REPUBLIC OF TURKEY
YILDIZ TECHNICAL UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

DETECTION OF PEDESTRIAN PROFILES AT TRAFFIC LIGHTS WITH MACHINE LEARNING

16011093 — Halil İbrahim Uluoğlu
17011703 — Kadir Şahin

COMPUTER PROJECT

Advisor
Assist. Prof. Dr. Hamza Osman İLHAN

December, 2020

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

YOLO	You Only Look Once
GUI	Graphical User Interface
PIR	Passsive Infrared
USA	United States of America
CPU	Central Processing Unit
GPU	Graphical Processing Unit
FP	False-Positive
FN	False-Negative
HOG	Histogram of Oriented Gradients
SVM	Support Vector Machines
COCO	Common Object in Context
R-CNN	Rural Community Carbon Network
CNN	Computer and Communications Network
HD	High Definition
CUDA	Compute Unified Device Architecture
YAML	YAML Ain't Markup Language

LIST OF FIGURES

LIST OF TABLES

1
Appreciation

Throughout this project, we would like to express our gratitude and respect to the esteemed Assist. Prof. Dr. Hamza Osman ILHAN, who did spare us his valuable knowledge and time.

2
Abstract

One of the main problem in the modern world is static traffic light time systems for pedestrians. Static systems are not responsive and not adjustable for various kinds of pedestrian traffic lights. Therefore, arranging traffic lights statically causes many not welcome problems such as long time waiting, short time waiting, anomaly time between two time periods, bad timing for different daytime periods. These problems can be solved by dynamic traffic light time considering different types of pedestrians and daytime periods using state-of-art machine learning methods.

The topic of this project is to detect pedestrians in a real-time system and making traffic light time decisions for traffic lights based on different categories of people and considering different daytime periods using advanced detection algorithms. In the project, the training limitation for the system has been chosen as a single point of view and collected data from the single point. For the first time in the literature, a data set has been successfully created using various labeling systems from a traffic light with a camera view. Using this data set, the system has managed to detect with a high percentage of pedestrians according to the right category and simulate the time of traffic lights in real-time. As a result of this project, it is shown that dynamic time traffic lights can be used in the future.

3
Prologue

According to the research of Grand View Research Company, intelligent transportation systems will grow rapidly 5.8% between 2020 to 2027 in North America; Europe; Asia Pacific; South America; Middle East Africa regional areas. Also, revenue of intelligent transportation systems will increase by approximately 12 billion $ [1].

Based on this research, intelligent transportation and intelligent traffic systems will affect a significant role in human lives in the future. This will lead to demand of intelligent signalization systems. There were traffic lights and signalization systems since even creation of vehicles in order to make regulations. However, the traffic light systems have used the same static time arrangement for every situation all around the world. Moreover, they have not considered to the specification of pedestrians near the crosswalk of the road. Therefore, every light signalization time is fixed. For instance, it arranges same time few pedestriansvehicles or lots of pedestriansvehicles. It causes to waste of time for each pedestrian and also vehicles on the road. Additionally, it causes more and more gasoline oscillation and on account of it accelerates air pollution and nature disasters.

In this project aims to solve this problem by providing dynamic traffic signalization system for detecting pedestrians by using image processing and machine learning technologies. The concept of this project is to detect pedestrians and categorize based on behavior of pedestrian on the crosswalk. In the project, pedestrians were categorized as challenged(unable to do particular things without difficulty), unchallenged and child. The system detects these categories and it calculates the signal time from pedestrians who waiting for the signal time at the moment.

4
Preliminary Examination

The first solution initiative in history on behalf of the traffic flow management was the traffic signs that started to be used in London in 1868 with the words "Stop" and "Pass" writing on it and which is controlled by a human waiting under the sign. The first traffic lights similar to nowadays' lights were electric and 3-color traffic lights, which were started to be used in Cleveland - USA in 1914 and developed by Garret Morgan. These lights, which were initially controlled by a human, were later turned unmanned by electronic advances. More recently, traffic lights with time indicators, semi-actuated or fully-actuated traffic signals have started to be used.

In actuated-signals, the triggers for cars are magnetic induction sensors that placed under the road in front of the light-pole. In this system, cables that can detect metal, trigger the light controller and create a red light request in the main direction of the intersection. On the pedestrian side of the actuated-signal systems, there are "request to cross" buttons on the traffic light-pole (passive pedestrian detector). Since the light duration triggered by these requests is still a fixed-time, it can not be adaptive enough. They are costly and unpractical to install and maintain as they will cause lane and pavement to dismantle. In addition, induction circuits work unstable for different weight cars. [2]

A "real-time adaptive signal control model" was developed to enable Actuated Signals to work "adaptively" to traffic density instead of working "pre-timed". In this model, the "current phase state, signal-timing plan, and detector output" data obtained in a signal cycle in the actuated signal control model were used to estimate the signal parameters of the next cycle. However, this model was only tested in a simulation with "fully actuated, that is, with sensors in all lanes" and less frequent traffic conditions. So its effectiveness for other conditions is not yet clear. [3]

In this project, pedestrian detection and classification methods will be analyzed and the most suitable detection method for smart traffic signalization systems will be determined. Some primitive methods for Detecting Pedestrian Presence are explained

in the next section.

4.1 Pedestrian Detection Devices

- **Infra-red Beam Counters:** Infra-red beam counters are the devices which has two different ends as IR receiver and the transmitter and works based on the approach where the object between them creates a cut between the two ends. This solution can be used in places such as turnstiles where only one person can pass. In addition, it can only count, not classify.
- **Passive Infra-Red (PIR) Counters:** PIR Counter detects the amount of infrared radiation reflected from objects in the environment. It counts according to the change in radiation. It also can not classify the people.
- **Piezoelectric Pad:** Piezoelectric pad is a material that creates an electric charge between its surfaces according to the pressure applied on it. It detects according to a threshold value of the voltage that rises as the pressure increases.
- **Laser Scanner:** Laser scanner detects whether an object exists in the environment, by sending laser beam and calculating the return time of light.

All of the methods above are used for simpler targets, such as the number of customers visiting the shopping center or the number of passengers entering a subway stop. However, the only way to determine the profile of pedestrians waiting on the sidewalk at an intersection or to examine the behavior of people on a street for safety is to examine the image of the environment with image processing techniques and make inferences with machine learning methods. [4]

4.2 Machine Vision

Some researchers basically divided person identification and counting with Machine Learning into 3 steps [5][6];

- 1. Determine whether any potentially interesting objects have entered into the scene (alerting phase)
- **2.** Track their motion until the counting line is reached (tracking phase)
- **3.** Establish how many people correspond to tracked objects (interpretation phase).

- **Step 1.**It is the separation of the people entering the scene by examining the image created by extracting the instant image from the background image.This decomposition process is done with two basic methods as "motion-based" and "shape-based". Motion-based approaches use human-specific rhythmic features and movement patterns, it is not possible to detect stationary pedestrians. Shape-based approaches use the general shape characteristics of the human body, and detection is possible for both stationary and mobile people. Processing power cost is high.

- **Step 2.** It is made by establishing connections between objects detected in consecutive frames.

- **Step 3.** It is analyzed whether the detected objects in the previous steps are real objects. For example, in order to understand whether the detected objects in crowded places are real people, the relevant part of the detected object can be passed through a face detection algorithm. Or, for instance, in order to accurately detect partial human body images, the shape-based model can be trained with parts of the human body as well as all human body samples. (head, arms, legs)

Bu, Fanping (2007), provides a comparison table between pedestrian detection devices mentioned above. It is shown at the figure-4.1 [7].

Counter	*Pros*	*Cons*	*Manufacturer and Cost*
Infra-red beam counter	Cheap and widely available commercially; Low power consumption; Easy installation; Highly portable.	Infrared beam counter cannot differentiate pedestrian and other objects; Transmitter and receiver need to be aligned carefully to ensure the reception of beam at the receiver end; Both transmitter and receiver should not be installed on a flexible structure; When several pedestrians cross the counting beam simultaneously, they are only registered as one count.	Jamar Technologies Inc $790
Passive infra-red counter	Cheap and widely available commercially; Low power consumption; Not affected by wet or foggy weather; Counter with multiple sensor arrays could achieve performance comparable with computer vision.	Single or double sensor counter cannot distinguish between individuals and groups; Temperature can affect counter performance; Limited coverage area.	Irisys $1400 for counter with multiple sensor array EcoCounter $2000 for counter, $600 for software
Piezo-electric pad	Low maintenance cost; Low power consumption; Capable of counting pedestrians on sidewalks.	Need physical contact between pedestrian and pad; Sub-surface installation is expensive; Limited coverage area; Some of products cannot differentiate between single pedestrian and group of pedestrians.	Eco-Counter Cost estimate not available
Laser scanner	Accurate range measurement; Can differentiate pedestrian according to their height; Easy setup; Large coverage area.	Expensive; Performance could be affected by different weather conditions.	LASE GmbH Around $9000 for counter only
Computer vision	Large coverage area; Has the potential to count accurately in various conditions such as crowded pedestrians, different lighting conditions; Can be manually reviewed to collect pedestrian characteristics; Easy installation and setup; The video can be recorded for manual review.	Most commercially available products are intended for indoor setting; The difficulty of counting pedestrians in crowded settings has not yet been resolved; The performance can be affected by different environmental conditions if not designed properly.	Video Turnstile Start from $1230

Figure 4.1 Summary of Automated Pedestrian Counting Devices

4.3 Pedestrian Detection Methods in Computer Vision

Hyunggi Cho proposed a detection system which is based on “star-cascade algorithm” [8]. This approach, relies on some assumptions, for example the position of the camera is known and in which areas of the frame the objects can be detected are also known. After these assumptions, some relationships are analyzed such as

- ‘pedestrian height in pixels’ vs. ‘distance from a vehicle’
- ‘pedestrians’ foot position in images’ vs. ‘distance from a vehicle’.

These relationships and geometry analysis accelerates the detection and eliminates the potential false-positives from irrelevant image space. [9]

David Fernández proposed a binocular (multi-camera source) vision-based pedestrian recognition system based on Support Vector Machine (SVM). This system uses a

example-based technique which means that the model is trained with many samples for every different human body part independently (distributed approach) and also uses a second camera to see the objects from different angles. The system has 2 stages.In first stage, 6 different classifier for 6 different body parts work. This makes the learning process simpler, because a single classifier has to learn individual features of local regions in certain conditions. The model has 6 different sub-regions. In the second classification stage, the outputs of 6 classifier merged into a simple classifier which outputs the final result. A comparison table against a Holistic approach they provided is given in the table-4.1. [10]

Table 4.1 Comparison of Holistic approach

	Distributed SVM Classifier			**Holistic SVM Classifier**		
	detection rate	***FP rate***	***FN rate***	***detection rate***	***FP rate***	***FN rate***
Coocurrence over normalized image	0.7437	0	0.2563	0.7789	0	0.2211
Coocurrence over normalized image	0.8643	0	0.1357	0.8593	0.0653	0.0754
Canny image	0.7940	0	0.2060	0.7236	0	0.2764
Magnitude orientation	0.7236	0	0.2764	0.7136	0	0.2864
Normalizad Orientation Histogram	0.9246	0	0.0754	0.8894	0.0402	0.0704
Texture Unit Number	0.8593	0	0.1407	0.7136	0	0.2864

Mohammed Saleh Ali Muthanna designed a system by combining a few different libraries for image processing and object detection. He indicates that he can process approximately 6 frames/second for pedestrian detection by using HOG gradients using OpenCV 3.4 on a test system equipped with Intel Core i7 6700, 8 GB of memory and NVIDIA GeForce GTX 960 4 GB of video memory. Using Haar cascades in OpenCV, he gets 11 frames/seconds and notes that these models are unable to properly detect people from different postures. Also these models sometimes defines some objects as human which are not human in real. Due to these disadvantages, he uses Tensorflow Detection Model Zoo, trained on the COCO (Common Object in Context) data set. Without using graphic acceleration, he gets approximately 15 frames/second. To improve the performance, he uses an additional YOLO(You Only Look Once) library which works with models trained with two different data sets VOC 2007 + 2012 and COCO. His system's detection performance results are given in table-4.2 and table-4.3 [11].

Table 4.2 Pedestrian Traffic Density Result

Test Number	Real Number	Amount Counted	Accuracy
1	3	3	100%
2	9	8	88.8%
3	18	15	83%

Table 4.3 Presents the Results of Tests on the same Pedestrian Areas in Low Light

Test Number	Real Number	Amount Counted	Accuracy
1	4	3	75%
2	7	4	57%
3	11	6	54%

After reviewing these related-works, YOLO Algorithm was chosen as the most suitable approach to achieve the project target. YOLO has several advantages against other object detection algorithms:

Other old detectors based on Haar Cascade Classifier or SVMs, or newer approaches which use Fast R-CNNs or Faster R-CNNs have slow response time for some detection values compared to YOLO.[12]

YOLO firstly divides the frame as a grid and runs seperately for every cell of this grid, while some of other methods use Sliding Window methods which require much more power and time cost. [13]

YOLO Algorithm can detect objects accurately in approximately 25 frames/second on HD image and this is one of the highest rates compared to others. [14]

In YOLO, each cell can predict 3 “bounding box” and later by applying “non max suppression algorithm”, false positive or duplicated bounding boxes are filtered out.

5
Feasibility Study

In this chapter, detailed analysis of the technical, resource, legal and economic feasibility studies related to the project are included.

- **Technical Feasibility:** Under this heading, technical feasibility analysis is divided into three sub-headings: software, hardware and communication feasibility.

  As a result of the related-works analysis, YOLO Algorithm was determined to use [15] which is capable of real-time pedestrian detection system. YOLO is an object detection tool which was introduced under Darknet framework in 2016. The project is developed using YOLO v5. Since YOLO performs better on NVIDIA CUDA GPU cores, CUDA drivers must be installed on running environment. YOLO can work on Linux, MacOS and Windows but Linux-based Google Colaboratory development environment was decided for this project which is provided by Google for free and allows users to use Tesla K80 GPU's. Required python packages for YOLO object detection system is given in the following table.

Table 5.1 List of required packages

Package	Version
Cython	
matplotlib	3.2.2
numpy	1.18.5
opencv-python	4.1.2
Pillow	
PyYAML	5.3
scipy	1.4.1
tensorboard	2.2
torch	1.6.0
torchvision	0.7.0
tqdm	4.41.0
PyQT5	

Recommended hardware configuration for YOLO algorithm is given in the table below. Online video streams can be used for YOLO. Therefore, any camera hardware was not needed.

Table 5.2 Specification of system hardware

Component	Specification
GPU	NVIDIA GeForce GTX 10xx Series+ GPU with CUDA Cores
CPU	A new generation multi-core processor (Intel Core i7, i9, Xeon+)
RAM	8 GB + DDR4
Storage	120 GB+ SSD

Due to the 2020 Covid pandemic conditions, works were done online from home while developing this project. Google Colab was preferred as a development platform that allows users to make a teamwork. Additionally, Google Meet and Zoom communication software were used for meetings.

- **Time Planning:** The scope of the project was to develop a Pedestrian Detection prototype and as future work, making an Optimized Traffic Signaling System Simulation for Challenged Pedestrians. The project is carried out by 2 Computer Engineering (B.Sc) students in phases. The progress schedule of the project is as follows.

Figure 5.1 Gantt Diagram of the Project

•

1) Literature review and algorithm decision- 2 week
2) System analysis and design - 1 week
3) Collecting image data for labelling -2 week
4) Training - 1 week
5) Detection - 1 week
6) Optimization of detection - 1 week
7) Decision making algorithm - 1 week
8) Reporting - 1 week

- **Legal Feasibility:** Since all the frameworks used on this are open-source, there is no legal restriction of making this project such as Copyright Terms. On the other hand, to make this project's real-life implementation, legal permissions should be taken from local authorities such as Mayors and Governors and also the Ministry of Transport of the related countries.

- **Economic Feasibility:** As indicated in the previous chapters, the scope of this project was determined a simulation. Hence, there is not a revenue model and also there is not a cost since free and open-source tools and resources were used in the project. In case of making a concrete implementation, the potential costs for camera establishment, network infrastructure and server system may come up.

6
System Analysis

Within the scope of the project, it is essential to recognize and detect pedestrians near the road, on the road, or the relative path of the road. For this purpose, the machine vision of the project needs to be highly trained with quality data. It needs to identify small changes of the pedestrians because of the angle of the machine vision(cameras on top of the traffic lights). After the detection, within decision-making algorithm needs to make a decision according to real-time pedestrian data.

For child category: It requires to shorten red light time because they are more active than other genres. Also, it needs to be responsive due to a lack of child control.

For unchallenged category: It is the regular pedestrian profile category. In this categorization, it can be important the number of pedestrian in this profile.

For challenged category: It requires longer red light time because they are less active than other genres. It contains disabled and senior people. Also, it needs to identify small pinpoints for this category. For instance, they can have a walking stick, wheelchairs, white hair due to senility, etc. .

For this purpose, the main requirements are high-quality machine vision, powerful hardware for machine learning processes, a GUI to show real-time detected pedestrians, and the time for related detection. For the high-quality machine vision main requirement, in this project, live demo video streams can use that have enough and varied amount of pedestrians. For powerful hardware, it can be used in cloud environments which companies provide such as Google Colaboratory. For a GUI program, it can be used PyQT5 Framework in Python. In this system, it will be enough only one user profile. An admin user profile can do required connections between platforms. This profile will control by the program itself. So, it will automatically run the background processes and will output the targeted results.

At the end of the project, It is considered that the project is successful rely on these substances:

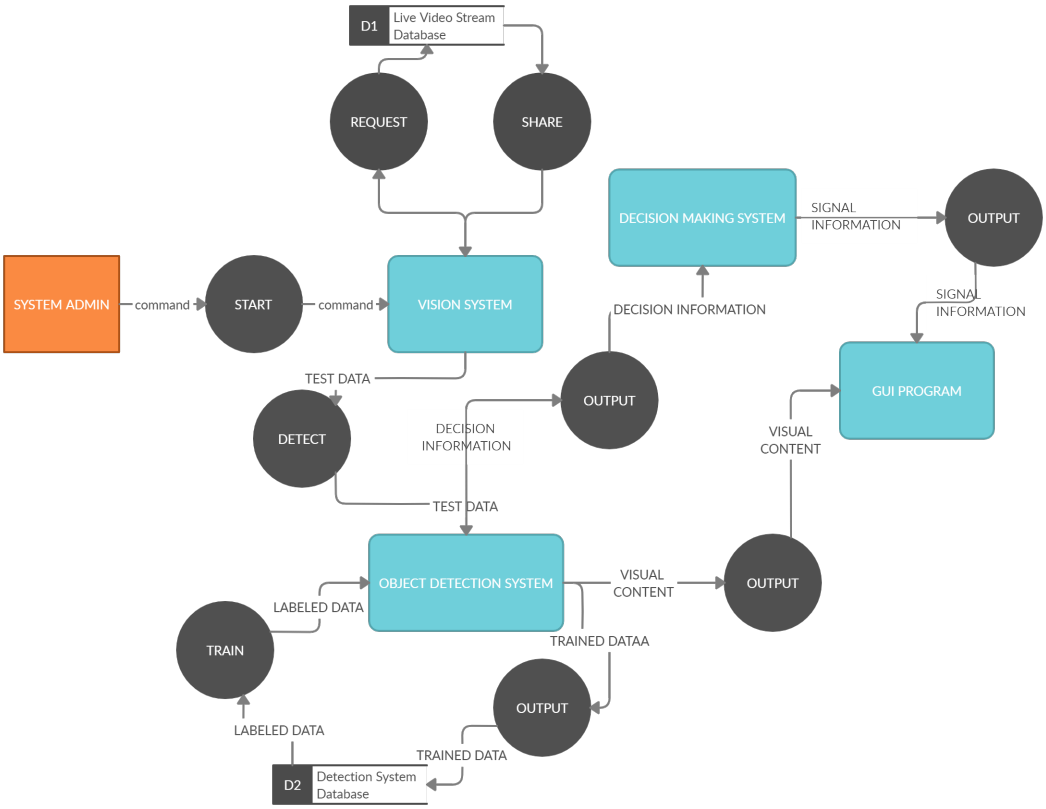

Figure 6.1 Flow Diagram

- **Criteria 1:** First of all, the system must be able to detect pedestrians from angles of cameras at the top of traffic lights.
- **Criteria 2:** Detection must follow with a high individual detection rate for each pedestrian in the category. So, the detection recognition success criteria are expected above and equal to 80% for each pedestrian.
- **Criteria 3:** Within a high success recognition rate, it should be processed by the decision-making algorithm. The algorithm must calculate the traffic light time for in the particular instant pedestrian group.
- **Criteria 4:** The final criteria is able to see marked real-time video output and related traffic light timer on a GUI program.

7
System Design

In this chapter, design of the system that belongs to the project is reported. The chapter contains Software Design, Database Design, and Input-Output Design.

7.1 Software Design

In this project, the software layer can divide into two pieces. The first part of the pieces belongs to YOLO object detection method. YOLO is an open-source, state-of-art, and the most advanced object detection method. It is 1000x faster than R-CNN and 100x faster than Fast R-CNN.

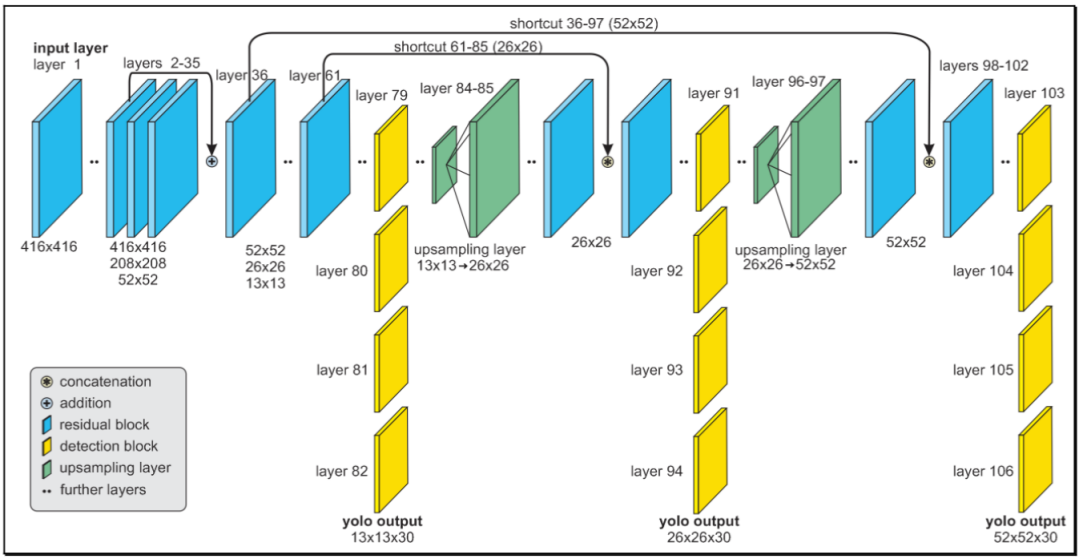

Figure 7.1 YOLO Network[14]

Other part is to make a GUI program using at decision-making algorithm. The YOLO algorithm starts extracting single image from single image. Then it scales image to 416*416. It uses neural network and non max suppression approach in order to detect objects.

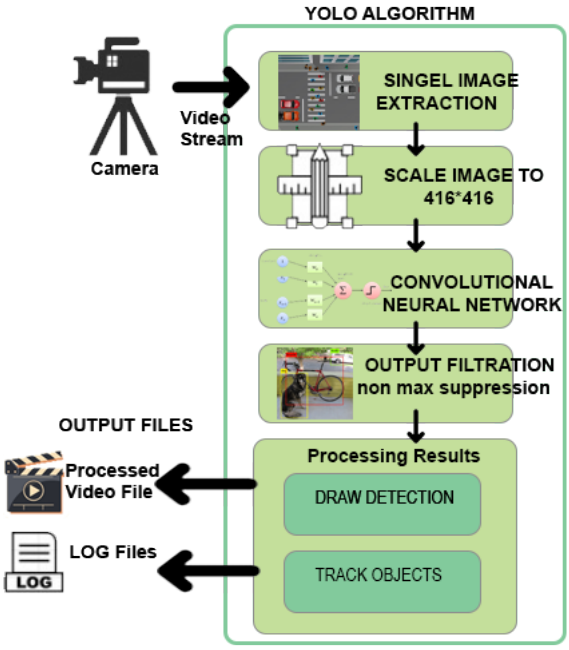

Figure 7.2 Overview of YOLO algorithm

In the project, because of the lack of a powerful hardware system, a cloud environment has been used. To accomplish the goal of the project Google Colaboratory fits the requirement. Google Colaboratory is a development environment for write and executes Python on a browser powered by Alphabet Company(Google). It is reliable to use for data science, machine learning, etc.[16]. Google Colab is a free environment with some resource limits. It allows using the required memory, GPU, and CPU. In the project, these specifications are enough to make work the system of the project.

Figure 7.3 Google Colaboratory Environment

On the Application side, PyQT5 Framework based on Python have been used. It

contains required outputs and information about the system.

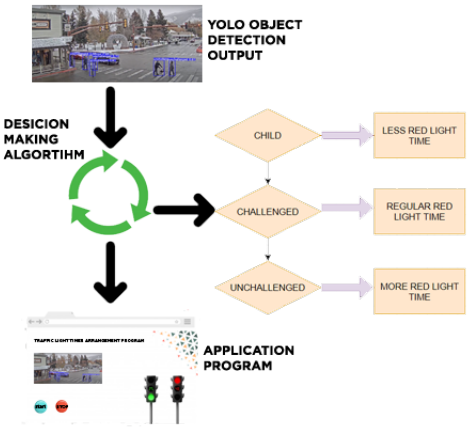

Figure 7.4 Application Design

7.2 Database Design

In this project, Google Colaboratory and Google Drive were used for the database system. All files and information sets are stored in Google Drive. An integration coded between Google Colab and Google Drive. Moreover, the Decision-making algorithm and application program use Google Drive to store the data.

8
Application

In this chapter, the steps of the project will show in detail. First of all, it needs to clone YOLOv5 repository from Github. After downloading it will go to a specific directory in order to install requirements for YOLO.

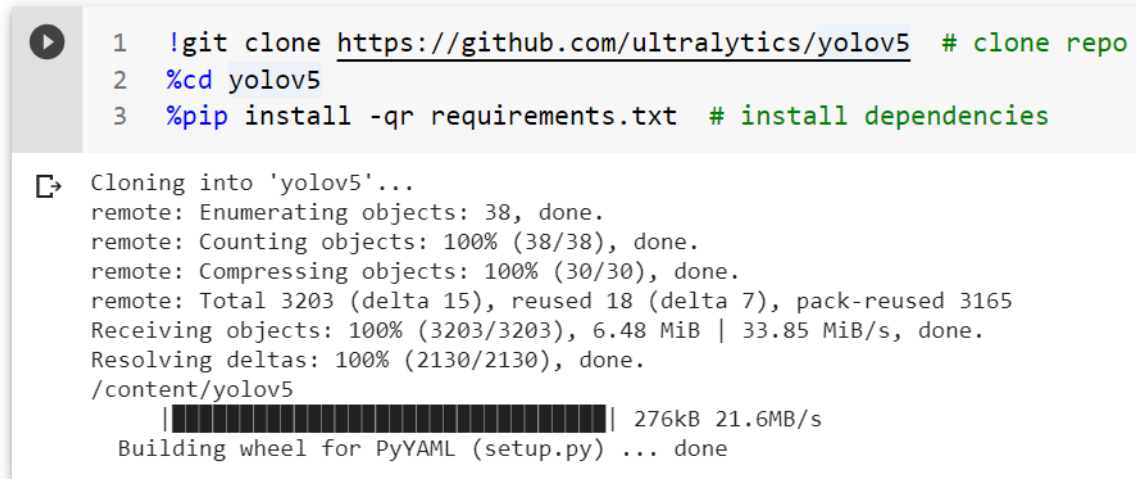

Figure 8.1 Clone and install requirements for YOLO

With the requirements of YOLO, the project needs some specific libraries such as Torch, Google utils from Utils, Display from IPython.

Figure 8.2 Necessary Libraries

Google Colab environment requires additional permission to connect and mount Google Drive. Also, "force_remount" parameter used for mount if it mounted before.

```
[10]  1  from google.colab import drive
      2  drive.mount('/content/gdrive' , force_remount=True)

Mounted at /content/gdrive
```

Figure 8.3 Mounting Google Drive into Google Colab

Secondly, "pedestrian.yaml" configs the file system of train images and labels and it indicates the name of the categories.

```
# COCO 2017 dataset http://cocodataset.org - first 128 training images
# Train command: python train.py --data coco128.yaml
# Default dataset location is next to /yolov5:
#   /parent_folder
#     /coco128
#     /yolov5

# train and val data as 1) directory: path/images/, 2) file: path/images.txt, or 3) list: [path1/images/, path2/images/]
train: ../pedestrian/images/train2017/
val: ../pedestrian/images/train2017/

# number of classes
nc: 3

# class names
names: ['child','unchallenged','challenged']
```

Figure 8.4 pedestrian.yaml file

Lastly, It downloads images and labels that created for this project from mounted Google Drive.

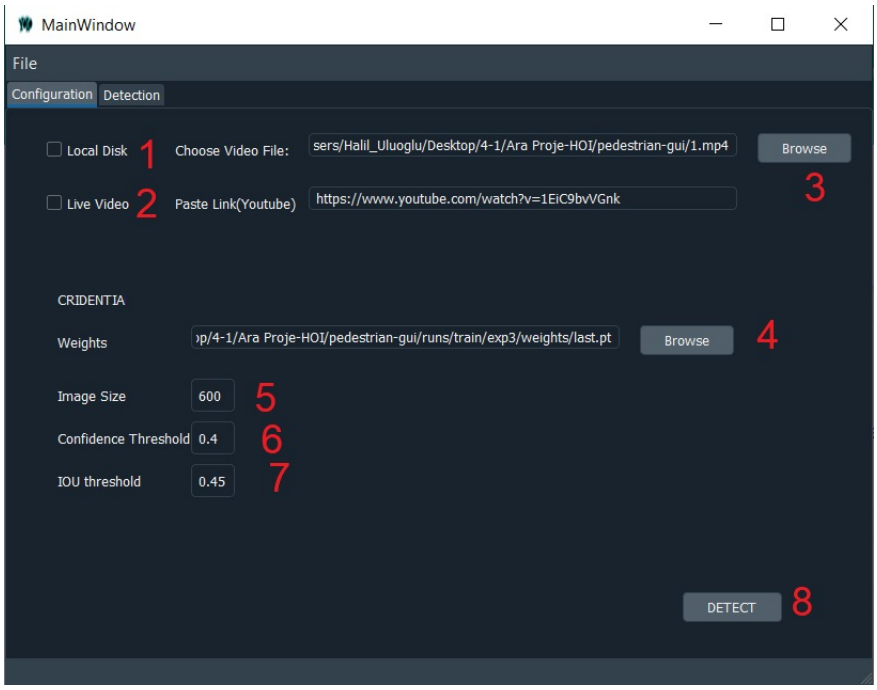

Figure 8.5 Configuration Window of the Application

In configuration window of the GUI, necessary paramaters are set by the user.

By checking the checkboxes shown as number 1 and number 2, it is set whether the detection system will work on recorded video or live stream video.

The button numbered with 3 opens a file browser dialog and gets the path of the recorded video selected via the file browser. The button numbered with 4 opens another file browser for the weights file.

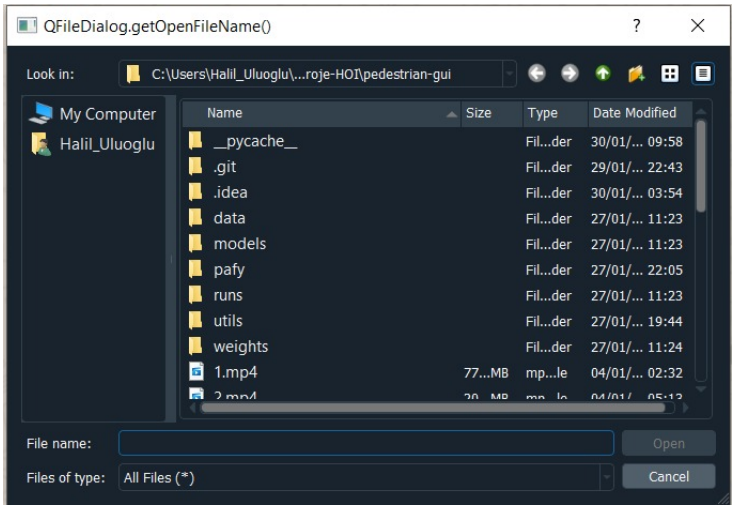

Figure 8.6 File Browser Dialog

The text boxes numbered with 5,6 and 7 arranges the object confidence threshold and

IoU threshold for the Non-Maxima-Suppression step of YOLO.

After all parameters are set, the detection is started by clicking DETECT button.

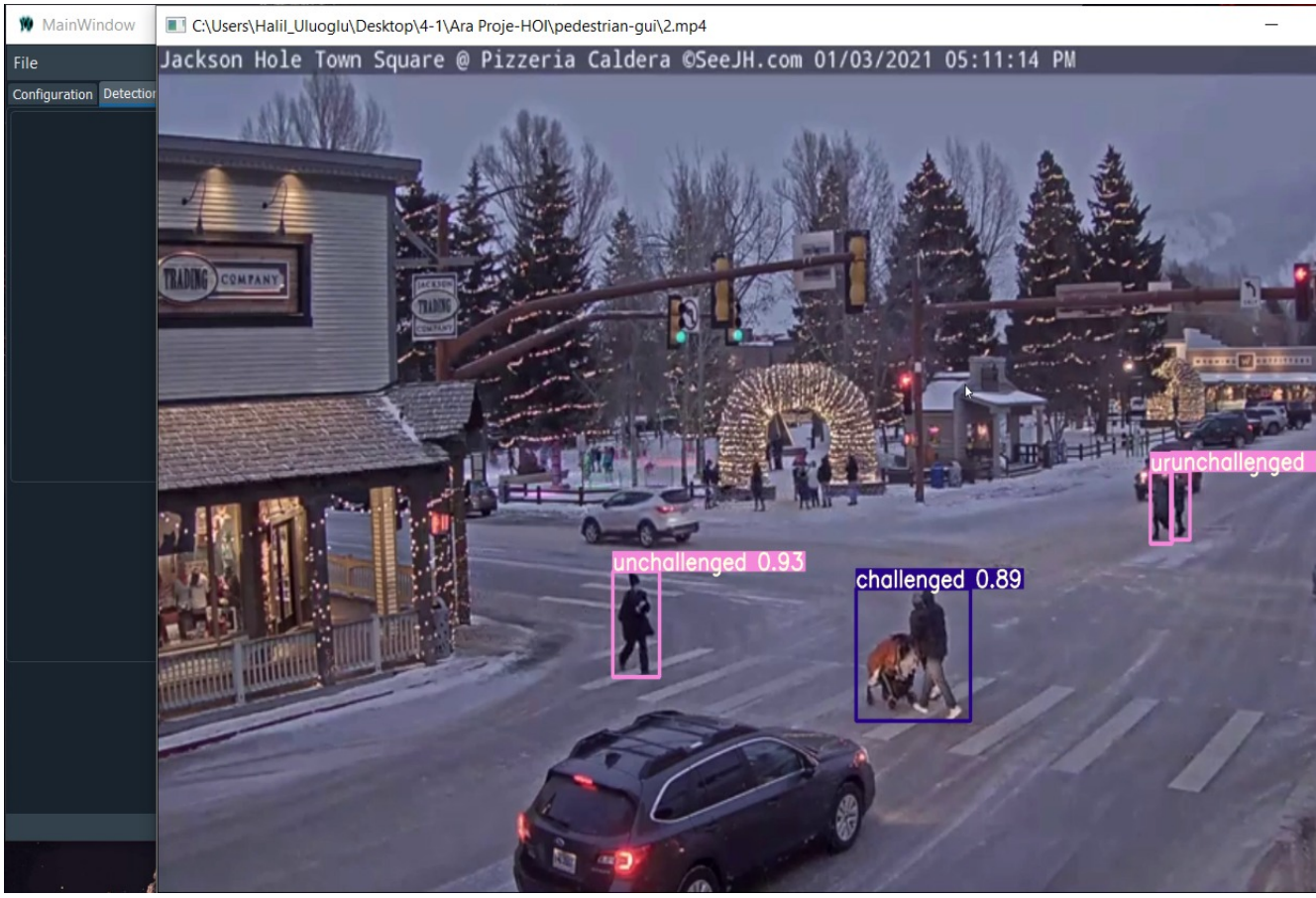

Figure 8.7 Detection Window

9
Experimental Results

During the training process of this pedestrian detection model, 3 different labeling methods have experimented. Further, for each method, the model was trained in different epoch numbers. All images of the project's dataset are collected from the live broadcast "Town Square Live Cam" provided by the "See Jackson Hole" Youtube channel. The live broadcast is reachable at the link on here.

The number of labels and frames for each class is given in the table-9.1 .

Table 9.1 Summary of labeling data by category

Labeling Method 1		**Labeling Method 2**		**Labeling Method 3**	
class	**# of labels**	**class**	**# of labels**	**class**	**# of labels**
child	200	child	200	child	297
unchallenged	200	unchallenged	200	unchallenged	1063
challenged	200	challenged	200	challenged	144
Total Label	600	**Total Label**	600	**Total Label**	1504
Total Frame	600	**Total Frame**	255	**Total Frame**	445

Labeling Method 1

In the first case, the labels of classes were taken in a way where every image includes only 1 class label. If there are multiple samples of any class in a frame, the image was cloned and separate .txt label files were created for every sample.

In both 50 and 500 epoch cases, it was realized that the system was not able to detect multiple samples of a class in a frame. It was detecting only one sample of a class in every frame and in the next frame, it was drawing another bounding box for another sample.

Labeling Method 2

In order to improve multiple pedestrian detection performance and ensure continuity of bounding boxes in the video, the labels of the image dataset were rearranged.

In this method, multiple pedestrians for every frame were marked and their labels were combined in one .txt file for every image.

This method fixed the multiple sample detection problem. There was still a problem in the detection of pedestrians with different postures and this was causing disruptions in the detection of walking pedestrians.

Labeling Method 3

In this method, multiple labels from walking pedestrians were taken. In this method, multiple labels were taken from walking pedestrians. Furthermore, labels of the unchallenged class have recollected and more apparent samples have been chosen such as pedestrians with a cane, hunchback, a stroller, etc. With this modification, tracking success of the system on walking pedestrians notably improved.

Summary of the comparison for different epochs and labeling methods are given at the table-9.2.

Table 9.2 Output summary of labeling methods

LABELING METHOD 1	**50 EPOCHS**		**500 EPOCHS**	
	Precision	0.265	Precision	0.5138
	Recall	0.8172	Recall	0.9618
	mAP@.5	0.4535	mAP@.5	0.826
	mAP@.5:.95	0.2664	mAP@.5:.95	0.7063
LABELING METHOD 2	**100 EPOCHS**		**500 EPOCHS**	
	Precision	0.2562	Precision	0.6135
	Recall	0.686	Recall	0.9552
	mAP@.5	0.4505	mAP@.5	0.9483
	mAP@.5:.95	0.2438	mAP@.5:.95	0.713
LABELING METHOD 3	**50 EPOCHS**		**500 EPOCHS**	
	Precision	0.3454	Precision	0.6729
	Recall	0.7762	Recall	0.9707
	mAP@.5	0.4897	mAP@.5	0.9594
	mAP@.5:.95	0.2539	mAP@.5:.95	0.724

10
Performance Analysis

As a result of experimental methods, the best labeling method has chosen as "Labeling Method 3". In this method, it used multiple labels for a single pedestrian while walking on the path. So, the training system managed to learn from various perspectives of a pedestrian. Therefore, it produced better detection for video input. It is given in the table-10.1

Table 10.1 Distribution of label date

Train Data (65.4%)		**Validation Data(20.6%)**		**Test Data (14.0%)**	
class	**# of labels**	**class**	**# of labels**	**class**	**# of labels**
child	217	child	53	child	27
unchallenged	667	unchallenged	224	unchallenged	172
challenged	100	challenged	33	challenged	11
Total Label	984	**Total Label**	310	**Total Label**	210
Total Frame	354	**Total Frame**	61	**Total Frame**	30

As a result of test performance, the detection system has precisely recognized the samples of the child, the unchallenged, and the challenged categories. The success result of the detection is given at the fig-10.1. Therefore, in this project, the algorithm of YOLO and labeling method 3 have reached the best outcomes for purpose of this project.

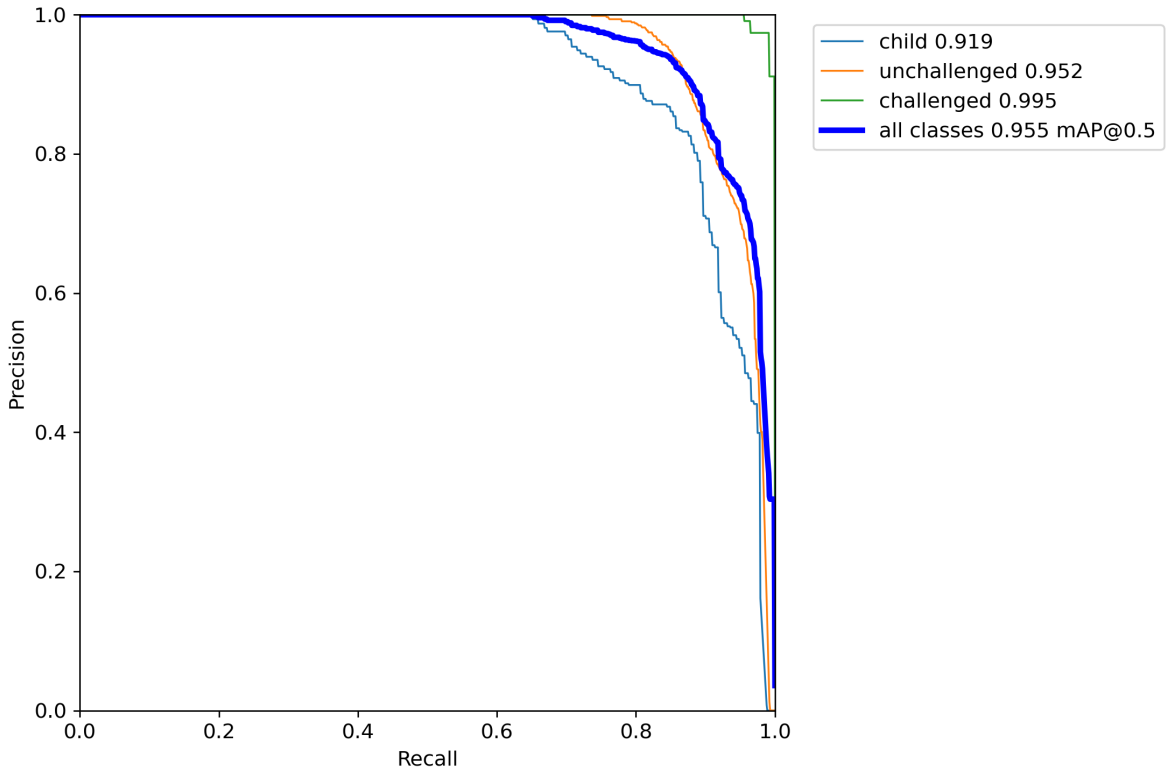

Figure 10.1 Precision Recall Curve of Test Results

11
Result

Traffic signalization systems have been in human life for over a century. During this period, they have gone through many changes. Despite all the changes, their functionality remained limited. Some technologies have been experienced to change their fixed-time light arrangement and reduce traffic jams caused by this arrangement. One of them is static traffic light systems for pedestrians. It can cause many unpleasant side effects such as long time waiting, short time waiting, anomaly time between two time periods, bad timing for different daytime periods. In this project, the state-of-art object detection technologies have been analyzed in detail and the YOLO real-time object detection algorithm has been used to contribute to solve this problem.

In the project, for the first time in the literature, a data set has been successfully created using various labeling systems from a traffic light with a camera view. Three main pedestrian category used for the detection. With the this data set, the system has identified these main pedestrian categories as aimed for this project. To get the best result, it is used diffrent labeling methods amd the best performing label method was chosen. Detection result using the best performing labeling method is shown at the figure-11.1.

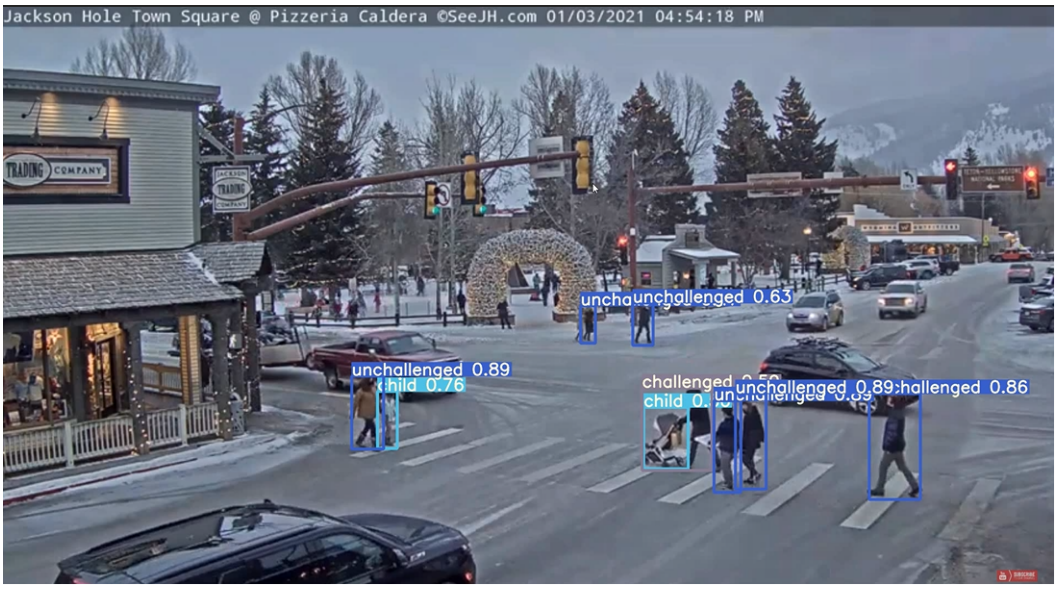

Figure 11.1 The Best Performing Detection Result

Detection result using the worst performing labeling method is shown at the figure-11.2.

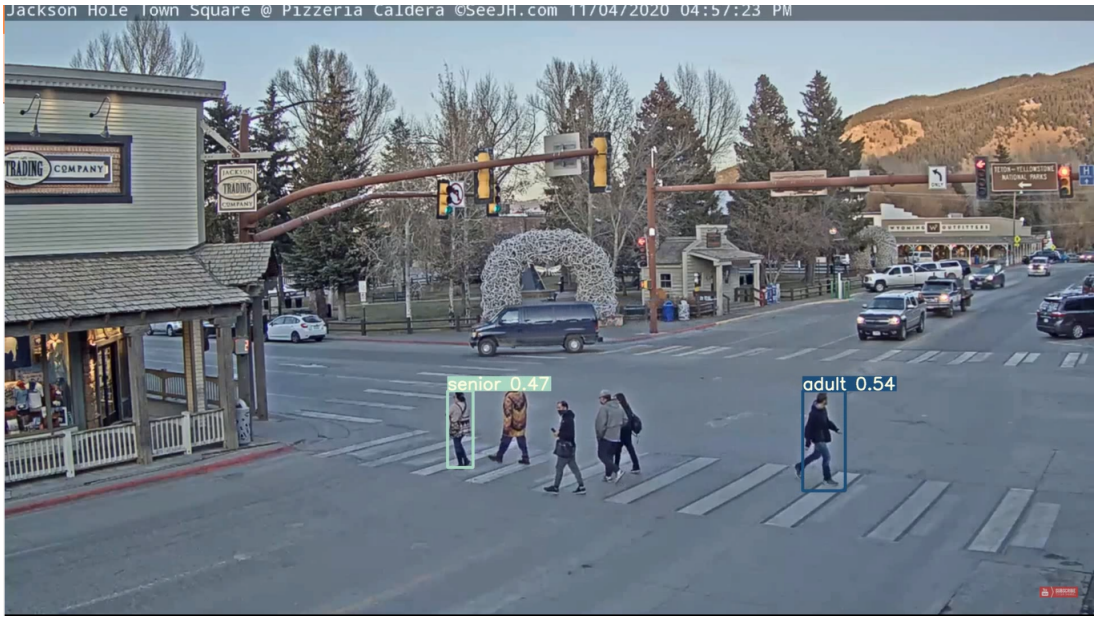

Figure 11.2 The Worst Performing Detection Result

Detection result from non-trained camera view is shown at the figure-11.3.

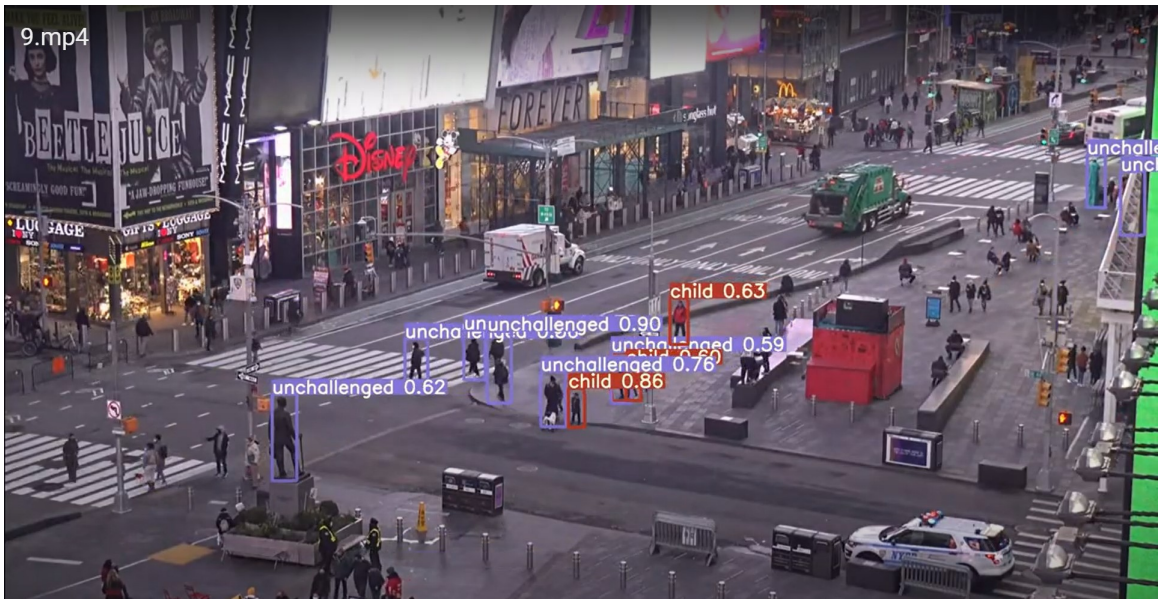

Figure 11.3 Detection Result From Non-Trained Camera View

The worst performing labeling output does not satisfy because the labels of classes were taken in a way where every image includes only one class label. The best performing output fulfills the aim of the project because multiple labels were taken from walking pedestrians and the labels of the unchallenged class have recollected. With this modification, tracking the success of the system on walking pedestrians significantly improved. For the different camera view, the detection is not enough to simulate time for the traffic light. In order to exceed this problem, it is better to get the same camera view training data and add into the previous training data set.

For the future work, it can be expandable with various camera view data and test the outcomes. Also, the aim of this project was to limit only pedestrians. However, for future works, it may be extended with vehicles such as cars, trucks, bicycles, etc.

References

[1] G. V. R. Company. (2020). "Intelligent transportation system market size, share trends by type (atis, atms, atps, apts, ems), by application (traffic management, public transport), by region, and segment forecasts, 2020 - 2027," [Online]. Available: https://www.grandviewresearch.com/industry-analysis/intelligent-transportation-systems-industry (visited on 12/01/2020).

[2] HowStuffWorks.com. (2000). "How does a traffic light detect that a car has pulled up and is waiting for the light to change?" [Online]. Available: https://auto.howstuffworks.com/car-driving-safety/safety-regulatory-devices/how-does-a-traffic-light-detect-that-a-car-has-pulled-up-and-is-waiting-for-the-light-to-change.htm (visited on 12/04/2020).

[3] X. Zheng, W. Recker, and L. Chu, "Optimization of control parameters for adaptive traffic-actuated signal control," *Journal of Intelligent Transportation Systems*, vol. 14, no. 2, pp. 95–108, 2010. DOI: 10.1080/15472451003719756. eprint: https://doi.org/10.1080/15472451003719756. [Online]. Available: https://doi.org/10.1080/15472451003719756.

[4] F. Bu, R. Greene-Roesel, M. C. Diogenes, and D. R. Ragland, "Estimating pedestrian accident exposure: Automated pedestrian counting devices report," 2007.

[5] M. Rossi and A. Bozzoli, "Tracking and counting moving people," in *Proceedings of 1st International Conference on Image Processing*, IEEE, vol. 3, 1994, pp. 212–216.

[6] J.-W. Kim, K.-S. Choi, B.-D. Choi, J.-Y. Lee, and S.-J. Ko, "Real-time system for counting the number of passing people using a single camera," in *Joint Pattern Recognition Symposium*, Springer, 2003, pp. 466–473.

[7] F. Bu, R. Greene-Roesel, M. C. Diogenes, and D. R. Ragland, "Estimating pedestrian accident exposure: Automated pedestrian counting devices report," 2007.

[8] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester, "Cascade object detection with deformable part models," in *2010 IEEE Computer society conference on computer vision and pattern recognition*, IEEE, 2010, pp. 2241–2248.

[9] H. Cho, P. E. Rybski, A. Bar-Hillel, and W. Zhang, "Real-time pedestrian detection with deformable part models," in *2012 IEEE Intelligent Vehicles Symposium*, IEEE, 2012, pp. 1035–1042.

[10] D. Fernández, I. Parra, M. Á. Sotelo, L. M. Bergasa, P. R. de Toro, J. Nuevo, and M. Ocaña, "Pedestrian recognition for intelligent transportation systems.," in *ICINCO*, 2005, pp. 292–297.

[11] M. S. A. Muthanna, Y. T. Lyachek, A. M. O. Musaeed, Y. A. H. Esmail, and A. B. Adam, "Smart system of a real-time pedestrian detection for smart city," in *2020 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*, IEEE, 2020, pp. 45–50.

[12] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[13] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 9, pp. 1627–1645, 2009.

[14] A. Ćorović, V. Ilić, S. Durić, M. Marijan, and B. Pavković, "The real-time detection of traffic participants using yolo algorithm," in *2018 26th Telecommunications Forum (TELFOR)*, IEEE, 2018, pp. 1–4.

[15] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv*, 2018.

[16] Google. (2020). "What is google colabrotary?" [Online]. Available: `https://colab.research.google.com/notebooks/intro.ipynb` (visited on 12/06/2020).