in Two Periods

Looking back through the history of object detection, it could be divided into two periods: traditional method and deep learning method with the separate line of 2014.

In general, object detection is a task to find all the objects of interest in the image for two sub-tasks, including object positioning and object classification[3]. The traditional object detection method, for example, the sliding window algorithm is generally divided into three stages: firstly, select some candidate regions on a given image, then extract features from these regions, and finally classify them using trained classifiers.

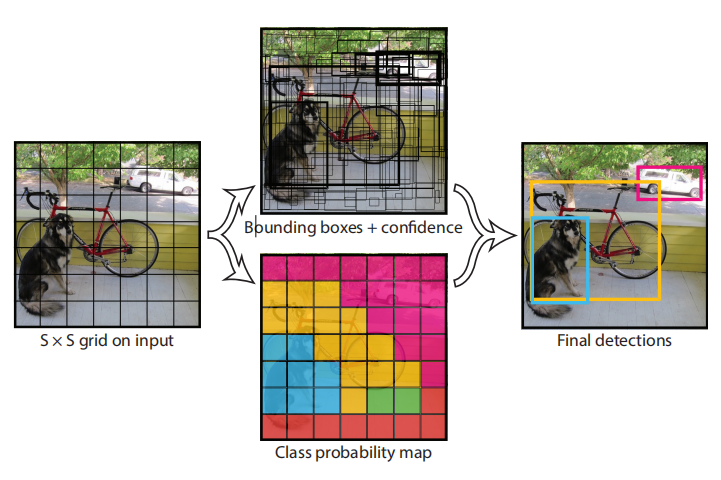
Traditional methods also include Viola Jones Detectors (2001)[10,11], HOG Detector(2005)[12] and DPM (2008). DPM, stands for Deformable Part-based Model, was the winner of the VOC-2007 competition, which also was the best technique in the end era of the traditional method, which focusing on detecting smaller parts of an object, for example, a human could be divided into arms, legs, head.

The efforts of traditional methods not only clear the path but also brought many inspirations to the researchers who study deep learning methods. Since 2006, in the lead of Hinton, Bengio, Lecun, and other researchers, an enormous number of deep neural network papers had been published, especially after Hinton's research group participated in the ImageNet image recognition competition in 2012 and won the championship using AlexNet[1], constructed by CNN (Convolutional Neural Network), then neural networks began to receive extensive attention from then on.

In the deep learning era, the methods were normally divided into two categories: one-stage methods and two-stage methods:

* One-stage object detection algorithms hold the philosophy of detect the objects in only one step. This kind of detection algorithm does not need the Region Proposal Stage and can directly generate the category probability and position coordinates of objects through only one stage. One-stage typical algorithms include YOLO, SSD, RetinaNet and CornerNet[4].
* Two-stage object detection algorithms embrace the idea of “from coarse to fine”[正序15], they divide detection problems into two stages, the first stage is the generation of Region Proposals, which includes the approximate location information of the object, and the second stage is the classification and location refinement of the candidate regions. The representatives of two-stages algorithms are R-CNN, Fast R-CNN, Faster R-CNN, etc.

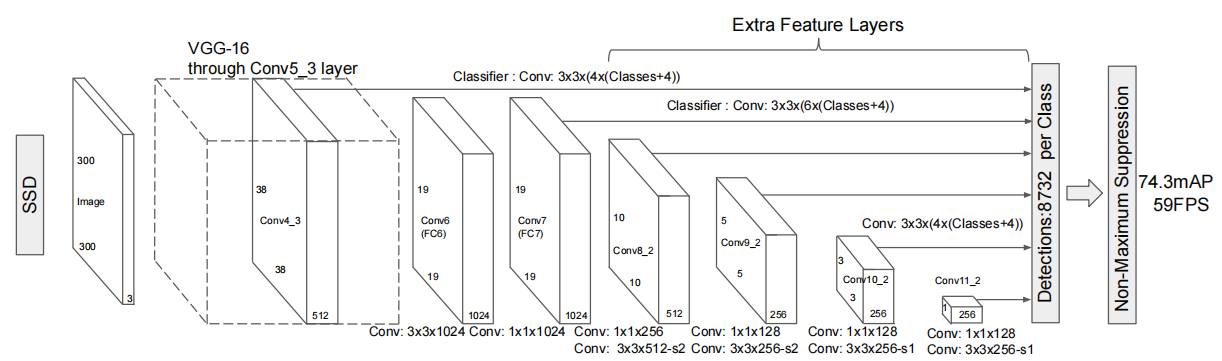
YOLO stands for “You Only Look Once”, is one of the most famous and first one-stage object detection algorithm, adopts a separate CNN model to achieve end-to-end object detection.. YOLO-v0 originates from the idea of transforming the classification network directly into positioning network by dividing the image into several parts, then predicts with bounding boxes (Figure 1.2.2.1). In general, the input image is resized and sent to the CNN network, then the detected object results are obtained by processing network prediction. Compared with the R-CNN algorithm, YOLO is a unified framework with faster speed while the training process is also end-to-end[11].



**Figure 1.2.2.1** Theory of YOLO Algorithm

Since YOLO both has rapid recognition speed and high accuracy, which is suitable for a lightweight website application, this project adopts YOLO-v5 as basis to develop a object detection website. More details will be concentrated on YOLO from v0 to v5 in later chapters.

SSD [21] stands for Single Shot MultiBox Detector, is a single-stage, multiple proposal object detection algorithm. The uniqueness of SSD is its multi-scale detection techniques which allow predicting different sizes of objects using different layers in the network, which significantly enhance the accuracy of single stage method. SSD also uses CNN with a multi-scale feature map to detect objects, the structure shows as the Figure1.2.2.2:



**Figure 1.2.2.2** SSD Network Structure

Additionally, SSD adopts VGG16 as the basic model, and then adds a convolution layer based on VGG16 to obtain more feature maps for detection[5]. There are also some improved algorithms based on SSD, for instance, DSSD[6] and FSSD[7], which have a different structure for their CNN module.

## 1.2.1 Significance of Object Detection

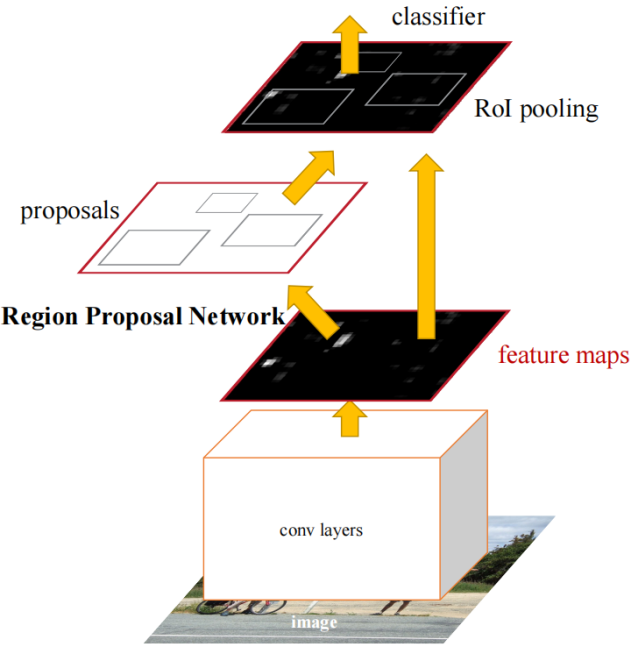
RetinaNet was proposed in 2017[23], which deeply analyse the differences between one-stage methods and two-stage methods, and designed a new loss function called “Focal Loss” in order to solve the problem of imbalanced data.

R-CNN, Fast R-CNN, and Faster R-CNN are a series two-stage algorithms. R-CNN introduces selective search to generate 2000 proposals, then resizes them to fed in a CNN network for feature extraction, a SVM classifier will handle the features and give predictions.[正序8]

Since the 2000 proposals of R-CNN needs large computational resources, leading a really slow speed, Fast R-CNN extracts features from the whole image instead of proposal regions to update weights, and adopts selective search to the outputs of convolution rather than raw image which significantly reduces computation cost and enhance detecting speed.[24]

The Faster R-CNN

Faster R-CNN, is a model after the evolvement of R-CNN and Fast R-CNN, when Ross B. Girshick proposed it in 2016[8]. It adopts RPN (Region Proposal Network) to replace original selective search, reducing a large mount of computation redundancy and improve accuracy at the same time.[正序9] In terms of structure, the Faster R-CNN integrated feature extraction, bounding box regression (rectangular refine), and classification into one network, which greatly improves the overall performance[9], especially in the detection speed. The structure of the network is shown as Figure 1.2.2.3 below:



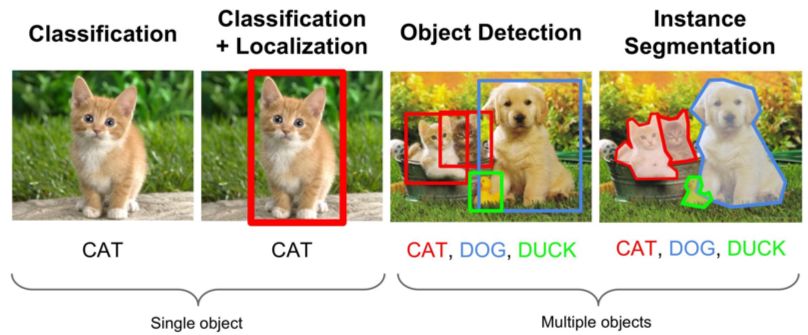
**Figure 1.2.2.3** Faster R-CNN basic structure

Furthermore, Cascade R-CNN was proposed as a multiple-stage method on the basis of previous versions[10] but has better accuracy in detection.

## 1.2.3 Data Set and Evaluation

Excellent algorithms still need proper data set to train and evolve, competitions normally will be hold to test performance on datasets for famous algorithms, also serving as benchmark for later research. Among these datasets, Pascal VOC, MS-COCO and Open Images are frequently used for object detection tasks.

Pascal VOC stands for The PASCAL Visual Object Classes (VOC) Challenges[50,51], supports many computer vision tasks, for example, object detection, instance segmentation, object tracking (Figure 1.2.3.1).



**Figure 1.2.3.1** Computer Vision Tasks

MS-COCO is also a famous data set that used in object detection task, which has more than 160 thousand images with about 900 thousand objects for 80 classes. Open Image data set is normally used in two tasks: object detection and predict relationships between objects.

Within the appropriate algorithm and datasets for training, then a evaluation metrix is needed to test the performance of a method. In object detection task, mAP (mean Average Precision) is one of the most important indicator, which was first mentioned in VOC-2007. To define AP, PR (Precision-Recall) curve is requested with specified IoU (Intersection over Union) threshould (usually 0.5), the area under the PR curve means the AP value. After that, sum up all the AP for every class then divide class number can obtain the final mAP.

## 1.2.4 Important Concepts and Future Work

Among the history of object detection, there are some important concepts that brought huge influence to current object detection research need to be mentioned.

CNN, stands for convolutional neural networks, had been used since 1990[96 ?是否引用] in a variety of computer vision field. CNN had began the new era of deep learning by allowing parameter sharing and sparsity of connections, which serve as the basis of all the one-stage and two-stage methods that mentioned above.

NMS, stands for Non-Maximum Suppression, is a technique that used to eliminate redundant bounding boxes for the same object generated by the network. There are also many other NMS versions, for instance, Soft-NMS, DIoU NMS, Conv NMS, Learning NMS, etc. The invention of NMS had greatly enhanced accuracy of the algorithms, which make it remains necessary component of modern network.

Moreover, some new strategies that used to applied in other filed are now put into experiment in object detection and yield remarkable results.

Adversarial training, or GAN (Generative Adversarial Networks) [286] is quite popular these days especially in researches that allowing AI to generate their own production (composing songs, draw paintings, etc.). Typically, GAN includes two networks, one “teacher” network to criticize the AI production, while another one “student” network generate AI production, two networks will learn together and yields better results after training. For object detection task, GAN had been put into usage to enhance the performance when detecting small or overlapping objects.

Inspite of the great progress that object detection ever had since start, there are still some serious challenges that had troubled many researchers till now, for instance, when detecting bad weather situation (strongly snowy, foggy, etc.), or many small targets overlapping (a really busy street with hundreds of pedestrians) performance of algorithms normally are [barely](javascript:;) [satisfactory](javascript:;). To solve these problems above, further researches are needed to enable object detection techniques to applied in more situations to provide people better lives. For the future, object detection researches may focused more on real-time techniques, which could provide video surveillance on road or autonomous cars. Therefore, video instead of images would become the mainstream media to apply object detection, which means more fast but accurate algorithms will be developed to serve the needs. Object detection application that in smartphones is also a promising direction, which may also trigger the development of more light-weight models that could used in mobile devices. It is reasonable to believed that in the future, object detection techniques will be more important in people’s lives than ever.

## 1.3 Description of Software Prototypes

This project is focused on developing a website that allow users to experience object detection conveniently. The software prototype of the project can be divided into three parts:

1. Application that could provide service for object detection tasks User can upload images, and view results in the website, including object classification, location, confidence, size, object number, etc.
2. Users are allowed to select different trained weights flexibly for object detection tasks.
3. Application that could connect to database, where stored user registration information. User can also bookmark their results into collection, and remove them through interface.

**Chapter 2**

**Methods**

## 2.1 Justification of YOLO

The traditional method -- sliding window algorithm use different sizes of boxes to go through the image step by step. However, this method has a serious drawback: to obtain higher accuracy, the stride of the boxes need to be smaller, which needs incredibly huge amount of computational resource and costs a long time. Moreover, two potential problems exist in this method:

* Different sizes of boxes means different sizes of inputs. Thus, normalization process need to be added int the network.
* Since the method will go through the whole image in brute-force way, the background areas must larger than the object areas, which will cause imbalanced data (imbalance between positive and negative samples).

To solve the problem above, the origin of YOLO is designed by simply transforming a classification detector into an object detector that could predict objects locations. The traditional classification network normally ends with a fully-connected layer outputs N dimensional one-hot vector, so the author of YOLO just change the output layer into an another vector (x,y,w,h,c) (x and y denotes the coordinates location of the top-left location of the bounding box, w and h means the width and height the bounding box, c is the confidence of the object), forming a object detector.

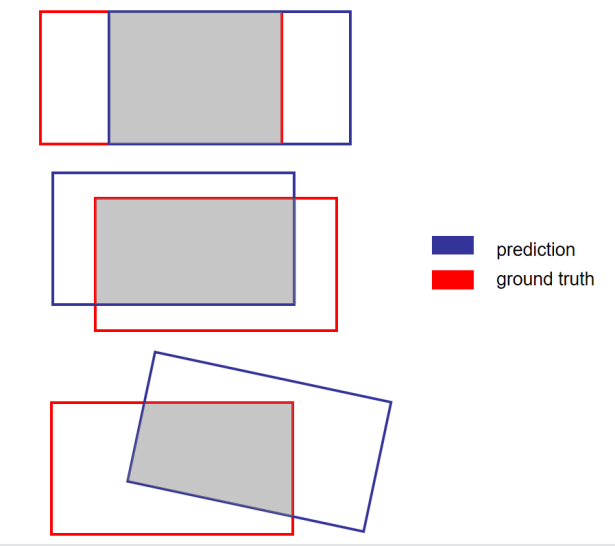
From the description above, YOLO-v0 can only output one object for one image, in order to detect multiple objects in one image, YOLO-v1 was introduced. First of all, instead of output only one set of one-hot vector (x,y,w,h,c), the network divided the image into 7\*7 in total 49 regions, one region is corresponding for one set one-hot vector output. Moreover, if one object spanned more than one regions, NMS is used to select the most confident region and give out prediction. If there are multiple classes, YOLO-v1 just simply increase the output, and repeat the 7\*7 region detect for each class. To solve the problem of detecting small objects, YOLO-v1 specifically add extra neuron network layers to handle them, which means there are two sets of one-hot vector (in total 98 bounding boxes), one for big targets while the other handling small objects. For the other components, YOLO-v1 takes GoogleNet as the backbone without neck, which belongs to dense prediction detector. For training, YOLO-v1 trains the classification network with 224\*224 resolution rate, then trains the detection network with 448\*448 resolution rate.

Nevertheless, YOLO-v1 still had some problems left unsolved. Despite the fast speed, the accuracy of the network is not satisfying, and the recall is relatively low which means many targets are missing.

To tackle the first problem, YOLO-v2 used anchor box for width and height, which was first introduced in R-CNN, aiming predict relative offsets instead absolute offsets. Because the offset after regularization is smaller than the original height and weight values, the network can learn better and provide better accuracy. For the second problem, YOLO-v2 evlove the network structure from 7\*7 to 13\*13 to promote recall, with maximum 169 objects could be detected. Even though YOLO-v2 still does not have a neck component, the backbone is switched to darknet-19, and fully-connect layer is substitued by GAP (Global Average Pooling) layer to enhance the accuracy for small objects. The training process for YOLO-v2 is also much more complicated: first of all, Darknet-19 is pre-trained in the ImageNet classification data set for 160 epochs with 224\*224 resolution rate, then they fine-tune the classification model for 10 more epochs with 448\*448 resolution rate, since the usage of GAP allow training could accept different size of inputs.

Even though YOLO-v1 added an extra output layer to detect small objects, the result was still not promising. The author of YOLO noticed this and changed CNN down-sampled rate into three branches: 32x-down sampling, 16x-down sampling, and 8x-down sampling, respectively detects for big, medium and small objects, in total generate 10467 bounding boxes, way larger than YOLO-v2 845 bounding boxes. YOLO-v3 also uses FPN as the neck, which could pass the feature information from bottom layer to upper layer, For the other components, YOLO-v3 updates the backbone to Darknet-53 with the introduction of Resnet, which makes the network deeper, contributes for higher accuracy in a large degree.

YOLO-v4 improved the head of the network by using multiple anchors to predict single ground truth, which could increase the number of positive samples, aiming at mitigate the problem of imbalance between the positive and negative samples. Moreover, from v0 to v3, YOLO adopts the traditional MSE (Mean Squared Error) loss when calculating loss function, but MSE loss can not tell the difference overlapping area or IOU are the same (Figure2.1.1).



**Figure 2.1.1** Same IOU but Different MSE Loss

Therefore, CIoU-loss is used to calculate the central point distance between ground truth and prediction. In the basis of YOLO-v3, YOLO-v4 further enhance the network structure in the neck, by adding SPP module, allows the multi-scale integration for pooling, and PAN structure to concatenate adjacent feature layers for prediction. Besides, YOLO-v4 also put more efforts in refining the inputs by using Mosaic, which could largely enhance data richness, leads to better network robustness.

Finally, YOLO-v5 was released in 2020. Compare to earlier version, YOLO-v5 adopts adaptive anchor, which allows anchor box could learn with the network, so previously predefined fixed (x,y,w,h) value will change by the learning process to compare with ground truth boxes, then update network parameters to obtain better training results. Therefore, the loss function also changed into GIoU loss. YOLO-v5 also adopts Focus module in backbone, which slice the data into 4 groups then concatenate them with channel, executing down-sampling procedure without largely losing information. The most interesting part of YOLO-v5 is that there are four options of pre-trained weights for choosing: s (small),m (medium),l (large),x (extra-large). The larger the weight is, the higher precision goes, while the processing time also increase. These four weights are the results of different parameters in network depth, width, and different number of res unit when training, which results in different numbers of convolutional kernel in each layer. At last, YOLO-v5 also use adaptive image re-scale module to improve detection speed.

## 2.2 Data Set and Training

Since Pascal VOC is one of the most important data set for object detection, so there are already benchmarks created by other famous algorithms. Moreover, with relatively smaller size than others, Pascal VOC is suitable for this lightweight project. This project finally selects VOC 2007 for training, because VOC 2007 focused on object detection tasks, but the latter version (VOC 2012) also used for other tasks (instance segmentation and object tracking). Despite its small size, it also has enough data to train a decent model, with 9963 images (train, validation and test), in total 24640 objects for 20 different classes.

Before training the model with the chosen data set, format transform code is needed to convert the original VOC format (xml format) into compatible YOLO version (txt format).(see Appendix?) Moreover, data set subdivision is also necessary to divide the whole data set into three parts: train, validation, and test, which is important to obtain better training results. Moreover, this project needs some dependency to install through pip: Flask, Pytorch, torchvision, torchaudio, cudatoolkit, SQL-Alchely, numpy. Other packages used in front-end could be installed through npm: ant-design-vue, axios, echarts, element-ui, jQuery, socket.io, vue.

In the training phase, the whole process was carried on by the server in school laboratory through SSH, equipped with Anaconda to prepare virtual environment with Python 3.8, the hardware configuration with driver version is listed in Table 2.2.1:

|  |  |
| --- | --- |
|  | **Specification** |
| **GPU** | 4 NVIDIA GTX TITAN X |
| **GPU Memory** | 12212MB (12GB) per GPU |
| **CUDA Version** | 11.2 |
| **Driver Version** | 460.80 |
| **CPU** | Intel Core i7-5930K 3.50GHz |
| **Memory** | 32834268KB(32GB) |

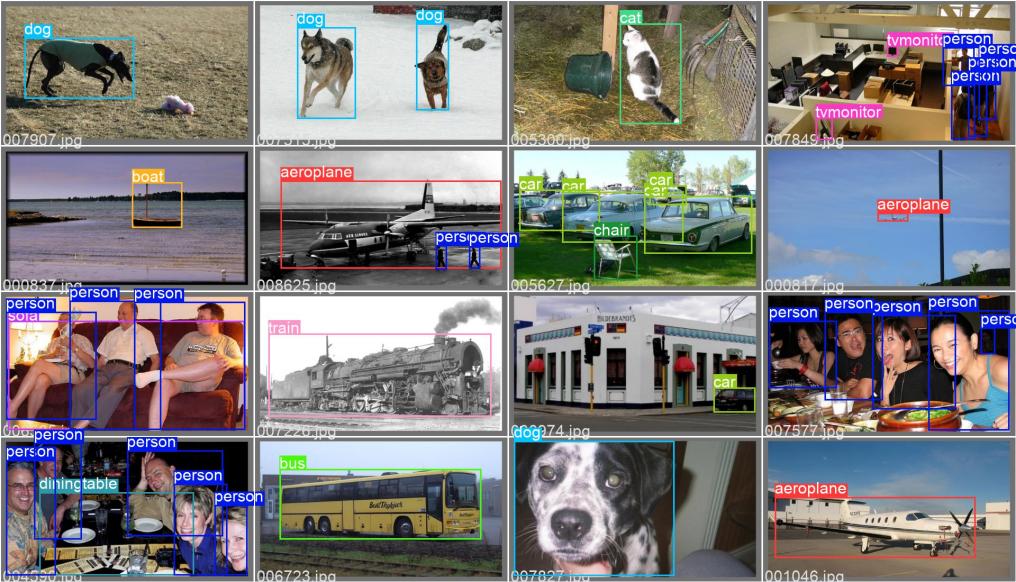
**Table 2.2.1** Training Hardware Configuration

In the developing phase, the application is running in personal laptop, the hardware configuration with driver version is listed in Table 2.2.2:

|  |  |
| --- | --- |
|  | **Specification** |
| **GPU** | NVIDIA GTX 1070 |
| **GPU Memory** | 8192MB (8GB) |
| **CUDA Version** | 11.6 |
| **Driver Version** | 511.79 |
| **CPU** | Intel Core i7-7700HQ 2.80GHz |
| **Memory** | 16659528KB(16GB) |

**Table 2.2.2** Developing Hardware Configuration

The following picture shows temporary results when training weight size ***m*** during last 10 epochs, using visualization tool ***wandb***:

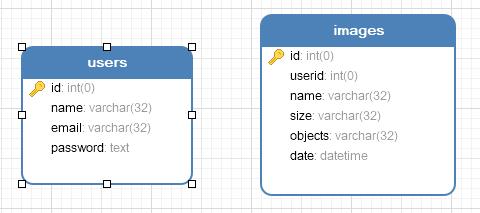


**Figure 2.2.1** Temporary Training Results in Wandb

All four weights (s,m,l,x) were trained until reaching the convergence loss, so there exists differences in training epochs for different weights, more details will be discussed in Chapter 3.

## 2.3 Front-end, Back-end, and Data base

This project adopts MySQL as database, and use SQL-Alchemy to connect Python Flask with MySQL. Since MySQL is a medium-weight relational database, it is suitable for this project, and it is also quite convenient to connect MySQL API through Python. There are two tables in total, *users* and *images* table designs are shown below:



**Figure 2.3.1** Database ER Diagram

In the ***users*** table, there are 6 fields with constraints:

|  |  |  |  |
| --- | --- | --- | --- |
| **Names** | **Type** | **Description** | **Constraints** |
| **id** | Integer | User Identification Number | Primary Key, Auto Increment |
| **name** | Varchar | User Name | Unique, 32 bits |
| **email** | Varchar | User Email | Unique, 32 bits |
| **password** | Text | User Password (SHA-256) | None |

**Table 2.3.1** User Table Constraints

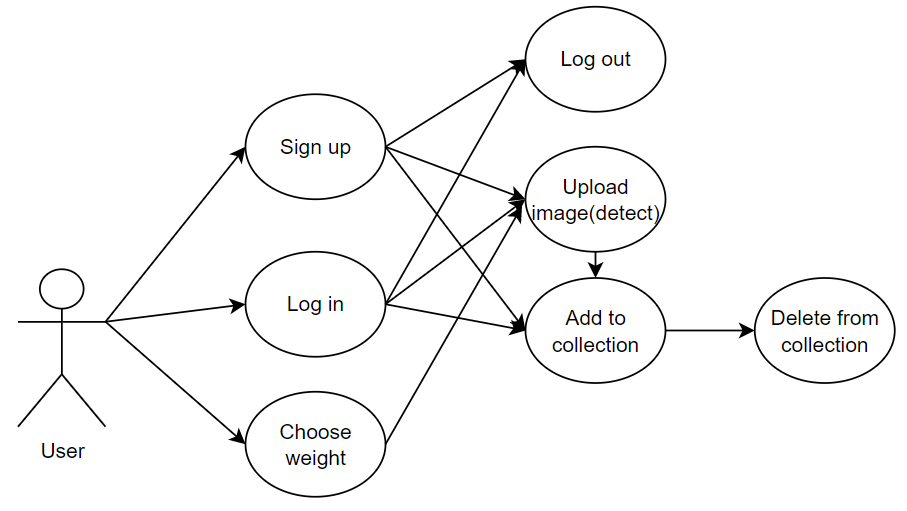
In the ***images*** table, there are 4 fields with constraints:

|  |  |  |  |
| --- | --- | --- | --- |
| **Names** | **Type** | **Description** | **Constraints** |
| **id** | Integer | Image Identification Number | Primary Key, Auto Increment |
| **userid** | Integer | User Identification Number | Nullable |
| **name** | Varchar | Image Name | 32 bits |
| **size** | Varchar | Image Size(MB) | 32 bits |
| **objects** | Varchar | Object Number | 32 bits |
| **date** | Datetime | Time Added To Collection | Nullable |

**Table 2.3.2** Images Table Constraints

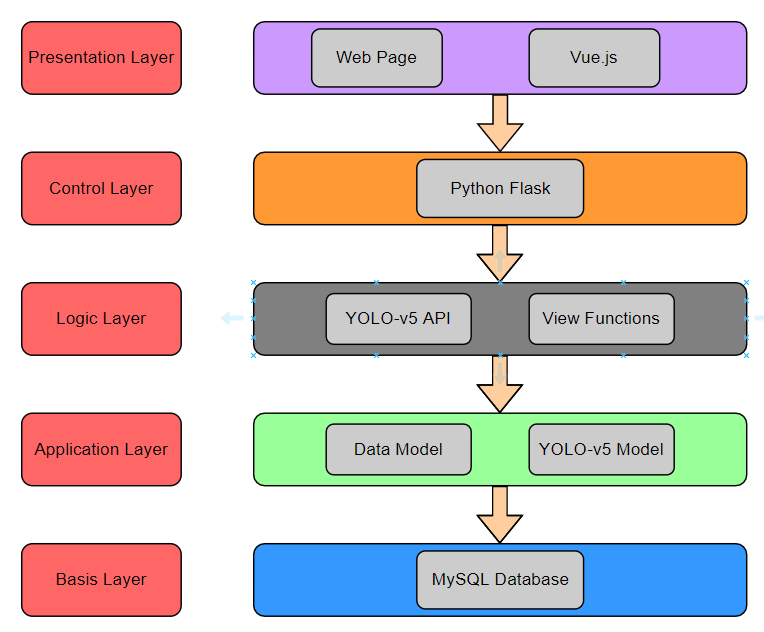
## 2.4 Functionality

The user has three options when first start the application: sign up, login, and select weight for object detection tasks. User could skip logging and use object detection functionality directly by upload the image. However, if the user logged in, object detection results are allowed to add to user’s personal collection, deletion is also permitted after login. The following use case diagram (Figure 2.4.1) shows the process:



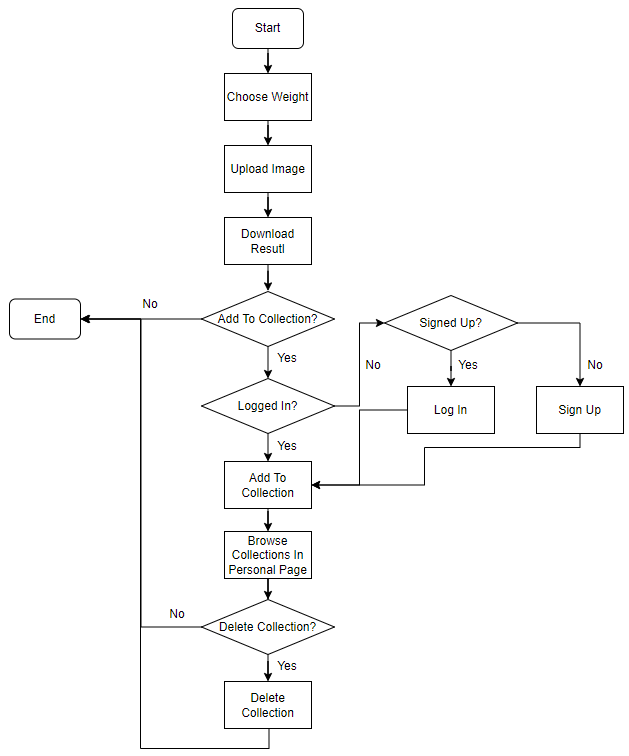
**Figure 2.4.1** Use Case Diagram

The architecture of the system could be divided into 5 layers separately (Figure 2.4.2): Presentation Layer, Control Layer, Logic Layer, Application Layer, and Basis Layer. From the Presentation Layer, web pages are rendered by Vue.js, where users could interact with the website and make requests through routers. Then, Python Flask in the Control Layer will handle the requests with the help of Logic Layer components (view functions and YOLO-v5 API) integrated in the code. After that, Data model and trained YOLO-v5 model in the Application Layer will be loaded for data storage and object detection tasks respectively. Finally, the data will be transmitted from the back-end to MySQL database in Basis Layer.



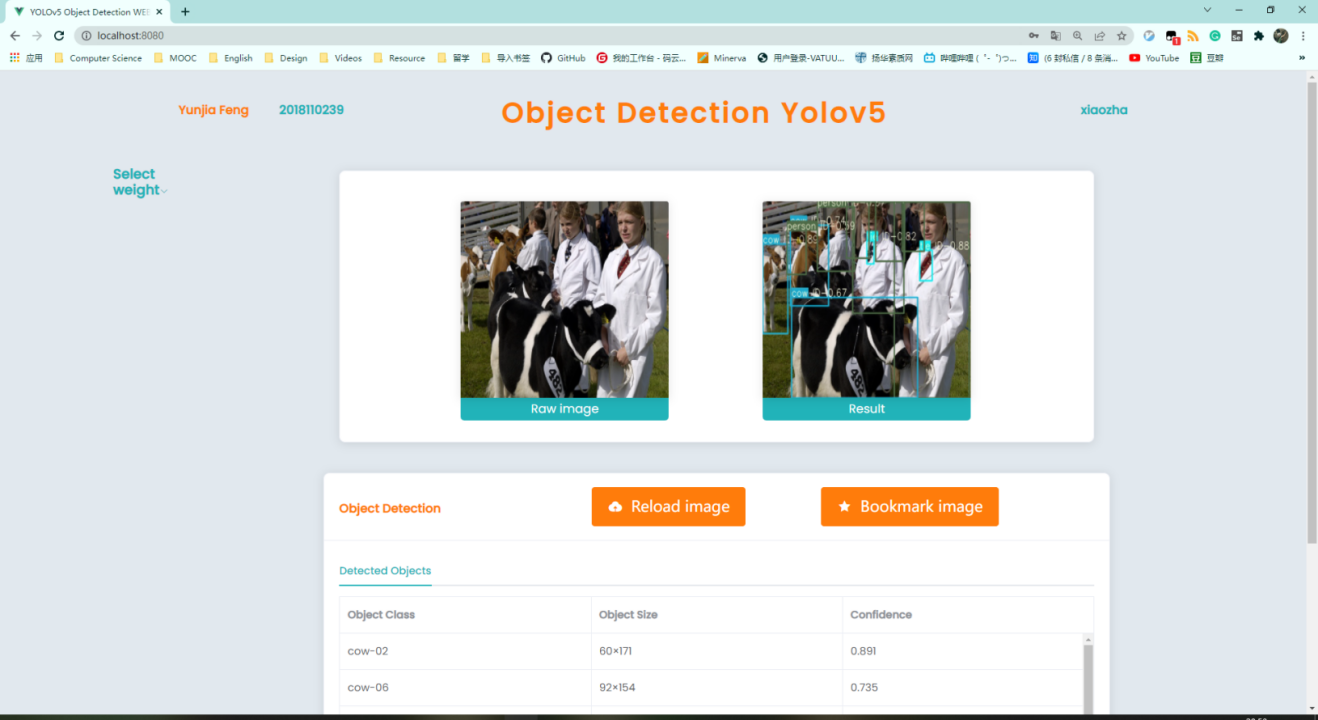
**Figure 2.4.2** Project Architecture Diagram

User could select weight (s,m,l,x) at first, then upload image for object detection tasks and download results. If the user want to bookmark the result to collection, then user need to log in or sign up. After bookmarking the image, user could continue browsing the remaining collection in their personal page where they could also delete them. The following flow diagram (Figure 2.4.3) shows the details when interacting with the website:



**Figure 2.4.3** Flow Diagram

The Figure 2.4.4 shows the layout of application page, where users could select 4 different weights in the top left corner, while top right side indicates the current logged in username. User could upload or reload images and add results to collection through two buttons in the middle, the results will show both by image and details(object class, size and confidence) below the two buttons.



**Figure 2.4.4** Application Page

## 2.5 Version Control and Project Management

Version control system evidence,

good practice in file structure

Details of project management methodology (列出sprints) ? Agile??? 甘特图？

**Chapter 3**

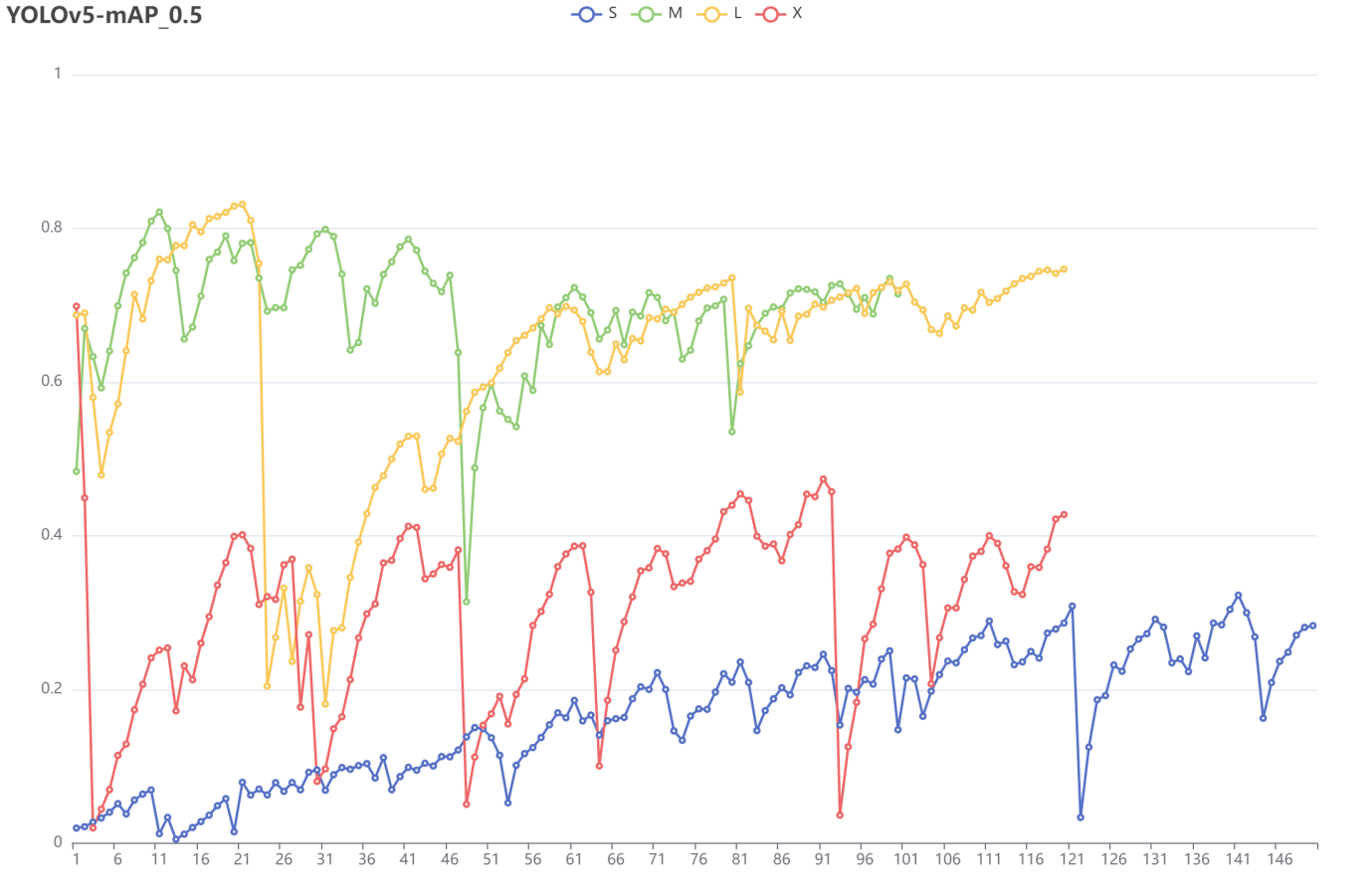
**Results**

## 3.1 Evaluation and Test Results

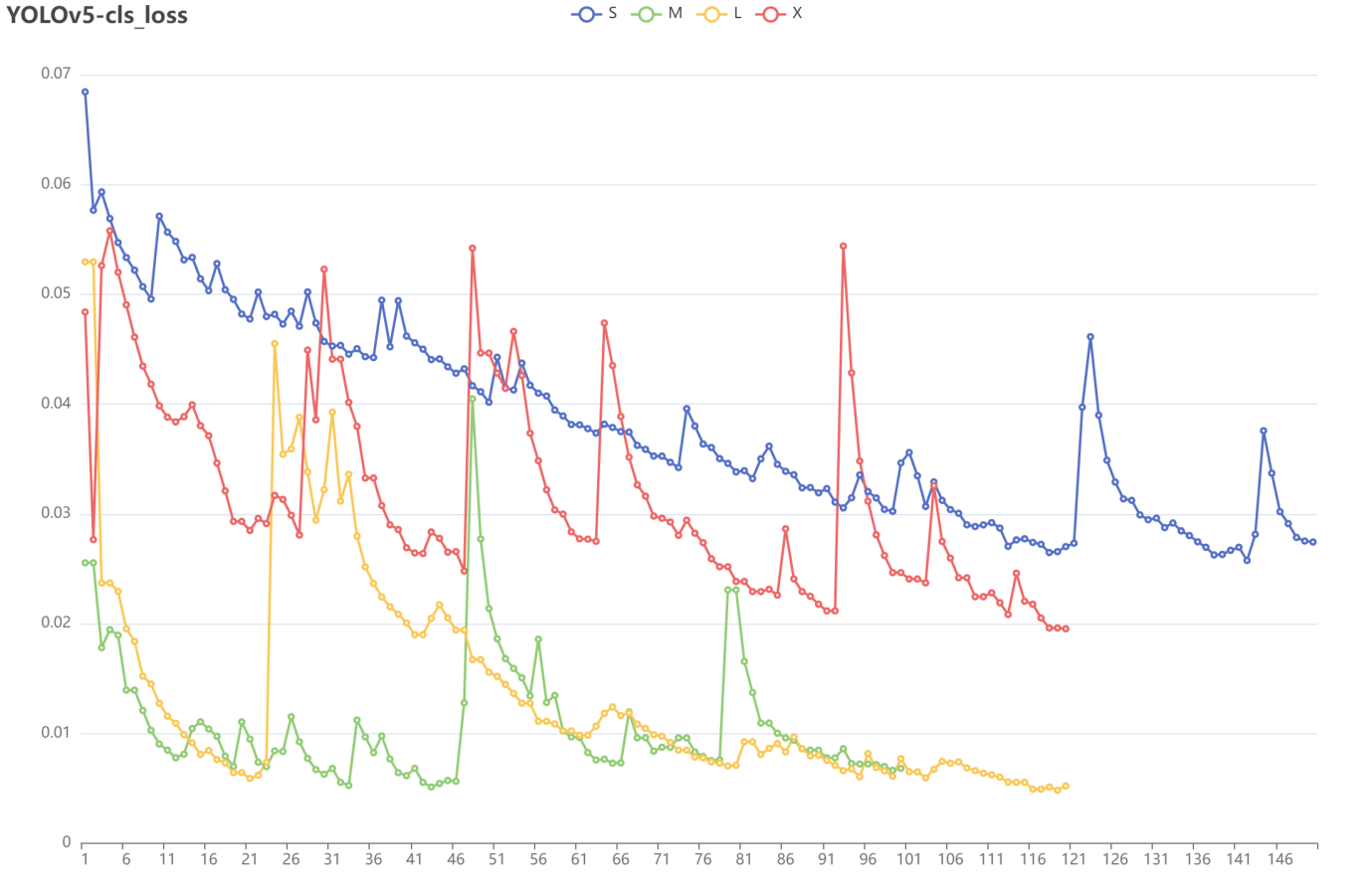
The Table 3.1.1 shows the training evaluation metrix, indicators including mAP, Class loss, Box loss, training epochs and speed (milliseconds per image) of four weights respectively.

**Table 3.1.1** Training Evaluation Metrix

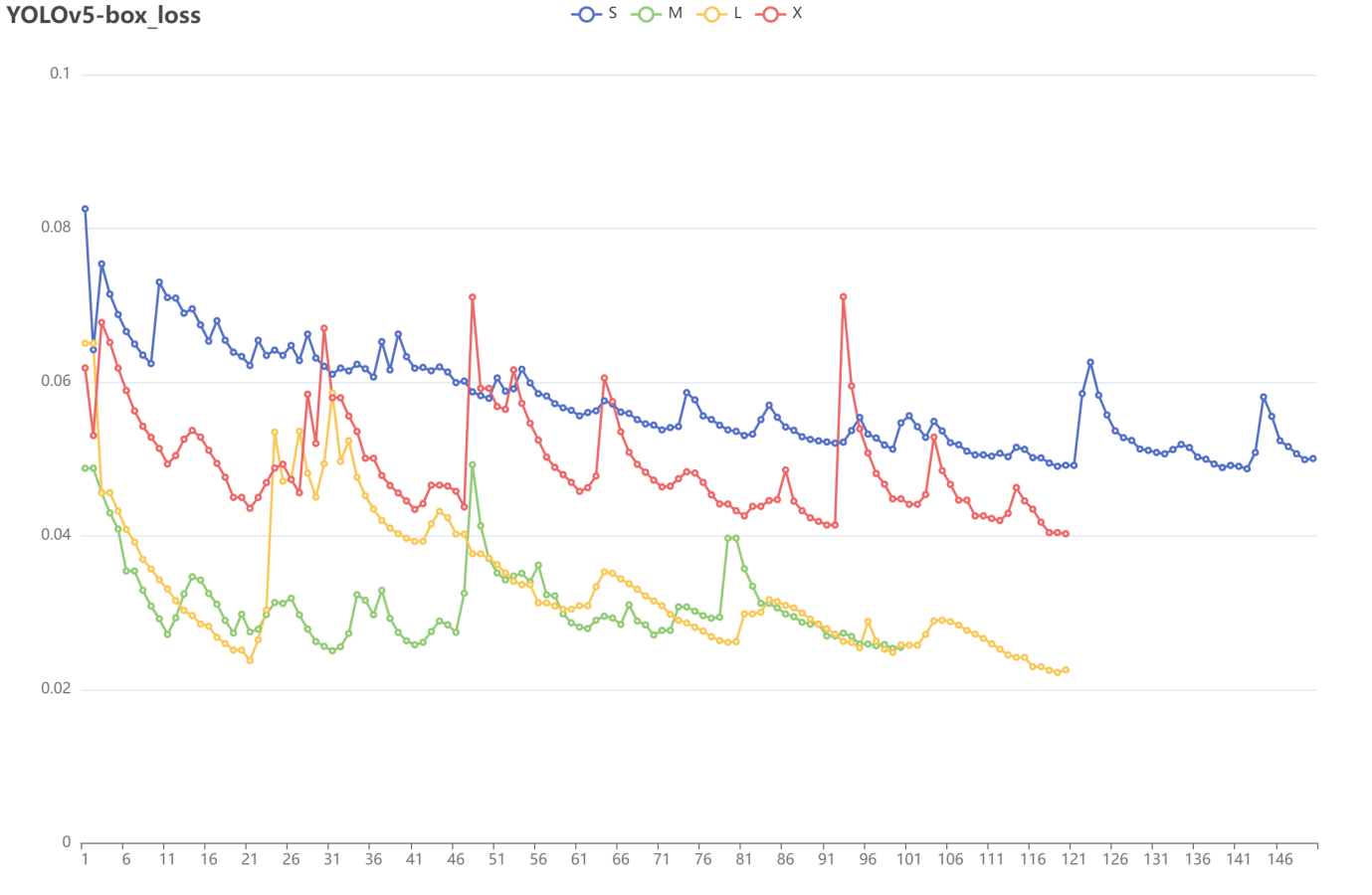
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **YOLO-V5** | **Small** | **Medium** | **Large** | **Extra Large** |
| **mAP(0.5)** | **0.2831** | **0.7147** | **0.7470** | **0.4277** |
| **Cls\_loss** | **0.0274** | **0.0068** | **0.0052** | **0.0195** |
| **Box\_loss** | **0.0500** | **0.0255** | **0.0225** | **0.0402** |
| **Train(epoch)** | **150** | **100** | **120** | **120** |
| **Speed(ms)** | **0.33** | **0.46** | **0.59** | **0.69** |

The Following Figures shows the trend of mAP (threshold > 0.5), class loss, and box loss of 4 different weights (Figure 3.1.1, 3.1.2, 3.1.3 respectively). 

**Figure 3.1.1** YOLO-v5 mAP(threshold > 0.5) for 4 Weights



**Figure 3.1.2** YOLO-v5 Class Loss for 4 Weights



**Figure 3.1.3** YOLO-v5 Box Loss for 4 Weights

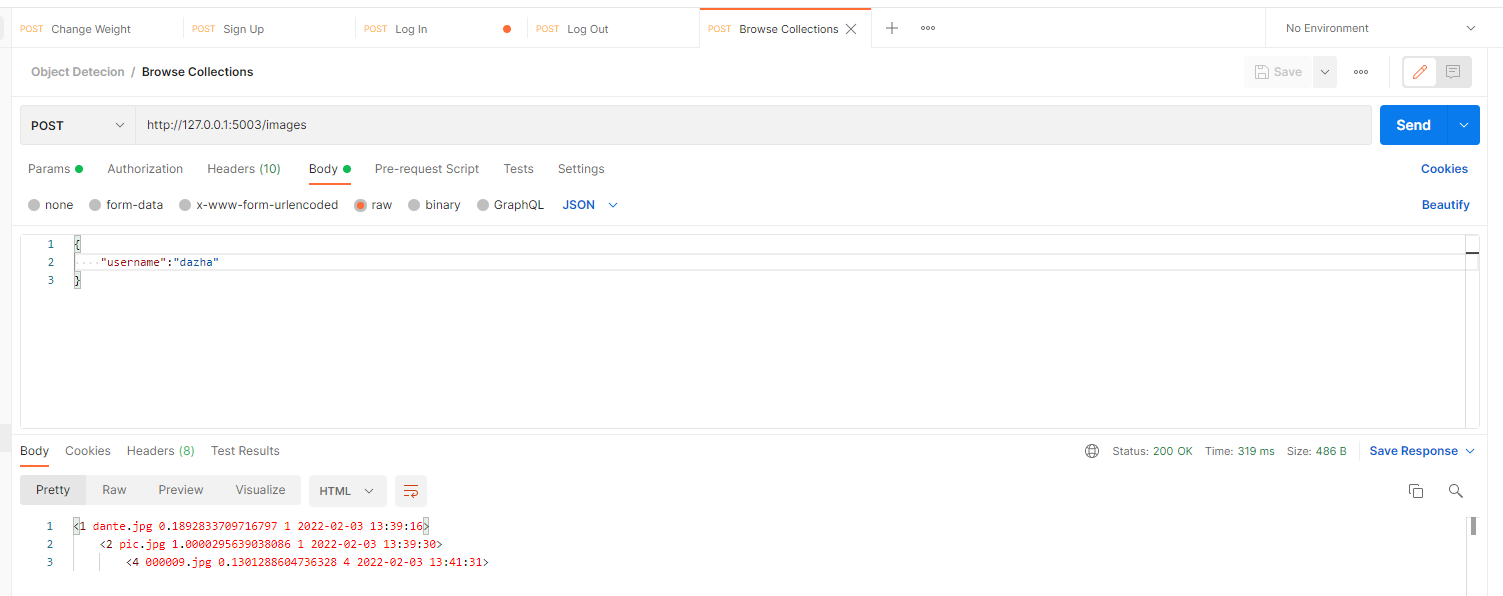
From the figures above, it is clearly to denote that weight M and L have higher mAP and lower losses than S and X.

The Table 3.1.2 shows unit test results of 6 different APIs using 3 Test tools: Postman (API test), Selenium (graphical user interface test), and Pytest (unit test with Python code).

|  |  |  |  |
| --- | --- | --- | --- |
| **API\Tool** | **Postman** | **Selenium** | **Pytest** |
| **Log In** | Pass√ | Pass√ | Pass√ |
| **Log Out** | Pass√ | Pass√ | Pass√ |
| **Sign Up** | Pass√ | Pass√ | Pass√ |
| **Browse Collection** | Pass√ | Pass√ | Pass√ |
| **Add Bookmark** | Pass√ | Pass√ | Pass√ |
| **Remove Bookmark** | Pass√ | Pass√ | Pass√ |

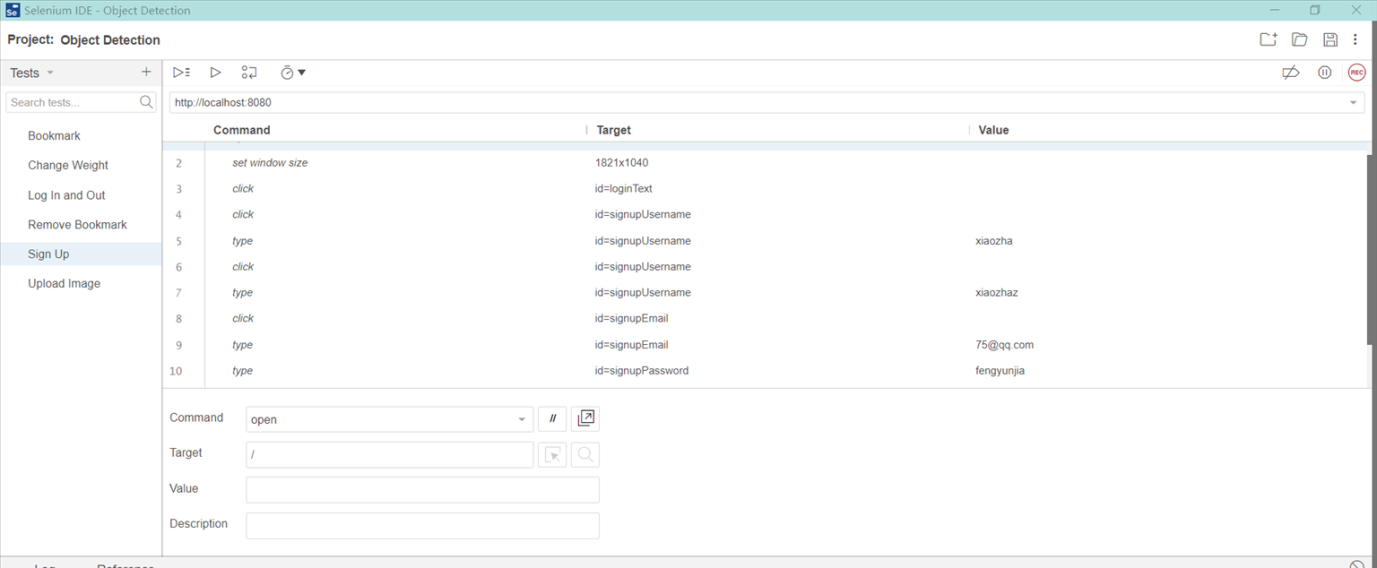
**Table 3.1.2** Unit Test Results

The Figure 3.1.4 shows an example when testing API browsing collections in Postman by submitting username in the request, the server will return response with image and detail information (image name, size, object number, creating date time), other API were also tested through Postman (see Appendix?).



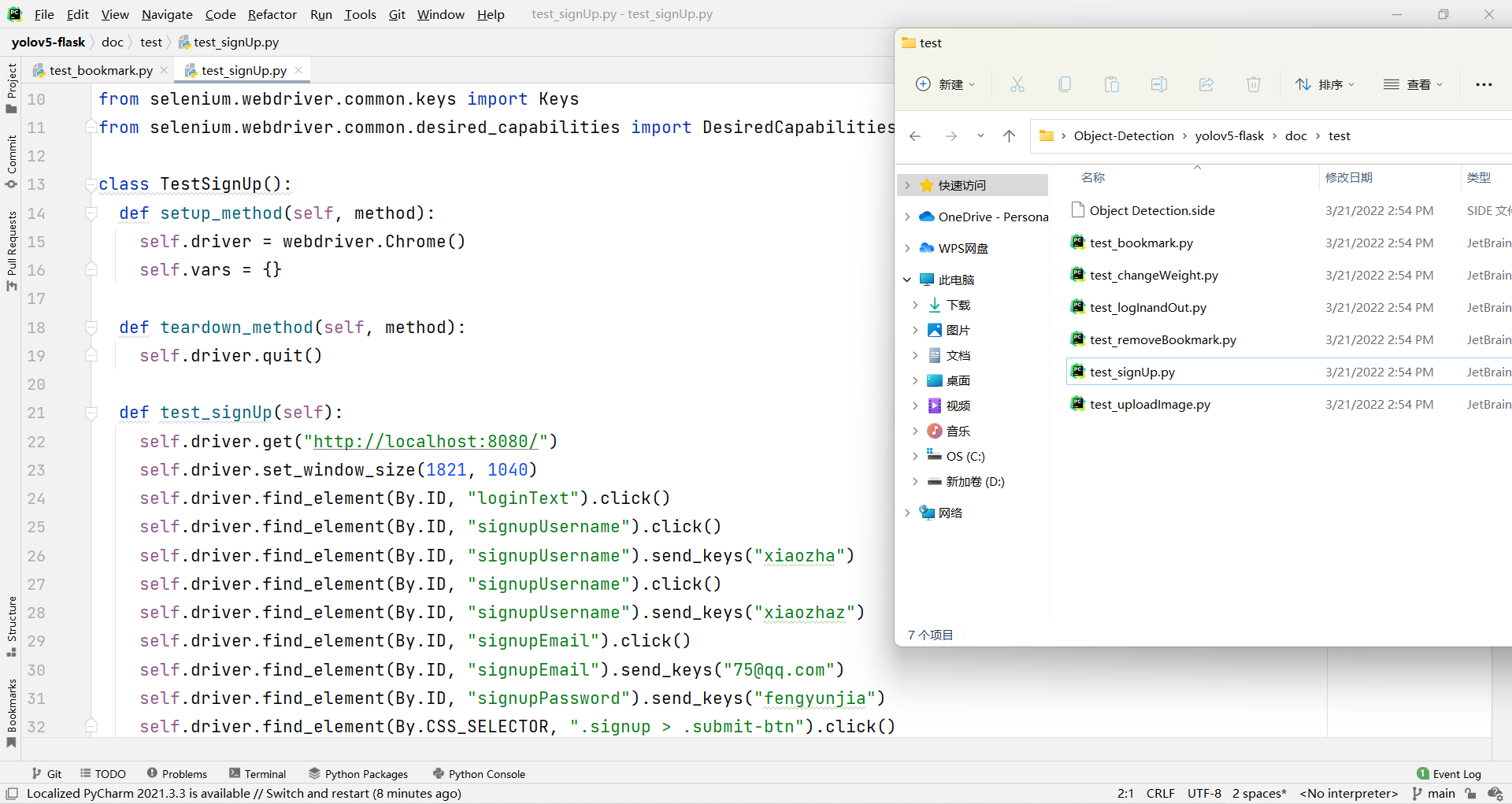
**Figure 3.1.4** Postman Test for API Browsing Collections

The Figure 3.1.5 shows an example when testing API sign up in Selenium, which is a test tool simulating user behaviour through browsers. Other API were also tested through Selenium (see Appendix?).



**Figure 3.1.5** Selenium Test for API Sign Up

The Figure 3.1.6 shows an example when testing API sign up in Pytest, which is a test tool sending requests to the server in Python. Other API were also tested through Selenium (see Appendix?).



**Figure 3.1.6** Pytest for API Sign Up

## 3.2 Comparing with Other Methods

The following figure shows state of art algorithms in 2007 with their mAP performance rank in VOC-2007 competition[yolov1 reference], the reason why comparing with some algorithms that were not used in recent year is that some new methods only shows their test results in COCO instead of VOC-2007.(引用VOC2007?) Comparing with these old methods, it is clear to see that models in this project is way better than the others.



**Figure 3.2.1** Comparing Old Methods Test Results Before YOLO-v1

The following Table 3.2.1 shows different methods(Selective Search, RPN+VGG) in SSD test in VOC-2007 and VOC-2012 (07 denotes for VOC-2007, 12 denotes for VOC-2012 in column data). Only comparing with gray row, which is trained by VOC-2007, model in this project still has some advantage over other methods.(引用SSD)

**Table 3.2.1** Comparing Results with SSD

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Proposals** | **data** | **mAP (%)** |
| **Selective Search** | 2000 | 07 | 66.9 |
| **Selective Search** | 2000 | 07+12 | 70.0 |
| **RPN + VGG, unshared** | 300 | 07 | 68.5 |
| **RPN + VGG, shared** | 300 | 07 | 69.9 |
| **RPN + VGG, shared** | 300 | 07+12 | 73.2 |
| **RPN + VGG, shared** | 300 | COCO+07+12 | 78.8 |
| **This Project** |  |  | 74.7 |

The following Table 3.2.2 shows YOLO-v1 and Fast R-CNN test in VOC-2007 and VOC-2012. Comparing with Fast R-CNN trained by VOC-2007, there is about 7% mAP improvement in this project model.(引用YOLO-v1)

**Table 3.2.2** Comparing with Faster R-CNN in VOC-2007

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **mAP (%)** | **Combined** | **Gain** |
| **Fast R-CNN** | 71.8 | - | - |
| **Fast R-CNN (VOC-2007)** | 66.9 | 72.4 | .6 |
| **Fast R-CNN (VGG-M)** | 59.2 | 72.4 | .6 |
| **Fast R-CNN (CaffeNet)** | 57.1 | 72.1 | .3 |
| **YOLO (old version)** | 63.4 | 75.0 | 3.2 |
| **This Project** | 74.7 | - | - |

The following Table 3.2.3 shows Fast R-CNN, Faster R-CNN test and SSD300, SSD512 in VOC-2007 and VOC-2012(07 denotes for VOC-2007, 12 denotes for VOC-2012 in column data). Comparing with method trained by VOC-2007, model in this project outweigh Fast R-CNN, Faster R-CNN, and SSD500. (引用Faster R-CNN?)

**Table 3.2.3** Comparing with Fast R-CNN, Faster R-CNN, SSD-300, SSD-512 in VOC-2007

|  |  |  |
| --- | --- | --- |
| **Method** | **data** | **mAP (%)** |
| **Fast R-CNN** | 07 | 66.9 |
| **Fast R-CNN** | 07+12 | 70.0 |
| **Faster R-CNN** | 07 | 69.9 |
| **Faster R-CNN** | 07+12 | 73.2 |
| **Faster R-CNN** | 07+12+COCO | 78.8 |
| **SSD300** | 07 | 68.0 |
| **SSD300** | 07+12 | 74.3 |
| **SSD300** | 07+12+COCO | 79.6 |
| **SSD512** | 07 | 71.6 |
| **SSD512** | 07+12 | 76.8 |
| **SSD512** | 07+12+COCO | 81.6 |
| **This Project** | 07 | 74.7 |

OR Technical /user evaluation 可以写给同学用？

**Chapter 4**

**Discussion**

## 4.1 Conclusions

<Text in 11-point size and 1.5 line spacing.>

## 4.2 Ideas for future work

There still exists some deficiencies in this project, which needs future work for improvement.

1. All the training in this project are taken by 10-30 consecutive epochs at one time instead of training for 100 epochs or until reaching the convergence. Although there are not solid evidence shows huge differences between these two methods, it is believed to train the model for once. However, due to the hardware limitation, school laboratory can not support to do that. In the future, high performance machine equipped with larger GPU memory or using online laboratory could eliminate this problem for all.
2. This project is running in local host instead of providing web service because it needs server to equip high performance Nvidia display card which most current server renting service could not provide. The future work for this project may need to rent high performance server for machine learning purpose specifically to provide users using the service online.
3. This project now only support object detection tasks for image. However, YOLO API can also support video and computer camera. The reason why this project abandoned video object detection is because it takes long time for video to upload and server to process even in local host situation. To overcome this problem, lower network latency and larger network bandwidth are needed in the future.
4. This project now is only focusing on developing YOLO for object detection tasks. However, other state-of-art algorithms also have their own strength and benefits in different situations. To provide users better experience and choices, multiple options to use different algorithms may be the direction for future development. Nevertheless, this need extra workload to implement multiple API to use different algorithms, and also costs time and performance of server since switching between different algorithm need bootstrap and load models from scratch.

# List of References

*<It is expected that the list would reflect the breadth and depth of scholarly research undertaken by the student during the course of the project.>*

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in Advances in neural information processing systems, 2012, pp. 1097–1105.

[2] Vishwakarma S, Agrawal A. A survey on activity recognition and behavior understanding in video surveillance [J]. The Visual Computer, 2012: 1-27.

[3] Jiang, B., Luo, R., Mao, J., Xiao, T., & Jiang, Y. (2018). Acquisition of localization confidence for accurate object detection. In ECCV (pp. 784–799).

[4] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In Proc. Eur. Conf. Comp. Vis., pages 734–750, 2018.

[5] Liu, Wei, et al. "SSD: Single Shot MultiBox Detector." European Conference on Computer Vision Springer International Publishing, 2016:21-37.S

[6] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg. DSSD: Deconvolutional single shot detector. arXiv:1701.06659, 2016. 1, 2, 8

[7] Z. Li and F. Zhou. Fssd: Feature fusion single shot multibox detector. arXiv preprint arXiv:1712.00960, 2017.

[8] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In CVPR, 2014.

[9] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In NIPS, 2015.

[10] Cai, Z., Vasconcelos, N.: Cascade r-cnn: Delving into high quality object detection. arXiv preprint arXiv:1712.00726 (2017)

[11] Redmon, Joseph, et al. "You Only Look Once: Unified, Real-Time Object Detection." (2015):779-788.

[12] Alexey Bochkovskiy, Chien-Yao Wang, and HongYuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934, 2020.

[13] Joseph Redmon and Ali Farhadi. YOLOv3: An incremental improvement. arXiv preprint arXiv:1804.02767, 2018.

[14] J. Redmon and A. Farhadi. YOLO9000: Better, faster, stronger. In CVPR, 2017.

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

[1] B. Hariharan, P. Arbel´aez, R. Girshick, and J. Malik, “Simultaneous detection and segmentation,” in European Conference on Computer Vision. Springer, 2014, pp. 297–312.

[2] ——, “Hypercolumns for object segmentation and fifine grained localization,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 447–456.

[3] J. Dai, K. He, and J. Sun, “Instance-aware semantic segmentation via multi-task network cascades,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3150–3158.

[4] K. He, G. Gkioxari, P. Doll´ar, and R. Girshick, “Mask rcnn,” in Computer Vision (ICCV), 2017 IEEE International Conference on. IEEE, 2017, pp. 2980–2988.

[8] K. Kang, H. Li, J. Yan, X. Zeng, B. Yang, T. Xiao, C. Zhang, Z. Wang, R. Wang, X. Wang et al., “T-cnn: Tubelets with convolutional neural networks for object detection from videos,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 28, no. 10, pp. 2896–2907, 2018.

[9] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” nature, vol. 521, no. 7553, p. 436, 2015.

[10] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on, vol. 1. IEEE,

2001, pp. I–I.

[11] P. Viola and M. J. Jones, “Robust real-time face detection,” International journal of computer vision, vol. 57, no. 2, pp. 137–154, 2004.

[12] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, vol. 1. IEEE, 2005, pp. 886–893.

[21] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y.Fu, and A. C. Berg, “Ssd: Single shot multibox detector,” in European conference on computer vision. Springer, 2016,

pp. 21–37.

[23] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Doll´ar, “Focal loss for dense object detection,” IEEE transactions on pattern analysis and machine intelligence, 2018.

[24] R. Girshick. Fast R-CNN. In ICCV, 2015.

[50] M. Everingham, L. Van Gool, C. K. Williams, J. Winn,and A. Zisserman, “The pascal visual object classes (voc) challenge,” International journal of computer vision, vol. 88, no. 2, pp. 303–338, 2010.

[51] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes challenge: A retrospective,” International journal of computer vision, vol. 111, no. 1, pp. 98–136, 2015.

[96] R. Vaillant, C. Monrocq, and Y. Le Cun, “Original approach for the localisation of objects in images,” IEEProceedings-Vision, Image and Signal Processing, vol. 141, no. 4, pp. 245–250, 1994.

[286] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in Advances in neural information processing systems, 2014, pp. 2672–2680.

[Social Issues]Berle, Ian. "The Future of Face Recognition Technology and Ethico: Legal Issues." Face Recognition Technology. Springer, Cham, 2020. 163-184.

## Appendix A

## Self-appraisal

<This appendix must contain everything covered under the ’self-appraisal’ criterion in the mark scheme. Although there is no length limit for this section, 2-4 pages will normally be suﬃcient. The format of this section is not prescribed, but you may like to consider the following sections and subsections.>

## A.1 Critical self-evaluation

## A.2 Personal reﬂection and lessons learned

## A.3 Legal, social, ethical and professional issues

<Refer to each of these issues in turn. If one or more is not relevant to your project, you should still explain *why* you think it was not relevant.>

### A.3.1 Legal issues

<Discussion of legal issues>

[Social Issues引用]

In object detection tasks, face recognition is one of the most important application. However, this technique is also controversial, which may raise some legal issues. The most common problems for face recognition technology are on privacy and confidentiality, since biological features like faces had been largely used in modern society to replace simple password for identification. Moreover, exclude commercial application mentioned above, video surveillance is also widely adopts for security concerns, aiming at stopping potential terrorism or accident monitoring.

Despite the purpose of monitoring is decent, the right to protect privacy is also important to individuals, and there still lacks impeccable legal regulations or law support in this burgeon field. To solve this legal issues need constant efforts from governments and legislation departments.

### A.3.2 Social issues

### <Discussion of social issues>

Still for face recognition task, the strong desire to protect privacy for citizens will cause social concerns and sensitivities. As A.3.1 had mentioned, the main applications for this task can be divided into two parts: commercial usage and video surveillance by government.

For the commercial side, using biometric features could bring convenience to users and reduce identification theft at some degree. However, the data storage for sensitive face features remains a serious problem. It is true that biological password can not be theft by personal negligence, but database attacked by professional hackers will lead to thousands of information leakage, which is the main source that cause social anxiety and concerns. Furthermore, people tend to hold distrust attitudes towards technology company, some even believe they would sell their personal information for profits. Hence, to tackle this social issues from corporation side, still needs every technical company’s effort to earn back trust of people and show them the benefits of this technique could bring.

Back to the video surveillance by government, citizens are quite sensitive for their private information captured by government especially after Snowden, but this is also a quite solid solution to prevent potential terrorism. In the famous 911 incident, the images of terrorists are recorded under the camera, they would be stopped if face recognition techniques were developed as present. This ambivalence between private information protection and social stability will continue under fierce debate, and finally need both sides to reach a consensus.

Additionally, each individual will share multiple identities in modern society (at work, in social medium, at home, etc.) instead of only one character. Therefore, to formulate a complex model which corporate all the human beings with their biometric information and manage them together is not feasible. Such system will cause a huge mount of money and man power from different organizations, while the system is also not able to guarantee high efficiency and accuracy.

To consider all these above, face recognition, or object detection still has social issues remaining for us to solve in the future.

### A.3.3 Ethical issues

### <Discussion of ethical issues>

Ethical issues are similar to legal issues and social issues in object detection field. Using camera equipped with face recognition techniques in public places to prevent terrorism also violate individual’s privacy. Using face features of potential terrorists stored in technology company’s databases to track their activities is equivalent to decrypt terrorists’ cellphone with phone company, it is right to do so, but may not be ethical. Ethical issues are also cannot be solved immediately, but requires long-term efforts along with society progress.

### A.3.4 Professional issues

<Discussion of professional Issues>

During the training phase, most of the training were carried on by 10 consecutive epochs (for example, 1-10 epochs, 11-20 epochs, etc.), few were adopted by 20 or 30 continuous epochs. The reason of not adopting training 100 epochs or until reaching the convergence is due to performance limitation of laboratory machines. Although there is not solid evidence shows huge difference between training 10 sets 10 epochs and 100 consecutive epochs, the results may be altered by this situation. It is difficult to solve this, since online laboratories are expensive and also suffered from network issues.

如果篇幅不够 用object detection challenge补足？

# Appendix B

# External Materials

<This appendix should provide a brief record of materials used in the solution that are not the student's own work. Such materials might be pieces of codes made available from a research group/company or from the internet, datasets prepared by external users or any preliminary materials/drafts/notes provided by a supervisor. It should be clear what was used as ready-made components and what was developed as part of the project. This appendix should be included even if no external materials were used, in which case a statement to that effect is all that is required.>

Code: Yolov5 developed as part of the project

Data set: VOC2007 ready-made components