Faculty of Computer Science and Information Technology

The GRADUATION PROJECT

# Smart Traffic Light Using AI and Image Processing

**Submitted by:**

| # | ID | Name |
|---|---|---|
| 1 | 41910239 | Mohamed Sobhi |
| 2 | 41910266 | Abdel-Rahman Tarek |
| 3 | 41910123 | Omar Yasser |
| 4 | 41910102 | Dina Ehab |
| 5 | 41910258 | Amr Tarek |
| 6 | 41910338 | Aly Mashhour |
| 7 | 41920027 | Noura Tarek |

**A dissertation submitted in partial fulfillment of the requirements for the degree of Bachelor of computer science and information technology.**

**Supervised by:**

**Dr. Sherif Eletriby**

ACADEMIC YEAR 2022 - 2023

## Committee Report

We certify we have read the graduation project report as an examining committee, examine the student in its content, and that in our opinion it's adequate as a project document for

**"Smart Traffic Light Using AI and Image Processing"**

**Chairman:**
**Name:**
**Signature:**
**Date:** / / 2023

**Examiner:**
**Name:**
**Signature:**
**Date:** / / 2023

**Supervisor:**
**Name:**
**Signature:**
**Date:** / / 2023

## Intellectual Property Right Declaration

This is to declare that the work under the supervision of **Dr. Sherif Eletriby** having title " **Smart Traffic Light Using AI and Image Processing**" carried out in partial fulfillment of the requirements of Bachelor of Science in Computer Science is the sole property of Ahram Canadian University and the respective supervisor. It is protected under the intellectual property right laws and conventions. It can only be considered/ used for purposes like extension for further enhancement, product development, adoption for commercial/organizational usage, etc. with the permission of the University and respective supervisor. This above statement applies to all students and faculty members.

**Submitted By:**

| Name | ID |
|---|---|
| Mohamed Sobhi | 41910239 |
| Abdel-Rahman Tarek | 41910266 |
| Omar Yasser | 41910123 |
| Amr Tarek | 41910258 |
| Dina Ehab | 41910102 |
| Aly Mashhour | 41910338 |
| Noura Tarek | 41920027 |

**Supervised By:**

Dr. Sherif Eletriby

## Anti-Plagiarism Declaration

This is to declare that the above publication produced under the supervision of **Dr. Sherif Eletriby** having title "**Smart Traffic Light Using AI and Image Processing**" is the sole contribution of the author(s) and no part hereof has been reproduced illegally (cut and paste) which can be considered as Plagiarism. All referenced parts have been used to argue the idea and have been cited properly. We will be responsible and liable for any consequence if violation of this declaration is proven.

**Submitted By:**

| Name | ID |
|---|---|
| Mohamed Sobhi | 41910239 |
| Abdel-Rahman Tarek | 41910266 |
| Omar Yasser | 41910123 |
| Amr Tarek | 41910258 |
| Dina Ehab | 41910102 |
| Aly Mashhour | 41910338 |
| Noura Tarek | 41920027 |

**Supervised By:**

Dr. Sherif Eletriby

## Acknowledgement

We would like to express this sincere gratitude to several individuals and organizations for supporting us throughout this Graduate study.

First, we wish to express this sincere gratitude to this supervisor, **Dr. Sherif Eletriby**, for his enthusiasm, patience, insightful comments, helpful information, practical advice, and unceasing ideas that have always helped us tremendously in this research and writing of this thesis.

We would also like to thank every faculty member at Al-Ahram Canadian University for teaching and guiding us throughout all these years during which their tremendous knowledge and deep and professional experience enabled us to learn many things that helped us learn machine learning and deep learning through which we completed a Bachelor of Science thesis in software successfully.

Without their support and guidance, this project would not have been possible.

# Table of Contents

## TABLE OF FIGURES

# CHAPTER 1

# INTRODUCTION

# CHAPTER 1 Introduction

## 1.1 Overview

Traffic lights consist normally of three signals, transmitting meaningful information to drivers and riders through colors and symbols including arrows and vehicles. The regular traffic light colors are red, yellow, and green arranged vertically or horizontally in that order. Although this is internationally standardized, variations exist on national and local scales as to traffic light sequences and laws. The method was first introduced in December 1868 on Parliament Square in London to reduce the need for police officers to control traffic. Since then, electricity and computerized control has advanced traffic light technology and increased intersection capacity.

The system is also used for other purposes, for example, to control pedestrian movements, variable lane control (such as tidal flow systems or smart motorways), and railway level crossings.

### Our Project is Called

"Smart Traffic Light Using AI and Image Processing"

It uses techniques such as machine learning, deep learning, neural networks and image processing to detect the type of vehicle and its density of each lane.

## 1.2 Motivation

-Think about the times you got stuck in traffic, and how they would have been even worse if there were no management systems to control these congested roads

-There are many overarching reasons why traffic lights are popping up all over our roads and neighborhoods. While some signals may serve to fill a very special niche, most signals share a common set of purposes.

-Maintain a safe flow of traffic: From managing travel times to maintaining safe roads, the majority of traffic lights are designed to keep cars, pedestrians, cyclists, and anyone else using the road safe. These signals may be anything from warning signals to street lights, from complex to simple, all in order to keep people in their own lanes. The idea here is to reduce the speed and volume of traffic to safe levels by speeding up or slowing down traffic along the stretch of road in question.

-Reducing the frequency and severity of accidents: Finally, traffic lights also play a role in reducing the amount and severity of accidents that occur. People are not infallible, so accidents happen even in some of the safest environments. In such situations, limiting speeds or designing a series of signals to slow or stop traffic can at least reduce the impact of accidents that occur. In practice, this usually looks like speed limits or restructuring of lanes and key signs in heavy traffic and accident-prone areas.

## 1.3 Objective

The Objective of artificial intelligence is not to remove the human factor from work, but to develop and make the largest difficulties to overcome has been making it possible for an AI program or a “robot” to do more than one task. It is very easy to program a system to complete a certain task.

From here we started to solve a problem that faces most of us every day and our goal is to solve it.

The objective behind our project is to limit the stoppage time and also regulate the traffic flow by means of the introduction of the sensors at all major traffic signals.

The proposal aims at reducing the traffic jams in order to reduce traffic congestion, optimize traffic flow and help proactively manage traffic conditions.

## 1.4 Aim

Our application aims to achieve many goals, but the most important of them are:

- To develop AI programs that can complete more and more complex tasks, helps people to save their time and to save our environment from pollution.
- Supervision and regulation of traffic on public roads and this leads us to several benefits:
    1. Reducing the time spent in roads and traffic light.
    2. Reduce traffic congestion.
    3. Reduce the pollution that affect the environment.
    4. Adapt a model that can learn from different situations.
    5. Help emergency forces to reach their destination on time

## 1.5 Scope

The general goal is to create a smart traffic light using artificial intelligence to reduce traffic congestion and preserve environment from pollution, reduce time spent in roads in traffic lights and Help emergency forces to reach their destinations on time. Using a camera and a sensor to count average cars, and on the basis of that, the condition of the traffic light changes.

## 1.6 General constraints

The proposed system is aimed at:

- Drivers who find it difficult to drive through roads due to high traffic, where it focuses on making it easier for them by decreasing the congestions, and possibly decreasing the occurrence of traffic altogether through better traffic control.
- Pedestrians who find it dangerous and risky to cross traffic roads.
- Office workers who wish to get to their work on time and waste no more time in transportations.
- Rescue staff trying to save their patients as soon as possible.

## 1.7 Proposed architecture

- **Project Lifecycle” Flow Chart”**

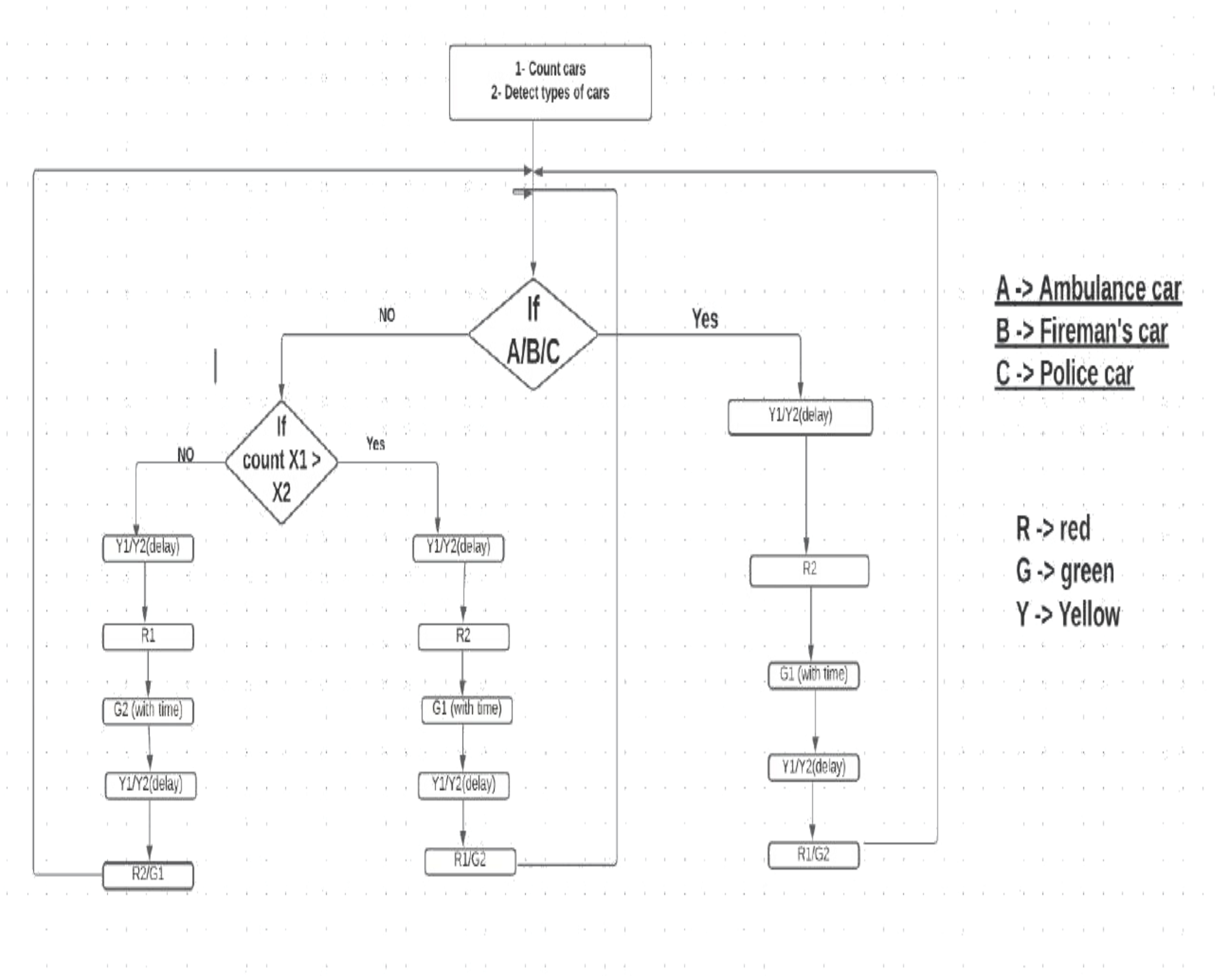


FIGURE 1 (LIFECYCLE FLOW CHART OF THE PROPOSED SYSTEM)

- <u>**Project architecture:**</u>

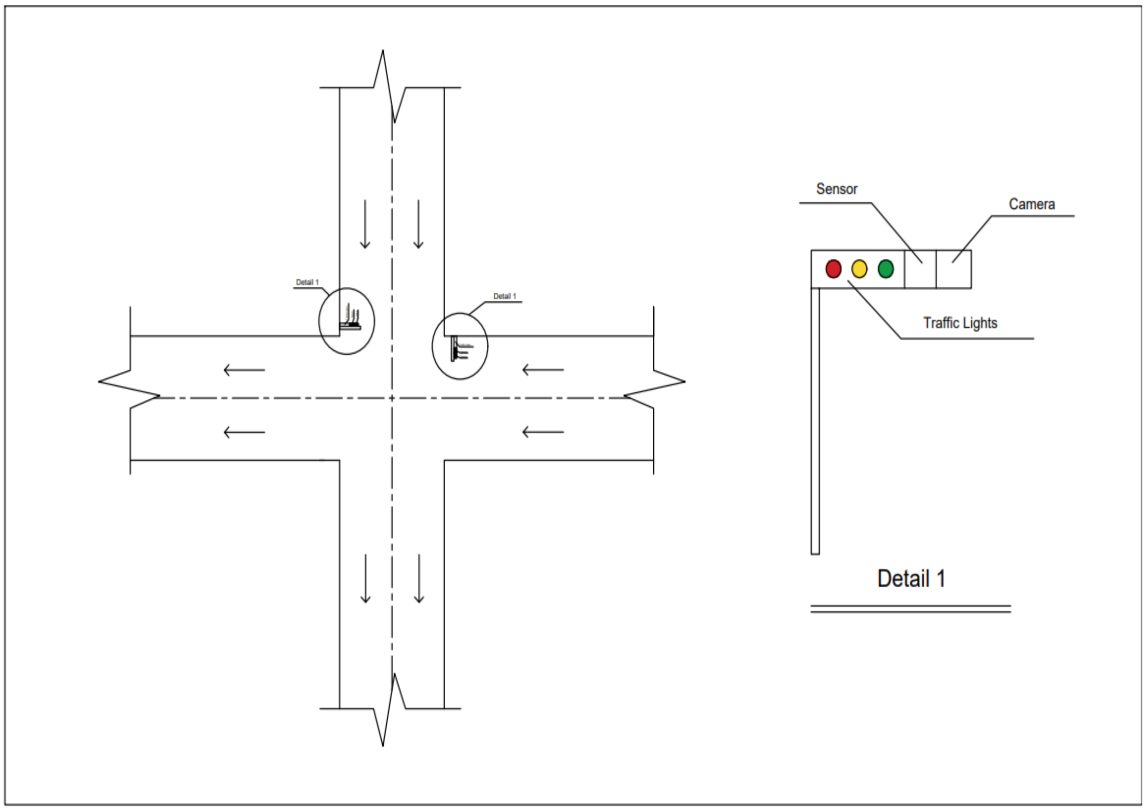


FIGURE 2 (ARCHITECTURE OF THE PROPOSED SYSTEM)

## 1.8 Organization of the dissertation

This thesis goes about the project in the following sequence:

**Part I**

- **Chapter 1:** First, we go through an Overview, and the Motivation behind our project. Next, we explain the Scope of the project, and what it entails, as well as our Objectives.
- **Chapter 2:** We give a quick Background and Literature Review discussing the previous studies published around the topic of this project.

**Part II**

- **Chapter 3:** Secondly, we go through the Analysis stage, explaining all about the current system, its drawbacks, and how the proposed system covers those shortcomings. We also go around the hardware, and software Requirements of our project.

**Part III**

- **Chapter 4:** In this chapter, we explain the Design of our project: Its components, their roles, and how they are all supposed to work together.
- **Chapter 5:** Details regarding the Implementation of the proposed system are discussed**.**
- **Chapter 6:** Finally, we discuss the project's Simulation stage and how well its Evaluation Results were based on different performance metrics.

# CHAPTER 2

## BACKGROUND AND PREVIOUS WORK

# CHAPTER 2 Background and previous work

## 2.1 Background

### What is Artificial Intelligence (AI)?

- Artificial Intelligence is the system's ability to correctly interpret external data, to learn from such data, and to use those learnings to achieve specific goals and tasks through flexible adaptation.
- AI (Artificial intelligence) is a branch of computer science in which machines are programmed and given a cognitive ability to think and mimic actions like humans and animals. The benchmark for AI is human intelligence regarding reasoning, speech, learning, vision, and problem solving, which is far off in the future.

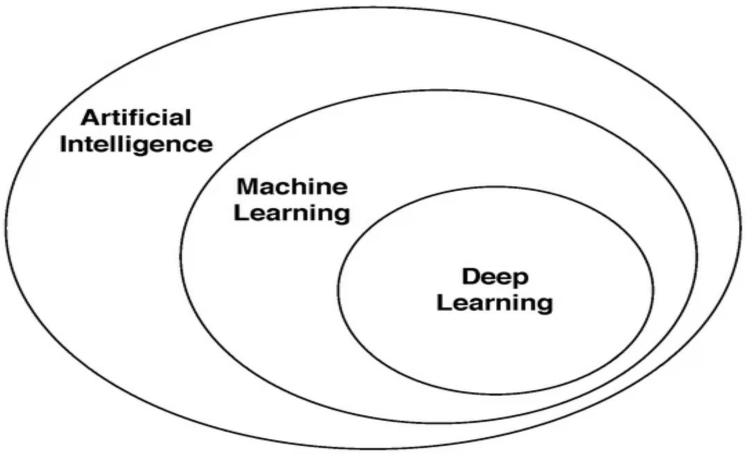


FIGURE3 (ARTIFICIAL INTELLIGENCE)

### What is Machine Learning?

-Machine Learning is a subset of artificial intelligence focusing on a specific goal: setting computers up to be able to perform tasks without the need for explicit programming.

-Machine learning: is a tool that allows systems the ability to learn and improve automatically based upon experience

## Machine Learning Techniques

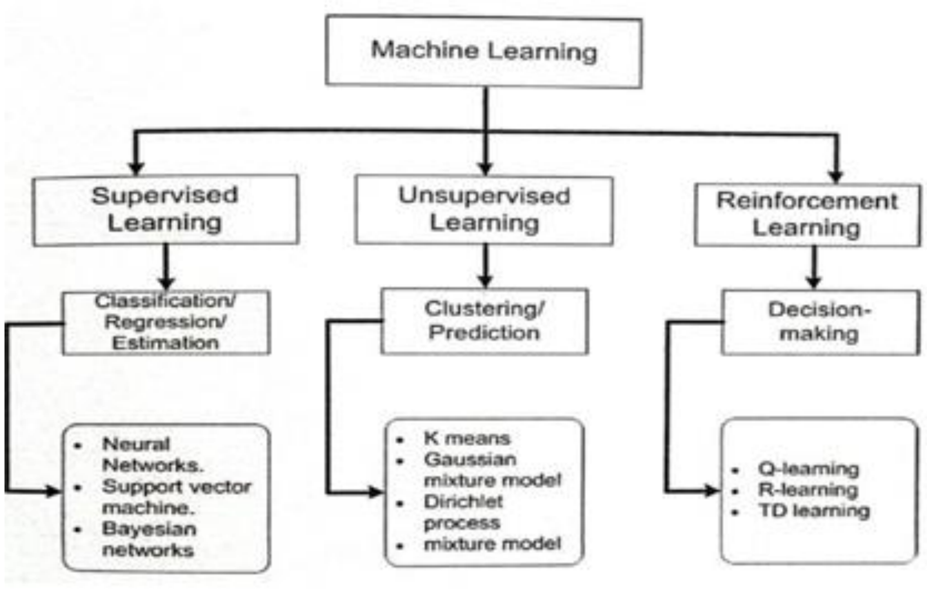


FIGURE4 (MACHINE LEARNING TECHNIQUES)

## What is Computer Vision?

Computer vision is a field of artificial intelligence (AI) that enables computers and systems to derive meaningful information from digital images, videos and other visual inputs — and take actions or make recommendations based on that information. If AI enables computers to think, computer vision enables them to see, observe and understand.

Computer vision trains machines to perform these functions, but it has to do it in much less time with cameras, data and algorithms rather than retinas, optic nerves and a visual cortex.

## How Does Computer Vision Work?

-Computer vision works much the same as human vision, human sight has the advantage of lifetimes of context to train how to tell objects apart.

-Computer vision needs lots of data.

-One essential technologies are used to accomplish this: called deep learning

-Then recognizing or seeing images in a way similar to humans.

-It runs analyses of data over and over until it discerns distinctions and ultimately recognize images.

## Object Recognition

Recognizing 3D objects from a single 2D image is one of the most challenging problems in computer vision. Computer needs to extract a set of features from the images to produce descriptions of the image different from an array of pixel values.

## Computer Vision Methods

Computer Vision, in a nutshell, tries to extract meaningful information from images and videos, using computers.

- Image Classification.
- Object Detection.
- Semantic Segmentation.
- Instance Segmentation.
- Panoptic Segmentation.
- Person Segmentation.

## Image Classification

Image classification sees an image and can classify it (a dog, an apple, a person's face). More precisely, it is able to accurately predict that a given image belongs to a certain class.

## Object Detection

<u>What is object detection and how it works?</u>

- Object detection is a computer vision technique that works to identify and locate objects within an image or video. Specifically, object detection draws bounding boxes around these detected objects, which allow us to locate where said objects are in (or how they move through) a given scene.

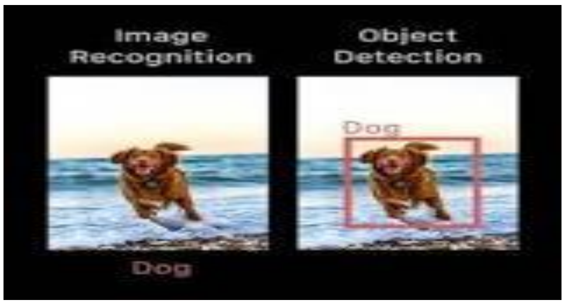


FIGURE 5 (OBJECT DETECTION)

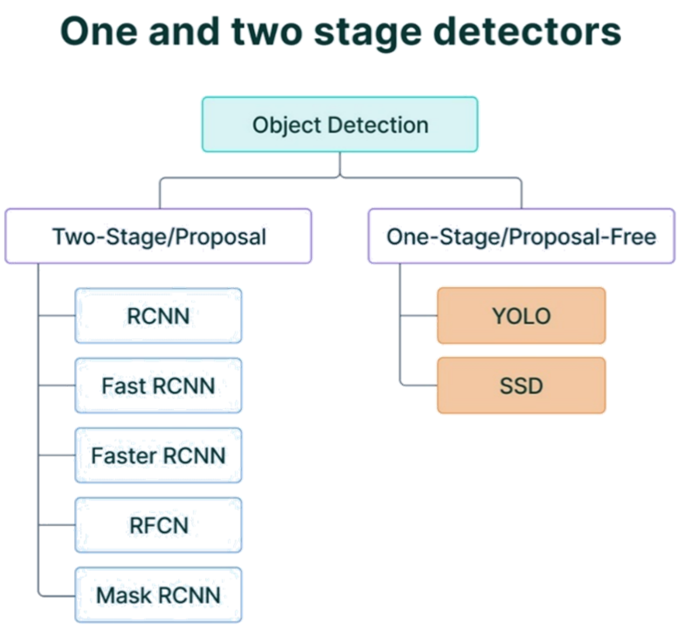


FIGURE6 (OBJECT DETECTION ALGORITHMS)

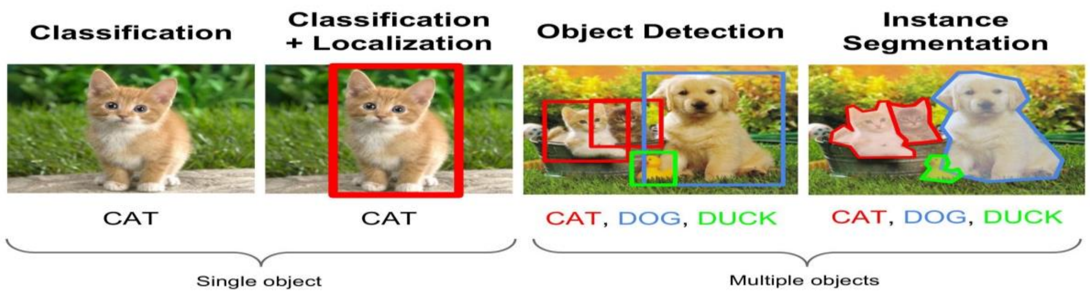


FIGURE 7 (DIFFERENT COMPUTER VISION TECHNIQUES)

## Semantic Segmentation and Instance Segmentation

### Semantic segmentation:

Semantic segmentation is a deep learning algorithm that associates a label or category with every pixel in an image. It is used to recognize a collection of pixels that form distinct categories.

Semantic segmentation treats multiple objects within a single category as one entity. Instance segmentation, on the other hand, identifies individual objects within these categories.

### Instance segmentation:

Instance segmentation is a computer vision task for detecting and localizing an object in an image. Instance segmentation is a natural sequence of semantic segmentation, and it is one of the biggest challenges compared to other segmentation techniques

### Instance segmentations VS Semantic segmentation:

Semantic segmentation classifies the pixel-level category assignments, while instance segmentation, assigns different labels for pixels belong to different instances of the same object type

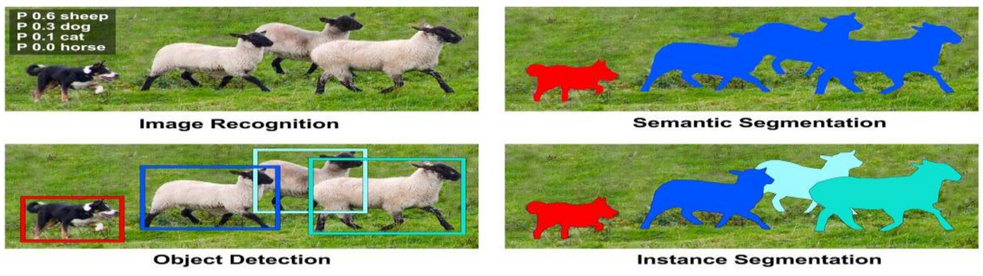


FIGURE 8 (INSTANCE SEGMENTATION)

## Grayscale Images

has only single channel and pixel intensity varied between 0 (black) to 255 (white). It usually has shades of gray.

FIGURE 9 (GRAYSCALE IMAGES)

## RGB Images

-RGB (red, green, and blue) refers to a system for representing the colors to be used on a computer display. Red, green, and blue can be combined in various proportions to obtain any color in the visible spectrum. Levels of R, G, and B can each range from 0 to 100 percent of full intensity.

-full intensity are white, while at the lowest intensity they're black. \

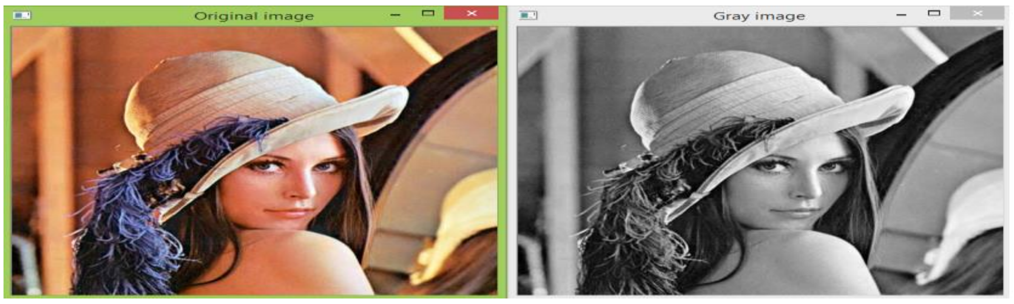


FIGURE 10 (RGB IMAGE)

## Comparison between Different Object Detection Techniques (Methods)

| Algorithm name | Overview | Approach |
|---|---|---|
| **Faster Region-Based Convolutional Neural Networks (Faster R-CNN) (Multi-stage detector)** | -Faster R-CNN is a two-stages object detection algorithm.<br>-It is an upgraded version of R-CNN and Faster R-CNN algorithms.<br>-It is faster than its previous versions owing to | It goes by the following steps:<br>1. image's ROIs' generation<br>2. feature extraction from ROIs.<br>3. object detection and classification.<br><br>It consists of these 2 |

| | | |
|---|---|---|
| | computation sharing among all ROIs in the image. | modules:<br>1. An RPN (Region Proposal Network) for generating ROIs.<br>2. Fast RCNN (An older version):<br>- It uses ROI pooling for extracting feature vectors from all ROIs in the image, thus sharing computations among all ROIs, saving time consequently.<br>- These features are then passed into some fully connected layers, and their output is split into 2 branches, where: Class scores of objects are output by a softmax layer.<br>Bounding boxes of the detected objects are predicted by another FC layer. |
| **Region-Based Fully Convolutional Neural Networks<br>(R-FCN)<br>(Multi-stage detector)** | -It is an algorithm which focuses mainly on the conflict between location invariance and location invariance when classifying and detecting an object respectively.<br>-It increases its speed by maximizing computation sharing.<br>-It fixes this problem of position sensitivity's deficiency faced in Faster R-CNN using what is known as position-sensitive score maps. | This approach mainly depends on the usage of position-sensitive score maps. These maps represent each a single relative position of an object class.<br>This works as follows:<br>1. A CNN is run over the image to create feature maps.<br>2. A fully CNN is used to generate a score bank (bank of all position sensitive score maps), whereas the number of score maps generated equals the number |

| | | |
|---|---|---|
| | -This algorithm is performed over 2 steps. | of relative positions into which an object was divided according to the feature maps, multiplied by the number of classes of objects plus the background $k^2(C + 1)$.<br>3. An RPN is used to generate ROIs.<br>4. Then for detection, each ROI is divided into subregions whose number is equal to that of the score-maps'.<br>5. Each subregion in the ROI is compared with the score maps of the corresponding position. Then these values are averaged to get a single score for each class.<br>6. The ROI is then classified using a softmax layer over the remaining C+1 dimensional vector. |
| **Single Shot MultiBox Detector (SSD)**<br>**(Single shot detector)** | -It is a single-stage object detection technique which turns to multi-scale feature maps to improve accuracy and detect objects of different scales.<br>-It has a higher speed due to using lower resolution images and eliminating the usage of RPNs. | 1. Feature maps extraction using VGG16 algorithm.<br>2. Convolutional filtering for object detection. Whereas the CNN creates several bounding boxes of different aspect ratios at each cell, then predict the score of each object class.<br>3. Finally, a non-maximum suppression algorithm is used for giving the final detected object class. |

| | | |
|---|---|---|
| **You Only Look Once (YOLO)**<br>**(Single shot detector)** | It is one of the single-stage object detection algorithms.<br>-It treats the task of object detection as a regression task by predicting the probabilities of the presence of class objects in the images.<br>-Its object detection's speed is extremely high, making it suitable for real-time object detection applications | 1. Image is split into grids, where as object detection will be performed for objects in each cell.<br>2. Bounding box regression is performed, in which each cell predicts the dimensions of the bounding box enclosing the object detected, as well as the probabilities for each object class.<br>3. An evaluation metric, known as "Intersection Over Union" is used to compare the predicted bounding boxes with the actual bounding box, eliminating all unnecessary boxes except the one perfectly overlapping with the actual bounding box. |
| **RetinaNet** | - RetinaNet is another algorithm that is performed over one stage.<br>-It solves the issue of class imbalance during training by using focal loss function which reduces the weight of easy training examples as the confidence in the prediction of their correct class rises, focusing learning on hard examples instead.<br>-It is suitable for the detection of small-scaled and dense objects. | 1. Feature map extraction using a backbone convolutional network.<br>2. Object Classification using a convolutional subnetwork.<br>3. Bounding box regression using a second convolutional subnetwork |

## What is Deep Learning?

Deep learning is a machine learning technique that teaches computers to do what comes naturally to humans

- It is a subset of machine learning and is called deep learning because it makes use of deep neural networks
- Deep learning is a computer software that mimics the network of neurons in a brain.
- In deep learning, the learning phase is done through a neural network.
- Models are trained by using a large set of labeled data and neural network architectures that contain many layers.

- It is a field that is based on learning and improving on its own by examining computer algorithms.

## Usages of Deep Learning

Used for processing large amounts of complex data.

Deep learning is a key technology behind driverless cars.

## Advantages of Using Deep Learning

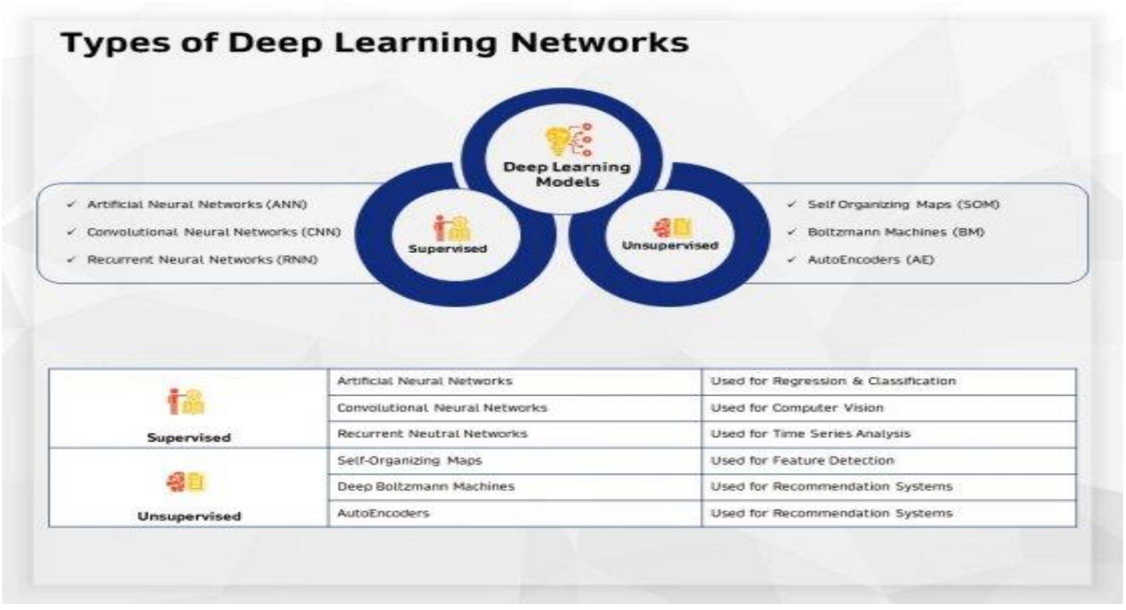
Deep learning models can achieve state-of-the-art accuracy, sometimes exceeding human-level performance.

Deep learning is distinguished by its learning of multiple levels of features.

## Types of Algorithms used in Deep Learning

- Convolutional Neural Networks (CNNs)
- Artificial Neural Networks (ANNs)
- Long Short-Term Memory Networks (LSTMs)
- Recurrent Neural Networks (RNNs)
- Generative Adversarial Networks (GANs)
- Radial Basis Function Networks (RBFNs)

- Multilayer Perceptrons (MLPs)Self-Organizing Maps (SOMs)
- Deep Belief Networks (DBNs)
- Restricted Boltzman Machine (RBMs)


Types of Deep Learning Networks
Deep Learning Models
Supervised
Unsupervised
✓ Artificial Neural Networks (ANN)
✓ Convolutional Neural Networks (CNN)
✓ Recurrent Neural Networks (RNN)
✓ Self Organizing Maps (SOM)
✓ Boltzmann Machines (BM)
✓ AutoEncoders (AE)

| | | |
|---|---|---|
| Supervised | Artificial Neural Networks | Used for Regression & Classification |
| | Convolutional Neural Networks | Used for Computer Vision |
| | Recurrent Neutral Networks | Used for Time Series Analysis |
| Unsupervised | Self-Organizing Maps | Used for Feature Detection |
| | Deep Boltzmann Machines | Used for Recommendation Systems |
| | AutoEncoders | Used for Recommendation Systems |

FIGURE 11 (TYPES OF DEEP LEARNING)

## Artificial Neural Networks (ANN)

Neural networks represent deep learning using artificial intelligence.

Artificial Neural Networks (ANN) are algorithms based on brain function and used to model complicated patterns and forecast issues.

The development of (ANN) was the result of an attempt to replicate the workings of the human brain.

## Artificial Neural Networks (ANN) Architecture

There are three layers in the network architecture:

- The input layer.
- The hidden layer (more than one)
    - Hidden layers are the ones that are actually responsible for the excellent performance and complexity of neural networks. They perform multiple functions.

- The output layer (Multi-Layer Perceptron)

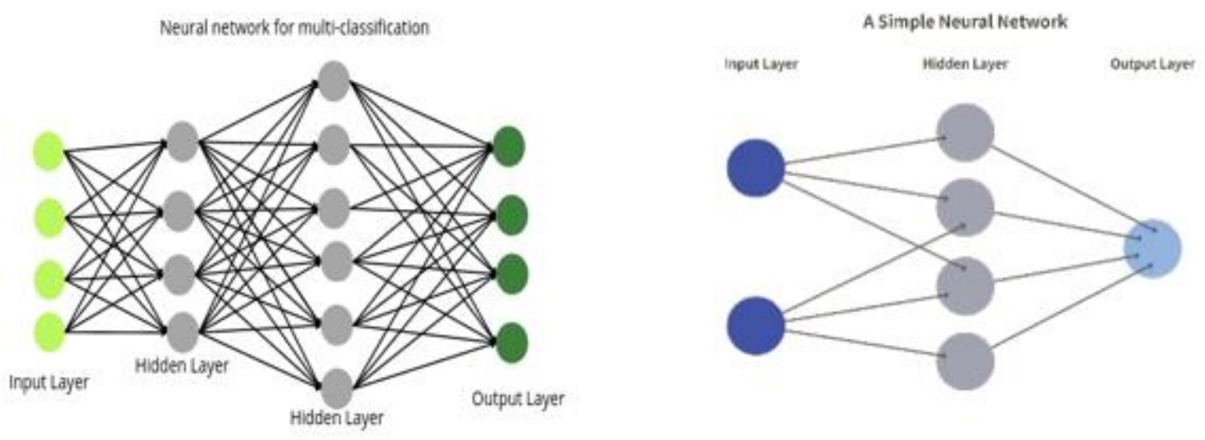


FIGURE 12 (ARCHITECTURE OF NEURAL NETWORKS)

## Convolutional Neural Networks (CNNs)

A CNN is a multilayer neural network that was biologically inspired by the animal visual cortex. first CNN was created by Yann LeCun, at that time, the architecture focused on handwritten character recognition,such as postal code interpretation.

A convolutional neural network (CNN or ConvNet), is a network architecture for deep learning which learns directly from data, eliminating the need for manual feature extraction.

As a deep network, early layers recognize features (such as edges), and later layers recombine these features into higher-level attributes of the input.

## Usages of Convolutional Neural Networks

CNN's are widely used to identify satellite images, process medical images. forecast time series, and detect anomalies
It was used for recognizing characters like ZIP codes and digits.

Convolutional Neural Networks (CNN) and Recursive Neural Networks (RNN) are used to accept unstructured and non-numeric data forms such as Image, Text, and Speech.

The building blocks of CNNs are filters a.k.a. kernels. Kernels are used to extract the relevant features from the input using the convolution operation.

CNN best to solve problems related to: image and video processing projects

Also used in smart grid applications as well.

## Examples of CNN's Applications

- image and video recognition.
- image classification.
- Object detection
- image segmentation.
- medical image analysis.
- natural language processing.

## CNN's Architecture

CNN has three of layers to build architectures apart from input layer:

1. Convolution layer (with RELU activation)

2. Pooling Layer

3. Fully connected

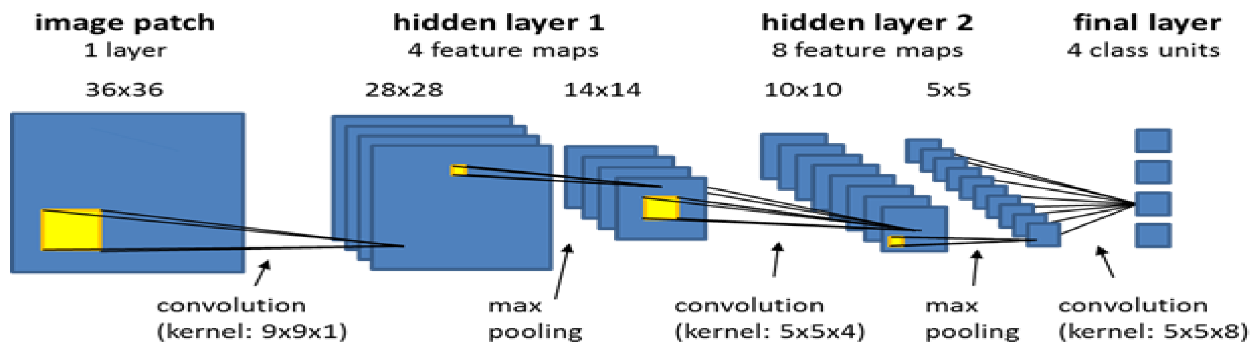


FIGURE 13 (ARCHITECTURE OF CONVOLUTIONAL NEURAL NETWORKS)

**1. Convolution Layer:**

CNN has a convolution layer that has several filters to perform the convolution operation.

The primary purpose of Convolution is to extract features from the input image. Convolution preserves the spatial relationship between input by learning input features. Convolution layer output y = 2" x * f

- **Rectified Linear Unit (ReLU):**

  CNN's have a ReLU layer to perform operations on elements. The output is a rectified feature map.

2. **Pooling Layer**

   It is also called a down-sampling operation which reduces the dimensionality of each map but retains important information.

   Pooling layers would reduce the numbers of parameters when the inputs are too large.

   There are three types of pooling namely, max pooling, average polling, sum pooling.

   **Flatten layer**: -the feature matrix will be converted as vector (x1, x2, x3).

**3. Fully Connected Layer:**

A fully connected layer forms when the flattened matrix from the pooling, every single neuron has a connection to every single one in the next layer.

## Comparison between Some Neural Networks Architecture

| Algorithm name | Description | How it works | Usages |
|---|---|---|---|
| **Long Short-Term Memory Nrtworks(LSTMS)** | -LSTMs are a type of recurrent Neural network that can learn and memorize long-term dependencies. | -First, they forget irrelevant parts of the previous state Next, they selectively update the cell-state values Finally, the output Of Certain parts Of the Cell state. | -They are useful in time- series prediction because they previous inputs. -used for speech recognition. and pharmaceutical development |

| | | | |
|---|---|---|---|
| **Recurrent Neural Networks (RNNs** | -RNNs have connections that Networks form directed cycles, which allow the outputs from the LSTM to be (RNNs) fed as inputs to the current phase. The output from the LSTM becomes an input to the current phase and can memorize previous inputs due to its internal memory. | -The output at time t-l feeds into the input at time t. Similarly, the output at time t feeds into the input at time t+1.<br>-RNNs can inputs of any length. The computation accounts for historical information.<br>and the model size does not increase with the input size. | -RNNs are commonly used for image captioning, time-series analysis, natural- language handwriting recognition, and machine translation |
| **Generative Adversarial Networks (GANs)** | -GANs are generative deep learning algorithms that create new data instances that resemble the training data.<br>-GAN has two components: a generator which learns to generate fake data, and a discriminator, which learns from that false information. | -The discriminator learns to distinguish between the generator's fake data and the real sample data.<br>-During the initial training, the generator produces fake data, and the discriminator quickly learns to tell that it's false.<br>-The GAN sends the results to the generator and the discriminator to update the model. | -The usage of GANs has increased over a period of time.<br>-They can be used to improve astronomical images and simulate gravitational lensing for dark-matter research.<br>-GANs help generate realistic images and cartoon characters, create photographs of human faces, and render 3D objects |

## Why do we use CNN?

Convolutional Neural Networks, or CNNs, were designed to map image data to an output variable.

They have proven so effective that they are the go-to method for any type of prediction problem involving image data as an input.

The benefit of using CNNs is their ability to develop an internal representation of a two-dimensional image.

This allows the model to learn position and scale in variant structures in the data, which is im1X)1tant when working with images.

Use CNNs for:

- Image data
- Classification prediction problems
- Regression prediction problems

## Why Have We Chosen YOLO for Vehicle Detection?

YOLO is better, faster, and stronger method of detecting objects which uses the latest technology in real time, which can detect more than 200 classes and 9,000 different categories of objects. It uses a CNN for performing real-time object detection.
Its effective real-time object detection has made it suitable for vehicle detection for traffic light control, which requires quick detections and priority estimation.
It divides the image into regions and predicts bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities.
YOLO is used in order to detect the number of vehicles and then set the timer of the traffic signal according to vehicle density in the corresponding direction.
YOLOv7, the latest version is also said to have surpassed all state-of-the-art object detection algorithms, not only in speed, but also in accuracy.
YOLOv7 is the fastest and most accurate real-time object detection model for computer vision tasks.

## YOLOv7 Architecture

- Extended Efficient Layer Aggregation Network (E-ELAN)
- Model Scaling for Concatenation based Models

## What is New in YOLO-v7?

YOLOv7 provides a greatly improved real-time object detection accuracy without increasing the inference costs. As previously shown in the benchmarks, when compared to other known object detectors, YOLOv7 can effectively reduce about 40% parameters and 50% computation of state-of-the-art real-time object detections, and achieve faster inference speed and higher detection accuracy. In general, YOLOv7 provides a faster and stronger network architecture that provides a more effective feature integration method, more accurate object detection performance, a more robust loss function, and an increased label assignment and model training efficiency. As a result, YOLOv7 requires several times cheaper computing hardware than other deep learning models. It can be trained much faster on small datasets without any pre-trained weights.

## Examples of Object Detection Datasets

- VOC 2007 / 2012:
  - 20 classes (such as people, cat, dog, car, chair, bottle...)
- ImageNet1000:
  - 1000 classes (such as German shepherd, golden retriever, European fire salamander…)

- MS COCO:
  - MS COCO (Microsoft Common Objects in Context) is a large-scale image dataset containing 328,000 images of everyday objects and humans, It contains about 80 classes of different objects (such as book, apple, teddy bear, scissors…)
  - The dataset contains annotations you can use to train machine-learning models to recognize, label, and describe objects.
  - Annotation file of COCO is in JSON format.

## Dataset

We wrote a code to select specific classes (car/motor cycle/truck/bicycle/bus) from COCO dataset and other datasets for emergency cars (police /ambulance/fire truck) and wrote a code to specify an id for each class:

- Police car: 0  Ambulance:1  Fire truck:2
- Car:3  Motorcycle:4  Truck:5
- Bicycle:6  Bus:7

Then we augmented the emergency car datasets to equivalent the number of pictures in each class, after that we split the data to train 80% & valid 20%

**Data augmentation: -**

Data augmentation is a set of techniques used to increase the amount of data in a machine learning model by adding slightly modified copies of already existing data or newly created synthetic data from existing data

**Benefits of Data Augmentation:**

A machine learning model performs better and is more accurate when the dataset is rich and comprehensive.

**Operations in data augmentation:**

Rotation Shearing Zooming Cropping Flipping Changing the brightens level

---

## Comparison between Python and Other Languages Used for AI

| | |
|---|---|
| **R** | -R is generally applied when you need to analyze and manipulate data for statistical purposes. R has packages such as Gmodels, Class, Tm, and RODBC that are commonly for building machine learning projects. These packages allow developers to implement machine learning algorithms without extra hassle and let them quickly implement business logic.<br>-R was created by Statisticians to meet their This language can give you in-depth statistical analysis whether you're handling data from an l0T device or analyzing financial models.<br>What's more, if your task requires high-quality graphs and Charts, you may want to use R. With ggplot2, ggvis, google Vis, Shiny, rCharts, and other packages, R's capabilities are greatly extended, helping you turn visuals into interactive web apps.<br>-Compared to Python, R has a reputation for being slow and lagging when it comes to large-scale data It's better to use Python or Java, with its flexibility, for actual product development. |

| | |
|---|---|
| **Java** | Another language worth mentioning is Java. Java is object-oriented, portable, maintainable, and transparent. It's supported by numerous libraries such as WEKA and RapidMiner.<br><br>Java is widespread when it comes to natural language processing, search algorithms, and neural networks. It allows you to quickly build large-scale systems with excellent performance.<br><br>But if you Want to perform statistical modeling and visualization, then Java is the last language you want to use. Even though there are some Java packages that statistical and visualization, they aren't sufficient. Python, on the other hand, has advanced that are well supported by the community.<br><br>we think that the Python ecosystem is well-suited for AI-based projects. Python, with its simplicity, large community, and allows developers to build architectures that are close to perfection while keeping the focus on business-driven tasks. |
| **Lisp** | -LISP is the oldest AI programming language. It is the second oldest programming language after Fortran. The term Artificial intelligence made up by John McCarthy invented LISP. Another pioneer "as Marvin Minsky, who founded the AI lab at MIT.<br>-There would no progress in AI at that time if it weren't LISP. It had fresh ideas (if-then-else, Construct, recursion) which were very useful to express the ideas programmers had. Because of the huge adoption of LISP, it became a standard AI language. LISP is a very flexible AI programming language and is often called “the most intelligent way to misuse a computer".<br>“Lisp uniformity of structure and power of self-reference gave the programmer capabilities whose content was well worth the sacrifice of visual form " — Marvin Minsky.<br><br>-LISP has influenced creating many AI programming languages, and the most worthwhile mentioning are R and Julia. Why aren‘t people using LISP as the main AI programming language today?<br>Because even though it's very flexible, it has many flaws. There is a lack of well-known libraries and a weird syntax that doesn't attract many people are the main Back then, it was ahead of its time, and that's why it deserves mention on this AI programming list. But there are many solutions nowadays than LISP |
| **Python** | -python is AI programming language that has gained huge popularity. The main reasons are the simple syntax. less coding and a large number of available libraries ready for use. Simple syntax means you can focus on the core value of programming, thinking, problem-solving. |

| | |
|---|---|
| | -The earlier mentioned libraries include NumPy, matplotlib, NLTK, Simple AI.<br>-Python is an open-source AI programming language. That's why it has a huge fan base among programmers. Because it can used broadly, to make small scripts and up to enterprise applications. it's suitable for AI.<br>Where Other AI programming languages use punctuation, uses English keywords. It's designed to be readable. It has a few keywords and has a clearly defined syntax. If you are a student, you will pick up the language quickly.<br>The libraries are across platforms such as UNIX, Windows, and Macintosh. It also provides interfaces for all major commercial databases. When it to scalability, it provides a structure and support large enterprise than it does for Simple shell scripts. Python supports Object-oriented programming (OOP), dynamic checking, automatic garbage collection, and can integrated with C++, C, Java, Cobra, and many other languages. |
| **C++** | Bjarne Stroustrup developed C++ in 1983, and it holds the title for the "fastest programming language." Time is important for AI projects and C++ is the usual when it comes to that.<br>Search engines use C++ to less resvKM1se time, and the development of game takes advantage of the fast execution.<br>Because C++ has a complex syntax, it might not your first choice as AI programming language, but if you are working in an environment and afford Java Virtual Machine, this is the perfect option for you.<br>There are limitations to C++ because the standard library is small. and it doesn't support garbage collection. Although have better efficiency of control. large C++ projects are hard to maintain and time-consuming to develop. That might the main reason why most people avoid C++ in AI programming. There is me very important use of C++ in AI programming, and that is Google Chrome.<br>The AI is used for search engine optimization and ranking |

## Why is Python Used for Al and Machine Learning?

1. **Python is easy to understand.**

   Its readability, non-complexity, and ability for fast prototyping make it a popular language among developers and programmers around the world.

2. **Python comes with a large number of libraries.**

   - **Scikit-Learn, Keras** for data mining, analysis, and Machine Learning.
   - **Numpy** for high-performance scientific computing and data

- **Tensorflow**, a high-level neural network library.
- **pylearn2** which is also ideal for data mining and Machine Learning, but more flexible than scikit-learn.
- **Pandas** for general pupose data analysis
- **Scipy** for advanced computing
- **Seaborn** for data visualization
- **OpenCV** for computer vision

**3. Python allows easy and powerful implementation.**

if you only have basic knowledge of the Python language, you can already use if for Machine Learning because of the huge amount of libraries, resources, and tools available for you.

Additionally, you will spend less time writing code and debugging errors on Python than on Java or C++.

ML and AI programmers, in general, would rather spend their time building their algorithms and heuristics, rather than debugging their code for syntax errors.

## 2.2 Related Works

-----------------------------------------------------------------------------------------------------

### AI in Transportation Systems

There have been many advances made by AI in many fields, such as medicine, industrial control, self-driving cars, smart houses and many more.

Its applications have been progressively increasing and evolving as a result of advancements in information technology, high-speed processers, as well as larger-capacity memory devices. Also, digitization of both audios and images have made it possible for AI systems to learn from them as inputs. Not to mention them being able to communicate with other systems and devices using fast wireless networks.

Artificial Intelligence has been making its entrance in many transportation-related applications, such as self-driving cars, traffic prediction, automated parking management, road condition monitoring, and pedestrians' behavior analysis and prediction. It has significantly been making advances in the field of traffic control. There have been many studies regarding that topic, as well as actual implementations of AI-driven traffic control management systems. Following are some of these experiments and studies.

In 2012, C. M. Mwangi, S. M. Kang'ethe and G. N. Nyakoe [1] have used Fuzzy logic to create a dynamic traffic control system. The proposed system focused on decreasing the delay time of vehicles, and increasing the road intersection's capacity by adaptively adjusting the sequence or duration of traffic signals' phases. The fuzzy logic controller would accept vehicle counts of each lane forming the intersection, as well as their delay times from inductive loop sensors, then determines whether to prolong or end the current green phase, and which phase should follow next. They used two sets of inductive loop sensors, where each set was installed at the stopping line of the intersection and 100m away at each lane in order to determine the queue length and delay time of each approach of the intersection.

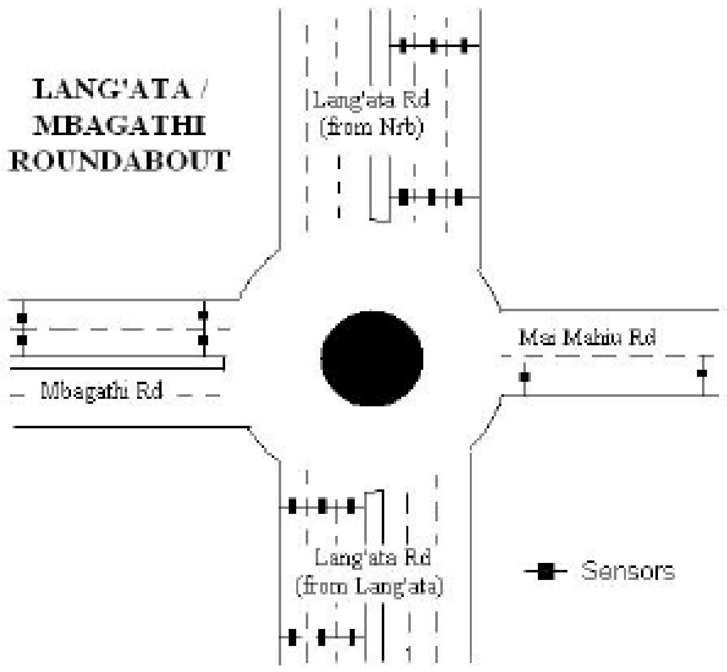
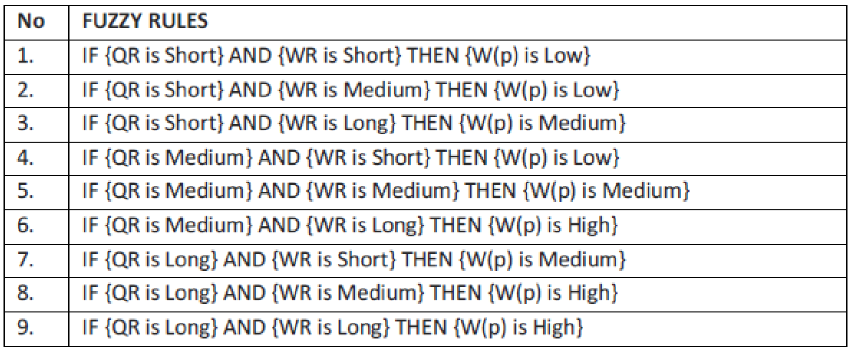


FIGURE 14 (INTERSECTION STRUCTURE)

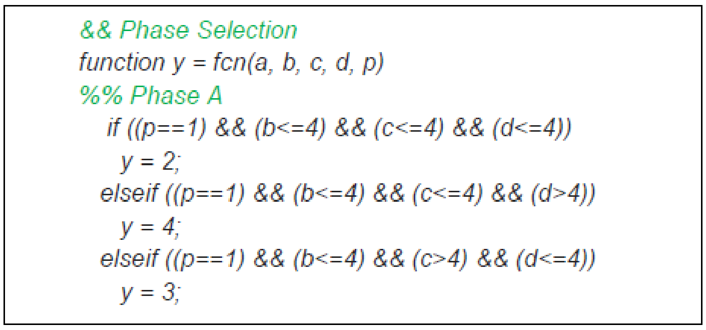
This data is then sent to the fuzzy-logic controller (FLC), where it determines the priority of each approach according to a set of fuzzy rules.



| No | FUZZY RULES |
|---|---|
| 1. | IF {QR is Short} AND {WR is Short} THEN {W(p) is Low} |
| 2. | IF {QR is Short} AND {WR is Medium} THEN {W(p) is Low} |
| 3. | IF {QR is Short} AND {WR is Long} THEN {W(p) is Medium} |
| 4. | IF {QR is Medium} AND {WR is Short} THEN {W(p) is Low} |
| 5. | IF {QR is Medium} AND {WR is Medium} THEN {W(p) is Medium} |
| 6. | IF {QR is Medium} AND {WR is Long} THEN {W(p) is High} |
| 7. | IF {QR is Long} AND {WR is Short} THEN {W(p) is Medium} |
| 8. | IF {QR is Long} AND {WR is Medium} THEN {W(p) is High} |
| 9. | IF {QR is Long} AND {WR is Long} THEN {W(p) is High} |

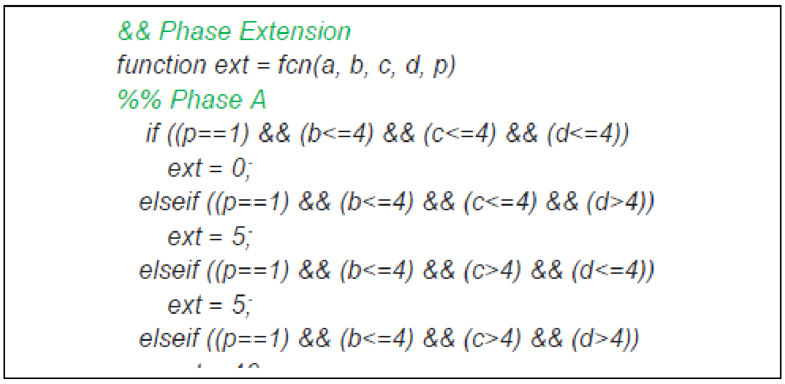
FIGURE 15 (FUZZY RULES FOR EVALUATING THE PRIORITY OF EACH RED-SIGNAL APPROACH)

This FLC system contains two sets of embedded Matlab if-then rules, one for determining whether to extend or terminate the current green phase, while the other is used for selecting the next phase. These two sub-systems perform their functions using the approaches' priority data resulting from the fuzzy logic unit. These decisions are then sent to a programmable logic controller (PLC) for actuation of the resulting decisions.



```
&& Phase Selection
function y = fcn(a, b, c, d, p)
%% Phase A
  if ((p==1) && (b<=4) && (c<=4) && (d<=4))
    y = 2;
  elseif ((p==1) && (b<=4) && (c<=4) && (d>4))
    y = 4;
  elseif ((p==1) && (b<=4) && (c>4) && (d<=4))
    y = 3;
```

FIGURE 16 (EMBEDDED MATLAB IF-THEN RULES FOR NEXT PHASE SELECTION)



```
&& Phase Extension
function ext = fcn(a, b, c, d, p)
%% Phase A
  if ((p==1) && (b<=4) && (c<=4) && (d<=4))
    ext = 0;
  elseif ((p==1) && (b<=4) && (c<=4) && (d>4))
    ext = 5;
  elseif ((p==1) && (b<=4) && (c>4) && (d<=4))
    ext = 5;
  elseif ((p==1) && (b<=4) && (c>4) && (d>4))
```

FIGURE 17 (EMBEDDED MATLABLE IF-THEN RULES FOR GREEN PHASE EXTENSION/TERMINATION)

The simulation was carried out using Matlab, Simulink, and Fuzzy Logic Toolbox. In normal traffic conditions, this approach shows performance levels similar to similar that of traditional fixed-cycle traffic control system. However, it achieves a significant increase in performance in heavy traffic situations, whereas it has achieved a reduction of 25% in average delay time, and a 6% increase in the number of vehicles successfully passing the intersection.

---

In 2017, a study was done by Ying Liu, Lei Liu, and Wei-Peng Chen [2] in which a traffic control system based on distributed reinforcement learning has been proposed. It aimed to improve both motorized and non-motorized traffic. The solution involved the usage of a multi-agent Q-learning approach for controlling traffic signals, whereas each agent would learn from the environment – which is the intersection it is assigned to - and find the best control actions for minimizing the queue lengths for both vehicles and pedestrians. The data observed at each intersection is exchanged with other agents over a network. This helps each agent take the traffic state of other surrounding intersections into consideration when determining the best suitable action.

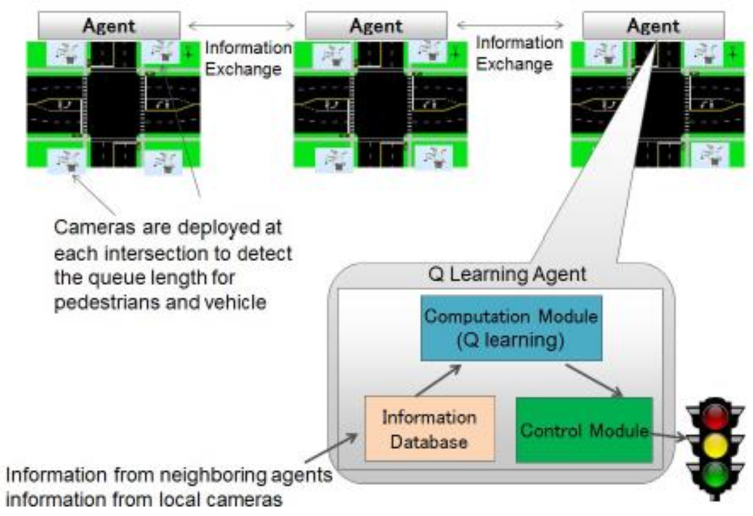

Agent
Information Exchange
Agent
Information Exchange
Agent
Cameras are deployed at each intersection to detect the queue length for pedestrians and vehicle
Q Learning Agent
Computation Module (Q learning)
Information Database
Control Module
Information from neighboring agents information from local cameras

FIGURE 18 (SYSTEM ARCHITECTURE FOR MULTI-AGENT Q-LEARNING)

Unlike other traffic light control solutions which only considered motorized traffic due to the difficulty of modelling the correlation between vehicular and pedestrian's mobility, this study considered both motorized and non-motorized traffic in their solution by dynamically monitoring and collecting queue lengths of both vehicles and pedestrians. It also considered other surrounding intersections when managing traffic, not only the intersection in question. Reinforcement learning is a learning method in which an agent could engage with the environment and learn to select optimal actions to apply on the environment which would optimize some-kind of a cumulative reward over time. There are two types of RL algorithms, model-based and model-free algorithms. Model based algorithms are able to predict the consequences of their actions on the environment and how it would change from one state to another based on these actions, and in turn refine their policy accordingly. On the other hand, model-free algorithms refine their policy by experimenting and learning from the consequence of their previous actions. Q-learning algorithm is a model-free reinforcement learning algorithm which selects the optimal action for a situation by computing and updating what is known as Q-values or action values Q($s_t$, $a_t$):

$$Q^t(s,a) = (1-\alpha)Q^{t-1}(s,a) + \alpha(R^t + \gamma \max_a Q^{t-1}(s,a))$$

Where:

α is the learning rate, γ is the discount, $R^t$ is the instantaneous reward at time t.

The agent could either work independently or cooperate with other agents by sharing knowledge among each other over an available network, so that a global optimal solution could be reached.

This study used this concept of multi-agent distributed Q-learning. Each intersection had a Q-learning agent implemented, where it received traffic information and queue lengths of both vehicles and pedestrians from surveillance cameras set at the intersection. This data gets stored in a local database so that it could then be shared with agents at neighboring intersections. The agent could also benefit from traffic data of other agents so that it could choose the best action for the intersection it is implemented in. This aims to achieve synchronization of traffic light control among neighboring intersections, instead of just shifting congestions to another area or intersection. The data received by the agent from the cameras are considered as the “state” of the environment. Queue lengths of both vehicles and pedestrians at each lane of the intersection are used to formulate the environment’s state. The optimal action evaluated by Q-computation module is then executed by a control module, which is a pre-programmed hardware. The actions available at each intersection may differ from one another, depending on the intersection’s structure.

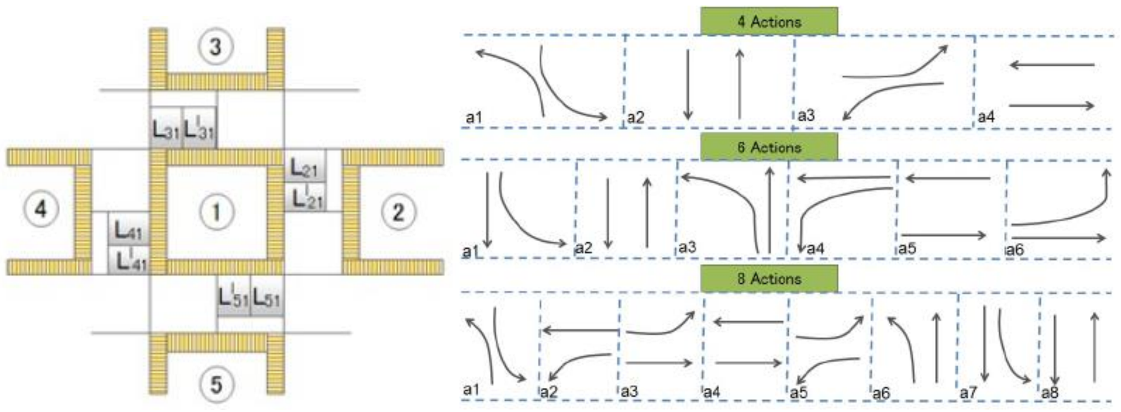


FIGURE 19 (ACTIONS AVAILABLE FOR AGENT FOR A "+"-SHAPED INTERSECTION)

The objective function here was to minimize the total average of queue length for both motorized and non-motorized traffic. The simulation was performed using a real-world map and real traffic data. Compared to other real-world solutions, this system has showed better results. In comparison with MARL – another machine learning based solution - It has achieved a 16.7% reduction in vehicle waiting time in traffic situations with high pedestrian rates, and a reduction of 12.2% and 7.0% in case of medium and low pedestrian rates respectively. The study concluded that there could be future works of increasing the algorithm’s convergence time, so that it could handle sharp-turning traffic situations. Also, different deployment models could be tried out.

---

In another study done by R. F. O. Adebiyi, K. A. Abubilal, A. M. S. Tekanyi, and B. H. Adebiyi [3] in 2017, an adaptive dynamic traffic light scheduling system using artificial bee colony algorithm was introduced. It focused on providing an intelligent green signal timing according to the dynamic traffic loads. Artificial Bee Colony algorithm is a population-based optimization technique. It resembles the behavior of honeybee swarms. A colony is composed of three components: employed bees, unemployed bees, and food sources. These food sources represent all actions or solutions available to perform in the current situation. The nectar amount in food represents the quality or fitness of the action. Both employed bees randomly search for richer food sources, which represent optimized solutions for the current situation. Unemployed bees learn about the locations of food sources through the navigation information provided by employed bees and choose which food source to utilize. Unemployed bees are classified into onlooker and scout bees. Onlooker bees collect information from employed bees and decide on the food source, while scout bees look for other rich sources in the same neighborhood as the chosen food source. The intersection simulated consisted of four roads, each consisting of 2 lanes going for different directions, as illustrated:

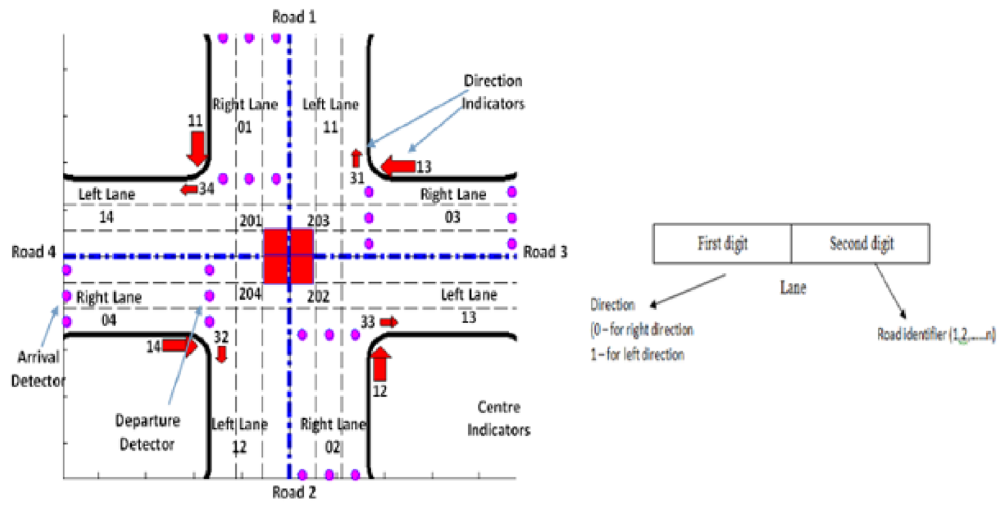


FIGURE 20 (SIMULATED ROAD INTERSECTION)

The results showed that the system has achieved a 76.67% reduction in average waiting time, and a 53.33% reduction in vehicular queue length.

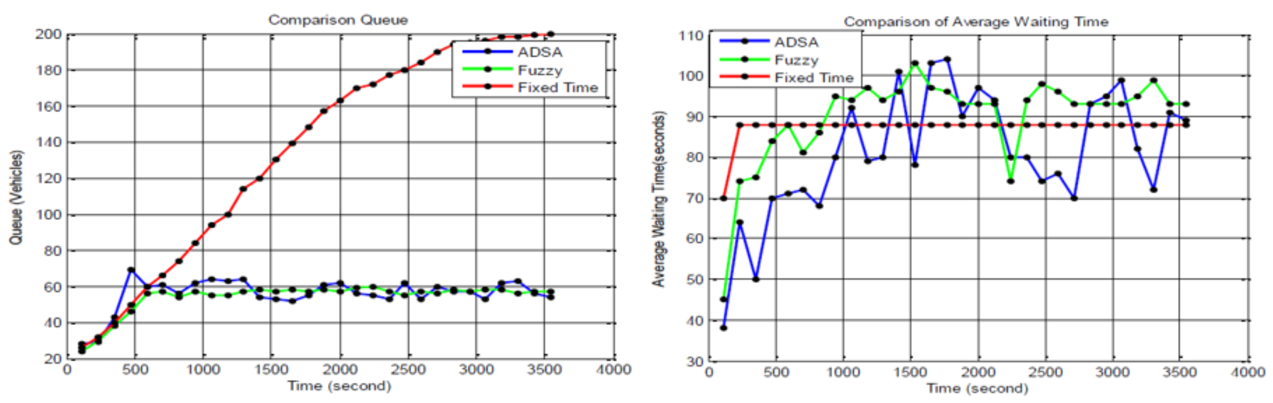


FIGURE 21 (PERFORMANCE COMPARISON BETWEEN FUZZY, FIXED-TIME, AND ADSA TRAFFIC CONTROLLERS)

---------------------------------------------------------------------------------------------------------------------------

Moreover, in 2018, K. Zaatouri and T. Ezzedine [4] proposed a real-time vehicle-detection model using a deep CNN, known as You Only Look Once. The main point here was the utilization of images taken by cameras at traffic junctions for a real-time traffic light control system based on traffic flow. The object detection algorithm used here, YOLO, is a state-of-the-art real-time object detection algorithm which uses fully-convolutional neural networks (FCN). It divides the image into NxN grid cells, whereas each cell is responsible for detecting objects it contains. Where bounding boxes are predicted in each cell, as well as the object's class and the probability of that class being contained in that cell.

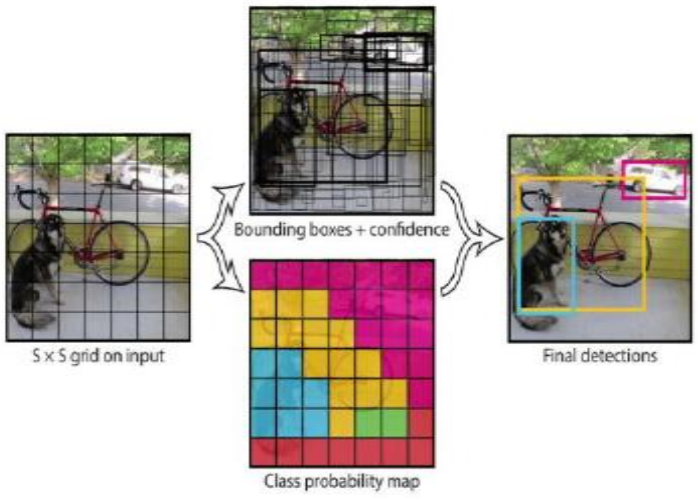


FIGURE 22 (OBJECT DETECTION APPROACH WITH YOLO)

The version used was YOLOv3, a version using a darknet variant constituting of 53 pre-trained layers, in addition to another 53 layers added for detection, summing up to

106. YOLOv2, the older version, which was only composed of a darknet architecture of 30 layers: 19 pre-trained layers, and 11 layers for detection which made it difficult for it to detect smaller objects. The cameras set at each lane in the intersection would capture images of each lane. YOLO-based model would then use these images to detect and count vehicles for each lane. This data is then used for traffic signal control optimization along with average waiting time for vehicles in each lane. The system's cycle went as follows: First the traffic signal starts with default green time. As the signal turns yellow, YOLO algorithm starts detecting vehicles and calculating their waiting time till they leave the intersection. It then sends this data to the controlling algorithm in-order for it to calculate the time required for the next green phase based on the maximum and average waiting times for each lane, and their queue lengths. Following the same cyclic order of phases, the controlling algorithm prioritizes lanes of vehicles with the highest waiting time.

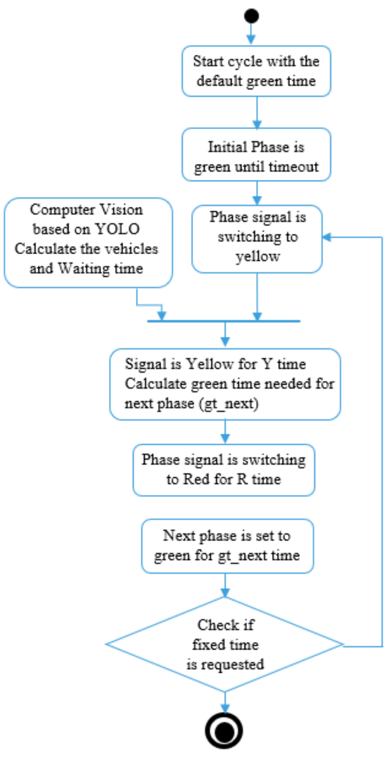

Start cycle with the default green time
Initial Phase is green until timeout
Computer Vision based on YOLO Calculate the vehicles and Waiting time
Phase signal is switching to yellow
Signal is Yellow for Y time Calculate green time needed for next phase (gt_next)
Phase signal is switching to Red for R time
Next phase is set to green for gt_next time
Check if fixed time is requested

FIGURE 23 (ACTIVITY DIAGRAM OF THE SYSTEM)

The study used COCO dataset, an object-detection dataset of about 80 different object classes. YOLO's efficiency and accurate results could give the possibility of replacing police officers managing the traffic. For more optimization, sensors could be used to compensate for the poor quality of cameras during bad weather.

----------------------------------------------------------------------------------------------------

Furthermore, in 2021, B. Kovári , L. Szoke, T. Bécsi, S. Aradi and P. Gáspár [5] proposed a study about deep reinforcement learning solutions for an intelligent traffic light control system. Its objective was the reduction of harmful environmental emissions and fuel consumption. This paper focused mainly on comparing two reinforcement learning algorithms, Vanilla Policy Gradient (PG) which is a policy-based algorithm and Deep Q-Learning (DQN) which is a model-free, value-based algorithm, along with other traditional traffic control algorithms. The agents learned from randomized traffic scenarios generated by Simulation of Urban Mobility software's random generator, so that they could adapt to a variety of traffic situations, not only the usual ones. Policy Gradient algorithm aimed to find the best policy to follow for traffic control. On the other hand, Deep Q-learning aimed to find the best action to take for the current traffic situation. A multi-layer perceptron (MLP), which is a simple fully-connected linear feedforward neural network has been used for both algorithms, along with a non-linear activation function. The imulation was done by Simulation of Urban Mobility (SUMO) simulator software. The intersection was designed as following: Each road in the intersection was 500m long and had two direction lanes, each of 3.2m width. The simulation had a uniform frequency in the range of [1, 200] by which vehicles entered the simulation at each road set. The simulation has been randomized for best training experience and generalization. A python component would manipulate the environment based on the agent's decisions. The component interacted with the agents via SUMO's TraCl interface, which gives access to a running simulation.

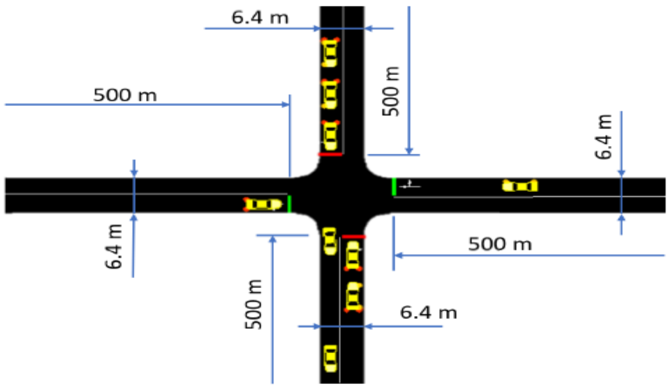


FIGURE 24 (STRUCTURE OF SIMULATED INTERSECTION)

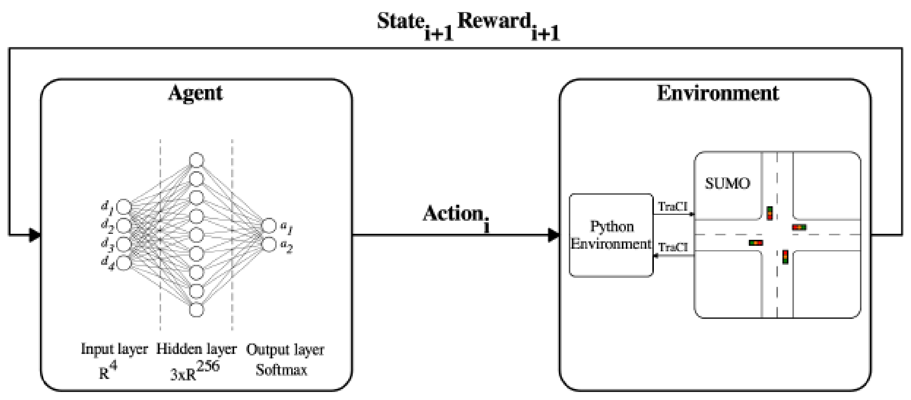


FIGURE 25 (INTERACTION CYCLE OF AGENTS AND ENVIRONMENT MANIPULATION COMPONENT)

The algorithms' performances were evaluated based on traditional and environmental sustainability metrics, as shown in the tables below. The results concluded that PG performed significantly better than the other algorithms, with DQN coming in second place. PG algorithm was capable of reducing both $CO_2$ emissions and fuel consumption by 9%.

| Agent | $CO_2$ Emission (kg) | CO Emission (kg) | $NO_x$ Emission (g) |
|---|---|---|---|
| PG | 66.7 | 2.2 | 29.1 |
| DQN | 70.9 | 2.3 | 31.1 |
| Loss-Based | 73.8 | 2.6 | 32.3 |

**Table 3.** Statistical comparison of the sustainability measures in 1000 consecutive runs.

| Agent | $PM_x$ Emission (g) | HC Emission (g) | Fuel Consumption (L) |
|---|---|---|---|
| PG | 1.4 | 11.4 | 28.7 |
| DQN | 1.5 | 12.6 | 30.5 |
| Loss-Based | 1.6 | 13.7 | 31.7 |

FIGURE 26 (ALGORITHMS' PERFORMANCE ACCORDING TO ENVIRONMENTAL SUSTAINABILITY METRICS)

| Agent | Travel Time (s) | Waiting Time (s) | Queue Length (No. of Veh.) |
|---|---|---|---|
| PG | $141.3 \pm 42$ | $29.4 \pm 21$ | $10.8 \pm 8$ |
| DQN | $150.7 \pm 37$ | $37.8 \pm 22$ | $13.2 \pm 8$ |
| Loss-Based | $141.7 \pm 45$ | $46.8 \pm 49$ | $16.1 \pm 16$ |

FIGURE 27 (ALGORITHMS' PERFORMANCE ACCORDING TO TRADITIONAL PERFORMANCE METRICS)

## Martial and Methods

This questionnaire was created through Microsoft Form and was sent to drivers who own cars, whether private cars, buses, motorcycles, etc., and sent to all drivers in all governorates of Egypt and for all ages and surveyed their opinion on traffic congestion and how to solve some problems related to traffic congestion.

## Survey (Questionnaire)

**The survey could be viewed via the following link :**

- https://forms.office.com/r/9mtPG5r6Hi

**Questionnaire by QR CODE:**

## Survey Analysis

**The questionnaire results could be analysed through the following link:**

https://forms.office.com/Pages/AnalysisPage.aspx?AnalyzerToken=WxzCQ3X7bhbGffgaszPiedqV4S8W7b2A&id=JkCz8l6unEu-DEwmjc5t6a399i6U-PlHqM0azL3yvF1UNUFSMFlKOEdXREZNMjhTWVFMNk1XWjk0RS4u

## Conclusion

Smart traffic light is a technology that is expected to be able to solve congestion problems. The methods and technologies used in the development of smart traffic light to reduce traffic density and detect emergency vehicles as well as pedestrian are analysed in this paper. In the future, smart traffic light is expected to manage traffic independently to reduce the human error factor in this task.

# CHAPTER 3

# PLANNING AND ANALYSIS

# CHAPTER 3 Planning and analysis

## Software Requirements

## 3.1 Planning

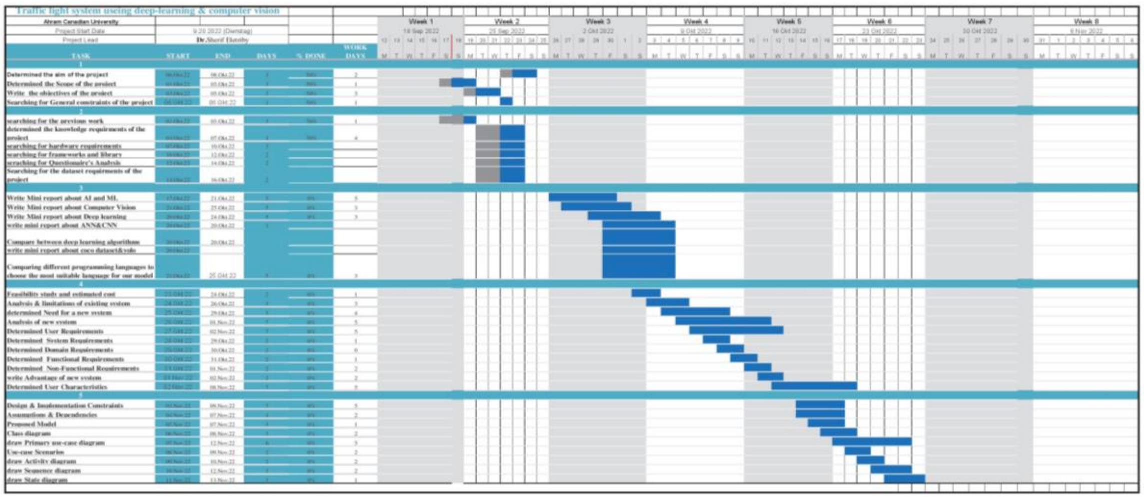
### 3.1.1 Feasibility study and estimated cost

The general goal of our project is to create a smart traffic light using artificial intelligence to reduce traffic congestion and preserve environment from pollution, reduce time spent in roads in traffic lights and Help emergency forces to reach their destinations on time.

Using a camera and a sensor to count average cars, and on the basis of that, the condition of the traffic light changes.

The proposal aims at reducing the traffic jams in order to reduce traffic congestion, optimize traffic flow and help proactively manage traffic conditions.

### 3.1.2 Gantt Chart



Traffic light system useing deep-learning & computer vision

| | |
|---|---|
| Ahram Canadian University | |
| Project Start Date | 9/20/2022 (Dienstag) |
| Project Lead | Dr.Sherif [illegible] |

| TASK | START | END | DAYS | % DONE | WORK DAYS |
|---|---|---|---|---|---|
| 1 | | | | | |
| Determined the aim of the project | [illegible] | 06 Oct 22 | [illegible] | [illegible] | 2 |
| Determined the Scope of the project | [illegible] | 03 Oct 22 | [illegible] | [illegible] | 1 |
| Write the objectives of the project | [illegible] | 05 Oct 22 | [illegible] | [illegible] | 3 |
| Searching for General constraints of the project | [illegible] | 05 Oct 22 | [illegible] | [illegible] | 1 |
| 2 | | | | | |
| searching for the previous work | [illegible] | 03 Oct 22 | [illegible] | [illegible] | 1 |
| determined the knowledge requirements of the project | [illegible] | 07 Oct 22 | [illegible] | [illegible] | 4 |
| searching for hardware requirements | [illegible] | 10 Oct 22 | [illegible] | | |
| searching for frameworks and library | [illegible] | 12 Oct 22 | [illegible] | | |
| serching for Questionnaire's Analysis | [illegible] | 14 Oct 22 | [illegible] | | |
| Searching for the dataset requirements of the project | [illegible] | 16 Oct 22 | [illegible] | | |
| 3 | | | | | |
| Write Mini report about AI and ML | 17 Oct 22 | 21 Oct 22 | [illegible] | [illegible] | 5 |
| Write Mini report about Computer Vision | 21 Oct 22 | 25 Oct 22 | [illegible] | [illegible] | 3 |
| Write Mini report about Deep learning | 20 Oct 22 | 24 Oct 22 | [illegible] | [illegible] | 3 |
| write mini report about ANN&CNN | 20 Oct 22 | 20 Oct 22 | [illegible] | | |
| Compare between deep learning algorithms | 20 Oct 22 | 20 Oct 22 | | | |
| write mini report about coco dataset&yolo | 20 Oct 22 | | | | |
| Comparing different programming languages to choose the most suitable language for our model | 20 Oct 22 | 25 Oct 22 | [illegible] | [illegible] | 3 |
| 4 | | | | | |
| Feasibility study and estimated cost | 23 Oct 22 | 24 Oct 22 | [illegible] | [illegible] | 1 |
| Analysis & limitations of existing system | 24 Oct 22 | 26 Oct 22 | [illegible] | [illegible] | 3 |
| determined Need for a new system | 25 Oct 22 | 29 Oct 22 | [illegible] | [illegible] | 4 |
| Analysis of new system | 26 Oct 22 | 01 Nov 22 | [illegible] | [illegible] | 5 |
| Determined User Requirements | 27 Oct 22 | 02 Nov 22 | [illegible] | [illegible] | 5 |
| Determined System Requirements | 28 Oct 22 | 29 Oct 22 | [illegible] | [illegible] | 1 |
| Determined Domain Requirements | 29 Oct 22 | 30 Oct 22 | [illegible] | [illegible] | 0 |
| Determined Functional Requirements | 30 Oct 22 | 31 Oct 22 | [illegible] | [illegible] | 1 |
| Determined Non-Functional Requirements | 31 Oct 22 | 01 Nov 22 | [illegible] | [illegible] | 2 |
| write Advantage of new system | 01 Nov 22 | 02 Nov 22 | [illegible] | [illegible] | 2 |
| Determined User Characteristics | 02 Nov 22 | 08 Nov 22 | [illegible] | [illegible] | [illegible] |
| 5 | | | | | |
| Design & Implementation Constraints | 04 Nov 22 | 09 Nov 22 | [illegible] | [illegible] | 5 |
| Assumptions & Dependencies | 04 Nov 22 | 07 Nov 22 | [illegible] | [illegible] | 2 |
| Proposed Model | 05 Nov 22 | 07 Nov 22 | [illegible] | [illegible] | 1 |
| Class diagram | 06 Nov 22 | 08 Nov 22 | [illegible] | [illegible] | 2 |
| draw Primary use-case diagram | 07 Nov 22 | 12 Nov 22 | [illegible] | [illegible] | 3 |
| Use-case Scenarios | 08 Nov 22 | 09 Nov 22 | [illegible] | [illegible] | 2 |
| draw Activity diagram | 08 Nov 22 | 10 Nov 22 | [illegible] | [illegible] | 2 |
| draw Sequence diagram | 10 Nov 22 | 12 Nov 22 | [illegible] | [illegible] | 2 |
| draw State diagram | 11 Nov 22 | 13 Nov 22 | [illegible] | [illegible] | 1 |

FIGURE 28 (GHANTT CHART OF THE PROJECT)

## 3.2 Analysis and limitations of existing system

Though there is a good body of research on the topic of smart traffic light management through images and sensors, you won't be able to find any publicly available models that perform with accuracy that's as good as what could be found in the literature. Further, no publicly available service or application is available to provide such a model to end users in an easy manner.
Given the lack of good, non-proprietary models, we've decided to go ahead and create our own models that can perform on par with the best in the literature. These models are designed to be useful in the Smart traffic light management , and are freely available to end-users in the open-source community.

## 3.3 Need for new system

Presence of an accurate system that helps reduce traffic congestion, improves the functions of traffic lights, benefits drivers, pedestrians, and emergencies, and helps reduce losses, whether material or human, by reducing accidents and saving time as well.

## 3.4 Analysis of the new system

### Purpose:

Creating a smart traffic light that facilitates traffic and reduces congestion and does not rely on traffic lights that depend on time and not the number of cars, congestion and emergency situations.

### Scope:

The general goal is to create a smart traffic light using artificial intelligence to reduce and save the time we consume on the roads, and thus reduce financial and environmental losses.

Using a camera and a sensor to count cars, and on the basis of that, the condition of the traffic light changes.

### Overview:

Intelligent traffic light to reduce traffic congestion through artificial intelligence and using a camera and sensor.

### General functions:

**Objective of user:**

Save time, effort and money by reducing traffic congestion caused by some traffic lights.

**User characteristic:**

The benefit from this project is not only for drivers, but for all people. It is not a requirement that you have a car to drive.

It is possible to be a passenger in a taxi or bus, so this project will benefit everyone, whether you have a car or not.

**Features and benefits:**

**For drivers:**

Reduce traffic congestion and thus reduce accidents.

**For pedestrians**:

Reducing the waiting time for pedestrians until the pedestrians signal is opened and not complying with a specified time for pedestrian traffic, for example, a men walking slowly, the pedestrian signal can calculate the speed and time of the mean’s arrival to the other side

**For emergency:**

Just picking up the camera and the sensor is an emergency, the traffic light is opened immediately without randomness in the decision, in order to avoid accidents and congestion, and thus reduce the loss of lives, money and time.

## 3.4.1 User Requirements:

User requirements mean the user’s needs and are provided by the software system. It is documented in the user requirements document. So they are written in a simplified form and ordinary language. In addition, she described the software system, the features it offers, and the limitations through which you can use it. And when the user's requirements are good, clear, and simplified, which leads to an increase in own quality and increased productivity.

As far as we see from the concern of users, from here we find that easy use and accuracy in the results are the basis. The user needs a program that works very efficiently and with high accuracy. The user cares about accuracy in results and speed.

One of the most important features that are necessary to create an attractive and high-quality software application. And see that we maintain all these qualities and features while planning to create the software application.

In the end, the developers will have to accept this reality and develop the software application so that it can achieve the expected results. They should consider the physical and other limitations of the users and the requirements of the application so that a good application can deliver what the user sees and wants.

### 3.4.2 System Requirements:

System requirements are the configuration that the system (application) must have for the application to work smoothly and efficiently.
failure to meet these requirements can lead to problems in installation or performance problems.
The first may prevent the installation of the application, while the other may cause disruption or performance below expectations or even suspend or disable it

From here it was discussed how to meet the different requirements of the application. It is worth noting that we will definitely need a remote server to run the inference tasks. If no remote server is selected to run the inference, the inference is made directly on the end user's device. This causes the GPU to be limited in the memory of the end user's device and cannot process many things at once. Therefore, you will be here for the need of a remote server to run the receiving tasks. This server is mainly dedicated to only one heuristic task and will communicate with the end user's device HTTP and some other technologies

### 3.4.3 Domain Requirements:

Main requirements are the requirements that characterize the application that will be created, and it is an ongoing process to proactively identify all the application-specific requirements that will be developed in the future.
We now need no well-disaggregated data for Streets and roads. We decided to compile several different data sets to minimize and alleviate this problem that was facing us

and help our model to achieve the best generalization, diversity of data, and accuracy thanks to diverse data sources.

The data set is a variety of real images of Streets and roads and their classification by a human expert. As with general classification, the classification algorithm will be based on a convolutional neural network using supervised learning with a deep learning framework.

Libraries for deep learning will be used (Tensorflow and Pytorch).

### 3.4.4 Functional Requirements:

The project is a smart traffic light using artificial intelligence. The camera is used to take pictures, and the pictures are sent to the model in the server, and the model detects the number of cars through the picture and puts it in a dataset and stores it.

The sensor counts the cars and sends the data to a server, and it is placed in a dataset and stored. The camera's data set is integrated with the data set sensor and stored.

Sending the integrated dataset to a model and through it the traffic light will change according to the number of cars

### 3.4.5 Non-Functional Requirements:

**Security:**
The security system must be strong to protect the server and the data stored on it, whether the dataset or model.

**Speed:**

The speed must be high in various operations, and that is fast results

**Capacity:**

It needs a large storage capacity for the various data contained in the dataset stored in the server.

**Reliability:**

Technology that is highly reliable functions with the same or similar efficiency after extensive use. Here are three ways you can assess a device's reliability:

- Percentage of the probability of failure: You can check the percentage of the probability of failure, or failure rate, to determine the reliability of a system. If the percentage is higher, the system is likely to function normally after substantial use.
- Number of critical failures: Consider recording the amount of critical failures a system experiences during testing to check its reliability. If the number of failures is low, it means that the system operates properly.
- Time between critical failures: Tracking the time between critical failures can help you understand the reliability of a system. When critical failures occur rarely, it means that a system functions normally most of the time.

## Hardware Requirement

### Types of Sensors:

LIDAR:
is a technology that can enhance traffic management systems by creating an accurate real-time 3D representation of roadways for multimodal analysis to detect number of Thus, high accuracy can be achieved in our model cars accurately

Let's take a look at a few ways LiDAR-based smart infrastructure solutions can be used.

security analysis: Smart infrastructure solutions using a single lidar sensor provide near-miss analysis that can be used to predict, diagnose and resolve road safety issues before the next collision occurs. Today's camera-based solutions require multiple cameras per intersection or identified public area, which often requires longer processing times for final analysis. Traffic studies are incomplete if they are only conducted at certain times or under certain conditions.

Transport efficiency and sustainability: These solutions provide reliable real-time traffic data to optimize traffic light timing based on congestion and throughput under various weather and light conditions. LiDAR-based solutions cover a variety of road users, including vulnerable pedestrians and cyclists, whereas current technology typically only provides data for vehicles.

Crowd Analytics: Smart infrastructure solutions can enable businesses and cities to increase revenue and improve infrastructure by providing foot traffic data analysis to learn more about traffic patterns, congested areas, congested points, and more. Understanding how people move and where they stop along the way is useful for designers, architects and urban planners.

Emergency Services: These solutions can detect collisions and near misses in real time, providing emergency services with data for faster deployment in both urban and rural environments. Wildlife protection. Smart infrastructure solutions can detect wildlife crossings and help prevent collisions, which often result in significant personal, environmental and economic loss, including personal injury, death, wildlife loss and vehicle damage.

### Type of Camera:

The CCTV

A camera used in most European countries for its high visibility and accuracy and thus will help us count cars accurately Thus, high accuracy can be achieved in our model

This will help the officers to keep a track of the events taking place on the roads.

CCTV cameras are present on busy roads, atop traffic signals, and located on highways with busy intersections. Cameras are used for monitoring traffic, recording traffic pattern data, and monitoring and observing traffic.

### Server Hardware Requirement:

- Processor: Minimum 1 GHz; Recommended 2GHz or more
- Ethernet connection (LAN) OR a wireless adapter (Wi-Fi)
- Hard Drive: Minimum 500 GB; Recommended 1 T or more
- Memory (RAM): Minimum 8 GB; Recommended 32 GB or above

## 3.5 Advantages of new system

The smart traffic management systems help to detect the congestion area and according to it, it helps to reduce congestion. For this action, it uses sensors data to analyze and synchronize. Traffic light blinks on the basis of congestion on given input data. Using this system it can reduce pollution and increase safety from road accident

# CHAPTER 4

# DESIGN

# CHAPTER 4 Design

## 4.1 Design and implementations Constraints

**a- Programming language constraints.**

**b- Server.**

**a- Programming language Constraints:**

- This project has restrictions, constraints on the programming language used.
- when we write machine learning programs we can use R, Python, Java, Julia..

We select python for:

**1- Python is easy to understand.**

Its readability, non-complexity, and ability for fast prototyping make it a popular language among developers and programmers around the world.

**2- Python comes with a large number of libraries.**

Many of these inbuilt libraries are for Machine Learning and Artificial Intelligence, and can easily be applied out of the box.

Some of the libraries are:

- **Scikit-Learn, Keras** for data mining, analysis, and Machine Learning.
- **Numpy** for high-performance scientific computing and data
- **Tensorflow**, a high-level neural network library.
- **pylearn2** which is also ideal for data mining and Machine Learning, but more flexible than scikit-learn.
- **Pandas** for general pupose data analysis
- **Scipy** for advanced computing
- **Seaborn** for data visualization
- **OpenCV** for computer vision

**3- Python allows easy and powerful implementation.**

If you only have basic knowledge of the Python language, you can already use if for Machine Learning because of the huge amount of libraries, resources, and tools available for you.

Additionally, you will spend less time writing code and debugging errors on Python than on Java or C++.

**b- Server constraints.**

## 4.2 Assumptions and dependencies

**1. Assumptions:**

- Project will follow waterfall methodology throughout execution.
- The team will write the algorithms in Python.

**2. Dependencies:**

- Python (PyTorch).
- Aimsun (simulations).
- Camera.
- Sensors.
- Hardware (computer board).

## 4.3 Risks & Risk Management:

**1. Traffic Light Malfunction**

- The malfunction leads to traffic congestion and road accidents.

**2. Not Everyone is Committed**

- Everyone's Commitment is the key to the project's success, and everyone should have the same goal to complete the project in the best possible way.

**3. Not Planning for the Long Term**

- Not planning caused many problems.

**4. Lack of Given Data**

- Lack of information collected may cause errors in the performance of tasks and this reduces the accuracy and efficiency of the application.

**5. Delayed Deliveries (Failing to Deliver on Time)**

- Failure to deliveries on time may result in badly estimated tasks, failed tests. Most of these issues can be solved with good planning.

## 4.4 Proposed Model

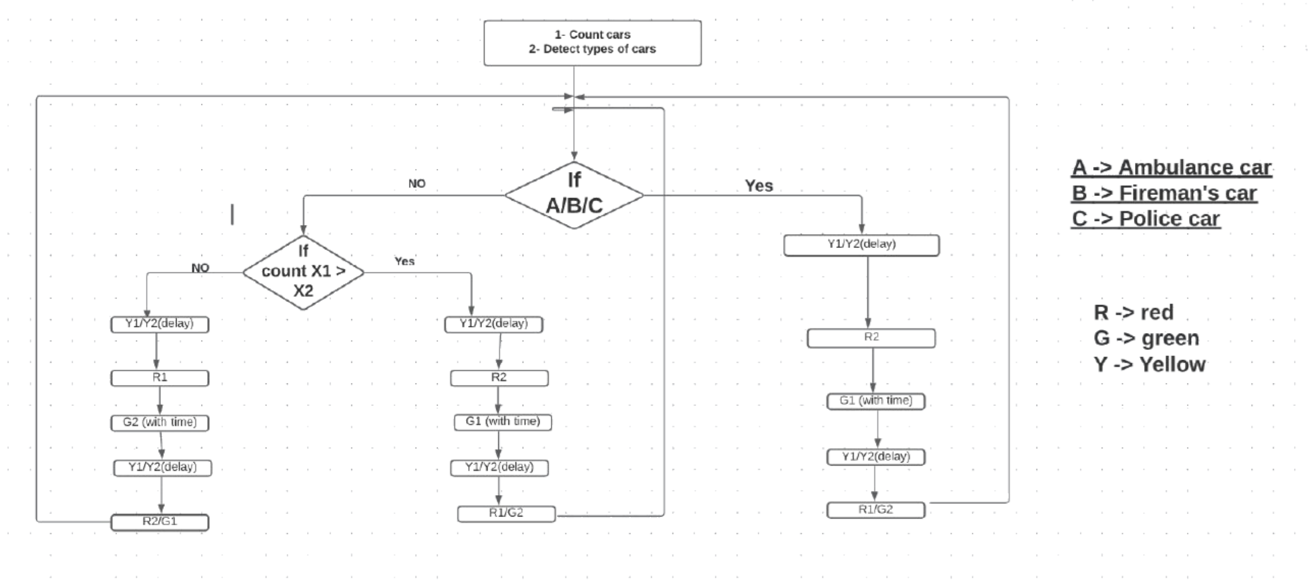


FIGURE 1 (LIFECYCLE FLOW CHART OF THE PROPOSED SYSTEM)

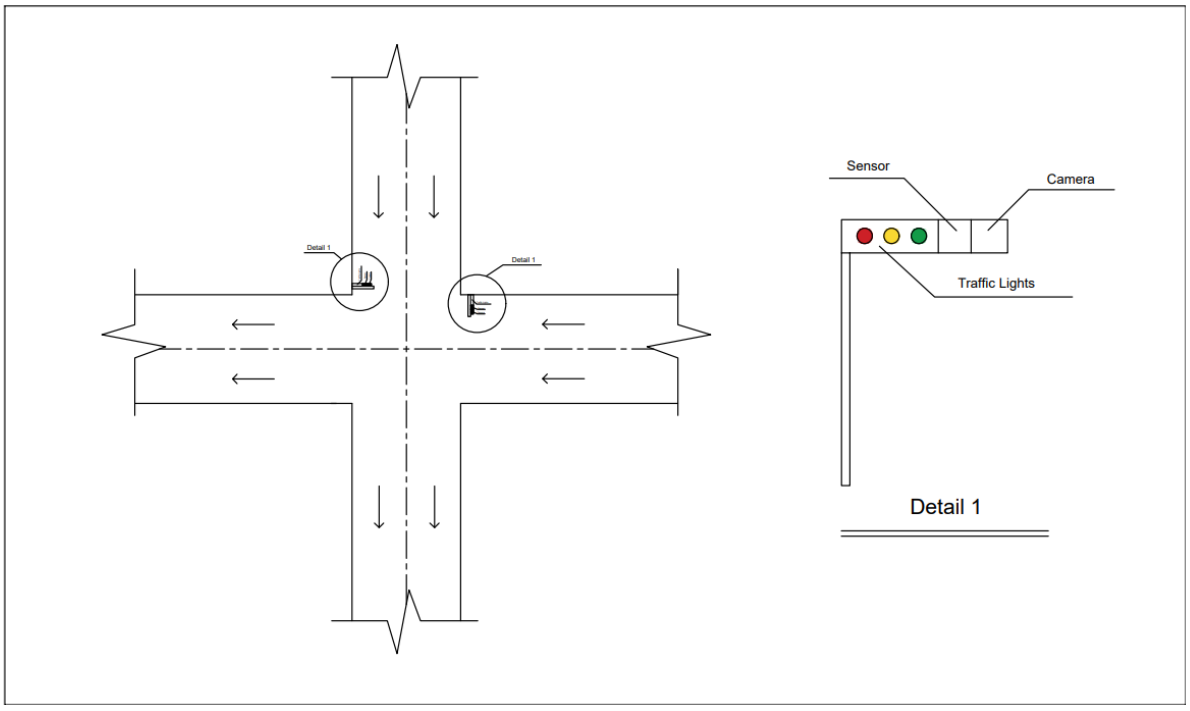


FIGURE 2 (ARCHITECTURE OF THE PROPOSED SYSTEM)

## 4.5 Use Case Diagram

### 4.5.1 Primary Use-Case Diagram

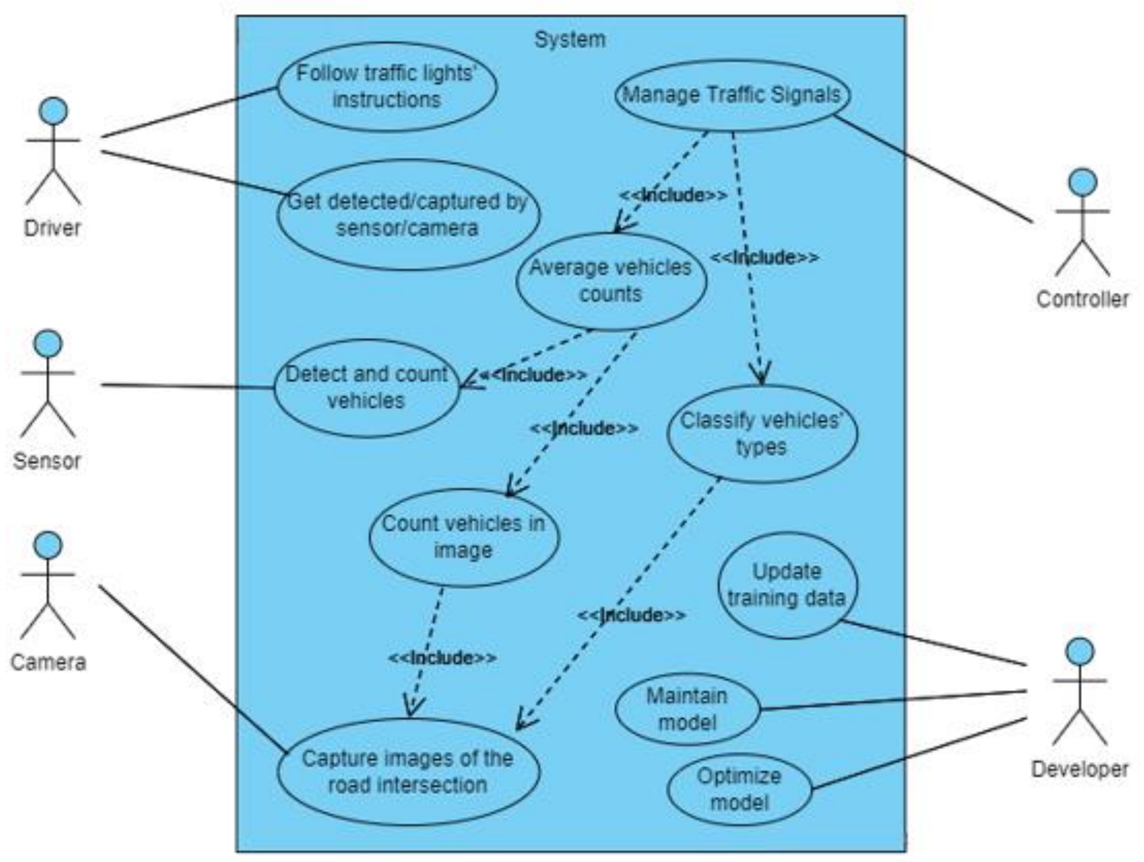


FIGURE 29 (USE-CASE DIAGRAM OF THE PROPOSED SYSTEM USING VISUAL PARADIGM)

### 4.5.2 Use-Case Scenarios

A use-case scenario of the proposed system would be of a driver passing through an intersection having the proposed traffic light management system installed.

As the **driver** arrives at the intersection, his car gets detected by the IR sensors or captured by the cameras set at the intersection.

Both **sensors and cameras** are used to guarantee accurate vehicle count, whereas the sensors detect each vehicle at both sides of the intersection, meanwhile the cameras capture images of the whole intersection. Using this data, the model could count the vehicles and determine what type of vehicles are at each side of the intersection. This data is then sent to the controller for it to manage the traffic signals accordingly.

The traffic light **controller** then accepts the vehicle counts and classification data generated by the model, then manage the traffic signals accordingly by opening the way for the side with higher priority.

After model deployment, the **developer** becomes responsible for model maintenance from time to time. He also tries to optimize it by using different techniques or parameters. In addition to that, he updates the data on which the model is trained whenever there is new-looking cars or new categories, in-order for the model to have up-to-date training data, and thus, achieve better performance.

**- Normal Traffic**

Figure 30 demonstrates a series of events during the normal traffic flow. During normal flow, signal 1 and signal 3 prompts red lights to turn on for 3 seconds, 3 seconds for yellow lights to signal the drivers and finally the green lights start for the time entered. Signal 1 and signal 3 work simultaneously until both goes on red to prompt the next cycle of traffic on the other lanes. When signal 1 and signal 3 are red, signals 2 and 4 will execute concurrently similar. The cycle then repeats itself in case of no interrupts such as pedestrian, busy mode or priority. The red lights are not assigned with timers, they stop when the yellow or green lights are on.

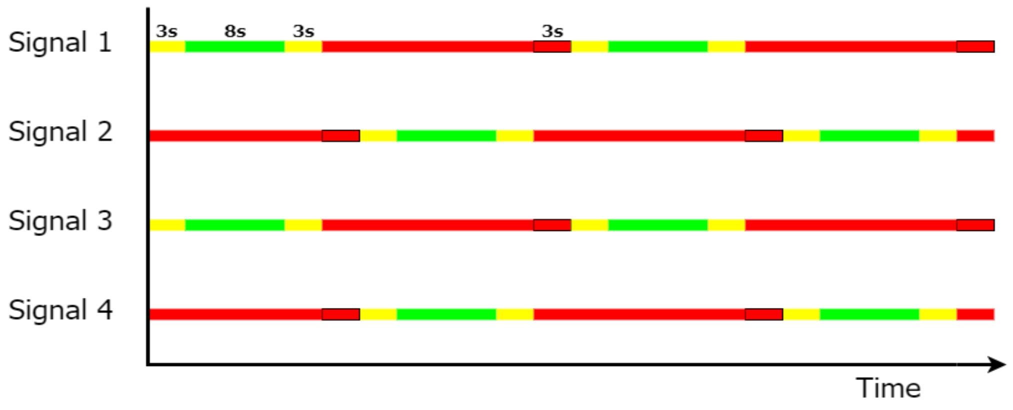


FIGURE 30 (NORMAL TRAFFIC SEQUENCE FLOW)

To put in another way, the end of the yellow light after the green light coincides with the beginning of the yellow light of the second Signal. This implies that the yellow lights can be synchronized to work at the same time, but due to the need to provide an allowance for a delay to avoid vehicle collisions at the intersection, this kind of synchronization is avoided. It can also be noticed that the red light does not have a timer because the driver is expected to stop whenever green or yellow lights are on. This cycle will continue repeating itself is no interruption is done due to the need for pedestrian or priority vehicle pass. In fact, the timing of the yellow and green lights is longer than 3 seconds and 8 seconds, respectively. The schedules shown above are for lab work only.  Setting the timing can quickly be done through the simulation's perimeter according to the desired timings for an appropriate intersection. Of course, the schedule will depend on the intersection conditions, such as whether it is usually a busy junction or not.

**- Pedestrian Pass**

The incorporation of the pedestrian traffic light interrupts the normal flow cycle, as shown in Figure 31.

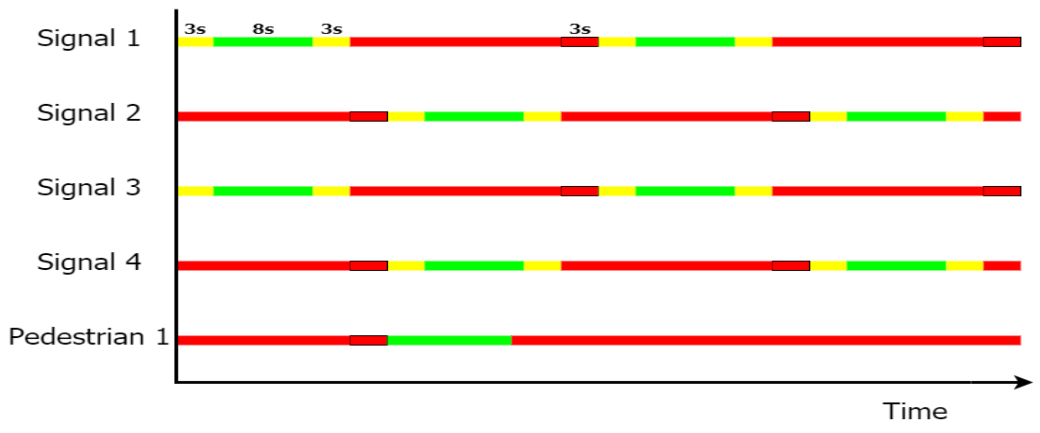


FIGURE 31 (PEDESTRIAN PASS IN NORMAL TRAFFIC FLOW)

The pedestrian signal will have to wait until the end of Signal 1 to turn the pedestrian light 1 green before Signals 2 and 4 start. Note that the pedestrian signal does not have yellow lights, only green and red. Also, the duration of the pedestrian green light can be set by operator, as an example 10 seconds. As a matter of fact, the allocation for pedestrian crossing is more extended because the pedestrians take a lot longer to cross the road than a vehicle exiting a junction. At the end of the 6 seconds of yellow and red lights delay signaling the end of Signals 1 and 3, the pedestrian green light will set on for 10 seconds then change to red. Signals 2 and 4 can begin parallel with the pedestrian green light. A closer observation of the cycles shows that the pedestrian signal can start at the end of any pair of signals. For example, if the junction experiences heavy pedestrian crossing, the pedestrian light can begin at the end of each Signals cycle. Since the sequences are controlled by the programmable logic controller, it is easy to program the device to allow more frequent pedestrian crossings at the intersection.

**- Busy Traffic**

The road situation adjacent to the intersection can be either a dense traffic flow or a sparse one. As such, the computation in the PLC assumes the Boolean inputs of the signal in which a set of decisions will take place as far as the duration of signals is concerned. Consequently, the period of the green light should increase on the side of the road that is experiencing high density, as shown in Figure 32.

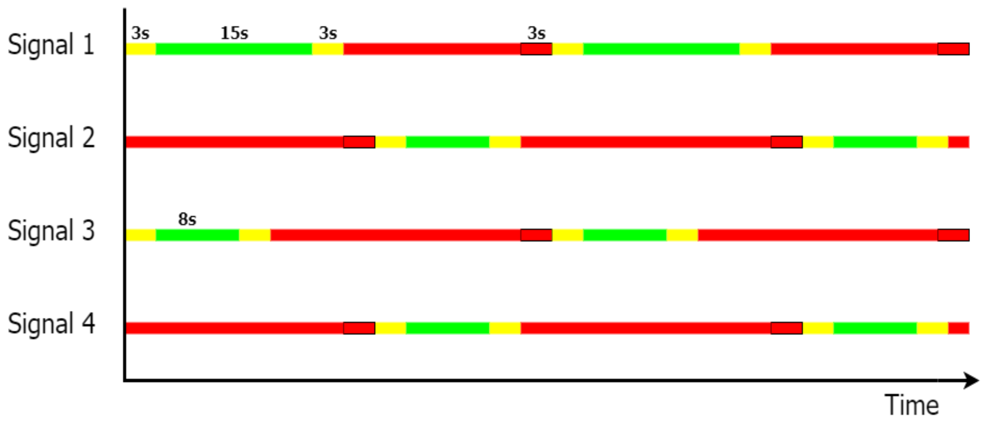


FIGURE 32 (BUSY TRAFFIC SEQUENCE FLOW)

If there is a busy traffic flow associated with Signal 1 as shown in Figure 32, vehicles can pass the intersection a little longer, i.e., in 15 seconds instead of 8 seconds. As explained before, the inductive sensors placed on the road will provide the input signal in the real-life situation to determine the density of traffic associated with Signal 1. The time allocation, therefore, will vary depending on the density of traffic related to Signal 1.

**- Priority Vehicle Pass**

Since a vehicle with a priority pass is allowed to pass the intersection at any moment, all signals are interrupted as both pedestrians, and other vehicles have to give way to the priority vehicle pass. Therefore, whenever such a case arises, the yellow light should be on for one second in all directions followed by a green light in the direction of the oncoming priority vehicle, but red lights for all the other routes to allow the priority vehicle to pass.

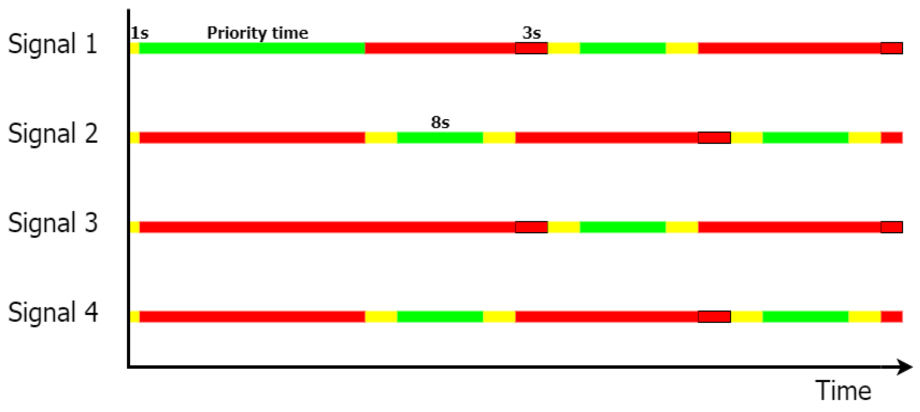


FIGURE 33 (PRIORITY VEHICLE PASS SEQUENCE FLOW)

Figure 33 demonstrates what happens when a priority vehicle approaches in the direction of Signal 1. The yellow traffic light signal set for one second in each direction to alert the other drivers at the intersection, and then the green traffic light for the direction of the priority vehicle will switch on while the lights in all the different routes, including the pedestrian lights will go red. What happens during the priority vehicle pass is that all the other lights pause and will continue from where they left as soon as the priority vehicle passes the intersection.

In another scenario, there may be more than one case of vehicle pass priority at the same time. The vehicle that triggers the signal first will be allowed to pass before the priority circle begins for the second case to avoid vehicle collisions. In other words, only the priority cycle will be activated during the priority vehicle pass request scenario before the normal cycles resume.

## 4.6 Activity Diagram

Activity diagram is the lifecycle of the project.

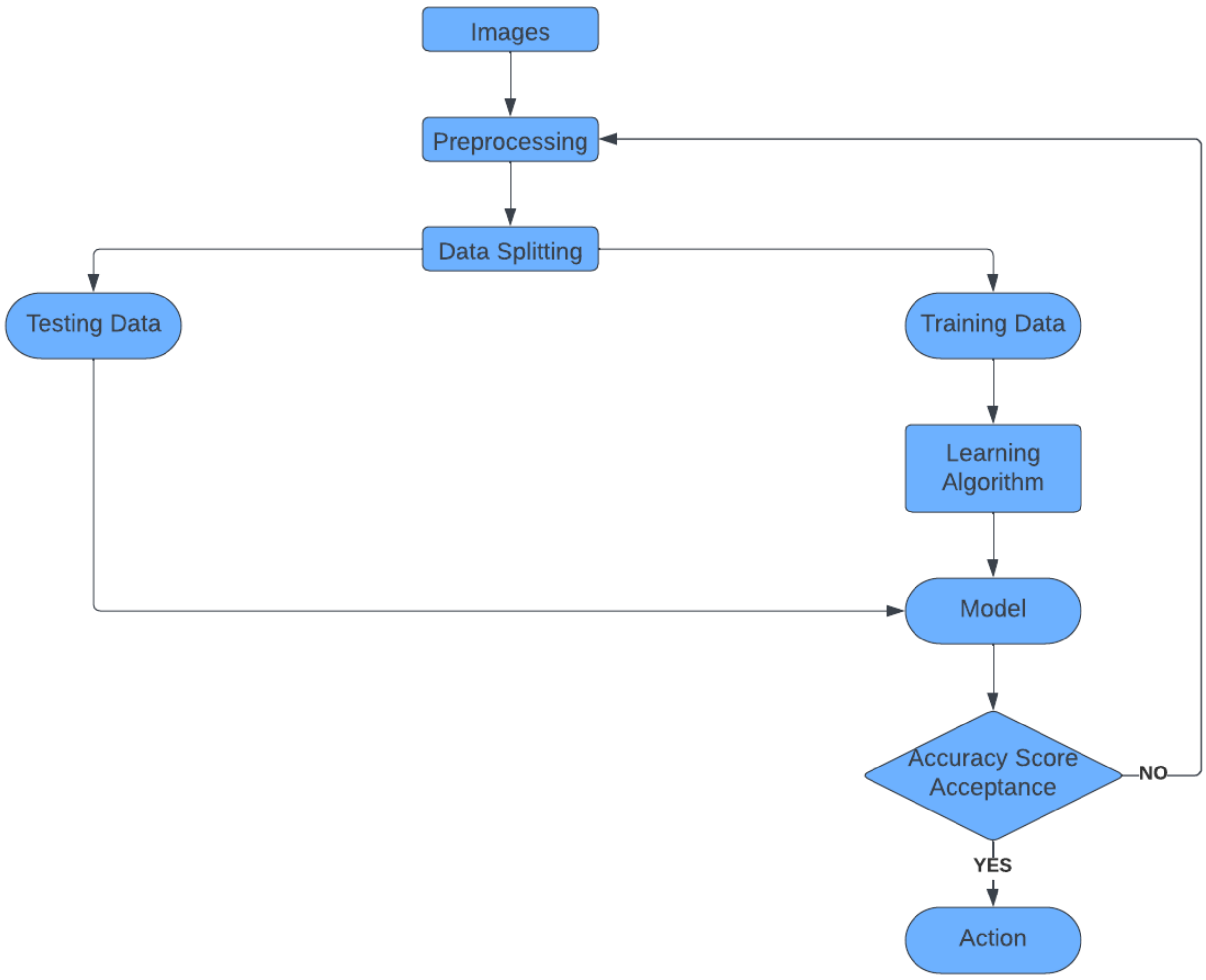


FIGURE 34 (ACTIVITY DIAGRAM OF THE PROPOSED SYSTEM)

## 4.7 Sequence Diagram

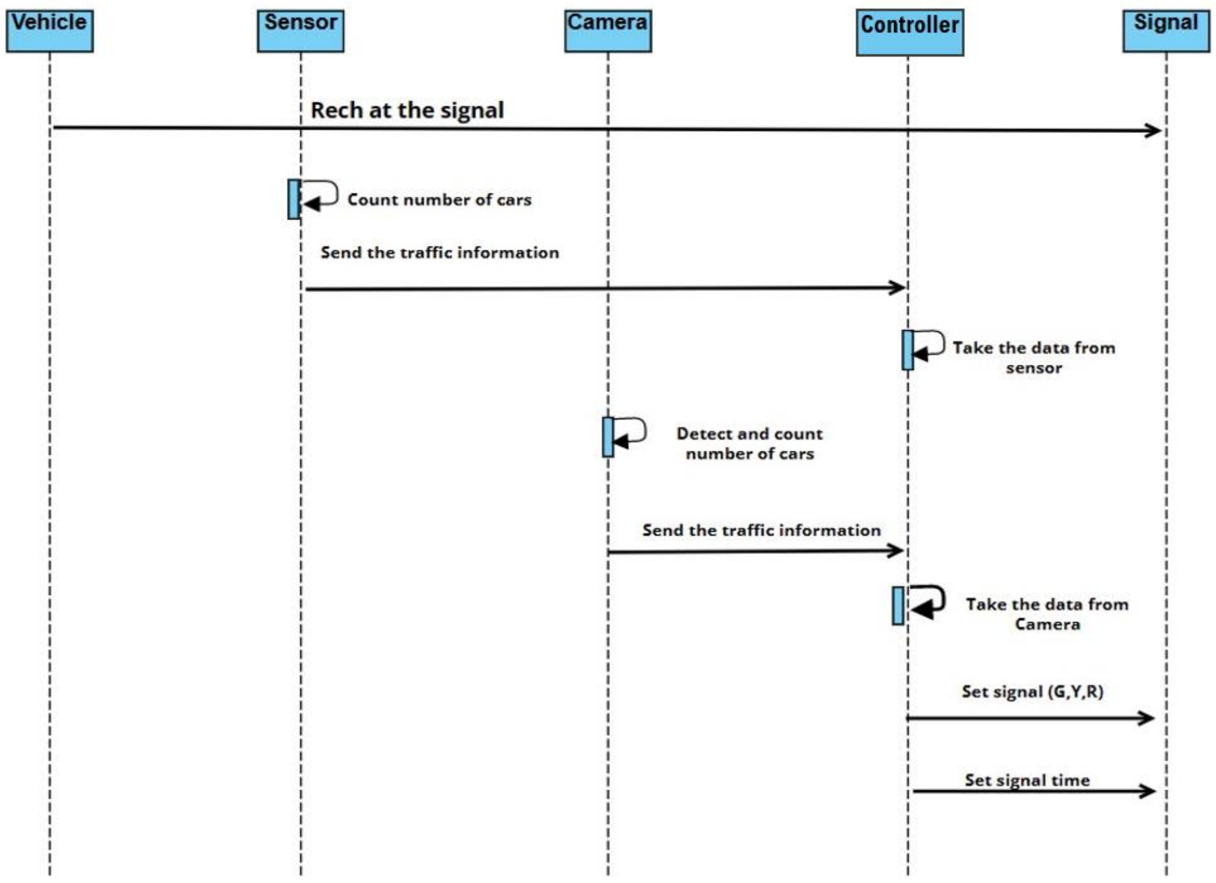

Vehicle
Sensor
Camera
Controller
Signal
Rech at the signal
Count number of cars
Send the traffic information
Take the data from sensor
Detect and count number of cars
Send the traffic information
Take the data from Camera
Set signal (G,Y,R)
Set signal time

FIGURE 35 (SEQUENCE DIAGRAM OF THE PROPOSED SYSTEM)

Unified Modeling Language (UML) diagram that illustrates the sequence of messages between objects in an interaction.

They collaborate together by capturing the interaction between objects, Sequence Diagrams are time focus, and they sequence the interaction visually using the vertical axis of the diagram to represent the time of when and what messages are sent.

Sequence Diagrams concerned with:

- the interaction that takes place in a collaboration that either realizes a use case or an operation (instance diagrams or generic diagrams)
- high-level interactions between actor and system, between system and other systems, or between subsystems

Main purpose of Sequence Diagram:

- Model high-level interaction between active objects in a system.

- Model the interaction between objects within a collaboration that realizes an operation.
- Either model generic interactions (Showing all possible paths through the interaction) or specific instances of an interaction (showing just one path through the interaction).

That figure shows the sequence diagrams in our architecture that explain the interaction from application until getting the data from server.

## 4.8 State Diagram

In addition to being a direct result of its inputs, an entity's behavior also depends on its previous state. A finite state machine diagram, also known as an automate UML state machine diagram, is the ideal tool for modelling an entity's history. These diagrams, which are also sometimes called state diagrams, state machines, or state charts, display an entity's various states. State machine diagrams can also demonstrate how an entity reacts to different events by shifting between different states. A state machine diagram is a UML diagram used to represent a system's dynamic nature.

Here we will show the state diagrams in our architecture that explain when application will be active and deactivate.

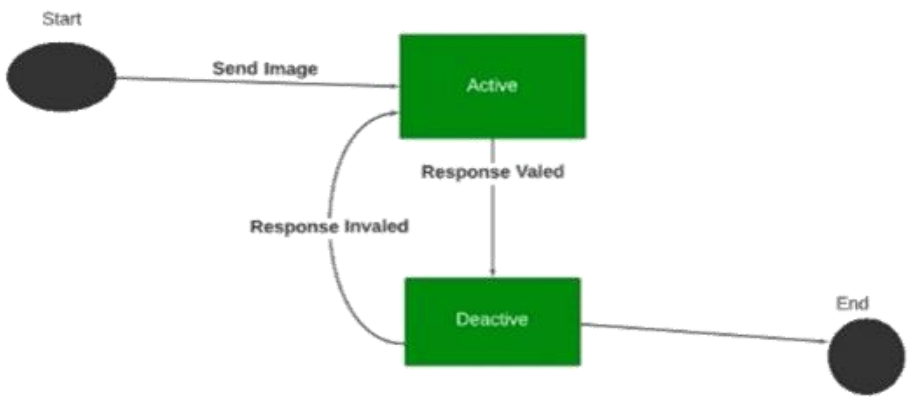


FIGURE 29 (STATE DIAGRAM OF THE PROPOSED SYSTEM)

# CHAPTER 5

## Implementation

# CHAPTER 5 Implementation

## 5.1 Software architecture

### Implementation:

YOLO (You Only Look Once) is a deep learning algorithm for object detection in images and videos. It was first introduced in 2016 by Joseph Redmon, and it has since become one of the most popular object detection algorithms in use today, algorithm works by dividing an image into a grid of cells and predicting bounding boxes and class probabilities for each cell. The algorithm uses a convolutional neural network (CNN) to extract features from the image, which are then used to make predictions. The YOLO algorithm is unique in that it can perform object detection in real-time, with the ability to process images at a rate of over 45 frames per second on a standard GPU. This makes it well-suited for applications that require real-time object detection, such as self-driving vehicles and surveillance systems. One of the key advantages of the YOLO algorithm is its ability to detect multiple objects in an image with high accuracy. The algorithm can detect objects of different sizes and aspect ratios, and it is robust to occlusion and clutter in the image, another advantage is its efficiency. Because the algorithm only needs to make a single pass through the image, it is faster than other object detection algorithms that use a sliding window approach.

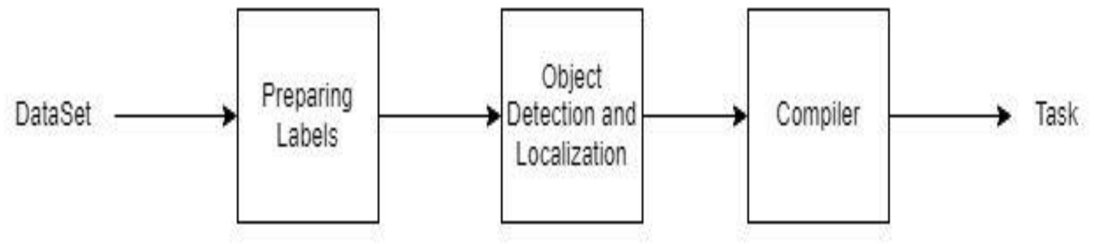


Figure 30 (Software Architecture)

We take photos using the camera, then pass them to the model, which we trained using a large data set. After this, emergency cars are extracted from the photo using Yolo v8. then the tokenizer knows the gesture, and the parser will translate these tokens into a specific task).

### 5.1.1 Preparing labels

To train a YOLO (You Only Look Once) object detector, the dataset needs to be in a specific format. The dataset should consist of images along with their corresponding annotation files. Each annotation file should have the same name as its corresponding image file and contain information about the objects present in the image. The annotations should be in the format of bounding boxes that represent the location of each object within the image. The bounding boxes should be specified in (x center, y center, width, height) format for bounding box annotations. In this format, the (x center, y center) coordinates represent the center of the bounding box, while the width and height specify its dimensions after normalization.

The dataset should also include a file that lists the classes of objects present in the dataset, and each class should be assigned a unique ID. With this format, the YOLO object detector can accurately recognize and localize objects in new images.

To train Yolo, we need the coordinates of the center of the bounding box, and the width and height need to be normalized.

### 5.1.2 Object detection and Localization

We used YOLOv8 nano.  YOLOv8 is the latest and most advanced iteration of the highly popular real-time object detection system, it has undergone continuous improvements and refinements to deliver exceptional performance in terms of speed, accuracy, and efficiency. With a focus on pushing the boundaries of real-time object detection, YOLOv8 introduces several groundbreaking advancements that make it a top contender in the field of computer vision. Here are some of the most notable **features of YOLOv8:**

**Enhanced Backbone Network:** The YOLOv8 architecture employs a re- designed backbone network that is more efficient and powerful than its predecessors. This new backbone enables the model to extract complex features from images, significantly improving detection accuracy while maintaining low latency.

**Advanced Data Augmentation Techniques:** YOLOv8 incorporates state- of-the-art data augmentation techniques to enrich the training dataset, ensuring that the model is robust and can generalize well to real-world scenarios. These techniques include advanced geometric transformations, color space augmentations, and more.

**Multi-Scale Feature Fusion:** YOLOv8 introduces a novel multi-scale feature fusion mechanism that combines features from various layers of the network. This approach allows the model to utilize both low-level and high-level features, enabling it to detect objects of different sizes and scales more accurately.

**Optimized Anchor Boxes:** The anchor box system in YOLOv8 has been refined and optimized, resulting in improved detection of objects with various aspect ratios. This enhancement plays a crucial role in reducing false positives and increasing overall detection accuracy.

**Efficient Model Pruning:** YOLOv8 employs a sophisticated model pruning strategy that removes redundant and less important connections within the neural network. This process significantly reduces the model size and computational requirements without compromising detection performance.

**Improved Training Techniques:** The training process of YOLOv8 has been enhanced with the latest optimization algorithms, loss functions, and learning rate schedulers. This makes the model converge faster while maintaining high accuracy across a wide range of object classes.

**Real-Time Inference on Edge Devices:** YOLOv8 is optimized for deployment on edge devices, such as smartphones, drones, and embedded systems. The efficient model architecture, coupled with advanced model compression techniques, enables YOLOv8 to deliver real-time object detection even on resource- constrained devices.

### 5.1.3 Summary

In Experiments 1,2,3 we used YOLOv7 tiny which contains 6055578 parameters and 263 layers, we trained our model with 20, 30 epochs, the batch size was 16, and the image size was 416*416, and in other experiments we used YOLOv8 nano which contains 3013968 parameters and 225 layers, we trained the model for 20,30 epochs, so it gave us better performance and higher fps.

| **Experiments** | **Model** | **mAP@.5** | **mAP@.5:.95** |
|---|---|---|---|
| Experiment 1 | YOLOv7-tiny | 0.7272 | 0.5104 |
| Experiment 2 | YOLOv7-tiny | 0.9527 | 0.7124 |
| Experiment 3 | YOLOv7-tiny | 0.9518 | 0.7084 |
| Experiment 4 | YOLOv8-nano | 0.98614 | 0.91705 |
| Experiment 5 | YOLOv8-nano | 0.92784 | 0.78957 |
| Experiment 6 | YOLOv8-nano | 0.94321 | 0.80924 |
| Experiment 7 (best) | YOLOv8-nano | 0.91989 | 0.7667 |

**Table 5.1:** *summarizing results*

## 5.2 Experiments

**Experiment 1**: We used specific classes from Microsoft COCO dataset by writing a code to extract them form the dataset and added emergency cars classes (8 classes) then we wrote a code to modify class id's after that we wrote a code to split data to train and validation, the result was 34428 of augmented images for Train and 11136 images for Validation, after that we pushed photos to YOLOv7 tiny for 20 epochs, after training finished, we got for all classes mAP@.5=0.7272 and mAP@.5:.95=0.5104 and for each class:

| Class | mAP@.5 | mAP@.5:.95 |
|---|---|---|
| Police | 0.98 | 0.672 |
| Ambulance | 0.91 | 0.613 |
| Fire truck | 0.94 | 0.621 |
| Car | 0.56 | 0.326 |
| Motor cycle | 0.62 | 0.452 |
| Truck | 0.39 | 0.271 |
| Bicycle | 0.45 | 0.313 |
| Bus | 0.69 | 0.472 |

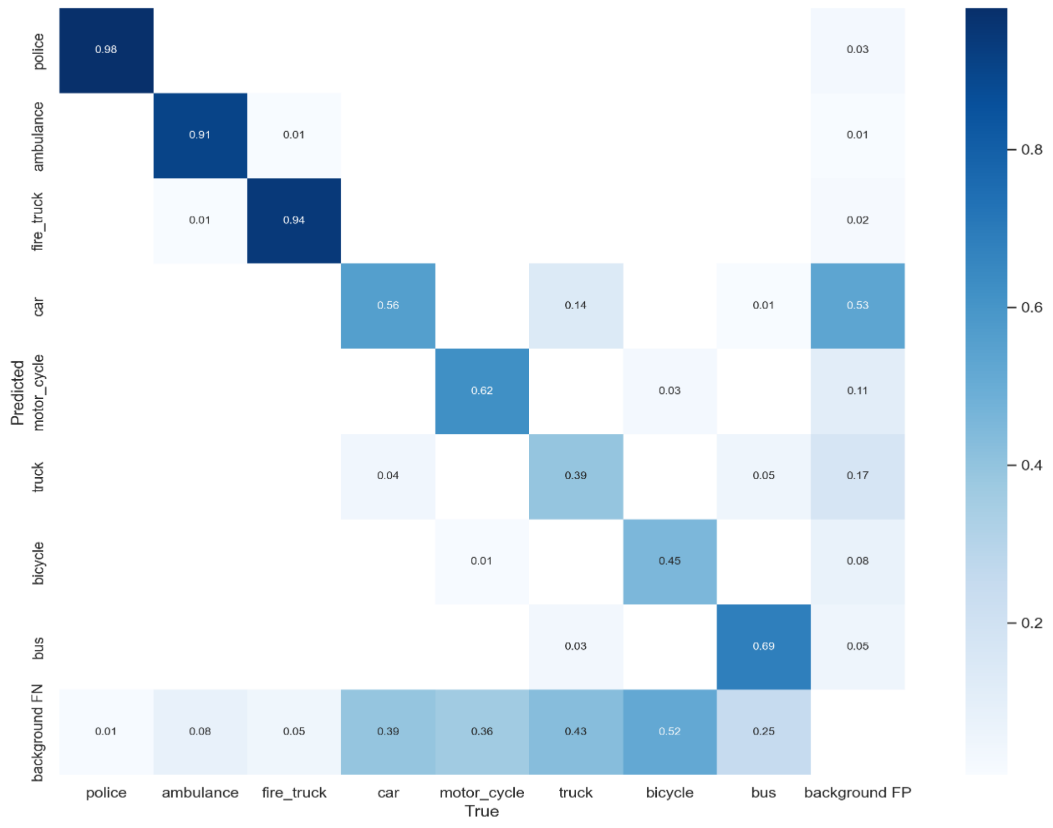


Figure 40 (*Experiment 1 confusion matrix)*

*At the end of this experiment, we did not find it important that the model sees all these types of vehicles.*

**<u>Experiment 2</u>**: We have used only emergency cars classes (3 classes) with 20220 of augmented images for Train and 5042 images for Validation, we have pushed photos to YOLOv7 tiny for 20 epochs, after training finished, we got for all classes mAP@.5=0.9527 and mAP@.5:.95=0.7124 and for each class:

| Class | mAP@.5 | mAP@.5:.95 |
|---|---|---|
| ambulance | 0.90 | 0.57 |
| police | 0.98 | 0.557 |
| Fire truck | 0.91 | 0.561 |

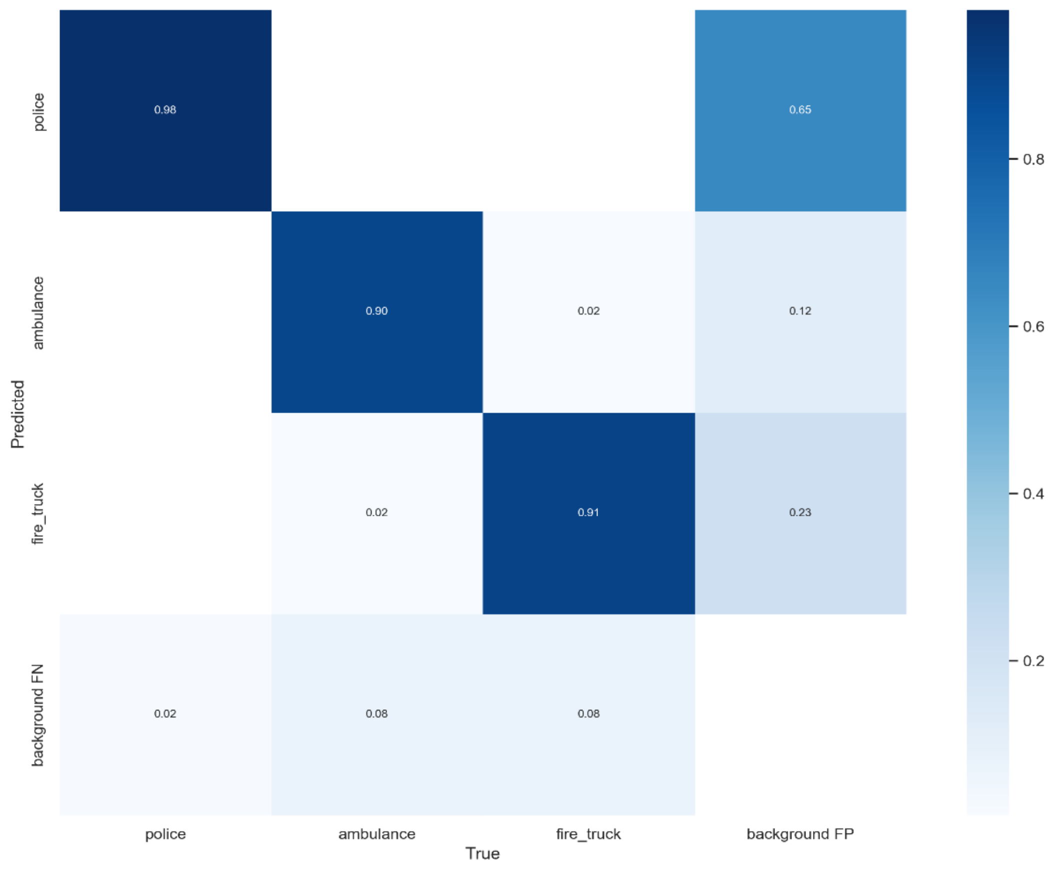


**Figure 41 (*Experiment 2 confusion matrix*)**

*phase were not good enough  At the end of this experiment, the results in testing*

**<u>Experiment 3</u>**: We have used emergency cars Dataset that contains 3 classes with 20220 of augmented images for Train and 5042 images for Validation, we have pushed photos to YOLOv7 tiny for **30** epochs, after training finished, we got for all classes mAP@.5=0.9518 and mAP@.5:.95=0.7084 and for each class:

| Class | mAP@.5 | mAP@.5:.95 |
|---|---|---|
| ambulance | 0.84 | 0.532 |
| police | 0.98 | 0.546 |
| Fire truck | 0.89 | 0.561 |

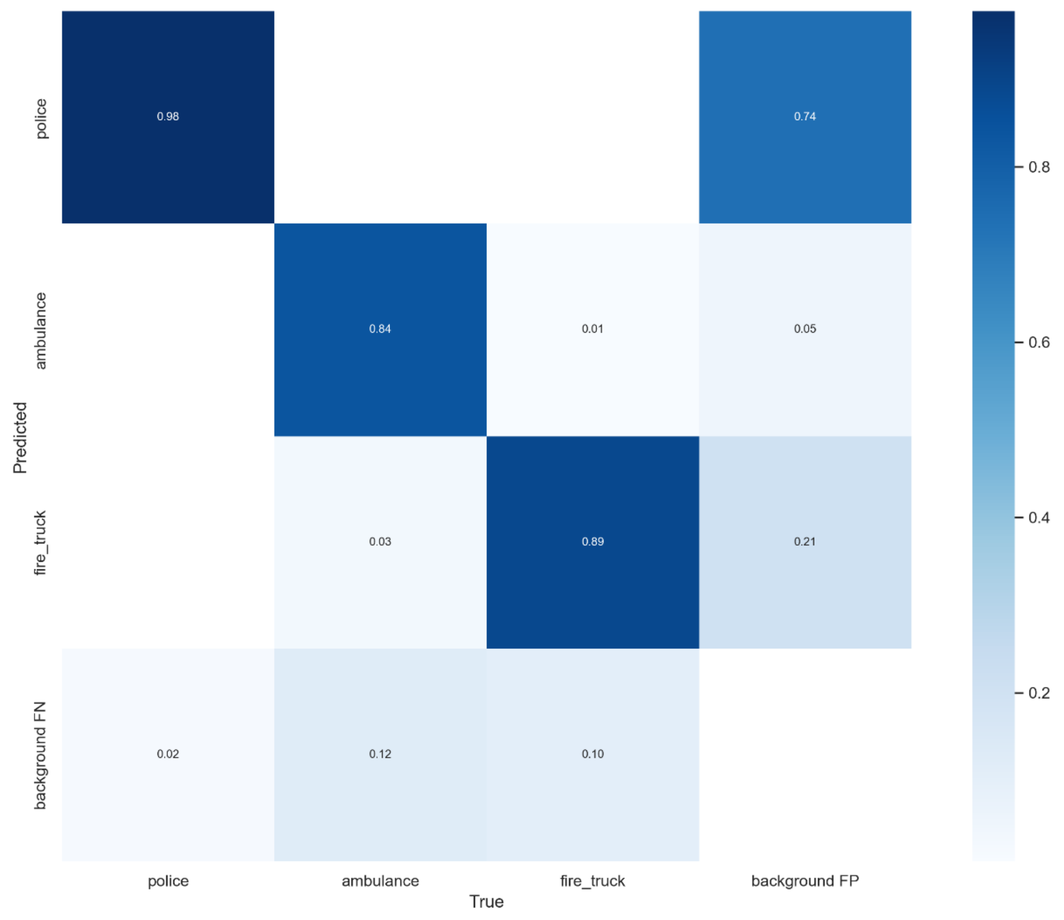


Figure 42 (*Experiment 3 confusion matrix*)

*phase were not good enough At the end of this experiment, the results in testing*

**<u>Experiment 4</u>**: here a new version of yolo was released so we used emergency cars Dataset that contains 3 classes with 20220 of augmented images for Train and 5042 images for Validation, we have pushed photos to YOLOv8 nano for 20 epochs, after training finished, we got for all classes mAP@.5=0.98614 and mAP@.5:.95=0.91705 and for each class:

| Class | mAP@.5 | mAP@.5:.95 |
|---|---|---|
| ambulance | 0.96 | 0.632 |
| police | 1.00 | 0.746 |
| Fire truck | 0.97 | 0.661 |

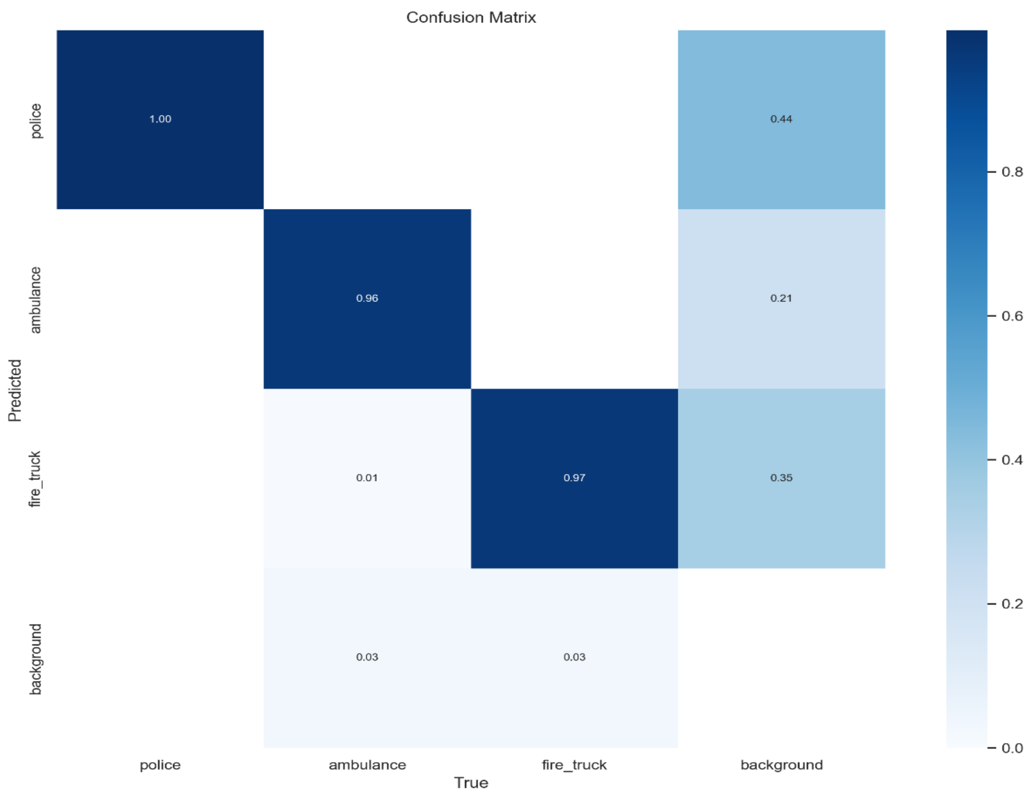


**Figure 43 (*Experiment 4 confusion matrix*)**

*phase were not good enough and At the end of this experiment, the results in testing we discovered a problem in the relation between the classes and the background.*

**<u>Experiment 5</u>**: We modified the data by adjusting the object's bounding box in the annotations and made some changes to the images by removing poorly captured images due to background issues. then we returned the images without augmentation. We used emergency cars classes and another class for other types of vehicles (4 classes) with 1528 images for Train and 420 images for Validation, we have pushed photos to YOLOv8 nano for 20 epochs, after training finished, we got for all classes mAP@.5=0.92784 and mAP@.5:.95=0.78957and for each class:

| Class | mAP@.5 | mAP@.5:.95 |
|---|---|---|
| ambulance | 0.81 | 0.532 |
| police | 0.96 | 0.646 |
| Fire truck | 0.95 | 0.611 |
| Other vehicles | 0.84 | .5213 |

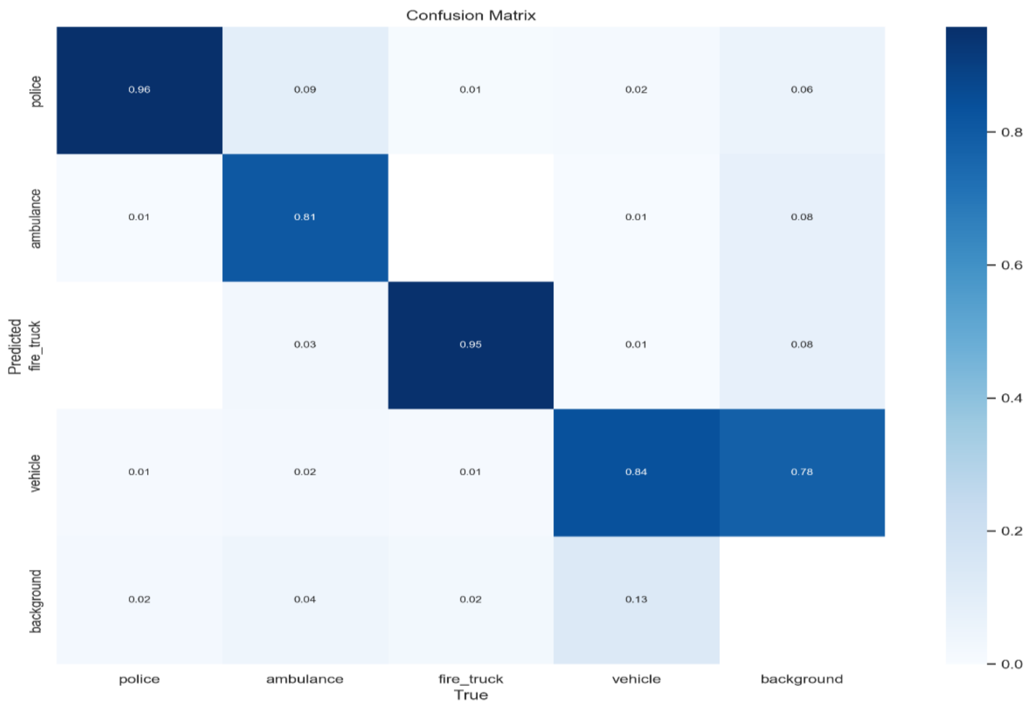


**Figure 44 (*Experiment 5 confusion matrix*)**

*phase were becoming better. At the end of this experiment, the results in testing*

**<u>Experiment 6</u>**: We used emergency cars classes and another class for other types of vehicles (4 classes) with 2120 images for Train and 531 images for Validation, we have pushed photos to YOLOv8 nano for 50 epochs, after training finished, we got for all classes mAP@.5=0.94321 and mAP@.5:.95=0.80924 and for each class:

| Class | mAP@.5 | mAP@.5:.95 |
|---|---|---|
| ambulance | 0.95 | 0.632 |
| police | 0.81 | 0.546 |
| Fire truck | 0.92 | 0.611 |
| Other vehicles | 0.88 | .5213 |

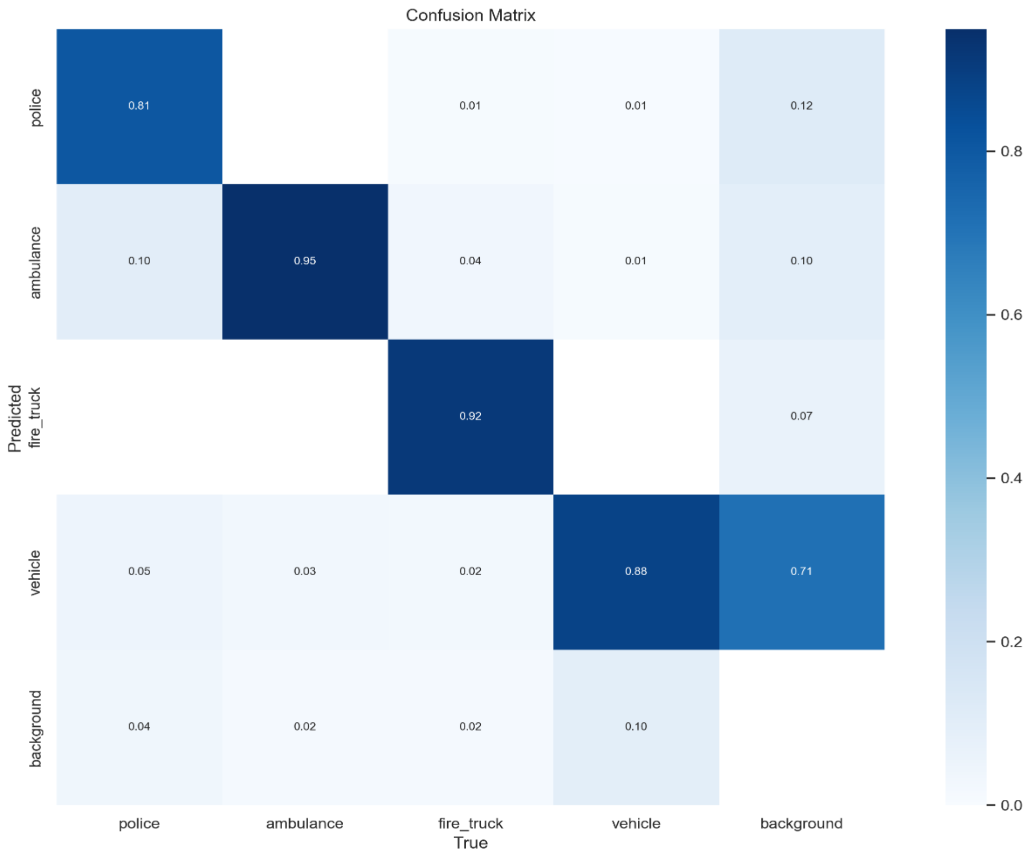


**Figure 45 (*Experiment 6 confusion matrix*)**

**<u>Experiment 7</u>**: We used emergency cars classes and another class for other types of vehicles (4 classes) with 4426 of augmented images for Train and 772 images for Validation, we have pushed photos to YOLOv8 nano for 30 epochs, after training finished, we got for all classes mAP@.5=0.91989 and mAP@.5:.95=0.7667and for each class:

| Class | mAP@.5 | mAP@.5:.95 |
|---|---|---|
| ambulance | 0.82 | 0.532 |
| police | 0.90 | 0.596 |
| Fire truck | 0.94 | 0.611 |
| Other vehicles | 0.85 | .5213 |

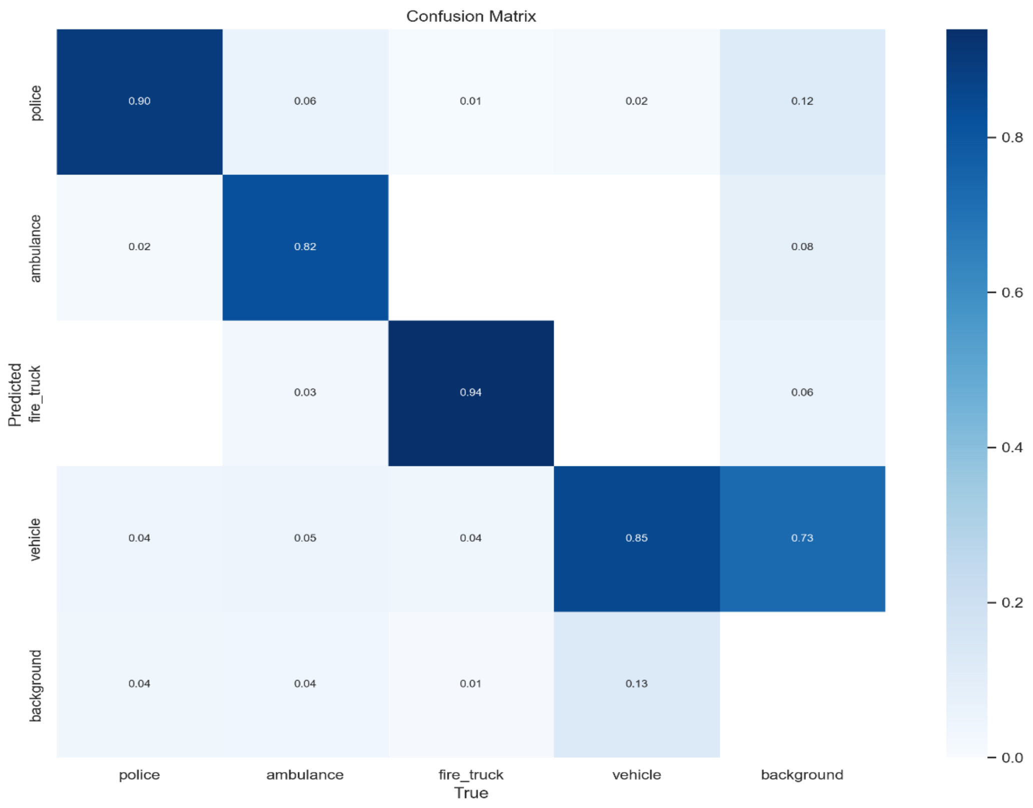


Figure 46 (*Experiment 7 confusion matrix*)

The experience we choose is experience 7, in this model we have used YOLOv8n (nano), we trained model using our generated dataset that contains 4 classes 3 for emergency cars and one for all other types of vehicles with 4426 images for Train and 772 images for Validation. The model contains 3013968 parameters and 225 layers. We trained the model for 30 epochs.

The advantages of yolo v8 that the model size is smaller in v8 nano and number of parameters in it is less than in yolov7 and it is more efficient and powerful and it can extract complex features from the image and it allows the model to utilize both low- and high-level features and detect objects of different sizes.

We also used YOLOv7 tiny and gave us good results but YOLOv8 nano gave us better results than YOLOv7 tiny, knowing that YOLOv8 nano's parameters are about half of YOLOv7 tiny which gives us higher fps.

# CHAPTER 6

Testing

# CHAPTER 6 Testing

## 6.1 Unit Testing

**Unit testing** is a software development practice that involves testing individual units, or the smallest testable components, of a software application in isolation. These units are typically functions, methods, or classes that perform specific tasks within the software.

**The primary purpose** of unit testing is to verify that each unit of the software performs as expected and produces the correct output given a particular input. By testing units in isolation, developers can identify and fix defects or errors at an early stage, preventing them from propagating to other parts of the codebase and causing more significant issues.

### Maquette and simulation:

- We have applied more than test case scenario in our project to make sure there is no error.

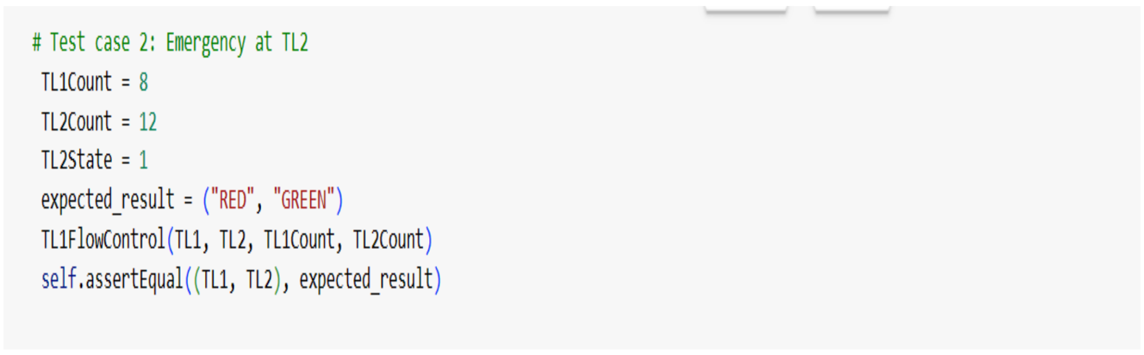
#### Test case 1:

```
def test_TL1FlowControl(self):
    TL1 = TL2 = "RED"

    # Test case 1: Normal flow control
    TL1Count = 10
    TL2Count = 5
    expected_result = ("GREEN", "RED")
    TL1FlowControl(TL1, TL2, TL1Count, TL2Count)
    self.assertEqual((TL1, TL2), expected_result)
```

test_TL1FlowControl_NormalFlow: This test case verifies the normal flow control behavior of TL1. It sets up TL1 and TL2 to "RED" state and assigns vehicle counts for TL1 and TL2. The expected result is that TL1 should transition to "GREEN" and TL2 should remain "RED" after the flow control operation.

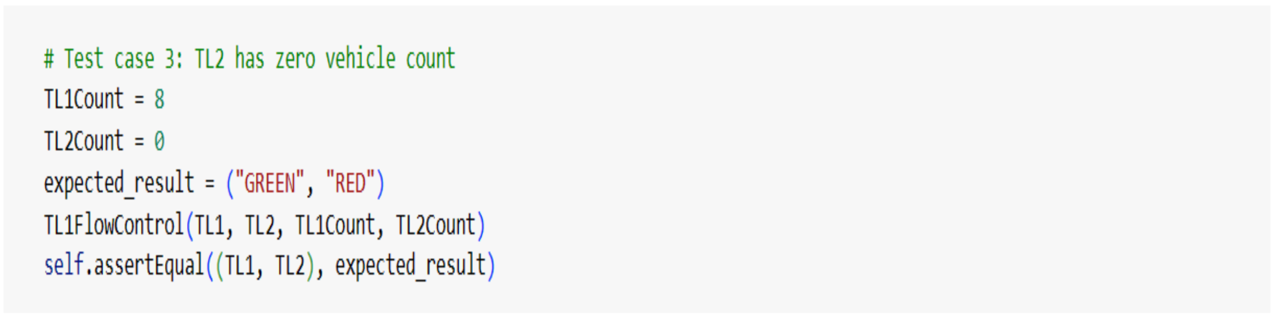
## Test case 2:



```
# Test case 2: Emergency at TL2
 TL1Count = 8
 TL2Count = 12
 TL2State = 1
 expected_result = ("RED", "GREEN")
 TL1FlowControl(TL1, TL2, TL1Count, TL2Count)
 self.assertEqual((TL1, TL2), expected_result)
```

test_TL1FlowControl_EmergencyAtTL2: This test case checks the flow control behavior of TL1 in the presence of an emergency at TL2. Similar to the previous test case, TL1 and TL2 are initially in the "RED" state, but this time the TL2State is set to 1 to indicate an emergency. The expected result is that TL2 should transition to "GREEN" and TL1 should remain "RED" since TL2State is in an emergency state.

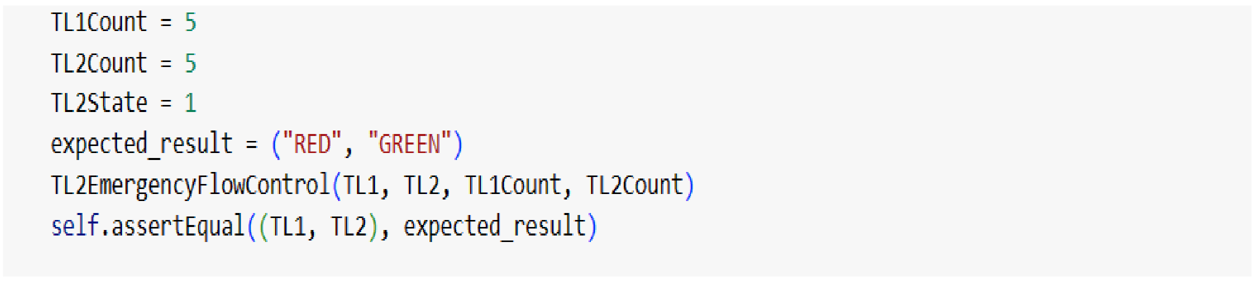
## Test case 3:



```
# Test case 3: TL2 has zero vehicle count
TL1Count = 8
TL2Count = 0
expected_result = ("GREEN", "RED")
TL1FlowControl(TL1, TL2, TL1Count, TL2Count)
self.assertEqual((TL1, TL2), expected_result)
```

test_TL1FlowControl_ZeroVehicleCountAtTL2: This test case covers the scenario when TL2 has zero vehicle count. TL1 and TL2 are again initially "RED" with assigned vehicle counts. However, the TL2Count is set to 0. The expected result is that TL1 should transition to "GREEN" and TL2 should remain "RED" since TL2 has no vehicles.

## Test case 4:



```
TL1Count = 5
TL2Count = 5
TL2State = 1
expected_result = ("RED", "GREEN")
TL2EmergencyFlowControl(TL1, TL2, TL1Count, TL2Count)
self.assertEqual((TL1, TL2), expected_result)
```

test_TL2EmergencyFlowControl_EmergencyFlow: This test case verifies the emergency flow control behavior of TL2. TL1 and TL2 are initially "RED" with assigned vehicle counts, and TL2State is set to 1 to simulate an emergency. The expected result is that TL1 should remain "RED" and TL2 should transition to "GREEN" during the emergency flow control operation.

## Simulation:

The simulation has been done via Simulation of Urban MObility (SUMO) software. SUMO is an open-source, and portable traffic simulator which can be used to model small and large traffic networks. It allows external interaction and control of a running simulation via TraCI (Traffic Control Interface) which is an API providing access to SUMO traffic simulations.

In the simulation, a smart traffic control designed for the quick navigation of emergency-forces vehicles is presented, where green light is given to the road with a higher traffic density, while keeping the arrival of emergency-forces vehicles in check, so that on arrival, the signal is either extended or closed, based on the direction the emergency-force vehicle is arriving from.

In comparison to a fixed-time traffic-controlled simulation, it was observed that the average waiting time of passenger vehicles is higher than that in the case of our traffic control system. However, in terms of emergency-forces' average waiting time, it was clear that our control system provided a lower value than the fixed-time control.

For balanced-traffic flows, fixed-time control would perform well. However, in a case where one of the intersection's roads have a little to no traffic flow, it may end up giving unnecessarily longer green time than needed to one of the two roads, while keeping the other congested, increasing the average waiting time in return.

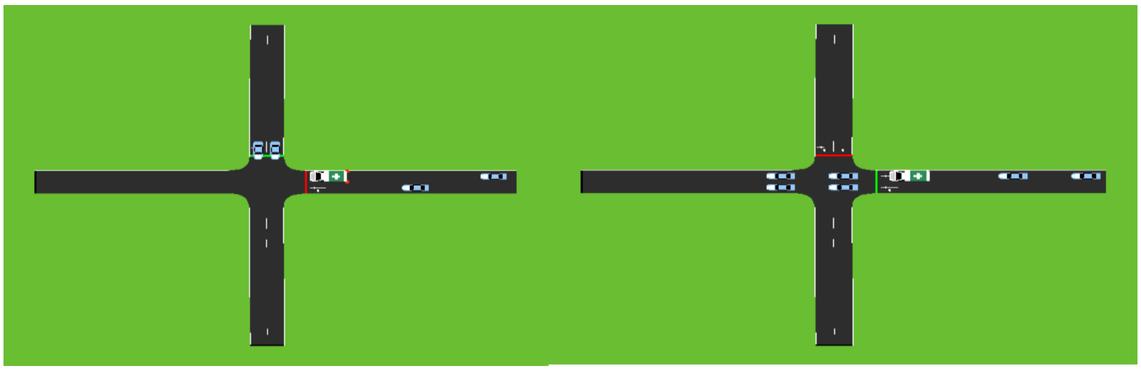


FIGURE 47(A SIMULATED COMPARISON BETWEEN THE PROPOSED CONTROL SYSTEM AND TRADITIONAL FIXED-TIME CONTROL)

### Model:

We make sure that the model doesn’t have overfitting, as it is known by dividing the data one part for training and the other part for testing. And through the results, it became clear to us that everything is correct.

## 6.2 Integration Testing

**Integration testing** is a software testing technique that focuses on verifying the correct interaction and communication between different components or modules of a software system. It involves testing the integration points where these components come together to ensure that they work together as expected.

The **primary goal** of integration testing is to identify defects or issues that may arise due to the interaction between different units or subsystems of the software. It helps to uncover problems such as incompatible interfaces, data corruption, communication failures, or incorrect data flow between components.

```python
class TrafficControlIntegrationTests(unittest.TestCase):

    def test_flow_control_with_emergency_at_TL1(self):
        TL1 = TL2 = "RED"
        TL1Count = 10
        TL2Count = 5
        TL1State = 0
        TL2State = 1

        # Expected outcome
        expected_result = ("RED", "GREEN")

        # Perform flow control
        TL1FlowControl(TL1, TL2, TL1Count, TL2Count, TL1State, TL2State)

        # Check the outcome
        self.assertEqual((TL1, TL2), expected_result)
```

test_flow_control_with_emergency_at_TL1: This test case verifies the flow control behavior when an emergency is present at TL1

Initial conditions:

TL1 and TL2 are initially set to "RED”

TL1Count is set to 10 (indicating the presence of vehicles at TL1)

TL2Count is set to 5 (indicating the presence of vehicles at TL2)

TL1State is set to 0 (indicating no emergency at TL1)

TL2State is set to 1 (indicating an emergency at TL2)

**Expected outcome:**

After performing flow control, TL1 should remain "RED" and TL2 should be set to "GREEN” This is because TL2 is in an emergency state, which overrides the flow control at TL1.

# Future Work

**In future work, we are wishing to add some extra work and features:**

- **Collecting** more type of vehicles photos to perform more training for the model.
- **Accuracy** increases the accuracy as much as it's possible, more accuracy means better and accurate results.
- **Adding** an audio sensor, so that in case of an ambulance arriving from both directions, the system would be able to identify the higher-priority ambulance based on its siren.
- **Working** on 4 lanes, not only 2 lanes.

# References

1. Mwangi, C.M., Kang'ethe, S., & Nyakoe, G.N. (2012). Design and Simulation of a Fuzzy Logic Traffic Signal Controller for a Signalized Intersection.
2. Ying, L., Lei, L., & Wei-peng, C. (2017). Intelligent traffic light control using distributed multi-agent Q learning. ITSC 2017.
3. O. Adebiyi, R. F., Ahmad Abubilal, K., Sunkary Tekanyi, A. M., & Hadir Adebiyi, B. (2017). Management of Vehicular Traffic System using artificial bee colony algorithm. International Journal of Image, Graphics and Signal Processing, 9(11), 18-28.
4. Zaatouri, K., & Ezzedine, T. (2018). A self-adaptive traffic light control system based on Yolo. 2018 International Conference on Internet of Things, Embedded Systems and Communications (IINTEC).
5. Kővári, B., Szőke, L., Bécsi, T., Aradi, S., & Gáspár, P. (2021). Traffic Signal Control via reinforcement learning for reducing global vehicle emission. Sustainability, 13(20), 11254.