# An Intelligent Weapon Detection System for Surveillance Cameras

Graduation Project I (Final Report)


By:

**Mohamed Nasser Hashem,** ID: **372029063**

**Mustafa Ahmed Abdulsami,** ID: **372029353**

**Husen Muhammad Kalaepeh,** ID: **352072163**

Supervised By:

**Dr. Emad Nabil**



**Faculty of Computer and Information Systems**

**Islamic University of Madinah**


January - 2022

# DEDICATION

This work is completely dedicated to my grandfather (whom my eyes have not seen), respectful Parents, beloved wife, and darling son. without whose constant support this was not possible.

Mohamed Nasser Hashem

# ACKNOWLEDGMENT

The success and results of this project have required a tremendous amount of guidance, and fortunately we have achieved that throughout our project.

All that we did was, firstly, by the grace of Allah Almighty for His bounty to complete the project, without his blessings, we would not be able to do anything.

Then, we like to express our special thanks and gratitude to our supervisor, Dr. Emad Nabil, who helped us and directed us throughout the period of work on this project then.

In the end, the support and help from people around us we think it is necessary to thank them, we thank our professors and colleagues who have been supportive of us throughout the project.

# DECLARATION

We hereby certify that this material, which we now submit for assessment on the program of study leading to the award of Bachelor of *(Computer Science*) is entirely my own work, that We have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge breach any law of copyright, and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

**Stud ID.:** 372029063, **Stud Name:** Mohamed Nasser Hashem, **Signed:** __Mohamed__

**Stud ID.:** 372029353, **Stud Name:** Mustafa Ahmed Abdulsami, **Signed:** __Mustafa__

**Stud ID.:** 352072163, **Stud Name:** Husen Muhammad Kalaepeh, **Signed:** __ Husen__

**Date:** 28/12/2021

# ABSTRACT

Preserving humans' life and valuables is the top requirement for any society. Due to the existing of so many criminals or menacing tactics, it is almost impossible to discern them by visual ability, as they easily hide inside large crowds. For this, the developers and the police are working to secure it both technically and realistically. Because security alone is unable to deal with such matters. Due to modern technology, the current era, to the researchers' background in computer science, we can work to create a detection system for surveillance.

In this project, we will develop an intelligent system that is able to detect both a human and some weapons that are used by criminals. The detected weapons are guns, rifles, and knives. After the detection of one of the weapons, an alarm will be raised. system can be used in public places. using modern machine learning technologies for object detection to achieve that goal.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

*Chapter One*

# 1 INTRODUCTION

In the age of technology, everything in this life has become connected to the world of technology. Coins, diamonds, gold, jewelry, antiques, and other valuables. Humans alone are no longer able to protect these precious things. This is on the physical side.

On the human side, the matter has become more different. The presence of humans has become more intense in public places, such as schools, universities, and parks. With so many, criminals or threatening tactics are virtually impossible to discern.

Security threats have become common and a reality in this time, threats may be from forced robbery to mass hostage-taking. For this, the developers and the police are working to secure it both technically and realistically. It is our duty as developers now to work seriously to obtain security monitoring systems in cooperation with security agencies.

In this project we will try to use the latest open-source object detection algorithms and datasets. To create an intelligent Weapon Detection system for surveillance. to start in our work, we started by doing a detailed and meticulous research into the two main sections of this project: the algorithms that will be used as a model for the system, and open-source datasets related to our project (knife, pistol, rifle).

For algorithms, the literature review began examining each algorithm by: research paper, and source-code. Which led to the arrangement of algorithms according to strength and speed, which led us to define the YOLO algorithm. As for data sets, all open-source databases were compiled and filtered, resulting in more than 35,000 images.

## 1.1 AIM

Create a system connected to a camera that can identify the criminal who enters public places, by identifying if he is carrying a gun, trifle, or a knife. and to give a warning to all those present in this facility and warn them.

## 1.2 PROPLEM STATEMENT

Criminals that enter public places is spreading all over the world as well. It spreads in remote places or branches due to the lack of sufficient security. The thief always carries a gun, a knife, and an ordinary person cannot stop him or warn others about him. He threatens the victim and is taken with hatred and leaves behind many economic problems. No one can defend himself in front of that thief because of the weapon he carries, and in many cases the police cannot reach that thief. Again, theft is repeated a lot without any deterrent to stop them, arrest them, or even flee from them.

## 1.3 OBJECTIVES:

The main objective of Weapon Detection System is recognized and find at least one viable focus from still picture or video information. It thoroughly incorporates an assortment of It comprehensively includes a variety of important techniques.[1] Following are the primary objectives:

1. Create model(s) for threat/Weapon Detection
2. Train the model(s) using relevant data sets
3. Measure the performance of the model(s).
4. Enhance the system until it reaches an accepted accuracy rate.

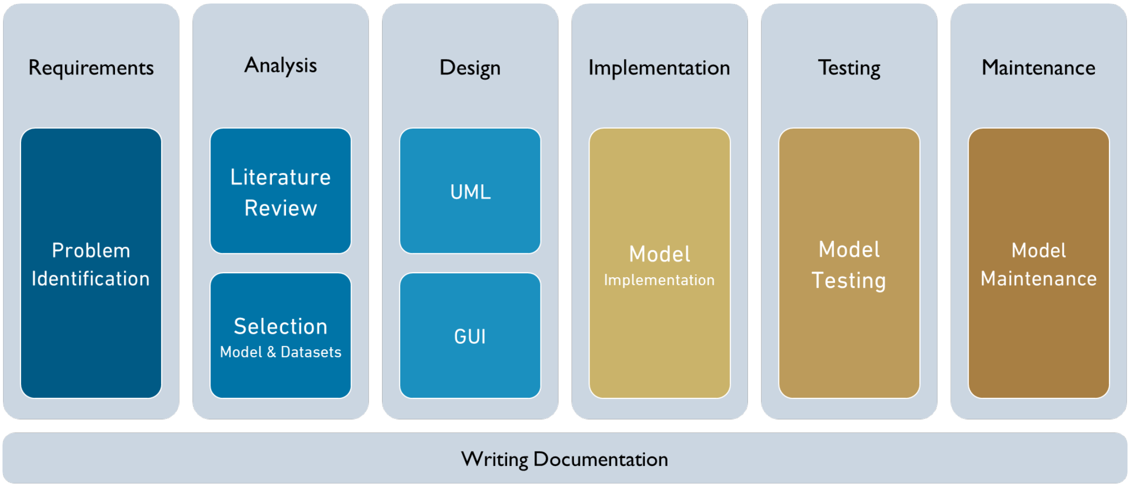
## 1.4 SCOPE:

- Detecting persons(s) holds a gun, trifle, or knife from a camera video stream.
- The system will be in the form of web/desktop application
- The system will give an alarm in case of detection a threat with the mentioned properties.

## 1.5 METHODOLOGY:

A methodology is '"a system that sets guidelines for solving a problem, with components including phases, tasks, methods, methods, and tools." depending on the requirements and the project type.

In this system we have looked carefully for the requirements, and we have decided that the best approach to develop the system is using the Waterfall Methodology. We believe that using this approach can help us reach our goals and deliver the best quality.




In next lines, there is a brief about the Waterfall Methodology.

### 1.5.1 Waterfall:

The waterfall Methodology is "a design process model that used in software development processes commonly." The workflow progress is in fixed form of pieces that start from top to down through these stages: The process includes the following steps: planning, analysis, design, construction, testing, production, implementation, and maintenance.

## 1.6 TIMELINE:

| Task Mode | Task Name | Duration | Start | Finish | % Complete |
|---|---|---|---|---|---|
| | **Graduation Project** | **202 days** | **30/08/2021** | **07/06/2022** | **43%** |
| | **1 Introduction** | **14 days** | **30/08/2021** | **16/09/2021** | **100%** |
| | 1.1 Aim | 3 days | 30/08/2021 | 01/09/2021 | 100% |
| | 1.2 Problem Statement | 3 days | 06/09/2021 | 08/09/2021 | 100% |
| | 1.3 Objectives | 3 days | 09/09/2021 | 13/09/2021 | 100% |
| | 1.4 Proposed Solution | 2 days | 14/09/2021 | 15/09/2021 | 100% |
| | 1.5 Methdology | 1 day | 22/09/2021 | 22/09/2021 | 100% |
| | **2 Literature Review** | **20 days** | **19/09/2021** | **14/10/2021** | **100%** |
| | 2.1 Background | 4 days | 27/09/2021 | 30/09/2021 | 100% |
| | 2.2 Algorithms | 7 days | 27/09/2021 | 05/10/2021 | 100% |
| | 2.3 Datasets | 7 days | 06/10/2021 | 14/10/2021 | 100% |
| | **3 Analysis and Design** | **30 days** | **18/10/2021** | **26/11/2021** | **100%** |
| | 3.1 Functional & Non-Functional Requirements | 10 days | 18/10/2021 | 29/10/2021 | 100% |
| | 3.2 UML | 10 days | 08/11/2021 | 19/11/2021 | 100% |
| | 3.3 GUI | 5 days | 22/11/2021 | 26/11/2021 | 100% |
| | **4 Implementation and Testing** | **130 days** | **28/11/2021** | **26/05/2022** | **20%** |
| | 4.1 Implementation | 100 days | 22/11/2021 | 08/04/2022 | 25% |
| | 4.2 Testing | 15 days | 18/04/2022 | 06/05/2022 | 0% |
| | 4.3 Maintenance | 9 days | 10/05/2022 | 20/05/2022 | 0% |
| | 5 Conclusion and future work | 8 days | 27/05/2022 | 07/06/2022 | 0% |

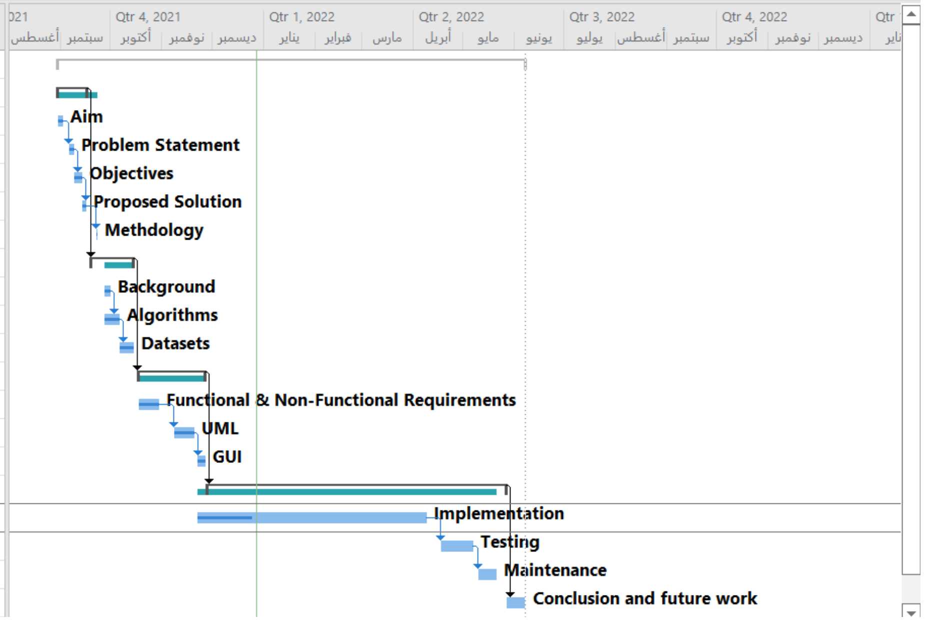



Figure 1-2: Giant Chart for Project

*Chapter Two*

# 2 LITERATURE REVIEW

Object detection is a computer vision task that detects instances of visual objects of specific classes (such as persons, animals, cars, or buildings) in digital pictures like photos or video frames. The purpose of object detection is to create computational models that give computer vision applications the most basic information they need, in our case, we will study it and we compare a set of algorithms trying to produce the best one or the one that works well on our idea

## 2.1 BACKGROUND

Object recognition is a wide term that refers to a set of related computer vision tasks that include identifying objects in digital pictures. Image classification is defined as "predicting the class of one item in a picture." Object localization is the process of determining the position of one or more things in a photograph and drawing a bounding box around their extent. Object detection combines these two tasks by identifying and categorizing one or more objects in a picture. Therefore, three distinct computer vision tasks may be identified:

### 2.1.1 Image Classification

Predict the type or class of an object based on a photograph.[2] As input, a single-object image, such as a photograph, is utilized. As a consequence, a class label is created (e.g., one or more integers that are mapped to class labels).[3]

### 2.1.2 Object Localization

Determine the presence of items in a photograph and use a bounding box to pinpoint their location. As input, an image with one or more things, such as a photograph, is utilized. As an output, one or more bounding boxes (e.g., defined by a point, width, and height).

### 2.1.3 Object Detection

Using a bounding box, determine the presence of things in an image and the types or classes of the objects discovered.[3] As input, an image with one or more things, such as a photograph, is utilized. One or more bounding boxes (e.g., specified by a point, width, and height) are produced, together with a class label for each bounding box.

To understand more about Object detection what and what are the most famous types and how they are used Object detection is a computer vision task that detects instances of visual objects of specific classes (such as persons, animals, cars, or buildings) in digital pictures like photos or video frames. The purpose of object detection is to create computational models that give computer vision applications the most basic information they need, in our case, we will study it and we compare a set of algorithms trying to produce the best one or the one that works well on our idea.

### 2.1.4 Neural networks

In the domains of AI, machine learning, and deep learning, these models replicate human brain behavior, allowing computer systems to spot patterns and solve common issues.

Artificial neural networks (ANN) and simulated neural networks (SNN) are a subset of machine learning that are at the heart of deep learning methods. Their name and structure derived from the human brain, and they resemble the way biological neurons communicate with one another.

A node layer contains an input layer, one or more hidden layers, and an output layer in artificial neural networks (ANN). Each node, or artificial neuron, which connected to the others and has a weight and threshold linked with it. If a node's output exceeds a certain threshold value, the node activated, then data sent to the next tier of the network. Otherwise, no data is sent on to the network's next tier [4]–[6].

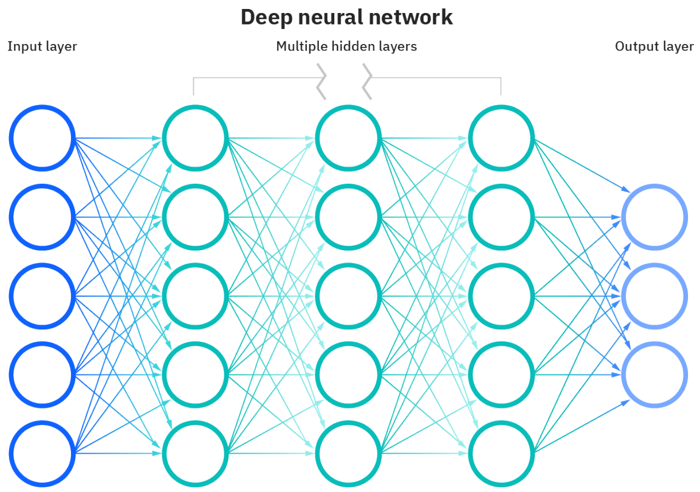



**Figure 2-1: Layers of Deep Neural Network**

Training data used by neural networks to learn and increase their accuracy over time. However, once these learning algorithms have fine-tuned for accuracy, they become formidable tools in computer science and artificial intelligence, allowing us to quickly classify and cluster data. When compared to manual identification by human experts, tasks in speech recognition or image recognition can take minutes rather than hours. Google's search algorithm is one of the most well-known neural networks

Distinct types of neural networks exist, each of which is employed for a different purpose. While this is not an exhaustive list, the following are some of the most popular types of neural networks that you will come across for common applications:

Frank Rosenblatt invented the perceptron in 1958, and it is the oldest neural network. It is the simplest type of a neural network, with only one neuron:

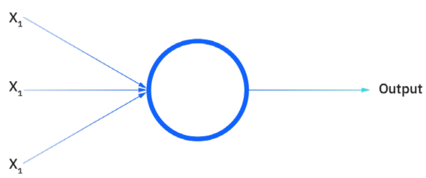



**Figure 2-2: Simplest Form of a Neural Network**

This part of the project has mostly focused on feedforward neural networks, often known as multi-layer perceptron’s (MLPS). An input layer, a concealed layer or layers, and an output layer make up these layers. While these neural networks are also known as MLPs, it is important to remember that they are made up of sigmoid neurons rather than perceptron’s because most real-world problems are not linear. These models provide the cornerstone for computer vision, natural language processing, and other neural networks, and they are typically fed data to train them.

Like feedforward networks, convolutional neural networks (CNNs) are used for image recognition, pattern identification, and/or computer vision. These networks use linear algebra principles, notably matrix multiplication, to find patterns in images.

The feedback loops distinguish recurrent neural networks (RNNs). These learning algorithms are used to create predictions about future outcomes using time-series data, such as stock market projections or sales forecasting [4]–[6].

In the upcoming pictures, best types of object detection help in a clear picture output, the difference between them, and their speed of identification and exploration.

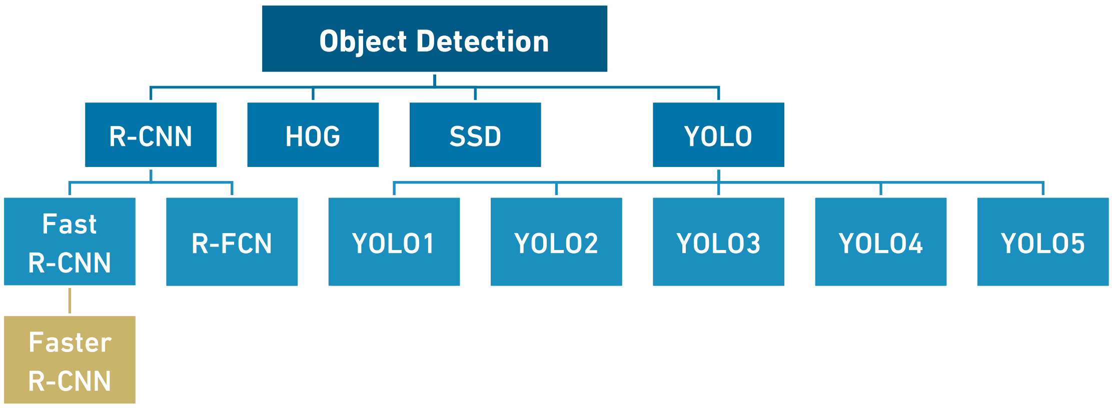



**Figure 2-3: Object Detection Algorithms series**

## 2.2 ALGORITHEMS

**Table 2-1: Summary of Algorithms**

| Algorithms | Definition | Creation |
|---|---|---|
| Histogram of Oriented Gradients (HOG) | Canny Edge Detector and SIFT are examples of feature descriptors (Scale Invariant and Feature Transform). counts how many times a gradient orientation appears in a certain region of a picture. | 1986 |
| Single Shot Detector (SSD) | Detect several objects inside a picture using a single shot. | 2016 |
| Region-based Fully Convolutional Network (R-FCN) | Reduces the amount of work necessary for each ROI, which speeds up the process. | 2016 |
| Fast R-CNN | The CNN is fed the input picture, which creates a convolutional feature map. As a starting point, use the convolutional feature map. | 2015 |
| Faster R-CNN | deep convolutional network for object detection that appears to the user as a single, end-to-end, unified network. | 2015 |
| Region-based Convolutional Neural Networks (R-CNN) | Combination of region recommendation and Convolutional Neural Networks (CNNs) | 2014 |
| You Only Look Once (YOLO) | A single neural network is used in an object detection method. In contrast to several other object detection algorithms that do a bit-by-bit scan of the picture. | 2016 |

In the next lines, we are going to have a brief for each algorithm.

### 2.2.1 Histogram of Oriented Gradients (HOG)

In 2005, Navneet Dalal and Bill Triggs presented highlights from Histogram of Situated Gradients (HOG). The Histogram of Arranged Inclinations (Hoard) is a component descriptor used in image processing, mostly for object detection. A representation of an image or a picture repair that enhances the picture by isolating useful data from it is called an element descriptor.[7], [8]

The histogram of organized inclinations descriptor is based on the idea that the dispersion of force angles or edge bearings might represent the look and form of nearby objects inside an image. Because the extent of angles is limited, the x and y subsidiaries of an image (Inclinations) are useful Because of the abrupt shift in power near edges and corners, is enormous, and we understand that edges and corners pack in far more info about object form than level locations. As a result, the histograms of inclination headings are used as items in this descriptor.

#### Object detection workflow with HOG

We will go on to how we compute the histograms and how the element vectors obtained from the Hoard descriptor are used by the classifier such as SVM to identify the concerned article now that we understand the fundamental idea of Histogram of Situated Slopes.

#### How does it work?

Normalizing the image is part of the preprocessing process, but it is entirely optional. Its purpose is to improve the Hoard descriptor's execution. We do not use any preprocessing standards because we are only creating a simple descriptor.

### 2.2.2 Single Shot Detector (SSD)

The Single Shot Detector (SSD) is an engineering improvement for the VGG16 that calculates article discovery. It was released at the end of November 2016 and achieved new standards for object identification accuracy and execution, reaching over 74% Guide (mean Normal Accuracy) at 59 edges per second on common datasets such as PascalVOC and COCO. [9], [10]

#### Architecture

The SSD architecture builds on the well-known VGG-16 engineering but eliminates all the layers that go with it.

VGG-16 was chosen as the basic organization for the following reasons: a reputation for excellent accomplishment in top-notch picture order endeavors situations in which move learning aids in the development of results Instead of the initial VGG fully associated layers, several auxiliary convolutional layers (from conv6 onwards) were introduced, allowing for the separation of components at different scales and the logical fall in the amount of the contribution to each resultant layer.

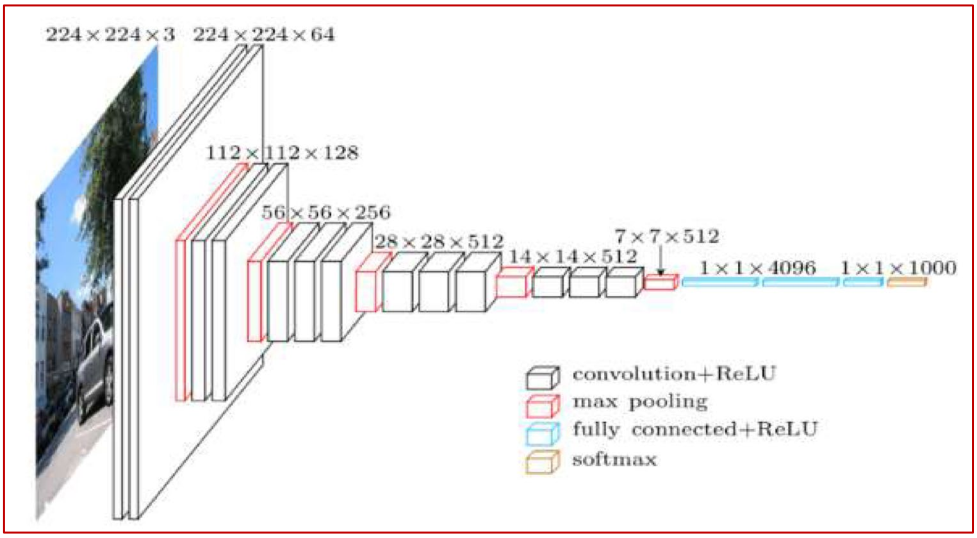



**Figure 2-4: VGG-16 Architecture**

### 2.2.3 Region-based Convolutional Neural Networks (R-CNN)

The Region-based Convolutions Network procedure (R CNN) is a blend of district proposal with Convolution Neural Networks (CNNs). R-CNN helps in confining articles with a huge affiliation and drawing up a high-line model with a smidgen of proportion of explained region information. It accomplishes shocking article conspicuous verification exactness by utilizing a critical ConvNet to organize object proposition. R-CNN can scale to a significant number of thing classes without going to procedures, including hashing.[7]–[11]

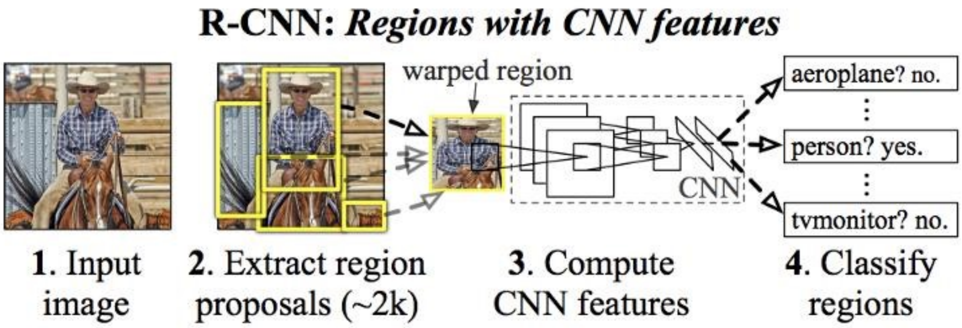



**Figure 2-5: R-CNN system overview**

1. Takes an image as input.
2. Input image as input.
3. Uses a massive convolutional neural network to compute features for each proposal (CNN).
4. Classify each area using class specific linear SVMs.

**Drawbacks**

- It finds Regions of Interest using the Selective Search Algorithm, which is a slow and time-consuming
- procedure that requires each image is used to categorize 2000 region proposals. As a result, training takes a long time. Detecting objects in a picture on GPU takes 49 seconds.
- A large amount of disk space is also required to store the region proposal's feature map.

### 2.2.4 Fast R-CNN

Fast R-CNN was a similar producer to the previous paper (R-CNN) that addressed a few problems of R-CNN to build a quicker article affirmation computation. The method looks to be like the R-CNN assessment. In any case, rather than controlling the region notion using CNN, we feed the information image to CNN to send a convolutional include map. We take the region of suggestion from the convolutional highlight map and curve it into squares, then reshape it into an acceptable size using a RoI pooling layer, so it can be managed into a completely relevant overall image.[11]–[16]

Based on the RoI highlight vector, we utilize a delicate max layer to forecast the class of the proposed zone and the offset respects for the jumping box. The reason (Fast R-CNN) is faster than R-CNN is because you do not have to manage 2000 locale concepts to the convolutional neural connection when in doubt. Taking everything into consideration, the convolution activity is done just once per picture, resulting in the generation of a section map.

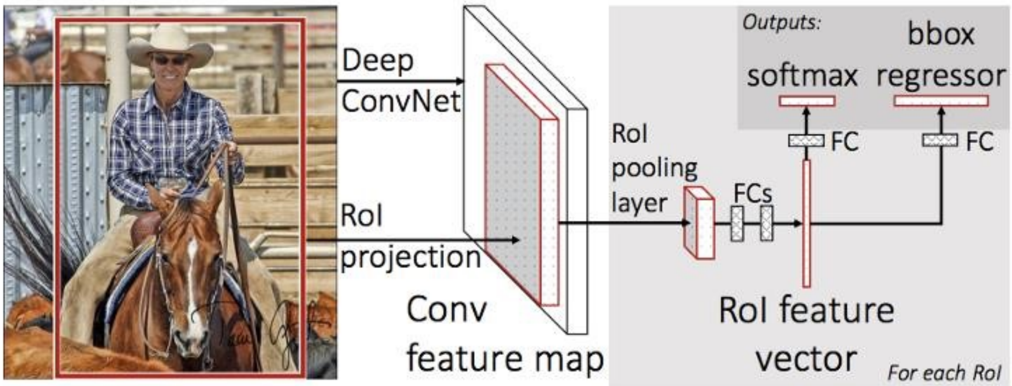



**Figure 2-6: Fast R-CNN system overview**

#### Benefits

- Higher identification quality (mAP) than R-CNN, SPPnet
- Training is single stage, utilizing a perform multiple tasks misfortune
- Training can refresh all organization layers
- No plate stockpiling is needed for include storing

### 2.2.5 Faster R-CNN

Both preceding algorithms (R-CNN and Fast R-CNN) employ requests to find area suggestions. Specific request is a torpid and monotonous cycle, impacting the introduction of the association. Accordingly, Shaoqing Ren, thought about a thing area computation that wipes out the pursuit estimation and permits the association to acquire capability with the region suggestion.[11]–[14], [17]–[19]

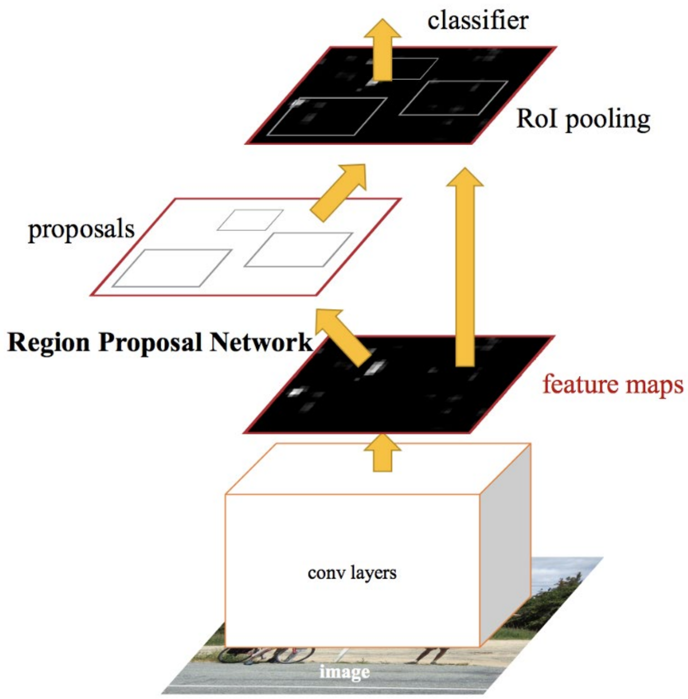



**Figure 2-7: Faster R-CNN system overview**

The picture is supplied as a commitment to a convolutional network that produces a convolutional incorporate guide, like Fast R-CNN. An alternate association is utilized to anticipate the district suggestions by utilizing special request calculation on the part manual for recognizing the region suggestion. A RoI pooling layer is then utilized to characterize the picture inside the suggested region and predict the offset respect for the skipping boxes, reshaping the expected region recommendations.

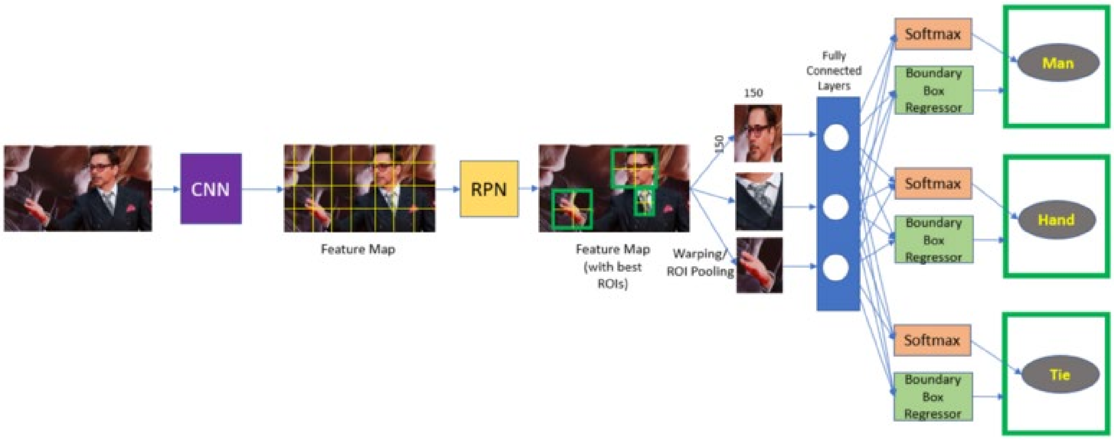



**Figure 2-8: Faster R-CNN system analysis**

Instead of Selective Search algorithm, it uses RPN (Region Proposal Network) to select the best ROIs automatically to be passed for ROI Pooling.

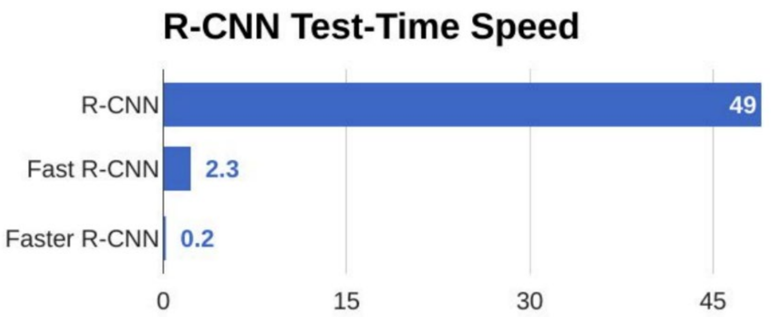



**Figure 2-9: Comparison of test-time speed of object detection algorithms**

Quicker R-CNN is clearly faster than its predecessors, as shown in the graph above. As a result, it may even be used to identify objects in real time.

### 2.2.6 Region-based Fully Convolutional Network (R-FCN)

District based Fully Convolutional Networks or R-FCN is an area-based locator for object identification. Not at all like other district-based locators that apply an exorbitant per-locale subnetwork like Fast R-CNN or Faster R-CNN, this area-based indicator is completely convolutional with all calculation shared on the whole picture.

R-FCN comprises of shared, completely convolutional designs just like the instance of FCN that is known to yield a preferred outcome over the Faster R-CNN. In this calculation, all learnable weight layers are convolutional and are intended to characterize the ROIs into object classifications and foundations.[7]

For conventional Region Proposal Network (RPN) approaches like R-CNN, Fast RCNN and Faster R-CNN, region suggestions are delivered by RPN first. Then ROI pooling is done and going through totally related (FC) layers for portrayal and bobbing box backslide. The collaboration (FC layers) after ROI pooling does not split among ROI, and saves time, which makes RPN approaches slow. Also, the FC layers increase the number of affiliations (limits) which moreover increase the multifaceted nature.

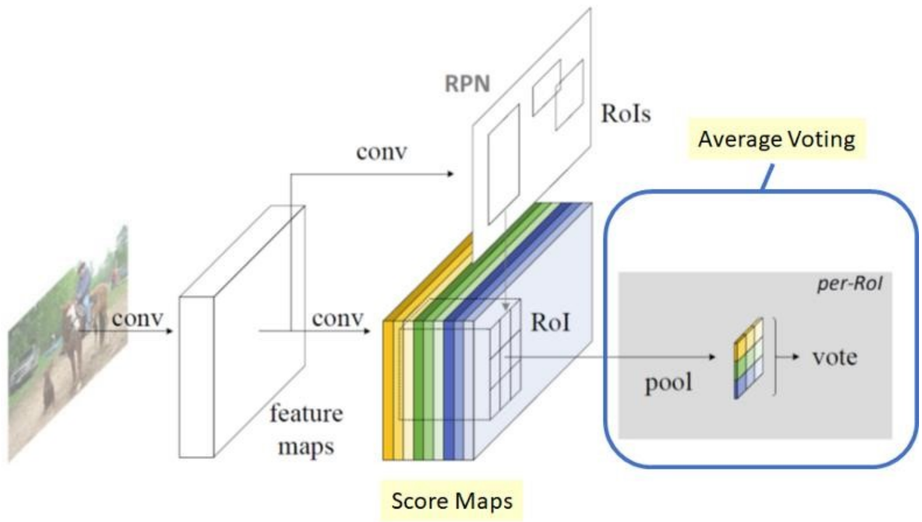



**Figure 2-10: R-FCN system overview**

In R-FCN, we use RPN to generate district recommendations, however unlike the R-CNN series, FC layers are removed after ROI pooling. All substantial complexity is shifted before ROI pooling to create the score maps, assuming all other factors are identical. Following ROI pooling, every district proposal will use a same set of score guidelines to conduct normal democratic, which is a fundamental estimation. As a result, there is no learnable layer after the ROI layer, which is costless. R-FCN is faster than Faster R-CNN with smaller mAP as a result.

### 2.2.7 You Just Look Once (YOLO)

YOLO is a real-time object identification technique that use neural networks. Because of its speed and precision, this algorithm is extremely popular. [20]–[23]

#### What is YOLO?

The term 'You Only Look Once' abbreviated as YOLO. This is an algorithm for detecting and recognizing different items in a photograph (in real-time). Object detection in YOLO done as a regression problem, and the identified photos' class probabilities provided.

Convolutional neural networks (CNN), that used in the YOLO method to recognize objects in real time. To detect objects, the approach just takes a single forward propagation through a neural network, as the name suggests.

This indicates that a single algorithm run used to forecast the entire image. The CNN used to forecast multiple bounding boxes and class probabilities at the same time.

The YOLO algorithm consists of various variants. include tiny

- YOLO1
- YOLO2
- YOLO3
- YOLO4
- YOLO5

#### What is the significance of the YOLO algorithm?

YOLO algorithm is important because of the following reasons:

- **Speed:** Because it can predict objects in real time, this approach enhances detection speed.
- **High accuracy:** YOLO is a predictive approach that yields precise findings with low background noise.
- **Learning capabilities:** The method has strong learning capabilities, allowing it to learn object representations and use them in object detection.

### How the YOLO algorithm works

YOLO algorithm works using the following three techniques :

- **Residual blocks**
- **Bounding box regression**
- **Intersection Over Union (IOU)**

Residual blocks

The image separated first into several grids. The dimensions of each grid are $S \, x \, S$. The graphic below shows how a grid is created from an input image.


**Figure 2-11: Residual blocks technique**

There are several grid cells of identical size in the image above. Objects that appear within grid cells will be detected by each grid cell. If an item center emerges within a specific grid cell, for example, that cell will be responsible for detecting it.

Bounding box regression

A bounding box is an outline that draws attention to a certain object in a picture.

The following attributes are present in every bounding box in the image:

- Width: $b_w$
- a certain height: $b_h$
- class (for example, person, car, traffic light, etc.): $c$
- Center of the bounding box: $(b_x, b_y)$

A bounding box is illustrated in the image below. A yellow outline has been used to depict the bounding box.

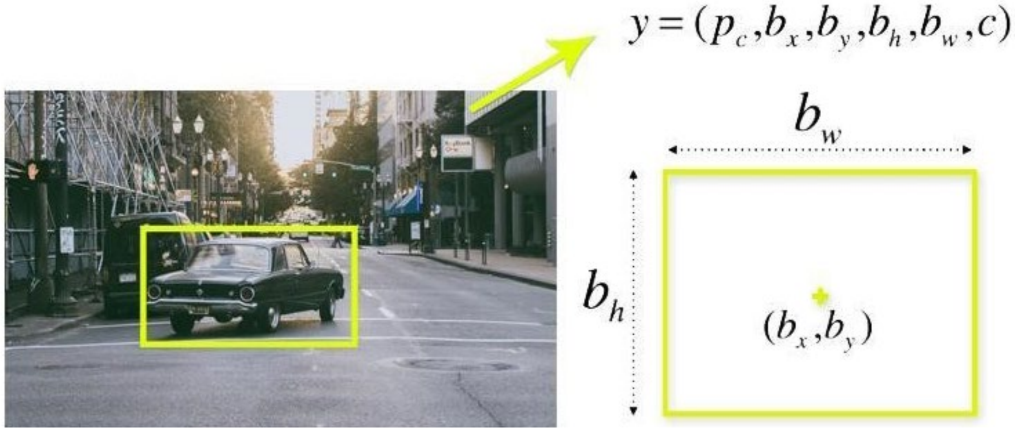



**Figure 2-12: Bounding box regression technique**

To forecast the height, width, center, and class of objects, YOLO use a single bounding box regression. The likelihood of an object appearing in the bounding box is represented in the graphic above.

Intersection Over Union (IOU)

The concept of intersection over union (IOU) illustrates how boxes overlap in object detection. YOLO uses IOU to create an output box that properly surrounds the items.

The bounding boxes and their confidence scores are predicted by each grid cell. If the anticipated and bounding boxes are identical, This approach removes bounding boxes that are not the same size as the actual box.

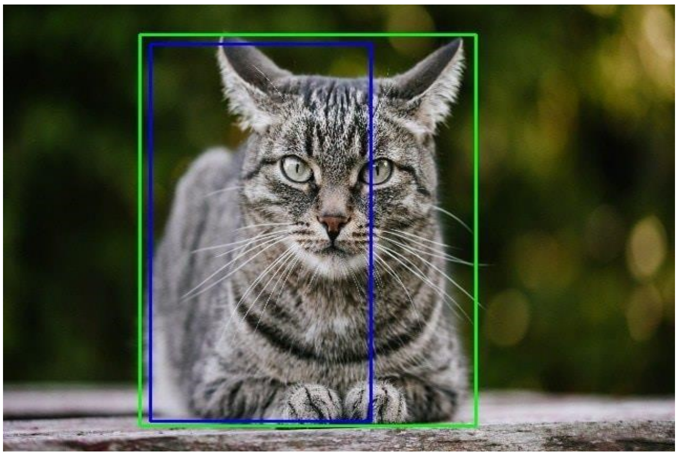


**Figure 2-13: Intersection Over Union technique**

There are two bounding boxes in the image above, one in green and the other in blue. The blue box represents the anticipated box, and the green box represents the actual box. YOLO makes sure the two boundary boxes are the same size.

### Using a combination of the three techniques

The graphic below depicts how the three techniques are combined to generate the final detection findings.

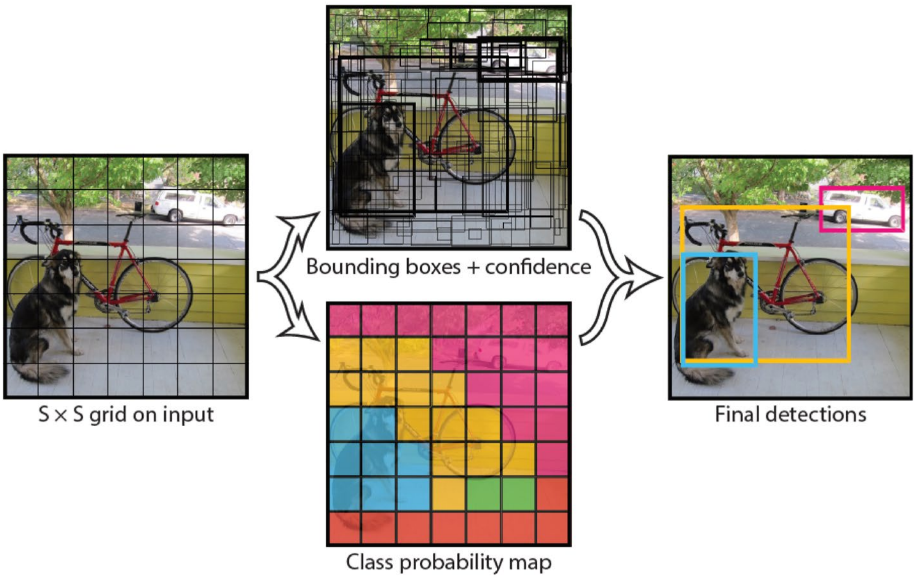



**Figure 2-14: Combination of the techniques (Final Algorithm)**

The image, first subdivided into grid cells. B bounding boxes are forecasted in each grid cell, along with their confidence scores. To determine the class of each object, the cells estimate the class probability. [24]

We can see at least three types of objects, for example: a car, a dog, and a bicycle. A single convolutional neural network used to make all the predictions at the same time.

The predicted bounding boxes are equal to the true boxes of the objects when intersection over union used. This phenomena gets rid of any extra bounding boxes that do not fit the objects' properties (like height and width). The final detection will be made up of distinct bounding boxes that exactly suit the objects.

The pink bounding box, for example, surrounds the car, whereas the yellow bounding box surrounds the bicycle. The blue bounding box has been used to highlight the dog.

### YOLOv5

This version is incredible; it outperforms all prior versions and comes close to Efficient AP in terms of FPS. This can be seen in the graph below.

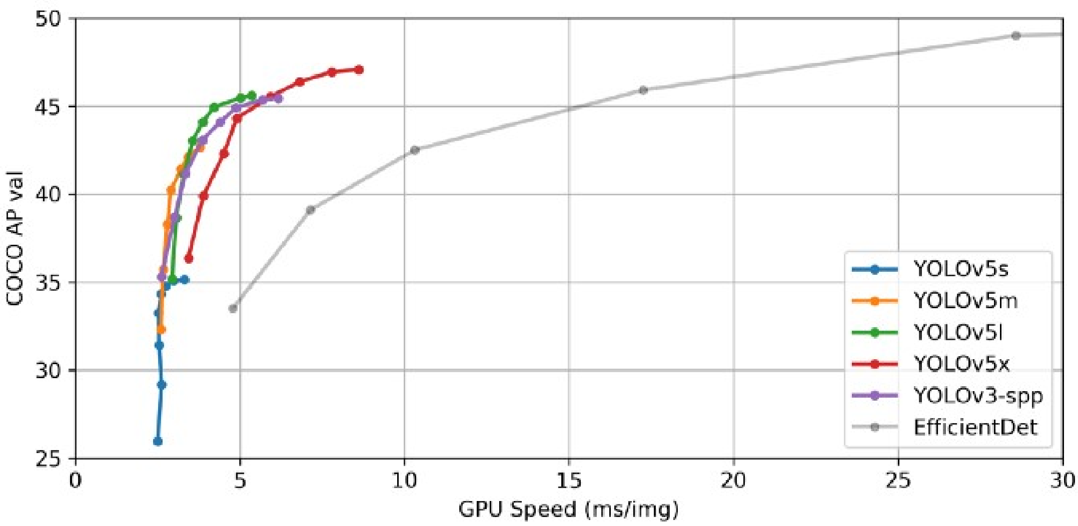



**Figure 2-15: YOLO models comparison**

The YOLOv5 model is the most current addition to the YOLO family of models. YOLO was the first object detection model to incorporate bounding box prediction and object classification into a single end-to-end differentiable network. It was created and is maintained using the Darknet framework. YOLOv5 is the first YOLO model to be written in the PyTorch framework, making it significantly lighter and easier to use. However, YOLOv5 does not outperform YOLOv4 on a standard benchmark, the COCO dataset, because it did not make fundamental architectural improvements to the network in YOLOv4. [25]

Data Augmentation in YOLOv5

YOLOv5 transmits training data through a data loader for online augmentation with each training batch. The data loader performs three types of augmentations: scaling, correction of the color space, and mosaic enhancement. The newest technique is mosaic data augmentation, which transforms four photos into four random ratio tiles. The mosaic data loader included as part of the YOLOv3 PyTorch and, more recently, YOLOv5 repositories. Mosaic augmentation is particularly beneficial for the widely used COCO object identification benchmark, since it aids the model in learning to address the well-known "small object problem" - in which little items are not spotted as reliably as bigger objects. It is worth noting that it is worthwhile to experiment with your own set of augmentations to optimize performance on your customized work.

Both YOLOv4 and YOLOv5 use the CSP Bottleneck to generate picture features, with credit for the research going to WongKinYiu and their recent study on Cross Stage Partial Networks for Convolutional Neural Network Backbone. The CSP eliminates duplicate gradient difficulties found in other larger ConvNet backbones, resulting in fewer parameters and FLOPS for equivalent significance. This is critical for the YOLO family, as inference speed and a small model size are critical. [25]

### CSP Backbone

Dense Net used to construct the CSP models. Dense Net was created to connect layers in convolutional neural networks to alleviate the vanishing gradient problem (it is difficult to backprop loss signals through a very deep network), to improve feature propagation, to encourage the network to reuse features, and to reduce the number of network parameters.

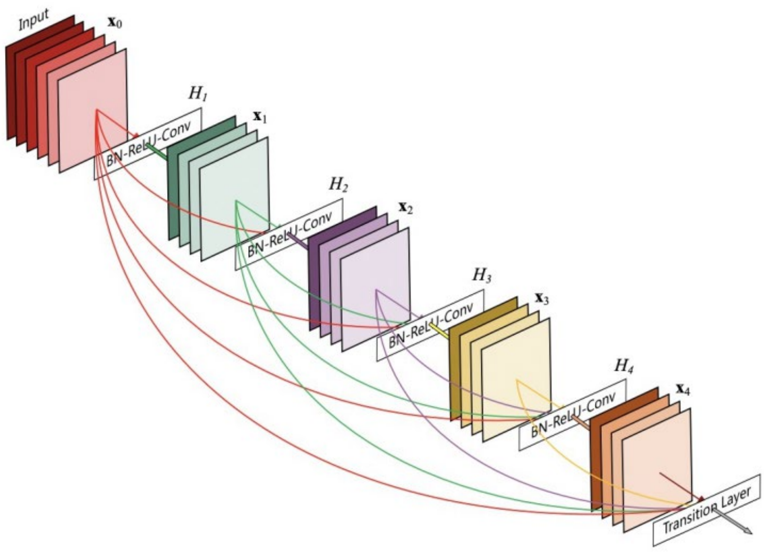



**Figure 2-16: Structure of DenseNet (DenseBlock)**

<u>General Quality of Life Updates for Developer</u>

In comparison to other object detection frameworks, YOLOv5 is incredibly simple to integrate into an application by a developer. These updates to my quality of life classified as follows. [26]

- **Simple Installation** - YOLOv5 requires simply torch and a few lightweight python modules.
- **Rapid Training** - The YOLOv5 models train exceptionally quickly, allowing you to save money on testing while building your model.
- **Working Inference Ports** - YOLOv5 supports inference on individual photos, batch images, video feeds, and webcam ports.
- **Intuitive File Folder Layout** - While working, the file folder layout is intuitive and simple to traverse.
- **Simple Translation to Mobile** - YOLOv5 can be easily translated from PyTorch weights to ONXX weights to CoreML weights and finally to iOS.

### Conclusion

YOLOv5's initial release is extremely fast, performant, and simple to use. While YOLOv5 does not provide novel model architecture enhancements to the YOLO model family, it does introduce a new PyTorch training and deployment methodology that advances the state of the art for object detectors. Additionally, YOLOv5 is extremely user-friendly and comes pre-configured for use with custom objects "out of the box."

## 2.2.8 Comparisons

While YOLOv3 is quite high and far to the left, you can know it is well. Is it possible for you to cite your own work? Guess who is going to give it a shot, this man. Oh, and we also fixed a data loading error in YOLOv2, which saved us about 2 mAP. I'm just slipping this in here to keep the layout from being thrown off.[24], [27]

**Table 2-2: Algorithms Comparison (Speed)**

| Algorithm | m/AP | ms |
|---|---|---|
| YOLOv2 | 21.6 | 25 |
| R-FCN | 29,9 | 85 |
| SSD513 | 31.2 | 125 |
| FPN FRCN | 36.2 | 172 |
| YOLOv3-329 | 28.2 | 22 |
| YOLOv3-416 | 31 | 29 |
| YOLOv3-608 | 33 | 51 |

While 63.4 mAP (mean average prediction) and 45 FPS in YOLO mode. YOLO can achieve real-time performance with similar mAP as R-CNN, Fast R-CNN, and Faster R-CNN. After we know all the algorithms and study their properties, we will do a comparison between them, which one is faster and takes more evidence.[28]

**Table 2-3 Algorithms Comparison (Performance)**

| Algorithm | mAP | FPS |
|---|---|---|
| R-CNN | 53.5 | 6 |
| Fast R-CNN | 70 | 0.5 |
| Faster R-CNN | 73.2 | 7 |
| Faster R-CNN ZF | 62.1 | 18 |
| YOLO VGG-16 | 66.4 | 21 |

### 2.2.9 Summary

We learned a light introduction to the topic of object identification in this post, as well as state-of-the-art deep learning models meant to solve it.

Also compare these algorithms to see which one is better, The phrase "object recognition" refers to a set of actions that are used to recognize objects in digital pictures.

- **HOG**, like the Canny Edge Detector and SIFT, is a feature descriptor (Scale Invariant and Feature Transform).
- **SSD**, To detect several items within an image, just one shot is required.
- **RCNN**, are a class of approaches for tackling object identification and localization problems that are optimized for model performance.
- **YOLO**, group of object identification algorithms that are geared for speed and real-time application.

Finally, we can say that the difference between the two most important methods of detecting that a group R-CNN is characterized by superior performance and a group YOLO characterized by real-time object detection

## 2.3 DATASETS

A set of data that is dealt with as a single unit by a computer. This means that a data set contains a lot of discrete pieces of data but can be used to train an algorithm with the goal of finding predictable patterns within the entire data set.

As mentioned in the previous chapter, we searched for open-source datasets and filtered them to the ones in this report. It relied heavily on the ones in GitHub, and the research focused on different types of weapons in this project, and people.

### 2.3.1 Crime Detection – using Deep learning

This project used YOLO Darknet framework. Project's Datasets is about Crime, and it is manually obtained from Google photos as well as the ImageNet database for crime detection. It has 3150 photos, txt files. It's categories: Gun, Knife, Person. [29], [30]

### 2.3.2 Weapon detection datasets

These datasets used converted YOLOV5 to practice. Project's Datasets is Primarily concerned with the development of intelligent video surveillance automatic systems. [31], It is from: Andalusian Research Institute in Data Science and Computational Intelligence (DaSCI). It has 18097 photos, txt files. It's categories: Pistol, Knife, Weapons, and similar handled objects [32]

### 2.3.3 Weapon Detection and Classification

These datasets used Deep Learning CNNs to practice. Weapon Detection & Classification through CCTV surveillance. It has 13215 photos. It's categories: Knife, Small Gun.[33]

### 2.3.4 Handgun Dataset

This dataset was used on the paper titled "Firearm Detection from Surveillance Cameras Using Image Processing and Machine Learning Techniques." It has Consists of positive (Handgun) Images and negative images (Images of various objects), in total 1900 photos. It's categories: Gun. [34], [35]

### 2.3.5 Knife Dataset

There are 400 training photos and 100 test images of knives in the dataset. It has 500 photos. It's categories: knife [36]

### 2.3.6 Summary

**Table 2-4: Summary of Datasets**

| Dataset | Gun | Rifle | Knife | People | Total |
|---|---|---|---|---|---|
| 1. Crime Detection | 2000 | - | 1050 | 100 | 3150 |
| 2. Weapon detection | 10770 | 187 | 7140 | - | 18097 |
| 3. Weapon Classification | 315 | - | 12900 | - | 13215 |
| 4. Handgun Dataset | 1900 | - | - | - | 1900 |
| 5. Knife Dataset | - | - | 500 | - | 500 |
| Total Datasets | | | | | 36862 |

Below, there is a percentage representation for each category in each dataset

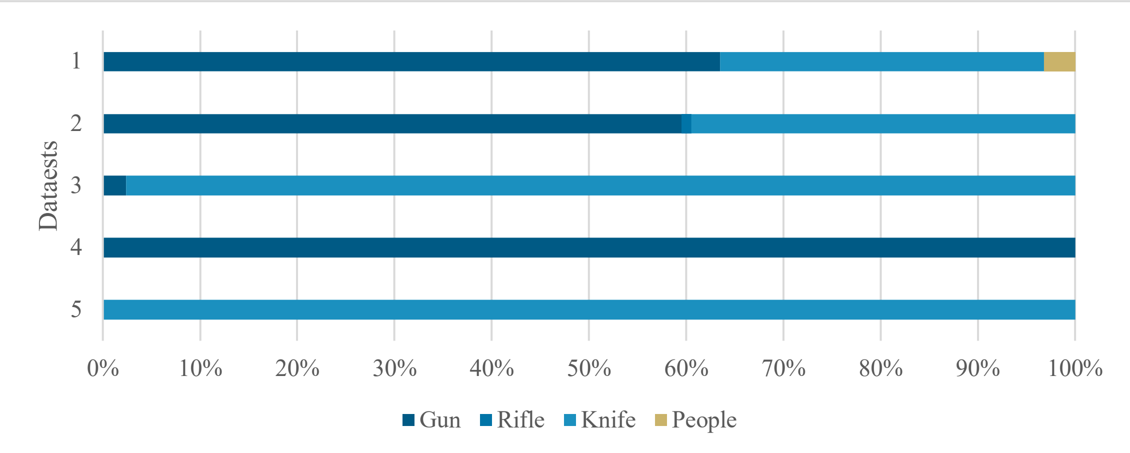



**Figure 2-17: Summary of Datasets (Data Type %)**

*Chapter There*

# 3 ANALYSIS AND DESIGN

In this chapter, the methods used in system analysis and design, as well as the user interfaces, will be explained.

That Include UML, functionality explanations, and user interfaces.

## 3.1 FUNCTIONAL REQUIREMENTS

Main Function that we implemented to enable users to accomplish their task

**Table 3-1: Functional Requirements**

| | |
|---|---|
| **1. Login** | To allow the user to enter the system and browse the camera, a login must be made to the system |
| **2. Create account** | If the user is new, you must create an account for the system to allow that to log in to the system |
| **3. Identify a person holding gun** | One of the most crucial functions of the system is how to identify any person carrying a gun by taking a picture of him and determining the type of detection |
| **4. Identify a person holding knives** | In many cases, the criminal is carrying our knife and passing through many monitoring devices, so it is a function in the system to identify anyone who carries a knife for the safety of customers. |
| **5. Identify a person holding Trifle** | The type of detection differs, and one of the most important types of guns is that one of the functions of the system is to identify the Trifle, if it is found with the criminal, and show it to the user. |
| **6. Create an alarm if model knows that there is something that triggers an alarm** | When the system recognizes the detection, it must send a warning to inform the users of the presence of danger and to take the necessary measures also to inform the customers to immunize themselves |
| **7. Display camera stream** | The camera content of the system must be displayed to determine the detection of the user who is following the system in action |
| **8. Logout** | The user must be logged out for the safety of the system if he finished his work or if he will leave the place |

## 3.2 NON-FUNCTIONAL REQUIREMENTS

Functions that we implemented to define the fundamental behavior of the system

**Table 3-2: Non-Functional Requirements**

| | |
|---|---|
| **1. System security** | Protect data from external attacks |
| **2. Visibility all 24⁄7** | Live information that allows us to monitor the possible danger and their exact location at any given point. |
| **3. Accuracy in recognizing object detection** | Full accuracy in taking out the image or detection clearly |
| **4. Speed in recognizing object detection** | One of the most key features that must be in the system is that it can quickly detect images to eliminate the danger |
| **5. Preserve the privacy of the place** | The system must preserve its data and not allow anyone who is not responsible for it or those who have the right to see the system |

## 3.3 USE-CASE DESCRIPTIONS

a text-based narrative of a functionality comprised of detailed, step-by-step interaction according to each function

### 3.3.1 Login

To access the system and use it

**Table 3-3: Use-Case Description (Login)**

| | |
|---|---|
| **Job name** | Login |
| **Related Requirements** | Admin, User |
| **Initiating Actor** | Admin, User |
| **Actor's Goal** | To login and use the model |
| **Participating Actors** | None |
| **Preconditions** | None |
| **Postconditions** | User is logged in |

| | |
|---|---|
| **Flow of Events for Main Success Scenario** | 1. User enters the username, password and clicks login.<br>2. system authenticates the sent info, after the authentication is successful, system lets the user login. |
| **Flow of Events for Alternate Scenario** | 1. User enters the username, password and clicks Login.<br>2. system authenticates the sent info, after the authentication is false, system prompts the user (username or password is incorrect). |

### 3.3.2 Add User

Admin can add new employees to access the system

**Table 3-4: Use-Case Description (Add User)**

| | |
|---|---|
| **Job name** | Login |
| **Related Requirements** | Admin |
| **Initiating Actor** | Admin |
| **Actor's Goal** | To Register the User and use the model |
| **Participating Actors** | None |
| **Preconditions** | None |
| **Postconditions** | Be logged in |
| **Flow of Events for Main Success Scenario** | 1. Admin clicks register.<br>2. System returns registration page.<br>3. Admin fills in required information, then clicks create.<br>4. System prompts user "account creation was successful". |
| **Flow of Events for Alternate Scenario** | 1. User enters the username, password and clicks Login.<br>2. System authenticates the sent info, after the authentication is false, system prompts the user (username or password is incorrect). |

### 3.3.3 View Camera

The ability for the user to display the content on the monitor

**Table 3-5: Use-Case Description (View Camera)**

| | |
|---|---|
| **Job name** | View Camera |
| **Related Requirements** | Admin, User |
| **Initiating Actor** | Admin, User |
| **Actor's Goal** | To see the Camera's |
| **Participating Actors** | None |
| **Preconditions** | None |
| **Postconditions** | None |
| **Flow of Events for Main Success Scenario** | 1. User clicks View Camera<br>2. System returns the display of the camera's |
| **Flow of Events for Alternate Scenario** | None |

### 3.3.4 Use Model

The system can access the cameras and use the mode

**Table 3-6: Use-Case Description (Use Model)**

| | |
|---|---|
| **Job name** | Use Model |
| **Related Requirements** | System |
| **Initiating Actor** | System |
| **Actor's Goal** | To use the model |
| **Participating Actors** | None |
| **Preconditions** | None |
| **Postconditions** | System using the model |

| | |
|---|---|
| **Flow of Events for Main Success Scenario** | 1. System use the model<br>2. Determine object's type<br>3. Detect the object's<br>4. If the object's is a weapon, issue an alert |
| **Flow of Events for Alternate Scenario** | 1-3. ------------------------------------------------------------<br>4. If the object's is not a weapon, do not issue an alert |

### 3.3.5 Determine Object's Type

Analysing the video stream to decide the type of the object

**Table 3-7: Use-Case Description (Determine Object's Type)**

| | |
|---|---|
| **Job name** | Determine Object's Type |
| **Related Requirements** | System |
| **Initiating Actor** | System |
| **Actor's Goal** | To analyze and classify the objects |
| **Participating Actors** | None |
| **Preconditions** | Use model |
| **Postconditions** | Object detection |
| **Flow of Events for Main Success Scenario** | 1. System use the model<br>2. Use camera to view<br>3. Analyze to determine object |
| **Flow of Events for Alternate Scenario** | None |

### 3.3.6 Object Detection

Analysing the video stream to detect each object

**Table 3-8: Use-Case Description (Object Detection)**

| | |
|---|---|
| **Job name** | Object Detection |
| **Related Requirements** | System |

| | |
|---|---|
| **Initiating Actor** | System |
| **Actor's Goal** | To detect objects |
| **Participating Actors** | None |
| **Preconditions** | None |
| **Postconditions** | Detect object (weapon) |
| **Flow of Events for Main Success Scenario** | 1. System use the model<br>2. Use the camera's<br>3. Determine objects<br>4. Detect the objects |
| **Flow of Events for Alternate Scenario** | None |

### 3.3.7 Issuing an alert

When weapon is detected, the system will announce an alarm to alert the surrounding

**Table 3-9: Use-Case Description (Issuing an alert)**

| | |
|---|---|
| **Job name** | Issuing an alert |
| **Related Requirements** | System |
| **Initiating Actor** | System |
| **Actor's Goal** | To make an alert if the system detect weapons |
| **Participating Actors** | None |
| **Preconditions** | None |
| **Postconditions** | Alert the users only if weapon detected |
| **Flow of Events for Main Success Scenario** | 1. System use the model<br>2. Determine object's type<br>3. Detect the object's<br>4. If the object's is a weapon, issue an alert |
| **Flow of Events for Alternate Scenario** | 1-3. ----------------------------------------------------------<br>4. If the object's is not a weapon, do not issue an alert |

### 3.3.8 Logout

When the employee using the system is done with his work, logout so others can login

**Table 3-10: Use-Case Description (Logout)**

| | |
|---|---|
| **Job name** | Logout |
| **Related Requirements** | Admin, User |
| **Initiating Actor** | Admin, User |
| **Actor's Goal** | To logout of the model |
| **Participating Actors** | Admin, User |
| **Preconditions** | None |
| **Postconditions** | User logged out |
| **Flow of Events for Main Success Scenario** | 1. User clicks log out.<br>2. system authenticates lets the user log out |
| **Flow of Events for Alternate Scenario** | None |

## 3.4 GRAPHIPHS USER INTERFASES

a form of user interface that allows users to interact with electronic devices through graphical icons and audio indicator such as primary notation.

We design a GUI's according to the main two users, with United Login Interface

- **Admin**
  - Main page
  - Add User
  - View Camera
- **Users**
  - Main page
  - View Camera

### 3.4.1 Login

The login page requires a valid username and a password

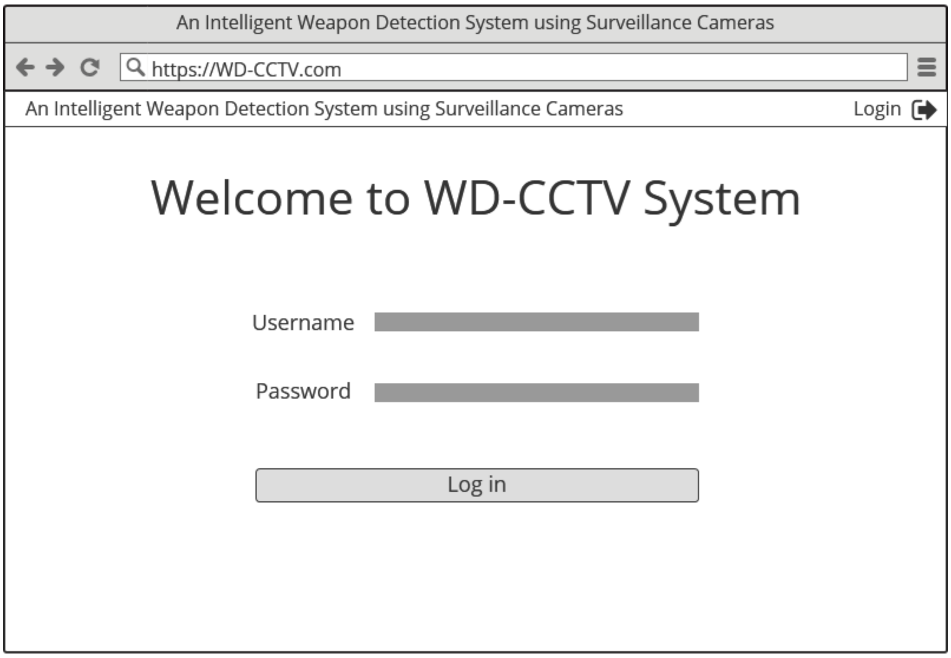



**Figure 3-1: GUI (Login)**

### 3.4.2 Main page (User)

the main page will show up after a success login, displaying the options for the (User)

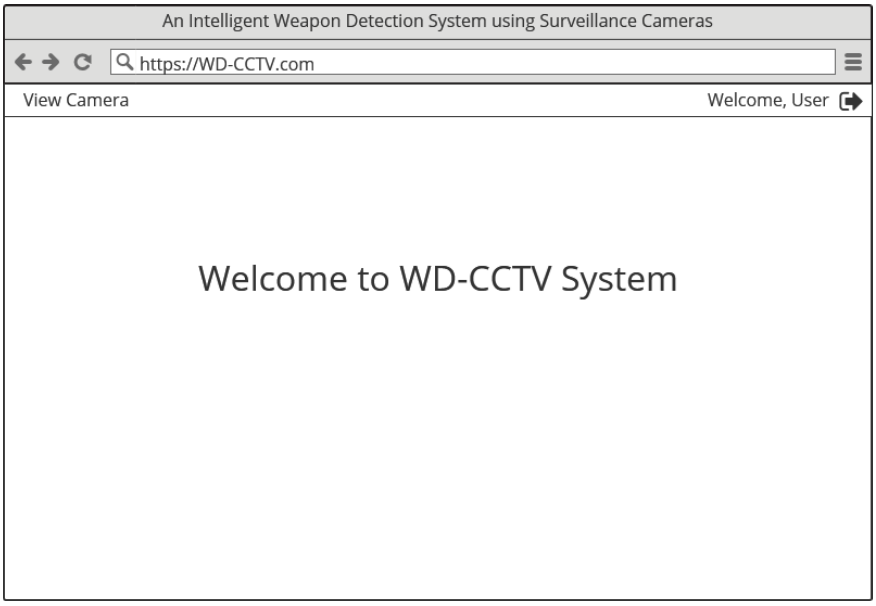



**Figure 3-2: GUI (Main page [User])**

### 3.4.3 View Camera (User)

let the user access to the cameras and display the content on the monitor


**Figure 3-3: GUI [View Camera (User)]**

### 3.4.4 Main page (Admin)

the main page will show up after a success login, displaying the options for the (Admin)

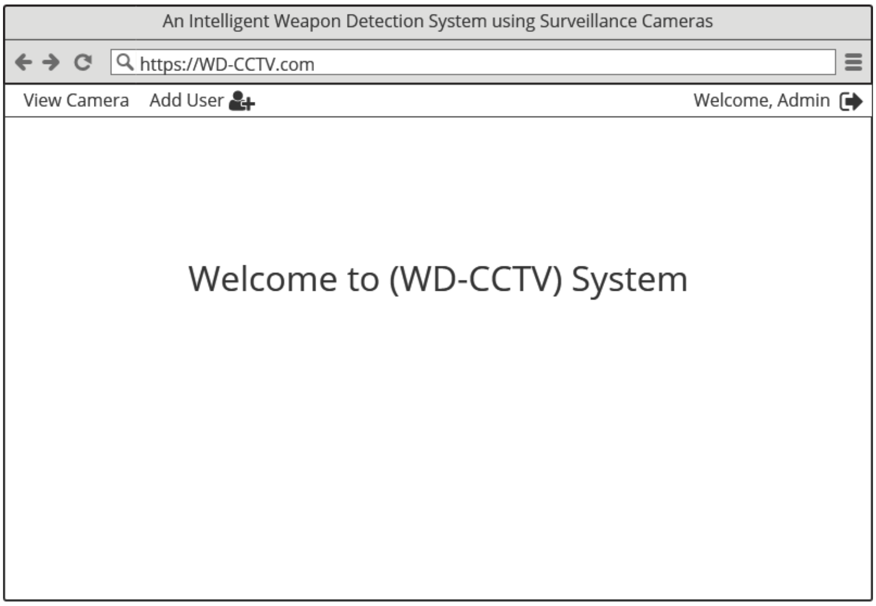



**Figure 3-4: Main page (Admin)**

### 3.4.5 Add User

the main page will show up after a success login, displaying the options for the (Admin)


**Figure 3-5: GUI (Add User)**

### 3.4.6 View Camera (Admin)

to let the user access to the cameras and display the content on the monitor


**Figure 3-6: [View Camera (Admin)]**

## 3.5 UNIFIED MODELING LANGUAGE (UML) DIAGRAMS

in software engineering field that is intended to provide a standard way to visualize the design of a system.

We choose 3 main kinds of UML Diagrams

- **Entity Relationship (ER)**
  The main and prime diagram of database.
- **Use-Case**
  The main diagram of project's actors and functions relationship.
- **Activity**
  The diagram of project's way of prosses.

### 3.5.1 Entity Relationship (ER)

#### Chen ERD

This graphic describes the relationships between an entity and how the system in the database operates

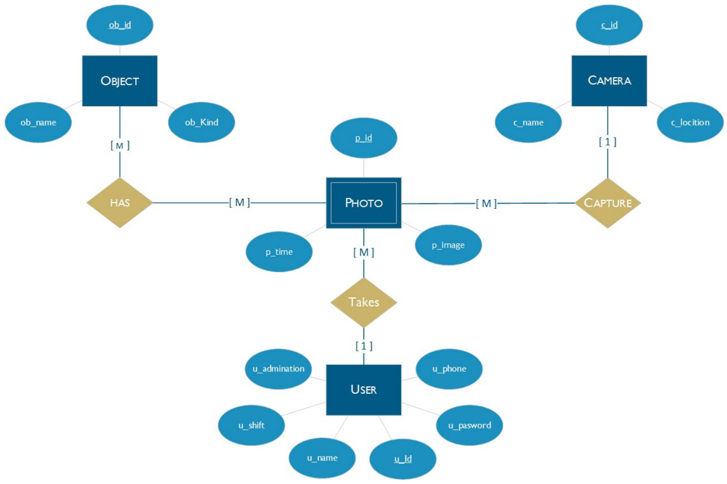



**Figure 3-7: Chen ERD**

#### ERD Schema

As for here, it is more clarification about the relationships in the database and its internal types

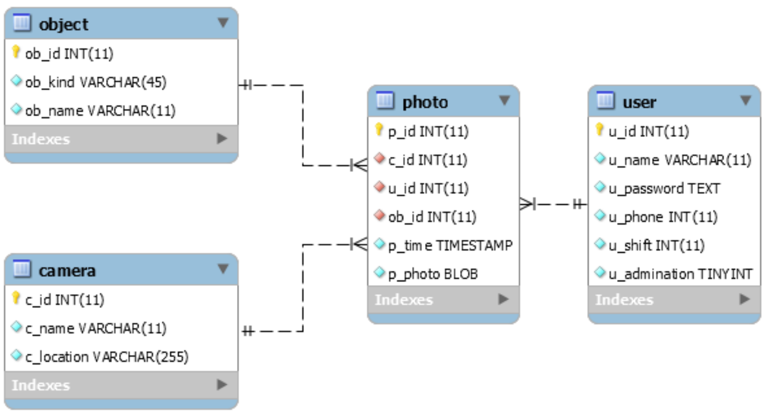



**Figure 3-8: ERD Schema**

### 3.5.2 Use-Case

The Use-Case describes the most important actors in the system and how the system works

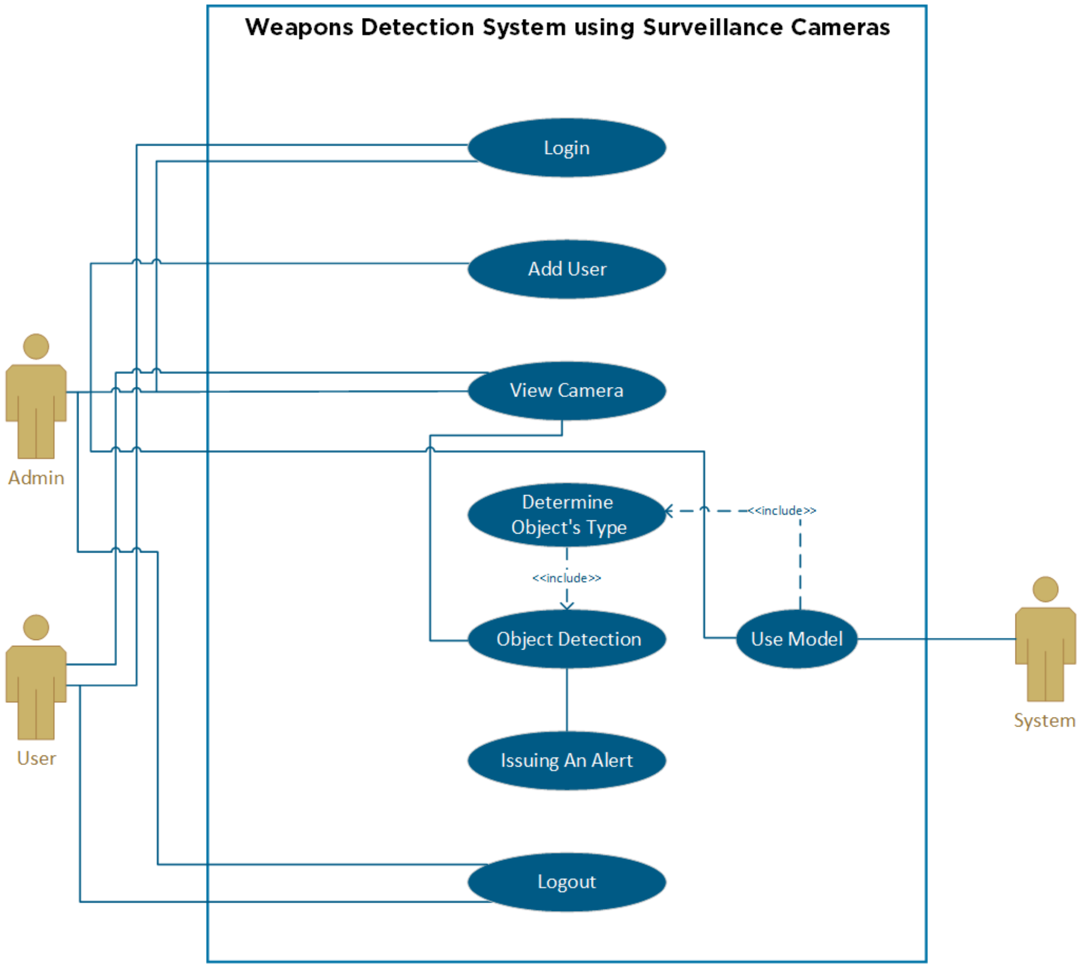



**Figure 3-9: UML Use-Case**

### 3.5.3 Activity

A graphical diagram to show the process of the how the model work in the System

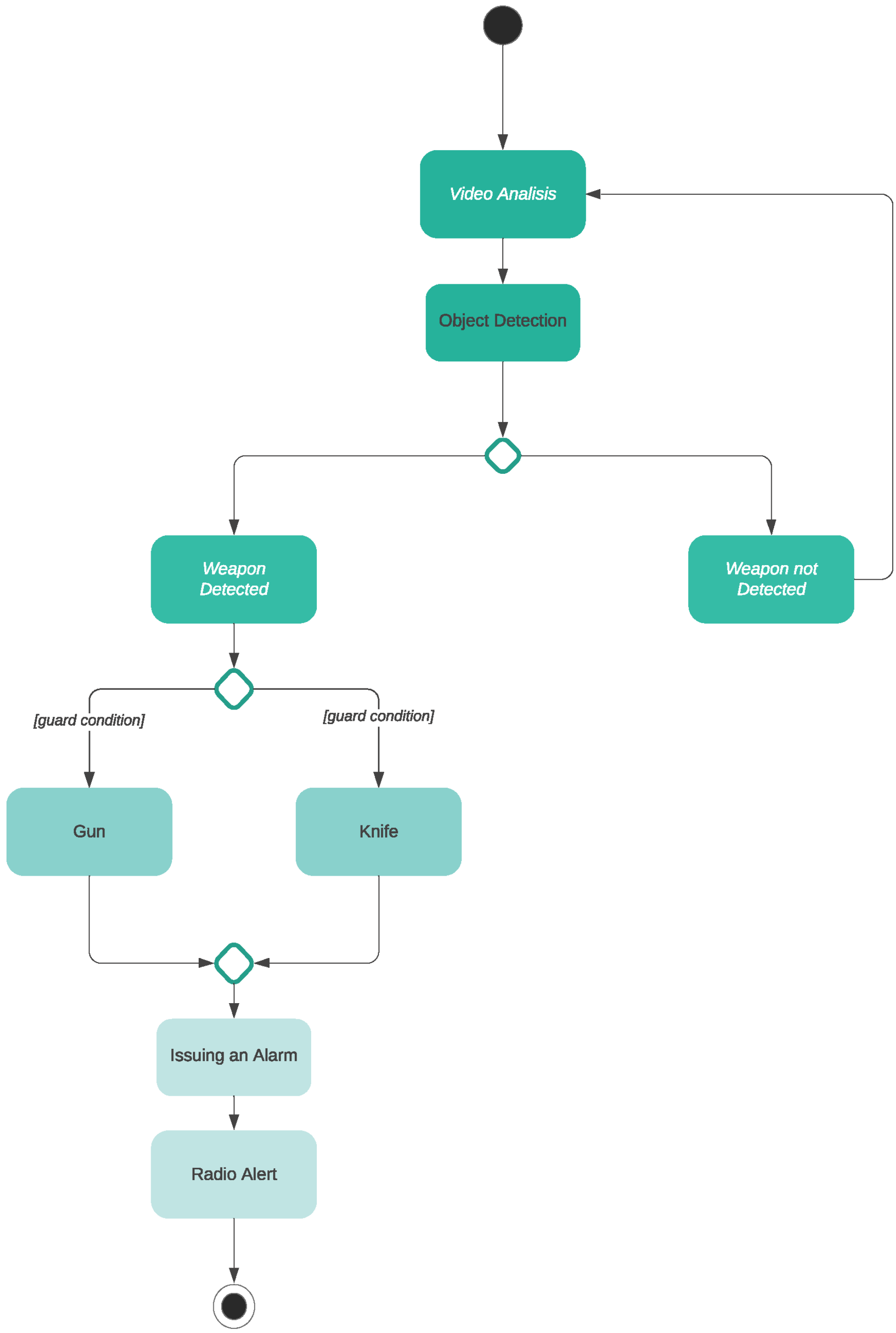



**Figure 3-10: UML Activity**

# REFERENCES


[1] D. Cao, Z. Chen, and L. Gao, “An improved object detection algorithm based on multi-scaled and deformable convolutional neural networks,” *Hum.-Centric Comput. Inf. Sci.*, vol. 10, no. 1, p. 14, Apr. 2020, doi: 10.1186/s13673-020-00219-9.

[2] O. Russakovsky *et al.*, “ImageNet Large Scale Visual Recognition Challenge,” *ArXiv14090575 Cs*, Jan. 2015, Accessed: Oct. 10, 2021. [Online]. Available: http://arxiv.org/abs/1409.0575

[3] J. Brownlee, “A Gentle Introduction to Object Recognition With Deep Learning,” *Machine Learning Mastery*, May 21, 2019. https://machinelearningmastery.com/object-recognition-with-deep-learning/ (accessed Oct. 10, 2021).

[4] V. Zhou, “Machine Learning for Beginners: An Introduction to Neural Networks,” *Medium*, Dec. 20, 2019. https://towardsdatascience.com/machine-learning-for-beginners-an-introduction-to-neural-networks-d49f22d238f9 (accessed Dec. 23, 2021).

[5] “AI vs. Machine Learning vs. Deep Learning vs. Neural Networks: What’s the Difference?,” Nov. 23, 2021. https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks (accessed Dec. 23, 2021).

[6] “What are Neural Networks?,” Aug. 03, 2021. https://www.ibm.com/cloud/learn/neural-networks (accessed Dec. 23, 2021).

[7] K. Mittal, “A Gentle Introduction Into The Histogram Of Oriented Gradients,” *Analytics Vidhya*, Dec. 21, 2020. https://medium.com/analytics-vidhya/a-gentle-introduction-into-the-histogram-of-oriented-gradients-fdee9ed8f2aa (accessed Oct. 10, 2021).

[8] “Histogram of Oriented Gradients and Object Detection,” *PyImageSearch*, Nov. 10, 2014. https://www.pyimagesearch.com/2014/11/10/histogram-oriented-gradients-object-detection/ (accessed Oct. 10, 2021).

[9] “Single Shot Detection (SSD) Algorithm,” *OpenGenus IQ: Computing Expertise & Legacy*, Jan. 21, 2019. https://iq.opengenus.org/single-shot-detection-ssd-algorithm/ (accessed Oct. 10, 2021).

[10] “13.7. Single Shot Multibox Detection — Dive into Deep Learning 0.17.0 documentation.” https://d2l.ai/chapter_computer-vision/ssd.html (accessed Oct. 10, 2021).

[11] A. Choudhury, “Top 8 Algorithms For Object Detection,” *Analytics India Magazine*, Jun. 16, 2020. https://analyticsindiamag.com/top-8-algorithms-for-object-detection/ (accessed Oct. 10, 2021).

[12] S. Majumder, “Object Detection Algorithms-R CNN vs Fast-R CNN vs Faster-R CNN,” *Analytics Vidhya*, Jul. 04, 2020. https://medium.com/analytics-vidhya/object-detection-algorithms-r-cnn-vs-fast-r-cnn-vs-faster-r-cnn-3a7bbaad2c4a (accessed Oct. 10, 2021).

[13] “R-CNN vs Fast R-CNN vs Faster R-CNN | ML,” *GeeksforGeeks*, Feb. 28, 2020. https://www.geeksforgeeks.org/r-cnn-vs-fast-r-cnn-vs-faster-r-cnn-ml/ (accessed Oct. 10, 2021).

[14] R. Gandhi, "R-CNN, Fast R-CNN, Faster R-CNN, YOLO — Object Detection Algorithms," *Medium*, Jul. 09, 2018. https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e (accessed Oct. 10, 2021).
[15] "Object Detection / Fast R-CNN - Convolutional Neural Networks for Image and Video Processing - TUM Wiki." https://wiki.tum.de/pages/viewpage.action?pageId=22578448#ObjectDetection/FastRCNN-Weblinks (accessed Oct. 11, 2021).
[16] R. Girshick, "Fast R-CNN," *ArXiv150408083 Cs*, Sep. 2015, Accessed: Oct. 11, 2021. [Online]. Available: http://arxiv.org/abs/1504.08083
[17] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *ArXiv150601497 Cs*, Jan. 2016, Accessed: Oct. 11, 2021. [Online]. Available: http://arxiv.org/abs/1506.01497
[18] R. Girshick, *py-faster-rcnn has been deprecated. Please see Detectron, which includes an implementation of Mask R-CNN.* 2021. Accessed: Oct. 11, 2021. [Online]. Available: https://github.com/rbgirshick/py-faster-rcnn
[19] "Object Detection / Fast R-CNN - Convolutional Neural Networks for Image and Video Processing - TUM Wiki." https://wiki.tum.de/pages/viewpage.action?pageId=22578448#ObjectDetection/FastRCNN-Weblinks (accessed Oct. 11, 2021).
[20] A. Singh, "YOLO: You Only Look Once," *Nerd For Tech*, May 29, 2021. https://medium.com/nerd-for-tech/yolo-you-only-look-once-65ea86104c51 (accessed Oct. 11, 2021).
[21] "Papers with Code - You Only Look Once: Unified, Real-Time Object Detection." https://paperswithcode.com/paper/you-only-look-once-unified-real-time-object (accessed Oct. 11, 2021).
[22] "Publications." https://pjreddie.com/publications/ (accessed Oct. 11, 2021).
[23] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," Apr. 2018, Accessed: Oct. 11, 2021. [Online]. Available: https://arxiv.org/abs/1804.02767v1
[24] "YOLO: Real-Time Object Detection." https://pjreddie.com/darknet/yolo/ (accessed Oct. 12, 2021).
[25] "How to Train A Custom Object Detection Model with YOLO v5 | by Jacob Solawetz | Towards Data Science." https://towardsdatascience.com/how-to-train-a-custom-object-detection-model-with-yolo-v5-917e9ce13208 (accessed Dec. 24, 2021).
[26] "YOLOv5 v6.0 is here - new Nano model at 1666 FPS." https://blog.roboflow.com/yolov5-v6-0-is-here/ (accessed Dec. 24, 2021).
[27] "Papers with Code - PASCAL VOC 2007 Benchmark (Object Detection)." https://paperswithcode.com/sota/object-detection-on-pascal-voc-2007 (accessed Oct. 11, 2021).
[28] J. Du, "Understanding of Object Detection Based on CNN Family and YOLO," *J. Phys. Conf. Ser.*, vol. 1004, p. 012029, Apr. 2018, doi: 10.1088/1742-6596/1004/1/012029.
[29] "Crime-Detection--using-Deep-learning/data at master · sowmyadvn/Crime-Detection--using-Deep-learning," *GitHub*. https://github.com/sowmyadvn/Crime-Detection--using-Deep-learning (accessed Sep. 28, 2021).

[30] "GitHub - sowmyadvn/Crime-Detection--using-Deep-learning: Developed an object detection model using YOLO Darknet framework to prevent crimes. Collected the crime data manually from Google images as well as Imagenet database." https://github.com/sowmyadvn/Crime-Detection--using-Deep-learning (accessed Sep. 28, 2021).

[31] *Weapon detection datasets*. ARI-DaSCI, 2021. Accessed: Sep. 28, 2021. [Online]. Available: https://github.com/ari-dasci/OD-WeaponDetection

[32] "Weapon Detection – DaSCI." https://dasci.es/transferencia/open-data/24705/ (accessed Sep. 28, 2021).

[33] "Weapon-Detection-And-Classification/train at master · ivaibhavkr/Weapon-Detection-And-Classification," *GitHub*. https://github.com/ivaibhavkr/Weapon-Detection-And-Classification (accessed Sep. 28, 2021).

[34] F. Gelana and A. Yadav, "Firearm Detection from Surveillance Cameras Using Image Processing and Machine Learning Techniques," in *Smart Innovations in Communication and Computational Sciences*, Singapore, 2019, pp. 25–34. doi: 10.1007/978-981-13-2414-7_3.

[35] F. Gelana, *Handgun-Dataset*. 2021. Accessed: Sep. 28, 2021. [Online]. Available: https://github.com/FrexG/Handgun-Dataset

[36] "Knife Dataset | Kaggle." https://www.kaggle.com/shank885/knife-dataset (accessed Sep. 28, 2021).