School of
Electronics and Communication Engineering

Mini Project Report

on

Traffic Light Detection in Diverse Weather Conditions Using Machine Learning

By:

1. **Naveenkumar Gumaste** USN:01FE22BEC407
2. **Chandan C Raikar** USN:01FE21BEC038
3. **Divya R Salmani** USN:01FE21BEC013
4. **Sanjana Adagimath** USN:01FE21BEC363

Semester: V, 2023-2024

Under the Guidance of

Rajeshwari K

K.L.E SOCIETY'S
KLE Technological University,
HUBBALLI-580031
2022-2023

SCHOOL OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that project entitled **"Traffic Light Detection in Diverse Weather Conditions Using Machine Learning"** is a bonafide work carried out by the student team of "**Naveenkumar Gumaste(01FE22BEC407)**, **Chandan C Raikar (01FE21BEC038)**, **Divya R Salmani (01FE21BEC013)**, **Sanjana Adagimath (01FE21BEC363)**" . The project report has been approved as it satisfies the requirements with respect to the mini project work prescribed by the university curriculum for BE (V Semester) in School of Electronics and Communication Engineering of KLE Technological University for the academic year 2023-2024.

Rajeshwari K	**Suneeta V Budihal**	**B. S. Anami**
Guide	**Head of School**	**Registrar**

External Viva:

Name of Examiners **Signature with date**

1.

2.

ACKNOWLEDGMENT

We extend our gratitude to Dr. Ashok Shettar, Vice-Chancellor of KLE Technological University, Hubballi, and Dr. P. G. Tewari, Dean Academics of KLE Technological University, for providing us with the opportunity to conduct our research and for their unwavering support throughout the CIM Mini Project. Special thanks to Dr. Suneeta V.B., our head of school, for her leadership and support. We are also thankful to Prof. Rajeshwari K. for her invaluable help and guidance during our work. Additionally, we acknowledge the contributions of all teaching and non-teaching staff for their assistance and encouragement.

-The project team

ABSTRACT

This project presents a current exploration into the critical realm of traffic light detection within intelligent transportation systems (ITS), which is crucial for enhancing road safety and traffic management. Motivated by the continuous evolution of methodologies, our research provides a comprehensive analysis of detection techniques. From traditional computer vision to deep learning integration, we scrutinize strengths, limitations, and potential future directions. The study is conducted using datasets from ROBOFLOW UNIVERSE, demonstrating novelty in the approach. Ongoing advancements in detection strategies are crucial for improving road safety and traffic efficiency, establishing this research's significance within the evolving landscape of intelligent transportation systems.

Contents

Chapter 1

Introduction

In the Smart Transportation Systems (STS) domain, the efficiency of traffic light detection is integral to advancing road safety and optimizing traffic flow. The burgeoning growth of urban environments amplifies the demand for sophisticated systems capable of managing intricate traffic networks. This project responds to this imperative by concentrating on elevating the precision and efficiency of traffic light detection, specifically addressing challenges posed by urban mobility. The methodology employed involves a thorough examination of detection approaches, spanning from traditional computer vision to cutting-edge deep learning. This exhaustive analysis explores the strengths, limitations, and applications of Intelligent Transportation Systems (ITS). By leveraging datasets from ROBOFLOW UNIVERSE, the project ensures its relevance in real-world scenarios. Integrating traditional computer vision with deep learning not only contributes valuable insights but also positions the project to make meaningful contributions within the dynamic field of ITS.

1.1 Motivation

Improving the precision and efficiency of traffic light detection within Smart Transportation Systems (STS) is pivotal for enhancing road safety in increasingly complex urban environments. As urban areas expand, the demand for advanced systems to manage intricate traffic networks grows, compelling this project to focus on optimizing traffic flow through enhanced traffic light detection. The motivation stems from addressing urban mobility challenges, prompting a thorough review of detection approaches, and leveraging datasets from ROBOFLOW UNIVERSE to develop solutions relevant to real-world scenarios. Integrating traditional computer vision with cutting-edge deep learning technologies adds a novel dimension, offering valuable insights and advancing traffic light detection methodologies within the dynamic realm of Intelligent Transportation Systems (ITS). In essence, this project is driven by the overarching goals of fostering road safety, optimizing traffic flow, addressing urban mobility challenges, and integrating innovative technologies to contribute meaningfully to the evolution of ITS.

1.2 Objectives

- **Precision in Detection:** Formulating algorithms that exhibit high precision in detecting the presence of traffic lights within captured images, aiming to enhance road safety and traffic flow optimization significantly.

- **Advanced Image Processing:** Developing sophisticated algorithms for extracting relevant features from input images. The system must demonstrate robustness in handling variations such as occlusions, distortions, and changes in perspective, ensuring accurate and reliable traffic light detection.

- **Intelligent Classification:** Implementation of classification algorithms to intelligently recognize the specific type and significance of each detected traffic light. This step is crucial for providing nuanced insights into the traffic signal status and facilitating effective decision-making in Intelligent Transportation Systems (ITS).
- **Robust Adaptability:** Ensuring the system's robustness and adaptability to diverse environmental conditions, encompassing different lighting scenarios, weather variations, and changes in the appearance of traffic lights. The objective is to create a system that performs reliably in real-world settings.

.

1.3 Literature survey

Mark P Philipsen, proposed the learning-based detector outperformed heuristic model-based detectors, exhibiting superior precision and recall—a critical parameter due to the irrevocable loss of false negatives. Emphasizing the significance of the learning-based detector's heightened recall, the study primarily assessed its success in detecting traffic lights. Proposed enhancements involved incorporating tracking methods to refine its output. The overall system performance was evaluated through precision-recall curves and the Area Under the Curve (AUC), providing a comprehensive analysis of the effectiveness of the implemented learning-based detector and associated methodologies.

Trung-Hieu Nguyen's, proposed real-time traffic sign recognition model integrated into a 1:7 RC vehicle, demonstrated remarkable effectiveness and robustness in challenging scenarios. The system exhibited an impressive average accuracy of 99.78 percent in detecting traffic signs on the embedded platform. Despite its success, limitations were identified, including the impracticality of many existing systems in real-time environments, the computational weight of deep learning methods on embedded systems, consideration of only five common traffic signs, and a response time of 22 to 23 frames per second. The study advocates for further system expansion and methodology refinement, emphasizing a lightweight model, optimal recognition methods, and image processing techniques for training data.

Noor Hussain Sarhan's study, the primary focus was on evaluating and comparing two traffic light detection models, emphasizing their accuracy in classifying images and video frames. The findings centered on the precision of traffic light detection in both static images and video frames. Identified limitations included the necessity for further research to amalgamate the strengths of the two models and the call for future investigations employing deep learning algorithms and an expanded dataset. The methodology consisted of developing two distinct models utilizing the OpenCV library for image and video processing, with the YOLO detection system fine-tuning the video model's weights and employing predefined color ranges for traffic light detection in images.

Amara Dinesh's, research highlights the success of capsule networks, achieving a state-of-the-art accuracy of 97.6 percent on the German traffic sign dataset. Capsule networks are superior to CNNs in the challenging task of traffic sign detection, excelling in recognizing pose and spatial variances. This enhances reliability and accuracy in image classification, even for blurred, rotated, and distorted images. The study evaluates the algorithm's performance in different orientations, emphasizing its proficiency in correctly identifying traffic signs. Limitations include inherent CNN limitations, the need for manual feature engineering, an unbalanced test set, and relatively lengthy training times. Methodologically, capsuleFig. 1. Functional Block Diagram networks are employed with specific layers, a route-by-agreement algorithm, and a decoder network.

1.4 Problem statement

Traffic Light Detection in Diverse Weather Conditions Using Machine Learning.

1.5 Organization of the report

Initiating with an introduction in Chapter 1, the project systematically explores various facets of the project. Chapter 2 extensively examines the complexities of system design, providing an in-depth overview of its structure and components. Progressing further, Chapter 3 thoroughly covers the planning and execution aspects of the design, illuminating strategic decisions made during development. Chapter 4 shifts focus to presenting and analyzing results, with detailed discussions to clarify findings. Finally, Chapter 5 concludes the report with a summary, offering insights into project outcomes, and implications, and paving the way for future exploration and development in the identified scope. This structured approach ensures a cohesive and comprehensive narrative, guiding readers through the project's initiation, design, execution, results, and future prospects.

Chapter 2

System design

In this Chapter, We will analyze YOLOv8's cutting-edge advancements and intricate design principles as we look into the system's inner workings. The block diagram for our project will show how YOLOv8's cutting-edge features were incorporated and how well our solution integrated them.

2.1 Functional Block Diagram

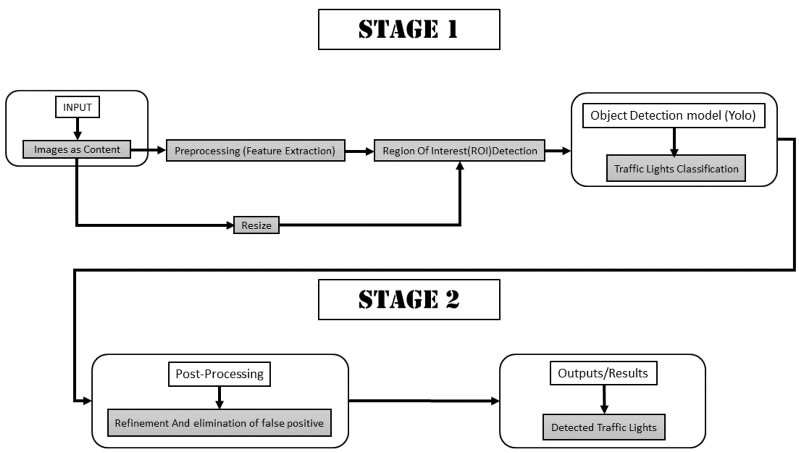

Figure 2.1: Shows Functional Block Diagram

2.1.1 Functional Block Diagram Explanation

- **Input:** Video frames or still images, maybe in different resolutions and formats (such as RGB or grayscale), are received.
- **Preprocessing:** Applies data normalization, and filtering (e.g., Gaussian blur) to mitigate noise and enhance relevant features. May involve color space conversion (e.g., RGB to HSV) for robust traffic light representation.

- **Region of Interest (ROI) Detection:** Employs object detection algorithms (e.g., YOLOv8) to localize traffic light bounding boxes within the frame. Can incorporate anchor boxes of varying sizes and aspect ratios to handle diverse traffic light positions and appearances.

- **Traffic Light Classification:** Leverages deep convolutional neural networks (CNNs) trained on labeled datasets of traffic light images. CNN architecture likely features convolutional layers for feature extraction, followed by pooling layers for spatial down-sampling and fully-connected layers for classification into red, yellow, or green states.

2.2 Yolov8 Architectural Diagram

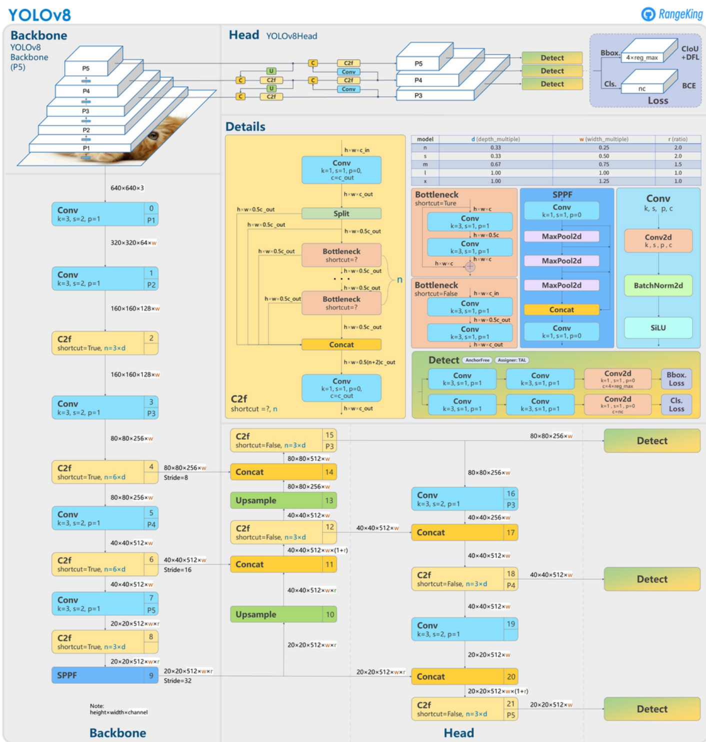

Figure 2.2: Shows YoloV8 Architectural Diagram

2.2.1 Yolov8 Architectural Diagram Explanation

- **Backbone:** The model's component that pulls features from the input image is this one. It is composed of several convolutional layers that increase the number of channels and gradually shrink the image's size. The backbone's architecture might differ based on the model version, however it usually uses a ResNet or CSPDarknet design.

- **Neck:** This part of the model processes the features extracted by the backbone to prepare them for the detection head. It typically consists of a spatial pyramid pooling (SPP) layer, which aggregates features from different spatial scales, and a few convolutional layers.

- **Head:** This part of the model is responsible for making the final predictions. It consists of a series of convolutional layers that predict the bounding boxes, class probabilities, and confidence scores for the objects in the image.
- **C2f module:** This is a recent development of the YOLOv8 architecture that raises the model's accuracy. It mixes characteristics from the neck and backbone to produce a more illuminating depiction of the picture.
- **Decoupled head:** This is another new addition to the YOLOv8 architecture that helps to improve the speed of the model. It eliminates the need for a separate objectness branch, which can be computationally expensive.

Chapter 3

Implementation details

In this chapter, We will dive into the technical intricacies of establishing a robust environment tailored for training a YOLOv8 model. This involves configuring essential dependencies, preparing the dataset meticulously, and executing the training and testing process for the model.

3.1 Environment Setup

1. **Operating system library for interacting with the OS:**

```
import os
```

2. **using it for file path expansion:**

```
import glob
```

3. **Displaying images in IPython environment:**

```
from IPython.display import Image, display
```

4. **It displays detailed information about each GPU on the system, such as its name, model, temperature, utilization, memory usage, and more:**

```
!nvidia-smi
```

5. **Installing and updating the Ultralytics library:**

```
pip install -U ultralytics
```

6. **Importing Ultralytics into the project:**

```
import ultralytics
```

7. **Checking if the Ultralytics is correctly setup:**

```
ultralytics.checks()
```

8. **Setting Constants, Creating Directories and Changing Directory :**

```
HOME = "/content/"
print(HOME)
!mkdir {HOME}/datasets
%cd {HOME}/datasets
```

3.2 Setting up YoloV8, Dataset, Training YoloV8s model and getting the results

1. **Installing the Roboflow library with the dataset using its unique api-key:**

```
!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="sRZFgPn7f9rMLPTIU8Rq")
project = rf.workspace("wawan-pradana").project("cinta_v2")
dataset = project.version(1).download("yolov8")
```

3.2.1 Setting up YoloV8, Training YoloV8s model

2. **Training the YOLOv8 model with specified parameters of 125 epochs with an image size of 1000 pixels and a batch size of 32:**

```
%cd {/content/drive/MyDrive}
!yolo task=detect mode=train model=yolov8s.pt data='/content/datasets/cinTA_v2/data.yaml' epochs=125 imgsz=1000
```

3. **Listing Detected Objects/files:**

```
!ls '/content/runs/detect/train'
```

3.2.2 Visualizing the Results In Environment

4. **Displaying confusion matrix using "Image" command imported from IPython.display:**

```
Image(filename = f"/content/runs/detect/train/confusion_matrix.png")
```

5. **Validating the model in validation mode using a trained model:**

```
!yolo task=detect mode=val model='/content/runs/detect/train/weights/best.pt' data='/content/datasets/cinTA_v2/data.yaml'
```

6. **Predicting on an "Image" in prediction mode on a single image file:**

```
%cd {HOME}
!yolo task=detect mode=predict model='/content/runs/detect/train/weights/best.pt' conf=0.25 source='IMG_0520.jpeg'
```

7. **Predicting on a "Video" in prediction mode on single video file:**

```
%cd {HOME}
!yolo task=detect mode=predict model='/content/runs/detect/train/weights/best.pt' conf=0.25 source='IMG_0524.mov'
```

Chapter 4

Results

In this chapter, we will delve into a comprehensive exploration of the results and project outcomes achieved through the implementation of YOLOv8. This entails an in-depth analysis of the results obtained.

4.1 Result Visualizing

Figure 4.1: Shows Results of YoloV8 implementation for our real-world sample images

fig 4.2 shows a few sample results of our YoloV8 implementation on our own images other than the dataset images. The results are quite good, and they can be used in real-world applications.

4.1.1 Result Analysis

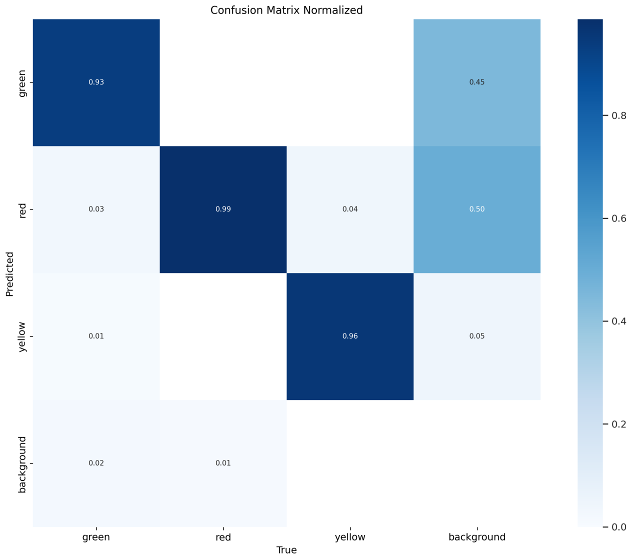

Figure 4.2: Shows Confusion Matrix Of YoloV8s Trained Model

fig 4.3 shows the confusion matrix of our YoloV8 trained model which is trained on our own custom dataset. The confusion matrix shows the accuracy of our model, which averages at 96 percent most of the time.

Epochs	mAP (50)	mAP(50-90)
50	0.771	0.345
100	0.943	0.572
125	0.981	0. 634

Figure 4.3: Shows Result Comparison Of Epochs For Which The Model as trained

As shown in fig 4.3 the study showcased significant improvements in Mean Average Precision (MAP) scores across training epochs. Starting at a MAP score of 0.345 after 50 epochs, the detection accuracy steadily increased, reaching 0.962 at 125 epochs. This trajectory underscores the efficacy of our methodology in advancing traffic light detection accuracy, emphasizing the ongoing importance of evolving detection strategies for optimizing safety and traffic efficiency in the dynamic landscape of ITS(Intelligent Transportation Systems).

Chapter 5

Conclusions and future scope

5.1 Conclusion

In conclusion, our comprehensive exploration and the achieved results underscore the potential of combining traditional and deep learning techniques for robust traffic light detection. This work contributes to the broader field of ITS by providing insights into enhancing the accuracy of detection systems, ultimately contributing to safer and more efficient transportation systems.

5.2 Future scope

1. **Enhanced Dataset Diversity:** Recommending the utilization of a more diverse dataset with increased variability in environmental conditions, lighting, and traffic scenarios to improve the robustness and generalization of the traffic light detection model.

2. **Detailed Annotation and Bounding Boxes:** Emphasizes the importance of meticulous annotation and precise bounding boxes in the dataset to provide a richer ground truth for training, ensuring the model's accuracy in identifying and localizing traffic lights under various circumstances.

3. **High-Quality Hardware Implementation:** Suggesting the adoption of advanced hardware configurations, possibly leveraging GPUs or TPUs, to expedite model training and inference processes. This can lead to enhanced efficiency and real-time performance, especially in scenarios with increased computational demands.

4. **Integration of Advanced Learning Techniques:** Exploring the incorporation of state-of-the-art deep learning techniques, such as transfer learning or ensemble methods, to further enhance the model's ability to generalize across diverse traffic light scenarios and potentially improve its performance on challenging instances.

5. **Continuous Model Refinement:** Encouraging an iterative approach to model refinement through continuous experimentation with hyperparameter tuning, architecture modifications, and advanced training strategies. This approach ensures the adaptability of the traffic light detection model to evolving real-world conditions and challenges.

References

Mark Philip Philipsen, "Traffic Light Detection: A Learning Algorithm and Evaluations on Challenging Dataset", 2015 IEEE 18th International Conference on Intelligent Transportation Systems. [1]

Trung-Hieu Nguyen, "A Lightweight Model For Real-time Traffic Sign Recognition", 2020 5th International Conference on Green Technology and Sustainable Development (GTSD). [2]

Noor Hussain Sarhan, "Traffic light Detection using OpenCV and YOLO", 2022 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT). [3]

Amara Dinesh Kumar, "Novel Deep Learning Model for Traffic Sign Detection Using Capsule Networks," CoRR, vol. abs/1805.04424, 2018. [4]

Fatma Nur Ortataş, "Performance Evaluation of YOLOv5, YOLOv7, and YOLOv8 Models in Traffic Sign Detection", 2023 8th International Conference on Computer Science and Engineering (UBMK). [5]

Abdellah Nabou, "Road Signs Recognition by Using YOLOv8 Model", 2023 14th International Conference on Intelligent Systems: Theories and Applications (SITA). [6]

Ultralytics, "YOLOv8 Docs", https://docs.ultralytics.com/models/yolov8/. [7]