

Helmet

Detection

#### Semester Project Documentation

###### Submitted by:

Usama Basharat (BITF20M528)

Subktgeen Babar (BITF20M541)

**Submitted To:**

**Dr. Gulzar Ahmad**

# 

# Modeling & Simulation

**Acknowledgement:**

We would like to express our special thanks of gratitude to our teacher Dr. Gulzar Ahmad who gave us the golden opportunity to do this wonderful project on the topic helmet detection, which also helped us in doing a lot of Research and we came to know about so many new things We are really thankful to them.  
Secondly we would also like to thank our friends who helped us a lot in finalizing this project within the limited time frame.

We are over helmed in all humbleness and gratefulness to acknowledge our depth to all those who have helped us to put these ideas, well above the level of simplicity and into something concrete.  
Thanking you.

**Abstract:**

Helmet detection is one of the most important system which is needed in traffic system. This project presents a real time detection whether a bike rider wears a helmet or not based on the Convolutional Neural Networks (CNN). This project uses the Yolov4 object detection algorithm which is implemented by deep neural network architecture. Our main goal of this project is to detect a bike rider and tell whether he/she wears helmet or not. This AI based project will automatically detect whether a person wears a helmet or not.

In this project machine learning and deep learning is also used, which automatically detect the object and shows the results with maximum accuracy.

**Table of Content**

1. Introduction
2. Overview of the Project
3. Problem Statement
4. Objective
5. System Model
6. Technologies Used
7. PYHTHON
8. CNN
9. ARTIFICIAL INTELLIGENCE
10. MACHINE LEARNING
11. DEEP LEARNING
12. Simulation and Result
13. Conclusion

## **INTRODUCTION**

### 

### **Overview of Project:**

Rapid Increase in traffic on the roads is one of the biggest problems now a days. Especially the bike riders do not follow the rules and most of the road accidents happened due to their carelessness. So, this application will easily detect the persons who wears helmet and those who don’t. And if one does not obey the rule then he/she has to pay the traffic challan fee.

## **Problem Statement:**

As the population grows there is an increase in the private vehicles and traffic on the roads which caused many issues to the people. This results in road accidents. We need a system which detects who is following the traffic rules and those who don’t follow.

## **Objective:**

To design a model which can efficiently detect the person who is wearing a helmet and who do not wear.





## **TECHNOLOGIES USED**

## **PYTHON:**

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. An interpreted language, Python has a design philosophy that emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly brackets or keywords), and a syntax that allows programmers to express concepts in fewer lines of code than might be used in languages such as C++or Java.

## **CNN:**

Convolutional Neural Networks (CNNs) are same as the human neural networks which are built with weights and neurons. So, by using Convolutional Neural Networks the task can be easily learned by the network. So, this is used with the preexisting algorithm to recognize in real-time the occupancy of the parking space. This includes the quality, accurate, efficient and the scalable answer for the real-time recognition of the parking lot. Convolutional Neural Networks are different kinds of multi-layered network, which is designed in such a way that with only some preprocessing it can detect the visual patterns from the image itself. The learning techniques, particularly the convolutional neural networks provide the solution to the problems such as parking occupancy detection.

### **Artificial Intelligence**

Artificial Intelligence is an approach to make a computer, a robot, or a product to think how

smart human think. AI is a study of how human brain think, learn, decide and work, when it tries to solve problems. And finally, this study outputs intelligent software systems. The aim of AI is to improve computer functions which are related to human knowledge, for example, reasoning, learning, and problem-solving.

### **Machine Learning**

Machine learning is a subfield of artificial intelligence (AI). The goal of machine learning generally is to understand the structure of data and fit that data into models that can be understood and utilized by people.

Although machine learning is a field within computer science, it differs from traditional computational approaches. In traditional computing, algorithms are sets of explicitly programmed instructions used by computers to calculate or problem solve.

### **Deep Learning**

Deep learning is a branch of machine learning which is completely based on artificial neural network. In deep learning, we don't need to explicitly program everything. Concept of deep learning has been around for a couple of years now but it's on hype nowadays because of huge amounts of data.

**System Model**

System modeling is the process of developing abstract models of a system, with each model representing a different view or perspective of the system. System modeling now means representing a system using some form of graphical notation, now almost always based on notation in the Unified Modeling Language (UML). System modeling helps the analyst understand the functionality of the system and the models are used to communicate with users.

* **Yolov4**

We use yolov4 in our project of helmet detection.

YOLO is an acronym of the term **'You Only Look One'**. It is an algorithm that detects and recognizes different objects in an image (in real time). Object detection in YOLO is performed as a regression problem and provides the class probabilities of the detected images.

In Yolov4, we performed 4 steps:

* Collection of datasets
* Labeling of dataset
* Training of dataset
* Testing of dataset

**Steps of Yolov4**

* First of all, after collecting and labeling dataset, we have to create a folder on drive named Yolov4. Next, we have to create another folder named training inside Yolov4Mask folder. This is where we will save our trained weights.
* Then we mount drive, link our dataset folder to drive and then go visit the previously created folder named Yolov4mask.
* Clone the darknet git repository of the custom dataset.
* Now, we have to create different folders to upload our dataset to drive.
* Create zip file obj.zip from obj folder containing input image ".jpg" files and their corresponding YOLO format labeled ".txt". Upload the zip file to the Yolov4Mask folder on your drive.
* Download the yolov4-custom.cfg file from the darknet /cfg directory, make changes to it, and upload it to the Yolov4Mask folder on your drive.
* Then create obj. data and obj. names files and upload both to drive. In obj. data we have to define that how many classes we have in our project and in obj. name we have to define the names of our project’s classes.
* Upload the process.py script file to the Yolov4Mask folder on your drive. This process.py script creates train.txt and test.txt files where train.txt file has paths to 90% of the images and test.txt has 10% image paths.
* Then, make changes to the make file to enable OPENCV and GPU.
* Run make command to create darknet.
* Copy all files from the Yolov4mask folder to the darknet directory.
* Run the process.py python script to create train.txt and test.txt files inside the data folder.
* Download pre-trained yolov4 weights.
* Now we arrive at the step where we have to train our dataset.
* First of all, we have to train our custom detector. For best results, you should stop training when the average loss is less than 0.05 or at least less than 0.3 if possible, otherwise train the model until the average loss don't show changes significantly over time.
* If you disconnect or lose your session, you don't have to start training your model all over again. You can resume training from where you left off. Use the weights that were last saved. The weights are saved every 100 iterations as yolov4 custom-last. weights in the yolov4/training folder on your drive.
* Use this simple hack for auto click to avoid exiting Colab VM. function ClickConnect(){  
  console.log("Working");   
  document  
    .querySelector('#top-toolbar > colab-connect-button')  
    .shadowRoot.querySelector('#connect')  
    .click()
* To check performance, we have to repeat the first step of creating a training folder in the Yolov4mask folder.
* At last, we have to test our custom detector to verify that our project is working properly or not.

# 

**Simulation and Results**

The YOLO object detection model is used to detect helmets in captured images. YOLO is a DCNN (Deep Convolutional Neural Network) object recognition model with good performance and results in terms of accuracy as well as estimation speed.

Training data is obtained from our surroundings.

# 

# 

# 

# 

# 

# 

# CONCLUSION

The YOLO project has shown how a single-shot detector can be used as a baseline to evolve a model intended to detect bike riders’ helmet. It highlights the effectiveness of containerization of responsibilities during training, where image processing and compilation occurs concisely and efficiently in separate services.

### **Future scope:**

In future the model can be trained by increasing the images in the dataset and making huge dataset in such a way that it can detected at different locations and with different environment.