HelmetDetect

Higher Diploma in Software Engineering

2023.1F

School of Computing and Engineering

National Institute of Business Management

GALLE

NATIONAL INSTITUTE OF BUSINESS MANAGEMENT

HIGHER DIPLOMA IN SOFTWARE ENGINEERING

Digital Image Processing

GA/HDSE23.1F

Real-Time Helmet Detection Using Advanced Image Processing Algorithms

SUBMITTED BY

GROUP 06

P.P. MADUBASHINI	**GAHDSE231F-007**
D.G.S.S. SATHSARA	**GAHDSE231F-008**
W.K.Y. NAWODYA	**GAHDSE231F-027**
F.M. MOHIDEEN	**GAHDSE231F-028**
S.N.R. SIRISINGHE	**GAHDSE231F-029**

SUPERVISED BY

MR. THILINA MADUSHAN

Declaration

We, the undersigned members of the group, collectively declare that this project, conducted as part of our Higher National Diploma, is an original work and does not incorporate, without acknowledgment, any material previously submitted for a Diploma in any institution. To the best of our collective knowledge and belief, it does not contain any material previously published or written by another person or ourselves, except where due reference is made in the text. We also hereby give our joint consent for our project report, if accepted, to be made available for photocopying and for interlibrary loans. Additionally, we grant permission for the title and summary of our project to be made available to outside organizations. This declaration affirms our commitment to academic integrity and acknowledges the importance of ethical research practices in our collaborative pursuit of knowledge.

D.G.S.S. SATHSARA

P.P. MADUBASHINI

S.N.R. SIRISINGHE

F.M. MOHIDEEN

W.K.Y. NAWODYA

Date: March 19,2024

Abstract

The HelmetDetect initiative introduces a strategy to enhance road safety by utilizing computer vision technology. Its goal is to create a system that can identify individuals without helmets in time from images or video feeds utilizing the YOLOv5 object detection model. The report delves into the problem-solving approach, methodology challenges faced, lessons learned and recommendations, for endeavors.

Key components of the project involve gathering and annotating datasets, training models with YOLOv5 developing a user interface using Flask and assessing performance metrics. The system demonstrates accuracy in detecting instances of helmet noncompliance showing promising outcomes in raising awareness about road safety and enforcing regulations.

Obstacles encountered during the project, such as data collection issues, model training challenges and technical obstacles were tackled through solutions and collaborative teamwork. Valuable insights gained include the significance of data quality, interdisciplinary collaboration, iterative development processes and effective project management techniques.

The project contributes to enhancing expertise in learning techniques data preprocessing methods, web development skills and deployment optimization practices. Additionally, it promotes awareness about road safety. Encourages behavior, among road users.

Suggestions, for tasks involve enhancement of the model extending into multiple sensing modes integrating user input and delving into wider safety applications.

To sum up the HelmetDetect initiative showcases how technology based solutions can tackle issues and enhance well-being. Through a dedication to learning, creativity and cooperation we are steadfast in our efforts to make a difference, in road safety and transportation systems.

Table of Contents

List of Keywords

1. HelmetDetect
2. Computer Vision
3. Road Safety
4. Object Detection
5. YOLOv5
6. Deep Learning
7. Dataset Annotation
8. Model Training
9. Flask
10. User Interface
11. Precision
12. Recall
13. Mean Average Precision
14. Data Preprocessing
15. Transfer Learning
16. Technical Skills
17. Project Management
18. Remote Desktop
19. Deployment Optimization
20. Public Awareness Campaigns

Acknowledgement

We would like to express our appreciation to Mr. Thilina Madushan, our respected lecturer for his unwavering guidance, support, and encouragement, throughout the duration of the "Digital Image Processing" module. His expertise, passion for the matter and dedication to our learning journey have significantly influenced our understanding of machine learning concepts and their practical applications.

We also want to extend our gratitude to our fellow group members—Savindu, Madubashini, Sasini, Mumthaz, and Yuwani. The collaboration among us has been enriched by each members skills, perspectives and commitment. Our shared dedication to excellence and effective teamwork has played a role in completing this project.

Furthermore we would like to thank the National Institute of Business Management (NIBM) for providing us with the resources and support, for our pursuits. The availability of the Learning Management System (LMS) and other educational tools greatly facilitated our data collection and analysis processes.

This project has been a learning experience that allowed us to apply knowledge in real world scenarios. The insights. Skills honed during this project will undoubtedly contribute to both our advancement and professional growth.

Our heartfelt thanks go out to everyone who has been a part of this journey.

Sincerely,

D.G.S.S. Sathsara

P.P. Madubashini

S.N.R. Sirisinghe

F.M. Mohideen

W.K.Y. Nawodya

1 Introduction

The integration of computer vision systems holds significant promise in improving efficiency and safety measures across various domains in the rapidly evolving technological landscape of today. Among these, road safety continues to be a top priority on a global scale, with many programs designed to lower the number of accidents and fatalities. The wearing of helmets appears to be essential for reducing the risk of head injuries and saving lives on the road, especially for motorcycle riders.

1.1 Background and Motivation

It is well known that wearing a helmet helps reduce the risk of brain injuries from auto accidents. The World Health Organization (WHO) has released statistics showing that wearing a helmet can lower the risk of fatalities by approximately 40% and severe head injuries by up to 70%. Notwithstanding the evident advantages, failure to adhere to helmet laws continues to be a problem in numerous areas. This is a result of a number of factors, such as insufficient awareness campaigns, lax enforcement, and cultural attitudes regarding safety gear.

Inspired by the urgent need to tackle this problem, we conceptualized our project "HelmetDetect". Through the application of deep learning and digital image processing, we sought to create a reliable system that could identify people not wearing helmets in real-time situations. A system like this could help traffic management organizations, law enforcement agencies, and safety advocacy groups properly enforce helmet laws and promote road safety.

1.2 Objectives of the Project

The main goal of the HelmetDetect project is to create an intelligent system that can recognize people in photos or video streams who are not wearing helmets. Principal goals consist of:

1. Gathering and annotating a dataset of images of people wearing and not wearing helmets.
2. Using a cutting-edge object detection model to detect helmets.
3. The annotated dataset is used to train the model to achieve high accuracy and generalization.

4. Assessing the model's performance in terms of robustness, speed, and accuracy of detection.
5. Providing examples of the system's usefulness in actual situations.

1.3 Scope of the Project

The creation and application of a computer vision system for helmet detection is included in the HelmetDetect project's scope. In particular, the project consists of:

1. The gathering and annotation of a varied dataset comprising pictures of people in different settings, like roads, crosswalks, and traffic scenes.
2. Making the main model architecture, the YOLOv5 object detection algorithm.
3. Developing the model's ability to discriminate between people who are and are not wearing helmets.
4. A qualitative analysis and quantitative metrics are used to assess the model's performance.
5. Taking into account realistic deployment scenarios and integrating with current monitoring or surveillance systems.

1.4 Overview of the Report

An extensive summary of the HelmetDetect project is given in this report, along with information on the methodology, results, and insights discovered during the project's lifetime. It includes several phases, such as gathering datasets, creating models, training them, evaluating them, and talking about the difficulties encountered and the lessons discovered. The report also makes recommendations for further study and real-world uses of the created system.

2 Methodology

This section delves into the methods used for the HelmetDetect project, detailing the procedures followed from gathering datasets to assessing models.

2.1 Dataset Collection and Annotation

The gathering and annotation of an appropriate dataset was an essential first step in the development of the HelmetDetect system. The dataset is made up of pictures of people in different situations that were taken from different sources, including public datasets, traffic intersections, and surveillance cameras. A well-liked tool for bounding box annotation, LabelImg, was used to annotate these photos in order to distinguish between people wearing helmets and those who weren't.

2.2 Preprocessing Techniques

Several preprocessing techniques were used to improve the quality and suitability of the data before feeding the annotated dataset into the model. These methods included augmentation, normalization, and resizing of images. While normalization served to standardize pixel values and facilitate faster convergence during training, resizing guaranteed consistent dimensions across all images. To strengthen the model's resilience and diversify the training set, augmentation methods like brightness modifications, flips, and random rotations were also used.

2.3 Model Selection: YOLOv5

The YOLOv5 architecture was selected for object detection tasks like helmet detection because of its higher efficiency and performance. Because it strikes a balance between speed and accuracy, YOLOv5 is a good choice for real-time applications. Multiple convolutional layers make up the model architecture, which culminates in a detection head that uses the input image to predict bounding boxes and class probabilities.

2.4 Training Procedure

The preprocessed dataset was fed into the YOLOv5 model during the training process, and its parameters were adjusted to maximize performance. Stochastic gradient descent (SGD) was used to train the model, and a learning rate scheduler was used to progressively change the learning rate over epochs. To increase detection accuracy during training, the model's weights were changed to minimize the loss function.

2.5 Evaluation Metrics

Several metrics were used, including Precision (P), Recall (R), and mean Average Precision (mAP), to assess the performance of the trained model. Recall measures the percentage of true positive detections among all ground truth positive instances, whereas precision measures the percentage of true positive detections among all positive detections made by the model. An overall indicator of the model's accuracy across various confidence levels is provided by mAP.

2.6 Model Summary

Metrics like the number of layers, parameters, gradients, and GFLOPs (Giga Floating Point Operations per Second) were used to summarize the model's performance after training. In addition, class-wise metrics like Precision, Recall, and mAP were computed to evaluate how well the model identified people who were wearing helmets.

Promising results were found in the model summary, which showed good mean Average Precision across all classes along with high Precision and Recall scores.

We then go on to talk about the specifics of the implementation and the outcomes that the HelmetDetect system produced.

3 Implementation

This section offers an overview of the HelmetDetect system's implementation details, including the tools and software employed, the system architecture, and the incorporation of a Flask user interface (UI).

3.1 Software and Tools Used

Several software libraries and tools were used during the HelmetDetect system's development to support different project lifecycle stages:

1. **Python:** The object detection model, dataset preprocessing, and system integration were all implemented using Python as the main programming language.
2. **PyTorch:** The YOLOv5 model was trained on the annotated dataset using PyTorch, a well-known deep learning framework. PyTorch is a good choice for deep learning tasks because it offers automatic differentiation and efficient tensor computations.
3. **YOLOv5:** The fundamental model architecture for helmet detection was based on the YOLOv5 implementation made available by the Ultralytics team. YOLOv5 streamlines the development process by providing pre-trained models and training scripts.
4. **Flask:** A user-friendly interface for interacting with the HelmetDetect system was created using Flask, a lightweight web framework for Python. Through a web interface, Flask enabled us to serve the model predictions in real-time and allow users to upload images or video streams for helmet detection.
5. **HTML, CSS, and JavaScript:** The Flask application's user interface elements were designed and styled using these web technologies. The web pages were structured by HTML, styled for aesthetic appeal by CSS, and validated client-side and dynamically by JavaScript.
6. **LabelImg:** The dataset was manually annotated using LabelImg, an open-source annotation tool, by defining bounding boxes around people who wore helmets and those who did not.
7. **Git:** To manage the project codebase, keep track of changes, and communicate with team members, the Git version control system was used.

3.2 System Architecture

The HelmetDetect system has a modular architecture, with multiple parts that cooperate to accomplish the goal of helmet detection:

1. **Input Module:** This module manages the data that is uploaded by users via the Flask interface, such as pictures or video streams. The input data is preprocessed and made ready for inference.
2. **YOLOv5 Model:** This is the main part that is in charge of identifying helmets in the input pictures or video frames. The model, which was trained on the annotated dataset, produces class probabilities and bounding box predictions for every helmet that is detected.
3. **Output Module:** This module processes the data and creates visualizations that are superimposed on the input images or video frames after acquiring predictions from the YOLOv5 model. For user interpretation, it highlights the helmets that have been detected using bounding boxes and class labels.
4. **Flask Application:** The HelmetDetect system and the user are interfaced with by the Flask application. It offers a web-based interface through which users can upload pictures or videos, start the helmet detection procedure, and see the real-time results.
5. **User Interface:** The HTML/CSS/JavaScript web pages that make up the user interface offer a smooth and simple way to interact with the HelmetDetect system. Users have the ability to download processed photos or videos, view detection results, and upload files.

The HelmetDetect system provides a comprehensive solution for detecting helmets in real-world scenarios, supporting the enforcement of safety regulations and raising awareness of road safety by integrating these components.

3.3 Code Overview

The HelmetDetect codebase is divided into multiple modules, each of which is in charge of handling particular system functions. Here is a high-level summary of the key elements:

1. **Data Preprocessing:** Scripts for resizing, normalizing, and enhancing the dataset images are included in this module. Prior to training the model, it verifies the input data's quality and consistency.

2. **Model Training:** Scripts for setting up and using the YOLOv5 model on the annotated dataset are included in the model training module. It makes use of Ultralytics' YOLOv5 implementation and PyTorch.
3. **Inference:** The trained model's deployment for generating predictions on fresh data is managed by the inference module. It contains scripts to process input images or video streams, load the model, and produce detection results.
4. **Flask Application:** The web interface for interacting with the HelmetDetect system is part of the Flask application module. For handling user interactions and UI design, it consists of JavaScript scripts, CSS stylesheets, and HTML templates.
5. **Integration:** The scripts in this module allow the Flask application to be integrated with the inference module. It controls the flow of information between the frontend and backend parts, making it easier to upload files, make predictions, and see the results.

4 Results and Discussion

This section discusses the performance metrics, results analysis, and road safety implications in addition to presenting the HelmetDetect system's results.

4.1 Performance Metrics

The metrics of mean average precision (mAP), recall (R), and precision (P) were combined to assess the HelmetDetect system's performance. These measures shed light on the helmet detection model's overall efficacy, accuracy, and completeness.

- **Precision (P)**: Precision measures the proportion of true positive detections among all positive detections made by the model. A high precision score indicates that the model makes fewer false positive predictions, leading to more reliable detections.

- **Recall (R)**: Recall measures the proportion of true positive detections among all ground truth positive instances. A high recall score indicates that the model captures a large percentage of true positive instances, minimizing false negatives.

- **Mean Average Precision (mAP)**: mAP provides an aggregate measure of the model's accuracy across different confidence thresholds. It calculates the average precision for each class and then computes the mean over all classes. A higher mAP score indicates better overall detection performance.

4.2 Analysis of Results

The results obtained from the HelmetDetect system indicate promising performance in detecting individuals wearing helmets. The model achieved the following metrics:

- **Precision (P):** 97.3%

- **Recall (R):** 66.3%

- **Mean Average Precision (mAP50):** 66.4%

These metrics suggest that the model exhibits high precision in detecting helmets, with a relatively good recall rate. The mean average precision score further confirms the model's effectiveness in accurately localizing and classifying helmet-wearing individuals in images.

However, it's important to note that while the precision score is high, the recall score is comparatively lower. This implies that while the model excels at minimizing false positives, it may miss detecting some instances of individuals wearing helmets, leading to false negatives. Improving the recall rate could be a focus area for future optimizations.

Despite these considerations, the overall performance of the HelmetDetect system demonstrates its potential utility in real-world applications, such as road traffic management, law enforcement, and safety advocacy initiatives. By accurately identifying individuals without helmets, the system can help enforce safety regulations, raise awareness about the importance of helmet usage, and ultimately contribute to reducing road traffic injuries and fatalities.

4.3 Practical Implications

The HelmetDetect system's findings have a number of useful ramifications for traffic control, public awareness campaigns, and the enforcement of traffic laws. This section covers the system's performance's practical implications and how to use it in real-world situations.

1. **Road Safety Enforcement:** Law enforcement organizations can enforce helmet laws on the road with the help of HelmetDetect, a potent tool. Authorities can identify and penalize violators by accurately detecting individuals who are not wearing helmets, which serves as an incentive for compliance with safety laws. This preventive measure can help lower the number of head injuries-related fatalities and injuries from auto accidents.
2. **Traffic Management:** Real-time tracking of helmet usage on roads is made possible by integrating HelmetDetect into currently in place traffic management systems. The technology can be used by traffic authorities to pinpoint areas with low rates of helmet compliance and implement focused interventions, like awareness campaigns or more enforcement, to raise the level of safety there. This data-driven strategy encourages safer driving practices among drivers and improves the effectiveness of traffic management initiatives.
3. **Public Awareness Campaigns:** To emphasize the value of helmet use and inform the public about safe driving practices, public awareness campaigns can make use of the visual feedback that HelmetDetect provides. Through the use of real-world incidents where riders without helmets are detected, these campaigns aim to connect with audiences and motivate them to put safety first when riding bicycles or motorbikes. These kinds of programs are essential for encouraging a safety-conscious culture and lowering dangerous driving habits.

4. **Data-driven Policy Making:** Evidence-based policy decisions targeted at enhancing road safety infrastructure and regulations can be influenced by the data produced by HelmetDetect, including helmet compliance rates and detection statistics. Policymakers can more efficiently allocate resources to address road safety challenges and identify areas for targeted interventions by analyzing trends and patterns in helmet usage. In order to make roads safer for everyone, this data-driven strategy encourages cooperation between governmental organizations, advocacy groups, and the general public.
5. **Personal Safety Awareness:** HelmetDetect is a tool for road users' personal safety awareness in addition to being an enforcement and policy measure. The system serves as a reminder to riders and motorcyclists of the significance of wearing protective gear in order to prevent severe head injuries in the event of an accident. This is achieved by visualizing the consequences of not wearing a helmet through detection results. A stronger focus on personal safety when driving and behavioral changes may result from this awareness at the individual level.

To sum up, HelmetDetect's practical implications go beyond its detection capabilities. They also include broader implications for traffic management, public awareness campaigns, data-driven policy making, road safety enforcement, and personal safety awareness. The system helps to create safer road environments and promotes a culture of responsible behavior among road users by utilizing computer vision technology.

5 Challenges Faced

We faced a number of obstacles during the HelmetDetect system's development that needed creative thinking and tenacity to get past. The main obstacles encountered during the project lifecycle are highlighted in this section, along with the solutions used.

5.1 Data Collection Challenges

Obtaining a representative and varied dataset to train the helmet detection model was one of the main obstacles we faced. Gathering pictures of people in different settings, with different lighting and camera angles, turned out to be a difficult undertaking. Furthermore, it was difficult to guarantee that a fair representation of people with and without helmets was provided because different demographics and geographical areas had varying helmet usage patterns.

5.2 Model Training Issues

The YOLOv5 model's training on the annotated dataset came with its own set of difficulties. Experimentation and iteration were necessary to fine-tune the model parameters, optimize hyperparameters, and address issues like underfitting and overfitting. Furthermore, careful monitoring and training strategy adjustment were required to maintain convergence and stability during training while preventing model degradation or divergence.

5.3 Technical Hurdles

Significant challenges were also presented by technical obstacles pertaining to software compatibility, hardware limitations, and performance bottlenecks. At first, our devices had trouble keeping up with the computational demands of training the model, which led to resource limitations and slow training times. Compatibility problems between library versions and software dependencies also caused delays in development and debugging difficulties.

5.4 Overcoming Challenges

To overcome these challenges, we adopted several strategies and solutions:

1. **Data Augmentation:** We used data augmentation methods including rotation, scaling, flipping, and brightness adjustments to overcome difficulties with data collection and enhance the dataset. This enhanced the training data's robustness and diversity, strengthening the model's capacity for generalization.

2. **Transfer Learning:** Using pretrained models as a foundation, we used transfer learning to minimize problems with model training and maximize training effectiveness. We were able to minimize the amount of time we needed to do extensive training from scratch by using this method to initialize the model with prelearned features and refine it on our unique dataset.

3. **Remote Desktops (RDPs):** We used remote desktops (RDPs) with more computational power and resources to get around performance problems with local devices. We were able to increase productivity and iterations more quickly as a result of being able to accelerate model training and inference.
4. **Community Support:** In order to solve technical problems and find advice on best practices, it was very helpful to use online forums, documentation, and community support channels. By actively participating in developer communities, we were able to accelerate our progress and problem-solving efforts by drawing on collective expertise and learning from shared experiences.

Through a combination of transfer learning, remote desktop utilization, data augmentation techniques, and community support, we were able to overcome the obstacles that arose during the HelmetDetect system's development. These experiences improved our technical proficiency while highlighting the value of flexibility, perseverance, and teamwork in overcoming challenges in challenging projects.

6 Lessons Learned

As we looked back on the HelmetDetect project's development, we gained insightful knowledge, picked up new technical expertise, and discovered crucial project management lessons. The main lessons we learned from the experience are summarized in this section, along with some possible future paths for the project.

6.1 Insights Gained from the Project

1. **Importance of Data Quality:** Machine learning model performance is greatly impacted by the caliber and diversity of the training data. We discovered that in order to guarantee reliable model training and precise detection outcomes, it is crucial to conduct extensive data collection, annotation, and preprocessing.
2. **Practical Applications of Computer Vision:** Examining computer vision technology's real-world uses, like helmet detection for traffic safety, offered insightful information about the potential benefits of AI-powered solutions for resolving social issues and enhancing public welfare.
3. **Interdisciplinary Collaboration:** Working with colleagues who have different backgrounds and specialties made me realize how important interdisciplinary collaboration is when taking on challenging projects. By integrating domain expertise with technical know-how, we were able to create more thorough and efficient solutions.
4. **Iterative Development Process:** The adoption of an iterative development process, which is marked by ongoing experimentation, feedback loops, and incremental improvements, was crucial for managing uncertainties and iteratively improving the system in response to user feedback and performance assessments.

6.2 Technical Skills Acquired

1. **Deep Learning:** We obtained practical experience with deep learning techniques, such as model architecture design, training, optimization, and evaluation, by implementing the YOLOv5 object detection model.
2. **Data Preprocessing:** We now have a better understanding of data preprocessing pipelines and how they contribute to increased model robustness and generalization thanks to preprocessing techniques like data augmentation, normalization, and resizing.

3. **Web development:** By creating the HelmetDetect system's Flask-based user interface, we were able to hone our skills in HTML, CSS, JavaScript, and the Flask framework, which helped us create dynamic and user-friendly AI application interfaces.
4. **Model Deployment and Optimization:** By utilizing remote desktops to overcome performance constraints and by optimizing the memory footprint and speed of model inference, we were able to gain hands-on experience in deploying and fine-tuning machine learning models for practical applications.

6.3 Project Management Lessons

1. **Effective Communication:** Team members that communicated well and on time were able to collaborate, assign tasks, and solve problems together, which improved the cohesiveness and output of the project.
2. **Agile Methodology:** By implementing agile concepts like adaptive planning, continuous integration, and iterative development, we were able to adjust our approach to changing needs and rank tasks according to their practicality and worth.
3. **Risk management:** By recognizing possible risks early in the project lifecycle, such as data availability, model convergence problems, and resource limitations, we were able to proactively reduce risks and create backup plans to reduce interruptions.

6.4 Future Directions

1. **Performance Optimization:** The HelmetDetect system's speed, efficiency, and scalability can be improved through additional optimization of the model architecture, training methods, and deployment infrastructure. This will allow for real-time deployment in environments with limited resources.
2. **Multimodal Integration:** Combining computer vision algorithms with other sensor modalities, like LiDAR or depth cameras, can improve the system's detection capabilities and supply supplementary data for stronger and more dependable detections.
3. **User Feedback Incorporation:** To ensure that the system's usability, functionality, and performance are in line with user needs and preferences, feedback from stakeholders, end users, and domain experts can guide iterative improvements.
4. **Wider Safety Applications:** The HelmetDetect system's usefulness in improving general road safety and transportation infrastructure can be increased by extending its

scope to include other safety-critical applications like pedestrian detection, traffic sign recognition, and vehicle tracking.

To sum up, the HelmetDetect project was a great learning opportunity that gave us important knowledge, practical abilities, and project management lessons that we can use in our future research, development, and implementation of artificial intelligence. As long as we embrace innovation and never stop learning, we will be able to use technology to solve societal problems and have a positive impact.

7 Conclusion

To sum up, the HelmetDetect project is a noteworthy attempt to use computer vision technology to improve road safety by means of helmet detection. This section offers suggestions for further research as well as a summary of the project's major contributions, accomplishments, and conclusions.

We successfully created and deployed the HelmetDetect system during the project, which makes use of the YOLOv5 object detection model to recognize people without helmets in pictures or video streams. Important conclusions from the study include:

1. With a precision score of 97.3%, the HelmetDetect system demonstrates a high level of accuracy in identifying individuals who do not wear helmets.
2. Despite having a somewhat lower recall rate (66.3%), the system performs admirably in terms of detecting helmet non-compliance.
3. Across various confidence thresholds, the system's accuracy in localizing and classifying individuals wearing helmets is demonstrated by its mean average precision (mAP) score of 66.4%.

We are dedicated to expanding the capabilities of the HelmetDetect system and helping to achieve the main objective of making road environments safer for everyone by embracing innovation and continuous improvement.

8 References

1. flask.palletsprojects.com. (n.d.). *Deploying to Production — Flask Documentation (3.0.x)*. [online] Available at: https://flask.palletsprojects.com/en/3.0.x/deploying/.
2. Jocher, G. (2020). *ultralytics/yolov5*. [online] GitHub. Available at: https://github.com/ultralytics/yolov5.
3. Paperspace Blog. (2021). *How to Train YOLO v5 on a Custom Dataset*. [online] Available at: https://blog.paperspace.com/train-yolov5-custom-data/.
4. www.pytorch.org. (n.d.). *PyTorch*. [online] Available at: https://pytorch.org/hub/ultralytics_yolov5/.

9 Appendix

- GitHub Repository: https://github.com/supunsathsara/HelmetDetect
- Web Demo Source: https://github.com/supunsathsara/HelmetDetect/tree/web
- Video Demo: https://www.youtube.com/watch?v=xVo9erXTSBQ
- Google Colab Notebook: https://colab.research.google.com/drive/1y0g8Sfyx02qi_59ENU4i4rX1dXnu6sF5?usp=sharing