A Comparative Study of Object Detection Models for Real Time Application in Surveillance Systems

by

Saimun Alam
17301060
Mahim Uddin Ahmed
17301061
Mehedi Hasan
17301046
Md. Morshedul Islam
17101052
Shahed Mehrab Arnob
17101134

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science and Engineering

Department of Computer Science and Engineering
Brac University
January 2022

Declaration

It is hereby declared that

1. The thesis submitted is our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Saimun Alam 17301060	Mahim Uddin Ahmed 17301061
Mehedi Hasan 17301046	Md. Morshedul Islam 17101052

Shahed Mehrab Arnob
17101134

Approval

The thesis/project titled "A Comparative Study of Object Detection Models for Real Time Application in Surveillance Systems" submitted by

1. Saimun Alam (17301060)
2. Mahim Uddin Ahmed (17301061)
3. Mehedi Hasan (17301046)
4. Md. Morshedul Islam (17101052)
5. Shahed Mehrab Arnob (17101134)

Of Fall, 2021 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on January 16, 2022.

Examining Committee:

Supervisor:
(Member)

Iqbal

Dr. Muhammad Iqbal Hossain
Assistant Professor
Department of Computer Science and Engineering
Brac University

Co-Supervisor:
(Member)

Mehnaz Seraj
Lecturer
Department of Computer Science and Engineering
Brac University

Abstract

In this paper, we attempted to give an overview based on thorough research and testing of the latest object detection methods with an aim to help developers to build a Real Time Responsive CCTV Camera Model. As we welcome the 5G network worldwide, the coming future will surely be heavily dependent on smart machines and internet-based technologies. Therefore, we can assume that our daily life security will also be managed by smart devices. In this research work, our aim is to do a thorough research on the latest models so that one can be chosen to implement and minimize the existing security system into a one device depended security system. The device we often use for surveillance and security purpose is CCTV camera. However, most of the cameras are not connected to the internet also they are not responsive. Which means, the outputs from the cameras cannot be used for further analysis by machines and can only be saved for manual check by humans. Our research will help to develop such a system that will make the camera act like more of a security guard itself rather than a video recording device only. As we need to find out the best suited detection method we will check the accuracy, implementation process, power usage, GPU and CPU usage and then choose between previously invented methods such as HOG (Histogram of Oriented Gradients), Viola Jones Detector or the latest inventions such as R-CNN, SSD YOLO. Finally, this research will help the security device developers to choose the best algorithm and build cost efficient systems. Also, the future works of the research will help to create alert for abnormal presence of unknowns under surveillance automatically. Overall, we can say that our research will help to build more affordable, efficient and digitally secured home, offices, schools or any other buildings and even roads and highways in coming days.

Keywords: Surveillance and Security Systems; Object Detection; YOLO; SSD; Modern Home Security; CCTV Camera; Computer Vision; Image Processing; Fire, Weapon and Threat Detection.

Dedication

Dedicated to all the scientists and researchers in the world who are working day and night to develop better systems and make life easier for all. We would also like to dedicate this work to our two amazing advisors Ms. Mehnaz Seraj and Dr. Muhammad Iqbal Hossain Sir. Thank you for your enormous support and motivation.

Acknowledgement

First of all, we are thankful to Almighty Allah for His countless blessings for which we have successfully passed through the obstacles in this pandemic and completed our thesis. Secondly, we would like to express our gratitude to our teacher and advisor Dr. Muhammad Iqbal Hossain Sir and co-advisor Mehnaz Seraj Ma'am for their kind support and advice during this research period. Without their help and critical analyzing, we could not have come this far. They both always helped us whenever we needed.
And finally thanks to our parents. With their kind support, unconditional love and prayers we are now on the verge of our graduation.

Table of Contents

List of Figures

List of Tables

Nomenclature

This is the list of symbols & abbreviation used in this paper

API Application Programming Interface

CFG Configuration Files

CNN Convolutional Neural Network

HOG Histogram Oriented Gradients

IOU Intersection Over Union

mAP mean Average Precision

$R-CFN$ Region-based Fully Convolutional Network

$R-CNN$ Region-based Convolutional Neural Network

SSD Single Shot Multi Box Detector

$YOLO$ You Only Look Once

$YOLOR$ You Only Learn One Representation

Chapter 1

Introduction

1.1 Motivation

All over the world we will soon be using 5G internet and this will help to connect greater number of devices at a time with lower latency and will also help to build more concrete and faster networks. At present, most of the existing methods vastly depend on various detection devices such as motion detector, metal detector, fire and smoke detector etc. where in a system, failure of one device may lead the entire system into failure. From this idea, we have decided to research on building a stronger, faster and more efficient digital security system by using smart security camera only, which will reduce the dependency on so many different and interconnected sensors and devices.
We know that, from home to shopping malls, schools, offices even in parks most commonly used surveillance device is CCTV camera. However, almost all of them are basically connected to a local storage device and not connected to the internet and not responsive. Our study will focus on finding the best available method which can help to develop the system in such a way that it will be able to detect and identify threatening weapons like guns, knives etc. and disastrous situation like fire hazard. In future works through further modeling and data training a system can be developed which will be able to identify intruders and criminals, accidents like vehicle crash, riots and respond according to the situation. Furthermore, in case the updated version of the system cannot find a match of a newly recorded sample, it will keep the data temporarily and check for consecutive appearances. This will alert the system owner to take necessary steps in case of any abnormal behavior. Our research is going to give an in-depth idea to those who are willing to work with surveillance systems and build devices. From this work, they will be able to choose the best suited detection methods according to their processing unit's capability and system's complexity.

1.2 Aims and Objectives

Our research will work on providing data and validity to develop a security system in future. We are aiming to provide the following features:
i) Reviewing the existing models in field of object detection and filtering out the older ones for real time applications.
ii) Testing the newer models to establish comparative study through training with custom dataset.
iii) Featuring the best suited object detection method for systems with high to low specifications.
iv) Help to detect any weapon like guns and sharp objects like knives, scissors etc. and threats like fire hazards, blasts etc. by using a trained model and custom dataset.
v) In future, similar method could be used to identify criminals in real time by using law enforcement agencies' database and send quick alert to the police or responsible authority.
vi) Future research results will also be able to detect any accidents vehicle crash, riot, fighting, kidnapping etc.

Chapter 2

Problem Statement

As the development of home/building security devices are becoming more available people are trying to get rid of human security guards and set up many sensor-based security systems at their house, office and at other places. Such devices sometimes work on an individual role and sometimes all together acts as a security system. Our research finds out that most of these devices are mainly sensor based and they cover a small amount of area. This raises an important security issue because the intruders can avoid the detectors range. On the other hand, if a primary security device fails to act the entire system becomes more vulnerable and weakens the next level devices' ability because the secondary devices more often rely on the output from the primary level. Therefore, we need to find out such a device that covers a larger area and can hardly be avoided. We can introduce security cameras in this step. Yet, our system is not totally secured, because most of the security cameras work simply as a video recording device. Hence, the intruder can steal/or cause any damage without getting caught on the spot. Furthermore, by using a mask can intruder can avoid getting caught afterwards as well. Therefore, we need to make our cameras responsive and act in real time.

Another issue comes up when we have to deal with usual CCTV cameras. The video the record are of very poor quality that creates confusion in detecting the actual culprit. On that account, we need high quality videos. This solution again, leads to another important issue which is analyzing such high definition videos in real time and responding quickly and accurately. Maintaining high accuracy and high speed will be a core challenge. While working on our goal we found many problems and some of them made us interested to work on them. According to [1] after considerable breakthroughs in the CCTV area, processing hardware, and deep learning models, the problem of real-time object recognition and categorization arose. There has been very little research in this sector before, and most of what has been done so far has been focused on hidden weapon detection. Prior to its usage in weapon

detection, hidden/concealed weapon detection (CWD) was applied in airports for luggage control and other security purposes, and was based on imaging processing techniques such as millimeter-wave and infrared imaging. For the detection and classification task, deep learning models faced the following challenges: The first and most serious issue is the data from which CNN learns its features, which will be used for classification and detection later. However, there was no standard dataset for firearms. Making a new and custom dataset manually for real-time scenarios was a lengthy and time-consuming operation. Also, labeling the intended dataset is a difficult operation because all data must be individually and manually labeled. Because the detection methods were different, a labeled dataset created for one technique could not be used for the other. For the same-labeled dataset, each algorithm requires distinct labeling and pre-processing. In terms of real-time implementation, detection systems require the precise location of the weapon, hence gun blocking or obscurity is a common issue, which can be caused by self, inter-object, or background blocking. Low-level features, discriminative learning, and pictorial structure were employed in conjunction with SVM in HOG's key work. For real-time applications with 14s per image, these methods were too slow to be accepted. Although these classifiers were good in terms of accuracy, the sliding window method's latency was a major drawback, especially for real-time application.
Now, as we have got a clear idea, we can come to a conclusion that, developers need to implement a good detection method in the system. However, the problem with existing devices are they are not capable of data processing since they only offer some basic video recording, rewind and manual review functions. On the other hand, older the modern image processing and detection methods have two types of drawbacks in terms of using them in such a system. Firstly, the older methods like HOG, Viola Jones Detection etc. takes a bit long processing time which is not suitable for a real time detection system. On the contrary, the modern systems like R-CNN, SSD and YOLO, they do provide a faster and more accurate detection result, but need a heavy processing unit to function properly. That is why, we are aiming to test the methods and find the one with best performance that also uses less processing power so that it can be implemented in a such system.

Chapter 3

Background Study

In this work, first of all we needed to learn about Object Detection. The technique of integrating recognition and localization in a photo or video is known as object detection. Object detection has been there for over two decades and plays an important role in Computer Vision objectives. Object recognition is essential in a wide range of applications, including autonomous vehicles, smart robots, and video monitoring.

The Viola-Jones face detection mechanism, released in 2001, set the foundation for Object Detection. The approach uses a simple method for Object Recognition by exploring all areas of the image using template matching of a variety of scales. The next step was to employ a revised Histogram of Oriented Gradients (HOG) as a feature representation for object detection and recognition. These earlier object detectors were known as Machines and then used customized characteristics. When the Deformable Part-based Model (DPM) sensor win the VOC-07, VOC-08, and VOC-09 detection challenges, classical object detectors reached their high point. Many experts used these tournaments to assess the performance of their sensors at the time.

With OverFeat in 2014, Object Recognition managed to enter the Deep Learning domain with a basic convolutional network. To improve identification probability, the scientists employed a variation in terms of the rolling windows technique and a greedy algorithm to collect bounding boxes. To forecast bounding box coordinates, they also had to insert a regression network to the bottom of the system. The ImageNet Large Scale Visual Recognition Challenge 2013 was won as a consequence of their efforts (ILSVRC2013).

Object detection with CNNs was once a two-stage process. It would create a region of interest in the early phase. If the reaction is powerful enough, it will categorize each suggested area in the second stage. Finally, after expanding bounding boxes, deleting repeat detection systems, and re-scoring boxes depending on some other items in the picture, there are post-processing procedures. The well-known R-CNN,

Fast R- CNN, and Faster R-CNN models, where the R stands for "Region-Based," are using this two-stage technique. Every sensor in Object Detection has such a structure. Numerous extracting features systems can be used as the basis.
Single Shot MultiBox Detector (SSD) and You Only Look Once (YOLO) would be the next only one approach for Objection Detection. These methods combine object categorization and localization in a single phase, resulting in a single end-to-end learning model. Because it rephrased the issue as a single equation problem, heading directly from picture frames to bounding boxes and classifiers, YOLO became essential in the area of Object Detection. While separate designs have been demonstrated to be quicker at interpretation than their two-stage competitors, they have proven to be extremely less accurate.
The term "anchor-free" refers to the most modern form of object detector. They have fewer options for leveraging massive amounts of data. for example, CornerNet, CenterNet, Fully Convolutional One-Stage Object Detection, and Bottom-up Object Detection are just a few examples of anchor-free detectors. Here, we will give brief summary of some of the modern models.

3.1 Latest Models

3.1.1 CNN

A convolutional neural network (CNN) is a type of machine learning used in object recognition. CNN takes an image or video frame as input and applies filters to conduct convolutional operations. The weights of a CNN are what these filters are called. Convolutional operations on the image produce feature maps, which are then transmitted to the network's future layers.[8] Image Classification, Object Detection, and Semantic Segmentation are three areas of Machine Learning where CNNs are applied. Image Classification is the process of providing a label to a photograph to categorize the contents of the image. A classification model may, for example, be given a picture of the beach and output the label "sand." There's also a reliability score, which ranges from 0 to 1, which indicates how sure the model is in providing that label. While Image Classification helps classify what's in an image, Object Detection takes this a step further by using bounding boxes to identify things in the image. It is possible to identify not just multiple items, but also several different categories of things. Since unmanned vehicles are becoming a very popular research and development topic now, Object Detection has also become a highly crucial part for this. A self-driving automobile must be able to recognize several objects at the same time and make intelligent decisions on how to drive the vehicle. At last, Semantic Segmentation also called Object Segmentation, extends Image Analysis and

Object Detection by creating a pixel-wise mask for every image identified in addition to bounding boxes, labels, and confidence scores. A pixel-wise mask is just a mask of an item with pixel-level boundaries. Satellite images, accurate decision-making in autonomous vehicles, and medical imaging all benefit from this fine-grained localization. Because of the precision necessary for detecting wildflowers in grasses for this research, Object Detection rather than Semantic Segmentation was used. This is because Semantic Segmentation needs more computations for its result, making it weaker than Object Detection.

3.1.2 R-CNN and Faster R-CNN

The R-CNN model influenced subsequent object detectors by providing as a foundation. R-CNN will use a greedy algorithm to collect ideas, then a feature extraction backbone to extract additional features, and finally a Support Vector Machine (SVM) to classify. By merging feature extraction techniques in a single CNN, Fast R-CNN increased the speed of R-CNN. Faster R-CNN developed a more complex technique, utilizing a unique Region Proposal Network (RPN) that prevented time-consuming greedy algorithm, resulting in increased speed. Despite these advancements, it was still hampered by the RPN's high computational cost. Its two steps provide high accuracy, but at the cost of detecting speed, as more calculation is needed. [9]

3.1.3 YOLO

The YOLO neural network is a single neural network that generates bounding boxes and the model is built in a single assessment. The Deep Learning period, it's the first object detection. Batch normalization, adjustable frequency input, anchor boxes, perfect features, and an improved Dark-net backbone were all implemented in YOLOv2. On the Common Objects in Context (COCO) dataset, improvements contributed to the same Mean Average Precision (MAP) scores as Single-Shot Detector (SSD), but 3 times faster. YOLOv5 was designed to improve accessibility, learning speed, inference speed, and deployment ease. This version of YOLO is written in PyTorch, as compared to the earlier versions, which were developed on the Darknet framework. [40]

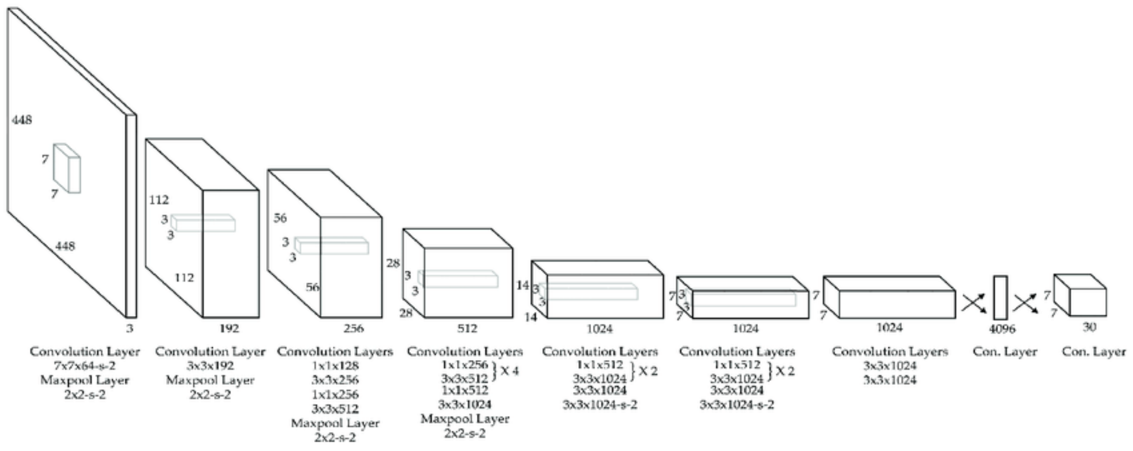


Figure 3.1: YOLO Architecture
[26]

3.1.4 SSD

In the Deep Learning era, the Single Shot MultiBox Detector (SSD) was the second one-stage image detector to arise. Multi-resolution and multi-reference techniques were developed by SSD, which considerably improved detection capability above YOLO now at moment. In 2016, SSD debuted with a 74.3 percent MAP on the VOC2007 dataset and a genuine learning speed of 59 frame rates on an NVIDIA Titan X.

MobileNetV2, a high dimensionality collector, is the foundation of SSD. Convolution layers begin to decrease at the top of the SSD detector, allowing for predictions. It is in contrast to YOLO, which only shows a particular scale feature map. SSD, like YOLOv3 and Faster R-CNN, predicts bounding box alignments and per-class ratings utilizing anchor boxes. SSD differs from another two sensors because it utilizes these basic anchor boxes to a variety of extracted features with various resolutions in order to make their sensor scale-invariant.

Smaller objects are a sector where SSD's struggle in terms of capability. Small objects have no reflection on the smallest fully connected layers at the bottom of the detector, therefore this is the situation. SSD is still an extremely appealing image detector despite this shortcoming due to its balance of reliability and quickness. It will provide state-of-the-art accuracy in real-time prediction, making it an easy option for testing plant recognition in this research. [38]

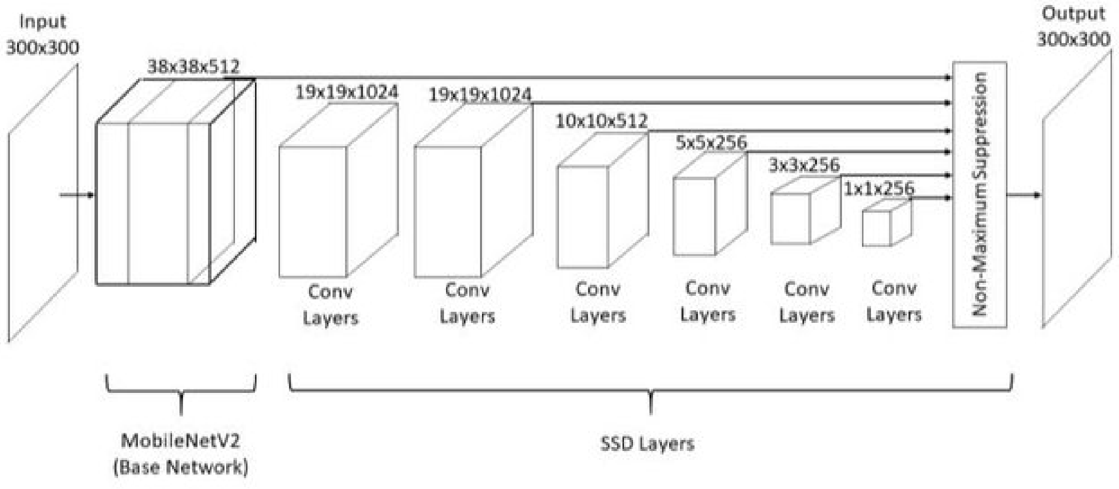
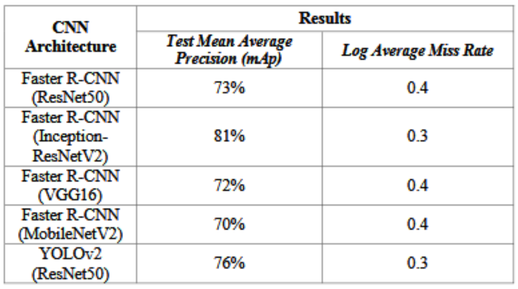


Figure 3.2: SSD Architecture
[18]

3.2 Literature Review

While searching for the existing works on the topic, we have mainly focused on detection through image processing, real time detection, quick matching and detection from an angular view. We also have worked on papers that mentioned other smart security systems using IoT devices. Some of the notable papers published in IEEE Access and other noteworthy places are briefly mentioned below.

According to[29] an Automated Gun Detection system was used by implementing the Faster RCNN model. VGG16, ResNet50, InceptionResNetV2 and MobileNetV2 have been used individually as feature extractors in Faster R-CNN. All of the proposed architectures were tested, trained and compared with YOLOv2 using the same dataset and they got better results using Inception-ResNetV2 of Faster R-CNN. [2] It's seen that for mobile applications MobileNetV2 was useful meanwhile Inception-ResNetV2 gave 82% of mAp among all the models. Their drawback is that each model was owning at their own field such as VGG16 was much faster among the extractors but no match for YOLOv2. Getting inspired from this we tried Faster R-CNN and SSD-MobileNetV2 and many versions of YOLO and gave comparative analysis on our system. Out of them YOLOv4 gave the best results in all terms and fields. Next, [4] the paper reviewed here, they proposed the idea of using multiple cameras for real time tracking in the security system. They divided their approaches into regional based and boundary based. [24] In the case of a regional based approach they used a background subtraction method for detecting objects considering its drawbacks. For solving the issues they created algorithms for generating frames, object detection and post processing. [36] They have compared unique methods for object detection and found the highest accuracy rate. They mainly focused on



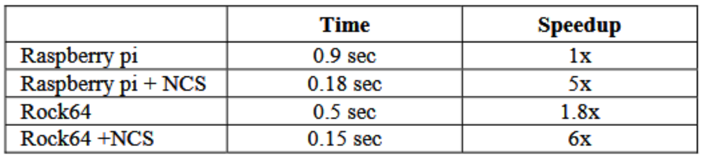
CNN Architecture	Results	
	Test Mean Average Precision (mAp)	*Log Average Miss Rate*
Faster R-CNN (ResNet50)	73%	0.4
Faster R-CNN (Inception-ResNetV2)	81%	0.3
Faster R-CNN (VGG16)	72%	0.4
Faster R-CNN (MobileNetV2)	70%	0.4
YOLOv2 (ResNet50)	76%	0.3

Figure 3.3: Results of CNN and YOLOv2

rigid objects and multi camera systems and have a future plan to improve much faster in case of detecting objects.We tried to detect objects with multiple cameras but were not successful with all algorithms. [22] Only YOLOv4 was able to be run with multiple cameras and we got significant results but there were no algorithms to compare with so we did not include this in our paper.

According to their paper [16] focused on automatic event detection by maintaining good image quality and field of view. They used reduced- reference features and analyzed the events whenever image quality drops or any significant changes are found. An interesting feature known as Kalman filter is used to smooth the images, thus reducing noise and improving false alarm rates. [21] Though they did not experiment their method in real time, they used recorded videos and proved that their method outran all existing methods. A linear cumulative strategy of two growth factors along with online Kalman filtering was derived in their paper. According to their [39] experimentations, they achieved high precision in detecting objects by considering the computational complexities found in their proposed algorithms. We understood that they worked on improving the datas in their system before training and detection. So we tried a new approach to reduce the hassle for picture cropping for annotations and introduced YOLOv4 deep sort to count persons while detecting [5] .

According to [30] in order to achieve the highest accuracy or recognition of an image, the system requires powerful calculations. For this they proposed a low cost CNN model which uses Neural Compute Stick or commonly known as NCS as an alternative for existing GPUs. [3] This is a hardware accelerator backed up with Rock64 for providing high speed calculation at a low cost. They used the SSD algorithm for their model and found that it is faster than other systems achieving a time of 0.15 sec for each image. [35] Compared to some powerful models such as CNN model and other algorithms that use ROI such as faster RCNN, their proposed method promised to give outstanding results on low end devices. In their paper they compared SSD with other algorithms such as TinyYOLO and other CNNs and found that SSD gave better results on average in all aspects. [34] To analyze the time complexities NCS is used both in Raspberry pi 3 and Rock64 and found a noticeable boost in performance in Rock64. They also proved this architecture was suitable for this kind of low end experiments and is fast enough to detect even in motion.We came to find that they focused on low end devices, so we also tried to analyze all the object detection algorithms and found out what suited this kind of device. [28] Relating to the paper we also tried the SSD algorithm, R-CNN and TinyYolo and found that SSD was actually faster but R-CNN gave much more significant performance compared to their paper, so this also falls under our contribution.



	Time	Speedup
Raspberry pi	0.9 sec	1x
Raspberry pi + NCS	0.18 sec	5x
Rock64	0.5 sec	1.8x
Rock64 +NCS	0.15 sec	6x

Figure 3.4: Results from Raspberry Pi and Rock64

According to this paper [32] demonstrates video surveillance with enhanced accuracy and minimal computational complexity. They have worked on face, recognition, detection and localization. [20] Their detection is possible from both recorded and real time videos. Their paper's [7] result is on the basis of comparison of the data in their database and the facial data detected in their system. Security alarm, signal generation and alerting security can be retrieved depending on the matches found in their system. [13] Compared to most of the systems, they proposed that their paper gives more accuracy, low cost and has better performance. For classification, they used a multi-layer neural network and included special facial features. The extracted functions are decided and provided as a sample vector to the neural network. The gaining knowledge of a set of rules acknowledges people's faces via means of gaining knowledge of the approximation of facial functions, no matter specific facial movements. [19] The function matrix modifications rely on the face motion. The use of 4 specific video sequences offers sufficient statistics to educate the classifier to become aware of someone in a crowd. The proposed version became examined under extraordinarily numerous conditions, and it achieved efficaciously and accurately. [6]

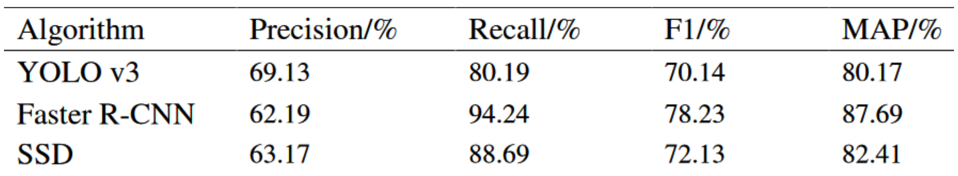
This study [27] is based on face recognition, detection and object detection and recognition. Their desired results are found through training neural network other models with a sufficient amount of data. First, neural networks were introduced into this study and then trained. [23] The creation of neural networks is realized on both

CPU and GPU and uses multiple GPUs with an algorithm called the backpropagation matrix format which is built with the CUDA kernel and cuBLAS library. Face application Detection was implemented using the pre-trained models Facenet and Deep Convolutional neural network. Python OpenCV library has been applied with a deep learning approach to implement face recognition, image registration, object recognition and YOLO Deep learning is applied in two simple steps. [14] The first step is to detect the presence of a face using face detection. However, the image or video stream is unidentified. The extraction process is the next step. 8D feature vector quantifies each face of an image or video stream. These vectors are also referred to as embedding. They trained during label encoding after embedding the face and loading the SVM model for face recognition [15]. Deep learning object recognition (YOLO) is trained against the loss feature, which is directly related to cognitive performance, and the entire model is trained simultaneously. To recognize an object in an image or in real life, current recognition systems obtain a classifier for that object and evaluate it at various scales and locations in the test image. To carry out. The YOLO design is end-to-end training, allowing for real-time speed while maintaining high accuracy rate. [33] Popular neural network models will undoubtedly form the foundation for such successful deep learning in the future. Deep learning techniques are explained by models such as CNN, Deep CNN, and other well-known training algorithms, as well as various encoders and decoders with noise or noise reduction, and deep learning produces promising results in many of its applications. And, while it is not shown, it contains the success and broader aspects associated with the research discussed, which explains why it is also promising potential.

The goal of this paper [37] is to improve traditional security systems. Security systems built on the IoT platform can interact with devices in real time. A camera, voice sensor , microphone, motion activity sensor, and LTE / Wi-Fi module are all connected to the processor at the system's heart. This entire economic system makes use of IoT in real time, allowing mobile devices and computers to remotely track activity occurring where IoT devices are located and save all activity to their cloud storage account. [12] This IoT-powered smart locker allows for 24-hour monitoring, alerts, and emergency notifications from anywhere in the world via mobile apps via a cloud connection. The main aim of this project is to monitor the locker by sending out instant video alerts whenever activity is detected by the locker. The owner's information can be quickly shared with police officers. Furthermore, user information such as locker opening and closing times, as well as store closing times, is recorded. [10] The number of times it has been opened and closed, as well as the name of the person who has unlocked it. The system is made up of a camera with a microphone, push buttons , an LTE / Wi-Fi module, a sensor, and a processor. Mobile devices and computers can remotely detect activity taking place where lockers are located throughout this economic system that uses IoT in real time, and activity data are stored in their own cloud-storage accounts.[17] It will be preserved. The document's challenge is that the data sent and received by existing intelligent systems is vulnerable to forgery and hacking. We need to expand this block chain-based system to solve this problem. This is especially true if these systems determine and respond to specific events in their surroundings based on data sent by sensors.

According to this [11] paper, for object detection they mainly focused on improving the SSD algorithm and also compared some models and functions such as Faster R-CNN and Loss function. They used a different method for classification and introduced a multilayer convolutional neural network. [1] Their limitations were visible as they discussed problems in detection for different factors such as slow processing speed, inability to detect new object classes and low results for smaller objects. In our paper we tried to analyze this situation and found that YOLOv4 solves all these problems and the SSD was also good in our system but overall all versions of YOLO surpassed with an average score.

Furthermore, [25] Comparison of YOLOv3, Faster R-CNN and SSD was done for pill identification. They highlighted YOLOv3 as the fastest among their proposed methods. According to their results we compared the results with our system and proved that YOLOv4 works much better in every aspects. [31] Comparative analysis also given for ResNet, SSD-MobileNetV2 and CNN algorithms on a mass and diverse dataset for face mask detection. Here, SSD-MobileNetV2 was detected much faster but in our paper we compared with all versions of YOLO and found that YOLOv4 would be much better for us and their case too.



Algorithm	Precision/%	Recall/%	F1/%	MAP/%
YOLO v3	69.13	80.19	70.14	80.17
Faster R-CNN	62.19	94.24	78.23	87.69
SSD	63.17	88.69	72.13	82.41

Figure 3.5: Results of YOLOv3, Faster RCNN, SSD

Chapter 4

Methodology

After filtering the older models by reviewing some of the most popular methods by researching related works, papers and initial testing in previous parts of our thesis, we chose to work with SSD model and YOLO since they offer such features which are highly relevant to our work. We have done thorough testing of both models. Moreover, we started working with YOLOv4 previously, however, its latest version YOLOv5 offers some more sub-versions and we worked with most of them, such as, YOLOv5m, YOLOv5n, YOLOv5L, YOLOv5s etc.

4.1 Data Collection

In this phase of research, we have worked on detecting objects that are threatening to humans such as guns, knives, scissors, fire etc. We collected pictures from internet sources and later labeled and annotated them. Also, we used real life images of available tools and videos by using our webcams. The weight file is a trained dataset that is specifically developed for YOLO.

4.1.1 Dataset

We have created our own dataset to train and test our object detection models. Here we collected data from various sources for our dataset. Our dataset consists of four objects. knife, gun, fire, and person. We first took photos of ourselves holding a knife and a gun. Then for fire, we took photos of it from different angles and positions in indoor situations. After that, we collected photos from Google and different open-source image collection websites like OpenImage dataset.
For the dataset, we have collected 2000 images, 500 for each object. We made sure that we collected the pictures from all possible angles and that all the images were of the same dimensions.

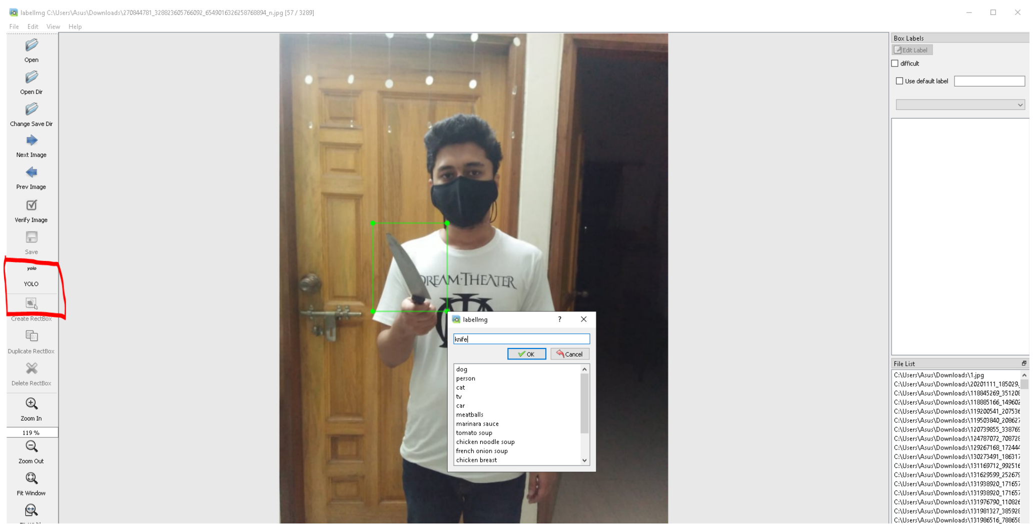


Figure 4.1: Data Labeling and Annotation

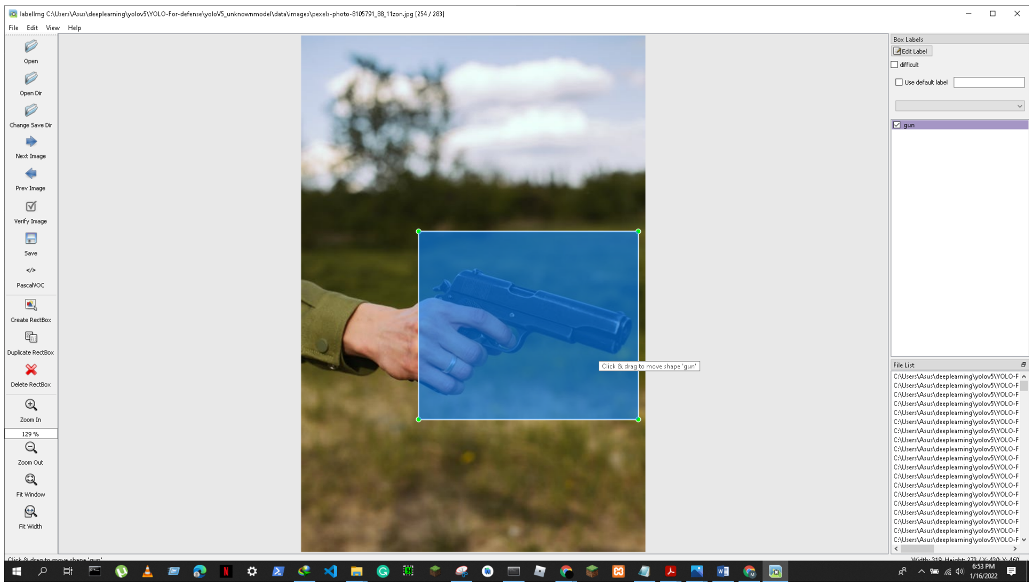


Figure 4.2: Data Labeling and Annotation

After labelling the data we converted the data to desired formats. For example for tensorflow object detection api we converted the data to TFRecord format. We tried to create a dataset to automatically detect threats in an indoor environment, like someone entering the area with a knife or a gun and if there is a fire in the house. We are going to detect these threats by object detection. Here are some samples of our custom dataset.

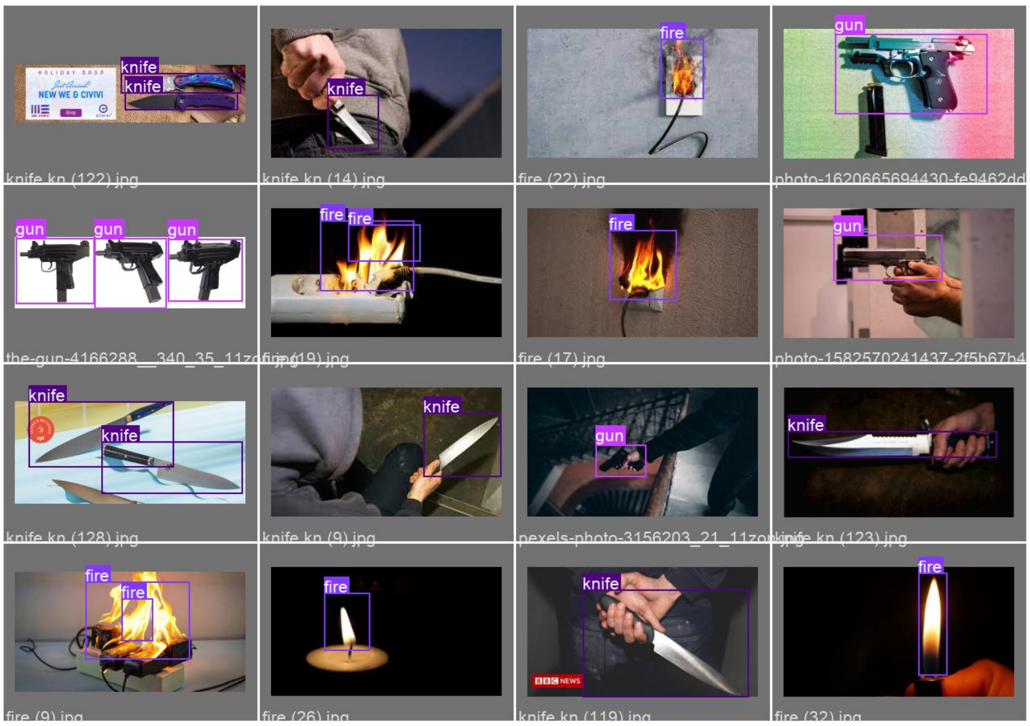


Figure 4.3: Custom Dataset Sample

4.2 Data Processing

To ensure better accuracy and ignore unnecessary objects we have created a feature that can detect objects and save them as cropped images from input video or image. Later these cropped images can be used for future reference and analyzing. Every image is analyzed by using some annotation values. And similar objects usually carry similar values as they are structurally same mostly. For exact same objects, these values mostly match fully unless lighting conditions are not disturbed badly. Similar objects with slight differences in shape can also be detected but accuracy may fall. For this reason, a pre-processed large dataset can be used for faster and more accurate result.

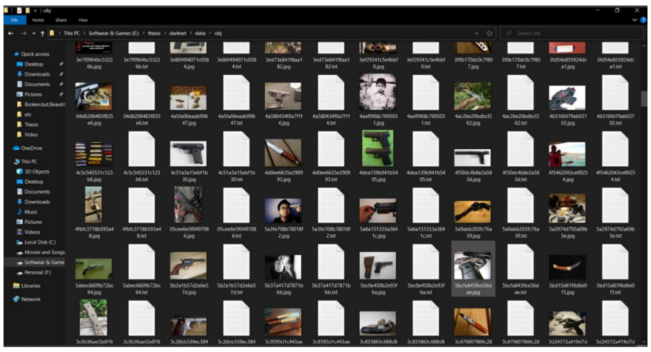


Figure 4.4: Cropped Images of Detected Objects

4.3 Training and Findings

For training the object detection models we have used varieties of software frameworks. We have used Darknet, PyTorch and tensorflow object detection API. After annotating the images of the dataset we started the training process. We prepared the data for Object Detection API. Then we converted the data to desired format as mentioned in the previous section. After that we configured our object detection model for training . We also used the pre-trained model to train our custom object detection.

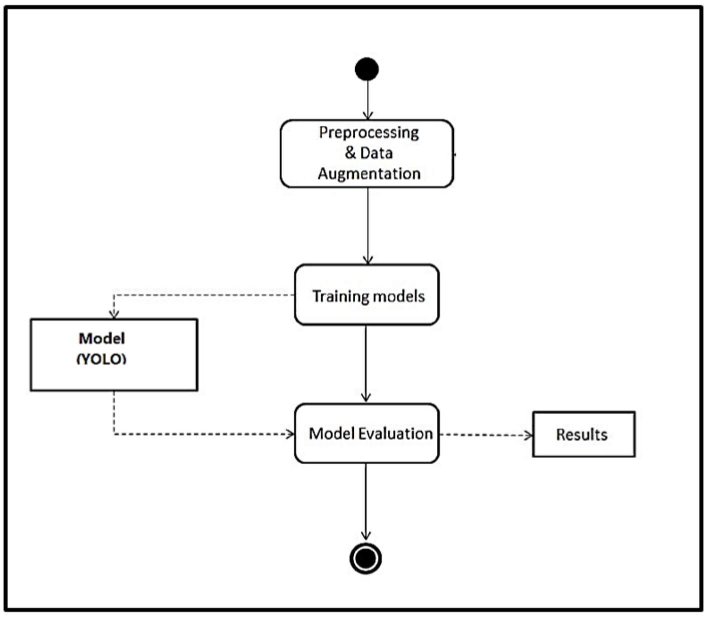


Figure 4.5: Training YOLO

4.3.1 Training and Findings of YOLOv4

Darknet is YOLO's standard training platform, and it was created with the intention of training YOLO object detectors. On the same MS COCO dataset, YOLOv4 was chosen over YOLOv3 because it provides many new features that enhance accuracy and speed. Mosaic was one of the features that were used.

Yolov4 is so far the finest object detector right now. It has an accuracy of close to 100%. Our prime object is to detect knives, guns, and fire and so far, it provides the best output right now. To train our datasets first we have collected our images from google open images v6. For each object, we have collected around 500 images.

First After labeling and annotating, For Yolo, we have made a cfg file to train our datasets. In our cfg file, it contains the information of how many batches we are going to use, the number of channels, max batches, and steps. For Each object, we are going to use 2500 max batches, in our case total object is 4 so we are using around 10000 batches. And for each batch, there will be 80% to 90% steps. So, for 10000, our steps are 8000 9000. Our total iteration will be 10000 and avg loss should be less than 2. And here our loss is 0.25 which is very low.

We have trained our datasets with an iteration of 10000 as we have a total of 4 objects. So each object contains 2500 iterations. When we first started our training the avg loss was higher. So we need to perform multiple iterations to get minimum loss. Minimum loss provides best results. Here after 10000 iterations our loss is now 0.25 which is very minimum.

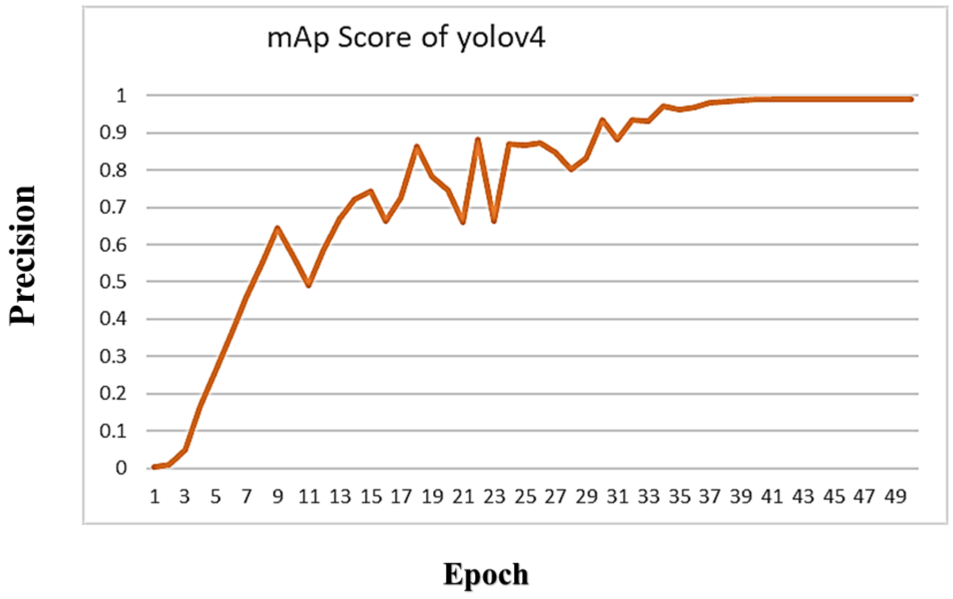


Figure 4.6: YOLOv4 MAP

We set the epoch 1 to 49 to get the best mAP score. And our mAP score is around 0.98 which is the best. THe maximum mAP score is 1 and our is close to 1 which is 0.98. As a result our detections perform really well.

We have tested our datasets in real-time as well as in the videos too. And on

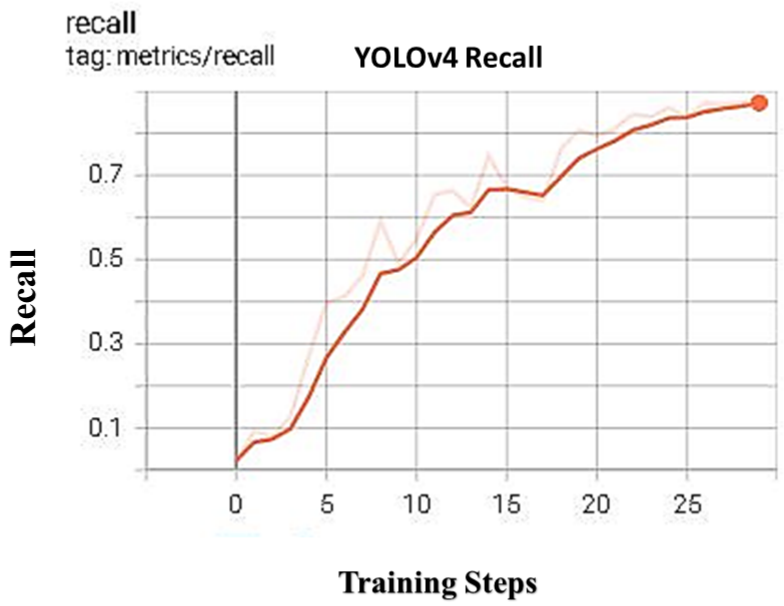
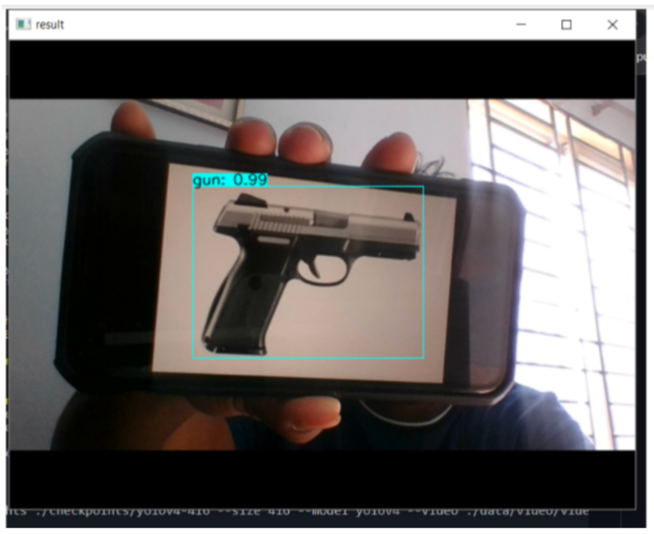


Figure 4.7: YOLOv4 Recall

an average gun detections results are around 99% which is maximum. Another important fact is In real time it just took 53ms to detect the objects. Which is a very big advantage. It works really very well in every aspect. And we got average fps 20-25 in our gtx 1050ti gpu. We tested in google colab too and it gives us a fps of 8-13. And in only cpu it is a bit laggy as it needs cuda core to perform well. So here we got around 2-6 fps.



```
ts ./checkpoints/yolov4-416 --size 416 --model yolov4 --video ./data/video/vide
```

Figure 4.8: YOLOv4 Result in Real Time

So if we have a better gpu with cuda core it will perform its best. Most of the time we will get maximum fps as well as almost 99% of accuracy. The more we iterate our training the more we get the best results on it. We have found a few drawbacks of this method, it needs higher gpu to get maximum fps and accuracy and also It takes a lot of time to train as each iteration takes almost 2 hours depending on CUDA core and CPU cores.

4.3.2 Training and Findings of YOLOv5

In the same way we have also training our next yolo methods which is yolov5s. Which is another latest version of yolo. And it also performs really well. In the same way, first we have collected our data from Open Image Datasets v6. Then we label and annotate it. After that we perform iterations over it. We have used the same yolov4 images to our yolov5s model. So that we can compare both and get an idea of which one is actually best and which one works much faster. The mAP score of YOLOv5s is not as much as YOLOv4. Also it is not consistent as YOLOv4. So after 49 epochs we get the mAp score of YOLOv5s is 72.

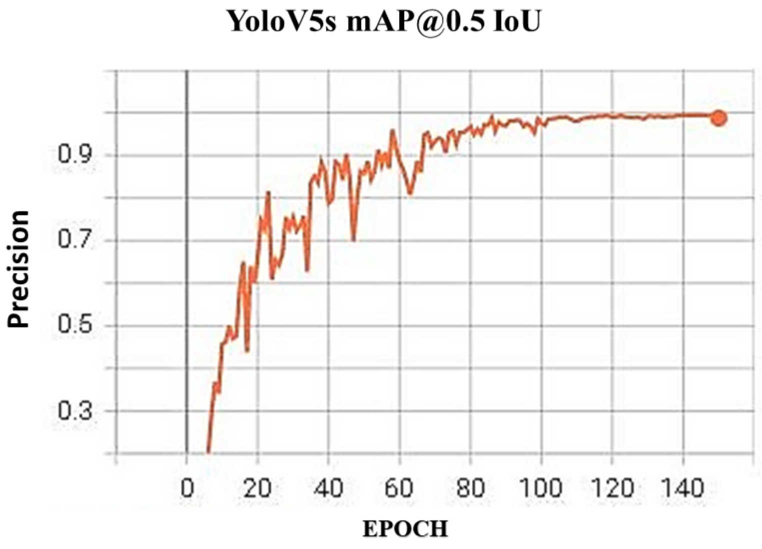
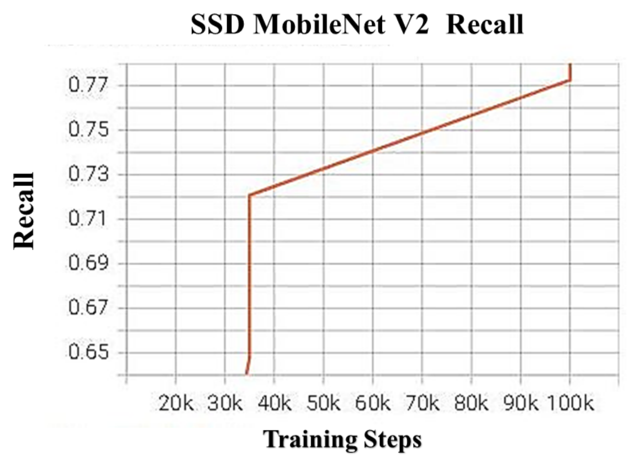


Figure 4.9: YOLOv5 MAP

4.3.3 Training and Findings of SSD MobileNet V2

Here for training our SSD model we choose MobileNet as the backbone. We choose to train SSD MobileNet v2 among all the SSD variations as from the related work we have got to know that this version can provide us with the best accuracy and inference speed of all the models using the SSD algorithm.[14]. Tensorflow object detection API, SSD mobilenetv2 feature extractor,TensorFlow directory were used for the experimental setup.Anaconda virtual environment was also used to set up our environment and Tensorflow-Gpu was enabled to make our training process faster. Tensorflow with Keras backend was used for training. Each epoch was saved a weight file during the entire training process so that after training we can choose the best weight which gave us the best performance metrics. Here we trained the model in 50 epochs.



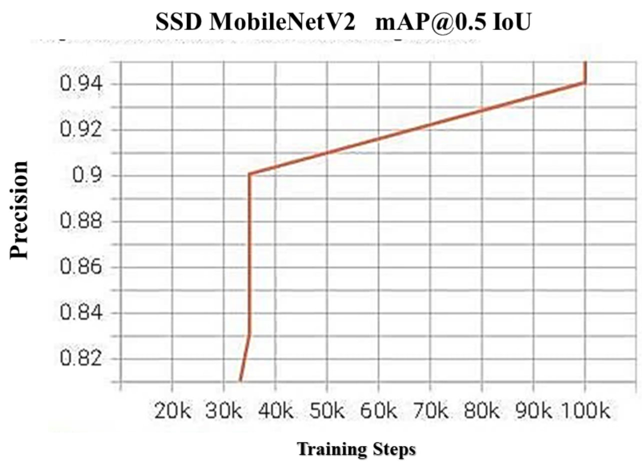


Figure 4.10: SSD Training Results

Chapter 5

Result and Findings

5.1 Evaluation Metrics

For the evaluation of object detection models, we choose mAP as evaluation Matic. Popular object detection algorithms like YOLO, MobilenNet SSD, FRCNN evaluate their model using mAP for publishing their work. Object detection is a complicated task. Therefore we had to use precision, recall along with IOU. Precision measures the accuracy of a model's predictions and then calculates how many of the predictions of the models are actually correct. For this research, Precision is the measure of successful threat detection like fire, knife, or gun out of all detection the models make. Here TP means that the object detector made a prediction of a threat and it is correct. And FP means that the object detector model made detection and it is incorrect.

$$\text{Precision } = \frac{TP}{TP + FP}$$

$TP =$ True Positives (Predicted as positive as was correct)
$FP =$ False Positives (Predicted as positive but was incorrect)
Recall measures the accurate detections taking into account of threats that couldn't be detected. A high recall means a higher rate of success of detecting objects with low failure to detect target objects.

$$\text{Recall } = \frac{TP}{TP + FN}$$

IoU determines the accuracy of object detection. Here for our research, we have used IoU threshold 0.5, which means anything below it is an FN and above it counts as a TP. Its calculation is basically an overlap ratio between the predicted bounding box and the ground truth table.

$$IoU = \frac{BB_{\text{overlap}}}{BB_{\text{union}}}$$

In this research, we are using mean average precision. As we are using mAP at 0.5, that means we are using Average precision at a 0.5 threshold for all the classes. For our evaluation mAP is the most important metric. Because it is better to detect a fire which not there to miss an existing fire. After evaluating our models with all these scores we can come to a conclusion that which model performs better.

5.2 Result and Analysis

We have covered Dataset preparing, processing, training process of major models and evaluation process in the previous sections. Now we are going to present our results from training and inference evaluation. Here we have evaluated the model on basis of accuracy and speed. After training the custom models on our dataset, the evaluation metrics were measured. mAP was recorded form checkpoint which had highest values. Then we ran the models in our local devices to see object detection performances and which model gives us the better accuracy and FPS.

We have trained a total of 13 models to detect objects. After that we evaluated those models and then we have tested them on our webcam for getting real-time performance. The highest mAP scores were recorded for each model and in our workstation, we have calculated the FPS. The information about our findings given in the table. The name of the model with their resolution that we selected for our training, training platform, model size mAP at 0.5 IOU and recall score, and FPS are given in the Metrics Data table below.

Model	*Res*	*Platform*	*Size(MB)*	*mAP*	*Recall*	*FPS*
YOLOv4	416	Darknet	255	0.97	0.91	31
YOLOv4-Custom	416	Darknet	121	0.95	0.82	24
YOLOv4-deepsort	416	Darknet	178	0.96	0.82	14
YOLOv4-tflite	320	Darknet	25.3	0.84	N/A	34
YOLOv4-tiny	320	Darknet	17.6	0.82	0.84	40
YOLOv4-tiny	416	Darknet	24.5	0.86	0.79	39
YOLOV5s	320	pytorch	9	0.88	0.915	42
YOLOV5s	416	pytorch	15	0.91	0.915	40
YOLOV5m	416	pytorch	150	0.88	0.86	15
YOLOV5l	640	pytorch	315	0.91	0.78	6
SSD-MobileNetV2	320	TLT3	20	0.78	0.73	29
SSD-MobileNetV2	640	TLT3	36	0.91	0.78	27
FRCNN	640	TLT3	36	0.82	N/A	3

Table 5.1: Metrics Data

In our Training, we have used a total of 3 resolutions of models. These resolutions were picked based on the prevalence of similar resolutions to detecting objects. Each of the resolutions works differently and provides different fps and mAP scores. First we have trained YOLOv4 in the resolutions of 414x416 and the platform is Darknet. The mAP score for YOLOv4 was 0.97 which is close to 1. Also response time is 55ms which is very fast. And the Recall rate is 0.91 which is also close to 1. It provides the finest accuracy to detect objects. We get 97-99% accuracy of detecting guns and knives. Also we get a maximum fps of 37 with 1050ti GPU. As it needs high computational power. So if we use a more powerful GPU with a high cuda core it will give us more fps with greater response time.

We have also trained some of the other versions of YOLOv4 such as YOLOv4-Custom, YOLOv4-deepsort,YOLOv4-tflite, YOLOv4-tiny. Those versions also perform really well. YOLOv4 custom we have trained in the resolutions of 416x416 and we get a mAP score of 0.95. Response time of YOLOv4 Custom is 63 which is higher than the YOLOv4 and accuracy also less than YOLOv4. We get roughly accuracy of 62-75%. Then we have trained the next model which is YOLOv4-deepsort. Deepsort also has some unique features. YOLOv4-deepsort exactly works like YOLOv4 and also it will count how many times the object detected. So if we set our CCTV camera in front of our house and applied the YOLOv4-deepsort method. We can see the total number of times the people have come to my house. After the YOLOv4 deepsort we have trained another model which is YOLOv4-tflite, this is one of the best models with very low size. It has an accuracy of around 88-93%. And we got a mAP score of 0.84. Though it is not as accurate as YOLOv4, it still works very

fine. It has response time avg 0.76 and recall rate is 0.83.
In YOLOv4-tinny we have trained it in the resolutions of 320x320 and get the mAP scores of 0.82. It is very small in size, only 17.6 mb. With this size it still works very fine. No need for extra computational power like YOLOv4. It doesn't require much Cudacuda core like YOLOv4. Tiny response time is 68 which is also very good in general. And we get average accuracy around 85%-89%. Another tiny model with the resolutions of 416 has the similar features of tiny 320. But it has a better mAP score of 0.86. And accuracy of 87-93% on average.
All the YOLOv5 models were trained on the PyTorch platform. The mAP score of "YOLOV5s" with 320 resolution gives us an mAP score of 0.915 and recall is 0.88 and after interfacing, it gives us 45 fps in real-time. The mAP score of "YOLOV5s" with 423 resolution gives us an mAP score of 0.915 and recall is 0.88 and after interfacing, it gives us 45 fps in real-time. Then the mAP score of "YOLOV5m" with 416 resolution gives us an mAP score of 0.915 and recall is 0.88 and after interfacing, it gives us 15 fps in real-time. The mAP score of "YOLOV5s" with 320 resolution gives us an mAP score of 0.88 and recall is 0.86 and after interfacing, it gives us 15 fps in real-time. Then the YOLOv5l 640 resolution gave us a high-accuracy 92% but it had low 9 fps, which is not feasible for real-time detection. Then we used another single shot detector, SSD MobileNetV2. we have trained this on 2 resolutions 320*320 and 640*640. In the 320 version we got a mAP score of 0.78 and a recall value of 0.73 And In 320 version we got mAP score of 0.91 and a recall value of 0.78. In running the detection we saw that SSD MobileNetv2 resolution 320 is very faster in detecting objects and gives us a good fps. Then lastly we have also tested the Faster RCNN model on our dataset here we got 0.82 mAP and 0.61 recall but it gave us a very low FPS in real-time detection.

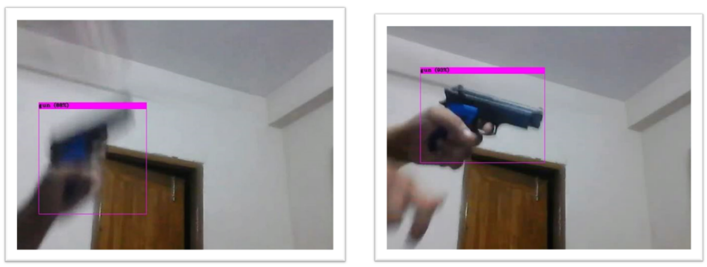


Figure 5.1: Gun Detected

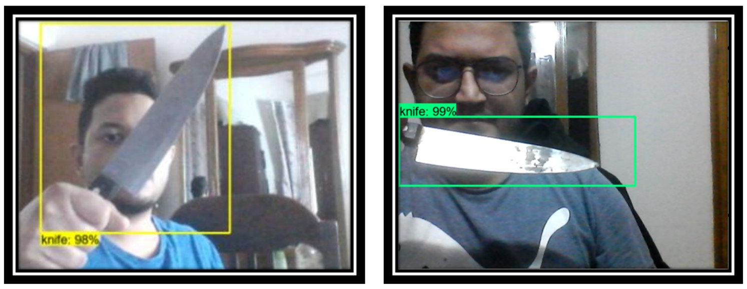


Figure 5.2: Knife Detected

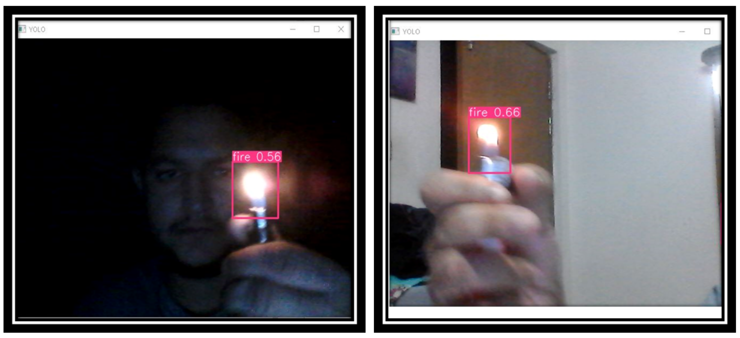


Figure 5.3: Fire Detected

5.3 Recommended Model

Of all our trained models YOLOv4, YOLOV5s, and SSD MobileNet performed the best in terms of performance and accuracy. YOLOV4 is the best model from our research with a high mAP score and good fps speed of 25-35 FPS, but this version has a larger weight file size and requires good computational power to run the interface. On the other hand, YOLOv5s is very small in size and yet it is not as accurate in tracking data as YOLOv4 and SSD is also very fast in detecting objects. In the end, YOLOv4 is the one we would recommend in terms of threat detection in Real Time.

Chapter 6

Discussion

We started this research during the peak time of COVID pandemic back in Fall 2020, sitting at our homes in different cities, with no access to proper lab facilities, computers with good processing units and powerful GPUs. Moreover, we had to do almost 100% of our work without being able to meet our group mates or advisors directly which created communication gaps as well. However, we have finally completed our work using our simple personal computers with so much dedication and efforts and guidance of our teachers. At the end, we found out the following factors about our research.

6.1 Limitations

First of all, during data collection and pre-processing, we used Google Colab for training the chosen model. However, due to some limitations and restrictions of GPU usage in Google Colab, we could not complete our training properly. Later, we switched to our personal devices and mostly used a laptop containing Intel Core i5 processor and NVIDIA GeForce 1050ti Graphics Processing Unit. This created a huge challenge for us to train the models with large number of Epoch which had some impact on results. Besides, at the last stage of testing, we omitted using GPU and ran the tests of all models by using CPU only to check the CPU usage of each model and their compatibility in systems with low configurations. Due to our CPU's limitations, we think some results may differ with other tests done in high configuration machines. We believe computers with higher and better configurations could produce even more accurate results.

Secondly, we have tested our models by using image inputs, video of many different qualities to check the model's accuracy and fastness. Next, to detect objects in real time from live video footage as inputs, we have used laptop webcams. In real life surveillance systems, we use CCTV camera which we could not manage during testing our models due to pandemic situation, hence we could not arrange a real life scenario. We believe the results may differ very slightly if we implement the models by using inputs collected from CCTV camera footage in a real life environment setup.

And lastly, due to the vastness of our research and complexity of the models, we depended heavily on our research before jumping into testing phase. We covered a large part of existing older models and decided to test the latest models. As the technology of computer vision and object detection models are evolving at a great pace, the results may not stand as constants.

6.2 Future Work

From our findings from the research, the experiences from testing and by evaluating the limitations of our work, we hope to develop a surveillance system on our own by using one of the models or a combination of the best suited models that will feature the followings.

i) A software/app that will be able to notify the responsible owner or authorities about intruders by analyzing database.

ii) Activate necessary and available alarm systems at the house or building by itself without manual help.

iii) Lock digital doors/gates (if available) according to the scan results after following action (ii).

iv) Detect any accidents like bomb blast, vehicle crash etc. and notify the admin/owner/law enforcement agencies instantly.

v) Notifying securities in case of any human made trouble detection such as fighting, kidnapping, riot etc.

vi) Detecting suspicious human activities by sensing long term data analyzing, so that threats like bank robbery, terrorist attacks etc. can be sensed by the machine without much involvement of direct human security personnel.

Chapter 7

Conclusion

As the world is moving towards a machine dependent life, there will be very few chances of human errors. Today, so many accidents, life threatening situations occur due to human made errors, negligence, incompatibility and limitations. Also, the existing security systems are mostly dependent on layers of different sensors where there lays a huge risk of system failure due to damage in one single layer. Our work heavily focused on bringing the current surveillance and security systems under one roof by utilizing the potential of CCTV cameras. Besides, we worked on finding the best models, where YOLOv4, YOLOv5s and SSD came out as the best, which can be implemented by the CCTV device manufacturers in their preferred systems from high-spec to a minimum spec device accordingly. Above all, we wanted to make a small contribution in building cost efficient systems and bring security systems within mass people's affordability so that not only military bases or high valued places, but also normal people can have a better security system at their own houses and live life with a little more safety.

Bibliography

[1] 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, 32(9):1627–1645, 2010.

[2] R. Kachhava, V. Shrivasta, R. Jain, and E. Chaturvedi, "Security system and surveillance using real time object tracking and multiple cameras," *Advanced Materials Research*, vol. 403-408, pp. 4968–4973, Nov. 2011. DOI: 10.4028/www.scientific.net/AMR.403-408.4968.

[3] ——, "Security system and surveillance using real time object tracking and multiple cameras," *Advanced Materials Research*, vol. 403-408, pp. 4968–4973, Nov. 2011. DOI: 10.4028/www.scientific.net/AMR.403-408.4968.

[4] Y.-K. Wang, C.-T. Fan, C. Yu Ke, and P. Deng, "Real-time camera anomaly detection for real-world video surveillance," vol. 4, Aug. 2011, pp. 1520–1525. DOI: 10.1109/ICMLC.2011.6017032.

[5] ——, "Real-time camera anomaly detection for real-world video surveillance," vol. 4, Aug. 2011, pp. 1520–1525. DOI: 10.1109/ICMLC.2011.6017032.

[6] K. A., S. I., and H. G. E., *"ImageNet classification with deep convolutional neural networks" Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS '12) ; Lake Tahoe, NV.* USA, Dec. 2012, pp. 1097–1105.

[7] H. Lahamy and D. D. Lichti, " "towards real-time and rotation invariant American sign language alphabet recognition using a range camera, " Sensors (Switzerland)," vol. 12, no. 11, p. 14, 2012.

[8] T. Dean, M. Ruzon, M. Segal, J. Shlens, S. Vijayanarasimhan, J. Yagnik, *et al.*, "Fast," *accurate detection of 100*, vol. 000, pp. 1814–1821, 2013.

[9] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, *Overfeat: Integrated recognition, localization and detection using convolutional networks.* CoRR, abs/1312.6229, 2013.

[10] D. Erhan, C. Szegedy, and A. Toshev, *“Scalable Object Detection using Deep Neural Networks”*. The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 2147–2154.

[11] C. Szegedy, W. Liu, Y. Jia, *et al.*, *Going deeper with convolutions.* CoRR, abs/1409.4842, 2014.

[12] J. Redmon, S. Divvala, R. Girshick, and R.-T. O. D. “You Only Look Once: Unified, *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2016, pp. 779–788.

[13] J. Redmon, S. Divvala, R. Girshick, and A. F. “ O. L. O. Unified, “Real-Time Object Detection” Proceedings of the IEEE conference on computer vision and pattern recognition,” *Las Vegas, NV, USA*, vol. 10. 2016.

[14] J. Wu, L. Sun, and R. Jafari, ““a wearable system for recognizing American sign language in real-time using IMU and surface EMG sensors,” *” IEEE journal of biomedical and health informatics*, vol. 20, no. 5, 2016.

[15] V. Bheda and D. Radpour, ““using deep convolutional networks for gesture recog- nition in American sign language,” *” arXiv preprint arXiv:1710*, p. 06 836, 2017.

[16] K. Gauen, R. Dailey, J. Laiman, *et al.*, “Comparison of visual datasets for machine learning,” in *IEEE International Conference on Information Reuse and Integration*, Aug. 2017, pp. 346–355.

[17] J. Redmon and A. Farhadi, *“YOLO9000: Better.* Faster, Stronger”, The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 7263–7271.

[18] Y. Wang, C. Wang, H. Zhang, C. Zhang, and Q. Fu, “Combing single shot multibox detector with transfer learning for ship detection using chinese gaofen-3 images,” Nov. 2017, pp. 712–716. DOI: 10.1109/PIERS-FALL.2017.8293227.

[19] R. J. and F. A., *YOLOv3: An Incremental Improvement.* 02767, 2018.

[20] M. A. Jalal, R. Chen, R. K. Moore, and L. Mihaylova, ““american sign language posture understanding with deep neural networks,” *” in 2018 21st International Conference on Information Fusion (FUSION), 573–579, IEEE*, vol. 10. 2018.

[21] J. Redmon, “Yolo: Real-time object detection,” *[Online] https: //pjreddie.com/darknet/yolo. Accessed May*, vol. 7, 2018.

[22] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” *arXiv*, 2018.

[23] W. Tao, M. C. Leu, and Z. Yin, "“american Sign Language alphabet recognition using Convolutional Neural Networks with multiview augmentation and infer- ence fusion,” " *Engineering Applications of Artificial Intelligence*, vol. 76, 2018.

[24] L. Wei Yang and C. Yen Su, "Low-cost cnn design for intelligent surveillance system," in *2018 International Conference on System Science and Engineering (ICSSE)*, 2018, pp. 1–4. DOI: 10.1109/ICSSE.2018.8520133.

[25] ——, "Low-cost cnn design for intelligent surveillance system," in *2018 International Conference on System Science and Engineering (ICSSE)*, 2018, pp. 1–4. DOI: 10.1109/ICSSE.2018.8520133.

[26] J. Wu, "Complexity and accuracy analysis of common artificial neural networks on pedestrian detection," *MATEC Web of Conferences*, vol. 232, p. 01 003, Jan. 2018. DOI: 10.1051/matecconf/201823201003.

[27] L. Y. Bin, G. Y. Huann, and L. K. Yun, *"Study of Convolutional Neural Network in Recognizing Static American Sign Language.* " in 2019 IEEE International Conference on Signal and Image Processing Applications (ICSIPA), 41–45, IEEE, 2019.

[28] V. Etten and A., *January).* Satellite imagery multiscale rapid detection with windowed networks, 2019, pp. 735–743.

[29] R. M. Alaqil, J. A. Alsuhaibani, B. A. Alhumaidi, R. A. Alnasser, R. D. Alotaibi, and H. Benhidour, "Automatic gun detection from images using faster r-cnn," in *2020 First International Conference of Smart Systems and Emerging Technologies (SMARTTECH)*, 2020, pp. 149–154. DOI: 10.1109/SMART-TECH49988.2020.00045.

[30] ——, "Automatic gun detection from images using faster r-cnn," in *2020 First International Conference of Smart Systems and Emerging Technologies (SMARTTECH)*, 2020, pp. 149–154. DOI: 10.1109/SMART-TECH49988.2020.00045.

[31] ——, "Automatic gun detection from images using faster r-cnn," in *2020 First International Conference of Smart Systems and Emerging Technologies (SMARTTECH)*, 2020, pp. 149–154. DOI: 10.1109/SMART-TECH49988.2020.00045.

[32] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, *"YOLOv4: Optimal Speed and Accuracy of Object Detection" arXiv:1506.* 02640, 2020.

[33] A. John and D. D. Meva, “A comparative study of various object detection algorithms and performance analysis,” *INTERNATIONAL JOURNAL OF COMPUTER SCIENCES AND ENGINEERING*, vol. 8, pp. 158–163, Oct. 2020. DOI: 10.26438/ijcse/v8i10.158163.

[34] A. Kumar, Z. J. Zhang, and H. Lyu, “Object detection in real time based on improved single shot multi-box detector algorithm,” *J Wireless Com Network 2020*, vol. 204, 2020.

[35] V. Singh, S. Singh, and P. Gupta, “Real-time anomaly recognition through cctv using neural networks,” *Procedia Computer Science*, vol. 173, pp. 254–263, 2020, International Conference on Smart Sustainable Intelligent Computing and Applications under ICITETM2020, ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2020.06.030. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050920315349.

[36] M. Bhatti, M. Khan, M. Aslam, and M. Fiaz, “Weapon detection in real-time cctv videos using deep learning,” *IEEE Access*, vol. PP, pp. 1–1, Feb. 2021. DOI: 10.1109/ACCESS.2021.3059170.

[37] U. Handalage and L. Kuganandamurthy, “Real-time object detection using yolo: A review,” May 2021. DOI: 10.13140/RG.2.2.24367.66723.

[38] R. Kanotra, N. W. Akash, and N. Jeyanth, *International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249-8958 (Online).* Volume-10 Issue-4, Apr. 2021.

[39] P. Kumar, N. Swamy s, P. Kumar, G. Purohit, and S. R. Kota, “Real-time, yolo-based intelligent surveillance and monitoring system using jetson tx2,” Jan. 2021, pp. 461–471, ISBN: 978-981-15-8334-6. DOI: 10.1007/978-981-15-8335-3_35.

[40] L. Tan, T. Huangfu, L. Wu, *et al.*, “Comparison of YOLO v3, Faster R-CNN, and SSD for Real-Time Pill Identification,” *30*, Jul. 2021.