School of Computing
FACULTY OF ENGINEERING

UNIVERSITY OF LEEDS

Final Report

Unsafe behaviors detection tool on construction sites

Xiao Fandi

Submitted in accordance with the requirements for the degree of BSc Computer Science

2019/2020

40 credits

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

Items	**Format**	**Recipient(s) and Date**
Deliverables 1, 2, 3	*Report*	*SSO (15/5/x20)*
Participant consent forms	*Signed forms in envelop*	*SSO (xx/xx/xx)*
Deliverable 4	*Software codes or URL*	*Supervisor, assessor (15/5/20)*
Deliverable 5	*User manuals*	*Client, supervisor (15/5/20)*

Type of Project: Exploratory Software

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) Xiao Fnadi

Summary

Accidents caused by unsafe behavior of workers often occur in the construction area. Once the accident happens, the impact on the normal production of construction workers and enterprise is catastrophic. Therefore, in order to prevent the occurrence of safety accidents caused by unsafe behavior of the workers, it is of great significance to study the real-time detection of these behaviors.

This project aims to apply machine learning method for object detection to train the model and build software system to make real-time detection on the image of surveillance camera, which make the unmanned monitoring possible and is potentially more accurate than manual monitoring.

Acknowledgements

I would like to thank my supervisor, Heng Liu, for the patiently guiding me to accomplish this challenging task and provide me with detailed advise.

I also want to thank Dr John Stell for the feedback he gave on my intermediate report.

Table of Contents (example of how to format)

Chapter 1
Introduction

1.1 Project Background

The safety of Construction Site has always been considered a serious issue to be improved. According to the survey, about 80% of the construction site accidents were human induced over the last 10 years, which rises the attention to prevent labours from getting hurt by miss-operation.

Among these accidents, 18% of them were caused by object strike, which was the second largest factor. And the most efficient way to reduce the risk of getting hurt from it is to make sure that the labors were wearing the safety helmet during their work.

1.2 Aim

The aim of this project is to create a surveillance video detection software system which can do real-time object detection on detecting workers unsafe behaviour. Once a dangerous behaviour is detected, a warning will be made.

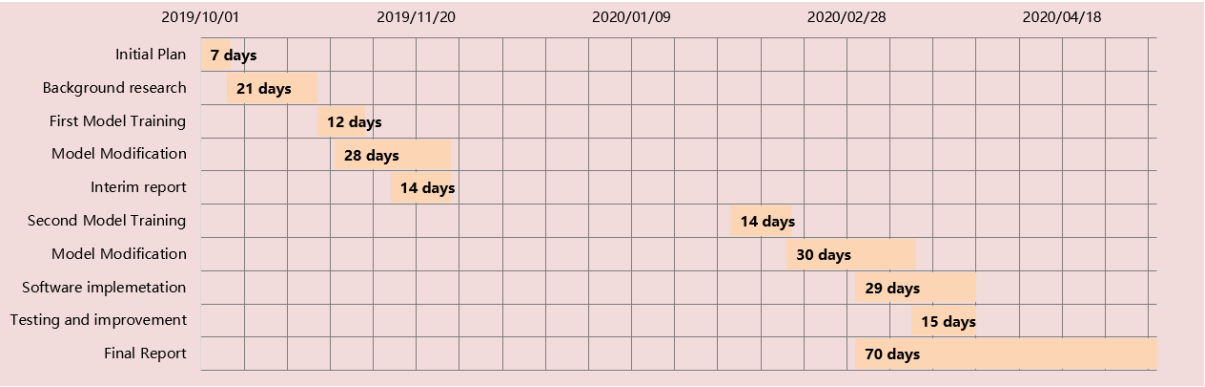
1.3 Objectives

The project will focus on detecting whether the workers wear a safe helmet or not, which can be divided into:

1. Collecting data and training the models based on chosen object detection method.
2. Evaluating the trained model and make modification to improve the performance.
3. Build software system to visualizing the detected result of the surveillance video and make alarms when people without safe helmet were detected.

1.4 Plan and Risk assessment

Figure 1.1 is a Gantt Chart which shows the overall plan of the project. The project is initially decided to explore at least two different machine learning methods, so the time management can be mainly divided into the first and second model training and modification followed by the implementation of software.



In order to maximize the precision and speed of detection, the time assigned to make modification of both first and second trained model may take longer than expected since data preparation and algorithms theoretical study can both be challenging, which may leave fewer days for the work behand.

1.5 Project Overview

The project is mainly divided into two parts:

1. First is the model training, evaluating and modification process which combines the first two objectives, but it involves the implementation of two kinds of object detection algorithms.
2. The second part is to build the software system which use the trained model for a surveillance camera detection and also make alarms while detecting, a further development of the project is also included in this part.

The report is structured in the order of the above steps with a background research in front and a conclusion behind. The evaluation is done at the end of each step.

Chapter 2
Background Research

2.1 Computer vision

Computer vision is the science that studies how to make the machine "see". Furthermore, it refers to the use of camera and computer instead of human eye to recognize, track and measure the target, and further do graphics processing, so that computer processing becomes more suitable for human eye observation or transmitted to the instrument for detection [1]. As a scientific discipline, computer vision studies related theories and technologies, trying to establish an artificial intelligence system that can obtain information from images or multidimensional data.

One of the major library which is used to handle computer vision is OpenCV. It is lightweight and efficient which is composed of a series of C functions and a small number of C + + classes. At the same time, it provides interfaces of python, ruby, MATLAB and other languages, and realizes many general algorithms in image processing and computer vision.

2.2 Object Detection

To analyse the information that can be understood by computer from the image is the central problem of machine vision. According to the needs of the task, there are three main levels of how the picture can be processed by computer as shown in Figure 2.1.

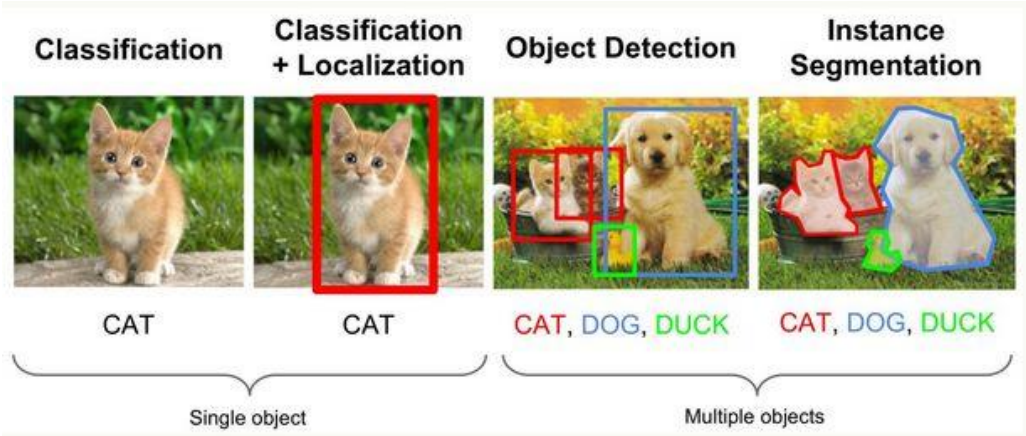


Figure 2.1 Three levels of how machine understand image

One is classification. The image is structured into a certain category of information, using a pre-determined category or instance ID to describe the image. This task is the simplest and most basic image understanding task.

The second is detection. The task of classification focuses on the whole image, which gives the content description of the whole picture, while the task of detection focuses on the specific object target in the image, which requires to obtain the category information and location information (classification and localization) of the target at the same time. Compared with classification, detection gives the understanding of pictures' foreground and background to separate the interested target from the background and determine the category and location of the target.

The third is segmentation. Segmentation includes semantic segmentation and instance segmentation. The former is the extension of background separation, which requires the separation of image parts under different semantics. The latter is the extension of detection task, which requires the description of the target outline, which is more refined than detection frame. Segmentation is a pixel level description of the image, which gives each pixel category and meaning, and is suitable for understanding the scenes which has high requirements [2].

2.3 Traditional object detection

Traditional object detection methods are divided into three parts which are region selection, feature extraction and classification.

Figure 2.2 Tradition object detection process

Region selection: this step is to locate the target location. Because the target may appear in any position of the image, the size, length and width ratio of the target are also uncertain, the process of sliding window is to set different scales, length and width ratios to traverse the whole image.

Feature extraction: The common methods of feature extraction in computer vision are usually divided into three categories.

1. Bottom features: colour, texture, which are the most basic features.

2. Middle level features: Based on the bottom features, machine learning is used to mine features after the learning process, including Haar-like features [3], HOG features [4], and other features based on optimization theory.

3. High level features: further mining and representation of low-level and middle level features, such as whether a person can wear a hat and glasses and other semantic features.

Classification: classify the target according to the features extracted in the second step. The main classifiers are SVM, AdaBoost, etc.

First, the feature is extracted from the candidate frame, which uses the method of sliding window to generate the candidate frame window, and then the local information in each window is extracted.

After that, the features extracted from candidate areas are classified and determined. This classifier needs to be learned and trained in advance. In this process, for single category target detection, we only need to distinguish whether the object contained in the current window is the background or the target. For the multi classification problem, we need to further distinguish the categories of objects in the current window. After the check box is determined, a series of candidate boxes that may be the detection target will be obtained. These candidate boxes may have some overlaps. In this case, an NMS is needed to merge the candidate boxes and finally get the target to be detected, that is, the final output of the algorithm.

2.4 Object detection based on deep learning

Object detection based on deep learning can be divided into two tasks: object classification and object location. The object classification task is responsible for judging whether there are objects of interest categories in the input image or the selected image areas, and outputting a series of labels with scores to indicate the possibility of objects' categories appearing in the input image or the selected image areas. The object location task is responsible for determining the position and range of the objects of interest in the input image or the selected image areas, and outputting the bounding box, or the centre of the object, or the closed boundary of the object.

2.4.1 Two stage object detection algorithm

For the two stage object detection network, it mainly uses a convolution neural network to complete the target detection process, which extracts the CNN features [5]. There are two steps in training the network, the first step is to train the Region Proposal Network, and the second step is to train the target area detection network.

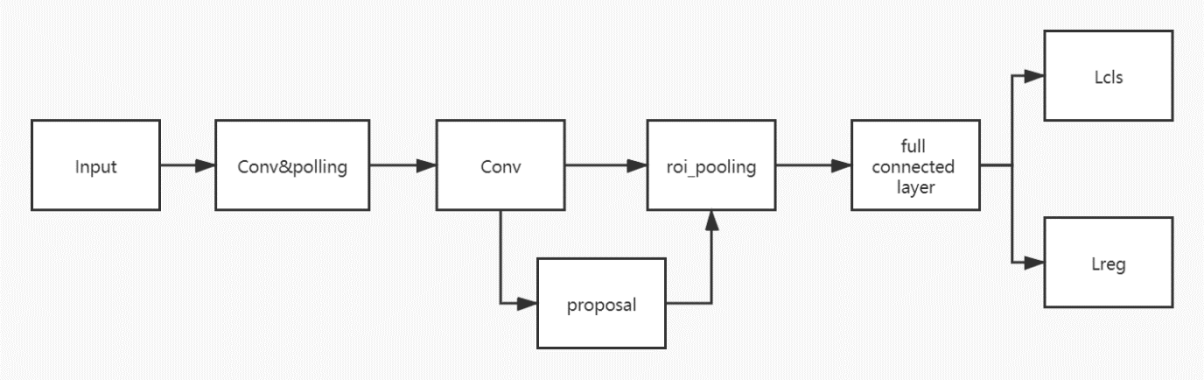


Figure 2.3 Two stage algorithm process [5]

First input a picture to the convolution neural network and extract high dimensional characteristics of the picture, this convolution neural network is called the backbone network, typical backbone network including VGGNet, ResNet , ZeNet and other classic convolutional neural network structures. Then the Region Proposal Network will generate candidate regions, meanwhile complete the region classification, which means the image can be divided into two different categories such as background and object, and make the initial prediction of the location of the object.

Next, the position in the candidate area is precisely located and corrected by the ROI pooling layer, then the candidate object obtained is match to the feature map, after that, a full connection layer will generate corresponding feature vector. Finally, through the process of classification and regression, the candidate target category and target location can be determined(the coordinates of the four points of the rectangle region, (x, y, w, h): (x, y) Is the coordinate of the top left corner vertex, w, h is the width and height of the rectangular box).

2.4.2 One stage object detection algorithm

One stage object detection algorithm does not need the stage of region proposal, which directly generates the category probability and position coordinate value of the object through a single stage.

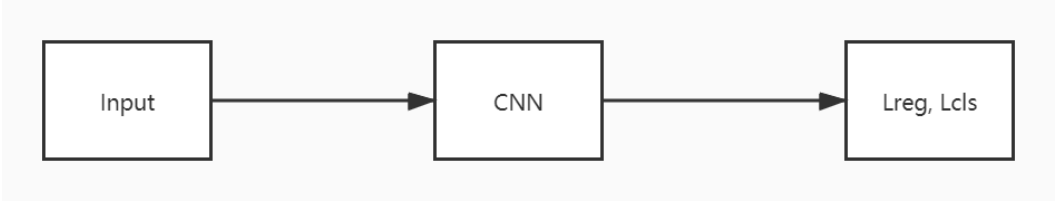


Figure 2.4 One stage algorithm process

After given the input image, one stage object detection algorithm will complete feature extraction through a backbone network, then it will directly carry out region regression and object classification [6]. Compared with two stage, the biggest difference between them is on

stage detection does not include the process of candidate region proposal which make it simpler and faster.

2.5 Evaluation methods

The evaluation of the trained model is mainly around two factor, which is the speed and accuracy.

2.5.1 Real-time detection

Generally, the frame rate of video will not be lower than 25 frames per second (FPS), which means, the maximum interval time of each frame is 40 ms, so in order to achieve real-time object detection, the detection speed is at least 40 ms per image.

2.5.2 Mean average precision

Mean average precision is used to evaluate the accuracy of object detectors such as Fast R-CNN and SSD.

In order to calculate precision and recall, like all machine learning problems, true positives, false positives, true negatives, and false negatives are needed. To get true and false positives, IoU value is required, the calculation of IoU is shown in Figure 2.5, which is the intersection ratio between bonding box and ground truth predicted by the model.

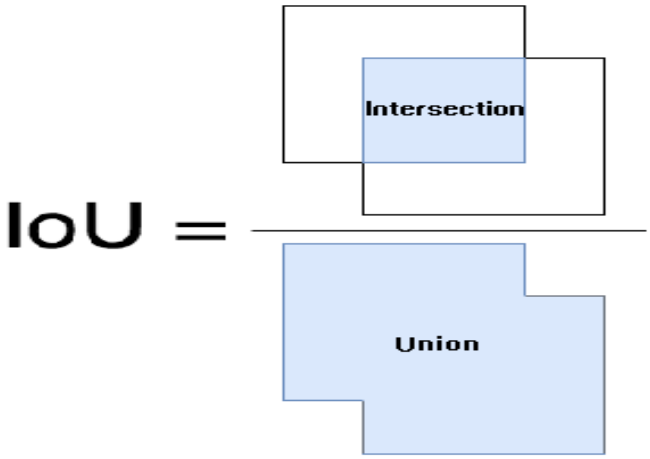


Figure 2.5 IoU definition

The calculation of IoU is used to determine whether a positive result is true or false[7]. The most commonly used threshold is 0.5, which means, if IOU > 0.5, the prediction is considered to be true positive, otherwise it is considered to be false positive.

And mean average precision refers to the average value of the maximum accuracy under different recall rates.

Chapter 3
Method selection and plan

3.1 Training methods compare and decision

To achieve the detection goal, this project will take the implementation of real-time detection as the first priority since the project is aimed to take detection on surveillance video, then try to increase the accuracy as much as possible.

Some of the tradition object detection method performed well on the early stage, like Haar-like feature with Adaboost algorithm for face detection, HOG feature with SVM for pedestrian detection. But they all have two main disadvantage [8]:

1. The features of manual design are not very robust to the change of context.
2. The region selection strategy based on sliding windows has no pertinence, which has high time complexity and window redundancy.

Compared with the methods based on CNN and deep learning which shows much higher accuracy and even faster speed nowadays, these traditional methods seems inefficient and Impractical for the purpose of this project.

As for the two type of deep learning methods, the process of one-stage network only needs to classify and regress the generated anchor frame which is just a logical structure or a data block, while the two-stage network is going to map the generated anchor frame to the feature Map area (except RCNN), and then reinput the area to the full connection layer for classification and regression, each anchor mapped area needs such classification and regression [9]. Therefore, compared with two-stage algorithm, the one-stage algorithms it is less time-consuming, but also less accurate. Since the detection speed is the first priority, one-stage algorithm should be more suitable for this project, and among those one-stage detection method, YOLO is the most outstanding one for its fast speed.

According to the evaluation of YOLOv3 tested shown in figure 3.1, It can process the pictures of coco test dev data set on Pascal Titan x-card, with a speed of 30 FPS and a mAP of 57.9% [10]. To be specific, the detection speed of YOLOv3 is very fast, about 1000 times faster than R-CNN and 100 times faster than the two-stage algorithm Fast R-CNN, which is the only algorithm that meets the requirement of less than 40 ms detection time under the same support of hardware. Besides that, YOLOv3 also behave well in mAP, which is almost as accurate as RetinaNet which is used to be the most accurate one-stage detection algorithm.

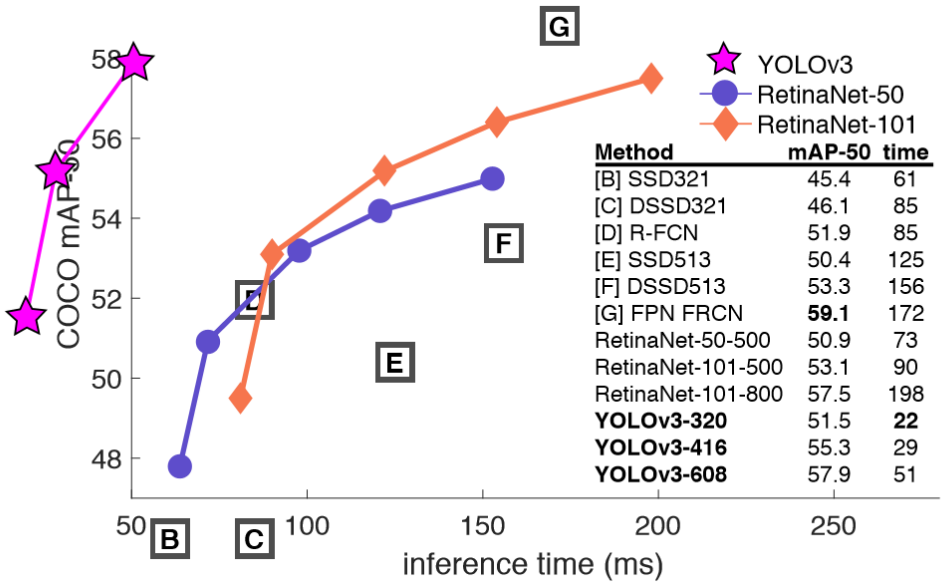


Figure 3.1 Evaluation picture by the author of YOLO algorithm [10]

As a result, YOLOv3 should be the best the best detection algorithm for this project. But there's a problem that can significantly affect the trained model, which is the quality of the data.

During the training, data with larger volume and more complexity can make the trained model more accurate in detection. Since there's no knowing prepared data for the detection target of this project, the data will be mainly get through a process of crawling pictures and labeling, which there may not have enough time to collect the data that is large and complex enough for a deep learning training. Therefore, a traditional method, Haar-like feature with Adaboost algorithm, will be implemented first during the early stage of data collection for:

1. As a traditional method, It requires less data volume than deep learning methods, and traditional method usually behave better than deep learning method when training on little data volume.
2. The algorithm is directly provided by OpenCV library, which can be easily used.
3. It is the first real-time detection operator which is famous of face detection, therefore, it is potentially capable of real-time detecting.

3.2 Overall implementation plan

The project aims to make detection on a surveillance camera video, to detect those workers who did not wearing a safe helmet and make alarms. A brief process of implementation consists of three parts: data collection and model training, model evaluation and modification, software system implementation.

As is descried in the above section of training method selection, the model training will be first implemented using Haar-like feature with Adaboost algorithm, and according to the evaluation of this model to decide whether to change to YOLOv3 method or not. There will be iterations of evaluation and modification, the final decision will be made when no modification can made or is necessary needed to make to the detection process.

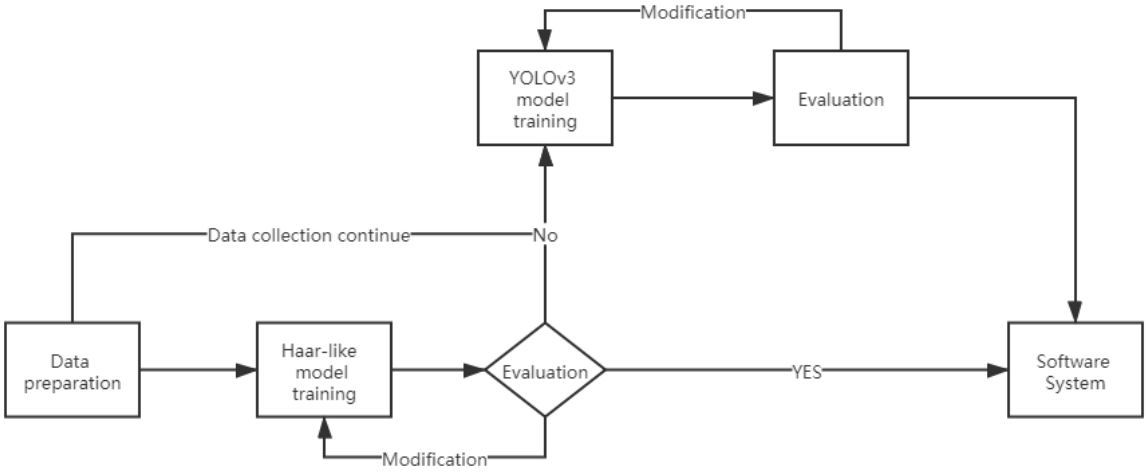


Figure 3.2 Flow chart of the implementation process

Chapter 4
Training process

4.1 Haar-like feature with AdaBoost algorithm

Haar Cascade training is a training method based on the cascade classifier of Haar-like feature. Which extract Haar-like features from the prepared positive and negative samples and produce several weak classifiers, these weak classifiers are later concatenated into a strong classifier which is the one used to detect the target.

4.1.1 Haar-like feature

Haar-like feature [11], named for it's like the haar wavelet, is a rectangle numeric future, which refers to the difference between the sum of all the pixel grey value of the two or more rectangles with the same shape and size.

Extended Haar-like feature consists of 3 types of features [12]: Edge features, Liner features and Centre-surround features (figure 4.1). Feature number is gained by the sum of pixel value in white area minus the sum of pixel value in black area. These types of feature value represent the feature of different direction of a target. In a sample picture, a large amount of haar features shall be produced according to the selected coordinate and the size of the rectangle, which causes a large massive of calculation, but with the implementation of Integral graph [13], the haar-like feature can be produced rapidly.

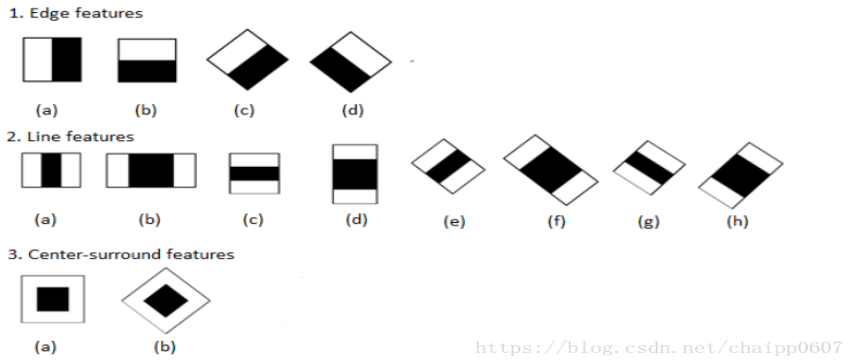


Figure 4.1 Haar like feature type

4.1.2 AdaBoost algorithm

Text AdaBoost is one of the best algorithms used to improve the accuracy of weak classifiers. For every Haar-like feature f, a trained weak classifier is defined by h(x, f, p, θ), x is the detection window, p is the direction of the inequality, θ is the threshold. To train a weak classifier h is to get the optimal threshold of feature f, which minimize the fault of a weak classifier [14-15].

$$h(x,f,p,\theta) = \begin{cases} 1 & pf(x) < p\theta \\ 0 & other \end{cases}$$

AdaBoost algorithm combines several weak algorithms to form a strong algorithm. Several strong algorithms are later forming a cascade classifier which further improves the accuracy. Cascade classifier discriminate target through binary decision tree, those which pass the discriminate can be recognized.

4.1.3 Implementation

The process of model training is done in windows environment, using C++ languages with Visual Studio IDE and OpenCV 4.1.1.

4.1.3.1 Detection plan

The method of Haar-like feature and Adaboost algorithm trains models on the picture data which consists of positive and negative data, where the positive data is the picture of the detection target, the negative data is the picture without the detection target. Therefore a classifier get after training can only detect one type of target, but the purpose of the project is to distinguish between who wears a safe helmet and who does not which in this case normally requires two classifiers. Since the project is mainly about to make alarm while detecting people who does not wear a safe helmet, the process can be simplified into just detecting who did not wearing a safe helmet, which in this case means only one classifier of detecting head (without safe helmet) is needed.

As a result, the initial plan is to train a classifier to detect head (without safe helmet).

4.1.3.2 Data preparation and model training

The dataset for head detection is a group of pictures download from the internet, the link for it is in appendix.

The final dataset used for training consists of 2200 samples of head without safe-helmet used as the positive data, 2200 background pictures of construction site and some other places used as negative data.

The positive data is a group of pictures which is the head part of human body and are resized into (20, 20), all positive and negative pictures are transformed into grey scale.

Figure 4.2 Positive data

First, using the program opencv_createsamples.exe provided by OpenCV to create samples which are the readable format of the training program.

Second, training the model by opencv_traincascade.exe, and set the parameters as follows:

-numStages 20 -minHitRate 0.999 -maxFalseAlarmRate 0.5 -mode ALL -featureType HAAR

After the training, a cascade.xml is formed, which is the classifier to be included in the programme.

4.1.4 Model evaluation and modification

The detection method in cooperate with Haar-like feature provided by OpenCV is detectMultiScale (image, objects, scaleFactor, minNeighbors, minSize, maxSize), where:

1. ‘’object’’ is a vector of rectangles where each rectangle contains the detected object.
2. ‘’scaleFactor’’ is the parameter specifying how mauch the image size is reduced at each scale.
3. ‘’minNeighbors’’ refers to how many times of detections to a specific target can be seen as a successful detection.
4. “minSize” and “maxSize” are the min and max size of the detection box.

Since the min and max size of the detection box can significantly affect the detection result for if the false positive detection increase with the span of detection box size. The evaluation is taking 100 pictures from surveillance camera video, which are all resized into (416, 416) to test the speed and accuracy of detection, accordingly the size of detection box is set to (15, 15) and (25, 25), which is the approximate size of the head in each picture. ScaleFactor and minNeighbors are set to default value of 1.1 and 3 respectively.

The IoU is set to standard 0.5 which if the IoU between the detection box and the true box is not less than 0.5, the detection is considered to be true positive.

The result is shown in table 3.1.

Table 4.1 Detection time per picture.

This detection is taken on CPU: Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz, the time may vary according to the parameter setting of the detection method.

Maximal time(s)	Minimal time(s)	Arithmetic average(s)
0.537	0. 418	0.449

Table 4.2 Confusion matrix table.

Real / Detected	**Head**	**Background and safe-helmet**
Head	144	632
Background and safe-helmet	40	___

For an object detection project, since number of Background is uncountable, only the $precision = \frac{TP}{TP+FP}$ and $recall = \frac{TP}{TP+FN}$ can be calculated, which in this case is:

1. Precision = 0.186
2. Recall = 0.783

As shown in the result Detection speed is fast enough for real-time detection of 40ms per picture, but it's mainly the problem of hardware of the testing computer and there's little we can do to improve it. Judging from the confusion matrix, there's too many false positive detection, which makes the precision value too low. According to the detected pictures, it can be seen that:

1. The classifier not able to distinguish between the those people who wear safe helmets and who does not.
2. Too many back ground false detection.

Figure 4.3 head detection result

In order to fix the first two problem, the training process is changed to train a classifier of detecting head with safe helmet, which takes 2200 samples of head with safe-helmet to be used as the positive data, 2200 background pictures of construction site and 2200 samples of head without safe-helmet or wearing other hat are used as negative data, which can help to reduce false positive detection in problem 1. Besides, a pedestrian classifier based on HOG feature, which is already provided by OpenCV library is used as an auxiliary to reduce the background false detection in problem 2. The process including:

1. Detect the human body, and take the detection box as the ROI (region of interest).
2. Detect the head with safe helmet based on the ROI generated by the first step.
3. Those ROI which failed to detect a head with safe helmet will be regarded as a person who did not wear a safe helmet and drawing in red, the other ROI will be drawing in green.

Figure 4.4 Positive data sample

Testing on the same 100 pictures, the detection time only slightly increased as shown below:

Table 4.3 Detection time per picture.

This detection is taken on CPU: Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz, the time may vary according to the parameter setting of the detection method.

	Maximal time(s)	**Minimal time(s)**	**Arithmetic average(s)**
Before	0.537	0. 408	0.429
After	0.498	0.423	0.447

Table 4.4 Precision and recall changing table.

	Precision	**Recall**
Before	0.186	0.783
After	0.834	0.693

From the change of precision, it can be seen that the false positive detection is significantly reduced, but instead the value of recall went down. According to the detected pictures (as shown in below), with modification to the training, the false positive detection both in problem 1 and problem 2 does reduced, but the combination of the two classifier shows that the detection method cannot detect all kinds of posture like squat and bend.

Figure 4.5 detection result

There's indeed a solution to fix the new problem by training a human body classifier, and continue to increase the data volume for training to make the classifier more accurate, but it is time consuming, and judging from the behavior of models trained so far, it is unlikely the classifier can be trained for a practical use, therefore the model training by Haar-like feature with Adaboost algorithm is abandoned.

Still the evaluation result rises some potential demands for the following work.

1. The support of GPU acceleration to increase the detection speed.
2. Expend the data volume and complexity to improve the accuracy.

4.2 Yolov3

Yolo algorithm is a one stage object detection algorithm, which is famous for its detection speed [16]. After given the input image, the yolo network will complete feature extraction through a backbone network, then it will directly carry out region regression and object classification.

This section will be divided into 5 part, which is the design concept of the algorithm, the structure of the network, the output format of the network, model training process and evaluation.

4.2.1 Design concept

Yolo algorithm uses a separate CNN model to achieve end-to-end target detection [17]. The basic process is shown in Figure 4.6: first, resize the input image to 416x416, then send it to CNN network, and finally process the network prediction results to get the detected object.

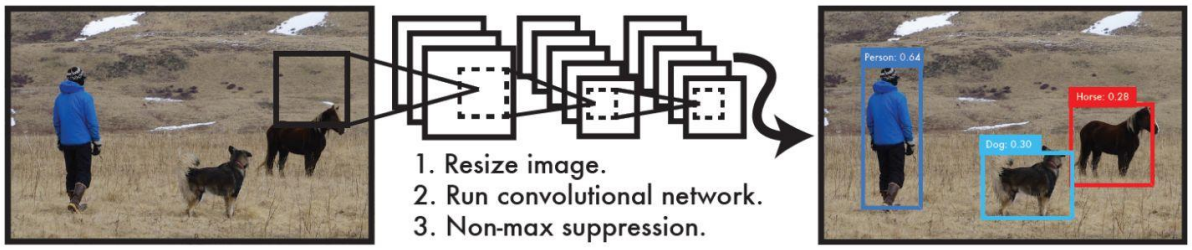


Figure 4.6 brief process of YOLO

Specifically, Yolo's CNN network divides the input image into a $S \times S$ grid, and then each cell is responsible for detecting the targets whose center points fall in the grid. As shown in Figure 4.7, it can be seen that the target center of the dog falls in a cell in the lower left corner, then the cell is responsible for predicting the dog. Each cell predicts B bounding boxes and the confidence score of the bounding boxes. The so-called confidence degree actually includes two aspects, one is the possibility of the target contained in the bounding box, and the other is the accuracy of the bounding box. The former is recorded as $\Pr(object)$, when the bounding box is the background, then $pr(object) = 0$. When the bounding box contains a target, $pr(object) = 1$. The accuracy of the bounding box can be characterized by the IOU (intersection over union) of the prediction box and the actual box (ground truth), which is recorded as $IOU\ {}^{truth}_{pred}$. So confidence can be defined as $pr(object) * IOU\ {}^{truth}_{pred}$. Yolo's confidence is the product of two factors, and the accuracy of the prediction box is also reflected in it. The size and position of the boundary box can be represented by 4 values: (x, y, w, h), where (x, y) is the center coordinate of the boundary box, and w and h are the width and height of the boundary box. It should also be noted that the predicted value (x, y) of the center coordinate is the offset value relative to the coordinate point in the upper left corner of each cell, and the unit is relative to the cell size. The definition of the cell coordinate is shown in Figure 4.7. The predicted values of the formula and formula of the bounding box are the ratio of the width to the height of the whole picture, so theoretically the size of the four elements should be in the range of the formula. In this way, the predicted value of each bounding box actually contains five elements: (x, y, w, h, c), among which the first four represent the size and position of the bounding box, and the last value is the confidence degree [18].

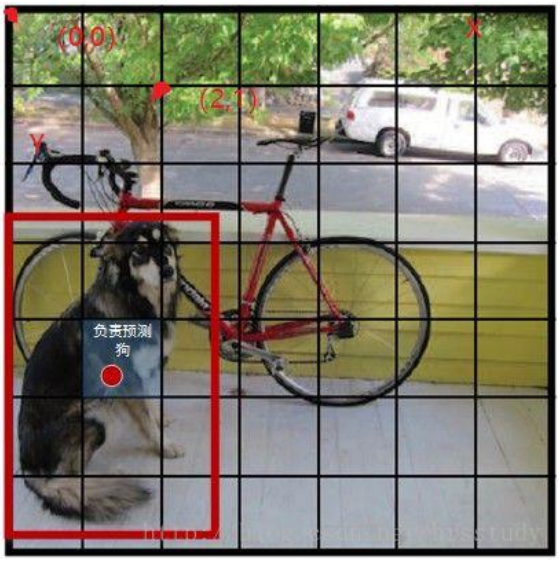


Figure 4.7 picture divided in to grid cells

As for the classification problem. For each cell, it is necessary to give the probability value of C categories, which represents the probability that the goal of the bounding box predicted by the cell belongs to each category. These probability values are actually conditional probabilities under the confidence of each bounding box, which is $pr(class|object)$, the confidence of bounding box category represents the possibility that the target in the bounding box belongs to each category and the matching target of the bounding box. Yolov3 binds the category probability prediction values with the bounding box.

4.2.2 Network structure (backbone)

Yolov3 uses a 53 layer convolution network, which is composed of residual elements. The experiment of Joseph Redmon shows that the model of darknet-53 performs better than resnet-101, resnet-152 and darknet-19 in the balance of classification accuracy and efficiency [17]. Yolov3 does not pursue speed so much, but pursues performance on the basis of ensuring real-time.

On the one hand, the darknet-53 network adopts the full convolution structure. In the forward propagation of Yolov3, the size transformation of tensor is realized by changing the step size of convolution kernel. The convolution step size is 2. After each convolution, the image edge length is reduced by half. As shown in Figure 4.8, there are 5 convolutions in darknet-53 in steps of 2. After 5 times reduction, the feature map is reduced to 1 / 32 of the original input size. So the size of the network input picture is a multiple of 32, which is 416 × 416.

On the other hand, the network of darknet-53 introduces the residual structure. Compared to Yolov2, the difficulty of training deep network is greatly reduced. Therefore, the accuracy of darknet-53 network is improved obviously.

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 4.8 YOLOv3 darknet 53 backbone[10]

4.2.3 Network output

Yolov3 uses multiple scale fusion to make prediction. The finer the grid cell is, the finer the object can be detected. Which improved the detection effect of small object.

Yolov3 sets three boxes for each grid cell, so each box needs to have five basic parameters (x, y, w, h, confidence) as described in Design concept. Yolov3 outputs three feature maps of different scales, as shown in Figure 4.9. The depth of Y1, Y2 and Y3 is 255, and the rule of side length is 13:26:52.

The feature size obtained by each prediction task is n × n × [3 * (4 + 1 + 80)], where n is the lattice size, 3 is the number of bounding boxes obtained by each lattice, 4 is the number of bounding box coordinates(x, y, h, w), 1 is the confidence score, and 80 is the number of categories. For coco categories, there are 80 categories of probability, each box should output a probability for each category, therefore the output tensor is n × n × 255 [16].

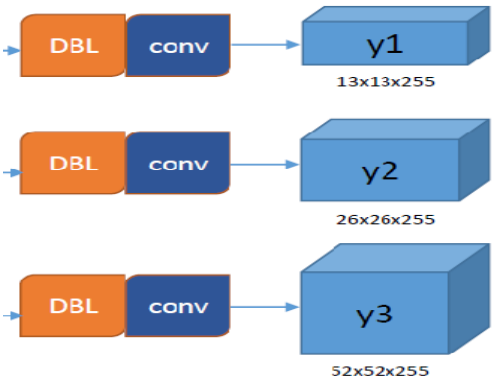


Figure 4.9 network output data

4.2.4 Implementation

The second model training uses python language with Pycharm IDE. The used YOLOv3 is developed in keras framework, which support the GPU acceleration. The hardware and programming environmental set up is listed in the table below.

Table 4.5 Environmental set up.

Graphic card	RTX600 (24GB)
CUDA	10.0
cuDNN	7.6.5
Python	3.7
Tensorflow-gpu	1.14.0
OpenCV (python)	4.2.0
Keras	2.3.1

4.2.4.1 Detection plan

Since Yolov3 is capable of doing multiclassification, the best way to achieve the detection purpose of the project is to train the model to detect both head with and without safety helmet.

4.2.4.2 Data preparation and model training

This time, the data prepared for yolov3 network training increased to 19351 pictures, which is a combination of crawled picture and a group of pre-labeled pictures of human head, the link is in Appendix. All the pictures are preprocess using tool – Labeling, and the label information is stored in XML file.

The data samples consists of approximately equivalent number of head with safe helmet and without safe helmet to sure the data balance, since the model trained with unbalanced data will inevitably lead to poor or even unpredictable detection performance on kind with few samples. The whole data is divided into 6 : 2 : 2 parts, where 6/10 is the training data, 2/10 is the testing data, the remaining 2/10 is the validation data.

The labeling information stored in XML file is processed to extract the <name>,<width>,<height>, <xmin>, <ymin>, <xmax> and <ymax>. The first two element

<width> and <height> is the size of the image and the last four element represents the coordinate of upper left corner and lower right corner.

The data format which is required for training is <object-class> <x_center> <y_center> <width> <height>, which is the similar representation of the bounding box coordinate (x, y, w, h) described in Design concept of Yolo (section 3.3.1).

<Object-class> in this project is classes Head and Safety Helmet which will be marked as number 0 and 1. The conversion formula for the last four data is

1. <x_center> = <absolute_x> / <image_width>
2. <y_center> = <absolute_y> / <image_height>
3. <width> = <absolute_width> / <image_width>
4. <height> = <absolute_height> / <image_height>

Where <absolute_height> is the abscissa of the index frame center which can be calculated by (<x_min> + <box_x_max>)/2.

Then some important hyper parameters is needed to be set, which are batch size and subdivision. Batch size in yolo network means how many samples the network accumulates to carry out a back propagation which is recommended to set to 32 or 64 since a reasonable larger batch size helps to reduce the violent change in updating weights, therefore reduce the occurrence the nun value during training and improve the behavior of the final model. Subdivision means to divided the batch of samples for a iteration of training into several small batches, for example: if the batch size is set to 32 and subdivision is set to 16, which means the batch is divided into 16 subdivision of size 32/16=2, after 16 times of forward propagation of sample size 2, a back propagation can be implemented. This setting of subdivision is used to reduce the GPU load, if the error: out of memory happened, increasing the size of subdivision will help. For this project, the batch size is set to 64 while the subdivision is set to 32

Finally, the training can be started. After 3 days of more than 30000 iterations, the average loss rate is stabilized at about 0.7 %, which is an ideal statues to stop the training.

4.2.5 Evaluation and modification

There are various ways to improve behavior of the trained model, which can be divided into modifications made before training and after training. Since the analysis of modifications made before training is very time consuming, besides most of the factors which may affect the behavior of the model before training are handled according to experience of other peoples' work like these talked in section 4.2.4.2, this section will mainly testing on the factors that have effect on the model's behavior after the training.

The first factor is the image size. The feature extraction network of yolov3 generate 3 deferent feature maps after processing on each pixel of the input picture, each cell on the feature map reflects to the information of an specific area in the input picture, therefore, increasing the size of a picture means more pixel will be will be process to generate a cell in the feature map, which gives more clear information of that area of the input picture, as a result the predictions made based on the feature map will be more accurate, but instead, it will take more time to processing a picture.

To get the most suitable size of input picture, the evaluation takes 3097 pictures from the dataset to calculate the mAP and average detection speed under each size of input picture.

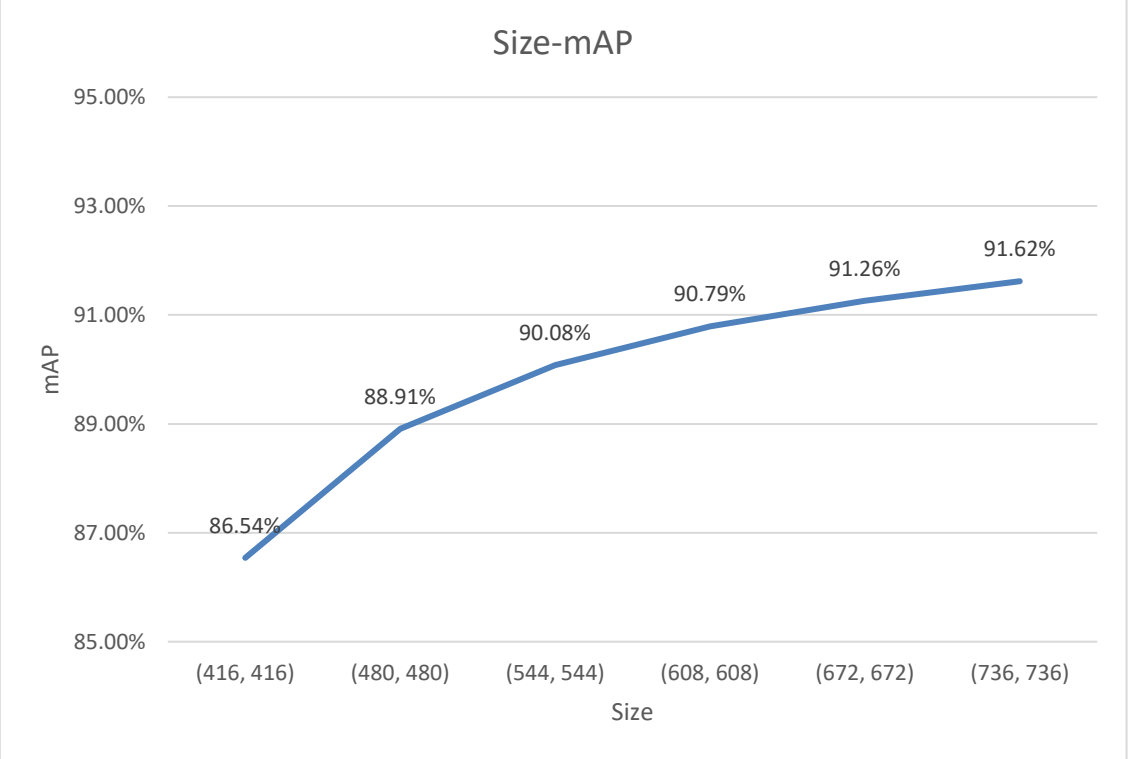


Figure 4.10 Tested under default detection threshold value of IoU = 0.45, score = 0.3, The mAP calculation using the default value of IoU = 0.5

Figure 4.11 Tested under default detection threshold value of IoU = 0.45, score = 0.3

It can be seen clearly from the picture, the trained model obtained a high mAP even when the input image is of the smallest size (416, 416), which make it unnecessary to make any change to the training process. Besides, the relationship between Size and speeds or mAP are generally nonlinear, the mAP grows slower, while the detection time grows faster with the increase of input image size, therefore there will be not cost-effective to increasing the size of the image for a higher mAP. For the purpose of this project, the size of (544, 544) is the most suitable one for it has the highest mAP under the threshold of 40 ms per picture speed for real-time detection.

Another factor which have impact on the mAP of the model is the process of filtering the bounding boxes from the out put of the Yolov3 network. The parameters used to control the process are an IoU value and a class score value as in the description of the above chart.

As is described in section 4.1.3, after an image is put into a trained network, 3 different scales of feature map is generated, which are of size (13, 13), (26, 26), (52, 52). Each size of feature map is assigned with 3 anchor boxes, which is generated from the training data using k-means, to make prediction on each grid cell. For example, a (13, 13) feature map is going to have 3 anchor boxes applied to each grid cell, which also means three predicted bounding boxes for each cell, so there will be $13 \times 13 \times 3$ boxes for the feature map (13, 13). Totally, there are $(13 \times 13 \times 3) + (26 \times 26 \times 3) + (52 \times 52 \times 3) = 10647$ bounding boxes generated.

But most of the bounding boxes stored in the output tensor does not contain detection target, therefore the class score is introduced to describe to what extend the bounding box may cover a category of the detection target. For each bounding box, a coordinate and

confidence score (x, y, w, h, c) is assigned and is bounded with the possibilities $pr(class|object)$ for the two classes which are Safe helmet and Head in this project. The class score of each bounding box can be calculated as:

$$pr(class|object) * pr(object) * IOU \begin{matrix} truth \\ pred \end{matrix} = pr(class) * IOU \begin{matrix} truth \\ pred \end{matrix}$$

which is the multiply of confidence score $pr(object) * IOU \begin{matrix} truth \\ pred \end{matrix}$ and probability of each class. The setting of the class confidence score threshold is then used for the filtering process to get rid of the bounding boxes with a class score under the threshold.

After that, only a few bounding boxes are left, but most of them were crowded together as shown in left picture, which only one bounding box is needed, to remove the redundant bounding boxes, the method called NMS (Non-maximum suppression) is applied.

The process of NMS in Yolo can be described as:

$$S_i = \begin{cases} S_i & IoU(M, b_i) < N_t \\ 0 & IoU(M, b_i) \geq N_t \end{cases}$$

Where M is the bounding box with the largest class confidence, Compute the IoU between M and the other bounding boxes b_i, if the IoU is over the threshold N_t, the confidence score of the bounding box S_i is set to 0 and then removed, then the rest bounding boxes are now used as the final detection result as shown in left picture.

Figure 4.12

But there's a problem which can be seen from the picture. Originally, the mothed of NMS is applied to process the bounding boxes of each category separately. The two type of detection targets in this project are highly resemble and are all detected at the same place of human body – head, as a result, for some of the pictures, two bonding boxes of type 'Safe helmet' and 'Head' drawing on single person's head may happen as shown in the left picture.

In order to prevent the false positive detections like that, the process is changed to apply NMS to all bounding boxes regardless of the category. The result is shown in the right picture below.

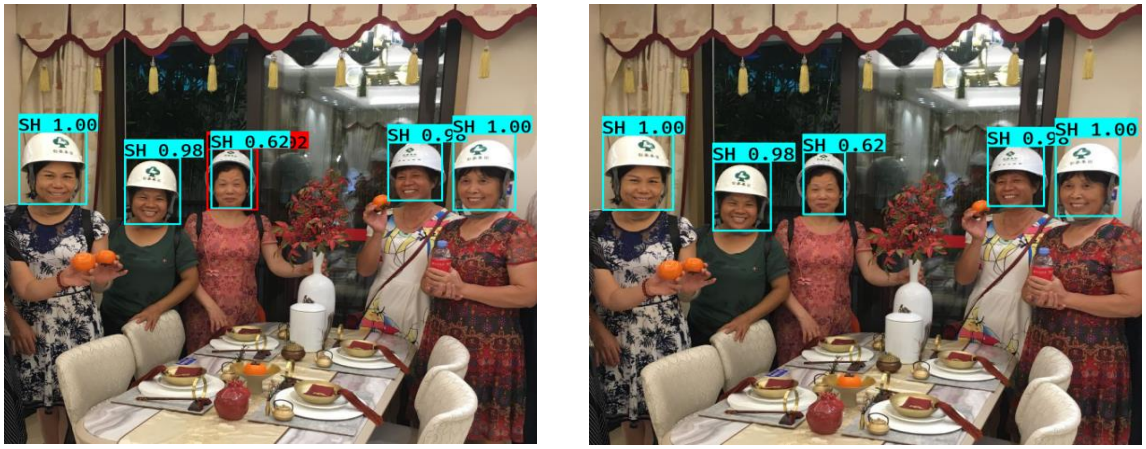


Figure 4.13

Table 3.1 Evaluation before and after the modification.

The testing is taken on input image size of (544, 544), which is the best input size as described before. detection threshold value IoU = 0.45, class confidence = 0.3, mAP calculation threshold IoU = 0.5

	mAP	**Recall (SH / Head)**	**Precision (SH / Head)**
Before	90.08%	95.35% / 87.15%	92.23% / 85.29%
After	89.96%	95.25% / 87.02%	92.39% / 85.51%

After the modification, recall along with mAP are all decreased, while precision rises, which indicate that the number of false positive detection like the one described above does reduced, but instead, the strategy of applying NMS to all categories of bounding boxes equally also affect the situation in which two types of bounding boxes are settled in the right place but IoU between the two boxes are above the thresh hold.

In order to reduced the impact on the above situation, this time the process is changed to apply another NMS after the original filtering process, which is after the first NMS processing of different categories separately, the remaining bounding boxes will go through another NMS which this time the threshold of IoU is set to a high level of 0.85 to avoid miss removement.

Table 3.1 Evaluation before and after the second modification.

	mAP	Recall (SH / Head)	Precision (SH / Head)
Before	90.08%	95.35% / 87.15%	92.23% / 85.29%
After	90.07%	95.29% / 87.12%	92.65% / 86.26%

It can be seen from the above result, the precision of the model is now considerably increased, though according the slight reduce in recall, there's still some miss removement to the bounding boxes, but it's within a negligible number.

The modification does not help to improve the mAP, which may because the model has already had a high mAP in the test set. And the Recall will not get higher than the original one because the modified process only reduce the number of boxes get. But the improvement in precision is useful for a practical usage to reduce false alarm.

If the model continues to be improved in the feature, the mAP will go higher and the benefit get from this method will get smaller or even turned to reduce the mAP. Besides, the modified process helps to improve the model only when the detection targets are highly resemble, so the method will be just provided as a selection in the late software development in case that multi bounding boxes of different category are drawing on the same person during the video detection.

Chapter 5
Implementation of the detecting system

5.1 Initial system design

The detection system mainly consists of three part, which are surveillance camera, detection unit and alarm unit.

The detection unit consists of the trained models and the control center. The control center load the model, receive the video stream data and use the build in detection network to make predictions.

The alarm unit receive the detection result send from the detection process unit and make alarm according to different detection result.

This project will just implement a simplified version of the system to imitate the process, in the real world, each part of the above system can be much more complex.

5.2 Surveillance camera

For the convenience of connecting to the camera, the programme will be implemented to support only webcam which is the most widely used type of camera that send video data via internet, so the user is free to change into their own camera to make connection. An example of the webcam is shown in the table below, which was used for testing this project.

Table 5.1 Important parameters of the surveillance camera.
HIKVISION 2CE16D1T-IT3F

Maximal image size	Maximal frame rate	Supporting protocol
(1920, 1080)	25 fps	TCP/IP, HTTP, HTTPS, RTSP

5.2.1 Video data transportation

For the above supporting protocols, this project choose to use RTSP (Real Time Streaming Protocol) to connect the webcam [19]. The RSTP protocol is an application layer protocol in TCP / IP protocol system, which is a multimedia streaming protocol used to control voice or video, and allows multiple streaming demand control at the same time.

To connect the webcam, the URL of the camera will be needed to receive the RSTP video stream. The URL is unique for each webcam, which should be offered by manufacturer.

The method VideoCapture() provided by OpenCV-python library is used to get access to the URL and receive the video stream in the software development, then each frame of the video is send to the yolo network for detection.

5.3 Control centre

Four operation choices are implemented for the control center of this project, which are parameter setting, image detection, video detection and camera detection.

Though there are default parameters set in the control center for the detection, it is recommended to set some important value like the image size for it may affect the speed and accuracy of the detection, users have to choose the proper size of input image according to the computer hardware.

The control center will automatically load the model trained in this project for the detection, but it also offers an option for the user to use their own trained model for the detection by input the path of model in the settings, accordingly there will be a choice to choose the detection target which the user wish to make alarm while detecting.

5.3.1 UI design

Since the system software will be mainly used by people without academic knowledge of the detection steps, a user Interface is necessarily needed to simplify the process of using control center.

The user interface use pyQT5 for implementation. The figure below shows the basic interface of the software, the function buttons are listed at left which leave the central part to show the predicted image or video. The alarm information will be listed on the left side of the image window.

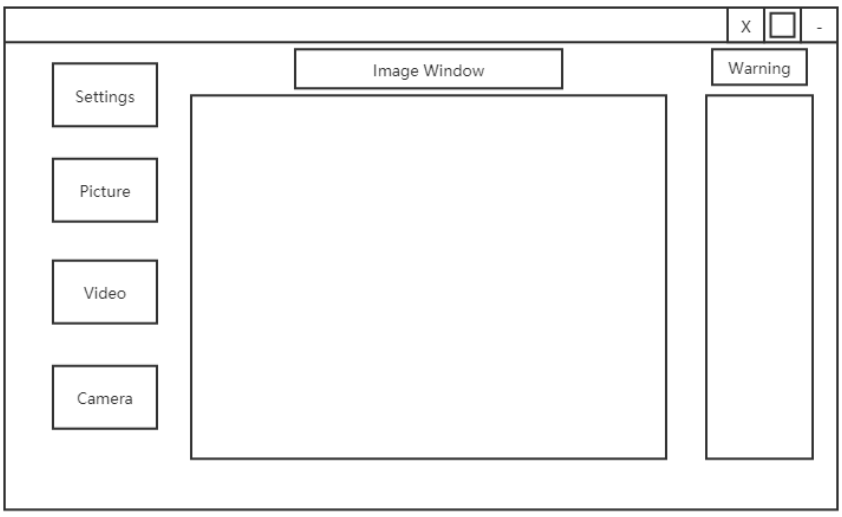


Figure 5.1

5.4 Detection results and alarm

For a practical usage, the alarm unit should contain various type of operations according to the detection like triggering the fire extinguishing system when fire or smoke is detected. Due to the limited resources, the alarm unit implemented in this project will only make alert and present the warning information on the user interface.

The alarm is bounded with the video detection process in this project. Since the precision of the model may not be high enough, after setting the detection target, the process won't trigger the alarm until the target is consecutively detected for over 3 frames of the video.

5.5 Further development

Considering a feature work of the large topic "unsafe behavior detection on the construction site'', there will be more detection target added into the system, when multi different alarms made at the same time, the user need to know more than just what they did, but also which one did it.

5.5.1 Object tracking

In order to that, another subarea of the computer vision is introduced, which is object tracking. The difference between object detection and object tracking is that, object detection tell you what the object is and where it is, while the object tracking does not tell you what it is but where it is and where it is going. After the target is identified first, the tracking algorithm or system needs to relocate the given target quickly and efficiently in the next sequential data.

Usually the object tracking includes the process of object detection especially multi-object tracking, which the method is defined as "tracking by detection" [20]. The basic process including:

1. Use the object detection algorithm to detect the interested targets in each frame, and get the corresponding (location coordinates, classification, confidence score), assuming that the number of detected targets is m.
2. Correlate the detection results in step 1 with the detection targets in the previous frame one by one (assuming that the number of detection targets in the previous frame is n). In another word, it is to find the most similar pair among m * n pairs.

The advantage of this method is that it can track new targets in the video at any time, and accordingly, this method requires a good "object detection" algorithm.

5.5.2 Deep sort

Deep sort is an detection based algorithm which is widely used in multi-object tracking, it is a modification of the sort algorithm (Simple Online And Realtime Tracking) [21].

The sort algorithm mainly uses two core algorithms to tracking a target: Kalman filter and Hungarian algorithm.

1. Kalman filter can find the "best" estimate value by combining the predicted value of mathematical model and the measured value to make data fusion and make predictions based on it. Specifically, it is a kind of denoising algorithm, which can get more accurate bounding box on the basis of object detection.
2. Hungarian algorithm is a data association algorithm, which is used to match the bounding boxes of two sequential frames.

Generally, Sort algorithm use Kalman filter to predict the movement of the targets to generate bounding boxes, Hungarian algorithm associate the predicted bounding boxes with the currently detected bounding boxes to find out if they are the same target.

Based on the process of Sort algorithm, the Deep sort algorithm invites appearance model (ReID), which is generated by putting training data into a CNN, and motion model (Mahalanobis distance) to calculate similarity along with Hungarian algorithm. Using the features extracted by CNN for association greatly reduces the ID switches in Sort algorithm, which is proved by the author's experiments to be reduced by about 45%.

5.5.3 Implementation

Process will implement Deep sort algorithm to track the workers on the construction site and give them a number for identification.

As is described above, the Deep sort is a detection based algorithm, which in this project, Yolov3 is just the appropriate detector to be used. Accordingly, the dataset used in Yolov3 model training will also be used for the training on the designed CNN to get the Deep Appearance Descriptor for data association.

For each frame of the video the tracking process is:

1. Input the image frame into the YOLOv3 model to make detections and get the bonding boxes of each target.
2. Put the bounding boxes into the Deep sort algorithms, the algorithms will match feature of these bounding boxes with the previous predicted bounding boxes to see if the target appears in previous frame or it’s a new one, then generate a group of predicted bounding boxes which indicate where the target will go in the next frame along with the matched ID number as the identification for each target.

3. The last is an modification made to the tracking process, which is matching the ID with the target class. The Deep sort algorithm only associate ID with each bounding boxes, but does know what kind of bounding boxes it has matched with, so an IoU computation is used to solve the problem.

   The process is to compute the IoU between the tracking box and the detecting box, if the IoU value is over 0.8 (the number is tested to be an appropriate threshold for matching boxes), then the ID will be bounded with the class. For simplicity, this matching process will only be done for the class which the alarm is for.

An example tracking result is shown in the picture below.

Figure 5.2

Since the track of each worker is not necessary needed yet, for this project only implement the detection of one type of unsafe behavior, it is provided as a selection which the user can change from detection into tracking in the settings of user interface.

5.6 Software testing

The software is tested in the order of the main function of the control center with the alarm function tested in the end.

Function	Expected result	Actual result
Settings	After making changes to the value, a printed result of the currently set parameter will be showing on the console. The result matches with the changes.	As expected
Picture detection	Bounding boxes are drawing on the picture, and the image is showing on the image frame of the UI	As expected
Video detection	Video showing on the image window with a FPS showing on the top left of it, and all bounding boxes were drawing properly.	As expected
Camera detection	Can show the image after the URL for rstp transportation is input.	As expected
Default	Pressing the button Default, a printed result will shown the currently set parameters, which are the default value in the code.	As expected
Alarm	While “Head” is detected, the alarm can make voice, and the information is showing on the left warning list.	Warnings are shown one the list well, but Software blocked while making alarm, but the software can still working after the alarm stopped.

A further testing of the core functions on videos is also conducted:

1. Detection generally performs well on both picture and video detection, except that the detection of “Head” is intermittent if the target is small in video. Increasing the input size of the image helps to improve it, but increase the detection time.
2. Tracking function does not perform well for the ID switch is frequent, for now there’s nothing that can be done to it. And it is found that using the tracking algorithm increased the processing time of each image by approximate 10 ms per image during the testing, which highly affect the requirement of real time detection if the memory space of the GPU is small.

The framework which the YOLOv3 detection network is based on is also affecting the performance of the software, to be specifically, compared with other deep learning frame work like Pytorch, this kears framework make the training and detecting process slower, which make the software harder to do real-time detection.

Chapter 6
Evaluation and Conclusion

5.1 Project evaluation

The aim of the project is to detect unsafe behavior of the workers on the construction site, which is heavily counting on the computer vision technology and image data. It's actually a tricky thing to balance between the quality of detection result and number of unsafe behavior achieved to detect. The former aim requires an systematic understanding of the overall detection method and the academic knowledge of few chosen detection algorithm, the latter has a large demand of data resources, both of them is time consuming. After found it difficult to get enough corresponding image data, I decide to focusing on exploring different methods of detecting a specific unsafe behavior. So the objective of the project is to collect data, training models based on different detection algorithm, modify the detection process to improve the result and build software system to use the model for detection and make alarm.

For the first two objectives of the project to collect data, training, evaluating and modifying the model, I did found a suitable algorithm to achieve the detection which is yolov3, an object detection algorithm both performed well on speed and mAP, especially the detection speed, which makes it quiet appropriate for this project to use surveillance camera for detection. But since the data collection process went quite slow, I decide to start the project by exploring a traditional method for they requires less data volume. Though after so many adjustment, the result of implementing Haar-like feature is still not a satisfactory one, but it's not surprising that the result will went like this. It may be much suitable to change the selection of the first method into other deep learning algorithms like SSD and Fast R-CNN which are comparative to YOLOv3, but as a beginner of this area, the process is meaningful for it helped a lot in understanding the process of object detection which make it easier to do the following implementation, evaluation and modification of Yolov3.

In order to improve the performance of YOLO3 model, I start by taking a detailed study of the prediction process of YOLOv3, so later, three factors which may affect the detection result were discovered, tested and changed properly, the highlight part of the modification should the applying of double NMS to furtherly get rid of the the false bounding boxes, though it's just helped to increase the precision, but considering a real world implementation, it is helpful to reduce the alarm of false detection.

For the third part of the objective, though the software developed is just a simple imitation of a detecting system, but its fully functioned and is developed to allow changes to detection settings and reload another model for the detection, which is a good frame work for a feature work. Furtherly, a tracking function is implemented for the purpose of identifying which one did the unsafe behavior, but it is found that the algorithm did not process data association

properly to make exact identification to each person, which makes the tracking function impractical for a usage for now, retraining the appearance feature with more data can help to improve the performance.

5.2 Future work

The future work is mainly around the changes to training data and possibly to exploring more object detection training network:

1. Add more categories to the database like reflective vest, working gloves, especially human body for more detection tacks of unsafe behavior.
2. Implement training and evaluating functions to the control center for a more simple usage of further study and work.
3. Change the process of tracking each person by head into human body, since head tracking is unstable when the tracked person wearing or removing the safe helmet form his or her head. Besides, human body provide much more appearance feature than head, which help to reduce the ID switch.
4. By the end of the project, YOLOv4 is already released with higher mAP and faster speed tested on COCO dataset. It is better to change the training algorithm in to YOLOv4, by the way, using another framework for implementation like Pytorch, which makes network faster both in training and detecting.

5.3 Self Assessment

Detecting unsafe behaviors of the workers on the construction site is quite a big topic. So before the project start, I have made an fully investigation of all kind of unsafe behavior which may affect the safety of the workers or make damages to the construction site, but it is found that most of the dangerous behaviors are quiet complex to detect, like standing on an inadequate scaffolding or using heavy equipment unproperly, which may need the involvement of hardware sensor to be placed on the area nearby. Simpler works like checking the wearing of safe helmet, reflective vest, working glove and so on, these tacks are capable to implement through surveillance camera video with computer vision technology, which I have later learned to be an object detection task.

The further study lay a solid foundation for my future work around the area of computer vison, where I found various ways to achieve the task, these methods can be mainly divided into the detection with or without the involvement of machine learning algorithm. For the method without machine learning algorithm, several image processing technologies which are capable of extracting color and contour profile of the image can potentially solve part of

detection task, like detecting safe helmet by its color, but since the image features generated by training on machine learning algorithm or CNN extraction can be much more representative and are now widely used for object detection project, I decided to start the implementation of the project to finding the appropriate machine learning methods.

After a thorough background research of all kinds of object detection algorithm, I'm actually confused about how the features of manmade and the features generated by CNN worked. It seems to me that the manmade features are more reasonable to successfully detecting a an object since it is simpler to understand compared with the process of CNN, yet the truth is the features generated by CNN is much more robust for the detection than manmade ones. But I still choose to implement the detection by training the traditional manmade features, not just because of the reasons I talked about in the project selection and plane, but also for it can provide me with a detailed understanding of the process of object detection as a beginner of this area.

The implementation of chosen traditional detection methods are actually much more complex than what has been described, I have tried many method to improve final detection result like trying different training data volume to experience the impact of the data complexity and quality to the detection result, I have also tried to combine the detection of Haar-like feature with human skin detection to reduce the background false detection, but none of these methods worked well. It is the failure during implementation of Haar-like feature detection that gave me a clear understanding of the limits of Haar-like feature detection as well as other traditional method, besides, I also learned quiet a lot of academic knowledge of this computer area, which helped to implement YOLOv3 detection with less blocks.

Due to the limited data resources, this project just implemented one type of unsafe behavior detection, but instead I put much more work into study the theory of the algorithm and try my best to make some changes to improve the detection result, though the improvement is just a little, but the effort behind and what I have learned is more than just a little, and I do enjoy the process of exploring method to make my work better.

5.3 Conclusion

The first aim of the project to train and evaluate different algorithms have been achieved thoroughly and generally in a scientific way despite several mentioned but unnecessary factors of evaluation are not included in the process. The modification of the detection process to improve the result though limited but it is still beyond expectation in improving the precision of model, which I found it helpful in a practical usage. The final detection software has been implemented to cover all the functions required in the aim and object, besides, the software is furtherly developed to be able to set different parameters used in the detection

and is allowed to load other trained model for detection, which a future study and development can be easily conducted. An additional function of object tracking is also development, though not practically useful for now, it is potentially helpful for the feature detection tasks. So generally, there’s some improvement which can be made to the final software, but it is still a successful one with all the objectives accomplished and several additional development.

List of References

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

[2] Mr. Sun. Introduction to basic knowledge of target detection. Available at: https://www.cnblogs.com/xiaoboge/p/10544336.html. 2019.

[3] OpenCV: Face Detection using Haar Cascades. Docsopencvorg. 2017.

[4] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In CVPR, 2005.

[5] Fm, Fei. Introduction of two stage target detection algorithm. Available at: https://blog.csdn.net/weixin_39568744/article/details/92798440. 2019.

[6] SIGAI. Summary and Prospect of the latest development of target detection. Available at: https://zhuanlan.zhihu.com/p/46595846. 2019

[7] MoLinFeiGong. mAP in target detection. Available at: https://www.cnblogs.com/gezhuangzhuang/p/10547504.html. 2019

[8] Smile Liu. Traditional target detection algorithm. Available at: https://zhuanlan.zhihu.com/p/53364144. 2019

[9] Smile Liu. The introduction, advantages and disadvantages of several deep learning target detection algorithms. Available at: https://www.zhihu.com/collection/362587276. 2019

[10] Joseph Redmon, Ali Farhadi. YOLOv3: An Incremental Improvement. Available at: https://pjreddie.com/media/files/papers/YOLOv3.pdf. 2018

[11] Lienhart R, Maydt J. An extended set of haar-like features for rapid object detection. International Conference on .IEEE, 2002.

[12] Yan wu, Zhu Jie. Algorithm of pedestrian detection with multiple classifiers ensemble based on Haar-like features. 2017.

[13] Viola P, Jones M. Rapid object detection using a boosted cascade of simple features. Computer Society Conference on IEEE, 2001.

[14] Freund Y, Schapire R, Abe N. A, short introduction to boosting. Journal-Japanese Society For Artificial Intelligence, 1999.

[15] Robert E. Schapire. The strength of weak learnability. Machine Learning, 1990.

[16] Mu Zhan, YoloV3 depth analysis, Available at: https://blog.csdn.net/leviopku/article/details/82660381. 2018

[17] Lv Dou, Analysis of the network structure of Yolov3 , Available at: https://blog.csdn.net/qq_37541097/article/details/81214953. 2018

[18] Ayoosh Kathuria. How to implement a YOLO (v3) object detector from scratch in PyTorch. Available at: https://blog.paperspace.com/how-to-implement-a-yolo-object-detector-in-pytorch/. 2018

[19] Real Time Streaming Protocol. Available at: https://en.wikipedia.org/wiki/Real_Time_Streaming_Protocol. 2018

[20] TeddyZhang. Multitarget tracking: sort and deep sort. Available at: https://blog.csdn.net/XSYYMY/article/details/81747134. 2019

[21] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, Ben Upcroft. SIMPLE ONLINE AND REALTIME TRACKING. Available at: https://arxiv.org/pdf/1602.00763.pdf.

Appendix A
External Materials

The external database used:

Human head1: https://download.csdn.net/download/NcepuKZH/11988669.

Human head2: https://linux.ctolib.com/HCIILAB-SCUT-HEAD-Dataset-Release.html.

The algorithms used:

Yolov3: https://github.com/qqwweee/keras-yolo3.

Deep sort: https://github.com/nwojke/deep_sort.

Library used:

Appendix B
Ethical Issue Addressed

The data used may cause ethical issue since it contains human head and surveillance camera images, which the links are all listed above, and all the material used are open to the public. Still there are few thousand image data which are crawled form the website, which may cause ethical issue too, the pages crawled are all picture searching engine, which specified to be allowed to craw.

Appendix C
Code Repository

The project can be viewed at GitHub:

https://github.com/XiaoFandi/Final-year-project.

Appendix D
Software Manual

Suggested Environment settings:

1. CUDA: 10.0
2. cuDNN: 7.6.5
3. Python: 3.7
4. Tensorflow-gpu: 1.14.0
5. OpenCV (python): 4.2.0
6. Keras: 2.3.1

Using the program:

The software is developed in Windows environment, which is also suggested to run it in Windows.

Open cmd and get into the root repository, type: **python main.py**

A UI will be presented, which have five functions:

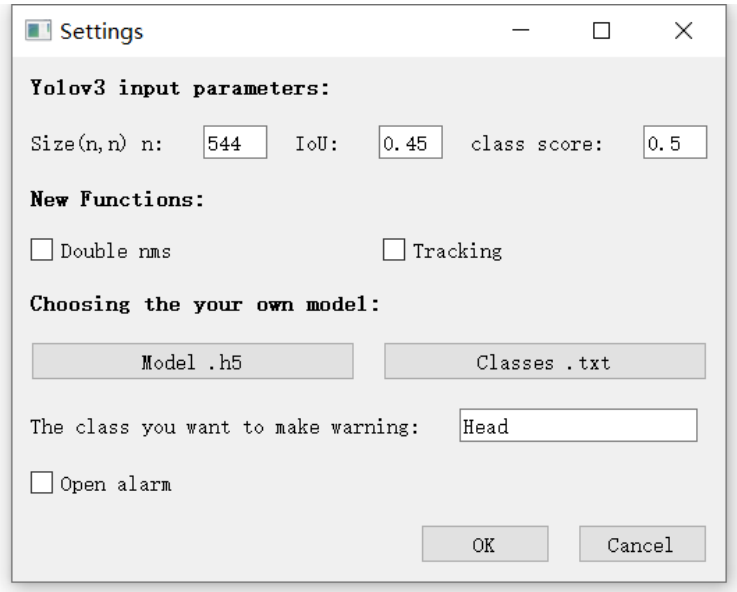
1. Setting: for this function you can set input size, IoU threshold for NMS and class score.

You can also try the new functions which are descried in the report, double NMS in bboxes filtering prcess, change form detection into tracking. These two functions are not applied in the default setting. If you want to use it, you have to tick it in the setting.

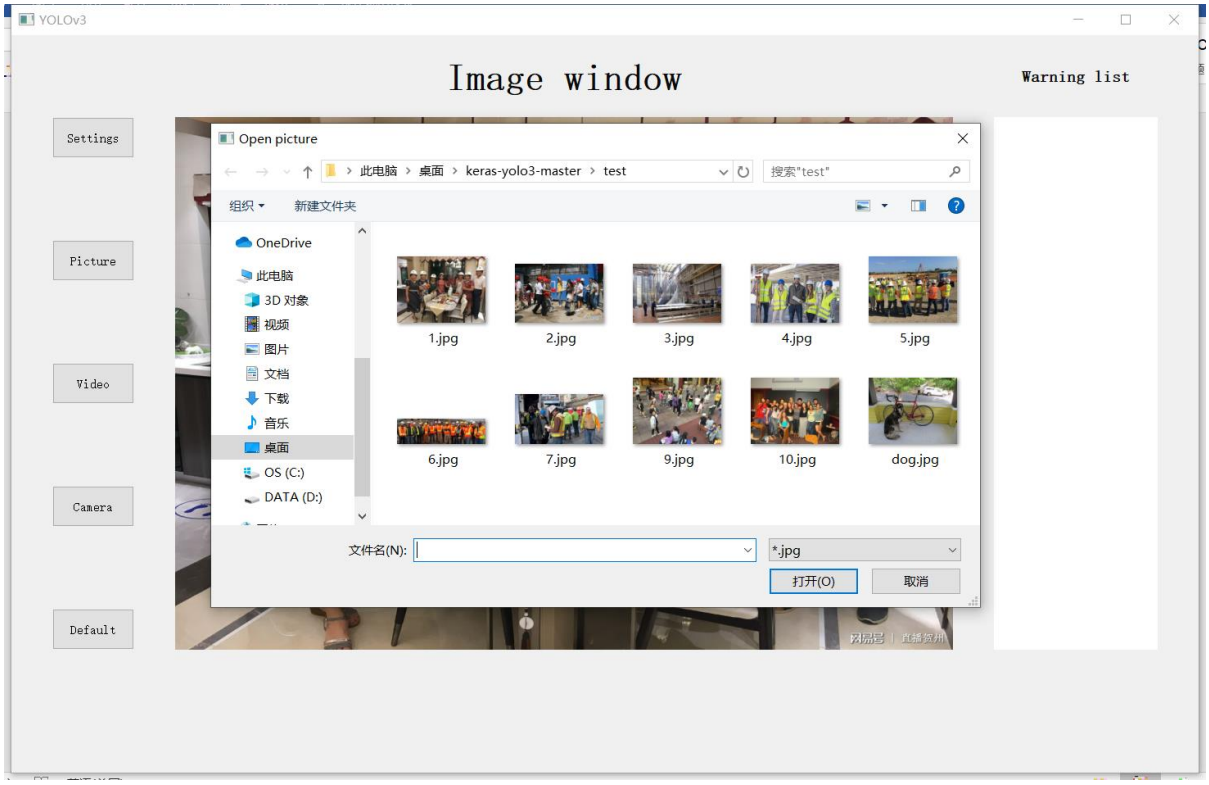
You can choose your own trained model and classes for detection by click the button, but once you choose the your own model file, you have to also choose your class file, vice versa.

Last you can choose the class you want to make warning while detecting by input the class name.

And since the alarm function can slow down the detection significantly, it is also not applied in the default settings, but you can see warning in the warning list window on the right side of the image window. if you want to try the alarm, just tick it.



2. Picture, video, camera: for these three functions, just click the button and choose the image, video or input the URL for step and you can see the detection in the main window.



3. The last default button is used to set all the setting back to original, incase you forget want you have set.